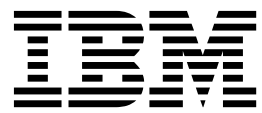


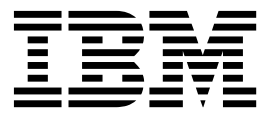
IBMWorkload Scheduler
Version 9 Edition 4

Guide d'utilisation et de référence



IBMWorkload Scheduler
Version 9 Edition 4

Guide d'utilisation et de référence



Important

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations figurant à la section «Remarques», à la page 1027.

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- <http://www.fr.ibm.com> (serveur IBM en France)
- <http://www.ibm.com/ca/fr> (serveur IBM au Canada)
- <http://www.ibm.com> (serveur IBM aux Etats-Unis)

*Compagnie IBM France
Direction Qualité
17, avenue de l'Europe
92275 Bois-Colombes Cedex*

© Copyright IBM France 2017. Tous droits réservés.

La présente édition s'applique à la version 9.4.0 d'IBM Workload Scheduler (référence de logiciel 5698-WSH), ainsi qu'à toutes les éditions et modifications ultérieures, sauf mention contraire dans les nouvelles éditions.

© Copyright IBM Corporation 1999, 2016. © Copyright HCL Technologies Limited 2016, 2017

Table des matières

Figures	ix

Tableaux	xi

Avis aux lecteurs canadiens	xv

A propos de cette publication	xvii

Nouveautés	xvii
Nouveautés de cette publication	xvii
A qui s'adresse cette publication	xvii
Accessibilité	xvii
Formation technique	xviii
Informations sur le support	xviii
Conventions du présent document	xviii
Conventions typographiques	xviii
Variables et chemins spécifiques au système d'exploitation	xix
Syntaxe des commandes	xix

Chapitre 1. Présentation du produit IBM Workload Scheduler	1

Explication des concepts de base	1
Objets de base de données IBM Workload Scheduler	1
Réseau IBM Workload Scheduler	20
Configuration de l'environnement d'exécution IBM Workload Scheduler	21
Définition d'activités de planification à l'aide de IBM Workload Scheduler	22
Contrôle du traitement des travaux et des flots de travaux	23
Gestion des activités de planification de production avec IBM Workload Scheduler	28
Automatisation de la charge de travail grâce aux règles d'événement	29
Interfaces utilisateur IBM Workload Scheduler	30
Démarrage de la production	33

Chapitre 2. Description des processus et des commandes de base	37

Exécution de commandes sous Windows	37
Processus de poste de travail IBM Workload Scheduler	37
Démarrage et arrêt de processus sur un poste de travail	43
Démarrage et arrêt de la agent	45
Communication interprocessus du poste de travail	45
Communication réseau de IBM Workload Scheduler	48
Prise en charge d'IPv6 (Internet Protocol version 6)	51

Chapitre 3. Configuration de l'environnement de travail	53

Présentation de l'environnement de travail	53
Variables d'environnement exportées par jobman	54
Personnalisation du format des dates dans stdlist	56
Personnalisation du traitement des travaux sur un poste de travail UNIX - jobmanrc	57
Personnalisation de la section MAIL_ON_ABEND de jobmanrc	59
Personnalisation du traitement des travaux pour un utilisateur sous UNIX - jobmanrc	60
Personnalisation du traitement des travaux sur un poste de travail Windows - jobmanrc.cmd	61
Personnalisation de la section MAIL_ON_ABEND de jobmanrc.cmd	62
Personnalisation du traitement des travaux sur un poste de travail Windows - djobmanrc.cmd	63
Configuration des options pour l'utilisation des interfaces utilisateur	65

Chapitre 4. Gestion du cycle de production	67

Concepts de base de la gestion de plan	67
Plan de préproduction	69
Identification des instances de flot de travaux dans le plan	70
Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux	71
Plan de production	82
Options de report	82
Plan d'essai	84
Plan prévisionnel	85
Personnalisation de la gestion du plan à l'aide des options globales	86
Création et extension d'un plan de production	91
JnextPlan	92
Ligne de commande Planman	94
Création d'un plan de production intermédiaire	95
Création d'un plan intermédiaire pour une extension de plan	97
Extraction des informations du plan de production	98
Création d'un plan d'essai	99
Création d'un plan d'essai avec une extension de plan de production	100
Création d'un plan prévisionnel	101
Déploiement de règles	102
Déverrouillage du plan de production	103
Réinitialisation du plan de production	103
Suppression du plan de préproduction	104
Réplication des données de plan dans la base de données	105
Surveillance de la réplication des données de plan dans la base de données	106

Commande stageman	107
Gestion des accès simultanés au fichier Symphony	109
Scénario 1 : Accès au fichier Symphony	
verrouillé par d'autres processus IBM Workload	
Scheduler.	109
Scénario 2 : Accès au fichier Symphony	
verrouillé par stageman	109
Gestion des dépendances de prédécesseur/	
successeur avec l'invite de report	109
Commande logman	110
Durée estimée d'un travail et facteur de fiabilité	/
associé.	112
Démarrage du traitement du plan de production	113
Automatisation du traitement du plan de	
production	114

Chapitre 5. Assurance de service de charge de travail 117

Activation et configuration de l'assurance de	
service de charge de travail.	118
Planification de travaux critiques.	122
Traitement et contrôle des travaux critiques	123
Scénario d'assurance de service de charge de	
travail.	126

Chapitre 6. Personnalisation de votre charge de travail à l'aide des tables de variables 129

Migration des paramètres globaux à partir de	
versions précédentes	130
Table de variables par défaut	131
Intégrité des données des tables de variables	131
Mécanisme de verrouillage des tables de variables	131
Sécurité de la table de variables	132
Résolution de variable	133

= Chapitre 7. Automatisation	
= conditionnelle de la charge de travail . 135	
= Scénario métier.	138

Chapitre 8. Exécution de l'automatisation de la charge de travail gérée par événements 141

Processus de gestion des règles d'événements	144
Utilisation des interfaces et commandes	
impliquées	147
Définition de règles d'événement.	150
Exemples de règles d'événement	152
Remarques sur le fonctionnement des règles	159
Eléments de règles déclenchés.	161
Définition d'événements personnalisés	161

Chapitre 9. Définition d'objets dans la base de données. 163

Définition des objets de planification	163
Définition de poste de travail	166
Définition de classe de postes de travail	184
Définition de domaine	185

Définition de travail	187
Définition d'utilisateur	200
Définition d'agenda	205
Définition des variables et des paramètres.	206
Définition de table de variables	211
Définition d'invite	213
Définition de ressource	215
Définition d'un groupe de cycle d'exécution	216
Définition de flot de travaux	226
Détail des mots clés pour la définition des flots	
de travaux	233
table de variables	291
Définition de règle d'événement	291
Définition des applications de charge de travail	304
Définition d'objet de sécuroté	306

Chapitre 10. Gestion des objets dans la base de données - composer . . . 321

Configuration du programme de ligne de	
commande composer.	321
Configuration de l'environnement composer	322
Exécution du programme Composer.	324
Exécution de commandes depuis composer	327
Filtres et caractères génériques	327
Délimiteurs et caractères spéciaux	330
Codes retour de composer	331
Commandes Composer	331
Vérification de l'intégrité référentielle	333
add.	338
authenticate	340
continue	340
delete	341
display	345
edit.	351
exit.	352
extract.	352
help	357
list	358
lock	365
modify	370
new	375
print	377
redo	377
rename	379
replace	381
Commande système	383
unlock.	383
update	388
validate	390
version	391

Chapitre 11. Gestion des applications de charge de travail 393

Création d'un modèle d'application de charge de	
travail.	394
Exportation d'une définition de flot de travaux	
comme modèle d'application de charge de travail	397
Importation d'un modèle d'application de charge	
de travail.	398
Résolution du fichier de mappage	398

=	Utilisation d'expressions régulières pour modifier	
=	le fichier de mappage	404
	Déploiement d'un application de charge de travail	405
	wappman	406

Chapitre 12. Gestion des objets dans le plan - conman 411

	Configuration du programme de ligne de commande conman	411
	Configuration de l'environnement Conman	412
	Exécution de conman.	413
	Exécution de commandes depuis Conman.	416
	Caractères génériques	417
	Délimiteurs et caractères spéciaux	417
	Traitement des commandes Conman	418
	Sélection de travaux dans les commandes	419
	Syntaxe	419
	Arguments	419
	Sélection de flots de travaux dans les commandes	428
	Syntaxe	428
	Arguments	428
	Gestion des travaux et des flots de travaux des agents de niveau antérieur	435
	Codes retour de Conman	435
	Commandes Conman.	436
	adddep job	439
	adddep sched	442
	altpass.	444
	altpri	445
	bulk_discovery	446
	cancel job.	446
	cancel sched.	448
	checkhealthstatus	450
	confirm	450
	console	453
	continue	454
	deldep job	455
	deldep sched	457
	deployconf	458
	display	459
	exit.	461
	fence	462
	help	463
	kill	464
	limit cpu	465
	limit sched	467
	link.	468
	listsym	470
=	Listsucc	472
	recall	474
	redo	475
	release job	476
	release sched	478
	reply	480
	rerun	481
=	Rerunsucc	484
	resetFTA	487
	resource	488
	setsym	489
	showcpus	490
	showdomain	497

	showfiles.	498
	showjobs.	501
	showprompts	520
	showresources	523
	showschedules	525
	shutdown	531
	start	532
	startappserver	534
	startbrokerapp	535
	starteventprocessor	535
	startmon	536
	status	537
	stop	538
	stop ,progressive	540
	stopappserver	541
	stopbrokerapp	542
	stopeventprocessor	543
	stopmon	544
	submit docommand	545
	submit file	549
	submit job	552
	submit sched	556
	switcheventprocessor.	561
	switchmgr	562
	Commande système	563
	tellop	564
	unlink.	565
	version	567

Chapitre 13. Utilisation de statistiques avancées pour prévoir la durée estimée d'un travail. 569

	Installation du sous-ensemble SPSS Statistics	569
	Sélection des travaux devant être mesurés par l'outil de statistiques avancées.	571
	Importation et configuration du flot de travaux ELAB_JOB_STAT_JS	571
	Exécution du flot de travaux ELAB_JOB_STAT_JS et affichage des résultats.	573
	Dépannage du flot de travaux ELAB_JOB_STAT_JS	573

Chapitre 14. Extension des fonctions IBM Workload Scheduler 575

	Procédure préalable de création de types de travaux avec options avancées.	580
	Création de définitions de travaux avancés	581
	Définition de travail - Travaux z/OS	582
	Travaux de type commande distante	582
	Travaux IBM i	585
	Travaux exécutables	588
	Travaux de la méthode d'accès	590
	Procédure préalable de création de travaux TPM (Provisioning)	592
	Travaux IBM SmartCloud Provisioning.	592
	Travaux reflet	596
	Travaux de transfert de fichiers	598
	Procédure préalable de création de travaux OSLC	
	Automation et OSLC Provisioning	606
	Définition de travail - OSLC Automation	608

	Définition de travail - Application des accès	
	OSLC	610
	Travaux de base de données	612
	Configuration requise pour exécuter des travaux	
	de branche	617
	travaux de services Web	618
	travaux de type services Web RESTful	620
	travaux Java	624
	Travaux J2EE	625
	JSR 352 Java Batch.	629
	MQTT	631
	Travaux de Table de variables	633
=	Travaux Gestion des travaux	636
=	Travaux Soumission d'un flot de travaux	641
	Codes retour	644
	Installation automatique de plug-in en exécutant	
	des types de travaux avec options avancées sur des	
	agents dynamiques	645
	Définition des variables et mots de passe pour la	
	résolution locale sur des agents dynamiques	645
	Indication de variables et de mots de passe	
	locaux dans les définitions de travail	646
	Définition de variables dans les travaux	
	Dynamic Workload Broker	648
	Transmission de variables entre travaux	650
	Transmission des propriétés de travail d'un	
	travail à l'autre dans la même instance de flot	
	de travaux	651
	Transmission de la sortie standard de travail	
	d'un travail à l'autre dans la même instance de	
	flot de travaux	661
	Transmission de la sortie standard de travail	
	d'un travail à l'autre en tant qu'entrée standard	
	dans la même instance de flot de travaux	662
	Transmission d'un jeu de variables d'un travail à	
	un autre dans la même instance de flot de	
	travaux à l'aide de l'utilitaire jobprop	663
	Transmission de variables d'un travail à un	
	autre, dans un même flot de travaux ou dans un	
	flot de travaux distinct, via les tables de	
	variables	665
	Exécution d'un script à l'issue d'un travail.	665

Chapitre 15. Gestion des fonctions de planification dynamique dans votre environnement. 667

Scénario métier sur fonction dynamique	669
Scénario : création d'une définition de travail et soumission à un pool dynamique.	670
Scénario : création d'une définition de travail et soumission à un pool.	671
Définition de relations d'affinité	671
Définition des dépendances de fichier dans la planification dynamique.	672
Promotion des travaux planifiés sur des pools dynamiques	674
Ajout de fonctions dynamiques aux travaux IBM Workload Scheduler existants	674
Limitations de la planification dynamique.	675

Limitations spécifiques aux travaux appartenant au flot de travaux USERJOBS lors de la planification dynamique	676

Chapitre 16. Utilisation des commandes d'utilitaire 677

Description des commandes	677
at et batch	679
cpuinfo	682
datecalc	685
datamigrate	689
delete	691
evtdef	692
evtsz	696
Filemonitor	698
jobinfo	701
jobstdl	703
maestro	706
makecal	706
metronome	708
morestdl	708
parms	710
release	713
rmstdlist	717
sendevent	719
showexec	720
shutdown	721
ShutDownLwa - Arrêt de l'agent	722
StartUp	722
StartUpLwa - Démarrage de l'agent	723
twinst_pull_info	723
version	723
Commandes non prises en charge	725

Chapitre 17. Utilisation des commandes d'utilitaire dans l'environnement dynamique 727

Fichier de configuration de ligne de commande	728
exportserverdata	732
importserverdata	733
jobprop	735
movehistorydata	736
param	738
resource	741
Utilisation de la commande resource à partir d'un agent	749
sendevent	751
twstrace	753

Chapitre 18. Extraction des états et des statistiques 755

Configuration de l'utilisation des commandes de génération d'états	755
Modification du format de date	756
Description des commandes	756
rep1 à rep4b.	757
rep7	758
rep8	759
rep11	761

reptr	762
xref.	763
Exemples de sorties d'état	764
Etat 01 - Liste des caractéristiques des travaux :	764
Etat 02 - Liste des invites :	767
Etat 03 - Liste des agendas :	767
Etat 04A - Liste des paramètres :	768
Etat 04B - Liste des ressources :	768
Etat 07 - Liste des historiques des travaux :	768
Etat 08 - Histogramme des travaux :	769
Etat 9B - Détail de la production planifiée :	769
Etat 10B - Détail de la production réelle :	770
Etat 11 - Calendrier de production planifiée :	771
Etat 12 - Etat de références croisées :	772
Programmes d'extraction d'états	774
jbxtract	775
prxtract	776
caxtract	777
paxtract	778
rextact	779
r11xtr	779
xrxtct	781
Exécution de rapports Dynamic Workload Console et de rapports de traitement par lots	785
Rapports historiques	788
Rapports de production	791
Exécution des rapports de traitement par lots à partir de l'interface de ligne de commande	792

Chapitre 19. Gestion des fuseaux horaires. 799

Activation de la gestion des fuseaux horaires.	799
Comment IBM Workload Scheduler gère les fuseaux horaires	800
Passage à l'heure d'été	802
Désactivation de l'heure d'été	802
Règles générales	803

Chapitre 20. Définitions des méthodes d'accès des agents 805

Interface de méthode d'accès	806
Syntaxe de la ligne de commande des méthodes	806
Messages de réponse des méthodes	808
Fichier d'options de méthode	809
Exécution des méthodes	812
Tâche de lancement d'un travail (LJ).	812
Tâche de gestion du travail (MJ)	813
Tâche de vérification d'un fichier (CF) - agents étendus uniquement	813
Tâche d'obtention d'un statut (GS) - agents étendus uniquement	814
Commande Cpuinfo pour agents étendus uniquement	815
Identification des incidents	815
Messages d'erreur de la liste standard d'un travail	815
Méthode non exécutable.	815
Messages du gestionnaire de console pour agents étendus uniquement	815

Messages des programmes Composer et Compiler pour agents étendus uniquement	816
Messages de Jobman pour agents étendus uniquement	816

Chapitre 21. Gestion des dépendances interréseaux. 817

Présentation sur les dépendances interréseaux	817
Affichage d'une dépendance interréseau	818
Configuration d'un agent réseau	819
Exemple de définition d'un agent réseau	820
Définition d'une dépendance interréseau	821
Gestion des dépendances interréseau dans le plan	822
Etats des travaux définis dans le flot de travaux EXTERNAL	822
Gestion des travaux définis dans le flot de travaux EXTERNAL	823
Exemples de scénarios de gestion des dépendances interréseaux	823
Dépendances interréseaux dans un environnement mixte	825

Chapitre 22. Application d'une logique de branchement conditionnelle. 827

Configuration de dépendances conditionnelles	829
Jointure ou combinaison de dépendances	831
Planification et soumission de dépendances conditionnelles	832
Evaluation et traitement d'un flux de dépendances conditionnelles	833
Surveillance de dépendances conditionnelles	840
Gestion du plan de dépendances conditionnelles	842

Chapitre 23. Définition et gestion des dépendances croisées 845

Présentation des dépendances croisées	845
Flux de processus dans l'environnement de planification distribué	847
Définition d'une dépendance croisée.	850
Surveillance d'une résolution de dépendance croisée dans le plan de production	851
Modification de l'état du travail reflet après que la liaison est établie	852
Modification de l'état du travail reflet après que la liaison est établie	857
Affichage des raisons de l'état d'échec du travail reflet (FAIL)	859
Etat d'un travail reflet lors de la récupération ou la réexécution d'un travail distant	859
Application du report aux dépendances croisées	859
Gestion des travaux reflet dans le plan de production	859

Chapitre 24. Gestion d'un environnement dynamique IBM i 861

Définition des agents sur les systèmes IBM i	861
Définition des travaux sur les systèmes IBM i	861
Gestion des agents sur les systèmes IBM i	862

Démarrage et arrêt des agents sur les systèmes IBM i	862
Utilisation des commandes d'utilitaire pour les agents sur les systèmes IBM i	863
Planification des travaux sur les systèmes IBM i	863
L'agent journal des travaux et la variable d'environnement TWSASPOOLS	864
Surveillance des travaux enfants sur les agents IBM i	865
Récupération du code retour agent	868
Contrôle de l'environnement de travail à l'aide du code retour utilisateur	869
Méthode alternative de définition du code retour utilisateur	870

Annexe A. Automatisation de la charge de travail commandée par les événements, définitions des événements et des actions 873

Fournisseurs d'événements et définitions	873
Evénements TWSObjectsMonitor	873
Evénements FileMonitor	876
Evénements TWSApplicationMonitor	884
Fournisseurs d'actions et définitions	885
Actions GenericAction	886
Actions MailSender	887
Actions MessageLogger	887
Actions SmartCloud Control Desk	887
Actions ServiceNow	887
Actions TBSMEventForwarder	888
Actions TECEventForwarder	888
Actions TWSAction	889
TWSForZosAction	889

Annexe B. Référence de schéma Job Submission Description Language . . . 893

Eléments JSDL	899
Ressources de la définition de travail	934

Annexe C. Aide-mémoire pour les commandes 937

Gestion du plan	937

Gestion des objets dans la base de données	938
Commandes universelles	939
Objets de planification	939
Commandes composer	945
Gestion des objets dans le plan	948
Commandes conman	949
Commandes d'utilitaire	954
Commandes de génération d'états	957

Annexe D. Définition et gestion des travaux de branche génériques. . . . 961

Introduction	961
Terminologie	962
Fonctions de travail de branche	965
Avantages du travail de branche	966
Exemples de scénarios	967
Scénarios basés sur le type de condition	967
Scénarios basés sur le type d'action	996
Scénario d'action de signal	1003
Utilisation du travail de branche	1006
Configuration requise pour exécuter des travaux de branche	1006
Définition du travail de branche et du travail de signal dans la base de données	1007
Placement du travail de branche dans le flot de travaux	1009
Utilisation du travail ABEND	1010
Indication des paramètres de travail de branche	1010
Référence de paramètres	1011
Respect de la casse	1016
Exemples de statut d'exemple	1016
Remarques importantes sur le travail de branche	1022

Annexe E. Accessibilité 1025

Remarques 1027

Marques	1029
Dispositions pour la documentation du produit	1030

Index 1031

Figures

1.	Réseau à domaine unique	17
2.	Réseau à domaines multiples	18
3.	Arborescence des processus sous UNIX	41
4.	Arborescence des processus sous Windows	42
5.	Communication inter-processus sur le gestionnaire de domaine maître	47
6.	Communication inter-processus sur le gestionnaire de domaine maître et l'agent tolérant aux pannes	48
7.	Instances devant s'exécuter le même jour utilisées comme critère de correspondance	71
8.	Instance précédente la plus proche utilisée comme critère de correspondance	72
9.	Instances définies dans un intervalle relatif utilisées comme critère de correspondance	73
10.	Instances définies dans un intervalle absolu utilisées comme critère de correspondance	73
11.	Travail remplacé le plus proche	74
12.	Instance de prédécesseur suspendu	75
13.	Critères de correspondance du même jour - Etape 1 : Au début de la journée le jeudi	76
14.	Critères de correspondance du même jour - Etape 2 : à 9:00	77
15.	Critères de correspondance du même jour - Etape 3 : à 15:00	77
16.	Critères de correspondance précédents les plus proches - Etape 1 : avant 08:00	78
17.	Critères de correspondance précédents les plus proches - Etape 2 : à 8h les jours de la semaine excepté le jeudi et le vendredi	78
18.	Critères de correspondance précédents les plus proches - Etape 3: à 09:00 le jeudi et le vendredi	78
19.	Critères de correspondance précédents les plus proches - Etape 4 : à 15:00 tous les jours	79
20.	Critères de correspondance d'intervalle relatif - au début de la journée le jeudi	80
21.	Critères de correspondance d'intervalle absolu - au début de la journée le jeudi	82
22.	Chemin critique.	125
= 23.	Automatisation conditionnelle de la charge de travail	137
= 24.	Définition d'utilisateur	204
25.	Liaisons réseau	469
26.	Exemple de réseau.	533
27.	Exemple de réseau.	539
28.	Exemple de réseau.	541
29.	Postes de travail réseau non liés	566
30.	Exemple où la conversion de début de journée n'est pas appliquée	801
31.	Exemple où la conversion de début de journée est appliquée	801
32.	Réseaux local et distant	820
33.	Dépendance de prédécesseur/successeur sur le travail ABSENCES	834
34.	Deux dépendances conditionnelles différentes aux états SUCC et ABEND sur le travail ABSENCES	834
35.	Dépendances conditionnelles sur les conditions de sortie du travail ABSENCES	835
36.	Dépendance de jointure contenant trois dépendances à l'état SUCC	836
37.	Dépendance conditionnelle d'état sur un travail avec le paramètre recovery stop	837
38.	Dépendance conditionnelle d'état ABEND	838
39.	Condition de sortie STATUS_OK	838
40.	Condition de sortie ERROR	839
41.	Condition de sortie STATUS_OK	839
42.	Logique de dépendance croisée	847
43.	Transition de l'état d'un travail reflet jusqu'à ce que la liaison soit établie	852
44.	Instance à lier si l'heure planifiée du travail reflet est incluse dans l'intervalle du plan en cours	855
45.	Instance à lier si l'instance qui précède de la façon la plus proche l'heure planifiée du travail reflet existe dans le plan à long terme mais a été annulée du plan en cours.. . . .	856
46.	L'heure planifiée du travail reflet est incluse dans le plan en cours mais aucune instance à lier n'existe	856
47.	L'instance à lier existe mais n'est pas encore incluse dans le plan en cours	857
48.	L'intervalle du plan à long terme ne contient toujours pas l'heure planifiée du travail reflet.	857
49.	Chaîne de transition d'état de travail reflet après que la liaison a été établie	858
50.	Objectif du travail de branche	962
51.	Termes associés à la définition de flot de travaux	964
52.	Termes liés à l'exécution de flot de travaux (instance de flot de travaux concrète)	965
53.	Définition de branche simple (SUCC)	968
54.	Statut final de branche simple (SUCC)	968
55.	Définition de branche simple (ABEND)	969
56.	Définition de branche longue (SUCC)	971
57.	Statut final de branche longue (ABEND)	973
58.	Travaux de branches multiples dans un même flot de travaux	975
59.	Définition de fin anormale de parent (SUCC)	977
60.	Scénario de modèle - définition	980
61.	Définition du scénario de modèle annulé	982
62.	Modèle dans la définition de ligne de modèle	985
63.	Modèle dans la définition annulée de ligne de modèle.	987
64.	Définition de branche de comparaison numérique	990
65.	Définition de condition complexe	993
66.	Définition d'actions de pause et de libération	998
67.	Définition de scénario d'actions de pause et de libération multiples	1002

68.	Définition d'action de signal	1004

Tableaux

1. Syntaxe des commandes	xix	31. Options et actions de reprise	196
2. Scénario 1. Aucune restriction de temps dans le groupe de cycle d'exécution	8	32. Traitement d'une barre oblique inversée dans la substitution de variable	208
3. Scénario 2. Restriction de temps dans le groupe de cycle d'exécution sans décalage	9	33. Mots clés auxquels peuvent être associés des paramètres locaux dans les commandes submit	209
4. Scénario 3. Restriction de temps dans le groupe de cycle d'exécution avec décalage (+1 12h00)	10	34. Mot clé d'accès nécessaire de la table de variables dans le fichier de sécurité (objet variable) et actions autorisées sur ses entrées de variable.	213
5. Astuce	19	35. Liste des mots clés de planification	228
6. Démarrage et arrêt de IBM Workload Scheduler sur un poste de travail	43	36. Explication de la notation définissant le nombre d'occurrences pour un élément de langage.	292
7. Démarrage et arrêt de la agent	45	37. Événements TWSObjectsMonitor.	295
8. Variables d'environnement de travail sous Windows	54	38. Événements TWSApplicationMonitor.	297
9. Variables d'environnement de travail pour UNIX	55	39. Événements FileMonitor.	297
10. Variables définies par défaut dans le fichier jobmanrc	57	40. Types d'action par fournisseur d'actions.	299
11. Variables définies par défaut dans le fichier jobmanrc.cmd	62	41. Types d'objet de sécurité	310
12. Paramètres des options globales de report	83	42. Actions que les utilisateurs ou les groupes peuvent exécuter sur les différents objets	311
13. Paramètres de report résultants	83	43. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la conception et de la surveillance de la charge de travail	312
14. Options globales de l'assurance de service de charge de travail	118	44. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la modification du plan courant	313
15. Options locales de l'assurance de service de charge de travail	121	45. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la soumission de la charge de travail	313
16. Relation entre les tables de variables et leurs variables correspondantes dans le fichier de sécurité IBM Workload Scheduler	132	46. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la gestion de l'environnement de charge de travail.	314
17. Commandes conman pour gérer les moteurs de surveillance	145	47. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la gestion des règles d'événement	315
18. Commandes conman pour gérer le serveur de traitement d'événements	146	48. Tâches administratives que les utilisateurs ou les groupes peuvent réaliser	315
19. Interfaces et commandes pour gérer l'automatisation de charge de travail gérée par événements	148	49. Actions que les utilisateurs ou les groupes peuvent réaliser sur les rapports de charge de travail	316
20. Liste des mots clés d'objet de planification pris en charge	164	50. Actions réalisables par les utilisateurs ou les groupes sur Application Lab	316
21. Liste des mots clés d'objet de sécurité pris en charge	164	51. Attributs des types d'objet de sécurité	316
22. Liste des mots réservés lors de la définition de travaux et de flots de travaux	165	52. Critères de filtrage des objets de planification	328
23. Liste des mots réservés lors de la définition de postes de travail	165	53. Délimiteurs et caractères spéciaux pour le programme composer.	330
24. Liste des mots réservés lors de la définition d'utilisateurs	166	54. Liste des commandes composer	332
25. Paramètres des attributs pour la gestion des types de poste de travail	167	55. Identificateurs d'objet pour chaque type d'objet défini dans la base de données	333
26. Paramètres des attributs pour les types de poste de travail cible	168	56. Mise à jour de la définition d'objet lors de la suppression de l'objet référencé	334
27. Type de communication en fonction de la valeur securitylevel	179	57. Vérification de l'intégrité référentielle lors de la suppression d'un objet de la base de données	335
28. Exemples : changement de nom de la définition de travail	189		
29. Opérateurs de comparaison.	192		
30. Opérateurs logiques	193		

58. Formats de sortie de l'affichage des objets de planification	349		90. Attributs obligatoires et facultatifs pour la définition d'un travail de type Table de variables	633
59. Formats de sortie de l'affichage des objets de planification	362			
60. Objets extraits au cours du processus d'exportation.	400	=	91. Attributs obligatoires et facultatifs pour la définition d'un travail de type Gestion des travaux	637
61. Résolution du fichier de mappage	403	=	92. Attributs obligatoires et facultatifs pour la définition d'un travail de type Soumission d'un flot de travaux	641
62. Délimiteurs et caractères spéciaux pour la commande conman	417	=	93. Variables IBM Workload Scheduler prises en charge dans les définitions JSDL.	649
63. Liste des commandes Conman.	436		94. Propriétés des travaux IBM InfoSphere DataStage.	654
64. Changement d'état après exécution de la commande confirm	451		95. Propriétés des travaux reflet	655
65. Liaisons ouvertes	470		96. Propriétés des travaux OSLC	655
= 66. Critères de récupération des options de reprise	482		97. Propriétés des travaux IBM WebSphere MQ	655
= 67. Statut des successeurs.	485		98. Propriétés des travaux IBM Sterling Connect:Direct	655
68. Postes de travail démarrés	534		99. Propriétés des travaux Salesforce	656
69. Postes de travail arrêtés	539		100. Propriétés des travaux SAP BusinessObjects BI	656
70. Postes de travail arrêtés par la commande stop ,progressive	541		101. Propriétés des travaux Oracle E-Business Suite	656
71. Postes de travail non liés.	566		102. Propriétés des travaux de transfert de fichier	657
72. Types de travail avec options avancées	576		103. Propriétés des travaux Hadoop Map Reduce	657
73. Attributs obligatoires et facultatifs pour la définition d'un travail z/OS.	582		104. Propriétés des travaux système de fichiers distribué Hadoop	658
74. Attributs obligatoires et facultatifs pour la définition d'un travail de type commande distante	583		105. Propriétés des travaux IBM BigInsights, section d'application	658
75. Attributs obligatoires et facultatifs pour la définition d'un travail IBM i.	585		106. Propriétés des travaux JSR 352 Java Batch, section d'application	658
76. Attributs obligatoires et facultatifs pour la définition d'un travail exécutable.	588		107. Propriétés des travaux MQTT, section d'application.	659
77. Attributs obligatoires et facultatifs pour la définition d'un travail de la méthode d'accès.	590		108. Propriétés des travaux Apache Oozie, section d'application.	659
78. Attributs obligatoires et facultatifs pour la définition d'un travail de type TPM (Provisioning)	593		109. Propriétés des travaux Cloudant, section d'application.	660
79. Attributs obligatoires et facultatifs pour la définition d'un travail de transfert de fichier	598		110. Propriétés des travaux OpenWhisk, section d'application.	660
80. Attributs obligatoires et facultatifs pour la définition d'un travail d'automatisation OSLC.	608	=	111. Propriétés des travaux Gestion des travaux, section d'application	661
81. Attributs obligatoires et facultatifs pour la définition d'un travail d'application des accès OSLC.	610	=	112. Propriétés des travaux Soumission d'un flot de travaux, section d'application	661
82. Attributs obligatoires et facultatifs pour la définition d'un travail de base de données.. . . .	612	=	113. Propriétés des travaux de base de données, section d'application	661
83. Attributs obligatoires et facultatifs pour la définition d'un travail MSSQL.	615		114. Fonctions partiellement ou pas du tout prises en charge pour la planification dynamique.	675
84. Attributs obligatoires et facultatifs pour la définition d'un travail de services Web	618		115. Fonctions partiellement ou pas du tout prises en charge pour les travaux du flot de travaux USERJOBS	676
85. Attributs obligatoires et facultatifs pour la définition d'un travail de type services Web RESTful	621		116. Liste des commandes d'utilitaire	677
86. Attributs obligatoires et facultatifs pour la définition d'un travail Java.	625		117. Propriétés supplémentaires pouvant être utilisées pour la définition d'événements personnalisés.	694
87. Attributs obligatoires et facultatifs pour la définition d'un travail J2EE.. . . .	626		118. Liste des commandes d'utilitaire pour les postes de travail dynamiques	727
88. Attributs obligatoires et facultatifs pour la définition d'un travail de type JSR 352 Java Batch	629		119. Formats de date.	756
89. Attributs obligatoires et facultatifs pour la définition d'un travail MQTT	631		120. Liste des commandes de génération d'états	756
			121. Programmes d'extraction d'états	774
			122. Zones de sortie Jbextract	776
			123. Zones de sortie Prxtract	777

124. Zones de sortie Cextract	778	151. Commandes utilisées sur le plan	937
125. Zones de sortie Pextract	778	152. Commandes universelles.	939
126. Zones de sortie Rextract	779	153. Commandes composer	945
127. Zones de sortie R11xtr	780	154. Commandes pouvant être exécutées à partir de conman	950
128. Zones de sortie Xdep_job	781	155. Commandes d'utilitaire disponibles à la fois pour UNIX et Windows	954
129. Zones de sortie Xdep_job (suite)	782	156. Commandes d'utilitaire disponibles uniquement pour UNIX	957
130. Zones de sortie Xdep_sched	782	157. Commandes d'utilitaire disponibles uniquement pour Windows.	957
131. Zones de sortie Xfile	783	158. Commandes de génération d'états	957
132. Zones de sortie Xjob	783	159. Programmes d'extraction d'états	958
133. Zones de sortie Xprompts	784	160. Paramètres d'entrée pour le scénario de travail de branche négatif	978
134. Zones de sortie Xresource	784	161. Paramètres d'entrée pour le scénario de travail de modèle	981
135. Zones de sortie Xsched	785	162. Paramètres d'entrée pour le scénario de travail de modèle refusé	983
136. Zones de sortie Xwhen	785	163. Paramètres d'entrée pour le modèle dans le scénario de ligne de modèle	986
137. Formats de sortie de rapport pris en charge	787	164. Paramètres d'entrée pour le modèle annulé dans le scénario de ligne de modèle	989
138. Récapitulatif des rapports historiques	788	165. Paramètres d'entrée pour le scénario de comparaison numérique	991
139. Récapitulatif des rapports de production	792	166. Paramètres d'entrée pour le scénario de condition complexe	994
140. Options de tâche des commandes des méthodes	806	167. Paramètres d'entrée pour le scénario de pause/libération	1000
141. Messages de la tâche de lancement d'un travail (LJ)	813	168. Paramètres d'entrée pour le scénario d'action de signal.	1005
142. Messages de la tâche de vérification d'un fichier (CF)	814	169. Paramètres et valeurs	1014
143. Messages de la tâche d'obtention d'un état (GS).	815	170. Description des opérateurs arithmétiques	1015
144. Dépendances interréseaux dans un environnement mixte	825		
145. Transition d'état de travail reflet	848		
146. Critères de correspondance pour les travaux reflet distribués	851		
147. Syntaxe d'expression régulière.. . . .	877		
148. Exemples d'expressions régulières.	879		
149. Structure hiérarchique du fichier JSDL	894		
150. Types de ressource et propriétés	935		

Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.

OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

France	Canada	Etats-Unis
 (Pos1)		Home
Fin	Fin	End
 (PgAr)		PgUp
 (PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
 (Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

Brevets

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

Assistance téléphonique

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

A propos de cette publication

IBM Workload Scheduler simplifie la gestion des systèmes d'un bout à l'autre des environnements distribués grâce à l'intégration de fonctions de gestion de système. IBM Workload Scheduler planifie, automatise et contrôle le traitement de la charge de travail de votre entreprise dans son intégralité. Le document *IBM Workload Scheduler - Guide d'utilisation et de référence* fournit des informations détaillées sur l'interface de ligne de commande, le langage de planification et les commandes d'utilitaire de IBM Workload Scheduler.

Nouveautés

Prenez connaissance des nouveautés de cette édition.

Pour plus d'informations à propos des fonctions nouvelles ou modifiées de cette édition, voir *IBM Workload Automation - Présentation*, section *Récapitulatif des améliorations*.

Pour plus d'informations sur les APAR résolus par la présente édition, voir le document de téléchargement IBM Workload Scheduler Notes sur l'édition à l'adresse <http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048863> et Dynamic Workload Console Notes sur l'édition à l'adresse <http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048864>.

= Le contenu nouveau ou modifié est signalé par les barres de révision. Pour le
= format PDF, le contenu de la version 9.4 nouveau ou modifié est marqué dans la
= marge gauche d'une barre verticale (|) et le contenu de la version 9.4FP1 nouveau
= ou modifié est marqué d'un signe égal (=).

Nouveautés de cette publication

Prenez connaissance des nouveautés de cette publication.

Les APAR et les problèmes ont été corrigés. Toutes les modifications sont signalées par les barres de révision.

A qui s'adresse cette publication

Prenez connaissance du public concerné par cette publication.

Le présent manuel s'adresse aux personnes chargées de la planification, de l'ordonnancement, de la surveillance ou de la gestion d'un environnement de planification de la charge de travail. Il s'agit généralement d'opérateurs et d'administrateurs IBM Workload Scheduler.

Accessibilité

Les fonctions d'accessibilité permettent aux personnes souffrant d'un handicap physique (par exemple, une mobilité réduite ou une déficience visuelle) de pouvoir utiliser les logiciels.

Avec ce produit, vous pouvez utiliser les technologies d'assistance pour parcourir l'interface à l'aide de messages sonores. Vous pouvez également utiliser le clavier au lieu de la souris pour toutes les fonctions de l'interface graphique.

Pour plus d'informations, consultez l'annexe Accessibilité du document *IBM Workload Scheduler - Guide d'utilisation et de référence*.

Formation technique

Cloud & Smarter Infrastructure propose une formation technique.

Pour plus d'informations sur la formation technique Cloud & Smarter Infrastructure, voir : <http://www.ibm.com/software/tivoli/education>

Informations sur le support

IBM vous propose plusieurs façons d'obtenir de l'aide lorsque vous êtes confronté à un problème.

Si vous rencontrez un problème avec un logiciel IBM, vous pouvez le résoudre rapidement. IBM vous permet d'obtenir l'assistance que vous souhaitez de plusieurs manières :

- En faisant des recherches dans les bases de connaissances : elles contiennent un grand nombre de problèmes recensés et de solutions, de remarques d'ordre technique et autres informations adéquates.
- En vous procurant des correctifs : vous trouverez les versions les plus récentes disponibles pour votre produit.
- En contactant le service de support logiciel IBM : si les deux solutions ci-dessus ne vous ont pas permis de résoudre votre problème, vous pouvez contacter directement un technicien IBM de plusieurs manières.

Pour plus d'informations sur ces trois manières de résoudre un incident, voir l'annexe sur les informations de support dans *IBM Workload Scheduler - Guide d'identification et résolution des problèmes*.

Conventions du présent document

Prenez connaissance des conventions utilisées dans cette publication.

Le présent document utilise plusieurs conventions pour identifier les actions et les termes spécifiques, les chemins et les commandes propres au système d'exploitation, la syntaxe des commandes et les graphiques des marges.

Conventions typographiques

Ce document utilise les conventions typographiques suivantes :

Gras

- Commandes en minuscules et commandes en minuscules et majuscules difficiles à distinguer du reste du texte
- Commandes d'interface (cases à cocher, boutons de commande, boutons d'option, sélecteurs rotatifs, zones, dossiers, icônes, zones de liste, éléments des zones de liste, listes à plusieurs colonnes, conteneurs,

options de menu, noms de menu, onglets, fiches techniques), libellés (par exemple **Conseil** : et **Remarques relatives au système d'exploitation** :)

- Mot clés et paramètres dans le texte

Italique

- Mot définis dans le texte
- Insistance sur les mots (mots en tant que tels)
- Nouveaux termes dans le texte (sauf dans une liste de définitions)
- Variables et valeurs que vous devez fournir

Espacement fixe

- Exemples et exemples de code
- Noms de fichiers, mots clés de programmation et autres éléments difficiles à distinguer du reste du texte
- Invites et texte des messages destinés à l'utilisateur
- Texte à entrer par l'utilisateur
- Valeurs des arguments ou options de commande

Variables et chemins spécifiques au système d'exploitation

Pour les variables d'environnement et la notation des répertoires, ce document utilise la convention UNIX, excepté lorsque le contexte spécifique de l'exemple de chemin d'accès est Windows.

Lorsque vous utilisez une ligne de commande Windows, remplacez *\$variable* par *%variable%* pour les variables d'environnement, et remplacez toutes les barres obliques (/) par des barres obliques inversées (\) dans les chemins d'accès aux répertoires. Les noms des variables d'environnement diffèrent parfois dans les environnements Windows et UNIX. Par exemple, %TEMP% sous Windows correspond à \$tmp sous UNIX.

Remarque : Si vous utilisez le shell bash sur un système Windows, vous pouvez utiliser les conventions UNIX.

Syntaxe des commandes

Lorsqu'il décrit des commandes, le présent document utilise la syntaxe suivante :

Tableau 1. Syntaxe des commandes

Convention de syntaxe	Description	Exemple
Nom de la commande	Premier terme ou série de caractères consécutifs.	conman
Crochets ([])	Les informations entre crochets ([]) sont facultatives. Toutes les autres informations doivent être fournies.	[-file fichier_définition]
Accolades ({ })	Les accolades ({ }) désignent un ensemble d'options s'excluant mutuellement, lorsqu'une seule option est requise.	{-prompts -prompt nom_invite }
Trait de soulignement (_)	Les traits de soulignement (_) relient plusieurs mots dans une variable.	<i>nom_invite</i>

Tableau 1. Syntaxe des commandes (suite)

Convention de syntaxe	Description	Exemple
Barre verticale ()	Les options s'excluant mutuellement sont séparées par une barre verticale (). Vous pouvez entrer une seule des options séparées par la barre verticale au cours de la même utilisation de la commande.	{-prompts -prompt <i>nom_invite</i> }
Gras	Le texte en gras désigne les informations qui doivent être saisies sur la ligne de commande exactement telles quelles. Cela s'applique aux noms de commande et aux options invariables.	composer add <i>nom_fichier</i>
Italique	Le texte en italique est variable et doit être remplacé par les informations qu'il représente. Dans l'exemple de droite, l'utilisateur remplacera la variable <i>nom_fichier</i> par le nom du fichier approprié.	<i>nom_fichier</i>
Points de suspension (...)	Les points de suspension (...) placés après une option indiquent que l'option peut être répétée plusieurs fois avec des valeurs différentes. Ils peuvent être employés à l'intérieur ou à l'extérieur des crochets.	[-x <i>nom_fichier</i>]... Les points de suspension à l'extérieur des crochets indiquent que la variable -x <i>nom_fichier</i> est facultative et peut être répétée comme suit : -x <i>nom_fichier1</i> -x <i>nom_fichier2</i> -x <i>nom_fichier3</i> [-x <i>nom_fichier</i> ...] Les points de suspension placés entre crochets signifient que la variable -x <i>nom_fichier</i> est facultative et qu'elle peut être répétée comme suit : -x <i>nom_fichier1</i> <i>nom_fichier2</i> <i>nom_fichier3</i> -x <i>nom_fichier</i> [-x <i>nom_fichier</i>]... Les points de suspension utilisés sous cette syntaxe indiquent que vous devez définir au moins une fois la variable -x <i>nom_fichier</i> .

Chapitre 1. Présentation du produit IBM Workload Scheduler

Grâce à IBM Workload Scheduler, vous pouvez gérer l'environnement de production et automatiser plusieurs activités de surveillance. IBM Workload Scheduler gère le traitement des travaux, résout les interdépendances, lance et suit le déroulement des travaux. Dans la mesure où les travaux sont exécutés immédiatement après résolution de leurs dépendances, le délai d'inactivité est réduit au minimum et la productivité est largement optimisée. Si un travail échoue, IBM Workload Scheduler assure le processus de reprise en sollicitant peu (voire pas du tout) l'aide de l'opérateur.

Le présent chapitre comprend les sections suivantes :

- «Explication des concepts de base»
- «Interfaces utilisateur IBM Workload Scheduler», à la page 30
- «Démarrage de la production», à la page 33

Explication des concepts de base

La présente section décrit les concepts de base de IBM Workload Scheduler et comprend les sections suivantes :

- «Objets de base de données IBM Workload Scheduler»
- «Réseau IBM Workload Scheduler», à la page 20
- «Configuration de l'environnement d'exécution IBM Workload Scheduler», à la page 21
- «Définition d'activités de planification à l'aide de IBM Workload Scheduler», à la page 22
- «Gestion des activités de planification de production avec IBM Workload Scheduler», à la page 28

Objets de base de données IBM Workload Scheduler

Cette section présente les objets de base de données IBM Workload Scheduler que vous utilisez. Les objets de base de données suivants sont décrits :

- Travail, voir «Travail», à la page 2
- Flot de travaux, voir «Flot de travaux», à la page 2
- Application de charge de travail, voir «Application de charge de travail», à la page 3
- Cycle d'exécution, voir «Cycle d'exécution», à la page 4
- Groupe de cycle d'exécution, voir «Groupe de cycle d'exécution», à la page 5
- Agenda, voir «Agenda», à la page 10
- Invite, voir «Invite», à la page 10
- Poste de travail, voir «Poste de travail», à la page 11
- Classe de poste de travail, voir «Classe de postes de travail», à la page 15
- Domaine, voir «Domaine, zone», à la page 16
- Règle d'événement, voir «Règle d'événement», à la page 19
- Ressource, voir «Resource», à la page 19
- Paramètre, voir «Paramètre», à la page 19
- Table de variables, voir «Table de variables», à la page 20

Travail

Un *travail* est une unité spécifiant une action, par exemple une sauvegarde des données hebdomadaire, à effectuer sur des postes de travail spécifiques du réseau IBM Workload Scheduler. Dans un environnement réparti IBM Workload Scheduler en revanche, ils peuvent être définis de manière indépendante à partir de flots de travaux ou dans une définition de flot de travaux.

Les types de travail peuvent être divisés entre des travaux IBM Workload Scheduler et des types de travaux avec options avancées. Les types de travail existants sont des scripts génériques ou des commandes que vous personnalisez en fonction de vos besoins. Les types de travaux avec options avancées sont des travaux conçus pour effectuer des opérations spécifiques, telles que des opérations de base de données, de transfert de fichier, Java et de service Web. Vous planifiez ces types de travaux uniquement sur les agents, les pools et les pools dynamiques.

Si vous souhaitez optimiser la fonction dynamique lors de la planification des types de travaux avec options avancées, vous les planifiez sur des pools et des pools dynamiques, ce qui affecte dynamiquement le travail aux meilleures ressources disponibles. Si n'avez besoin de définir que des types de travaux avec options avancées, sans utiliser la fonction de planification dynamique, vous planifiez ces travaux sur un agent spécifique, sur lequel le travail s'exécute de façon statique.

Que le moteur IBM Workload Scheduler soit distribué ou basé sur z/OS, vous pouvez définir localement un *travail reflet* pour mapper l'instance de travail distant qui s'exécute sur un moteur IBM Workload Scheduler différent.

Pour plus d'informations sur la manière de définir des travaux, voir «Définition de travail», à la page 187.

Pour plus d'informations sur la définition des postes de travail, voir «Définition de poste de travail», à la page 166.

Flot de travaux

Un *flot de travaux* est une séquence de travaux à exécuter, ainsi que des heures, des priorités et d'autres dépendances déterminant l'ordre du traitement. A chaque flot de travaux est affectée une heure d'exécution symbolisée par un cycle d'exécution avec un agenda, un ensemble de dates ou des fréquences de répétition.

Dépendances dans un environnement réparti :

Vous pouvez avoir des dépendances entre des travaux et des flots de travaux. Il peut s'agir de :

Dépendances internes

Certaines dépendances sont établies entre des travaux appartenant au même flot de travaux.

Dépendances externes

Il s'agit de dépendances entre des flots de travaux, entre des flots de travaux et des travaux appartenant à d'autres flots, ou encore entre des travaux appartenant à divers flots de travaux.

Dépendances interréseaux

Dépendances par rapport à des travaux ou des flots de travaux s'exécutant sur un autre réseau IBM Workload Scheduler. Les dépendances interréseaux requièrent un poste de travail agent réseau pour communiquer avec le réseau IBM Workload Scheduler externe.

Les dépendances sur des ressources sont prises en charge par IBM Workload Scheduler tant dans les environnements répartis que z/OS.

Pour plus d'informations sur la définition des flots de travaux, voir «Définition de flot de travaux», à la page 226.

Application de charge de travail

Une application de charge de travail représente un ou plusieurs flots de travaux ainsi que tous les travaux référencés qui peuvent être partagés avec d'autres environnements IBM Workload Scheduler par un processus de déploiement simple.

Une application de charge de travail est un objet de base de données IBM Workload Scheduler qui agit en tant que conteneur pour un ou plusieurs flots de travaux. Vous pouvez utiliser une applications de charge de travail pour normaliser une solution d'automatisation de charge de travail de sorte que la solution puisse être réutilisée dans un ou plusieurs environnements IBM Workload Scheduler automatisant de ce fait des processus métier.

Vous pouvez préparer un modèle de application de charge de travail dans un environnement IBM Workload Scheduler source, puis l'exporter de sorte qu'il puisse être déployé dans un environnement cible. Le processus d'exportation extrait de l'environnement source tous les éléments nécessaires pour faire une copie la solution dans un autre environnement. Il produit un fichier compressé contenant un certain nombre de fichiers nécessaires pour importer application de charge de travail dans l'environnement cible. Ces fichiers contiennent une définition des objets dans l'environnement source extrait de la base de données IBM Workload Scheduler. Pour les éléments qui dépendent de la topologie de l'environnement cible, une configuration manuelle est nécessaire. Par exemple, les définitions extraites de l'environnement source contiennent des références aux postes de travail qui n'existent pas dans l'environnement cible. Pour cette raison, avant de procéder à l'importation, un mappage de certains des éléments doit être fait en associant le nom de l'objet dans l'environnement cible.

Le modèle exporté de l'application de charge de travail contient des définitions ou des références pour tous les objets suivants :

- Flots de travaux
- Travaux
- Postes de travail, classes de postes de travail
- Agendas
- Invites
- Cycles d'exécution
- Groupes de cycle d'exécution
- Ressources
- Dépendances interréseaux
- Dépendances externes

Pour plus d'informations sur la définition de modèles d'application de charge de travail, voir "Définition d'application de charge de travail" dans le document *IBM Workload Scheduler - Guide d'utilisation et de référence*.

Cycle d'exécution

Un *cycle d'exécution* indique pour quels jours l'exécution d'un flot de travaux est planifiée. Un cycle est défini pour un flot de travaux spécifique et ne peut pas être utilisé par plusieurs flots de travaux. Vous pouvez définir les types suivants de cycle d'exécution :

Simple

Jours définis par l'utilisateur pendant lesquels un flot de travaux sera exécuté. Un cycle d'exécution simple est défini pour un flot de travaux spécifique et il ne peut pas être utilisé par d'autres flots de travaux.

Quotidien

Cycle d'exécution indiquant que l'exécution du flot de travaux s'effectue selon une certaine fréquence et un type de jour que vous avez définis. Par exemple, il peut s'exécuter quotidiennement, tous les trois jours ou uniquement les jours ouvrés.

Hebdomadaire

Cycle d'exécution spécifiant les jours de la semaine au cours desquels un flot de travaux est exécuté. Par exemple, en faisant appel à ce cycle, vous pouvez demander que l'exécution ait lieu chaque lundi, mercredi et vendredi.

Mensuel

Cycle d'exécution indiquant que l'exécution du flot de travaux s'effectue selon une certaine date ou un jour que vous avez défini mensuellement. Par exemple, il peut s'exécuter tous les premiers et deuxièmes jours du mois, tous les deux mois ou tous les premiers lundis et deuxièmes mardis du mois, tous les trois mois.

Annuel

Cycle d'exécution indiquant que l'exécution du flot de travaux s'effectue, par exemple, tous les ans ou tous les trois ans.

Basé sur des décalages

Cycle d'exécution qui utilise une combinaison de périodes et décalages définis par l'utilisateur. Par exemple, un décalage de 3 sur une période de 15 jours est le troisième jour à partir du début de la période. Pour les périodes cycliques, il est préférable d'utiliser des cycles d'exécution basés sur des décalages. Ce terme est utilisé comme tel uniquement dans IBM Workload Scheduler for z/OS, mais le concept s'applique également au produit distribué.

Basé sur des règles

Cycle d'exécution qui utilise des règles basées sur des listes de nombres ordinaux, des types de jour, des intervalles d'agenda communs (ou noms de période dans IBM Workload Scheduler for z/OS). Par exemple, le dernier jeudi de chaque mois. Les cycles d'exécution basés sur des règles sont fondés sur des périodes conventionnelles, telles que les mois, les semaines et les jours de la semaine. Dans IBM Workload Scheduler for z/OS, les cycles d'exécution peuvent également reposer sur des périodes que vous définissez, par exemple des semestres. Ce terme est utilisé comme tel uniquement dans IBM Workload Scheduler for z/OS mais le concept s'applique également au produit distribué. Vous pouvez également indiquer une règle afin d'établir quand le flot de travaux doit s'exécuter si l'exécution définie tombe un jour chômé.

Ces types de cycle d'exécution peuvent être inclusifs ou exclusifs, c'est-à-dire :

Inclusif

Cycle d'exécution indiquant les jours et heures auxquels l'exécution d'un flot de travaux a été planifiée. Les cycles d'exécution inclusifs ne sont pas prioritaires par rapport aux cycles d'exécution exclusifs.

Exclusif

Cycle d'exécution indiquant les jours et heures auxquels un flot de travaux ne peut pas être exécuté. Les cycles d'exécution exclusifs sont prioritaires par rapport aux cycles d'exécution inclusifs.

Groupe de cycle d'exécution

Vous pouvez éventuellement définir un groupe de cycle d'exécution pour votre flot de travaux au lieu, ou en plus, d'un certain nombre de cycles d'exécution uniques.

Un groupe de cycle d'exécution est une liste de cycles d'exécution qui sont combinés pour produire un ensemble de dates d'exécution.

A l'aide des groupes de cycle d'exécution, vous pouvez bénéficier des avantages suivants :

Un groupe de cycle d'exécution est un objet de base de données distinct

Il est défini par lui-même et peut être mis en correspondance avec un ou plusieurs flots de travaux. Il n'est pas défini en tant qu'élément d'un flot de travaux spécifique comme les cycles d'exécution uniques.

Le même groupe de cycle d'exécution peut être utilisé sur différents flots de travaux

Ceci améliore la convivialité globale des cycles d'exécution, parce que vous pouvez indiquer le même groupe de cycle d'exécution dans des flots de travaux multiples, ce qui évite de recourir à des définitions de cycle d'exécution multiples pour les mêmes règles de planification.

Les groupes de cycle d'exécution étendent l'utilisation des cycles d'exécution exclusifs

Des cycles d'exécution exclusifs (ou négatifs) sont utilisés pour générer des occurrences négatives, qui identifient les jours où un flot de travaux serait normalement planifié mais où il n'est pas nécessaire. La somme des cycles d'exécution exclusifs est soustraite de celle des cycles d'exécution inclusifs. Une occurrence négative annule toujours toutes les occurrences positives correspondantes et vous pouvez indiquer une occurrence négative seulement si l'équivalent positif existe déjà. Une correspondance exacte des jours, ainsi que les restrictions de temps, est nécessaire entre les cycles d'exécution exclusifs et inclusifs pour que l'annulation puisse se produire. Les groupes de cycle d'exécution ajoutent beaucoup de flexibilité en permettant aux utilisateurs d'appliquer des cycles d'exécution exclusifs à un sous-ensemble de cycles positifs plutôt qu'à tous les cycles. Regroupez vos cycles d'exécution dans des *sous-ensembles* de sorte que les cycles d'exécution exclusifs puissent être appliqués seulement aux occurrences positives générées par les cycles d'exécution appartenant au même ensemble.

Les cycles d'exécution doivent être organisés dans des *sous-ensembles* au sein d'un groupe de cycle d'exécution. Les sous-ensembles sont toujours dans une relation logique **OU** les uns avec les autres. Le résultat du groupe de cycle d'exécution est toujours une date ou un ensemble de dates ; il ne peut pas être négatif.

Par exemple, vous voudrez peut-être que votre flot de travaux s'exécute chaque jour du mois excepté le dernier jour du mois. Mais, vous voulez

également qu'il soit planifié le dernier jour de l'année (le dernier jour de décembre). Vous pouvez définir un groupe de cycle d'exécution à l'aide de sous-ensembles, comme suit :

Sous-ensemble 1

- **Cycle d'exécution 1** - cycle d'exécution inclusif chaque jour du mois
- **Cycle d'exécution 2** - cycle d'exécution exclusif le plus dernier jour du mois

Sous-ensemble 2

- **Cycle d'exécution 3** - cycle d'exécution inclusif le 31 décembre

où, le cycle d'exécution 2 annule le dernier jour de chaque mois dans le sous-ensemble 1, alors que le cycle d'exécution 3 génère le 31 décembre en tant que date distincte ; par conséquent, vous pouvez planifier le flot de travaux le 31 décembre.

Les groupes de cycle d'exécution permettent d'utiliser des cycles d'exécution logiques ET entre des cycles d'exécution individuels dans le sous-ensemble

Par défaut, les cycles d'exécution dans un sous-ensemble sont dans une relation logique **OU** mais vous pouvez la changer en relation **ET** logique si le résultat du groupe de cycle d'exécution est une date ou un ensemble positif de dates (inclusif). Pour chaque cycle d'exécution, vous pouvez indiquer l'un ou l'autre des opérateurs (**ET**, **OU**), et obtenir le comportement suivant :

1. Tous les cycles d'exécution du groupe qui sont dans la relation **ET** sont calculés en premier. Le résultat de ce calcul est une date ou un ensemble de dates.
2. Puis, tous les cycles d'exécution dans la relation **OU** sont ajoutés au résultat de l'étape précédente.

Un comportement semblable est appliqué aux cycles d'exécution inclusifs et exclusifs pour déterminer la date limite ou l'ensemble de dates d'un groupe.

Inclusif (A)

Cycle d'exécution basé sur des règles. Sélectionnez les jours où le flot de travaux doit être exécuté s'ils appartiennent à tous les types A de l'ensemble de cycles d'exécution.

Exclusif (D)

Cycle d'exécution basé sur des règles d'exclusion. Sélectionnez les jours où le flot de travaux NE doit PAS être exécuté s'ils appartiennent à tous les types D de l'ensemble de cycles d'exécution.

Par exemple, vous pouvez ajouter deux conditions ensemble :

Exécuter le mercredi "ET" le 8ème jour ouvrable du mois.

De cette façon, les seules dates planifiées sont n'importe quel 8ème jour ouvrable du mois qui tombe un mercredi.

Compatibilité complète avec des cycles d'exécution *traditionnels*

Les cycles d'exécution *traditionnels* indiqués dans la définition du flot de travaux peuvent référencer des groupes de cycle d'exécution, avec la possibilité d'indiquer le ou les décalages (tout comme avec les périodes pour z/OS ou avec les agendas pour les systèmes répartis).

Un ensemble de dates (début d'intervalle) est créé automatiquement au niveau de cycle d'exécution directement (inclusivement ou exclusivement avec des décalages, ou dans la règle. Il s'agit d'un processus en deux étapes avec les cycles d'exécution suivants :

1. Définir l'"événement métier" clé, tel que la Fin du mois, à l'aide des cycles d'exécution et des règles de jours libres
2. Définir les règles qui utilisent les dates de l'"événement métier" comme intervalles par rapport auxquels l'autre exécution par lots peut être planifiée.

Par exemple, vous disposez d'un *processus de fin du mois* qui s'exécute le dernier vendredi d'un mois, mais qui est repoussé au jour ouvrable suivant, sauf en décembre où il s'exécute le 3ème vendredi du mois. Cette règle de planification peut être définie avec quelques règles, cycles d'exécution et règles de plage horaire libres.

Deux jours ouvrables avant fin du mois, vous devez exécuter un processus de pré-validation pour régler les problèmes métier avant l'exécution. Vous ne pouvez pas sélectionner le dernier mercredi du mois, parce qu'avec certains mois, celui-ci peut se trouver après le dernier vendredi. De même, si le dernier vendredi était un jour chômé, le dernier mercredi ne se situera pas 2 jours ouvrables avant celui-ci, car la règle de jour chômé s'applique UNIQUEMENT au jour où la règle entre en vigueur.

Il se peut que beaucoup d'autres exécutions par lot doivent également être exécutées un certain nombre de jours avant ou après la fin du mois, mais les mêmes restrictions s'appliquent.

Vous pouvez maintenant définir le travail pour qu'il s'exécute relativement à un élément défini par une combinaison de cycles d'exécution et pour libérer les règles quotidiennes.

Utilisation des agendas avec des cycles d'exécution au sein d'un groupe de cycle d'exécution

Vous pouvez éventuellement indiquer plusieurs agendas pour calculer la définition de jours ouvrables et chômés pour un cycle d'exécution. L'agenda principal est utilisé pour calculer qui les jours ouvrables qui sont valides, et un agenda secondaire est utilisé pour calculer les dates chômées spécifiques. Si les dates calculées selon l'agenda secondaire correspondent aux jours ouvrables dans l'agenda principal, le travail est planifié ; s'ils ne correspondent pas, le travail n'est pas planifié.

Par exemple, une société globale qui exécute la charge de travail aux Etats-Unis pour beaucoup d'autres pays nécessite un grand nombre de combinaisons d'agenda pour s'assurer que les travaux par lots fonctionnent seulement un jour qui est un jour ouvrable aux Etats-Unis et dans l'autre pays. L'agenda peut être défini au niveau du flot de travaux et, s'il n'est pas spécifié, un agenda par défaut est utilisé. Cependant, lorsqu'il est défini, l'agenda au niveau du cycle d'exécution peut être utilisé comme agenda secondaire et l'agenda de flot de travaux (ou agenda par défaut) peut être utilisé comme agenda principal.

Par exemple, le calendrier principal peut être *JOURS OUVRABLES*, défini du LUNDI au VENDREDI à l'exclusion des dates de congés des Etats-Unis. Vous pourriez également devoir calculer les exécutions de travail basées sur l'agenda *HKWORK*, qui est défini du lundi au vendredi, à l'exclusion des dates de congés de Hong Kong. Le travail pourrait avoir plusieurs planifications :

- Exécution les jours ouvrables, mais pas le dernier jour ouvrable et pas le lundi
- Exécution le lundi, mais pas le dernier jour ouvrable
- Exécution le dernier jour ouvrable

Puisque chaque planification est calculée en fonction de l'agenda *WORKHK*, elle est également contrôlée par rapport à l'agenda *WORKDAYS* pour s'assurer qu'elle est planifiée un jour ouvrable des Etats-Unis.

Utilisation des restrictions de temps avec les groupes de cycle d'exécution

Vous pouvez indiquer des contraintes de temps pour définir le moment où le traitement doit commencer ou la durée au bout de laquelle le traitement ne doit plus commencer. Pour ce faire, vous pouvez associer les *restrictions de temps* au travail, aux flots de travaux, aux cycles d'exécution et aux groupes de cycle d'exécution. Quand vous définissez une restriction de temps, vous obtenez fondamentalement un *moment*. Puisque vous pouvez associer des restrictions de temps à plusieurs objets, la hiérarchie suivante indique l'ordre dans lequel les différentes restrictions de temps sont prises en compte pour définir réellement quand commencer le traitement :

1. Restriction de temps définie dans le cycle d'exécution du flot de travaux
2. Restriction de temps définie dans le flot de travaux
3. Restriction de temps définie dans le cycle d'exécution contenu dans le groupe de cycle d'exécution associé au flot de travaux.
4. Restriction de temps définie dans le groupe de cycle d'exécution associé au flot de travaux.
5. Début de la journée

Cela implique les points suivants :

Les restrictions de temps du flot de travaux

Substituent et sont prioritaires *sur toutes les autres restrictions de temps* définies dans les cycles d'exécution ou dans les groupes de cycle d'exécution associés au flot de travaux.

Aucune restriction de temps dans le flot de travaux ni dans le groupe de cycle d'exécution

Le groupe génère seulement une date qui est le *Début de journée*. Si des décalages et des règles de jours chômés doivent être calculés, le calcul commence toujours à partir du *Début de journée*.

Restrictions de temps dans le groupe de cycle d'exécution (pas dans le flot de travaux)

Les restrictions de temps (et le décalage possible) sont calculés à partir du *Début de journée* et la date et l'heure qui en résulte indique le début du traitement.

Exemples

Tableau 2. Scénario 1. Aucune restriction de temps dans le groupe de cycle d'exécution

Groupe de cycle d'exécution	Date planifiée	Premier démarrage
Groupe de cycle d'exécution	10/24	10/24
Groupe de cycle d'exécution avec décalage (+ 3 jours)	10/27 (samedi)	10/27/ (samedi)
Groupe de cycle d'exécution avec règle de jours chômés	10/29/ (lundi)	0/29/ (lundi)

Tableau 2. Scénario 1. Aucune restriction de temps dans le groupe de cycle d'exécution (suite)

Groupe de cycle d'exécution	Date planifiée	Premier démarrage
Cycle d'exécution dans le flot de travaux avec restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec le début au plus tôt +1 13h	11/02 (vendredi)	11/03 (samedi) 13h
Cycle d'exécution dans le flot de travaux sans restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/02 (vendredi) Début de journée
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/02 (vendredi) Début de journée

Tableau 3. Scénario 2. Restriction de temps dans le groupe de cycle d'exécution sans décalage

Groupe de cycle d'exécution	Date planifiée	Premier démarrage
Groupe de cycle d'exécution	10/24	10/24
Groupe de cycle d'exécution avec décalage d'agenda (+ 3 jours)	10/27/ (samedi)	10/27/ (samedi)
Groupe de cycle d'exécution avec règle de jours chômés	10/29/ (lundi)	0/29/ (lundi)
Cycle d'exécution dans le flot de travaux avec restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec le début au plus tôt +1 13h	11/02 (vendredi)	11/03 (samedi) 13h
Cycle d'exécution dans le flot de travaux sans restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/02 (vendredi) Début de journée
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/02 (vendredi) Début de journée

Tableau 4. Scénario 3. Restriction de temps dans le groupe de cycle d'exécution avec décalage (+1 12h00)

Groupe de cycle d'exécution	Date planifiée	Premier démarrage
Groupe de cycle d'exécution	10/24	10/24
Groupe de cycle d'exécution avec décalage d'agenda (+ 3 jours)	10/27/ (samedi)	10/27/ (samedi)
Groupe de cycle d'exécution avec règle de jours chômés	10/29/ (lundi)	10/29/ (lundi)
Groupe de cycle d'exécution avec décalage +1 12h00	10/29/ (lundi)	10/30 12h00 (mardi)
Cycle d'exécution dans le flot de travaux avec restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/02 (vendredi)
Cycle d'exécution dans le flot de travaux avec le début au plus tôt +1 13h	11/02 (vendredi)	11/03 (samedi) 13h
Cycle d'exécution dans le flot de travaux sans restriction de temps		
Cycle d'exécution dans le flot de travaux avec décalage de + 4 jours ouvrables	11/02 (vendredi)	11/03 12:00 (samedi)
Cycle d'exécution dans le flot de travaux avec règle de jours chômés	11/02 (vendredi)	11/03 12:00 (samedi)

z/OS Disponibilité de la commande GENDAYS au niveau du groupe de cycle d'exécution

En utilisant GENDAYS, vous pouvez vérifier le résultat de la combinaison de tous les cycles d'exécution du groupe.

Agenda

Un *agenda* est une liste de dates indiquant si un flot de travaux s'exécute et quand.

Un agenda peut également être sélectionné pour être utilisé comme agenda des *jours chômés* dans un flot de travaux. Un agenda des jours chômés est un agenda affecté à un flot de travaux afin de représenter les jours où le flot de travaux et les travaux associés ne s'exécutent pas. Il peut également être configuré pour considérer les samedis et/ou les dimanches comme jours ouvrables. Par convention, de nombreux utilisateurs définissent un agenda des jours chômés nommé *holidays*, dans lequel le samedi et le dimanche sont spécifiés comme jours chômés.

Pour plus d'informations sur la manière de définir des agendas, voir «Définition d'agenda», à la page 205.

Invite

Une *invite* est un texte s'affichant sous forme de message et interrompant le traitement d'un travail ou flot de travaux jusqu'à ce qu'une réponse affirmative soit reçue. (La réponse peut être indiquée manuellement par l'opérateur ou

automatiquement par une action de règle d'événement.) Une fois la réponse à l'invite obtenue, le traitement reprend. Les invites peuvent servir de dépendances dans les travaux ou les flots de travaux. Vous pouvez également utiliser des invites pour informer un opérateur de l'exécution d'une tâche spécifique. Dans ce cas, aucune réponse n'est nécessaire de la part de l'opérateur.

Il existe trois types d'invites :

Invite globale ou nommée

Invite définie dans la base de données en tant qu'objet de planification. Elle est identifiée par un nom unique et peut être utilisée par n'importe quel travail ou flot de travaux.

Invite locale ou ad-hoc

Invite définie au sein d'une définition de travail ou de flot de travaux. Elle ne possède pas de nom et elle n'est pas définie en tant qu'objet de planification dans la base de données ; elle ne peut donc pas être utilisée par d'autres travaux ou flots de travaux.

Invite de reprise ou fin anormale

Type spécifique d'invite dont l'utilisation est réservée aux fins de travail anormales. La réponse à cette invite détermine le résultat du travail ou du flot de travaux auquel le travail appartient. Une invite de reprise peut également être associée à une action et à un type spécial de travail appelé *travail de reprise*.

Pour plus d'informations sur la manière de définir des invites, voir «Définition d'invite», à la page 213.

Poste de travail

Cette section fournit des informations concernant l'utilisation des postes de travail pour la planification des travaux et des flots de travaux. Si vous souhaitez en savoir plus sur les postes de travail dans le cadre de la planification de votre réseau, toutes les informations dont vous avez besoin se trouvent dans le document *IBM Workload Scheduler: Planification et installation*.

Le système informatique sur lequel vous exécutez vos travaux et flots de travaux est appelé *poste de travail*. Lorsque vous définissez un travail ou un flot de travaux dans la base de données IBM Workload Scheduler, vous identifiez les définitions de poste de travail pour les systèmes informatiques physiques ou virtuels sur lesquels l'exécution de votre travail est planifiée. Les postes de travail peuvent être regroupés de manière logique selon des *classes de postes de travail* et organisés dans une hiérarchie de *domaines*, laquelle est gérée par des *gestionnaires de domaine*.

Pour plus d'informations relatives aux classes de poste de travail, voir «Classe de postes de travail», à la page 15. Pour les domaines, voir «Domaine, zone», à la page 16.

Lorsque vous créez une définition de poste de travail pour un système appartenant à votre réseau, vous déterminez un ensemble de caractéristiques qui identifie de manière unique ce système et affecte son mode d'exécution des travaux. Voici quelques exemples de ces caractéristiques : l'adresse IP du poste de travail, s'il est ou non derrière un pare-feu, si ses communications sont sécurisées ou non, le fuseau horaire du poste de travail et l'identité de son gestionnaire de domaine.

Les postes de travail du réseau de planification IBM Workload Scheduler peuvent non seulement traiter les travaux et les flots de travaux, mais aussi remplir d'autres

rôles. Lors de la conception du réseau, des rôles ont été affectés aux postes de travail en fonction des besoins de votre activité. Vous trouverez ci-dessous une liste complète des rôles de poste de travail :

Gestionnaire de domaine maître

Poste de travail agissant comme concentrateur de gestion pour le réseau. Il gère l'ensemble de vos objets de planification. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que poste **maître**.

Gestionnaire de domaine maître de sauvegarde

Poste de travail qui peut servir de sauvegarde pour le gestionnaire de domaine maître, en cas d'incident. Il s'agit véritablement d'un gestionnaire de domaine maître, en attente d'activation. Son utilisation est facultative. Pour vous familiariser avec le basculement vers un gestionnaire de domaine maître de sauvegarde, reportez-vous au document *IBM Workload Scheduler - Guide d'administration*. Ce poste de travail doit être installé comme gestionnaire de domaine maître configuré en tant que copie de sauvegarde. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant qu'**agent tolérant aux pannes**.

Gestionnaire de domaine

Poste de travail qui contrôle un domaine et qui partage des responsabilités de gestion pour une partie du réseau IBM Workload Scheduler. Il est installé comme un agent, puis il est configuré comme un poste de travail de domaine maître lorsque vous définissez le poste de travail dans la base de données. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que poste **manager** (gestionnaire).

Gestionnaire de domaine dynamique

Composant installé d'un réseau IBM Workload Scheduler distribué jouant le rôle de concentrateur de gestion dans un domaine. Toutes les communications à destination et en provenance des agents d'un domaine passent par le gestionnaire de domaine dynamique. Lorsque vous installez un gestionnaire de domaine dynamique, les types de postes de travail suivants sont créés dans la base de données :

fta Composant agent tolérant aux pannes configuré manuellement en tant que gestionnaire de domaine

broker Composant serveur courtier

agent Composant agent dynamique

gestionnaire de domaine dynamique de secours

Poste de travail qui peut servir de sauvegarde pour le gestionnaire de domaine dynamique lorsque des problèmes surviennent. Il s'agit véritablement d'un gestionnaire de domaine dynamique, en attente d'activation. Son utilisation est facultative. Pour vous familiariser avec le basculement vers un gestionnaire de domaine dynamique de sauvegarde, reportez-vous au document *IBM Workload Scheduler - Guide d'administration*. Lorsque vous installez un gestionnaire de domaine dynamique, les types de postes de travail suivants sont créés dans la base de données :

fta Composant agent tolérant aux pannes.

broker Composant serveur courtier

agent Composant agent dynamique

Agent tolérant aux pannes

Poste de travail qui reçoit et traite des travaux. En cas de problème de communication avec son gestionnaire de domaine, il peut exécuter les travaux en local. Il est installé comme un agent, puis il est configuré comme un poste de travail de l'agent tolérant aux pannes lorsque vous définissez le poste de travail dans la base de données. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant qu'**agent tolérant aux pannes**.

Agent standard

Poste de travail qui reçoit et exécute uniquement les travaux sous le contrôle de son gestionnaire de domaine. Il est installé comme un agent, puis il est configuré comme un poste de travail d'agent standard lorsque vous définissez le poste de travail dans la base de données. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que **s-agent** (agent standard).

Agent étendu

Poste de travail sur lequel une méthode d'accès IBM Workload Scheduler a été installée en tant que pont, de manière à pouvoir planifier des travaux sur des applications SAP R/3, Oracle E-Business Suite, PeopleSoft et z/OS ou des applications personnalisées. Il doit être physiquement hébergé par un gestionnaire de domaine maître, un gestionnaire de domaine, un agent standard ou un agent tolérant aux pannes puis défini en tant qu'agent étendu dans la base de données. Pour plus d'informations, voir *IBM Workload Scheduler - Guide d'utilisation*. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que poste **x-agent** (agent étendu).

Courtier de charge de travail

Poste de travail où s'exécutent à la fois des types de travaux existants et des types de travaux avec options avancées. Il s'agit du serveur courtier installé avec le gestionnaire de domaine maître et le gestionnaire de domaine dynamique. Il peut héberger l'un ou plusieurs des postes de travail suivants :

- Agent étendu
- Moteur distant
- Pool
- Pool dynamique
- Agent. Cette définition comprend les agents suivants :
 - Agent
 - agent IBM Workload Scheduler for z/OS
 - Agent pour z/OS

Pour plus d'informations sur l'agent et l'agent IBM Workload Scheduler for z/OS, voir *Scheduling Workload Dynamically*. Pour plus d'informations sur le agent pour z/OS, voir *Planification avec l'agent pour z/OS*.

Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que **broker** (courtier).

Agent dynamique

Poste de travail qui gère une grande variété de types de travaux, comme par exemple les travaux d'une base de données ou d'un protocole FTP spécifiques, en plus des types de travaux existants. Ce poste de travail est automatiquement créé et enregistré dans la base de données IBM Workload Scheduler lorsque vous installez l'agent. L'agent est hébergé par le poste de

travail Workload Broker. Etant donné que les processus d'installation et d'enregistrement sont effectués automatiquement, lorsque vous affichez l'agent dans Dynamic Workload Console, il apparaît mis à jour par le Resource Advisor Agent. Vous pouvez regrouper les agents dans des pools et des pools dynamiques. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant qu'**agent**.

Dans une configuration simple, les agents dynamiques se connectent directement à un gestionnaire de domaine maître ou à un gestionnaire de domaine dynamique. Toutefois, dans des topologies de réseau plus complexes, si la configuration de réseau empêche le gestionnaire de domaine maître ou le gestionnaire de domaine dynamique de communiquer directement avec le agent dynamique, vous pouvez configurer vos agents dynamiques pour utiliser une passerelle locale ou distante.

Remarque : Si l'option globale `enAddWorkstation` vaut "yes", la définition de poste de travail d'agent dynamique est automatiquement ajoutée au plan après que le processus d'installation a créé le poste de travail d'agent dynamique dans la base de données.

Pool Un regroupement de postes de travail logiques est un ensemble d'agents avec des caractéristiques matérielles ou logicielles similaires, sur lequel sont soumis les travaux. IBM Workload Scheduler équilibre les travaux entre les agents du pool et réaffecte automatiquement les travaux aux agents disponibles lorsqu'un agent n'est plus disponible. Pour créer un pool d'agents dans votre environnement IBM Workload Scheduler, définissez un poste de travail de type **pool** hébergé par le poste de travail Workload Broker, puis sélectionnez les agents que vous souhaitez ajouter au pool. Vous pouvez définir le pool à l'aide de Dynamic Workload Console ou de la commande **composer**. Ce poste de travail est enregistré dans la base de données IBM Workload Scheduler en tant que **pool**. Lorsque vous créez un pool dans votre environnement IBM Workload Scheduler, une ressource logique portant le même nom est automatiquement créée dans Dynamic Workload Broker. La ressource logique est utilisée pour lier et regrouper les agents appartenant au même pool, et en tant que condition requise pour les travaux planifiés dans le pool IBM Workload Scheduler. Supposons que ces objets de base de données sont deux objets différents. Si vous renommez le pool IBM Workload Scheduler, cette modification n'est pas apportée à la ressource logique Dynamic Workload Broker.

Pool dynamique

Un regroupement de postes de travail logiques est un ensemble d'agents, défini dynamiquement en fonction des exigences de ressources que vous indiquez et qui est hébergé par le poste de travail Workload Broker. Par exemple, si vous avez besoin d'un poste de travail sollicitant peu l'unité centrale et que Windows est installé pour exécuter votre travail, vous spécifiez les exigences suivantes à l'aide de Dynamic Workload Console ou de la commande **composer**. Lorsque vous sauvegardez l'ensemble d'exigences, un nouveau poste de travail est automatiquement créé dans la base de données IBM® Workload Scheduler. Ce poste de travail mappe tous les agents de votre environnement qui répondent aux exigences spécifiées. Le pool généré est mis à jour dynamiquement dès qu'un nouvel agent approprié devient disponible. Les travaux planifiés sur ce poste de travail héritent automatiquement des exigences définies pour le poste de

travail. Ce poste de travail est hébergé par le poste de travail Workload Broker et est enregistré dans la base de données IBM Workload Scheduler en tant que **d-pool** (pool dynamique).

Moteur distant

Poste de travail qui gère l'échange des informations sur la résolution des dépendances croisées entre votre environnement et un moteur IBM Workload Scheduler distant (contrôleur) ou un moteur IBM Workload Scheduler (gestionnaire de domaine maître ou gestionnaire de domaine maître de sauvegarde). Ce poste de travail est hébergé par le poste de travail Workload Broker et est enregistré dans la base de données IBM Workload Scheduler en tant que **rem-eng** (moteur distant).

Remarque : Si vous planifiez la modification des types de postes de travail, prenez en compte les règles suivantes :

- Vous pouvez modifier les postes de travail d'agent tolérant aux pannes, d'agent standard, d'agent étendu, de gestionnaire de domaine et de courtier de charge de travail dynamiques par tout type de poste de travail, à l'exception de l'agent dynamique, du pool, du pool dynamique et du moteur distant.
- Vous ne pouvez pas modifier le type de l'agent dynamique, du pool, du pool dynamique et du moteur distant.

Pour plus d'informations sur la définition des postes de travail, voir «Définition de poste de travail», à la page 166.

Classe de postes de travail

Les postes de travail peuvent être regroupés par classe. Une *classe de postes de travail* est un groupe de postes de travail ayant des caractéristiques de planification de travail similaires. Une classe peut contenir un nombre illimité de postes de travail et un poste de travail peut figurer dans plusieurs classes. Les travaux et flots de travaux peuvent être définis pour s'exécuter dans une classe de postes de travail spécifique. Cela permet de faciliter l'exécution de travaux et flots de travaux sur plusieurs postes de travail.

Par exemple, vous pouvez définir les types suivants de classes de postes de travail :

- Classes de postes de travail regroupant les postes selon votre structure de services interne qui vous permet de définir un travail qui s'exécutera sur tous les postes d'un même service
- Classes de postes de travail regroupant les postes selon les logiciels qui y sont installés et qui vous permettent de définir un travail qui s'exécutera sur tous les postes ayant une même application.
- Classes de postes de travail regroupant les postes selon le rôle de l'utilisateur associé, de sorte que vous puissiez définir un travail qui s'exécutera sur tous les postes appartenant, par exemple, aux responsables.

Dans cet exemple, un poste donné peut se trouver dans une classe définie pour son service, dans une autre spécifique à son utilisateur et dans diverses classes selon les logiciels qui y sont installés.

Les postes de travail peuvent être regroupés par domaine. Cela est fait lorsque votre réseau est configuré. Le nom de domaine n'est pas un critère de sélection pour le choix de l'emplacement d'exécution d'un travail, il est donc possible que vous deviez reproduire votre structure de domaines avec des classes de postes de travail si vous souhaitez planifier l'exécution d'un travail sur tous les postes d'un domaine.

Pour plus d'informations sur les domaines, voir «Domaine, zone»

Pour plus d'informations relatives à la définition des classes de poste de travail, voir «Définition de classe de postes de travail», à la page 184.

Domaine, zone

Tous les postes de travail d'un réseau IBM Workload Scheduler répartis sont organisés en un ou plusieurs *domaines*, chacun d'eux étant composé d'un ou de plusieurs agents et d'un gestionnaire de domaine faisant office de concentrateur de gestion. La plupart des communications à destination et en provenance des agents d'un domaine passe par le gestionnaire de domaine.

Tous les réseaux ont un domaine maître dans lequel le gestionnaire de domaine est le gestionnaire de domaine maître. Il gère la base de données de tous les objets de planification du domaine et les fichiers de configuration centraux. Le gestionnaire de domaine maître génère le plan et crée et distribue le fichier Symphony. En outre, il gère les journaux et les rapports du réseau.

Vous pouvez organiser tous les agents de votre réseau dans un seul domaine ou dans plusieurs domaines.

Réseaux à domaine unique

Un réseau à domaine unique comprend un gestionnaire de domaine maître et un certain nombre d'agents. Un réseau à domaine unique est présenté ci-après. Ce type de réseau convient parfaitement aux sociétés ayant un nombre limité de sites et de fonctions métier. Toutes les communications du réseau sont acheminées via le gestionnaire de domaine maître. Lorsque le réseau couvre un seul site, vous ne vous préoccupez que de la fiabilité de votre réseau local et de la quantité de trafic qu'il peut gérer.

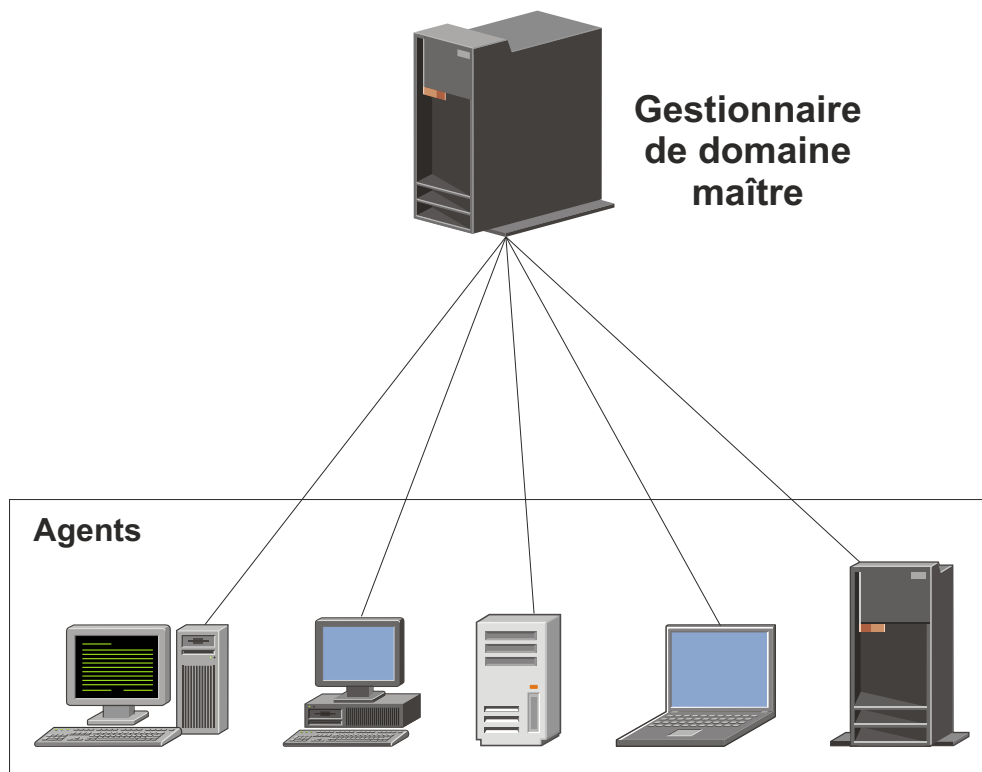


Figure 1. Réseau à domaine unique

Réseaux à domaines multiples

Les réseaux à domaines multiples conviennent particulièrement aux sociétés s'étendant sur plusieurs sites, régions ou fonctions métier. Un réseau à domaines multiples comprend un gestionnaire de domaine maître, un certain nombre de gestionnaires de domaine de niveau inférieur et divers agents dans chacun des domaines. Les agents communiquent uniquement avec leurs gestionnaires de domaine et les gestionnaires de domaine communiquent, quant à eux, avec leurs gestionnaires de domaine parent. La hiérarchie de domaine peut comporter un nombre indéfini de niveaux.

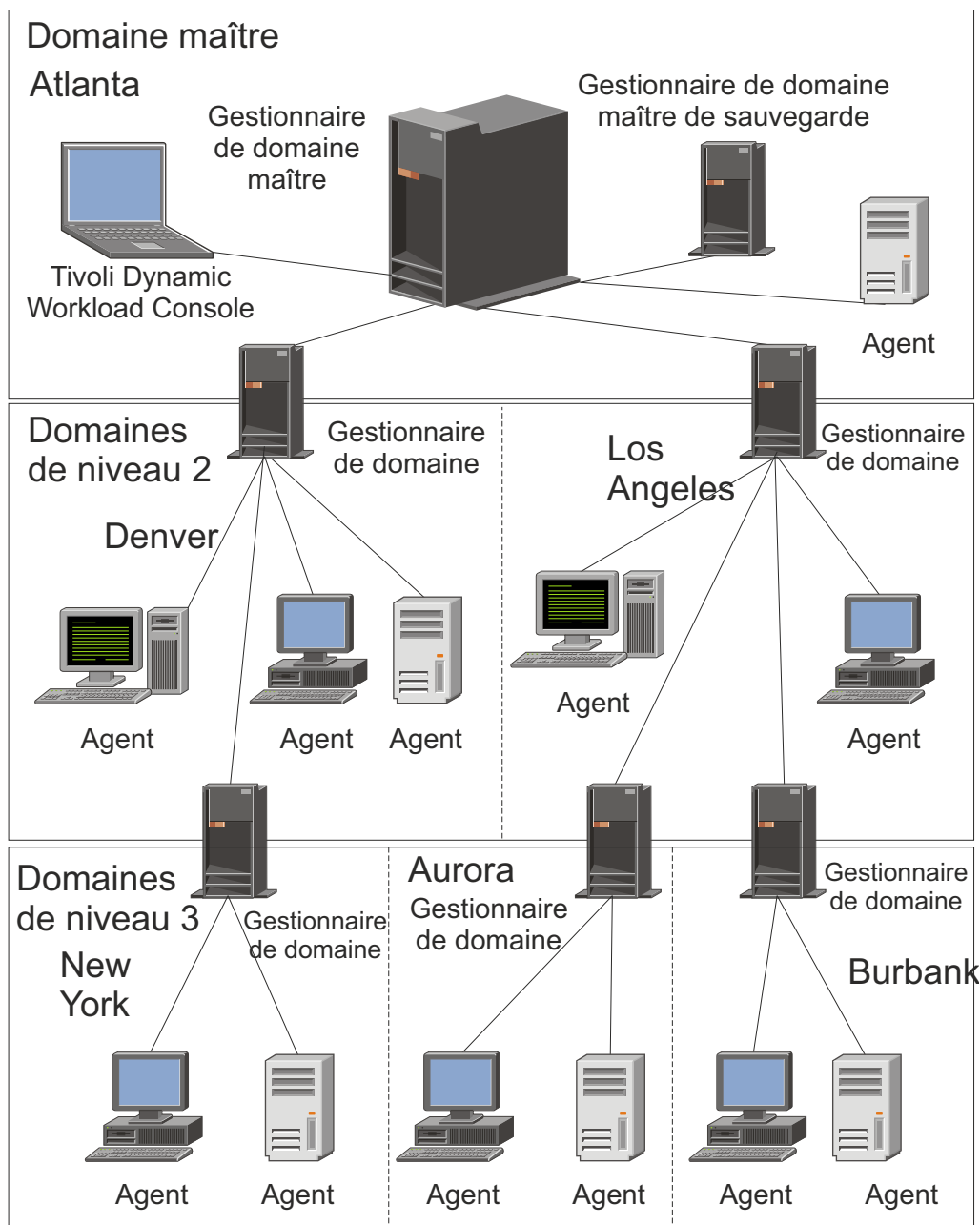


Figure 2. Réseau à domaines multiples

Dans cet exemple, le gestionnaire de domaine maître est situé à Atlanta. Il contient les fichiers de base de données utilisés pour documenter les objets de planification et assurer la distribution du fichier Symphony auprès de ses agents et gestionnaires de domaine à Denver et Los Angeles. Les gestionnaires de domaine de Denver et Los Angeles distribuent ensuite à leur tour le fichier Symphony auprès de leurs agents et gestionnaires de domaine subordonnés à New York, Aurora et Burbank. Le gestionnaire de domaine d'Atlanta est chargé de la diffusion des informations inter-domaines sur l'ensemble du réseau.

Toute communication à destination et en provenance du gestionnaire de domaine de Boulder est acheminée via son gestionnaire de domaine parent situé à Denver. Si des calendriers ou des travaux du domaine de Boulder

dépendent de calendriers ou de travaux du domaine d'Aurora, ces dépendances sont résolues par le gestionnaire de domaine de Denver. La plupart des dépendances interagents sont gérées en local par les gestionnaires de domaine de niveau inférieur, ce qui permet de réduire le trafic sur le réseau de manière notable.

Il est possible de modifier dynamiquement l'infrastructure de domaine à mesure que le réseau se développe. Le déplacement d'un poste de travail vers un autre domaine implique simplement de modifier le nom de domaine dans sa définition de base de données.

Tableau 5. Astuce

Conseil : Il n'est pas possible de planifier l'exécution de travaux ou de flots de travaux sur tous les postes de travail dans un domaine en identifiant ce dernier dans la définition de travail ou de flot de travaux. Pour ce faire, créez une *classe de postes de travail* contenant tous les postes de travail dans le domaine.

Pour plus d'informations relatives aux classes de poste de travail, voir «Classe de postes de travail», à la page 15

Pour plus d'informations sur la manière de définir des domaines, voir «Définition de domaine», à la page 185.

Règle d'événement

Une *règle d'événement* définit un ensemble d'actions qui s'exécutent lorsqu'il existe des conditions d'événement spécifiques. La définition d'une règle d'événement fait référence à des événements et des actions du déclencheur.

Pour plus d'informations sur la définition des règles d'événement, voir «Définition de règles d'événement», à la page 150.

Ressource

Une *ressource* est une ressource système physique ou logique que vous utilisez comme dépendance pour les travaux et flots de travaux. Un travail ou un flot de travaux avec une dépendance de ressource ne peut démarrer tant que la quantité de ressource spécifiée n'est pas disponible.

Pour plus d'informations sur la manière de définir des ressources, voir «Définition de ressource», à la page 215.

Paramètre

Un *paramètre* est un objet auquel vous affectez différentes valeurs qui seront remplacées dans les travaux ou flots de travaux, soit à partir de valeurs de la base de données soit au moment de l'exécution. Les paramètres sont utiles lorsque vous avez des valeurs qui changent en fonction de votre travail ou flot de travaux. Les définitions de travail ou de flot de travaux utilisant des paramètres sont automatiquement mises à jour avec la valeur appropriée au début du cycle de production. Utilisez les paramètres en remplacement des valeurs répétitives lors de la définition de travaux ou flots de travaux. Par exemple, si vous utilisez des paramètres pour les ouvertures de session utilisateur et les noms de fichiers script dans les définitions de travaux, ainsi que pour les dépendances de fichier et d'invite, vous utilisez des valeurs permettant une gestion centralisée dans la base de données du gestionnaire maître.

Pour plus d'informations relatives à la définition des paramètres, voir «Définition des variables et des paramètres», à la page 206.

User

Un *Utilisateur* est le nom d'utilisateur utilisé comme valeur de connexion pour plusieurs définitions de travail du système d'exploitation. Les utilisateurs doivent être définis dans la base de données.

Si vous planifiez un travail sur un agent, un pool ou un pool dynamique, le travail s'exécute avec l'utilisateur défini sur le pool ou pool dynamique. Cependant, l'utilisateur doit exister sur tous les postes de travail du pool ou du pool dynamique où vous planifiez d'exécuter le travail.

Remarque : Si l'option globale `enAddUser` vaut "yes", la définition d'utilisateur est automatiquement ajoutée au plan après la création ou la modification de la définition d'utilisateur dans la base de données.

Table de variables

Une *table de variables* est une table comportant plusieurs variables et leurs valeurs. Tous les paramètres globaux, désormais appelés *variables*, sont inclus dans au moins une table de variables.

Il n'est pas nécessaire de créer des tables de variables pour utiliser des variables. En effet, le planificateur fournit une table de variables par défaut.

Toutefois, il est possible que vous souhaitiez définir une variable portant le même nom mais dont les valeurs diffèrent, selon le moment et le contexte d'utilisation. Pour ce faire, vous affectez différentes valeurs à la même variable dans différentes tables de variables. Vous pouvez alors utiliser le même nom de variable dans différentes définitions de travail ou lorsque vous définissez des dépendances de fichier et d'invite. Les tables de variables peuvent être affectées au niveau des cycles d'exécution, des flots de travaux et des postes de travail.

Les tables de variables peuvent s'avérer particulièrement utiles pour les définitions de travail, notamment lorsque vous utilisez une définition de travail en tant que modèle pour un travail appartenant à plusieurs flots de travaux. Par exemple, vous pouvez affecter différentes valeurs à la même variable et réutiliser la même définition de travail dans plusieurs flots de travaux.

Pour plus d'informations sur la définition des tables de variables, voir «Définition de table de variables», à la page 211.

Réseau IBM Workload Scheduler

Un réseau IBM Workload Scheduler est composé d'un ensemble de *postes de travail* reliés entre eux sur lesquels vous exécutez un traitement des travaux par lots à l'aide des fonctionnalités de gestion IBM Workload Scheduler.

Les postes de travail communiquent via TCP/IP et une technologie de stockage avant transfert qui permet de préserver la cohérence et la tolérance aux pannes à travers le réseau. En d'autres termes, si un poste de travail n'est pas connecté, toutes les informations sont stockées dans le fichier de messages et ne sont envoyées qu'une fois la liaison rétablie.

Le réseau IBM Workload Scheduler comprend un ou plusieurs domaines, chacun d'eux comportant un poste de travail *gestionnaire de domaine* jouant le rôle de concentrateur de gestion et un ou plusieurs postes de travail *agents*.

Il existe quatre types d'agent : *standard*, *tolérant aux pannes*, *étendu* et *Workload Broker*. Les agents standard et tolérants aux pannes peuvent être définis sur des ordinateurs UNIX et Windows. Les agents étendus sont des définitions logiques hébergées chacune par un poste de travail physique et permettent de traiter les travaux lorsqu'aucun agent n'est installé. Par exemple, ils sont utilisables avec Peoplesoft, SAP R/3, z/OS, CA-7, JES, OPC, Oracle EBS et VMS mais vous pouvez également les installer sur des systèmes UNIX et Windows. Les agents Workload Broker sont des postes de travail qui gèrent le cycle de vie des travaux de type IBM Workload Scheduler Workload Broker dans Dynamic Workload Broker.

Un *poste de travail de moteur distant* est un autre type de poste de travail que vous pouvez définir dans votre réseau. Ce type de poste de travail permet de gérer la communication avec un moteur IBM Workload Scheduler distant, distribué ou basé sur z/OS, afin de gérer les dépendances des travaux locaux à partir des travaux définis sur le moteur distant. Pour plus d'informations, voir Chapitre 23, «Définition et gestion des dépendances croisées», à la page 845.

Pour plus d'informations sur les postes de travail, voir «Définition de poste de travail», à la page 166.

Dans la topologie hiérarchique de IBM Workload Scheduler, le *gestionnaire de domaine maître* est le gestionnaire de domaine du domaine supérieur. Toutes les tâches de configuration de la production et la génération du *plan de production* sont effectuées sur le gestionnaire de domaine maître. Un plan de production contient toutes les activités de gestion des travaux devant être effectuées à travers le réseau IBM Workload Scheduler au cours d'une période déterminée. Une copie du plan de production est distribuée à partir du gestionnaire de domaine maître aux autres postes de travail. Sur chaque poste de travail, IBM Workload Scheduler lance ses propres travaux, en assure le suivi et envoie le statut du traitement des travaux au gestionnaire de domaine maître.

Pour plus d'informations sur les fonctionnalités IBM Workload Scheduler de gestion du plan, voir Chapitre 4, «Gestion du cycle de production», à la page 67

Configuration de l'environnement d'exécution IBM Workload Scheduler

La présente section vous offre un aperçu évolué de la configuration possible de l'environnement d'exécution IBM Workload Scheduler.

Définition des propriétés de configuration

Vous pouvez définir deux types de propriété pour configurer votre environnement d'exécution IBM Workload Scheduler : les propriétés définies sur le gestionnaire de domaine maître qui concernent le traitement sur tous les postes de travail du réseau IBM Workload Scheduler et les propriétés définies localement sur un poste de travail qui concernent uniquement le traitement sur ce poste de travail. Les premières sont gérées à l'aide du programme de ligne de commande IBM Workload Scheduler intitulé **optman**, tandis que les deuxièmes sont définies en local sur le poste de travail par personnalisation des fichiers de configuration **useropts**, **localopts** et **jobmanrc**.

Pour plus d'informations sur l'utilisation de la ligne de commande **optman** pour gérer les options globales et sur les options locales définies dans le fichier **localopts**, voir *Guide d'administration*.

Pour plus d'informations sur les options locales définies dans le fichier `useropts`, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

Configuration de la sécurité

Chaque fois que vous exécutez un programme IBM Workload Scheduler ou que vous appelez une commande IBM Workload Scheduler, les informations de sécurité sont lues à partir d'un fichier spécial appelé *fichier de sécurité* pour déterminer vos droits d'utilisateur. Ce fichier contient une ou plusieurs *définitions d'utilisateur*. Une définition d'utilisateur est un groupe d'un ou plusieurs utilisateurs autorisés ou non à exécuter des actions spécifiques sur des types d'objet de planification spécifiques.

L'utilisateur IBM Workload Scheduler principal, *utilisateur_TWS*, est défini au moment de l'installation dans le fichier de sécurité. Il permet d'exécuter la procédure de configuration, de définir des propriétés et de gérer les définitions d'utilisateur dans le fichier de sécurité. Vous pouvez modifier le fichier de sécurité à tout moment de façon à répondre aux besoins de votre configuration système.

Pour plus d'informations sur la gestion des droits d'utilisateur, consultez le document *IBM Workload Scheduler - Guide d'administration*.

Définition d'activités de planification à l'aide de IBM Workload Scheduler

Pour exécuter des activités de planification à l'aide de IBM Workload Scheduler, vous devez d'abord définir l'environnement que vous souhaitez gérer en termes d'*objets de planification* et de règles à appliquer lors de la planification d'opérations à exécuter sur ces objets. Ces informations sont stockées par IBM Workload Scheduler dans une base de données relationnelle DB2 ou Oracle, que nous appellerons ultérieurement *base de données*.

Outre les définitions des objets de planification (les travaux, les flots de travaux, les ressources, les postes de travail, par exemple), la base de données contient également des statistiques relatives aux travaux et flots de travaux traités, ainsi que des informations relatives à l'utilisateur qui a créé un objet et à la dernière modification d'un objet. Vous pouvez gérer les définitions d'objets de planification dans la base de données soit à l'aide du programme de ligne de commande IBM Workload Scheduler nommé **composer**, soit à l'aide de l'interface graphique **Dynamic Workload Console**. Vous pouvez extraire de la base de données des statistiques ou des informations historiques concernant les travaux et les flots de travaux traités à l'aide :

- Des **utilitaires de rapport** de IBM Workload Scheduler à partir de la ligne de commande.
- De Dynamic Workload Console.
- Des vues de la base de données.

Pour plus d'informations relatives à la définition des objets de planification, voir Chapitre 9, «Définition d'objets dans la base de données», à la page 163.

Pour plus d'informations sur les commandes de l'utilitaire de génération d'états, voir Chapitre 18, «Extraction des états et des statistiques», à la page 755.

Pour de plus amples informations sur Dynamic Workload Console, reportez-vous à la documentation correspondante.

Pour plus d'informations sur les vues de base de données, consultez le document *Vues de la base de données*.

Contrôle du traitement des travaux et des flots de travaux

Vous pouvez contrôler le traitement des travaux et des flots de travaux en définissant une ou plusieurs règles à partir de ce qui suit :

Définition de dépendances

Une *dépendance* est un prérequis devant être respecté pour que le traitement se poursuive. Vous pouvez définir des dépendances pour les travaux et les flots de travaux afin de garantir leur traitement dans l'ordre approprié. Dans votre environnement de planification distribué IBM Workload Scheduler vous pouvez sélectionner les différents types de dépendance suivants :

- **Sur réussite des travaux et flots de travaux** : un travail ou un flot de travaux nommé *successeur* ne doit pas commencer tant que d'autres travaux et flots de travaux nommés *prédécesseurs* n'ont pas abouti. Pour plus d'informations, voir «follows», à la page 248.
- **Sur satisfaction de conditions spécifiques par des travaux et flots de travaux** : un travail ou un flot de travaux nommé *successeur* ne doit pas commencer tant que d'autres travaux et flots de travaux nommés *prédécesseurs* n'ont pas satisfait une, toutes ou un certain nombre de conditions spécifiques qui peuvent être liées au statut du travail ou du flot de travaux, au code retour, à des variables de sortie ou au contenu du journal de travail. Lorsque les conditions ne sont pas remplies par le prédécesseur, tous les travaux successeurs avec une dépendance conditionnelle associée sont placés à l'état **suppress**. Les travaux successeurs avec une dépendance standard sont évalués de la manière habituelle.

Vous pouvez également joindre ou agréger des dépendances conditionnelles associées à différents prédécesseurs dans une seule dépendance de jointure. Une jointure contient plusieurs dépendances, mais vous décidez lesquelles de ces dépendances doivent être satisfaites pour que la jointure soit considérée comme étant satisfaite. Vous pouvez définir un nombre illimité de dépendances conditionnelles, standard, ou les deux, dans une jointure. Assurez-vous que tous les composants de l'environnement IBM Workload Scheduler sont à la version 9.3, groupe de correctifs 1, ou version ultérieure. Ce type de dépendance n'est pas pris en charge sur l'Agent limité tolérant aux pannes IBM i. Pour plus d'informations, voir Chapitre 22, «Application d'une logique de branchement conditionnelle», à la page 827, «follows», à la page 248 et «Jointure», à la page 256.

- **Ressource** : un travail ou un flot de travaux a besoin d'une ou plusieurs ressources avant de pouvoir démarrer. Pour plus d'informations, voir «needs», à la page 265.
- **Fichier** : un travail ou un flot de travaux a besoin d'avoir accès à un ou plusieurs fichiers avant de pouvoir démarrer. Pour plus d'informations, voir «opens», à la page 274.
- **Invite** : un travail ou un flot de travaux doit attendre une réponse affirmative à une invite avant de pouvoir démarrer. Pour plus d'informations, voir «Définition d'invite», à la page 213 et «prompt», à la page 278.

Vous pouvez définir jusqu'à 40 dépendances pour un même travail ou flot de travaux. Si vous devez définir plus de 40 dépendances, vous pouvez les regrouper dans une dépendance de jointure. Dans ce cas, la jointure est utilisée simplement comme un conteneur de dépendances standard. Toutes les dépendances standard qu'il contient qui ne sont pas remplies sont traitées comme d'habitude et évitent que la dépendance de jointure soit considérée comme supprimée. Pour plus

=
=
=
=
=

= d'informations sur les dépendances de jointure, voir «Jointure ou combinaison de
= dépendances», à la page 831 et «Jointure», à la page 256.

Dans un réseau IBM Workload Scheduler, les dépendances peuvent se répercuter sur plusieurs postes de travail. Par exemple, vous pouvez rendre un travail job1, exécuté sur votre environnement local IBM Workload Scheduler site1, dépendant de l'aboutissement du travail job2, exécuté sur un environnement IBM Workload Scheduler distant site2. L'environnement de planification distant peut être soit des moteurs IBM Workload Scheduler for z/OS (contrôleur), soit d'autres moteurs IBM Workload Scheduler (gestionnaire de domaine maître). Deux types de dépendances implémentent cette exigence :

Dépendance interréseau

Il s'agit d'une implémentation simple et distribuée. Utilisez ce type de dépendance lorsque :

- L'environnement local IBM Workload Scheduler est distribué.
- Vous voulez rechercher une instance de travail prédécesseur distant uniquement dans le plan en cours d'exécution (plan de production) sur l'environnement distant.
- Vous avez besoin de faire correspondre une instance prédécesseur dans le plan distant et non *cette* instance de prédécesseur spécifique.
- Vous pouvez attendre que l'intervalle d'interrogation expire avant d'être mis à jour quant à la transition du statut du travail distant.
- Vous pouvez utiliser différentes syntaxes et configurations reposant sur l'environnement IBM Workload Scheduler distant distribué plutôt que sur z/OS.
- Vous acceptez d'utiliser un protocole de connexion propriétaire pour la communication avec le moteur distant.

Pour plus d'informations, voir Chapitre 21, «Gestion des dépendances interréseaux», à la page 817.

Dépendance croisée

Il s'agit d'une implémentation plus complète. Utilisez ce type de dépendance lorsque :

- Votre environnement local IBM Workload Scheduler peut être soit distribué, soit z/OS.
- Vous voulez également rechercher l'instance de prédécesseur distante parmi les instances planifiées qui ne sont pas encore incluses dans le plan en cours d'exécution sur le moteur distant.
- Vous recherchez la correspondance d'une instance de prédécesseur distante précise dans le plan du moteur distant. Pour ce faire, vous pouvez utiliser différents critères de recherche prêts à l'emploi.
- Vous souhaitez que votre dépendance soit mise à jour dès que l'instance de travail distant change de statut. Pour ce faire, le produit utilise des notifications asynchrones depuis le moteur distant sur le moteur local.
- Vous souhaitez utiliser les mêmes syntaxe et configuration, que l'environnement IBM Workload Scheduler local soit distribué ou z/OS.
- Vous voulez utiliser des connexions HTTP ou HTTPS pour la communication avec le moteur distant.

Pour plus d'informations, voir Chapitre 23, «Définition et gestion des dépendances croisées», à la page 845.

Définition des contraintes temporelles

Des *contraintes temporelles* peuvent être indiquées tant pour des travaux que pour des flots de travaux. Pour un cycle d'exécution spécifique, vous pouvez indiquer l'heure de début du traitement à l'aide du mot clé **at**, ou l'heure après laquelle le traitement ne démarre plus avec le mot clé **until**. En indiquant ces deux heures, vous définissez une période d'exécution d'un travail ou d'un flot de travaux. Les mots clés **at** et **until** correspondent à des dépendances temporelles.

En tant qu'alternative au mot clé **until**, vous pouvez spécifier le mot clé **jsuntil**. Ce dernier définit également la dernière heure de début d'un flot de travaux. Il détermine également le comportement des travaux dans le flot de travaux lorsque ce dernier approche de sa dernière heure de début. Utilisez **jsuntil** pour éviter que le flot de travaux soit supprimé, annulé ou défini sur continue (en fonction de l'action indiquée dans le mot clé **onuntil**) s'il commence avant sa dernière heure de début. Par exemple, si vous avez un flot de travaux avec **jsuntil** défini sur 10:00 et que l'un des travaux commence son exécution à 9:59, le travail et ses successeurs sont exécutés conformément à leur planification.

Il existe également une différence importante entre les mots clés **until** et **jsuntil** :

Si vous spécifiez le mot clé **until** dans votre définition de flot de travaux

Ce mot clé est évalué également après que le flot de travaux a été lancé. Par conséquent, si la dernière heure de début expire avant que le flot de travaux se termine avec succès, l'action indiquée dans le mot clé **onuntil** associé est effectuée sur le flot de travaux et ses travaux qui n'ont pas encore été lancés.

Si vous spécifiez le mot clé **jsuntil** dans votre définition de flot de travaux

Ce mot clé n'est évalué qu'une fois, dès que toutes les dépendances du flot de travaux sont satisfaites et que ce dernier passe à l'état READY. Si la dernière heure de début définie à l'aide du mot clé **jsuntil** n'a pas encore expiré, elle n'est plus évaluée et le flot de travaux est exécuté indépendamment d'elle. Toutefois, pour empêcher que le flot de travaux ne reste indéfiniment à l'état READY, deux jours après que l'heure spécifiée dans le mot clé **jsuntil** ait été atteinte, le flot de travaux est supprimé par défaut.

Il est également possible de spécifier le paramètre de temps **schedtime** pour désigner l'heure de référence lors du calcul des dépendances de travaux et de flots de travaux. Vous pouvez également spécifier une *fréquence de répétition* pour que IBM Workload Scheduler lance par exemple le même travail toutes les 30 minutes entre 8:30 et 13:30.

Vous pouvez également indiquer une **durée maximale** ou une **durée minimale** pour un travail défini dans un flot de travaux. Si un travail est en cours d'exécution et que la durée maximale a été dépassée, le travail peut être abandonné ou peut continuer à s'exécuter. Si un travail ne s'exécute pas assez longtemps pour atteindre la durée minimale spécifiée, le travail peut être défini sur l'état Fin anormale, sur l'état Confirmer en attendant la confirmation de l'utilisateur ou il peut continuer à s'exécuter.

Pour plus d'informations, voir «at», à la page 233, «deadline», à la page 237, «every (utilisé pour les travaux)», à la page 240, «schedtime», à la page 279, «until», à la page 287, «jsuntil», à la page 257, «maxdur», à la page 262 et «mindur», à la page 264.

Définition de la priorité d'un travail et de la priorité minimale d'un poste de travail

IBM Workload Scheduler possède son propre système de mise en file d'attente qui utilise des niveaux de *priorité*. L'attribution d'une priorité à des travaux ou des flots de travaux vous offre un contrôle accru sur leur priorité et leur ordre d'exécution.

La *priorité minimale de travail* permet d'exercer un autre type de contrôle sur le traitement de travaux sur un poste de travail. Lorsqu'elle est définie sur un niveau de priorité, seuls les travaux et les flots de travaux dont la priorité dépasse la priorité minimale de travail peuvent à s'exécuter sur ce poste de travail. Par exemple, si la priorité minimale est de 40, les travaux dont la priorité est inférieure ou égale à 40 ne pourront pas être lancés.

Pour plus d'informations, voir «fence», à la page 462 et «priority», à la page 277.

Définition de limites

La *limite* offre un moyen de définir le nombre maximal de travaux que IBM Workload Scheduler est autorisé à lancer. Vous pouvez fixer une limite :

- dans la définition du flot de travaux avec l'argument *job limit*,
- Dans la définition d'un poste de travail à l'aide de la commande *limit cpu*.

La définition d'une limite de 25 sur un poste de travail, par exemple, autorise IBM Workload Scheduler à limiter à 25 le nombre de travaux s'exécutant simultanément sur ce poste de travail.

Pour plus d'informations, voir «limit cpu», à la page 465 et «limit sched», à la page 467.

Définition de ressources

Vous pouvez définir des *ressources* afin de représenter des actifs physiques ou logiques sur votre système. Chaque ressource est représentée par un nom et un nombre d'unités disponibles. Si vous disposez de trois unités de bande par exemple, vous pouvez définir une ressource bandes avec trois unités disponibles. Un travail utilisant deux unités de la ressource bandes empêche alors le lancement d'autres travaux demandant plus d'une unité. Toutefois, étant donné qu'une ressource n'est pas liée à un actif de façon stricte, vous pouvez utiliser une ressource fictive en tant que dépendance pour contrôler le traitement des travaux.

Pour plus d'informations, voir «Définition de ressource», à la page 215.

Demande de confirmation d'un travail

Dans certains scénarios, l'état d'achèvement d'un travail ne peut pas être déterminé tant que vous n'avez pas effectué certaines tâches. Par exemple, vous pouvez décider de vérifier les résultats imprimés dans un rapport. Dans ce cas, vous pouvez déterminer dans la définition du travail que ce dernier nécessite une *confirmation* pour que IBM Workload Scheduler attende votre réponse avant de marquer le travail comme ayant abouti ou échoué.

Pour plus d'informations, voir «confirm», à la page 450.

Définition des actions de réexécution et de reprise des travaux

Vous avez le choix entre plusieurs options pour définir des actions de reprise pour vos travaux, lors de la création de la définition de travail dans la base de données et lors de la surveillance de l'exécution du travail dans le plan.

Lorsque vous créez une définition de travail sur la ligne de commande Composer ou dans le Concepteur de charge de travail, vous pouvez indiquer le type de reprise que IBM Workload Scheduler doit effectuer en cas d'échec du travail. Les options de reprise prédéfinies sont :

arrêter (stop)

En cas de fin anormale du travail, le traitement ne se poursuit pas avec le travail suivant.

Vous pouvez également arrêter la séquence de traitement après l'affichage d'une invite nécessitant une réponse de l'opérateur.

continuer (continue)

En cas de fin anormale du travail, le traitement se poursuit avec le travail suivant.

Vous pouvez également passer au travail suivant après l'affichage d'une invite nécessitant une réponse de l'opérateur.

réexécuter (rerun)

En cas de fin anormale du travail, le traitement est réexécuté.

Vous pouvez offrir plus de souplesse à l'option de réexécution en définissant une séquence de réexécution avec des propriétés spécifiques. Les options décrites ci-dessous s'excluent mutuellement.

repeatevery *hhmm* for *nombre* attempts

Vous pouvez indiquer la fréquence à laquelle vous souhaitez qu'IBM Workload Scheduler réexécute le travail ayant échoué et le nombre maximal de tentatives de réexécution à effectuer. Si une réexécution de la séquence aboutit, le reste de la séquence de réexécution est ignoré et les éventuelles dépendances de travail sont libérées.

réexécuter après invite (rerun after prompt)

IBM Workload Scheduler réexécute le travail ayant échoué après que l'opérateur a répondu à une invite.

same_workstation

Si le travail parent s'exécutait sur un poste de travail faisant partie d'un pool ou d'un pool dynamique, vous pouvez choisir s'il doit être réexécuté sur le même poste de travail ou sur un autre poste de travail. Cela est dû au fait que la charge de travail sur les pools et les pools dynamiques est affectée dynamiquement en fonction d'un certain nombre de critères et que le travail peut alors être réexécuté sur un autre poste de travail.

Vous pouvez également choisir de réexécuter le travail dans le plan IBM Workload Scheduler. Dans ce cas, vous avez le choix entre réexécuter le travail seul ou avec ses successeurs - soit avec tous les successeurs dans le même flot de travaux, soit avec tous les successeurs en général dans le même flot de travaux et dans d'autres flots de travaux éventuels. A partir de la ligne de commande **conman**, utilisez la commande «Listsucc», à la page 472 pour identifier les successeurs du travail, et la commande «Rerunsucc», à la page 484 pour les réexécuter.

L'option rerun est particulièrement utile pour la gestion des flots de travaux longs et complexes. Lorsqu'un travail se termine sur une erreur, vous pouvez le réexécuter avec tous ses successeurs. Vous pouvez aisément identifier ces derniers, dans le même flot de travaux ou dans des flots de travaux externes à partir des commandes **conman** «Listsucc», à la page 472

et «Rerunsucc», à la page 484 ou de Dynamic Workload Console. A partir de Dynamic Workload Console, vous pouvez aisément afficher la liste de tous les successeurs du travail avant de les réexécuter à partir de la vue **Surveiller la charge de travail** en sélectionnant le travail et en cliquant sur **Actions supplémentaires > Réexécuter avec les successeurs**. Vous pouvez également choisir d'exécuter tous les successeurs du travail parent ou uniquement ceux qui font partie du même flot de travaux que le travail parent. Pour gérer l'option rerun pour les successeurs du travail parent, voir «Rerunsucc», à la page 484 et «Listsucc», à la page 472.

travail de reprise (recovery job)

Si le travail se termine de manière anormale, exécutez un travail de reprise que vous avez défini précédemment et corrigez la condition d'erreur. Par exemple, vous savez qu'un travail pour lequel une connexion à la base de données est requise peut échouer si cette dernière n'est pas accessible. Pour corriger cette condition d'erreur, définissez un travail de reprise qui redémarre la base de données.

Vous pouvez combiner la séquence de réexécution avec le travail de reprise afin qu'en cas d'échec du travail parent un travail de reprise soit lancé. Lorsque le travail de reprise se termine avec succès, le travail parent est redémarré à l'issue de l'intervalle indiqué (le cas échéant), un nombre de fois défini, avec ou sans ses successeurs.

Par exemple, si vous définissez une séquence de réexécution dans laquelle un travail parent est associé à un travail de reprise et que le travail parent est planifié pour être réexécuté trois fois après un délai d'attente d'une minute (nécessaire à l'exécution du travail de reprise), la séquence de réexécution se déroule de la manière suivante :

1. Le travail parent s'exécute et se termine de manière anormale.
2. Le travail de reprise démarre et se termine avec succès.
3. Le travail parent respecte le délai d'attente indiqué après la fin du travail de reprise avant de redémarrer, puis redémarre.
4. Si le travail aboutit, le reste de la séquence de réexécution est ignoré et les éventuelles dépendances du travail sont libérées. Si le travail parent se termine à nouveau sur une erreur, les étapes 2 et 3 sont répétées trois fois, à moins qu'une seule réexécution suffise.
5. Si toutes les ré exécutions se terminent de manière anormale, le flot de travaux échoue ou reste à l'état STUCK.

Pour plus d'informations, voir «Définition de travail», à la page 187.

Gestion des activités de planification de production avec IBM Workload Scheduler

Chaque fois qu'un nouveau plan de production est généré, IBM Workload Scheduler sélectionne les flots de travaux qui s'exécutent pendant la période spécifiée pour le plan et reporte les flots de travaux non terminés à partir du plan de production précédent. Toutes les informations requises sont écrites dans un fichier nommé *Symphony* qui est continuellement mis à jour au cours du traitement pour indiquer le travail terminé, le travail en cours et le travail restant à effectuer. Le programme de ligne de commande **conman** (Console Manager) de IBM Workload Scheduler permet de gérer les informations du fichier Symphony. Le programme de ligne de commande **conman** peut être utilisé pour :

- démarrer et arrêter les processus de contrôle de IBM Workload Scheduler,

- afficher le statut des travaux et des flots de travaux,
- modifier les priorités et les dépendances,
- modifier la priorité minimale de travail et le nombre maximal de travaux,
- réexécuter des travaux,
- annuler des travaux et des flots de travaux,
- soumettre de nouveaux travaux et flots de travaux,
- répondre à des invites,
- connecter et déconnecter des postes de travail dans le réseau IBM Workload Scheduler,
- modifier le nombre de ressources disponibles.

A partir de la version 9.1, toutes les informations du plan écrites dans le fichier Symphony sont ensuite répliquées dans la base de données. Plusieurs opérations de surveillance demandées par Dynamic Workload Console accèdent à la base de données, plutôt qu'au fichier Symphony, ce qui entraîne des temps de réponse et des performances globales plus rapides. Les opérations suivantes demandées par Dynamic Workload Console accèdent aux informations à partir de la base de données :

- Surveillance des travaux et des flots de travaux
- Actualisation des vues de surveillance des travaux et flots de travaux
- Surveillance des postes de travail
- Surveillance des ressources, fichiers et invites
- Exécution des rapports de version de référence
- Affichage du plan dans une vue graphique
- Affichage du flot de travaux dans une vue graphique

=

Remarque : Cette fonction n'est pas prise en charge par le pilote JDBC DB2 de type = 2. IBM Workload Scheduler est fourni avec le pilote JDBC de type 4

Automatisation de la charge de travail grâce aux règles d'événement

Avant de procéder à la planification de travaux, vous pouvez automatiser la charge de travail sur la base de la demande à l'aide des règles d'événement. L'objet des règles d'événement est de réaliser un ensemble d'actions prédéfinies en réponse à des événements spécifiques affectant les objets IBM Workload Scheduler et non liés à IBM Workload Scheduler.

Du point de vue des objets IBM Workload Scheduler, le produit est doté d'un module d'extension capable de détecter les événements suivants :

- Pour un travail ou flot de travaux spécifique :
 - Modification de l'état
 - Dépassement de l'heure du dernier démarrage
 - Soumission
 - Annulation
 - Redémarrage
 - Retard
- Pour un poste de travail donné :
 - Modification de l'état
 - Modification de l'état du lien à partir du poste de travail parent
 - Modification de l'état du lien à partir du poste de travail enfant
- Une invite spécifique est affichée ou reçoit une réponse

- Le serveur d'applications sur un poste de travail donné est démarré ou arrêté

Lorsque l'un de ces événements se produit, l'une des actions suivantes peut être déclenchée :

- Soumission d'un flot de travaux, d'un travail ou d'une tâche
- Réponse à une invite
- Exécution de commandes non liées à IBM Workload Scheduler
- Connexion à un message opérateur
- Notification des utilisateurs par courriel
- Envoi de messages vers Tivoli Enterprise Console

Vous pouvez également définir et exécuter des règles d'événement chargées de détecter, selon les cas, un ou plusieurs de ces événements, ou bien une séquence ou un ensemble de ces événements dont l'exécution ne s'achève pas dans le délai imparti.

Pour plus d'informations, voir Chapitre 8, «Exécution de l'automatisation de la charge de travail gérée par événements», à la page 141.

Interfaces utilisateur IBM Workload Scheduler

Pour utiliser IBM Workload Scheduler, vous disposez d'une combinaison de programmes d'interface de ligne de commande, d'interface graphique et d'API. En particulier, l'interface de ligne de commande est proposée par certaines fonctions avancées indisponibles dans l'interface graphique. Les programmes d'interface utilisateur IBM Workload Scheduler disponibles sont les suivants :

Single Entry Point

Single Entry Point est une page Web permettant d'accéder à toutes les interfaces utilisateur d'IBM Workload Scheduler :

- Dynamic Workload Console
- Interfaces utilisateurs libre service (Catalogue libre-service et Tableaux de bord libre-service)
- Application Lab

Single Entry Point est une interface basée sur les rôles, à laquelle vous pouvez accéder de n'importe quel ordinateur de votre environnement à l'aide d'un navigateur Web et grâce à une connexion HTTPS sécurisée.

Single Entry Point fournit des liens rapides vers les tâches Dynamic Workload Console les plus importantes : connectez vos moteurs, concevez votre charge de travail, surveillez votre charge de travail et votre tableau de bord. Avec Single Entry Point, vous pouvez accéder aux applications mobiles, à Catalogue libre-service et à Tableaux de bord libre-service, grâce au lien ou à Qrcode. Pour accéder à Application Lab, vous devez ajouter des paramètres de configuration afin de définir votre moteur et votre ID environnement. L'ID environnement est défini par les administrateurs au moment de la configuration de l'accès à Application Lab dans le fichier de sécurité. Il est identifié par un préfixe à deux lettres.

Dynamic Workload Console

Une interface utilisateur Web disponible pour visualiser et contrôler les activités de planification dans la production sur les environnements IBM Workload Scheduler répartis et z/OS. Avec le Dynamic Workload Console vous pouvez utiliser tout navigateur supporté pour accéder à l'environnement IBM Workload Scheduler à partir de tout emplacement de votre réseau.

Vous pouvez utiliser la console Dynamic Workload pour :

- Définir des objets de planification dans la base de données IBM Workload Scheduler
- Visualiser et gérer les objets de planification impliqués dans les activités de planification en cours
- Créer et contrôler les connexions aux environnements IBM Workload Scheduler
- Soumettre des travaux et des flots de travaux en production
- Définir les préférences utilisateurs
- Créer et gérer des règles d'événement
- Définir et gérer des travaux critiques

Dynamic Workload Console doit être installé sur un serveur pouvant atteindre les noeuds IBM Workload Scheduler en utilisant les connexions réseaux. Pour plus d'informations, voir *IBM Workload Scheduler: Planification et installation*.

Application Lab

Une interface utilisateur Web est disponible pour la création, l'exécution et la surveillance d'un ensemble de processus métier simples. Elle se connecte à un moteur distribué Dynamic Workload Console existant partagé ou configuré en connexion unique. Cette interface facile d'emploi est dédiée aux développeurs d'applications qui ont besoin de contrôle et d'autonomie lors de la gestion des charges de travail, mais qui n'ont pas à gérer des concepts d'automatisation de charge de travail complexes. Elle utilise une terminologie simplifiée pour exprimer des concepts tels que les travaux, les flots de travaux et les cycles d'exécution. Les développeurs d'applications créent leurs *processus* constitués d'une ou plusieurs *étapes*. Les processus peuvent être planifiés pour s'exécuter (sur des agents dynamiques uniquement) selon les *déclencheurs* définis pour eux. Vous pouvez utiliser tout navigateur pris en charge pour accéder à l'environnement IBM Workload Scheduler à partir de tout emplacement de votre réseau via un protocole HTTPS sécurisé. Un administrateur doit autoriser des utilisateurs à accéder à Application Lab via la définition d'un environnement de travail.

Vous pouvez utiliser Application Lab pour :

- Définir un processus qui exécute un certain nombre d'étapes, lesquelles peuvent exécuter des opérations simples ou s'intégrer à une application externe pour effectuer des opérations plus complexes. Reportez-vous à la section de présentation dans *IBM Workload Automation : Application Lab - Guide d'utilisation* pour des informations plus détaillées concernant les types d'opération pris en charge.
- Importer des fichiers crontab existants ou des tâches du planificateurs de tâches Windows et bénéficier de l'ensemble complet des fonctions d'IBM Workload Scheduler.
- Configurer des processus qui s'exécutent en fonction d'un planificateur.
- Publier des processus dans le Catalogue libre-service et les soumettre pour exécution à la demande à partir d'un périphérique mobile.
- Déplacer vos processus vers Dynamic Workload Console pour une exécution avec des conditions de planification plus complexes.
- Surveiller la progression et le statut des processus.

Application Lab n'est pas un composant installable. Ce module est accessible via une adresse URL de connexion à un moteur Dynamic

Workload Console existant. Un administrateur doit effectuer certaines étapes préalables pour que les développeurs d'applications puissent accéder à Application Lab et l'utiliser. Pour plus d'informations, voir *IBM Workload Automation : Application Lab - Guide d'utilisation*.

composer

Ce programme de ligne de commande permet de définir et de gérer des objets de planification dans la base de données. Pour plus d'informations sur ce programme d'interface et son utilisation, voir Chapitre 9, «Définition d'objets dans la base de données», à la page 163 et Chapitre 10, «Gestion des objets dans la base de données - composer», à la page 321.

conman

Ce programme de ligne de commande permet de surveiller et de contrôler le traitement du plan de production IBM Workload Scheduler. Pour plus d'informations sur ce programme d'interface, voir Chapitre 12, «Gestion des objets dans le plan - conman», à la page 411.

API et plug-in Java™

Ensemble de classes et de méthodes disponibles s'exécutant dans un environnement JAVA que vous utilisez pour créer votre interface personnalisée afin de gérer les objets de planification dans la base de données et dans le plan. Cette API ne peut pas être utilisée pour créer votre interface personnalisée afin de définir des options globales. De plus, vous pouvez utiliser et modifier un ensemble de plug-in qui réalisent des tâches spécifiques, ou créer vos propres plug-in. Cette API est disponible via un kit de développement de logiciels qui fait partie du produit. Pour plus d'informations et pour apprendre comment accéder à la documentation de l'API et des plug-in, voir *IBM Workload Automation - Guide du développeur : Extension d'IBM Workload Automation*.

optman

Programme de ligne de commande permettant de gérer les paramètres qui affectent la globalité de l'environnement IBM Workload Scheduler. Ces paramètres, également appelés options globales, sont stockés dans la base de données. Pour plus d'informations sur ce programme d'interface, voir *IBM Workload Scheduler - Guide d'administration*.

planman

Ce programme de ligne de commande permet de gérer la fonctionnalité de planification de IBM Workload Scheduler. Pour plus d'informations sur ce programme d'interface, voir «Ligne de commande Planman», à la page 94.

Interface des services Web

Cette interface offre un mécanisme d'accès basé sur des services Web à un sous-ensemble de fonctionnalités servant à gérer les travaux et les flots de travaux dans le plan. Elle ne vous permet pas de gérer le plan, de définir des options globales ni de gérer des objets dans la base de données. Pour plus d'informations, voir *IBM Workload Automation - Guide du développeur : Gestion de Tivoli Workload Scheduler for z/OS*.

Vous devez installer la fonction IBM Workload Scheduler Command Line Client sur les agents et systèmes tolérants en panne à l'extérieur du réseau IBM Workload Scheduler pour utiliser les programmes de ligne de commande **composer** et **optman** et pour exécuter les commandes **planman showinfo** et **planman unlock**.

Pour plus d'informations sur la définition des options permettant d'autoriser un utilisateur à accéder aux interfaces de ligne de commande, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

Démarrage de la production

La présente section contient les procédures de base qui vous permettront d'implémenter rapidement IBM Workload Scheduler dans votre environnement à l'aide de l'interface de ligne de commande. Ces informations supposent que :

- Ces procédures sont effectuées sur le gestionnaire de domaine maître immédiatement après l'installation du produit sur les systèmes où vous souhaitez effectuer vos activités de planification.
- L'ID utilisateur utilisé pour effectuer les opérations est le même que celui qui a servi lors de l'installation du produit.

Si vous ne connaissez pas bien IBM Workload Scheduler, vous pouvez suivre les procédures de définition d'un nombre limité d'objets de planification et en ajouter d'autres à mesure que vous vous familiarisez avec le produit. Vous pouvez commencer, par exemple, par deux ou trois de vos applications les plus fréquemment utilisées et ne définir des objets de planification que pour répondre à leurs besoins.

Sinon, vous pouvez utiliser Dynamic Workload Console pour procéder à la modélisation et réaliser les tâches de fonctionnement. Pour plus d'informations, reportez-vous à la documentation du produit correspondant.

La première activité à effectuer consiste à accéder à la base de données IBM Workload Scheduler et à définir l'environnement dans lequel vous souhaitez effectuer vos activités de planification à l'aide des types d'objet de planification IBM Workload Scheduler. Pour ce faire, procédez comme suit :

1. Configuration des variables d'environnement IBM Workload Scheduler

Exécutez l'un des scripts suivants :

```
./racine_TWS/tws_env.sh pour les shells Bourne et Korn sous UNIX
./racine_TWS/tws_env.csh pour les shells C sous UNIX
racine_TWS\tws_env.cmd sous Windows
```

dans un interpréteur de commandes pour définir les variables *PATH* et *TWS_TISDIR*.

2. Connexion à la base de données IBM Workload Scheduler

Vous pouvez utiliser la syntaxe suivante pour vous connecter au gestionnaire de domaine maître avec l'ID *utilisateur_TWS* :

```
composer -user <utilisateur_TWS> -password <mot_de_passe_utilisateur_TWS>
```

où *utilisateur_TWS* est l'ID utilisateur que vous avez spécifié au moment de l'installation.

Remarque : Si vous souhaitez exécuter cette tâche et les suivantes à partir d'un système autre que le gestionnaire de domaine maître, vous devez indiquer les paramètres de connexion lors du lancement de **composer** (voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65).

3. Ajout éventuel à la base de données des définitions permettant de décrire la topologie de votre environnement de planification en termes de :

• Domaines

Exécutez cette tâche si vous souhaitez créer une arborescence hiérarchique du chemin dans votre environnement. L'utilisation de plusieurs domaines diminue le trafic réseau en réduisant les communications entre le

gestionnaire de domaine maître et les autres postes de travail. Pour plus d'informations, voir «Définition de domaine», à la page 185.

- **Poste de travail**

Définissez un poste de travail sur chaque machine appartenant à votre environnement de planification, à l'exception du gestionnaire de domaine maître, qui est automatiquement défini pendant l'installation du produit IBM Workload Scheduler. Pour plus d'informations, voir «Définition de poste de travail», à la page 166. Le gestionnaire de domaine maître est défini automatiquement dans la base de données au moment de l'installation.

4. **Définition éventuelle des utilisateurs autorisés à exécuter des travaux sur des postes de travail Windows**

Définissez les utilisateurs autorisés à exécuter des travaux à l'aide de IBM Workload Scheduler en spécifiant un nom d'utilisateur et un mot de passe. Pour plus d'informations, voir «Définition d'utilisateur», à la page 200.

5. **Définition éventuelle d'agendas**

Les agendas vous permettent de déterminer si un flot de travaux ou un travail doit être exécuté et à quel moment. Vous pouvez les utiliser pour inclure ou exclure des jours et des heures du traitement. Les agendas ne sont pas requis pour définir les jours de planification pour les flots de travaux (le cycle d'exécution simple ou *rule* peut être utilisé) ; l'objectif principal est de définir des ensembles de dates *global* pouvant être réutilisés dans plusieurs flots de travaux. Pour plus d'informations, voir «Définition d'agenda», à la page 205.

6. **Définition éventuelle de paramètres, d'invites et de ressources**

Pour plus d'informations, voir «Définition des variables et des paramètres», à la page 206, «Définition d'invite», à la page 213 et «Définition de ressource», à la page 215.

7. **Définition de travaux et de flots de travaux**

Pour plus d'informations, voir «Travail», à la page 940 et «Définition de flot de travaux», à la page 226.

8. **Définition éventuelle de restrictions et de paramètres pour contrôler l'heure d'exécution des travaux et des flots de travaux**

Vous pouvez définir des dépendances pour les travaux et les flots de travaux. Le programme accepte jusqu'à 40 dépendances pour un flot de travaux. Si vous devez définir plus de 40 dépendances, vous pouvez les regrouper dans une dépendance de jointure. Dans ce cas, la jointure est utilisée simplement comme un conteneur de dépendances standard. Toutes les dépendances standard qu'il contient qui ne sont pas remplies sont traitées comme d'habitude et évitent que la dépendance de jointure soit considérée comme supprimée. Pour plus d'informations sur les dépendances de jointure, voir «Jointure ou combinaison de dépendances», à la page 831 et «Jointure», à la page 256. Il peut s'agir de :

- Dépendances de ressource
- Dépendances de fichier
- Dépendances de travail et de flot de travaux, sur la réussite des travaux et des flots de travaux et sur la satisfaction des conditions spécifiques par les travaux et les flots de travaux
- Dépendances d'invite

Vous pouvez définir des paramètres temporels pour que les travaux et les flots de travaux s'exécutent en termes de :

- Cycles d'exécution

- Contraintes temporelles

Vous pouvez adapter l'exécution simultanée des travaux sur un poste de travail ou dans un flot de travaux en définissant les deux paramètres suivants :

- Limite
- Priorité

9. Extension automatique du plan à la fin du terme de production courant

Ajoutez le flot de travaux final à la base de données pour étendre automatiquement le plan de production à l'issue de chaque terme de production courant en exécutant la commande suivante :

```
add Sfinal
```

Pour plus d'informations, voir «Automatisation du traitement du plan de production», à la page 114.

10. Génération du plan

Exécutez la commande **JnextPlan** pour générer le plan de production. La commande **JnextPlan** permet de lancer le traitement des informations de planification stockées dans la base de données et de créer le plan de production pour la période spécifiée. La période par défaut est 24 heures. Si vous avez automatisé la génération du plan (voir étape précédente), vous devez seulement exécuter la commande **JnextPlan** la première fois.

Après cette procédure, votre environnement de planification est prêt à fonctionner. Il traite par lots une séquence ordonnée de travaux et de flots de travaux par rapport à des ressources définies sur un groupe de postes de travail, le cas échéant. Par défaut, le nombre de travaux pouvant être exécutés simultanément sur un poste de travail est nul la première fois que vous exécutez la commande **JnextPlan**. Par conséquent, vous devez augmenter cette valeur en modifiant la valeur du mot clé `limit cpu` de façon à autoriser l'exécution des travaux sur ce poste de travail (voir «limit cpu», à la page 465).

Si vous souhaitez apporter une modification quelconque alors que le plan de production est déjà en cours de traitement, utilisez le programme **conman**. Pendant le traitement du plan de production sur le réseau, vous pouvez continuer à définir ou à modifier des travaux et des flots de travaux dans la base de données. Cependant, ces modifications ne seront utilisées que si vous soumettez les travaux (commande **sbj**) ou les flots de travaux (commande **sbs**) sur un poste de travail qui a déjà reçu le plan, ou après la génération d'un nouveau plan de production via **JnextPlan**. Pour plus d'informations sur le programme **conman** et sur les opérations réalisables sur le plan de production en cours, voir Chapitre 12, «Gestion des objets dans le plan - conman», à la page 411.

Chapitre 2. Description des processus et des commandes de base

Dans un réseau multiniveau IBM Workload Scheduler, un groupe de processus de planification spécialisés assure en local sur chaque poste de travail la gestion des travaux et renvoie les informations sur le traitement des travaux dans l'arborescence hiérarchique jusqu'au gestionnaire de domaine maître. A l'aide des informations reçues des postes de travail, le gestionnaire de domaine maître met alors à jour sa copie du fichier Symphony et le plan répliqué dans la base de données qui contiennent les enregistrements décrivant les activités de traitement des travaux à effectuer sur le réseau IBM Workload Scheduler pendant le plan de production courant. Il renvoie ensuite aux postes de travail concernés les mises à jour portant sur les activités à effectuer.

Exécution de commandes sous Windows

Sous Windows, assurez-vous d'exécuter les commandes IBM Workload Scheduler à partir d'une invite de commande dotée du niveau de privilège **Exécuter en tant qu'administrateur**.

Processus de poste de travail IBM Workload Scheduler

La gestion des communications entre les postes de travail, le traitement local des travaux et la notification des mises à jour d'état sont réalisés sur chaque poste de travail IBM Workload Scheduler par une série de processus de gestion qui sont actifs pendant que le moteur s'exécute. Sur les agent tolérant aux pannes et les gestionnaires de domaine, ces processus s'appuient sur l'infrastructure WebSphere Application Server. Cette infrastructure est automatiquement installée avec le poste de travail et permet à IBM Workload Scheduler de :

- Communiquer sur le réseau IBM Workload Scheduler.
- Gérer les mécanismes d'authentification pour les clients distants, comme les programmes de ligne de commande, qui se connectent au gestionnaire de domaine maître à l'aide des protocoles HTTP ou HTTPS.

Pour plus d'informations sur le démarrage et l'arrêt de l'infrastructure WebSphere Application Server et des processus IBM Workload Scheduler sur un poste de travail, voir «Démarrage et arrêt de processus sur un poste de travail», à la page 43. En dehors du démarrage et de l'arrêt de WebSphere Application Server et de la gestion des paramètres de connexion lors de la communication sur le réseau IBM Workload Scheduler, l'infrastructure de WebSphere Application Server est transparente lors de l'utilisation de IBM Workload Scheduler.

Dans ce guide, les *processus IBM Workload Scheduler* ou les *processus de poste de travail* permettent d'identifier les processus suivants :

netman
monman
writer
mailman
batchman
jobman

A l'exception des agents standard, ces processus sont lancés sur les postes de travail IBM Workload Scheduler selon l'ordre suivant :

netman

Netman est le processus de gestion du réseau. Il est lancé par la commande **Startup** et se comporte comme un programme d'écoute du réseau, dont il reçoit des requêtes start, stop, link ou unlink. **Netman** examine chaque demande entrante et génère un processus IBM Workload Scheduler local.

monman

Le processus **Monman** est démarré par **netman** et est utilisé dans le cadre de la gestion d'événements. Il permet de démarrer la surveillance et les services de **ssmagent** ayant pour rôle de détecter les services qui sont définis dans les règles déployées et activées sur le poste de travail concerné. Lorsque ces services interceptent les événements, ils les envoient, après une action de filtrage provisoire, vers le serveur de traitement d'événements généralement exécuté sur le gestionnaire de domaine maître. Si aucune configuration de règle d'événement n'est téléchargée sur le poste de travail, les services de surveillance demeurent à l'état de veille.

Les services **Ssmagent** ne sont utilisés que pour les types d'événement de surveillance de fichiers. Pour plus d'informations, voir le paragraphe *"Événements de surveillance de fichier"* dans la section *"Gestion des événements"* du chapitre *"IBM Workload Scheduler Concepts"* du document *Dynamic Workload Console - Guide d'utilisation*.

Le processus de communication entre les agents de surveillance et le serveur de traitement d'événements est indépendant de la topologie du réseau IBM Workload Scheduler. Il repose directement sur le numéro de port EIF du processeur d'événements et les informations sur les événements transitent directement à partir des agents de surveillance, sans nécessiter de passage par des gestionnaires de domaine intermédiaires. A certain de degré de tolérance aux pannes est garanti par des mémoires locales qui stockent temporairement les occurrences d'événements sur les agents en cas de rupture des communications avec le processeur d'événements.

writer **Writer** est un processus lancé par **netman** pour transférer les messages entrants au processus **mailman** local. Les processus **writer** (il peut y en avoir plusieurs sur un poste de travail gestionnaire de domaine) sont démarrés par les demande de connexion (voir «link», à la page 468) et sont arrêtés par les demandes de connexion (voir «unlink», à la page 565) ou lorsque le processus **mailman** en cours de communication se termine.

mailman

Mailman est le processus de gestion du courrier (Mail Management). Il achemine les messages vers les postes de travail locaux ou distants. Sur un gestionnaire de domaine, des processus **mailman** supplémentaires peuvent être créés pour diviser la charge de travail de **mailman** due à l'initialisation des agents et pour améliorer le caractère opportun des messages. Lorsque le gestionnaire de domaine démarre, il crée une instance distincte du processus **mailman** pour chaque *ID serveur* spécifié dans les définitions de poste de travail de l'agent tolérant aux pannes et des agents standard qu'il gère. Chaque poste de travail est contacté avec son propre *ID serveur* sur le gestionnaire de domaine. Pour plus d'informations, voir «Définition de poste de travail», à la page 166.

batchman

Batchman est le processus de contrôle de production. Il interagit directement avec la copie du fichier Symphony distribuée aux postes de travail au début de la période de production et la met à jour. **Batchman** assure plusieurs fonctions :

- Gestion locale du traitement et de la mise à jour du plan.
- Résolution des dépendances des travaux et des flots de travaux.
- Sélection des travaux à exécuter.
- Mise à jour du plan avec les résultats du traitement des travaux.

Batchman est le seul processus habilité à mettre à jour le fichier Symphony.

jobman

Jobman est le processus de gestion des travaux. Il lance les travaux sous le contrôle de **batchman** et transmet le statut des travaux au processus **mailman**. Il est chargé de suivre l'état des travaux et de définir l'environnement conformément aux scripts `jobmanrc` et `.jobmanrc` lors des demandes de lancement de travaux. Pour plus d'informations sur ces scripts, voir Chapitre 3, «Configuration de l'environnement de travail», à la page 53. Lorsque le processus **jobman** reçoit un message de lancement de travail de **batchman**, il génère un processus de surveillance des travaux. Le nombre maximal de processus de surveillance des travaux pouvant être générés sur un poste de travail est déterminé à l'aide de la commande **limit cpu** à partir de l'invite de ligne de commande **conman** (voir «limit cpu», à la page 465).

moniteur de travaux (jobman sous UNIX, JOBMON.exe et joblnch.exe sous Windows)

Le processus de surveillance des travaux exécute d'abord un ensemble d'actions qui définissent l'environnement avant que le travail ne soit lancé, puis il lance le travail en exécutant la commande ou le fichier script spécifié dans la définition du travail. Pour savoir comment spécifier la commande ou le fichier script lancé avec le travail, voir «Travail», à la page 940.

Les activités de configuration se composent du lancement du fichier de configuration standard (*TWS_home/jobmanrc* sous UNIX et *TWS_home/jobmanrc.cmd* sous Windows) qui contient les paramètres applicables à tous les travaux s'exécutant sur le poste de travail. De plus, sur les postes de travail UNIX un script de configuration local *utilisateur_TWS/.jobmanrc* est lancé, s'il existe dans le répertoire d'accueil de l'utilisateur lançant de travail. Ce fichier de configuration local contient des paramètres qui s'appliquent uniquement aux travaux lancés par cet utilisateur spécifique. En cas d'échec de l'une de ces étapes, le travail se termine à l'état FAIL.

Avertissement : Si, sous Windows, une variable système nommé TEMP existe, l'utilisateur *utilisateur_TWS* doit être autorisé à créer des fichiers dans le répertoire pour lequel la variable est définie. Si cette exigence n'est pas remplie, le fichier binaire **JOBMON.exe** ne réussit pas à démarrer correctement.

Tous les processus, à l'exception de **jobman**, s'exécutent en tant qu'*utilisateur_TWS*. **Jobman** s'exécute en tant que root.

Sur les postes de travail d'agent standard, le processus **batchman** n'est pas lancé car ce type de poste de travail ne gère pas la planification des travaux. Ces postes de travail lancent uniquement des travaux sous la direction de leur gestionnaire de domaine. Localement, sur le poste de travail, les processus de gestion attendent de recevoir une demande de lancement d'un travail émise par le gestionnaire de domaine en mode écoute. Une fois la demande reçue, le travail est lancé localement et le résultat est renvoyé au gestionnaire de domaine. Pour plus d'informations relatives aux postes de travail de l'agent standard, voir *IBM Workload Scheduler - Guide de planification et d'installation*.

La figure 3, à la page 41 affiche l'arborescence du processus sur les postes de travail IBM Workload Scheduler autres que les agents standard et installés sous UNIX :

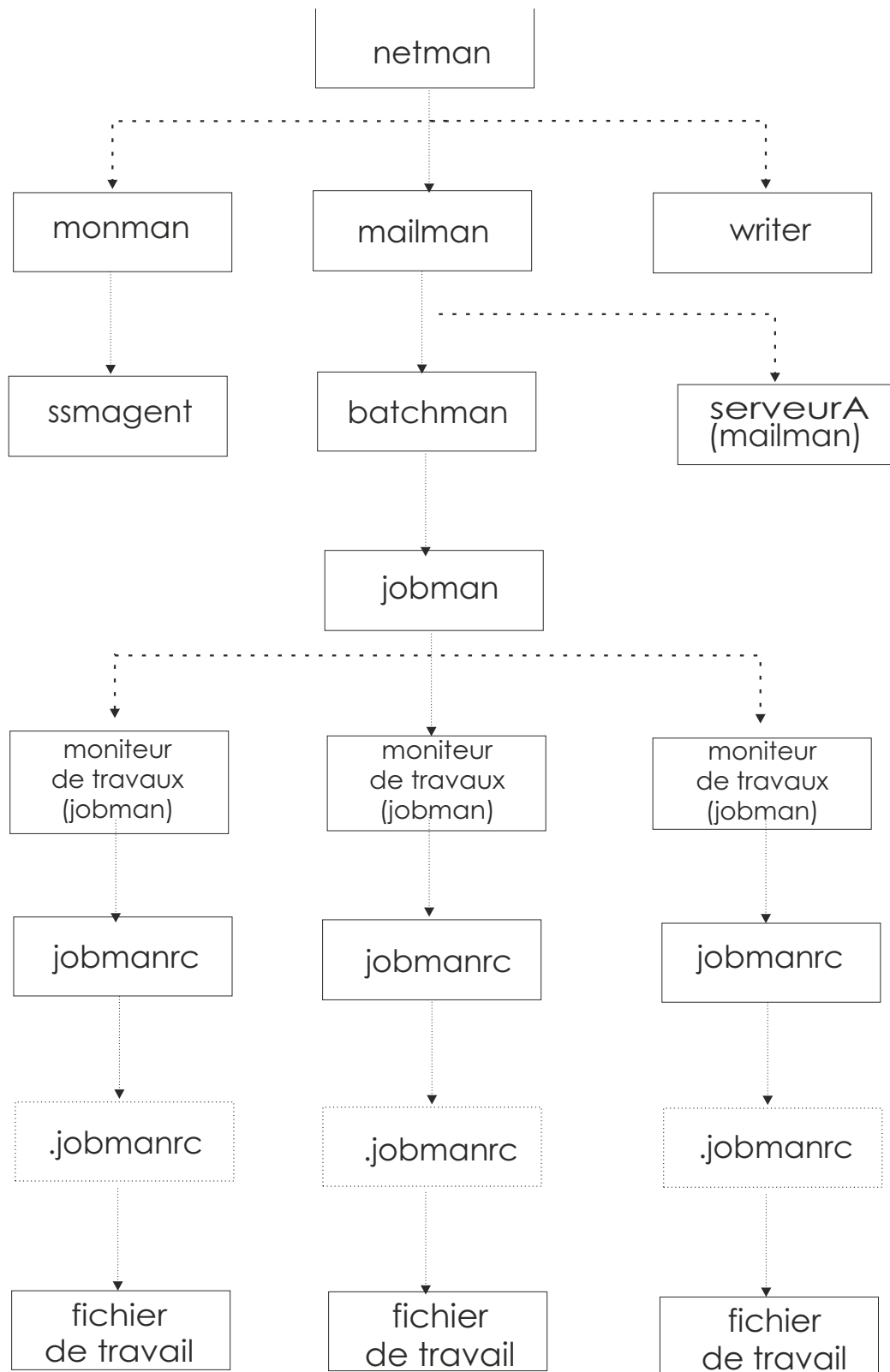


Figure 3. Arborescence des processus sous UNIX

La figure 4, à la page 42 affiche l'arborescence du processus sur les postes de travail IBM Workload Scheduler autres que les agents standard et installés sous

Windows :

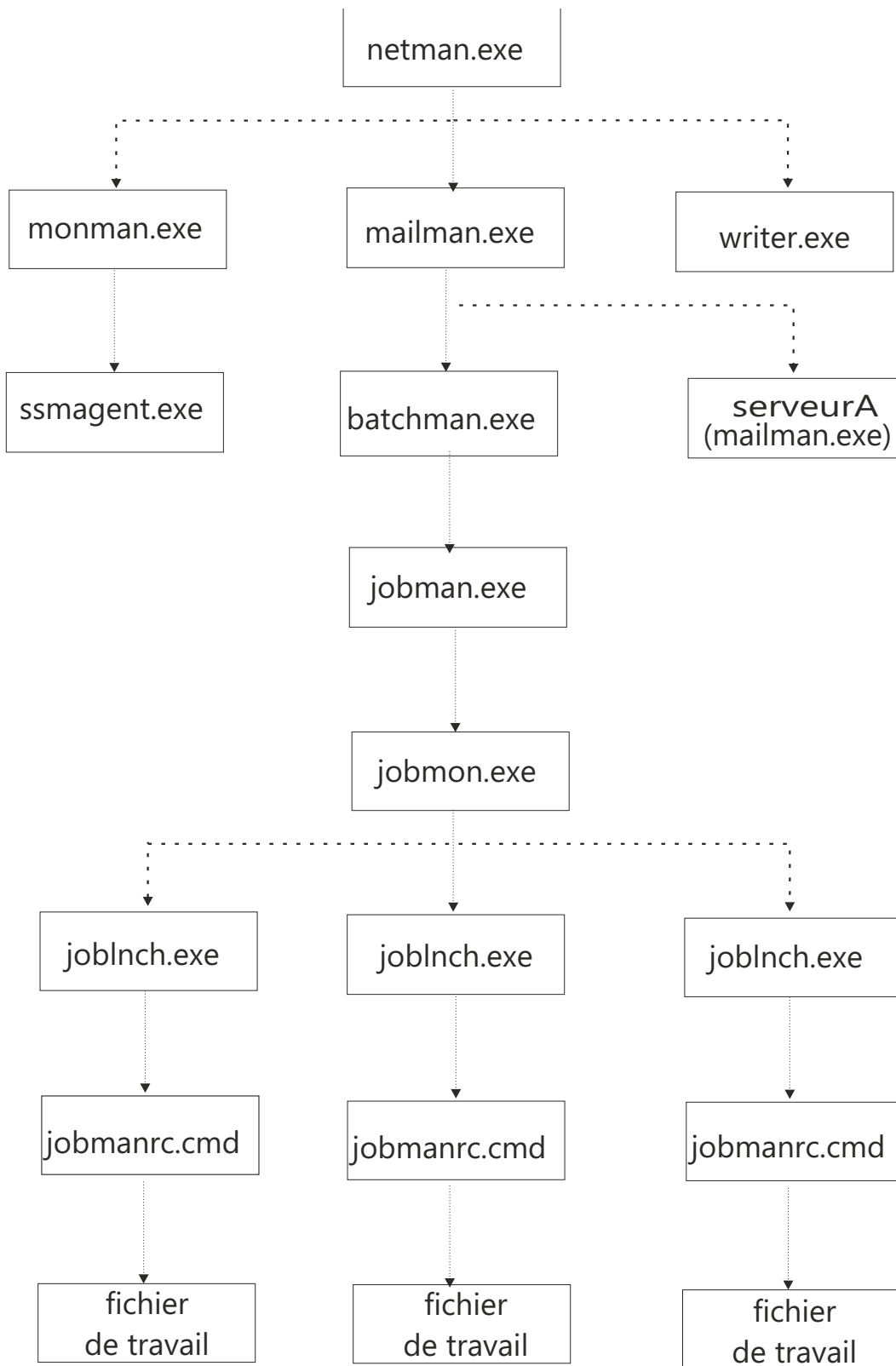


Figure 4. Arborescence des processus sous Windows

Sur les plateformes Windows, il existe un service supplémentaire, Tivoli Token Service, qui permet aux processus IBM Workload Scheduler d'être lancés comme s'ils étaient émis par l'utilisateur IBM Workload Scheduler.

Démarrage et arrêt de processus sur un poste de travail

Le type de système d'exploitation installé sur le poste de travail détermine la manière dont les processus IBM Workload Scheduler peuvent être lancés depuis la ligne de commande. Le tableau 6 explique comment démarrer et arrêter l'infrastructure WebSphere Application Server et les processus IBM Workload Scheduler sur un poste de travail en fonction du système d'exploitation.

Tableau 6. Démarrage et arrêt de IBM Workload Scheduler sur un poste de travail

Action	Commandes utilisées sur la plateforme UNIX	Commandes utilisées sur la plateforme Windows
Démarre tous les processus IBM Workload Scheduler dont WebSphere Application Server et le moteur de surveillance d'événements.	conman start conman startappserver conman startmon	conman start conman startappserver conman startmon
Démarre netman et WebSphere Application Server. Sous Windows démarre également Tivoli Token Service	./StartUp.sh	StartUp
Arrête tous les processus IBM Workload Scheduler et WebSphere Application Server.	conman shutdown ./stopWas.sh	conman shutdown -appsrv shutdown -appsrv
Arrête tous les processus IBM Workload Scheduler, à l'exception de WebSphere Application Server.	conman shutdown	conman shutdown shutdown
Démarre tous les processus IBM Workload Scheduler, à l'exception de WebSphere Application Server et du moteur de surveillance d'événements.	conman start	conman start
Arrête tous les processus IBM Workload Scheduler, à l'exception de netman , monman , writer et appservman .	conman stop	conman stop
Arrête tous les processus IBM Workload Scheduler (y compris netman).	conman shutdown	conman shutdown shutdown
Démarre WebSphere Application Server	./startWas.sh ou conman startappserver	startWas ou conman startappserver
Arrête WebSphere Application Server	./stopWas.sh ou conman stopappserver	stopWas ou conman stopappserver

Tableau 6. Démarrage et arrêt de IBM Workload Scheduler sur un poste de travail (suite)

Action	Commandes utilisées sur la plateforme UNIX	Commandes utilisées sur la plateforme Windows
Démarrer le moteur de surveillance d'événements.	conman startmon	conman startmon
Arrêter le moteur de surveillance d'événements	conman stopmon	conman stopmon
Démarrer l'agent localement	./StartUpLwa.sh Remarque : peut être exécuté par l'utilisateur_TWS ou par le superutilisateur uniquement.	startuplwa Remarque : Sous Windows 2008, doit être exécuté par un administrateur.
Arrêter l'agent localement	./ShutDownLwa.sh Remarque : peut être exécuté par l'utilisateur_TWS ou par le superutilisateur uniquement.	shutdownlwa Remarque : Sous Windows 2008, doit être exécuté par un administrateur.

Remarque : Sous Windows, les systèmes s'abstiennent d'utiliser les services Windows pour arrêter WebSphere Application Server. Utilisez à la place l'une des commandes figurant dans ce tableau. Si vous utilisez les services Windows pour arrêter WebSphere Application Server, le processus *appserverman*, dont l'exécution se poursuit, le redémarre.

Pour plus d'informations sur la commande d'utilitaire **StartUp**, voir «StartUp», à la page 722.

Pour plus d'informations sur la commande d'utilitaire **shutdown**, voir «shutdown», à la page 721.

Voir *IBM Workload Scheduler - Guide d'administration* pour plus d'informations sur les commandes **startWas** et **stopWas**.

Pour plus d'informations sur la commande **conman start**, voir «start», à la page 532.

Pour plus d'informations sur la commande **conman stop**, voir «stop», à la page 538.

Pour plus d'informations sur la commande **conman shutdown**, voir «shutdown», à la page 531.

Voir «startappserver», à la page 534 pour plus d'informations sur la commande **conman startappserver**.

Voir «stopappserver», à la page 541 pour plus d'informations sur la commande **conman stopappserver**.

Voir «startmon», à la page 536 pour plus d'informations sur la commande **conman startmon**.

Voir «stopmon», à la page 544 pour plus d'informations sur la commande **conman stopmon**.

Si l'agent est installé sur un système Windows, WebSphere Application Server et les processus **netman** sont automatiquement lancés au moment du démarrage en tant que services avec Tivoli Token Service. Si l'agent est installé sur un système UNIX, WebSphere Application Server et les processus **netman** peuvent être automatiquement lancés au démarrage en ajoutant une instruction appelant **Startup** dans le fichier `/etc/inittab`.

Démarrage et arrêt de la agent

Le type de système d'exploitation installé sur le poste de travail détermine comment les agents peuvent être démarrés à partir de la ligne de commande.

Tableau 7. Démarrage et arrêt de la agent

Action	Commandes utilisées sur la plateforme UNIX	Commandes utilisées sur la plateforme Windows
Démarrer l'agent localement	<code>./StartUpLwa.sh</code> Remarque : peut être exécuté par l'utilisateur_TWS ou par le superutilisateur uniquement.	<code>startuplwa</code> Remarque : Sous Windows 2008, doit être exécuté par un administrateur.
Arrêter l'agent localement	<code>./ShutDownLwa.sh</code> Remarque : peut être exécuté par l'utilisateur_TWS ou par le superutilisateur uniquement.	<code>shutdownlwa</code> Remarque : Sous Windows 2008, doit être exécuté par un administrateur.

Pour plus d'informations relatives à l'arrêt et au démarrage d'un agent, voir ShutDownLwa et StartUpLwa.

Communication interprocessus du poste de travail

IBM Workload Scheduler utilise les files d'attente de messages pour les communications interprocessus locales. Il existe 11 fichiers de messages ; ils résident dans le répertoire `rép_base_TWS` :

Appserverbox.msg

Ce fichier de messages est écrit par les processus **conman** et **mailman** et lu par le processus **appservman**. Il reçoit les messages de type START et STOP de WebSphere Application Server.

auditbox.msg

Ce fichier de messages est écrit par le processus **mailman** et lu par le processus WebSphere Application Server. Il reçoit des messages d'audit à stocker dans la base de données.

Courier.msg

Ce fichier de messages est écrit par le processus **batchman** et lu par le processus **jobman**.

Intercom.msg

Ce fichier de messages est lu par le processus **batchman** et il contient les instructions envoyées par le processus **mailman** local.

Mailbox.msg

Ce fichier de messages est lu par le processus **mailman**. Au moyen de l'interface utilisateur graphique (Dynamic Workload Console) ou du gestionnaire de console (**conman**), il reçoit les messages entrants des processus **batchman** et **jobman** locaux ainsi que d'autres postes de travail IBM Workload Scheduler du réseau.

mirrorbox.msg

Ce fichier de messages est écrit par le processus **mailman** et lu par le processus WebSphere Application Server. Il reçoit des messages entrants **batchman**.

Monbox.msg

Ce fichier de messages est écrit par les processus **mailman**, **batchman** et **appservman** et lu par le processus **monman**. Il reçoit les messages de type QUEUE_EVENT.

Moncmd.msg

Ce fichier de messages est écrit par les processus **conman**, **batchman** et **mailman** et par Dynamic Workload Console, et lu par le processus **monman**. Il reçoit les messages de type STOP, DEPLOY_CONFIG, UPGRADE_WORKSTATION.

NetReq.msg

Ce fichier de messages est lu par le processus **netman** pour les commandes locales. Il reçoit les messages de type START, STOP, LINK et UNLINK (demandes de démarrage, d'arrêt, de connexion et de déconnexion).

PlanBox.msg

Ce fichier de messages est écrit par le processus **batchman** et est lu par le moteur.

Server.msg

Ce fichier de messages est écrit par le processus **batchman** et est lu par le moteur.

Cette figure illustre la communication inter-processus sur le gestionnaire de domaine maître.

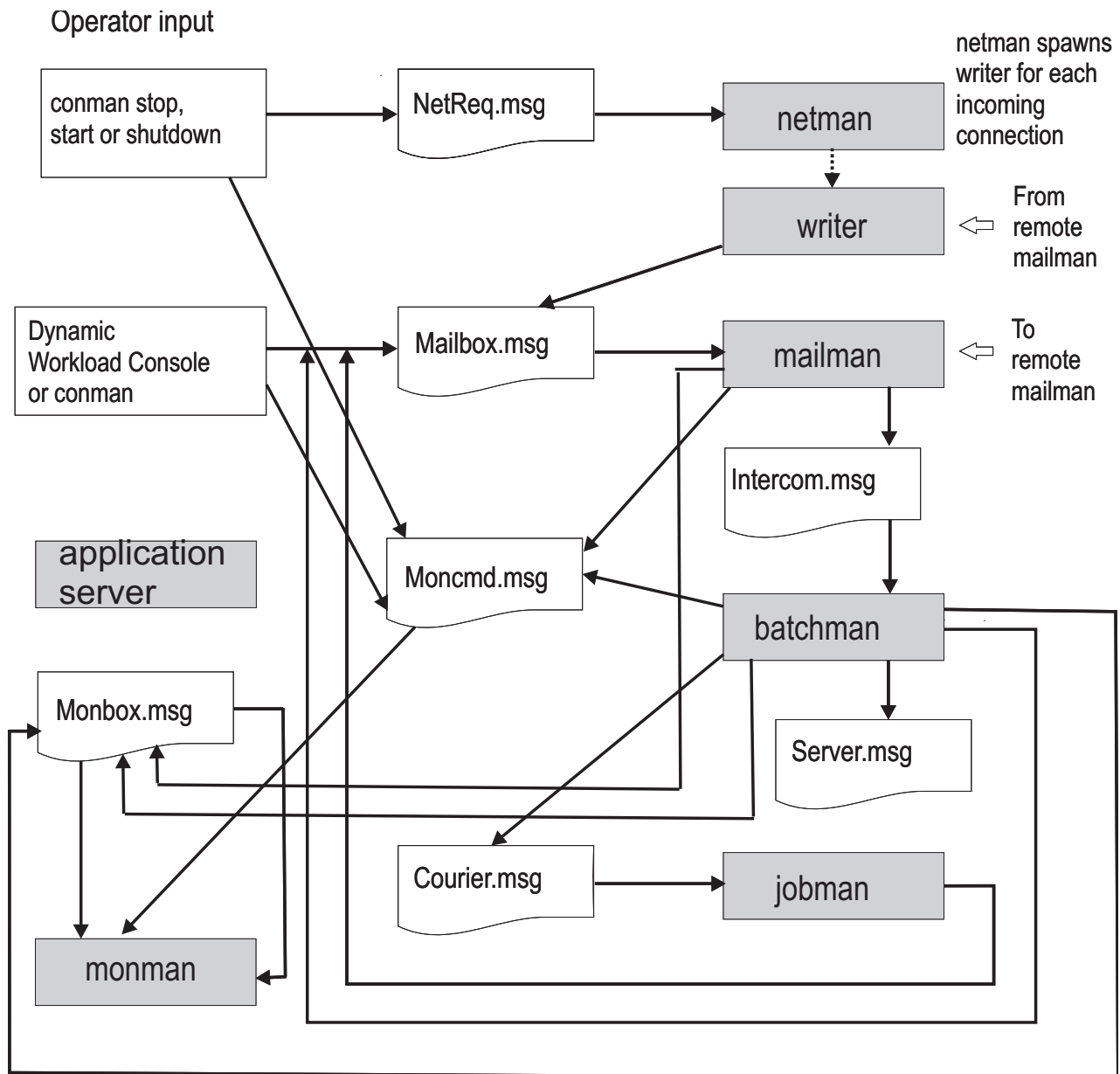


Figure 6. Communication inter-processus sur le gestionnaire de domaine maître et l'agent tolérant aux pannes

Ces fichiers ont une taille maximale de 10 Mo par défaut. Cette taille maximale peut être modifiée à l'aide de l'utilitaire **evtsize** (voir «evtsize», à la page 696).

Communication réseau de IBM Workload Scheduler

IBM Workload Scheduler utilise le protocole TCP/IP pour les communications réseau. Le nom de noeud et le numéro de port utilisés pour établir la connexion TCP/IP sont définis pour chaque poste de travail dans la définition qui leur est associée. Pour plus de détails, voir «Définition de poste de travail», à la page 166.

IBM Workload Scheduler recourt à une technologie de *stockage et réacheminement* pour maintenir la cohérence et la tolérance aux pannes à tout moment sur l'ensemble du réseau en mettant les messages en attente dans des fichiers de messages lorsque la connexion n'est pas active. Lorsque la communication TCP/IP

est établie entre les systèmes, IBM Workload Scheduler fournit des communications bidirectionnelles entre les postes de travail à l'aide de liaisons. Ces liaisons sont contrôlées par l'indicateur **autolink** défini (voir «Définition de poste de travail», à la page 166) et par des commandes émises par le programme de ligne de commande **conman** (voir «link», à la page 468 et «unlink», à la page 565).

Lorsqu'une liaison est ouverte, les messages sont transmis entre les postes de travail. Lorsqu'une liaison est fermée, le poste de travail expéditeur stocke les messages dans un fichier local de messages et les envoie au poste de travail de destination dès que la liaison est de nouveau ouverte.

Il existe essentiellement deux types de communication dans l'environnement IBM Workload Scheduler : l'initialisation des connexions et la livraison des événements de planification sous la forme des messages de changement d'état pendant la période de traitement. Ces deux types de communication sont décrits ici de façon détaillée.

Initialisation de la connexion et configuration de la communication bidirectionnelle

Voici les étapes permettant l'établissement d'une liaison IBM Workload Scheduler bidirectionnelle entre un gestionnaire de domaine et un agent tolérant aux pannes distant :

1. Sur le gestionnaire de domaine, le processus **mailman** lit le nom d'hôte, l'adresse TCP/IP et le numéro de port de l'agent tolérant aux pannes à partir du fichier Symphony.
2. Le processus **mailman** s'exécutant sur le gestionnaire de domaine établit une connexion TCP/IP avec le processus **netman** de l'agent tolérant aux pannes à l'aide des informations obtenues à partir du fichier Symphony.
3. Le processus **netman** s'exécutant sur l'agent tolérant aux pannes détermine que la demande provient du processus **mailman** qui s'exécute sur le gestionnaire de domaine et il génère un nouveau processus **writer** pour traiter la connexion entrante.
4. Le processus **mailman** s'exécutant sur le gestionnaire de domaine est désormais connecté au processus **writer** de l'agent tolérant aux pannes. Le processus **writer** de l'agent tolérant aux pannes communique le numéro d'exécution courant de sa copie du fichier Symphony au processus **mailman** du gestionnaire de domaine. Ce numéro d'exécution est l'identificateur utilisé par IBM Workload Scheduler pour reconnaître chaque fichier Symphony généré par **JnextPlan**. Cette étape est indispensable pour permettre au gestionnaire de domaine de vérifier si le plan courant a déjà été envoyé à l'agent tolérant aux pannes.
5. Le processus **mailman** du gestionnaire de domaine compare le numéro d'exécution de son fichier Symphony au numéro d'exécution du fichier Symphony de l'agent tolérant aux pannes. Si ces numéros diffèrent, le processus **mailman** du gestionnaire de domaine envoie au processus **writer** de l'agent tolérant aux pannes la dernière copie du fichier Symphony.
6. Une fois que le fichier Symphony courant est en place sur l'agent tolérant aux pannes, le processus **mailman** du gestionnaire de domaine envoie une commande **start** à l'agent tolérant aux pannes.

7. Le processus **netman** de l'agent tolérant aux pannes démarre le processus **mailman** local. A ce stade, une liaison de communication unidirectionnelle est établie depuis le gestionnaire de domaine vers l'agent tolérant aux pannes.
8. Le processus **mailman** de l'agent tolérant aux pannes lit le nom d'hôte, l'adresse TCP/IP et le numéro de port du gestionnaire de domaine à partir du fichier Symphony et il utilise ces informations pour établir une liaison en retour vers le processus **netman** du gestionnaire de domaine.
9. Le processus **netman** du gestionnaire de domaine détermine que la demande provient du processus **mailman** de l'agent tolérant aux pannes et il génère un nouveau processus **writer** pour traiter la connexion entrante. Le processus **mailman** de l'agent tolérant aux pannes est désormais connecté au processus **writer** du gestionnaire de domaine et une liaison de communication bidirectionnelle est établie. En conséquence, le processus **writer** du gestionnaire de domaine enregistre les messages reçus depuis l'agent tolérant aux pannes dans le fichier Mailbox.msg sur le gestionnaire de domaine et le processus **writer** de l'agent tolérant aux pannes enregistre les messages reçus depuis le gestionnaire de domaine dans le fichier Mailbox.msg sur l'agent tolérant aux pannes.

Le traitement des travaux et la transmission des événements de planification sous la forme de messages de changement d'état au cours du jour de traitement effectués localement par l'agent tolérant aux pannes

Au cours de la période de production, le fichier Symphony présent sur l'agent tolérant aux pannes est lu et mis à jour avec les informations de changement d'état relatives aux travaux exécutés localement par les processus de poste de travail IBM Workload Scheduler. Voici les opérations effectuées localement sur l'agent tolérant aux pannes pour lire et mettre à jour le fichier Symphony et pour traiter les travaux :

1. Le processus **batchman** lit un enregistrement dans le fichier Symphony qui indique que travail1 doit être lancé sur le poste de travail.
2. Le processus **batchman** écrit dans le fichier Courier.msg que travail1 doit être lancé.
3. Le processus **jobman** lit ces informations dans le fichier Courier.msg, lance travail1 et écrit dans le fichier Mailbox.msg que travail1 a démarré avec son *id_processus* et son *horodatage*.
4. Le processus **mailman** lit ces informations dans son fichier Mailbox.msg et envoie un message indiquant que travail1 a démarré avec son *ID_processus* et son *horodatage*, sur le fichier Mailbox.msg du gestionnaire de domaine et sur le fichier Intercom.msg local de l'agent tolérant aux pannes.
5. Le processus **batchman** de l'agent tolérant aux pannes lit le message dans le fichier Intercom.msg et met à jour la copie locale du fichier Symphony.
6. Une fois que le traitement du travail1 est terminé, le processus **jobman** met à jour le fichier Mailbox.msg avec les informations indiquant que travail1 est terminé.
7. Le processus **mailman** lit ces informations dans le fichier Mailbox.msg et envoie un message indiquant que travail1 est terminé à la fois au fichier Mailbox.msg du gestionnaire de domaine et au fichier Intercom.msg local de l'agent tolérant aux pannes.

8. Le processus **batchman** de l'agent tolérant aux pannes lit le message dans le fichier `Intercom.msg`, met à jour la copie locale du fichier `Symphony` et détermine le travail suivant à exécuter.

Pour plus d'informations sur l'ajustement du traitement du travail sur un poste de travail, voir *IBM Workload Scheduler : Guide d'administration*.

Prise en charge d'IPv6 (Internet Protocol version 6)

IBM Workload Scheduler prend en charge le protocole IP version 6 (IPv6) en complément de la version IPv4. Pour aider les clients à passer d'un environnement IPv4 à un environnement IPv6 complet, IBM Workload Scheduler prend en charge l'utilisation des protocoles IP en double pile. Cela signifie que le produit est conçu pour l'utilisation simultanée des adresses IPv4 et IPv6 lors de la communication avec les autres applications utilisant le protocole IPv4 ou IPv6.

Pour cela, les fonctionnalités `gethostbyname` et `gethostbyaddr` ont été supprimées de IBM Workload Scheduler car elle ne prenaient en charge que le protocole IPv4. Elles sont remplacées par la nouvelle interface API `getaddrinfo`, qui rend le mécanisme de communications client/serveur entièrement indépendant du protocole.

La fonction `getaddrinfo` traite à la fois les conversions de type nom/adresse et service/port et renvoie des structures `sockaddr` à la place d'une liste d'adresses. Ces structures `sockaddr` peuvent ensuite être utilisées directement par les fonctions de `socket`. Ceci permet à `getaddrinfo` de masquer toutes les dépendances vis-à-vis des protocoles dans les fonctions des bibliothèques auxquelles elles appartiennent. L'application gère uniquement les structures d'adresse de `socket` renseignées par `getaddrinfo`.

Chapitre 3. Configuration de l'environnement de travail

Le présent chapitre décrit la personnalisation de la gestion des travaux sur un poste de travail. Cette personnalisation consiste à attribuer sur chaque poste de travail des valeurs à des variables qui ont une incidence sur le traitement de **jobman**. Ce chapitre contient les sections suivantes :

- «Présentation de l'environnement de travail»
- «Variables d'environnement exportées par jobman», à la page 54
- «Personnalisation du traitement des travaux sur un poste de travail UNIX - jobmanrc», à la page 57
- «Personnalisation du traitement des travaux pour un utilisateur sous UNIX - jobmanrc», à la page 60
- «Personnalisation du traitement des travaux sur un poste de travail Windows - jobmanrc.cmd», à la page 61
- «Personnalisation du traitement des travaux sur un poste de travail Windows - djobmanrc.cmd», à la page 63

Présentation de l'environnement de travail

Sur chaque poste de travail, les travaux sont lancés par le processus de contrôle de la production **batchman**. Le processus **batchman** résout toutes les dépendances des travaux pour que les travaux soient traités dans le bon ordre, puis met en file d'attente un message de lancement de travail destiné au processus **jobman**.

Chaque processus lancé par **jobman** (scripts de configuration et travaux inclus) conserve le nom d'utilisateur enregistré lors de l'ouverture de session du travail. Les travaux soumis (travaux, fichiers ou commandes soumis(es) *manuellement* par un utilisateur et non par un plan programmé) conservent le nom de l'utilisateur qui a effectué la soumission.

Le processus **jobman** démarre un processus de surveillance des travaux qui commence par définir un groupe de variables d'environnement, puis exécute un script de configuration standard appelé *racine_TWS/jobmanrc*, qui peut être personnalisé. Le script **jobmanrc** définit des variables utilisées pour configurer localement sur le poste de travail la façon dont les travaux sont lancés, quel que soit l'utilisateur.

Sur les postes de travail UNIX, si l'utilisateur est autorisé à utiliser un script de configuration local et que le script *RACINE_UTILISATEUR/.jobmanrc* existe, le script de configuration local **.jobmanrc** est également exécuté. Le travail est ensuite exécuté par le script de configuration standard ou par le script de configuration local. Les résultats du traitement du travail sont signalés à **jobman** qui, à son tour, met à jour le fichier *Mailbox.msg* avec les informations sur l'état d'achèvement du travail. Pour exécuter des travaux sur l'environnement de l'utilisateur, ajoutez l'instruction suivante au script de configuration local :

```
. $USER_home/.profile
```

Remarque : Avant d'ajouter *.profile* au fichier *.jobmanrc*, assurez-vous qu'il ne contient pas de paramètre *stty* ni d'étapes qui exigent une intervention manuelle de l'utilisateur. Le cas échéant, ajoutez au fichier *.jobmanrc* uniquement les étapes nécessaires de *.profile*.

Sur les postes de travail Windows, le script de configuration local djobmanrc.cmd est exécuté s'il existe dans le répertoire Documents and Settings de l'utilisateur qui est représenté par la variable d'environnement %USERPROFILE% et dépend de l'installation du langage Windows. Le script djobmanrc.cmd sera exécuté par le script jobmanrc.cmd.

Variables d'environnement exportées par jobman

Les variables répertoriées ci-dessous (voir tableau 8) sont définies localement sur le poste de travail et exportées par **jobman** sur les systèmes d'exploitation Windows.

Tableau 8. Variables d'environnement de travail sous Windows

Nom de la variable	Valeur
COMPUTERNAME	Valeur de <i>COMPUTERNAME</i> définie dans l'environnement utilisateur.
HOME	Chemin d'installation de l'instance IBM Workload Scheduler.
HOMEDRIVE	Valeur de <i>HOMEDRIVE</i> définie dans l'environnement utilisateur.
HOMEPATH	Valeur de <i>HOMEPATH</i> définie dans l'environnement utilisateur.
LANG	Valeur de <i>LANG</i> définie dans l'environnement utilisateur. Si elle n'est pas spécifiée, sa valeur est définie sur C.
LOGNAME	Nom d'utilisateur de la connexion.
MAESTRO_OUTPUT_STYLE	Paramètre définissant le style de sortie pour les noms d'objet longs.
SystemDrive	Valeur de <i>SYSTEMDRIVE</i> définie dans l'environnement utilisateur.
SystemRoot	Valeur de <i>SYSTEMROOT</i> définie dans l'environnement utilisateur.
TEMP	Valeur de <i>TEMP</i> définie dans l'environnement utilisateur. Si elle n'est pas spécifiée, sa valeur est définie sur c:\temp.
TIVOLI_JOB_DATE	Date planifiée d'un travail.
TMPTMP	Valeur de <i>TMP</i> définie dans l'environnement utilisateur. Si elle n'est pas spécifiée, sa valeur est définie sur c:\temp.
TMPDIR	Valeur de <i>TMPDIR</i> définie dans l'environnement utilisateur. Si elle n'est pas spécifiée, sa valeur est définie sur c:\temp.
TWS_PROMOTED_JOB	S'applique aux fonctions Assurance de service de charge de travail. Peut prendre la valeur YES ou No. Si la valeur est YES, le travail (un travail critique ou l'un de ses prédécesseurs) a été promu.
TZ	Le fuseau horaire, s'il a été défini dans l'environnement du système d'exploitation.
UNISON_CPU	Nom de ce poste de travail.
UNISON_DIR	Valeur de <i>UNISON_DIR</i> définie dans l'environnement utilisateur.
UNISON_EXEC_PATH	Chemin complet de jobmanrc.

Tableau 8. Variables d'environnement de travail sous Windows (suite)

Nom de la variable	Valeur
UNISONHOME	Chemin d'installation de l'instance IBM Workload Scheduler.
UNISON_HOST	Nom du poste de travail hôte.
UNISON_JOB	L'identificateur de travail absolu : poste_de_travail#id_plan.travail.
UNISON_JOBNUM	Numéro du travail.
UNISON_MASTER	Nom du poste de travail maître.
UNISON_RUN	Numéro d'exécution de production en cours d'IBM Workload Scheduler.
UNISON_SCHED	Nom du flot de travaux.
UNISON_SCHED_DATE	Date de production d'IBM Workload Scheduler (aammjj) signalée dans l'en-tête du fichier Symphony.
UNISON_SCHED_ID	ID_flot_travaux du flot de travaux qui contient le travail en cours.
UNISON_SCHED_IA	Date à laquelle le flot de travaux a été ajouté au plan.
UNISON_SCHED_EPOCH	Date de production d'IBM Workload Scheduler (exprimée au format Epoch).
UNISON_SHELL	Shell de connexion de l'utilisateur qui exécute le travail.
UNISON_STDLIST	Nom du chemin d'accès du fichier de liste standard du travail.
UNISON_SYM	Numéro d'enregistrement Symphony du travail lancé.
USERDOMAIN	Valeur de <i>USERDOMAIN</i> définie dans l'environnement utilisateur.
USERNAME	Valeur de <i>USERNAME</i> définie dans l'environnement utilisateur.
USERPROFILE	Valeur de <i>USERPROFILE</i> définie dans l'environnement utilisateur.

Les variables répertoriées ci-dessous (voir tableau 9) sont définies localement sur le poste de travail et exportées par **jobman** sur les systèmes d'exploitation UNIX.

Tableau 9. Variables d'environnement de travail pour UNIX

Nom de la variable	Valeur
HOME	Répertoire principal de l'utilisateur.
LANG	Valeur de <i>LANG</i> définie dans l'environnement utilisateur.
LD_LIBRARY_PATH	Valeur de <i>LD_LIBRARY_PATH</i> définie dans l'environnement utilisateur.
LD_RUN_PATH	Valeur de <i>LD_RUN_PATH</i> définie dans l'environnement utilisateur.
LOGNAME	Nom d'utilisateur de la connexion.
MAESTRO_OUTPUT_STYLE	Paramètre définissant le style de sortie pour les noms d'objet longs.
PATH	/bin:/usr/bin

Tableau 9. Variables d'environnement de travail pour UNIX (suite)

Nom de la variable	Valeur
TIVOLI_JOB_DATE	Date planifiée d'un travail.
TWS_PROMOTED_JOB	S'applique aux fonctions Assurance de service de charge de travail. Peut prendre la valeur YES ou No. Si la valeur est YES, le travail (un travail critique ou l'un de ses prédécesseurs) a été promu.
TWS_TISDIR	Valeur de <i>TWS_TISDIR</i> définie dans l'environnement utilisateur.
TZ	Fuseau horaire défini.
UNISON_CPU	Nom de ce poste de travail.
UNISON_DIR	Valeur de <i>UNISON_DIR</i> définie dans l'environnement utilisateur.
UNISON_EXEC_PATH	Chemin complet de .jobmanrc.
UNISONHOME	Chemin d'installation de l'instance IBM Workload Scheduler.
UNISON_HOST	Nom du poste de travail hôte.
UNISON_JOB	L'identificateur de travail absolu : poste_de_travail#id_plan.travail.
UNISON_JOBNUM	Numéro du travail.
UNISON_MASTER	Nom du poste de travail maître.
UNISON_RUN	Numéro d'exécution de production en cours d'IBM Workload Scheduler.
UNISON_SCHED	Nom du flot de travaux.
UNISON_SCHED_DATE	Date de production d'IBM Workload Scheduler (aammjj) signalée dans l'en-tête du fichier Symphony.
UNISON_SCHED_ID	<i>ID_flot_travaux</i> du flot de travaux qui contient le travail en cours.
UNISON_SCHED_IA	Date à laquelle le flot de travaux a été ajouté au plan.
UNISON_SCHED_EPOCH	Date de production d'IBM Workload Scheduler (exprimée au format Epoch).
UNISON_SHELL	Shell de connexion de l'utilisateur qui exécute le travail.
UNISON_STDLIST	Nom du chemin d'accès du fichier de liste standard du travail.
UNISON_SYM	Numéro d'enregistrement Symphony du travail lancé.

Personnalisation du format des dates dans stdlist

Vous pouvez utiliser une variable d'environnement nommée ***UNISON_DATE_FORMAT*** pour définir le format de date utilisé pour la date de l'en-tête et du bas de page du fichier *stdlist*. Cette variable peut être définie sur les postes de travail UNIX et Windows, impérativement avant le lancement des processus IBM Workload Scheduler sur ce poste de travail pour être effective. Pour définir cette variable, procédez comme suit :

Sous UNIX

1. Ajoutez l'instruction pour exporter la variable `UNISON_DATE_FORMAT` dans le fichier `.profile` racine.
2. Exécutez le fichier `.profile`.
3. Exécutez **conman shutdown**, puis **./StartUp.sh**.

Sous Windows

1. Dans les Propriétés système, définissez la variable système `UNISON_DATE_FORMAT`.
2. Exécutez **conman shutdown**, puis **StartUp.sh**.

Voici quelques exemples de paramètres utilisés pour afficher le format de l'année dans la zone de date de l'en-tête et du bas de page du fichier `stdlist`. Le paramètre :

```
UNISON_DATE_FORMAT = "%a %x %X %Z %Y"
```

produit une sortie au format suivant :

```
Ven 15/10/04 11:05:24 AM GMT 2004
```

Le paramètre :

```
UNISON_DATE_FORMAT = "%a %x %X %Z"
```

produit une sortie au format suivant :

```
Ven 15/10/04 11:05:24 AM GMT
```

Définissez cette variable localement sur chaque poste de travail pour lequel vous souhaitez afficher le format de l'année à 4 chiffres. Si ce paramètre est omis, le format standard à deux caractères est utilisé.

Personnalisation du traitement des travaux sur un poste de travail UNIX - `jobmanrc`

IBM Workload Scheduler propose un modèle de script de configuration standard nommé `racine_TWS/config/jobmanrc`. Il est installé automatiquement sous le nom `racine_TWS/jobmanrc`. L'administrateur système peut utiliser ce script pour définir l'environnement nécessaire avant l'exécution de chaque travail. Pour modifier le script, apportez vos modifications dans la copie de travail (`racine_TWS/jobmanrc`), sans manipuler le fichier modèle. Le fichier contient des variables que vous pouvez configurer et quelques commentaires qui vous aident à assimiler la méthodologie. Pour découvrir les variables `jobmanrc`, voir tableau 10.

Tableau 10. Variables définies par défaut dans le fichier `jobmanrc`

Nom de la variable	Valeur
UNISON_JCL	Le nom du chemin d'accès au fichier script du travail
UNISON_STDLIST	Le nom du chemin d'accès au fichier liste standard du travail

Tableau 10. Variables définies par défaut dans le fichier jobmanrc (suite)

Nom de la variable	Valeur
UNISON_EXIT	yes no Si cette variable a pour valeur yes, le travail s'arrête immédiatement lorsqu'une commande renvoie un code de sortie non nul. Si elle est définie sur no, le programme continue l'exécution du travail même si une commande renvoie un code de sortie non nul. Les autres valeurs sont interprétées comme étant définies sur no.
LOCAL_RC_OK	yes no Si la valeur yes est attribuée à cette variable, le script de configuration local de l'utilisateur s'exécute (s'il existe) et passe <code>\$UNISON_JCL</code> comme premier argument. Dans certains cas, l'utilisateur peut ne pas être autorisé à utiliser cette option. Pour plus d'informations, voir «Personnalisation du traitement des travaux pour un utilisateur sous UNIX - .jobmanrc», à la page 60. Si la valeur no est attribuée à cette variable, le programme ignore l'existence d'un script de configuration local et exécute <code>\$UNISON_JCL</code>. Les autres valeurs sont interprétées comme étant définies sur no.
MAIL_ON_ABEND	yes no Pour les systèmes d'exploitation UNIX : Si la variable est définie sur yes, le programme envoie un message à la boîte aux lettres de l'utilisateur connecté dès que le travail se termine avec un code de sortie non nul. Vous pouvez également définir la variable sur un ou plusieurs noms d'utilisateur (préalablement séparés par des espaces) de façon à envoyer un message à chacun d'entre eux. Par exemple, vous pouvez taper "root mis sam mary". Si la variable est définie sur no, le programme n'envoie aucun message en cas de fin anormale du travail. Les messages d'arrêt anormal utilisent le format suivant : <pre>cpu#sched.job fichier-jcl failed with code-sortie Please review nom_fichier_liste-standard</pre> Vous pouvez modifier ou traduire le texte du message dans une autre langue. Pour plus d'informations à ce sujet, voir «Personnalisation de la section MAIL_ON_ABEND de jobmanrc», à la page 59.

Tableau 10. Variables définies par défaut dans le fichier jobmanrc (suite)

Nom de la variable	Valeur
SHELL_TYPE	standard user script Si cette variable a pour valeur standard, la première ligne du fichier JCL est lue pour déterminer quel shell utiliser pour exécuter le travail. Si la première ligne ne commence pas par le symbole #!, le programme utilise le répertoire /bin/sh pour exécuter le script de configuration local ou \$UNISON_JCL. Les commandes sont répercutées sur le fichier de liste standard du travail. Si la variable est définie sur user, le shell de connexion de l'utilisateur (\$UNISON_SHELL) exécute le script de configuration local ou \$UNISON_JCL. Les commandes sont répercutées sur le fichier de liste standard du travail. Si la valeur script est attribuée à la variable, le programme exécute directement le script de configuration local ou \$UNISON_JCL, et les commandes ne sont pas répercutées tant que le script de configuration local ou \$UNISON_JCL contient une commande set -x. Les autres valeurs sont interprétées comme étant définies sur standard.
USE_EXEC	yes no Si cette variable a pour valeur yes, le travail ou le script de configuration local de l'utilisateur est exécuté à l'aide de la commande exec, ce qui élimine un processus supplémentaire. Cette option est remplacée si MAIL_ON_ABEND est également défini sur yes. Les autres valeurs sont interprétées comme étant définies sur no, auquel cas le travail ou le script de configuration local est exécuté par un autre processus du shell.

Personnalisation de la section MAIL_ON_ABEND de jobmanrc

Vous pouvez modifier le texte utilisé dans le message envoyé aux utilisateurs spécifiés dans la zone **MAIL_ON_ABEND** du fichier de configuration *racine_TWS/jobmanrc* en accédant à ce fichier et en modifiant le texte dans les parties qui apparaissent en caractères gras :

```
# Mail a message to user or to root if the job fails.
```

```
if [ "$MAIL_ON_ABEND" = "YES" ]
then
  if [ $UNISON_RETURN -ne 0 ]
  then
    mail $LOGNAME <<--!
      $UNISON_JOB
      \'$UNISON_JCL\' failed with $UNISON_RETURN
      Please review $UNISON_STDLIST
  !
  fi
elif [ "$MAIL_ON_ABEND" = "ROOT" ]
then
```

```

if [ $UNISON_RETURN -ne 0 ]
then
    mail root <<-!
        $UNISON_JOB
        \'$UNISON_JCL\' failed with $UNISON_RETURN
        Please review $UNISON_STDLIST
    !
fi
elif [ "$MAIL_ON_ABEND" != "NO" ]
then
    if [ $UNISON_RETURN -ne 0 ]
    then
        mail $MAIL_ON_ABEND <<-!
            $UNISON_JOB
            \'$UNISON_JCL\' failed with $UNISON_RETURN
            Please review $UNISON_STDLIST
        !
    fi
fi

```

Personnalisation du traitement des travaux pour un utilisateur sous UNIX - .jobmanrc

Sur les postes de travail UNIX, le script de configuration local **.jobmanrc** permet aux utilisateurs de définir l'environnement nécessaire lors du traitement de leurs propres travaux. Contrairement au script **jobmanrc**, le script **.jobmanrc** peut être personnalisé pour effectuer différentes actions pour différents utilisateurs. Chaque utilisateur défini comme *utilisateur_tws* peut personnaliser le script **.jobmanrc** de son répertoire personnel pour effectuer des actions de prétraitement et de posttraitement. Le script **.jobmanrc** est une étape supplémentaire qui s'exécute avant le lancement du travail proprement dit.

Le script **.jobmanrc** s'exécute uniquement dans les conditions suivantes :

- Le script de configuration standard **jobmanrc** est installé et la variable d'environnement **LOCAL_RC_OK** définie sur **yes** (voir tableau 10, à la page 57).
- Si le fichier *racine_TWS/localrc.allow* existe, le nom d'utilisateur doit figurer dans le fichier. Si le fichier *racine_TWS/localrc.allow* n'existe pas, le nom de l'utilisateur ne doit pas apparaître dans le fichier *racine_TWS/localrc.deny*. Si aucun des fichiers n'existe, l'utilisateur est autorisé à utiliser un script de configuration local.
- Le script de configuration local est installé dans le répertoire personnel de l'utilisateur (*racine_UTILISATEUR/.jobmanrc*) et dispose de droits en exécution.

Les travaux ne sont pas automatiquement exécutés ; la commande ou le script doit être lancé à partir de **.jobmanrc**. La commande ou le script est lancé différemment selon le type de processus que vous voulez exécuter. Suivez les règles générales ci-dessous lorsque vous lancez des scripts à partir de **.jobmanrc** :

- Utilisez **eval** si vous voulez lancer une commande.
- Utilisez soit **eval**, soit **exec** si vous voulez lancer un script qui n'a pas besoin d'activités de posttraitement.
- Utilisez **eval** si vous voulez lancer un script qui a besoin d'activités posttraitement.

Si vous avez l'intention d'utiliser un script de configuration local, il doit au minimum exécuter le fichier script du travail (*\$UNISON_JCL*). Le script de configuration standard fourni par IBM Workload Scheduler, **jobmanrc**, exécute votre script de configuration local comme suit :

```
$EXECIT $USE_SHELL $USER_home/.jobmanrc "$UNISON_JCL" $IS_COMMAND
```

où :

- La valeur de *USE_SHELL* est celle de la variable *SHELL_TYPE* de jobmanrc (voir tableau 10, à la page 57).
- La valeur **yes** est attribuée à *IS_COMMAND* si le travail a été planifié ou soumis dans la production à l'aide de **submit docommand**.
- *EXECIT* a pour valeur **exec** si la variable *USE_EXEC* a pour valeur **yes** (voir tableau 10, à la page 57), sinon elle est indéfinie (null).

Toutes les variables exportées dans **jobmanrc** sont disponibles dans le shell **.jobmanrc** ; cependant, les variables définies mais non exportées ne sont pas disponibles.

Dans l'exemple suivant, nous expliquons comment exécuter le fichier script d'un travail ou une commande dans votre script de configuration local :

```
#!/bin/ksh
PATH=racine_TWS:racine_TWS/bin:$PATH
export PATH
/bin/sh -c "$UNISON_JCL"
```

Voici un exemple de script .jobmanrc qui effectue les traitements en fonction du code de sortie du travail de l'utilisateur :

```
#!/bin/sh
#
PATH=racine_TWS:racine_TWS/bin:$PATH
export PATH
/bin/sh -c "$UNISON_JCL"
#Ou utilisez eval "$UNISON_JCL" (guillemets obligatoires)
RETVAL=$?
if [ $RETVAL -eq 1 ]
then
    echo "Code de sortie 1 - Erreur non bloquante"
    exit 0
elif [ $RETVAL -gt 1 -a $RETVAL -lt 100 ]
then
    conman "tellop : Erreur de base de données. Contactez l'administrateur
de la base de données"
elif [ $RETVAL -ge 100 ]
then
    conman "Abandon du travail tellop. Contactez l'administrateur"
fi
```

Personnalisation du traitement des travaux sur un poste de travail Windows - jobmanrc.cmd

IBM Workload Scheduler propose un modèle de script de configuration standard nommé *racine_TWS\config\jobmanrc.cmd*. Il est installé automatiquement sous le nom *racine_TWS\jobmanrc.cmd*. Vous pouvez utiliser ce fichier de commandes pour définir l'environnement nécessaire avant l'exécution de chaque travail. Pour modifier le fichier, apportez vos modifications dans la copie de travail

(*racine_TWS\jobmanrc.cmd*), sans manipuler le fichier modèle. Le fichier contient des variables que vous pouvez configurer et quelques commentaires qui vous aident à assimiler la méthodologie. Pour découvrir les variables *jobmanrc.cmd*, voir tableau 11.

Tableau 11. Variables définies par défaut dans le fichier *jobmanrc.cmd*

Nom de la variable	Valeur
HOME	Chemin d'accès au répertoire <i>racine_TWS</i>
POSIXHOME	Chemin d'accès au répertoire <i>racine_TWS</i> au format POSIX
LOCAL_RC_OK	- Si cette variable a pour valeur yes, le script de configuration local de l'utilisateur est exécuté, s'il existe. - Si cette variable a pour valeur no, le programme ignore l'existence d'un script de configuration local. Les autres valeurs sont interprétées comme étant définies sur no.
MAIL_ON_ABEND	- Si cette variable est définie sur YES, un courrier électronique est envoyé à l'ID de messagerie électronique défini dans la variable <i>email_ID</i>, si le travail se termine par une erreur. - Si cette variable est définie par une autre valeur que YES ou NO, un courrier électronique est envoyé à l'ID de messagerie électronique spécifié dans cette variable, si le travail se termine par une erreur. - Si cette variable est définie par NO, aucun message n'est envoyé si le travail se termine par une erreur. Pour plus de détails, voir «Personnalisation de la section MAIL_ON_ABEND de <i>jobmanrc.cmd</i>».

Personnalisation de la section MAIL_ON_ABEND de jobmanrc.cmd

Vous pouvez modifier le texte utilisé dans le message envoyé aux utilisateurs spécifiés dans la zone *MAIL_ON_ABEND* du fichier de configuration *TWS_home/jobmanrc.cmd* en accédant à ce fichier et en modifiant le texte dans les parties qui apparaissent en caractères gras. Pour expliquer plus clairement comment générer un message électronique, un modèle de programme de messagerie portant le nom *bmail.exe* est utilisé.

```
if /I "%MAIL_ON_ABEND%"=="NO" (goto :out) else (goto :mail_on_abend)

:mail_on_abend
REM *****email, task or other action inserted here *****
if /I "%MAIL_ON_ABEND%"=="YES" (goto :email) else (goto :email_spec)

:email
c:\Program Files\utils\bmail.exe -s smtp.yourcompany.com -t %EMAIL_ID%
-f %COMPUTERNAME%@yourcompany.com -h -a "Subject: Job %UNISON_JOB% abended"
-b "Job %UNISON_JOB% Job Number %UNISON_JOBNUM% abended"
goto :out
```

```
:email_spec
REM set > c:\tmp\abended_jobs\%UNISON_JOB%.j%UNISON_JOBNUM%
c:\Program Files\utils\bmail.exe -s smtp.yourcompany.com -t %MAIL_ON_ABEND%
-f %COMPUTERNAME%@yourcompany.com -h -a "Subject: Job %UNISON_JOB% abended"
-b "Job %UNISON_JOB% Job Number %UNISON_JOBNUM% abended"
```

Personnalisation du traitement des travaux sur un poste de travail Windows - djobmanrc.cmd

Sur les postes de travail Windows, vous pouvez utiliser le script de configuration local djobmanrc.cmd pour établir un environnement spécifique lors du traitement de vos travaux personnalisés. Contrairement au script jobmanrc.cmd, vous pouvez personnaliser le script djobmanrc.cmd afin d'effectuer différentes actions pour différents utilisateurs.

Les conditions suivantes s'appliquent :

- Le script doit contenir toutes les variables ou chemins d'application d'environnement nécessaires pour que IBM Workload Scheduler soit lancé correctement.
- Le script doit exister si un environnement spécifique à l'utilisateur pour l'exécution du travail est requis ou si un courrier électronique doit être envoyé à l'utilisateur de connexion du travail lorsque le travail IBM Workload Scheduler se termine par une erreur.

Pour créer un script djobmanrc.cmd personnalisé, procédez comme suit :

1. Connectez-vous en tant que l'utilisateur qui définit les variables d'environnement pour le lancement des travaux IBM Workload Scheduler.
2. Ouvrez une invite de commande DOS.
3. Entrez la commande **set** qui réachemine la sortie standard vers un fichier à plat nommé user_env.
4. Créez un fichier appelé djobmanrc.cmd dans le répertoire Documents and Settings de l'utilisateur, comportant le texte par défaut suivant au début :

```
@ECHO OFF
echo Invoking %USERNAME% DJOBMANRC.CMD V.1
set USERPROFILE=%USERPROFILE%
::Setup User Environment Phase
```
5. Editez le fichier user_env créé à l'étape 3.
6. Insérez la commande **set** sur chaque ligne avant chaque variable d'environnement.
7. Ajoutez les modifications à la variable PATH à la fin du fichier djobmanrc.cmd dans une chaîne du type suivant :

```
set PATH=<TWSHOME>;<TWSHOME>\bin;%PATH%
```
8. Ajoutez le texte suivant à la fin du fichier user_env et remplacez la chaîne *user_email id* par l'ID de messagerie électronique de l'utilisateur qui reçoit la notification par courrier électronique, si le travail se termine par une erreur.

```
set EMAIL_ID=<user_email id>
::Launch Operation Phase
%ARGS%
::Post Operations Phase
:out
```
9. Ajoutez le fichier user_env mis à jour à la fin du fichier djobmanrc.cmd. Le fichier djobmanrc.cmd modifié doit ressembler à l'exemple suivant :

```

@ECHO OFF
echo Invoking %USERNAME% DJOBMANRC.CMD V.1
set USERPROFILE=%USERPROFILE%
::Setup User Environment Phase
set ALLUSERSPROFILE=C:\Documents and Settings\All Users
set APPDATA=C:\Documents and Settings\petes\Application Data
set CommonProgramFiles=C:\Program Files\Common Files
set COMPUTERNAME=PSOTOJ
set ComSpec=C:\WINDOWS\system32\cmd.exe
set CURDRIVE=C
set FP_NO_HOST_CHECK=NOset
set HOMEDRIVE=c:
set HOMEPATH=\docs
set LOGONSERVER=\\PSOTOJ
set NEWVAR=c:\tmp\tmp\mlist1
set NUMBER_OF_PROCESSORS=1
set OPC_CLIENT_ROOT=C:\opc\Client
set OS=Windows_NT
set Path=C:\Program Files\utils;C:\PROGRAM
FILES\THINKPAD\UTILITIES;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Program
Files\IBM\Infoprint Select;C:\Utilities;C:\Notes;C:\Program Files\IBM\Trace Facility;C:\Program
Files\IBM\Personal Communications;C:\Program Files\XLView;C:\lotus\compnent;C:\WINDOWS\Downloaded
Program Files;C:\Program Files\Symantec\pcAnywhere\;"C:\Program Files\Symantec\Norton Ghost
2003";C:\Infoprint;
set PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
set PCOMM_Root=C:\Program Files\IBM\Personal Communications\
set PDBASE=C:\Program Files\IBM\Infoprint Select
set PDHOST=
set PD_SOCKET=6874
set PROCESSOR_ARCHITECTURE=x86
set PROCESSOR_IDENTIFIER=x86 Family 6 Model 9 Stepping 5, GenuineIntel
set PROCESSOR_LEVEL=6
set PROCESSOR_REVISION=0905
set ProgramFiles=C:\Program Files
set PROMPT=$P$G
set SESSIONNAME=Console
set SystemDrive=C:
set SystemRoot=C:\WINDOWS
set TEMP=C:\DOCUMENT1\petes\LOCALS1\Temp
set TMP=C:\DOCUMENT1\petes\LOCALS1\Temp
set tvdebugflags=0x260
set tvlogsessioncount=5000
set TWS4APPS_JDKHOME=c:\win32app\TWS\pete\methods\_tools
set USERDOMAIN=PSOTOJ
set USERNAME=petes
set USERPROFILE=C:\Documents and Settings\petes
set windir=C:\WINDOWSPATH=c:\win32app\TWS\TWSuser:c:\win32app\TWS\TWSuser\bin;%PATH%
set PATH=c:\win32app\TWS\TWSuser;c:\win32app\TWS\TWSuser\bin;%PATH%
set EMAIL_ID=johndoe@yourcompany.com
::Launch Operation Phase
%ARGS%
::Post Operations Phase
:out

```

La phase des opérations de lancement dans le script a lieu quand le script, le fichier binaire ou la commande définie pour le travail s'est terminé. Le texte "%ARGS%" est obligatoire.

La phase des opérations ultérieures dans le script a lieu quand un code d'exit de travail peut être à nouveau réglé d'un état ABEND à SUCC, en remplaçant un code d'exit différent de zéro par un code d'exit zéro. Certaines applications peuvent comporter des codes d'exit pouvant être des avertissements. IBM Workload Scheduler évalue les codes d'exit à zéro ou une valeur différente de zéro. Les codes d'exit zéro indiquent un travail à l'état "SUCC". Tous les autres codes indiquent un travail à l'état ABEND. Les codes d'exit autres que zéro peuvent être réglés si nécessaire. L'exemple suivant présente ce qui peut être inclus dans la phase d'opérations ultérieures. L'exemple extrait le code d'exit du travail défini pour déterminer comment le gérer en fonction des instructions If :

```

set EMAIL_ID=johndoe@yourcompany.com
::Launch Operation Phase
%ARGS%
::Post Operations Phase
set RETVAL=%ERRORLEVEL%
if "%RETVAL%"=="0" goto out
if "%RETVAL%"=="1" set RETVAL=0
if "%RETVAL%"=="6" set RETVAL=0
:out
exit %RETVAL%

```

Configuration des options pour l'utilisation des interfaces utilisateur

Pour utiliser Dynamic Workload Console, les paramètres de connexion sont fournis au sein de la console et sauvegardés lors de sa configuration.

Pour utiliser le client de ligne de commande IBM Workload Scheduler, vous devez fournir les informations de configuration suivantes (appelées *paramètres_connexion*) afin de vous connecter au gestionnaire de domaine maître via HTTP/HTTPS à l'aide de l'infrastructure WebSphere Application Server :

nom d'hôte

Nom d'hôte du gestionnaire de domaine maître.

numéro de port

Numéro de port utilisé lors de la connexion au gestionnaire de domaine maître.

nom d'utilisateur, mot de passe

Les données d'identification, le nom d'utilisateur et le mot de passe, de *l'utilisateur_TWS*.

nom d'hôte proxy

Le nom d'hôte proxy utilisé dans la connexion avec le protocole HTTP.

numéro de port proxy

Le numéro de port proxy utilisé dans la connexion avec le protocole HTTP.

protocole

Protocole utilisé lors de la communication. Il peut s'agir de HTTP avec une authentification standard ou de HTTPS avec une authentification par certificat.

timeout

Le délai d'attente indiquant le temps maximum pendant lequel le programme de l'interface utilisateur peut attendre la réponse du gestionnaire de domaine maître avant de considérer la demande de communication comme un échec.

poste de travail par défaut

Le nom de poste de travail du gestionnaire de domaine maître auquel vous souhaitez vous connecter.

paramètres SSL

Si vous avez configuré votre réseau pour utiliser SSL afin de communiquer entre les interfaces et le gestionnaire de domaine maître, vous devez également fournir l'ensemble de paramètres SSL appropriés (qui dépend de la configuration de votre couche SSL).

Au cas où le client de ligne de commande est installé sur le gestionnaire de domaine maître, cette configuration est effectuée automatiquement lors de l'installation.

Pour le client de ligne de commande installé sur d'autres postes de travail, ces informations peuvent être fournies soit par stockage dans les fichiers de propriétés de ces postes de travail ou en les fournissant dans la chaîne des commandes que vous utilisez.

Les fichiers de propriétés auxquels il est fait référence sont les fichiers **localopts** et **useropts** :

localopts

Ce fichier contient un ensemble de paramètres applicables au poste de travail local pour une instance spécifique du produit installé.

useropts

Ce fichier contient un sous-ensemble des paramètres de localopts qui ont des valeurs personnalisées pour un utilisateur spécifique. Le chemin de ce fichier est compris dans le répertoire de base de l'utilisateur, ce qui conserve la confidentialité de ces informations.

Etant donné que IBM Workload Scheduler prend en charge l'installation de plusieurs instances du produit sur la même machine, il peut y avoir plusieurs instances de fichier useropts de chaque utilisateur. La possibilité d'avoir plusieurs fichiers useropts ayant chacun un nom différent permet de spécifier différents ensembles de paramètres de connexion pour les utilisateurs définis sur plusieurs instances du produit installées sur la même machine.

Dans le fichier localopts de chaque instance du produit installé, l'option nommée *useropts* identifie le nom du fichier useropts auquel le système doit accéder pour se connecter à cette instance d'installation.

Par conséquent, si deux IBM Workload Scheduler sont installées sur une machine et qu'un utilisateur système appelé operator est défini en tant qu'utilisateur dans les deux instances, l'option locale *useropts* = useropts1 du fichier localopts de la première instance identifie le fichier useropts1 qui contient les paramètres de connexion nécessaires à l'utilisateur operator pour se connecter à cette instance de IBM Workload Scheduler. D'autre part, l'option locale *useropts* = useropts2 du fichier localopts de la deuxième instance de IBM Workload Scheduler identifie le fichier useropts2 qui contient les paramètres de connexion nécessaires à l'utilisateur operator pour se connecter à cette instance de IBM Workload Scheduler.

Des détails complets sur la configuration de cet accès sont fournis dans *IBM Workload Scheduler - Guide d'administration*, dans la rubrique intitulée "Configuration de l'authentification de l'accès au client de ligne de commande".

Chapitre 4. Gestion du cycle de production

Le coeur d'une solution de gestion des travaux et de planification est la création et la gestion du *plan de production*. Le plan de production est une liste de tâches qui contient toutes les actions à effectuer dans un intervalle de temps donné sur les postes de travail du réseau de planification à l'aide des ressources disponibles et en respectant les relations et les restrictions définies.

Le présent chapitre décrit comment IBM Workload Scheduler gère les plans.

Il comprend les sections suivantes :

- «Concepts de base de la gestion de plan»
- «Personnalisation de la gestion du plan à l'aide des options globales», à la page 86
- «Création et extension d'un plan de production», à la page 91
- «Ligne de commande Planman», à la page 94
- «Démarrage du traitement du plan de production», à la page 113
- «Automatisation du traitement du plan de production», à la page 114

Concepts de base de la gestion de plan

Le *plan de production* contient des informations sur les travaux à exécuter, sur quel agent tolérant aux pannes et quelles dépendances doivent être satisfaites avant le lancement de chaque travail. IBM Workload Scheduler crée le plan de production à partir des données de modélisation stockées dans la base de données et d'un plan intermédiaire appelé *plan de préproduction*. Ce dernier est automatiquement créé et géré par le produit. Pour éviter les problèmes, la base de données est verrouillée pendant la génération du plan et déverrouillée lorsque la génération est terminée ou lorsqu'une erreur se produit. Le plan de préproduction permet d'identifier à l'avance les instances de flot de travaux et les dépendances externes de prédécesseur/successeur des flots de travaux intervenant pendant la période indiquée.

Vous devez utiliser le script **JnextPlan** sur le gestionnaire de domaine maître pour générer le plan de production et le distribuer sur le réseau IBM Workload Scheduler.

Vous pouvez exécuter la commande JnextPlan à partir d'un interpréteur de commandes sur le gestionnaire de domaine maître si vous êtes l'un des utilisateurs suivants :

- L'utilisateur utilisateur_TWS pour lequel vous avez installé le produit sur la machine, s'il n'est pas désactivé par les paramètres définis dans le fichier de sécurité.
- L'utilisateur Root sur les systèmes d'exploitation UNIX ou l'utilisateur administrateur sur les systèmes d'exploitation Windows, s'il n'est pas désactivé par les paramètres définis dans le fichier de sécurité.

Pour plus d'informations sur le script **JnextPlan**, voir «Création et extension d'un plan de production», à la page 91.

Pour générer et démarrer un nouveau plan de production, IBM Workload Scheduler exécute les étapes suivantes :

1. Il met à jour le plan de préproduction avec les objets définis dans la base de données qui ont été ajoutés ou mis à jour depuis la dernière création ou extension du plan.
2. Il extrait du plan de préproduction les informations sur les flots de travaux à exécuter dans la période spécifiée et enregistre ces informations dans un plan de production intermédiaire.
3. Il inclut dans le nouveau plan de production les flots de travaux inachevés du plan de production précédent.
4. Il crée le nouveau plan de production et le stocke dans un fichier nommé Symphony. Les données du plan sont également répliquées dans la base de données.
5. Il distribue une copie du fichier Symphony aux postes de travail concernés par le traitement du nouveau plan de production.
6. Journalise toutes les statistiques du plan de production précédent dans une archive
7. Met à jour les états de flot de travaux dans le plan de préproduction

La copie du fichier Symphony qui vient d'être générée est déployée en commençant par les agents tolérants aux pannes du domaine le plus haut et les gestionnaires de domaine des domaines enfants, puis en descendant dans l'arborescence vers tous les domaines subordonnés.

Chaque agent tolérant aux pannes et gestionnaire de domaine qui reçoit le nouveau fichier Symphony, archive le fichier Symphony précédent sous Symphony.last au chemin `<rép_base_TWA>/TWS/`, de sorte qu'une copie de sauvegarde soit conservée. Ce mécanisme permet d'afficher les données Symphony précédentes dans le cas où des mises à jour de message sont appliquées aux états de travail et de flot de travaux perdus entre l'agent et son gestionnaire de domaine maître.

Chaque agent tolérant aux pannes qui reçoit le plan de production peut poursuivre le traitement même si la connexion réseau avec le gestionnaire de domaine associé s'interrompt.

Sur chaque agent tolérant aux pannes destinataire, les processus IBM Workload Scheduler exécutent les actions suivantes pour gérer le traitement des travaux :

1. Accès à la copie du fichier Symphony et lecture des instructions sur les travaux à exécuter.
2. Appel du système d'exploitation pour lancer les travaux nécessaires.
3. Mise à jour de leur copie du fichier Symphony avec les résultats de traitement de travail et envoi d'une notification au gestionnaire de domaine maître et à tous les agents tolérants aux pannes de statut Intégral. La copie originale du fichier Symphony stockée sur le gestionnaire de domaine maître et les copies stockées sur les gestionnaires de domaine maîtres de sauvegarde, si définies, sont mises en jour en conséquence.

Ceci signifie que pendant le traitement du travail, chaque agent tolérant aux pannes a sa propre copie du fichier Symphony mise à jour avec les informations sur les travaux qu'il exécute (ou qui sont exécutés dans son domaine et ses domaines enfants si le agent tolérant aux pannes est de statut Intégral ou est un gestionnaire de domaine). De plus le gestionnaire de domaine maître (et le gestionnaire de domaine maître de sauvegarde s'il est défini) a la copie du fichier Symphony qui

contient toutes les mises à jour venant de tous les agents tolérants aux pannes. Le fichier Symphony reste ainsi à jour sur le gestionnaire de domaine maître avec les travaux qui doivent être exécutés, ceux qui sont en cours d'exécution et ceux qui sont terminés.

Pour plus d'informations sur le traitement effectué sur chaque poste de travail concerné par les activités du plan de production courant, voir «Processus de poste de travail IBM Workload Scheduler», à la page 37.

Remarque : Les modifications apportées à l'aide de **conman** au plan de production en cours de traitement n'affectent pas les définitions dans la base de données, mais la réplique des données du plan dans la base de données est mise à jour conformément aux modifications. Les modifications des objets dans la base de données n'affectent pas le plan tant qu'il n'est pas étendu ou recréé à l'aide du script **JnextPlan** ou de l'interface de ligne de commande **planman**. Les mises à jour des objets de la base de données n'affectent pas les instances de ces objets résidant déjà dans le plan de production.

Plan de préproduction

Le plan de préproduction permet d'identifier à l'avance les instances de flot de travaux et les dépendances des flots de travaux intervenant pendant la période indiquée.

Cette approche améliore les performances lors de la génération du plan de production en préparant à l'avance un calendrier à haut niveau de la charge de travail de production prévue.

Le plan de préproduction contient :

- Les instances de flot de travaux à exécuter pendant l'intervalle de temps couvert.
- Les dépendances de prédécesseur/successeur externes entre les flots de travaux et les travaux des différents flots.

Un travail ou un flot de travaux qui ne peut pas démarrer avant qu'un autre travail ou flot de travaux ne soit terminé est nommé *successeur*. Un travail ou un flot de travaux externe qui doit se terminer avant que le travail ou le flot de travaux successeur ne puisse démarrer est nommé *prédécesseur*.

IBM Workload Scheduler génère, étend et met à jour, si nécessaire, le plan de préproduction automatiquement en procédant comme suit :

- Il supprime les instances de flot de travaux à l'état COMPLETE et CANCEL.
- Sélectionne tous les flots de travaux planifiés après la fin du plan de production en cours et génère leurs instances.
- Il résout toutes les dépendances de travaux et de flots de travaux, y compris les dépendances de prédécesseur/successeur externes, en fonction des critères de correspondance définis.

Pour éviter les conflits, la base de données est verrouillée pendant la génération du plan de préproduction et elle est déverrouillée lorsque la génération est terminée ou lorsqu'une erreur se produit.

A cette étape, seuls les flots de travaux, l'heure à laquelle ils doivent commencer et leurs dépendances sont mis en évidence. Les autres informations concernant les flots de travaux et les autres objets de planification (agendas, invites, domaines, postes de travail, ressources, fichiers et utilisateurs) qui seront concernés par le

plan de production pour cette période ne sont pas incluses. Elles seront cependant extraites de la base de données au début de la génération du plan de production.

Lors de l'extension du plan de production, les anciennes instances de flot de travaux sont automatiquement retirées. Les critères utilisés pour retirer ces instances prennent en compte les informations suivantes :

- La première instance de flot de travaux qui n'est pas à l'état COMPLETE au moment de la génération du nouveau plan (FNCJSI). Cette instance de flot de travaux peut être planifiée, à savoir ajoutée au plan lorsque le plan de production est généré. Elle peut aussi être soumise à partir de la ligne de commande pendant la production à l'aide de la commande **conman sbs**.
- La période entre le démarrage prévu de FNCJSI et l'heure de fin du plan de production précédent.

En supposant que cette période est **T**, l'algorithme utilisé pour définir les instances de flot de travaux à retirer du plan de préproduction est le suivant :

si $T < 7$

Toutes les instances de flot de travaux datant de plus de 7 jours avant l'heure de début du nouveau plan de production sont retirées du plan de préproduction ; toutes les instances de flot de travaux de moins de 7 jours avant cette heure sont conservées quel que soit leur état.

si $T > 7$

Toutes les instances de flot de travaux antérieures à FNCJSI sont retirées du plan de préproduction et toutes les instances de flot de travaux ultérieures sont conservées.

Cet algorithme est utilisé pour éviter un accroissement continu de la taille du plan de préproduction et, par la même occasion, pour éviter la suppression d'une instance de flot de travaux susceptible de précéder un flot de travaux récemment ajouté au nouveau plan de préproduction.

Pour plus d'informations sur la façon d'ouvrir le plan de préproduction en mode vue à partir de Dynamic Workload Console, voir Dynamic Workload Console - Guide d'utilisation, section "Affichage du plan de préproduction".

Remarque : Dans la terminologie d'IBM Workload Scheduler for z/OS, le concept qui correspond au plan de préproduction est le *plan à long terme* (LTP).

Identification des instances de flot de travaux dans le plan

Dans les versions antérieures à la version 8.3 le plan avait une durée fixe d'un jour. Depuis la version 8.3 le plan peut couvrir une période durant plusieurs jours ou moins d'un jour. Ce changement a ajouté la possibilité d'avoir dans le même plan plus d'une instance du même flot de travaux avec le même nom et également la nécessité de définir une nouvelle convention pour identifier de façon unique chaque instance de flot de travaux dans le plan. Chaque instance de flot de travaux est identifiée dans le plan par les valeurs suivantes :

poste de travail

Indique le nom du poste de travail sur lequel l'exécution du flot de travaux est planifiée.

nom_flot_travaux

Nom du flot de travaux utilisé dans les versions antérieures de IBM Workload Scheduler.

scheddateandtime

Date et heure de démarrage prévues de l'instance de flot de travaux dans le plan de préproduction. Elles correspondent au jour indiqué par une clause **on** dans le cycle d'exécution dans la définition du flot de travaux et à l'heure indiquée par un mot clé **at** ou **schedtime** dans la définition du flot de travaux. S'il est défini, le mot clé **schedtime** est utilisé uniquement pour mettre dans l'ordre chronologique les instances de flot de travaux dans le plan de préproduction, alors que le mot clé **at**, s'il est défini, représente également une dépendance pour le flot de travaux. Pour plus d'informations sur ces mots clés, voir «on», à la page 267, «at», à la page 233 et «schedtime», à la page 279.

A partir de ces deux valeurs, que vous pouvez spécifier dans la définition du flot de travaux, IBM Workload Scheduler génère et affecte un identificateur alphanumérique unique à chaque instance de flot de travaux, à savoir *id_flot_travaux*, pour son usage interne. Pour plus d'informations relatives au format de l'*ID_flot_travaux*, voir «showjobs», à la page 501.

Vous pouvez utiliser au choix les deux types d'identificateur *poste_de_travail#nom_flot_travaux* et *date_et_heures_prévues* au lieu de *poste_de_travail#id_flot_travaux*, pour identifier de façon unique une instance de flot de travaux lorsque vous gérez des flots de travaux dans le plan avec le programme de ligne de commande **conman**. La convention par défaut utilisée pour identifier une instance de flot de travaux, à la fois dans le présent guide et dans les interfaces de ligne de commande du produit, est celle qui utilise *poste_de_travail_nom_flot_travaux* et *date_et_heures_prévues*. Pour plus d'informations sur la définition d'une instance de flot de travaux dans une commande à l'aide de **conman**, voir «Sélection de flots de travaux dans les commandes», à la page 428.

Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux

Lors de la création du plan de préproduction, toutes les dépendances externes de prédécesseur/successeur vers des flots de travaux et des travaux sont résolues avec quatre *critères de correspondance* possibles :

Même jour

Le programme prend en compte les instances de travail ou de flot de travaux qui doivent s'exécuter le même jour. Dans ce cas, la clause **follows...sameday** est indiquée dans la définition de l'objet. La figure 7 montre un flot de travaux nommé Js1 doté d'une dépendance externe de prédécesseur/successeur avec l'instance du flot de travaux Js2 planifiée pour démarrer le même jour.

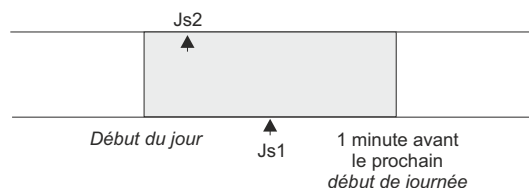


Figure 7. Instances devant s'exécuter le même jour utilisées comme critère de correspondance

Voici un exemple décrivant la définition des flots de travaux impliqués.

```

schedule Js2
on everyday
at 0700
;job2
end

```

```

schedule Js1
on everyday
at 1000
follows wk1#Js2 sameday
;job1
end

```

Le flot de travaux Js1 n'est pas lancé tant que l'instance de flot de travaux Js2 sur le poste de travail wk1 n'a pas abouti.

Valeur précédente la plus proche

Utilisation de l'instance de flot de travaux ou de travail précédente la plus proche (plus tôt ou simultanément). L'instance de travail ou de flot de travaux utilisée par IBM Workload Scheduler pour résoudre la dépendance est la plus proche dans le temps précédant l'instance qui inclut la dépendance. Dans ce cas, la clause **follows ... previous** est indiquée dans la définition de l'objet. La figure 8 illustre un flot de travaux nommé Js1 doté d'une dépendance de prédécesseur/successeur externe provenant de l'instance précédente la plus proche du flot de travaux Js2. La fenêtre de temps dans laquelle le prédécesseur est recherché apparaît en gris dans la figure.

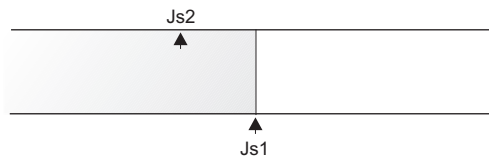


Figure 8. Instance précédente la plus proche utilisée comme critère de correspondance

Voici un exemple décrivant la définition des flots de travaux impliqués.

```

schedule Js2
on Th
at 0700
;job2
end

```

```

schedule Js1
on Fr
at 1000
follows wk1#Js2 previous
;job1
end

```

Le flot de travaux Js1 n'est pas lancé tant que l'instance de flot de travaux Js2 précédente la plus proche sur le poste de travail wk1 n'a pas abouti.

Dans un intervalle relatif

Prend en compte les instances de travail ou de flot de travaux définies dans une plage avec un décalage relatif à l'heure de début du travail ou du flot de travaux dépendant. Par exemple, de -25 heures avant l'heure de début du flot de travaux dépendant jusqu'à -5 heures après l'heure de début du flot de travaux dépendant. Dans ce cas, la clause **follows ... relative from ... to ...** est indiquée dans la définition de l'objet. La figure 9, à la page 73 montre un flot de travaux nommé Js1 doté d'une dépendance de prédécesseur/successeur externe sur l'instance de flot de travaux Js2 qui commence avec un décalage de 2 heures par rapport à Js1. L'instance de travail ou de flot de travaux prise en compte par IBM Workload Scheduler pour résoudre la dépendance est la plus proche dans l'intervalle de temps relatif que vous avez choisi.

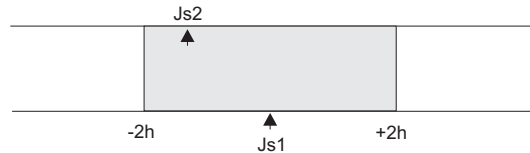


Figure 9. Instances définies dans un intervalle relatif utilisées comme critère de correspondance

Voici un exemple décrivant la définition des flots de travaux impliqués.

```

schedule Js2
on everyday
at 0900
;job2

end

```

```

schedule Js1
on everyday
at 1000
follows wk1#Js2 relative from 0200 to
0200
;job1

end

```

Le flot de travaux Js1 n'est pas lancé tant que l'instance de flot de travaux Js2 du poste de travail wk1, s'exécutant entre 8h et 12h, n'a pas abouti.

Dans un intervalle absolu

Le programme utilise uniquement les instances de travail ou de flot de travaux définies dans un intervalle, par exemple entre aujourd'hui 6h et après-demain 5h59. Dans cas, la clause **follows ... from ... to ...** est indiquée dans la définition de l'objet. La figure 10 illustre un flot de travaux nommé Js2 qui possède une dépendance de prédécesseur/successeur externe sur l'instance du flot de travaux Js1 positionnée dans le plan de préproduction entre 7:00 et 9:00. L'instance de travail ou de flot de travaux prise en compte par IBM Workload Scheduler pour résoudre la dépendance est la plus proche dans l'intervalle de temps absolu que vous avez choisi. L'intervalle de temps indique le moment de la journée auquel l'intervalle commence et se termine, le même jour que l'instance qui inclut la dépendance ou un jour défini relatif à ce jour.

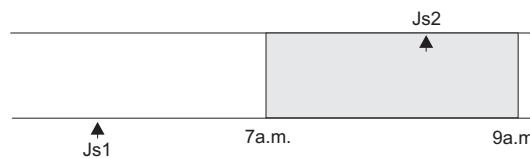


Figure 10. Instances définies dans un intervalle absolu utilisées comme critère de correspondance

Voici un exemple décrivant la définition des flots de travaux impliqués.

```

schedule Js1
on everyday
at 0800
;job1

end

```

```

schedule Js2
on everyday
at 1000
follows wk1#Js1 from 0700 to 0900
;job2

end

```

Le flot de travaux Js2 n'est pas lancé tant que l'instance de flot de travaux Js1 sur le poste de travail wk1, s'exécutant entre 7:00 et 9:00 le même jour, n'a pas abouti.

Quels que soient les critères de correspondance utilisés, si plusieurs instances d'un flot de travaux predecessor potentielle existent dans l'intervalle de temps spécifié, la règle utilisée par le produit pour identifier l'instance predecessor correcte est la suivante :

1. IBM Workload Scheduler recherche l'instance la plus proche précédant l'heure de début du travail ou du flot de travaux dépendant. Si une telle instance existe, il s'agit de l'instance prédécesseur.
2. S'il n'y a pas d'instance précédente, IBM Workload Scheduler considère l'instance prédécesseur correcte comme étant l'instance la plus proche qui commence après l'heure de début du travail ou du flot de travaux dépendant.

Ce comportement s'applique aux dépendances externes de prédécesseur/successeur entre les flots de travaux. Pour les dépendances externes de prédécesseur/successeur d'un flot de travaux ou d'un travail avec un autre travail, le programme met les critères en correspondance en prenant en compte l'heure de début du flot de travaux hébergeant le travail remplacé, au lieu de l'heure de début de ce travail proprement dit. La figure 11 indique en gras les instances de job1 dont dépend le travail ou le flot de travaux remplaçant.



Figure 11. Travail remplacé le plus proche

Les dépendances externes de prédécesseur/successeur sont identifiées entre les travaux et les flots de travaux stockés dans la base de données et dont les instances sont ajoutées au plan de préproduction lors de sa création ou de son extension automatique. Les instances de travail et de flot de travaux soumises en production à partir de la ligne de commande **conman** sont écrites dans le plan de préproduction mais ne sont pas utilisées pour recalculer les prédécesseurs des dépendances externes de prédécesseur/successeur déjà résolues dans le plan de préproduction.

L'agenda classe les dépendances de prédécesseur/successeur comme *internes* lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le format `jobStreamName.workstationName.jobName`.

Lorsqu'un flot de travaux inclut un travail possédant une dépendance de prédécesseur/successeur qui partage le même nom de flot de travaux (par exemple, le flot de travaux schedA inclut un travail appelé job6 qui possède une dépendance de prédécesseur/successeur sur schedA.job2), la dépendance est ajoutée au plan en tant que dépendance de prédécesseur/successeur *externe*. Depuis la version 8.3, contrairement aux versions précédentes, étant donné que l'agenda utilise le critère de correspondance *sameday* pour résoudre des dépendances externes, les dépendances générées de cette manière ne sont jamais ajoutées lors de la première soumission de l'objet.

Un travail ou un flot de travaux qui n'est pas encore inclus dans le plan de production mais constitue un prédécesseur potentiel d'instances de travaux et de flots de travaux ajoutés au plan de production lors de son extension, est appelé *prédécesseur suspendu*. Un prédécesseur suspendu s'apparente à une occurrence factice créée par le processus de planification pour respecter une dépendance qui a été résolue dans le plan de préproduction, mais qui ne peut pas l'être dans le plan de production en cours parce que l'heure de début du prédécesseur n'est pas antérieure à l'heure de fin du plan de production en cours. La figure 12 montre comment un prédécesseur suspendu et son successeur sont positionnés dans le plan de préproduction.

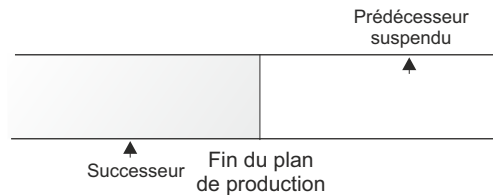


Figure 12. Instance de prédécesseur suspendu

La gestion des prédécesseurs suspendus est étroitement liée au report ou non du travail ou flot de travaux remplaçant dans le plan suivant :

- Si le successeur est reporté lors de l'extension du plan de production, le prédécesseur est inclus dans le nouveau plan de production et la dépendance devient valide. Un travail de prédécesseur ou un flot de travaux suspendu est indiqué par un [P] dans la colonne Dépendances de la sortie des commandes conman showjobs et conman showschedules.
- Si le successeur n'est pas reporté lors de l'extension du plan de production, le prédécesseur est inclus dans le nouveau plan de production mais la dépendance devient *orpheline*. Ce cas peut se présenter, par exemple, si lors de l'extension du plan de production, le successeur est reporté et le prédécesseur suspendu n'est pas ajouté au plan car il a été marqué *en projet* dans la base de données. Les dépendances orphelines sont indiquées par un [0] dans la colonne Dependencies de la sortie de la commande conman showjobs. Lorsque vous avez affaire à une dépendance orpheline, vous devez vérifier si elle peut être libérée et l'annuler, si c'est le cas.

Notez que, lorsqu'un réseau IBM Workload Scheduler comporte des agents s'exécutant sur des versions IBM Workload Scheduler antérieures à 8.3 et que la valeur yes est attribuée à l'option enLegacyId sur le gestionnaire de domaines maîtres, la présence de plusieurs instances d'un flot de travaux en tant que prédécesseurs suspendus génère des erreurs suite aux incidents d'identification au moment de la soumission.

Résolution de dépendances de prédécesseur/successeur et exemples de transition d'état

Cette section comprend des exemples de chacun des quatre critères de correspondance décrits dans les paragraphes précédents. Dans tous les exemples, le début de l'heure de journée est défini à 06:00 AM.

Même jour

L'instance du travail ou du flot de travaux à prendre en compte dans la résolution de la dépendance est le plus proche le jour où l'instance qui inclut la dépendance est planifiée pour s'exécuter. Dans cet exemple, deux

flots de travaux, Js1 et Js2, ont chacun un travail. Le flot de travaux Js1 est planifié pour s'exécuter tous les jours à 08:00 et le jeudi également à 07:00. Js1.Job1 s'exécute à 09:00. Le flot de travaux Js2 n'a pas de restrictions de temps et est planifié par défaut à la première heure de journée définie. Js2.Job2 est planifié pour s'exécuter à 15:00 et possède une dépendance de prédécesseur/successeur externe sur l'instance précédente la proche du flot de travaux Js1 qui s'exécute le même jour. Les deux flots de travaux sont définis de la manière suivante :

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0700)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 0800)
:
MY_MASTER#JOB1
AT 0900
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE2 "FREQ=DAILY;"
FOLLOWS MY_MASTER#JS1.@ SAMEDAY
:
MY_MASTER#JOB2
AT 1500
END
```

Lorsque les calendriers sont inclus dans le plan, la séquence de graphiques illustre comment la dépendance est résolue :

1. Le jeudi, l'instance du flot de travaux Js2 planifié à 06:00 dépend de l'instance de Js1 planifié pour s'exécuter à 07:00. Tous les autres jours de la semaine, Js2 possède une dépendance sur l'instance de Js1 planifiée à 08:00. La figure 13 présente l'état des deux flots de travaux du plan à 06:00 (heure de début de journée) le jeudi :

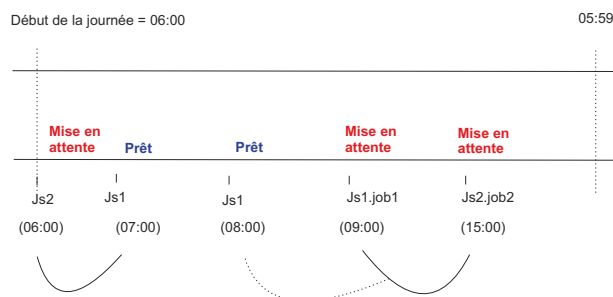


Figure 13. Critères de correspondance du même jour - Etape 1 : Au début de la journée le jeudi

2. A 09:00, Js1.job1 commence et Js1 change d'état. Js2.job2 est mis en attente jusqu'à son heure planifiée. La figure 14, à la page 77 présente l'état des flots de travaux dans le plan à 09:00.

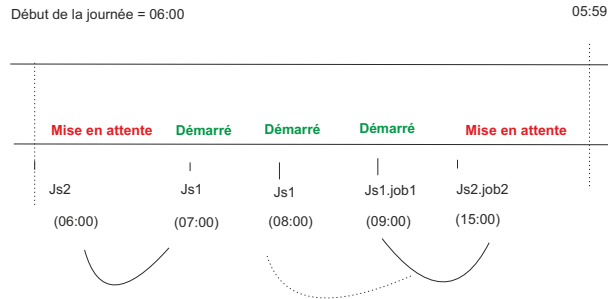


Figure 14. Critères de correspondance du même jour - Etape 2 : à 9:00

3. Le jeudi à 15:00, Js2 passe à l'état prêt et Js2.job2 commence. La figure 15 présente l'état des deux flots de travaux du plan à 15:00.

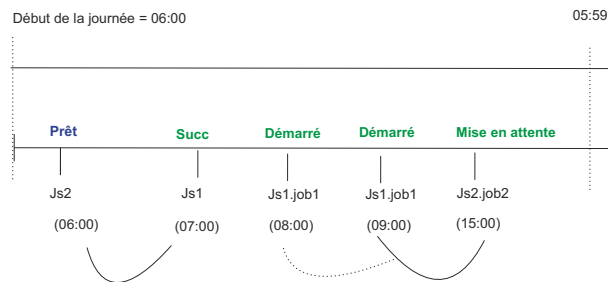


Figure 15. Critères de correspondance du même jour - Etape 3 : à 15:00

Valeur précédente la plus proche

- Dans cet exemple, deux flots de travaux, Js1 et Js2, ont chacun un travail. Le travail dans Js2 possède une dépendance de prédécesseur/successeur externe sur l'instance précédente la plus proche du travail dans Js1. Les deux flots de travaux sont définis de la manière suivante :

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=DAILY;"
(AT 0800)
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=TH,FR"
(AT 0900)
:
MY_MASTER#JOB1
END
```

```
SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE1 "FREQ=DAILY;"
(AT 1200)
FOLLOWS MY_MASTER#JS1.@ PREVIOUS
:
MY_MASTER#JOB2
AT 1500
END
```

Le flot de travaux Js1 s'exécute tous les jours à 0800 et également le jeudi et le vendredi à 0900. Le flot de travaux Js2 s'exécute tous les jours à 1200, et possède une dépendance externe sur l'instance précédente la plus proche de Js1. Lorsque les flots de travaux sont inclus dans le plan, la séquences de graphiques illustre comment la dépendance est résolue :

1. Avant 12:00 le jeudi et le vendredi, il y a deux instances de Js1.Job1. Le flot de travaux Js2 possède une dépendance sur l'instance de

Js1.Job1 qui est planifiée pour s'exécuter à 09:00, car il s'agit de l'instance précédente la plus proche en terme de temps. La figure 16 présente l'état des deux flots de travaux du plan le jeudi et le vendredi.



Figure 16. Critères de correspondance précédents les plus proches - Etape 1 : avant 08:00

2. Tous les autres jours de la semaine, la seule instance de Js1.Job1 dans le plan est celle planifiée pour s'exécuter à 08:00. Dans ce cas, Js2 possède une dépendance sur cette instance. Lorsque Job1 aboutit, l'état de Js2 devient **Ready** (prêt). La figure 17 présente l'état des deux flots de travaux du plan tous les autres jours de la semaine à l'exception du jeudi et du vendredi.

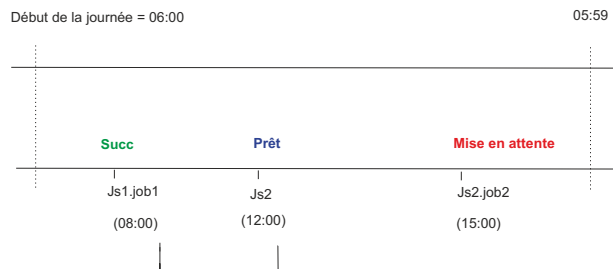


Figure 17. Critères de correspondance précédents les plus proches - Etape 2 : à 8h les jours de la semaine excepté le jeudi et le vendredi

3. Le jeudi et le vendredi à 09:00, la seconde instance de Js1.Job1 s'exécute avec succès. Le flot de travaux Js2 passe à **Ready**. Js2.Job2 est mis en attente jusqu'à son heure de début planifiée. La figure 18 présente l'état des deux flots de travaux dans le plan.

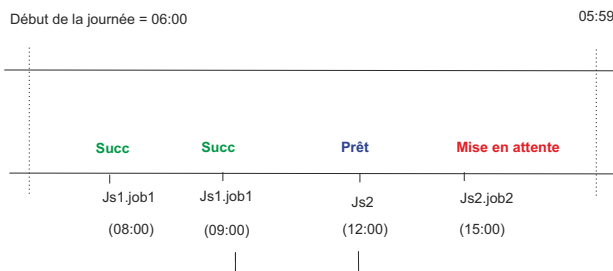


Figure 18. Critères de correspondance précédents les plus proches - Etape 3 : à 09:00 le jeudi et le vendredi

4. A 15:00, la dépendance horaire de Js2.Job2 est satisfaite et Job2 démarre. La figure 19, à la page 79 présente l'état des deux flots de

travaux dans le plan à 15:00.

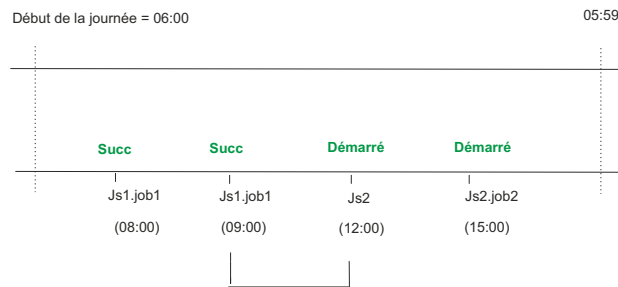


Figure 19. Critères de correspondance précédents les plus proches - Etape 4 : à 15:00 tous les jours

Dans la définition de flot de travaux, le cycle d'exécution *Rule1* peut être remplacé par les mots clés ON EVERYDAY.

- Dans ce second exemple, est décrite la différence entre l' utilisation de sameday et closest preceding correspondant à des critères dans un plan. Le flot de travaux Js1 s'exécute tous les vendredi à 09h00, tandis que les flots de travaux Js2 et Js3 s'exécutent tous les samedi à 09h00. Les trois flots de travaux sont définis comme suit :

```
SCHEDULE ACCOUNTING#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=FR"
:
ACCOUNTING#JOB1
AT 0900
END

SCHEDULE ACCOUNTING#JS2
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=SA"
FOLLOWS ACCOUNTING#JS1.@ PREVIOUS
:
ACCOUNTING#JOB1
AT 0900
END

SCHEDULE ACCOUNTING#JS3
ON RUNCYCLE RULE2 "FREQ=WEEKLY;BYDAY=SA"
FOLLOWS ACCOUNTING#JS1.@
:
ACCOUNTING#JOB1
AT 0900
FIN
```

Le flot de travaux Js2 possède une dépendance externe sur l'instance précédente la plus proche de Js1, qui est résolue comme décrit dans l'exemple précédent. Le flot de travaux Js3 est défini par le critère sameday, ainsi, il ne possède pas de dépendance avec le flot de travaux Js1, puisque Js1 n'est pas défini pour s'exécuter le même jour que Js2.

Dans un intervalle relatif

Dans cet exemple, l'instance de travail ou de flot de travaux qui résout la dépendance est la plus proche, dans un intervalle de temps de votre choix, définie relativement à l'heure à laquelle l'instance qui inclut la dépendance est planifiée pour s'exécuter. Le flot de travaux Js1 est planifié pour s'exécuter tous les jours à 15:00 et également le jeudi à 08:00. Js2 est planifié pour s'exécuter tous les jours à 13:00 et également le jeudi à 06:00 ; étant donné qu'aucune heure spécifique n'est définie dans le cycle

d'exécution, il est planifié au début de la journée. Js2 utilise le critère d'intervalle relatif (-04:00 à +04:00) pour déterminer l'instance qui est utilisée pour résoudre la dépendance. L'intervalle est fonction de l'heure à laquelle le flot de travaux entre dans le plan. Les flots de travaux sont définis comme suit :

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0800)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 1500)
:
MY_MASTER#JOB1
END
```

```
SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE3 "FREQ=WEEKLY;BYDAY=TH"
ON RUNCYCLE RULE2 "FREQ=DAILY;"
(AT 1300)
FOLLOWS MY_MASTER#JS1.@
RELATIVE FROM -0400 TO 0400
:
MY_MASTER#JOB2
AT 1300
END
```

Lors de la création du plan, **conman showjobs** génère le résultat suivant :

```
%sj @#@
(Est) (Est)
CPU      Schedule SchedTime Job   State Pr Start Elapse RetCode Deps
MY_MASTER#JS1 0800 11/13 ***** READY 10 (00:06)
              JOB1    HOLD 10 (00:06)
MY_MASTER#JS1 1500 11/13 ***** READY 10 (00:06)
              JOB1    HOLD 10 (00:06)
MY_MASTER#JS2 0600 11/13 ***** HOLD 10 JS1(0800 11/13/09).@
              JOB2    HOLD 10(13:00)
MY_MASTER#JS2 1300 11/13 ***** HOLD 10(13:00) JS1(1500 11/13/09).@
              JOB2    HOLD 10(13:00)
```

La figure 20 présente l'état des flots de travaux du plan au début de la journée le jeudi.

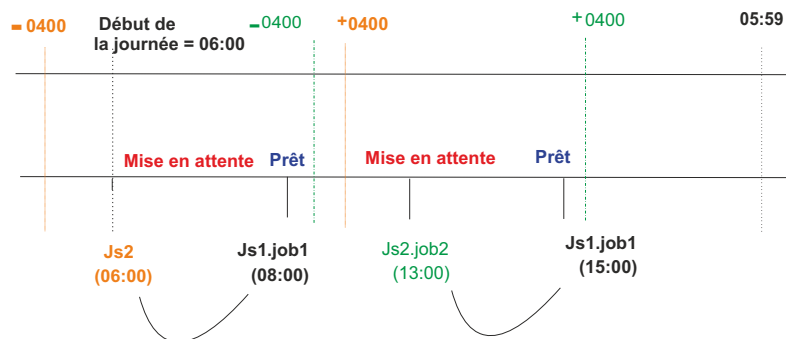


Figure 20. Critères de correspondance d'intervalle relatif - au début de la journée le jeudi

L'instance de Js2 planifié à 06:00 possède une dépendance sur Js1.job1 qui est planifié à 08:00, dans l'intervalle relatif basé sur l'heure planifiée (06:00). Js2.job2 dépend de l'instance de Js1.job1 dans l'intervalle relatif basé sur l'heure planifiée (13:00). Lorsque l'instance de Js1.job1 commence à 08:00, l'état de Js2 passe à Ready (prêt). A partir de ce point, la séquence dans laquelle s'exécutent les travaux et flots de travaux suivent le processus classique.

Dans un intervalle absolu

Dans cet exemple, l'instance du travail ou du flot de travaux prise en compte pour résoudre la dépendance est la plus proche dans un intervalle de temps fixe de votre choix. L'intervalle de temps indique le moment de la journée auquel l'intervalle commence et se termine, le même jour que l'instance qui inclut la dépendance ou un jour défini relatif à ce jour. Js1 est planifié pour s'exécuter tous les jours à 08:00 et également le jeudi à 07:00. Le travail Js1.job1 est planifié pour s'exécuter à 09:00. Le flot de travaux Js2 est planifié tous les jours à 10:00 et également le jeudi au début de la journée (06:00) et possède une dépendance sur Js1 en fonction d'un intervalle absolu ayant lieu le même jour entre 06:00 et 11:00. Les flots de travaux sont définis comme suit :

```
SCHEDULE MY_MASTER#JS1
ON RUNCYCLE RULE1 "FREQ=WEEKLY;BYDAY=TH"
(AT 0700)
ON RUNCYCLE RULE2 "FREQ=DAILY"
(AT 0800)
:
MY_MASTER#JOB1
AT 0900
END

SCHEDULE MY_MASTER#JS2
ON RUNCYCLE RULE3 "FREQ=WEEKLY;BYDAY=TH"
ON RUNCYCLE RULE2 "FREQ=DAILY;"
(AT 1000)
FOLLOWS MY_MASTER#JS1.@ FROM 0600 TO 1100
:
MY_MASTER#JOB2
AT 1300
END
```

Lors de la création du plan, **conman showjobs** génère le résultat suivant :

```
%sj @#@
              (Est) (Est)
CPU      Schedule SchedTime Job   State Pr Start Elapse RetCode Deps
MY_MASTER#JS1  0700 11/13***** READY 10      (00:06)
              JOB1    HOLD 10(09:00)(00:06)
MY_MASTER#JS1  0800 11/13***** READY 10      (00:06)
              JOB1    HOLD 10(09:00)(00:06)
MY_MASTER#JS2  0600 11/13***** HOLD 10      JS1(0700 11/13/09).@
              JOB2    HOLD 10(15:00)
MY_MASTER#JS2  1000 11/13***** HOLD 10(10:00) JS1(0800 11/13/09).@
              JOB2    HOLD 10(15:00)
```

La figure 21, à la page 82 présente l'état des flots de travaux du plan au début de la journée le jeudi.

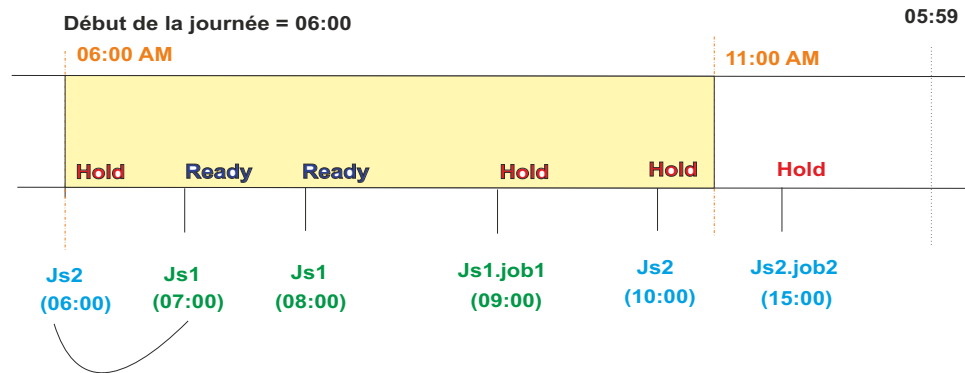


Figure 21. Critères de correspondance d'intervalle absolu - au début de la journée le jeudi

A 09:00, Js1.job1 commence et à 10:00 la dépendance est libérée et Js2 devient prêt. A partir de ce point, la séquence est la même que celle décrite dans les critères de correspondance précédents.

Plan de production

Après avoir créé ou mis à jour le plan de préproduction, IBM Workload Scheduler complète les informations stockées dans ce plan par des informations stockées dans la base de données sur les opérations à réaliser dans la période sélectionnée et les autres objets de planification concernés par le traitement du plan, puis le copie dans un nouveau fichier Symphony. Il ajoute également à ce fichier les dépendances entre les plans, comme les flots de travaux reportés à partir du plan de production déjà traité, puis il archive l'ancien fichier Symphony dans le répertoire schedlog.

A la fin de ce processus, le nouveau fichier Symphony contient toutes les informations permettant d'implémenter le nouveau plan de production. De plus, toutes les données du plan sont répliquées dans la base de données afin de faciliter les demandes émises à partir de Dynamic Workload Console et des API.

Une copie du fichier Symphony est ensuite distribuée à tous les postes de travail concernés par l'exécution de travaux ou de flots de travaux pour ce plan de production.

Dans le fichier de sécurité, l'autorisation utilisateur nécessaire pour générer le plan de production est le mot clé d'accès *build* sur les fichiers prodsked et Symphony.

Remarque : Pour éviter le manque d'espace disque, n'oubliez pas que chaque instance de travail ou de flot de travaux augmente la taille du fichier Symphony de 512 octets.

Pour plus d'informations sur la génération du plan de production, voir «Création et extension d'un plan de production», à la page 91.

Options de report

Les flots de travaux sont reportés lors de la génération du plan de production. Le report des flots de travaux dépend des éléments suivants :

- Mot clé **carryforward** du flot de travaux. Voir «carryforward», à la page 235.
- Option globale **enCarryForward**. Voir *IBM Workload Scheduler - Guide d'administration*.

- Mot clé de ligne de commande **stageman -carryforward**. Voir «Commande stageman», à la page 107.
- Option globale **carryStates**. Voir *IBM Workload Scheduler - Guide d'administration*.

Si un travail est un cours d'exécution lorsque le plan de production est généré, et que le flot de travaux qui le contient n'est pas reporté, le travail poursuit son exécution et il est placé dans un flot de travaux dédié appelé USERJOBS dans le nouveau plan de production.

Le tableau 12 montre comment les options globales de report fonctionnent ensemble.

Tableau 12. Paramètres des options globales de report

Options globales	Opération de report
enCarryForward=all carryStates=()	Les flots de travaux sont reportés uniquement s'ils ne sont pas terminés. Tous les travaux sont reportés avec les flots de travaux. Cette option est définie par défaut.
enCarryForward=no	Aucun flot de travaux n'est reporté. Si cette option est réglée sur "Non", les travaux en cours d'exécution sont déplacés vers le flot de travaux USERJOBS.
enCarryForward=yes carryStates=(states)	Les flots de travaux sont reportés uniquement s'ils ont des travaux dans les états indiqués et que le mot clé carryforward est présent dans la définition de flot de travaux. Seuls les travaux se trouvant dans les états indiqués sont reportés avec les flots de travaux.
enCarryForward=yes carryStates=()	Les flots de travaux sont reportés uniquement s'ils ne sont pas terminés et que le mot clé carryforward est présent dans la définition de flot de travaux. Tous les travaux sont reportés avec les flots de travaux.
enCarryForward=all carryStates=(states)	Les flots de travaux ne sont reportés que s'ils comprennent des travaux dans les états indiqués. Seuls les travaux se trouvant dans les états indiqués sont reportés avec les flots de travaux.

Le tableau 13 montre le résultat des paramètres de report selon l'option globale **enCarryForward** et les mots clés **stageman -carryforward**.

Tableau 13. Paramètres de report résultants

enCarryForward	stageman -carryforward	Paramètre de report résultant
NON	YES	NON
NON	ALL	NON
YES	NON	NON
ALL	NON	NON
ALL	YES	ALL
YES	ALL	ALL
YES	YES	YES

L'option de report présente dans la définition du flot de travaux est permanente. En d'autres termes, un flot de travaux en échec marqué **carryforward** est continuellement reporté jusqu'à ce qu'un des événements ci-dessous se produise :

- Il se termine à l'état SUCC

- Son heure UNTIL est atteinte
- Il est annulé.

La convention de nom de flot de travaux exécutée est affectée par la valeur attribuée à l'option globale *enLegacyId*. Pour davantage d'informations sur les différents paramètres de cette option, voir «Personnalisation de la gestion du plan à l'aide des options globales», à la page 86.

Remarque : Quelles que soient les options de report définies, les flots de travaux qui ne contiennent pas de travaux ne sont pas reportés.

Si vous définissez **carryStates=(succ)** et **enCarryForward=all** ou **enCarryForward=yes**, le plan de préproduction et le nouveau fichier Symphony ne correspondront pas lors de la prochaine exécution de **JnextPlan**. Effectivement, le plan de préproduction ne contient pas les instances des flots de travaux qui se sont terminées correctement, contrairement au nouveau fichier Symphony. Suite à cette incohérence, les dépendances ne sont pas résolues en fonction des instances des flots de travaux reportées terminées correctement car celles-ci n'existent plus dans le plan de préproduction.

La décision de reporter un travail répétitif, c'est-à-dire un travail dont la définition contient un paramètre de temps **every** ou une chaîne de travaux réexécutés, repose sur l'état de son exécution la plus récente. Seuls les premier et dernier travaux de la chaîne sont reportés.

Plan d'essai

Un plan d'essai est une prévision de ce que serait un plan de production s'il couvrait une période plus longue. Par exemple, si vous générez un plan de production couvrant deux jours mais voulez savoir à quoi il ressemblerait sur trois jours, vous pouvez créer un plan d'essai.

Voici les caractéristiques d'un plan d'essai :

- Sa date de début concorde avec :
 - La date de début du plan de préproduction.
 - La date de fin du plan de production.
- Il s'appuie sur les informations statiques stockées dans le plan de préproduction courant.
- Il ne peut pas être exécuté pour gérer la production.
- Il peut être géré par des utilisateurs bénéficiant des droits d'accès *build* sur le type d'objet fichier **trialsked** défini dans le fichier de sécurité du gestionnaire de domaine maître.
- Il produit un fichier qui est stocké dans le répertoire schedlog et qui a les propriétés suivantes :
 - Même format que le fichier Symphony.
 - Le nom de ce fichier commence par un T.

Les générations de plan d'essai peuvent rallonger l'heure de fin du plan de préproduction. Cette situation dépend des valeurs des options globales minLen et maxLen. Dans ce cas, la base de données est verrouillée jusqu'à la fin de l'opération.

Toute période peut être sélectionnée sans restriction pour un plan d'essai mais la taille du fichier résultant qui contient toutes les informations du plan d'essai doit être prise en compte.

Comme le plan d'essai est basé sur les informations statiques du plan de préproduction, il ne prend pas en compte les mises à jour effectuées de manière dynamique sur le fichier Symphony pendant le traitement du plan de production. Par conséquent, tous les flots de travaux qu'il contient sont dans un des deux états suivants :

HOLD

S'ils sont dépendants d'autres flots de travaux ou si leur heure de début est postérieure à l'heure de début du plan.

READY

S'ils n'ont aucune dépendance et que leur heure de début est dépassée.

Les opérations qui peuvent être effectuées sur un plan d'essai du gestionnaire de domaine maître sont les suivantes :

creation

Permet de créer un plan d'essai pour avoir un aperçu de la production lorsque le plan de production n'existe pas encore.

extension

Permet de créer un plan d'essai de l'extension du plan de production courant pour avoir un aperçu de l'évolution future de la production.

Pour plus d'informations sur la création ou l'extension d'un plan d'essai, voir «Ligne de commande Planman», à la page 94.

Plan prévisionnel

Le *plan prévisionnel* est une prévision de ce que serait le plan de production sur une période donnée. Par exemple, si vous avez généré un plan de production qui couvre deux journées et que vous souhaitez savoir ce que serait ce plan la semaine suivante, vous pouvez générer un plan prévisionnel.

Voici les caractéristiques d'un plan prévisionnel :

- Il couvre n'importe quelle période future, passée ou recouvrant partiellement le plan courant.
- Il s'appuie sur un échantillon du plan de préproduction qui couvre la même période que le plan prévisionnel. Cet échantillon du plan de préproduction est supprimé après la création du plan prévisionnel.
- Il ne peut pas être exécuté pour gérer la production.
- Il peut être géré par des utilisateurs bénéficiant des droits d'accès *build* sur le type d'objet fichier **trialsked** défini dans le fichier de sécurité du gestionnaire de domaine maître.
- Il produit un fichier qui est stocké dans le répertoire schedlog et qui a les propriétés suivantes :
 - Même format que le fichier Symphony.
 - Le nom de ce fichier commence par un F.
- Lorsque l'assurance de service de charge de travail est activée, elle peut calculer l'heure de début prévue de chaque travail du flot de travaux. Vous pouvez activer et désactiver cette fonction à l'aide de l'option globale **enForecastStartTime**. IBM Workload Scheduler calcule la durée d'exécution

moyenne de chaque travail en fonction de toutes les exécutions précédentes. Pour les plans complexes, activer cette fonction peut avoir un impact négatif sur le temps nécessaire pour générer le plan prévisionnel.

Pendant la création du plan prévisionnel, la base de données est verrouillée jusqu'à la fin de l'opération.

Toute période peut être sélectionnée sans restriction pour un plan prévisionnel mais la taille du fichier résultant qui contient toutes les informations du plan prévisionnel doit être prise en compte.

Comme le plan prévisionnel est basé sur les informations statiques de la base de données, il ne prend pas en compte les mises à jour effectuées de manière dynamique sur le fichier Symphony pendant le traitement du plan de production ou du plan de préproduction. Par conséquent, tous les flots de travaux qu'il contient sont dans un des deux états suivants :

HOLD

S'ils sont dépendants d'autres flots de travaux ou si leur heure de début est postérieure à l'heure de début du plan.

READY

S'ils n'ont aucune dépendance et que leur heure de début est dépassée.

La seule opération qui peut être effectuée sur un plan prévisionnel du gestionnaire de domaine maître est la suivante :

creation

Permet de créer un plan prévisionnel pour avoir un aperçu de la production à une période donnée.

Pour plus d'informations sur la création d'un plan prévisionnel, voir «Ligne de commande Planman», à la page 94.

Personnalisation de la gestion du plan à l'aide des options globales

Vous pouvez personnaliser certains critères à utiliser par IBM Workload Scheduler lors de la gestion des plans en définissant des options spécifiques dans le gestionnaire de domaine maître à l'aide du programme de ligne de commande **optman**. Vous devez générer le plan à nouveau pour activer les nouveaux paramètres. Vous pouvez personnaliser les options suivantes :

Propriétés ayant une incidence sur la génération du plan de préproduction :

minLen

Longueur minimale, en jours, du plan de préproduction laissé sous la forme de tampon à la fin du plan de production qui vient d'être généré. La valeur affectée à cette option est utilisée lors de l'exécution du script **UpdateStats** dans **JnextPlan**. La valeur peut être de 7 à 365 jours. La valeur par défaut est 8 jours.

maxLen

Longueur maximale, en jours, du plan de préproduction laissé sous la forme de tampon à la fin du plan de production qui vient d'être généré. La valeur peut être de 8 à 365 jours. La valeur par défaut est 14 jours.

Si les valeurs *minLen* et *maxLen* sont égales, le plan de préproduction est mis à jour lors de la phase **MakePlan**. En général,

la valeur de `maxLen` doit dépasser la valeur de `minLen` d'au moins 1 jour afin que le plan de préproduction puisse être mis à jour lors de la phase `UpdateStats`.

Propriétés ayant une incidence sur la génération ou l'extension du plan de production :

startOfDay

Heure de début de la journée de traitement de IBM Workload Scheduler au format 24 heures : hhmm (0000-2359). Le paramètre par défaut est 0000.

enCarryForward

Cette option affecte la façon dont la commande **stageman** reporte les flots de travaux. Sa valeur détermine si les flots de travaux non terminés sont reportés depuis l'ancien plan de production vers le nouveau. Les valeurs possibles de *enCarryForward* sont **yes**, **no**, et **all**. La valeur par défaut est **all**.

carryStates

Cette option affecte la façon dont la commande **stageman** gère les travaux des flots de travaux reportés. Sa valeur détermine selon leur état les travaux qui doivent être inclus dans les flots de travaux à reporter. Par exemple si :

```
carryStates='abend exec hold'
```

tous les travaux à l'état `abend`, `exec` ou `hold` sont inclus dans les flots de travaux reportés. La valeur par défaut est :

```
carryStates=null
```

ce qui signifie que tous les travaux sont inclus, quels que soient leurs états.

untilDays

Si une heure **until** (dernière heure de début) n'est pas indiquée pour un flot de travaux, alors une heure **until** par défaut est calculée en ajoutant la valeur de cette option, exprimée en nombre de jours, à l'heure planifiée pour le flot de travaux. Si l'option *enCarryForward* est définie sur **all**, et que le nombre de jours spécifié pour *untilDays* est atteint, alors toutes les instances de flot de travaux du plan qui ont échoué sont automatiquement supprimées du plan et ne sont pas ajoutées au nouveau plan de production. La valeur par défaut est 0. Si la valeur par défaut est utilisée, alors aucune heure par défaut n'est définie pour l'heure **until** (dernière heure de début).

enCFInterNetworkDeps

Cette option affecte la façon dont la commande **stageman** gère les dépendances interréseaux. Saisissez **yes** pour que tous les flots de travaux EXTERNES soient reportés. Saisissez **no** pour désactiver entièrement la fonction de report pour les dépendances interréseaux. La valeur par défaut est **yes**.

enCFResourceQuantity

Cette option affecte la façon dont la commande **stageman** gère les ressources. Lors de l'extension du plan de production, une des situations suivantes se présente :

- Une ressource utilisée par aucun flot de travaux reporté du plan de production précédent est référencée par de nouvelles

instances de travail ou de flot de travaux ajoutées au nouveau plan de production. Dans ce cas, la quantité de cette ressource figure dans la définition de ressource dans la base de données.

- Une ressource utilisée par un ou plusieurs flots de travaux reportés du plan de production précédent n'est référencée par aucune instance de travail ou de flot de travaux ajoutée au nouveau plan de production. Dans ce cas, la quantité de cette ressource figure dans l'ancien fichier Symphony.
- Une ressource utilisée par un ou plusieurs flots de travaux reportés du plan de production précédent est référencée par des instances de travaux ou de flots de travaux ajoutées au nouveau plan de production. Dans ce cas, la quantité de cette ressource dépend de la valeur affectée à l'option *enCFResourceQuantity* :

Si la valeur YES est attribuée à *enCFResourceQuantity*

La quantité de la ressource figure dans l'ancien fichier Symphony.

Si la valeur NO est attribuée à *enCFResourceQuantity*

La quantité de la ressource figure dans la définition de ressource dans la base de données.

La valeur par défaut est **yes**.

enEmptySchedsAreSucc

Cette option régit le comportement des flots de travaux qui ne contiennent pas de travaux. Les valeurs possibles sont les suivantes :

Yes	Les flots de travaux qui ne contiennent pas de travaux sont marqués SUCC lorsque leurs dépendances sont résolues.
No	Les flots de travaux qui ne contiennent pas de travaux restent à l'état READY.

enPreventStart

Cette option permet de gérer, pour les plans de production étalés sur plusieurs jours, les flots de travaux sans contrainte temporelle **at** définie. Elle est utilisée pour éviter que les instances de flot de travaux non soumises à une dépendance horaire ne démarrent toutes en même temps lors de la création ou de l'extension du plan de production. Les valeurs possibles sont les suivantes :

Yes	Un flot de travaux ne peut pas démarrer avant le début de la journée <i>startOfDay</i> défini dans son heure planifiée, même s'il n'a pas de dépendances.
No	Un flot de travaux peut démarrer immédiatement au lancement du plan de production si toutes ses dépendances sont résolues.

enLegacyId

Cette option affecte la manière de nommer les flots de travaux dans le plan. Elle a pour fonction de maintenir la cohérence lors de l'identification des flots de travaux dans le plan dans les environnements mixtes avec des versions de IBM Workload Scheduler antérieures à la 8.3 gérées par la version 8.4 des gestionnaires de domaine maîtres. Cette option n'est pas prise en charge par le catalogue libre-service, lequel l'ignore, même si sa valeur est définie sur OUI. La valeur affectée à cette option est lue

soit lors de la création ou de l'extension du plan de production, soit lors de la soumission des flots de travaux dans la production via la commande **conman**. Les valeurs possibles sont les suivantes :

Yes L'identificateur *ID_flot_de_travaux* du flot de travaux nommé *nom_flot_de_travaux* est défini sur la valeur *jobstream_nameN*, où *N* est un nombre incrémentiel attribué par un compteur interne. Ce paramètre vaut *null* s'il n'existe qu'une seule instance de ce flot de travaux dans le plan.

Si le flot de travaux intitulé *nom_flot_de_travaux* est reporté par la suite, son identificateur est défini sur la valeur *CFjobstream_nameN*.

Ce paramètre est utile pour assurer la cohérence de gestion des flots de travaux, même en cas de report de ceux-ci via la commande **conman**, lors d'une connexion à un agent IBM Workload Scheduler 8.2.x sur un réseau IBM Workload Scheduler doté d'un gestionnaire de domaine maître 8.4.

Si, notamment, la période de production dure une journée et qu'un même flot de travaux n'est pas soumis sous forme d'instances multiples, la compatibilité avec les versions antérieures est complète lors de la gestion de flots de travaux en mode production à partir d'un agent IBM Workload Scheduler version 8.2.x.

No L'identificateur de flot de travaux *ID_flot_de_travaux* est généré conformément à «showjobs», à la page 501. Les flots de travaux reportés conservent leurs noms et identificateurs d'origine, et signalent entre accolades ({}) la date de leur report.

logmanSmoothPolicy

Cette option affecte la façon dont la commande **logman** gère les statistiques et l'historique. Elle définit le facteur de pondération qui favorise le travail le plus récemment exécuté lors du calcul du délai d'exécution normal (moyen) d'un travail. Le résultat de ce calcul est exprimé sous la forme d'un pourcentage. La valeur par défaut est 10.

logmanMinMaxPolicy

Cette option permet de définir comment les temps d'exécution minimal et maximal des travaux sont consignés et signalés par **logman**. Les valeurs possibles de l'option *logmanMinMaxPolicy* sont les suivantes :

elapsedtime

Les temps d'exécution minimal et maximal et les dates consignés sont basés uniquement sur le temps écoulé d'un travail. L'activité du système affecte de façon significative le temps écoulé, exprimé en minutes. Ce temps inclut non seulement le temps pendant lequel un travail a utilisé l'UC, mais aussi le temps pendant lequel il a dû attendre que d'autres processus libèrent l'UC. Pendant les périodes d'activité système élevée, par exemple, le temps écoulé pour un travail a pu être long alors que le temps du poste de travail n'aura pas été plus long que dans les périodes

d'activité système basse. Les valeurs sont mises à jour uniquement si le temps écoulé de l'exécution de travail la plus récente est supérieur au maximum existant ou inférieur au minimum existant.

cputime

Les temps d'exécution minimal et maximal et les dates consignés sont basés uniquement sur le temps UC d'un travail. Le temps UC est une mesure en secondes du temps pendant lequel un travail a utilisé l'UC ; il ne prend pas en compte les attentes. Les valeurs sont mises à jour uniquement si le temps du poste de travail de l'exécution de travail la plus récente est supérieur au maximum existant ou inférieur au minimum existant.

both Le temps écoulé et le temps UC sont mis à jour indépendamment pour indiquer leurs valeurs maximales et minimales mais les dates d'exécution correspondent uniquement à la valeur du temps écoulé. Dans ce cas, aucun enregistrement des dates d'exécution n'est conservé pour les temps UC maximal et minimal.

La valeur par défaut est **both**.

enTimeZone

Cette option permet d'activer ou désactiver la gestion des fuseaux horaires sur le réseau IBM Workload Scheduler. Les valeurs possibles de l'option *enTimeZone* sont les suivantes :

- No** Désactivation de la gestion des fuseaux horaires. En d'autres termes, les valeurs attribuées à tous les mots clés **timezone** dans les définitions sont ignorées.
- Yes** Activation de la gestion des fuseaux horaires. En d'autres termes, les valeurs attribuées aux paramètres **timezone** sont utilisées pour calculer l'heure où les travaux et flots de travaux s'exécuteront sur les postes de travail cibles.

Voir «Activation de la gestion des fuseaux horaires», à la page 799 pour plus d'informations sur la variable *enTimeZone*.

enLegacyStartOfDayEvaluation

Cette option influe sur la manière dont la variable *startOfDay* est gérée sur le réseau IBM Workload Scheduler. Pour fonctionner, la valeur **yes** doit être attribuée à la variable *enTimeZone*. Les valeurs possibles de l'option *enLegacyStartOfDayEvaluation* sont les suivantes :

- No** La valeur affectée à l'option *startOfDay* sur le gestionnaire de domaine maître n'est pas convertie au fuseau horaire local défini sur chaque poste de travail du réseau.
- Yes** La valeur affectée à l'option *startOfDay* sur le gestionnaire de domaine maître est convertie au fuseau horaire local défini sur chaque poste de travail du réseau.

Voir «Comment IBM Workload Scheduler gère les fuseaux horaires», à la page 800 pour plus d'informations sur la variable *enLegacyStartOfDayEvaluation*.

Pour plus d'informations sur la définition des options à l'aide du programme de ligne de commande **optman**, voir *IBM Workload Scheduler - Guide d'administration*.

Création et extension d'un plan de production

L'ensemble du processus qui permet de passer d'un ancien plan de production à un nouveau, y compris l'activation sur l'ensemble du réseau IBM Workload Scheduler, est géré par le script **JnextPlan**. Vous pouvez exécuter **JnextPlan** à tout moment pendant la journée de traitement. Le nouveau plan de production généré est immédiatement activé sur les postes de travail cible, quelle que soit l'heure définie dans la variable *startOfDay*. Lorsque la commande **JnextPlan** est exécutée, la variable *\$MANAGER* est gérée comme suit :

- La variable est résolue si le poste de travail et un agent tolérant aux pannes d'une version antérieure à 8.6.
- La variable est laissée non résolue pour les postes de travail d'agents tolérants aux pannes version 8.6.

Lorsque vous exécutez le script **JnextPlan**, les processus des postes de travail sont arrêtés et démarrés à nouveau sur tous les postes de travail du réseau IBM Workload Scheduler. Pour plus d'informations sur les processus de poste de travail, voir Chapitre 2, «Description des processus et des commandes de base», à la page 37.

Le script **JnextPlan** peut uniquement être exécuté à partir du gestionnaire de domaine maître. Il utilise les paramètres de connexion par défaut définis dans le fichier *localopts* ou *useropts* (voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65). Si vous souhaitez exécuter **JnextPlan** avec des paramètres de connexion différents, vous pouvez éditer le script **MakePlan** et changer l'appel à l'instruction **planman** comme décrit dans «Ligne de commande Planman», à la page 94.

Le script **JnextPlan** est composé de la séquence de commandes et des scripts spécialisés suivants, chacun gérant un aspect spécifique de la génération du plan de production.

conman startappserver

Cette commande est appelée pour lancer WebSphere Application Server s'il n'est pas déjà en cours d'exécution.

MakePlan

Ce script reçoit de **JnextPlan** les indicateurs et les valeurs qui leurs sont affectés. Sa syntaxe est :

```
MakePlan [-from mm/jj/aa]aa[hh[:mm[tz | timezone fuseau_horaire]]] {-to mm/jj/aa]aa[hh[:mm[tz | timezone fuseau_horaire]] | -for [h]hh[:mm [-days n] | -jours n]}
```

Le script **MakePlan** appelle en interne la ligne de commande **planman**. Il exécute les actions suivantes :

1. Il crée un nouveau plan ou étend le plan courant et stocke les informations dans un plan de production intermédiaire qui contient :
 - Tous les objets de planification (travaux, flots de travaux, agendas, invites, ressources, postes de travail, domaines, fichiers, utilisateurs et dépendances) définis pour la période sélectionnée.
 - Toutes les dépendances entre les nouvelles instances des travaux et flots de travaux et les travaux et flots de travaux de l'ancien plan de production.

- Toutes les demandes de liaison dont l'heure planifiée est incluse dans la période de temps sélectionnée.
2. Supprime toutes les demandes de liaison à l'état final.
 3. Il imprime les rapports de préproduction.

SwitchPlan

Ce script appelle en interne la commande **stageman**. Pour plus d'informations, voir «Commande stageman», à la page 107. Le script **SwitchPlan** exécute les actions suivantes :

1. Arrête les processus IBM Workload Scheduler.
2. Il génère le nouveau fichier Symphony à partir du plan de production intermédiaire créé par **MakePlan**.
3. Il archive le fichier de l'ancien plan avec la date et l'heure actuelles dans le répertoire schedlog.
4. Il crée une copie du fichier Symphony à distribuer aux postes de travail.
5. Il redémarre les processus IBM Workload Scheduler qui distribuent la copie du fichier Symphony aux postes de travail cibles pour exécuter les travaux dans le plan.

Remarque : Assurez-vous qu'aucune commande **conman start** n'est exécutée pendant le traitement du plan de production.

CreatePostReports

Ce script imprime les rapports de postproduction.

UpdateStats

Ce script appelle en interne la commande **logman**. Pour plus d'informations, voir «Commande logman», à la page 110. Le script **UpdateStats** exécute les actions suivantes :

1. Il journalise les statistiques des travaux.
2. Il vérifie les règles et étend le plan de préproduction le cas échéant.
3. Il met à jour le plan de préproduction en signalant les états des instances de flot de travaux.

Pour plus d'informations sur l'utilisation du script **JnextPlan**, voir «JnextPlan».

Remarque : Pour obtenir des informations sur les scénarios spécifiques qui peuvent nécessiter la personnalisation **JnextPlan**, consultez *IBM Workload Scheduler - Guide d'administration*

JnextPlan

Le script **JnextPlan** gère l'ensemble du processus qui permet de passer d'un ancien plan de production à un nouveau (Symphony), notamment l'activation sur tout le réseau IBM Workload Scheduler. Chaque fois que vous exécutez **JnextPlan**, tous les postes de travail sont arrêtés et redémarrés.

Lorsque vous exécutez la commande **JnextPlan**, un fichier journal des travaux est créé dans le répertoire `<REP_INST_TWS>\TWS\stdlist\<DATE>`, où `<REP_INST_TWS>` est le répertoire d'installation de IBM Workload Scheduler et `<DATE>` est la date d'exécution du script.

Autorisation

Vous pouvez exécuter la commande **JnextPlan** à partir d'un shell d'invite de commande sur le gestionnaire de domaine maître si vous êtes l'un des utilisateurs suivants :

- L'utilisateur *utilisateur_TWS* pour lequel vous avez installé le produit sur la machine, s'il n'est pas désactivé par les paramètres définis dans le fichier de sécurité.
- L'utilisateur Root sur les systèmes d'exploitation UNIX ou l'utilisateur administrateur sur les systèmes d'exploitation Windows, s'il n'est pas désactivé par les paramètres définis dans le fichier de sécurité.

Syntaxe

JnextPlan

```
[ -V | -U ] |  
[ -from mm/jj/[aa]aa[hh[:]mm[tz | timezone nom_fh]]  
{ -to mm/jj/[aa]aa[hh[:]mm[tz | timezone nom_fh]] |  
  -for [h]hh[:]mm [-days n] | -days n }  
[ -noremove ]
```

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- from** Définit l'heure de début du plan de production. Le format de la date est défini dans le fichier *localopts*. *hhmm* indique les heures et minutes et *fuseau_horaire* est le fuseau horaire. Cet indicateur n'est utilisé que si aucun plan de production n'existe. Si l'argument **-from** n'est pas spécifié, la valeur par défaut est "today +startOfDay".
- Si le fuseau horaire n'est pas spécifié, le fuseau GMT est utilisé par défaut.
- to** heure de fin du plan de production Le format pour la date est le même que celui utilisé pour l'argument **-from**. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.
- Si le fuseau horaire n'est pas spécifié, le fuseau GMT est utilisé par défaut.
- for** Durée de l'extension de plan. Le format est *hh:mm*, où *hhh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec **-to**.
- days n**
Nombre de jours souhaités pour créer ou étendre le plan de production. Le paramètre **-days** est mutuellement exclusif avec le paramètre **-to**.
- noremove**
Vérifie que les instances de flot de travaux terminées ne sont pas supprimées du nouveau plan de production.

Si aucun argument **-to**, **-for**, ou **-days** n'est spécifié, la durée du plan de production par défaut est de un jour.

JnextPlan -for 0000

La commande JnextPlan - for 0000 étend de 0 heures et 0 minutes le plan de production et ajoute dans le plan de production (Symphony) les définitions de poste de travail, d'utilisateur et d'agenda nouvellement créées de la base de données. Elle supprime également toutes les instances de flot de travaux correctement terminées.

Si vous utilisez la commande JnextPlan - for 0000 - noremove, toutes les instances de flot de travaux correctement terminées du fichier Symphony ne sont pas supprimées.

Le paramètre de l'option globale enCarryForward détermine si les flots de travaux non terminés sont reportés de l'ancien plan de production vers le nouveau. Avant d'exécuter la commande, vérifiez que l'option enCarryForward est définie sur ALL pour que toutes les instances des flots de travaux non terminés soient reportées dans le nouveau plan de production, ou bien utilisez l'option -noremove.

Exemple

En supposant que la valeur affectée à *startOfDay* est 00:00 a.m. et que le format de date défini dans le fichier localopts est *mm/jj/aaaa*, si les valeurs définies sont **-from** 07/05/2011 et **-to** 07/07/2011, alors le plan est créé pour couvrir le cadre temporel allant de 07/05/2011 à 00:00 a.m. à 07/06/2011 à 11:59 p.m. et non pas au 07/07/2011 à 11:59 p.m.

Ligne de commande Planman

La ligne de commande **planman** est utilisée pour gérer les plans de production *intermédiaires*, les plans *d'essai* et les plans *prévisionnels*. Elle est également utilisée pour obtenir des informations sur le plan de production actif, pour déverrouiller les entrées de base de données verrouillées par le processus de gestion de plan pour déployer les règles d'événement de planification, et pour répliquer les données de plan dans la base de données. Cette commande s'exécute sur le gestionnaire de domaine maître. Utilisez la syntaxe suivante lorsque vous exécutez **planman** :

planman -U

planman -V

planman [*paramètres_connexion*] *commande*

où :

-U Permet d'afficher des informations sur la syntaxe de la commande et de quitter.

-V Permet d'afficher la version de la commande et de quitter.

paramètres_connexion

Si vous utilisez **planman** à partir du gestionnaire de domaine maître, les paramètres de connexion ont été configurés lors de l'installation et n'ont pas besoin d'être fournis, sauf si vous ne voulez pas utiliser les valeurs par défaut.

Si vous utilisez **planman** à partir du client de ligne de commande sur un autre poste de travail, les paramètres de connexion peuvent être fournis par l'une ou plusieurs de ces méthodes :

- Ils sont stockés dans le fichier `localopts`
- Ils sont stockés dans le fichier `useropts`
- Ils sont fournis à la commande dans un fichier de paramètres
- Ils sont fournis à la commande au sein d'une chaîne de commande

Pour une présentation de ces options, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65. Pour des détails complets sur les paramètres de configuration, voir la rubrique relative à la configuration de l'accès du client de ligne de commande dans *IBM Workload Scheduler - Guide d'administration*.

commande

Représente la commande que vous exécutez sur les plans à l'aide de l'interface **planman**. Voici les actions que vous pouvez effectuer avec les plans :

- «Création d'un plan de production intermédiaire»
- «Création d'un plan intermédiaire pour une extension de plan», à la page 97
- «Extraction des informations du plan de production», à la page 98
- «Création d'un plan d'essai», à la page 99
- «Création d'un plan d'essai avec une extension de plan de production», à la page 100
- «Création d'un plan prévisionnel», à la page 101
- «Déverrouillage du plan de production», à la page 103
- «Suppression du plan de préproduction», à la page 104
- «Réinitialisation du plan de production», à la page 103
- «Réplication des données de plan dans la base de données», à la page 105
- «Surveillance de la réplication des données de plan dans la base de données», à la page 106

Vous pouvez également utiliser **planman** pour déployer les règles d'événement de planification.. La commande est décrite dans : «Déploiement de règles», à la page 102. Consultez les sous-sections appropriées pour plus d'informations sur ces commandes.

Création d'un plan de production intermédiaire

La commande **planman** avec l'option **crt** est appelée à partir de la commande **JnextPlan** dans une de ces deux situations :

- La première fois que la commande **JnextPlan** est exécutée après l'installation du produit.
- Lors de la génération d'un plan de production après réinitialisation du plan de production à l'aide de la commande **ResetPlan**.

Cette commande entraîne la création d'un nouveau plan de production intermédiaire, appelé **Symnew**, qui couvre la totalité de la durée couverte ultérieurement par le plan de production en cours de génération. La syntaxe utilisée est la suivante :

planman [*paramètres_connexion*] **crt**

```
[-from mm/jj/[aa]aa [hh[:]mm [tz | timezone nom_fh]]]
{-to mm/jj/[aa]aa[hh[:]mm[tz | fuseau_horaire nom_fh]] |
-for [h]hh[:]mm [-days n] |
-days n}
```

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

-from Définit l'heure de début du nouveau plan de production.

Si l'argument **-from** est ignoré, alors :

- La date par défaut est *aujourd'hui*.
- L'heure par défaut est la valeur définie dans l'option globale *startOfDay* via **optman** sur le gestionnaire de domaine maître.

-to Heure de fin du nouveau plan de production. Le format pour la date est le même que celui utilisé pour l'argument **-from**. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.

-for Durée de l'extension de plan. Le format est *hh:mm*, où *hh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec l'argument **-to**.

-days n

Nombre de jours souhaités pour créer le plan de production. L'argument **-days** est mutuellement exclusif avec l'argument **-to**.

Remarque :

1. Veillez à exécuter la commande **planman** à partir de la commande **JnextPlan**.
2. Le format utilisé pour la date dépend de la valeur attribuée à la variable *format_date* dans le fichier *localopts*.

Si aucun argument **-to**, **-for**, ou **-days** n'est spécifié, la durée du plan de production par défaut est de un jour.

Voici quelques exemples d'utilisation de la commande **planman**, supposant que le format de date défini dans le fichier *localopts* est *mm/jj/aaaa* :

1. Cette commande crée le plan de production à partir du 03/21/2011 à 23:07 jusqu'au 03/22/2011 à 23:06, heure locale :

```
planman crt -from 03/21/05 2307
```
2. Cette commande crée le plan de production à partir du 03/21/2011 à 09:00 jusqu'au 03/21/2011 à 15:00 :

```
planman crt -from 03/21/2011 0900 for 0600
```
3. Si nous sommes le 03/21/05 et que la valeur définie pour la variable *startOfDay* dans la base de données est 0600, cette commande crée le plan de production à partir du 03/21/2011 à 6:00 jusqu'au 03/25/2011 à 5:59 :

```
planman crt -to 03/25/2011
```

4. Cette commande crée un plan de production à partir du 03/21/2011 à 18:05 jusqu'au 03/24/2011 à 23:00 dans le fuseau horaire Europe\Paris :

```
planman crt -from 03/21/2011 1805 tz Europe\Rome  
-to 03/24/2011 2300 tz Europe\Rome
```

Création d'un plan intermédiaire pour une extension de plan

La commande **planman** avec l'option **ext** est appelée à partir de la commande **JnextPlan** lorsque :

- **JnextPlan** est appelé.
- Un plan de production, représenté par le fichier Symphony sur le gestionnaire de domaine maître, existe déjà.

Cette commande entraîne la création d'un nouveau plan de production intermédiaire, appelé Symnew, qui couvre la durée supplémentaire couverte ultérieurement par le plan de production en cours de génération. La syntaxe utilisée est la suivante :

planman [*paramètres_connexion*] **ext**

{**-to** *mm/jj/aa*[*aa*[*hh*[:*mm*][*tz* | *fuseau horaire nom_fh*]] |

-for [*h*][*hh*[:*mm*] **-days** *n*] |

-days *n*}

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

-to Définit l'heure de fin du plan de production étendu. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.

-for Définit la longueur de l'extension du plan de production. Le format est *hhmm*, où *hh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec l'argument **-to**.

-days *n*

Nombre de jours souhaités pour étendre le plan de production. L'argument **-days** est mutuellement exclusif avec l'argument **-to**.

Remarque :

1. Veillez à exécuter la commande **planman** à partir de la commande **JnextPlan**.
2. Le format utilisé pour la date dépend de la valeur attribuée à la variable *format_date* dans le fichier *localopts*.
3. Lors de l'extension du plan de production, les chiffres associés aux invites déjà présentes dans le plan sont modifiés.

Si aucun argument **-to**, **-for** ou **-days** n'est spécifié, le plan de production est étendu d'une journée.

Extraction des informations du plan de production

La syntaxe ci-dessous est utilisée pour afficher des informations sur le plan de production courant :

planman [*paramètres_connexion*] **showinfo**

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

Remarque : Vous pouvez installer la fonction IBM Workload Scheduler Command Line Client sur les agents tolérants aux pannes et les systèmes extérieurs au réseau IBM Workload Scheduler pour émettre à partir de ces systèmes la commande **planman showinfo**.

Le résultat de cette commande affiche :

- Le chemin de l'installation.
- L'heure de début du plan de production.
- L'heure de fin du plan de production.
- La durée du plan de production, après la dernière extension, le cas échéant.
- La date et l'heure de la dernière mise à jour du plan effectuée via **JnextPlan** ou via **planman**.
- L'heure de fin du plan de préproduction.
- L'heure de début de la première instance de flot de travaux non terminée.
- Le *nombre d'exécutions*, c'est à dire le nombre de fois où le plan a été généré.
- Le *nombre d'exécutions confirmées*, c'est à dire le nombre de fois où le plan a été généré correctement.

Les heures de début et de fin des plans de production et de préproduction sont affichées au format spécifié par la variable *format_date* définie dans le fichier localopts et dans le fuseau horaire de la machine locale.

Voici un exemple de résultat de cette commande :

```
# planman showinfoIBM Workload Scheduler (UNIX)/PLANMAN 8.6 (20100715)
Licensed Materials - Property of IBM*
5698-WSH
(C) Copyright IBM Corp. 1998, 2011 All rights reserved.
* Trademark of International Business Machines
Installed for user "aix6lusr".
Locale LANG set to the following: "en"
Plan creation start time: 07/21/2010 06:00 TZ Europe/Rome
Production plan start time of last extension: 07/21/2010 06:00 TZ Europe/Rome
Production plan end time: 07/22/2010 05:59 TZ Europe/Rome
Production plan time extension: 024:00
Plan last update: 07/21/2010 10:05 TZ Europe/Rome
Preproduction plan end time: 08/05/2010 06:00 TZ Europe/Rome
Start time of first not complete preproduction plan job stream instance:
    07/21/2010 10:30 TZ Europe/Rome
Run number: 1
Confirm run number: 1
```


Création d'un plan d'essai

La syntaxe ci-dessous est utilisée pour créer un plan d'essai :

planman [*paramètres_connexion*] **crtrial** *nom_fichier*

[-from *mm/jj/aa* [*hh[:mm]* [*tz* | **fuseau horaire** *nom_fh*]]]

{-to *mm/jj/aa* [*hh[:mm]* [*tz* | **fuseau horaire** *nom_fh*]] |

-for [*h*]*hh[:mm]* [**-days** *n*] |

-days *n*}

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

nom_fichier

Attribue un nom au fichier à créer dans le répertoire *racine_TWS/schedTrial* et qui contient le plan d'essai. Le fichier qui contient le plan d'essai s'appelle **Tnom_fichier**. Cela signifie que si la valeur attribuée à *nom_fichier* est *myfile*, le nom du fichier qui contient le plan d'essai généré est *Tmyfile*.

-from Définit l'heure de début du plan d'essai.

Si l'argument **-from** est ignoré, alors :

- La date par défaut est *aujourd'hui*.
- L'heure par défaut est la valeur définie dans l'option globale **startOfDay** via **optman** sur le gestionnaire de domaine maître.

-to Définit l'heure de fin du plan d'essai. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.

-for Définit la durée du plan d'essai. Le format est *hh:mm*, où *hh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec l'argument **-to**.

-days *n*

Nombre de jours souhaités pour le plan d'essai. L'argument **-days** est mutuellement exclusif avec l'argument **-to**.

Remarque : Le format utilisé pour la date dépend de la valeur attribuée à la variable *format_date* dans le fichier *localopts*.

Si aucun argument **-to**, **-for**, ou **-days** n'est spécifié, la durée du plan d'essai par défaut est de un jour.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Création d'un plan d'essai avec une extension de plan de production

La syntaxe ci-dessous est utilisée pour créer un plan d'essai avec une extension du plan de production courant :

planman [*paramètres_connexion*] **exttrial** *nom_fichier*

{**-to** *mm/jj/aa*[*hh[:mm][tz | fuseau_horaire nom_fh]*] |

-for [*h*]*hh[:mm]* [**-days** *n*] |

-days *n*}

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

nom_fichier

Attribue un nom au fichier à créer dans le répertoire *racine_TWS/schedTrial* et qui contient le plan d'essai. Le fichier qui contient le plan d'essai s'appelle **Tnom_fichier**. Cela signifie que si la valeur attribuée à *nom_fichier* est *myfile*, le nom du fichier qui contient le plan d'essai généré est *Tmyfile*.

-to Définit l'heure de fin du plan d'essai contenant l'extension du plan de production. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.

-for Définit la longueur du plan d'essai contenant l'extension du plan de production. Le format est *hh:mm*, où *hh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec l'argument **-to**.

-days *n* Définit le nombre de jours souhaités pour le plan d'essai contenant l'extension du plan de production. L'argument **-days** est mutuellement exclusif avec l'argument **-to**.

Remarque : Le format utilisé pour la date dépend de la valeur attribuée à la variable *format_date* dans le fichier *localopts*.

Si aucun argument **-to**, **-for**, or **-days** n'est spécifié, par défaut, le plan de production contenu dans le plan d'essai est étendu d'une journée.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Génération de plans d'essai et prévisionnels".

Création d'un plan prévisionnel

La syntaxe ci-dessous est utilisée pour créer un plan prévisionnel :

planman [*paramètres_connexion*] **crtfc** *nom_fichier*

[-from *mm/jj/aa* [*hh[:mm]* [*tz* | **fuseau horaire** *nom_fh*]]]

[-to *mm/jj/aa* [*hh[:mm]* [*tz* | **fuseau horaire** *nom_fh*]] |

-for [*h*]*hh[:mm]* [**-days** *n*] |

-days *n*}

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

nom_fichier

Attribue un nom au fichier à créer dans le répertoire *racine_TWS/schedForecast* et qui contient le plan prévisionnel. Le fichier qui contient le plan prévisionnel s'appelle **Fnom_fichier**. Cela signifie que si la valeur attribuée à *nom_fichier* est *myfile*, le nom du fichier qui contient le plan prévisionnel généré est *Fmyfile*.

La longueur maximum de *nom_fichier* peut être de 148 caractères.

-from Définit l'heure de début du plan prévisionnel. La minute spécifiée est incluse.

Si l'argument **-from** est ignoré, alors :

- La date par défaut est *aujourd'hui*.
- L'heure par défaut est la valeur définie dans l'option globale **startOfDay** via **optman** sur le gestionnaire de domaine maître.

-to Définit l'heure de fin du plan prévisionnel. La minute spécifiée est exclue. L'argument **-to** est mutuellement exclusif avec les arguments **-for** et **-days**.

-for Définit la durée du plan prévisionnel. Le format est *hh:mm*, où *hh* représente les heures et *mm* les minutes. L'argument **-for** est mutuellement exclusif avec l'argument **-to**.

-days *n*

Nombre de jours souhaités pour le plan prévisionnel. Si l'intervalle contient l'heure d'été, celle-ci est automatiquement incluse dans le calcul. L'argument **-days** est mutuellement exclusif avec l'argument **-to**.

Remarque : Le format utilisé pour la date dépend de la valeur attribuée à la variable *format_date* dans le fichier *localopts*.

Si aucun argument **-to**, **-for**, ou **-days** n'est spécifié, la durée par défaut du plan prévisionnel est de un jour.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Génération de plans d'essai et prévisionnels".

Déploiement de règles

La commande `planman deploy` est utilisée dans la gestion d'événements. Vous pouvez l'utiliser pour déployer manuellement toutes les règles qui ne sont pas à l'état Brouillon (Draft) (la propriété `isDraft` à la valeur `N0` dans leur définition). La commande fonctionne comme suit :

1. Sélectionne toutes les définitions de règles d'événement qui ne sont pas à l'état Brouillon dans la base de données IBM Workload Scheduler.
2. Construit des fichiers de configuration de règles d'événement.
3. Déploie les fichiers de configuration sur les moteurs de surveillance s'exécutant sur les agents IBM Workload Scheduler.

Les nouveaux fichiers de configuration mettent à jour les règles d'événement s'exécutant sur chaque moteur de surveillance en termes de :

- Nouvelles règles
- Règles modifiées
- Règles supprimées ou ramenées à l'état draft

Vous pouvez utiliser cette commande en complément ou en remplacement de l'option de configuration `deploymentFrequency (df) optman`, qui vérifie périodiquement toute modification apportée aux définitions de règle d'événement à déployer (pour plus d'informations sur cette option, voir le *Guide d'administration*).

Les modifications appliquées aux définitions de règles d'événement dans la base de données prennent effet uniquement une fois le déploiement effectué.

La syntaxe de la commande est :

planman [*paramètres_connexion*] **deploy** [-scratch]

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

-scratch

Sans cette option, la commande affecte uniquement les règles qui ont été ajoutées, changées, supprimées ou ramenées à leur état draft.

Avec cette option, la commande déploie toutes les règles non provisoire de la base de données y compris celles qui sont déjà déployées et n'ont pas changé.

Remarque : le temps de déploiement augmente de manière proportionnelle avec le nombre de règles actives à déployer. Si vous avez besoin de

déployer manuellement un grand nombre de règles nouvelles ou modifiées, exécutez `planman deploy -scratch` avec cette option pour réduire le temps de déploiement.

L'utilisation de cette option aboutit à une réinitialisation complète du processeur d'événements et doit donc être utilisée avec précaution. La commande peut provoquer la perte de toutes les instances de règle en cours au moment de son émission. Le cas typique est une règle séquentielle qui a été déclenchée et attend des événements additionnels : si vous utilisez l'option à ce moment là l'environnement de règles d'événement est réinitialisé et les éléments suivis sont perdus.

Pour exécuter cette commande, vous devez avoir un accès **build** sur le fichier **prodsked**.

Déverrouillage du plan de production

Lorsque IBM Workload Scheduler commence à créer le plan de production, il verrouille les définitions des objets de planification dans la base de données puis les déverrouille soit lorsque la création du plan est terminée, soit si une erreur se produit. Le verrouillage est appliqué pour éviter que les définitions des objets ne soient modifiées lors de la génération ou de l'extension du plan de production. Si le traitement ne se termine pas correctement, les entrées de la base de données risquent de rester verrouillées. Seuls les utilisateurs ayant un droit d'accès **build** au type d'objet fichier **prodsked** défini dans le fichier de sécurité sur le gestionnaire de domaine maître peuvent déverrouiller la base de données. La commande à utiliser est la suivante :

planman [*paramètres_connexion*] **unlock**

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

Remarque : Vous pouvez installer la fonction IBM Workload Scheduler Command Line Client sur les agents tolérants aux pannes et les systèmes extérieurs au réseau IBM Workload Scheduler pour émettre à partir de ces systèmes la commande **planman unlock**.

Réinitialisation du plan de production

Le script ci-dessous permet de réinitialiser ou de supprimer le plan de production :

ResetPlan [*paramètres_connexion*] [**-scratch**]

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

La différence entre une réinitialisation et une suppression du plan de production est la suivante :

- Si vous *réinitialisez* le plan de production, le plan de préproduction est conservé, mis à jour avec les statistiques des travaux, puis utilisé plus tard pour générer un nouveau plan de production. Par conséquent, lorsque vous créez le nouveau plan de production, il contiendra toutes les instances de flot de travaux qui n'étaient pas à l'état COMPLETE lors de l'exécution de **ResetPlan**. Lors de la réinitialisation du plan de production, le produit procède comme suit :
 1. Il archive le fichier Symphony courant.
 2. Il met à jour les statistiques des travaux.
- Si vous *supprimez* le plan de production, le plan de préproduction est également supprimé. Le plan de préproduction sera recréé sur la base des informations de modélisation stockées dans la base de données lors de la prochaine génération du plan de production. Par conséquent, le nouveau plan de production contiendra toutes les instances de flot de travaux devant être exécutées sur la période couverte par le plan, quel que soit leur état (COMPLETE ou autre) lors de la suppression du plan. Lors de la suppression du plan de production, le produit procède comme suit :
 1. Le fichier Symphony actuel est archivé et les données de plan répliquées dans la base de données sont supprimées.
 2. Il met à jour les statistiques des travaux.
 3. Il supprime le plan de préproduction.

Remarque : Si vous utilisez l'option **-scratch**, veuillez à exécuter **dbrunstats** avant le script **JnextPlan**. Pour plus d'informations sur **dbrunstats**, voir *Guide d'administration*.

Lorsque vous exécutez la commande **ResetPlan**, un fichier journal des travaux est créé dans le répertoire `<REP_INST_TWS>\TWS\stdlist\<DATE>`, où `<REP_INST_TWS>` est le répertoire d'installation de IBM Workload Scheduler et `<DATE>` est la date d'exécution du script.

Suppression du plan de préproduction

Le script suivant est utilisé pour supprimer le plan de préproduction lors de la gestion du fichier Symphony :

Planman reset -scratch

Lorsque vous exécutez cette commande, le plan de préproduction est supprimé. Le plan de préproduction sera recréé sur la base des informations de modélisation stockées dans la base de données lors de la prochaine génération du plan de production. Par conséquent, le nouveau plan de production contiendra toutes les instances de flot de travaux devant être exécutées sur la période couverte par le plan, quel que soit leur état (COMPLETE ou autre) lors de la suppression du plan. Lors de la suppression du plan de production, le produit procède comme suit :

1. Le fichier Symphony est conservé.
2. Il met à jour les statistiques des travaux.
3. Il supprime le plan de préproduction.

Remarque : Si vous utilisez l'option **-scratch**, veuillez à exécuter **dbrunstats** avant le script **JnextPlan**. Pour plus d'informations sur **dbrunstats**, voir le *Guide d'administration*.

Réplication des données de plan dans la base de données

Le stockage d'informations sur des objets du plan dans une base de données rend l'accès aux données de plan beaucoup plus facile et rapide. Lorsque de nombreux utilisateurs accèdent à l'environnement de back end IBM Workload Scheduler en même temps, les performances et la fiabilité peuvent être compromises. Le fait de répliquer le plan dans une base de données relationnelle peut rendre l'accès aux données plus rapide et plus fiable.

Pour les versions antérieures à la version 9.1, les opérations suivantes nécessitaient un accès au plan Symphony :

- Exécution des rapports de version de référence
- Affichage du plan dans une vue graphique
- Affichage d'une vue graphique de flot de travaux
- Déclenchement des actions
- Actualisation des vues de surveillance des travaux et flots de travaux

Toutes ces activités de modélisation ou de surveillance doivent accéder au plan et, si vous multipliez ces scénarios par le nombre d'utilisateurs qui demandent en même temps un accès au plan pour effectuer une ou plusieurs de ces activités, cela résulte en des temps de réponse et en des performances globales lents.

Pour corriger ce problème, IBM Workload Scheduler réplique le plan dans une base de données relationnelle, dans laquelle des instructions SQL permettent de retrouver les données de façon rapide et fiable. De nouvelles boîtes de message, `mirrorbox.msg` et `mirrorbox<n>.msg`, permettent de synchroniser la base de données avec le fichier Symphony. Si le fichier `mirrorbox.msg` arrive à saturation, par exemple si la base de données reste indisponible pendant une longue période, le plan est automatiquement rechargé dans la base de données à l'aide des informations du fichier Symphony.

En particulier, les performances et les temps de réponse du script `UpdateStats` ont été considérablement améliorés avec cette nouvelle façon de gérer les données de plan.

En outre, à chaque fois que le plan est étendu, le plan dans la base de données est effacé et recréé, ce qui rend les informations les plus récentes accessibles pour tous les utilisateurs. Pour répliquer manuellement les données de plan depuis le fichiers Symphony vers la base de données, exécutez la commande **planman resync**.

La synchronisation du fichier Symphony est automatiquement activée lorsque vous ajoutez le fichier `Sfinal` à la base de données avec la commande `composer add Sfinal`. Un nouveau travail, `CHECKSYNC`, a été ajouté au flot de travaux `FINALPOSTREPORTS` contenu dans le fichier `Sfinal` qui est responsable de la surveillance de l'état du processus de réplication du fichier Symphony dans la base de données. Il envoie le progrès et l'état de ce processus au journal de travail. Si le travail `CHECKSYNC` doit échouer, reportez-vous au journal du travail `CHECKSYNC`, ainsi qu'au journal WebSphere Application Server pour déterminer le problème. Après résolution du problème, exécutez la commande **planman resync** pour recharger les données du plan à partir du fichier Symphony dans la base de données.

Pour simplifier les intégrations, un ensemble de vues de la base de données pour un ensemble de tables contenant des données de plan dans la base de données IBM Workload Scheduler est fourni.

Pour afficher les vues de base de données contenant des informations sur les objets IBM Workload Scheduler dans le plan, reportez-vous aux vues qui commencent par "PLAN_" dans *IBM Workload Scheduler - Vues de la base de données*.

Lors de l'exécution des opérations qui récupèrent les données du plan en cours à partir de Dynamic Workload Console, si vous soupçonnez les données de ne pas être à jour, vous pouvez exécuter **planman resync** pour mettre à jour les données du plan dans la base de données avec les informations les plus récentes du fichier Symphony. Si la boîte de message, `mirrorbox<n>.msg`, chargée de synchroniser la base de données avec le fichier Symphony arrive à saturation (par exemple, lorsque la base de données est indisponible pour une longue période), la commande **planman resync** est automatiquement émise pour que le plan soit entièrement rechargé dans la base de données.

La syntaxe suivante permet de répliquer les données du plan dans la base de données avec les données du fichier Symphony :

planman [*paramètres_connexion*] **resync**

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour de plus amples informations, voir «Ligne de commande Planman», à la page 94.

Remarque :

- Si vous mettez à jour votre environnement IBM Workload Scheduler vers la version 9.2, vous devez alors implémenter quelques étapes manuelles avant que vos données de plan ne puissent être répliquées dans la base de données. Consultez la section relative à la personnalisation et à la soumission du flot de travaux final en option dans *Planification et installation* et «Automatisation du traitement du plan de production», à la page 114 pour plus d'informations.
- La réplication des données de plan dans la base de données nécessite que le pilote JDBC DB2 type 4 soit installé. Le pilote JDBC DB2 type 2 n'est pas pris en charge.
- Assurez-vous que dans la configuration de base de données la propriété CUR_COMMIT est définie sur ENABLED. Pour plus d'informations sur ce paramètre, voir la documentation DB2.

Surveillance de la réplication des données de plan dans la base de données

La syntaxe suivante permet de surveiller la progression et l'issue de la réplication des données du plan dans la base de données avec les données du fichier Symphony :

planman [*paramètres_connexion*] **checksync**

où :

paramètres_connexion

Définit les paramètres utilisés lors de l'établissement de la connexion avec

HTTP ou HTTPS par WebSphere Application Server vers le gestionnaire de domaine maître. Pour plus d'informations, voir «Ligne de commande Planman», à la page 94.

Les messages sont écrits dans la sortie standard avec la progression et le statut de la commande. La commande `planman checksync` est également définie dans le travail CHECKSYNC, contenu dans le flot de travaux FINALPOSTREPORTS au sein du fichier Sfinal. Si le travail CHECKSYNC doit échouer, alors reportez-vous au journal des travaux du travail CHECKSYNC ainsi qu'au journal WebSphere Application Server pour déterminer le problème. Après résolution du problème, exécutez la commande `planman resync` pour recharger les données du plan à partir du fichier Symphony dans la base de données.

Commande stageman

La commande **stageman** reporte les flots de travaux non terminés, archive l'ancien plan de production et installe le nouveau. Une copie de Symphony est envoyée aux gestionnaires de domaine et aux agents dans le cadre du processus d'initialisation du nouveau plan de production. Lors de l'exécution de **JnextPlan**, **stageman** est appelé à partir du script **SwitchPlan**.

Vous devez avoir un droit d'accès *build* au fichier Symphony pour exécuter **stageman**.

Syntaxe

stageman -V | -U

stageman
 [-carryforward{yes | no | all}]
 [-log fichier_journal | -nolog]
 [symnew]

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-carryforward

Définit comment les flots de travaux non terminés sont traités lors du passage au nouveau plan de production. Les valeurs possibles sont les suivantes :

no Aucun flot de travaux n'est reporté.

yes Seuls les flots de travaux non terminés dont les définitions contiennent le mot clé **carryforward** sont reportés.

all Tous les flots de travaux non terminés sont reportés, quelle que soit la valeur du mot clé **carryforward** dans leur définition.

Si vous omettez ce mot clé, la valeur définie globalement à l'aide de **optman** pour l'option *enCarryForward* lui est attribuée par défaut. Pour plus d'informations sur les paramètres de report utilisés lorsque l'option globale *enCarryForward* et les mots clés **-carryforward** sont tous deux définis, voir «Options de report», à la page 82.

-log Archive l'ancien plan de production dans le répertoire *racine_TWS/schedlog* dans un fichier nommé *fichier_journal*. Les plans de production archivés peuvent alors être répertoriés et sélectionnés (voir «listsym», à la page 470 et «setsym», à la page 489). Si aucun des mots clés **-log** ou **-nolog** n'est défini, IBM Workload Scheduler archive l'ancien plan de production selon la convention d'attribution de nom suivante :

Maaaammjjhhtt

où *aaaammjjhhtt* correspond à l'année, au mois, au jour, à l'heure et à la minute d'archivage de l'ancien plan de production. Si vous générez le plan de production via **JnextPlan**, vous pouvez personnaliser cette convention d'attribution de nom dans le script **SwitchPlan**.

Remarque : Veillez à contrôler l'espace disque du répertoire *schedlog* et à supprimer les fichiers journaux obsolètes de façon régulière.

-nolog Pas d'archivage de l'ancien plan de production.

symnew

Nom attribué au fichier du plan de production intermédiaire créé par **planman**. Si cette valeur n'est pas indiquée, **stageman** utilise le nom de fichier *Symnew*.

Commentaires

Pour que les procédures de report fonctionnent correctement en réseau, le fichier du plan de production du gestionnaire de domaine maître, *Symphony*, doit être mis à jour avec les statuts les plus récents des flots de travaux à partir des agents et des gestionnaires de domaine subordonnés. Exécutez la commande :

```
conman "link @"
```

avant de lancer **stageman**. Cette commande connecte les postes de travail non connectés pour que les messages sur le statut du traitement des travaux mis en file d'attente dans le fichier *Mailbox.msg* soient renvoyés au gestionnaire de domaine maître afin de mettre à jour le fichier *Symphony*.

Remarque : Sous UNIX uniquement, **stageman** détermine également les fichiers exécutables associés aux travaux soumis via les commandes d'utilitaire **at** et **batch** qui peuvent être supprimés lorsque IBM Workload Scheduler démarre pour la nouvelle période de production. Ces travaux ne sont pas reportés.

Exemples

Procédez au report de tous les flots de travaux non terminés (quel que soit l'état de l'option *Carry Forward*), conservez l'ancien fichier *Symphony* et créez le nouveau fichier *Symphony* :

```
DATE='datecalc today pic AAAAMMJJHHTT'  
stageman -carryforward all -log schedlog/M$DATE
```

Procédez au report des flots de travaux non terminés conformément à l'option globale *carryforward* ; ne conservez pas l'ancien fichier *Symphony* mais créez un nouveau plan de production intermédiaire appelé *mysym*:

```
stageman -nolog mysym
```

Gestion des accès simultanés au fichier Symphony

La présente section décrit deux exemples de scénarios où IBM Workload Scheduler gère d'éventuels accès simultanés au fichier Symphony lors de l'exécution de **stageman**.

Scénario 1 : Accès au fichier Symphony verrouillé par d'autres processus IBM Workload Scheduler

Si des processus IBM Workload Scheduler sont encore actifs et accèdent au fichier Symphony lors de l'exécution de **stageman**, le message suivant s'affiche :

Impossible d'obtenir l'accès exclusif à Symphony.
Arrêt de batchman et de mailman.

Pour continuer, arrêtez IBM Workload Scheduler et réexécutez **stageman**. Si **stageman** se termine de manière anormale pour une raison quelconque, vous devez réexécuter **planman** et **stageman**.

Scénario 2 : Accès au fichier Symphony verrouillé par stageman

Si vous essayez d'accéder au plan par l'interface de ligne de commande pendant le remplacement du fichier Symphony, le message suivant s'affiche :

Le fichier Symphony est obsolète. Remplacement par un nouveau fichier Symphony.
Agenda mm/jj/aaaa (nnnn) sur poste de travail, changement de Symphony.

Gestion des dépendances de prédécesseur/successeur avec l'invite de report

Pour assurer la continuité lors du report des flots de travaux, **stageman** génère des invites pour chaque flot de travaux reporté qui a une dépendance prédécesseur/successeur avec un autre flot de travaux non reporté. Ces invites sont émises après le début de la nouvelle période de traitement, lorsque IBM Workload Scheduler vérifie si le travail ou le flot de travaux est prêt à être lancé. Les réponses ont la forme d'invites standard. L'exemple qui suit montre une *invite de report*:

```
INACT 1(SYS2#SKED2[(0600 01/11/06),(0AAAAAAAAAAAAA2Y)]) follows  
SYS1#SKED1, satisfied?
```

Cette invite indique qu'un flot de travaux, reporté à partir de l'ancien plan de production, (SYS2#SKED2[(0600 01/11/06),(0AAAAAAAAAAAAA2Y)]), a une dépendance de prédécesseur/successeur avec un flot de travaux nommé SYS1#SKED1 qui n'a pas été reporté. Pour plus d'informations sur la syntaxe permettant d'indiquer le flot de travaux reporté, voir «Sélection de flots de travaux dans les commandes», à la page 428.

L'état de l'invite, **INACT** dans ce cas, définit l'état de la dépendance de prédécesseur/successeur correspondante. Les différents états possibles sont les suivants :

INACT

L'invite n'a pas été émise et la dépendance n'est pas satisfaite.

ASKED

L'invite a été émise et attend une réponse. La dépendance n'est pas satisfaite.

- NON** Une réponse négative a été reçue ou bien le programme a constaté avant que le report ne s'effectue que le flot de travaux (SKED3) ne s'est pas achevé avec succès. La dépendance n'est pas satisfaite.
- OUI** Une réponse positive a été reçue ou bien le programme a constaté avant que le report ne s'effectue que le flot de travaux (SKED3) s'est achevé avec succès. La dépendance est satisfaite.

Commande logman

La commande **logman** consigne les statistiques de travaux depuis un fichier journal de plan de production.

Syntaxe

logman -V|-U

logman

[*paramètres_connexion*]
 {-**prod** | *fichier-Symphony*]
 [-**smooth** *pondération*]
 [-**minmax** {*temps_écoulé* | *UC*}]}

Arguments

- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- V** Affiche la version de la commande et quitte l'application.

paramètres_connexion

Représente le jeu de paramètres qui contrôle les interactions entre l'interface du produit, **logman** s'exécutant dans ce cas sur le gestionnaire de domaine maître, et l'infrastructure WebSphere Application Server utilisant HTTP ou HTTPS. Utilisez la syntaxe suivante pour définir les paramètres de connexion :

[**-host** *nom_hôte*] [**-port** *numéro_port*] [**-protocol** *nom_protocole*][**-proxy** *nom_proxy*] [**-proxypport** *numéro_port_proxy*] [**-password** *mot_de_passe_utilisateur*] [**-timeout** *délai d'attente*] [**-username** *nom_utilisateur*]

où :

nom_hôte

Nom d'hôte du gestionnaire de domaine maître.

numéro_port

Numéro de port utilisé lors de la connexion au gestionnaire de domaine maître.

nom_protocole

Protocole utilisé lors de la communication. Il peut s'agir de HTTP avec une authentification standard ou de HTTPS avec une authentification par certificat.

nom_proxy

Nom de l'hôte proxy utilisé pour la connexion.

numéro_port_proxy

Numéro de port proxy utilisé pour la connexion.

mot_de_passe_utilisateur

Mot de passe de l'utilisateur utilisé pour exécuter **logman**.

Remarque : Lorsque vous définissez, sur un poste de travail Windows, un mot de passe contenant deux apostrophes (") ou d'autres caractères spéciaux, assurez-vous qu'une séquence d'échappement est définie pour ces caractères. Si par exemple le mot de passe utilisé est `twsl1"twsl`, spécifiez-le sous la forme `"twsl1\"twsl"`.

délai_attente

Délai maximal, en secondes, pendant lequel le programme de ligne de commande de connexion peut attendre la réponse du gestionnaire de domaine maître avant de considérer que la demande de communication a échoué.

nom_utilisateur

Nom de l'utilisateur qui exécute **logman**.

Si un de ces paramètres est oublié lors de l'appel de la commande **logman**, IBM Workload Scheduler le recherche d'abord dans le fichier `useropts`, puis dans le fichier `localopts`. Si aucune valeur n'est trouvée pour ce paramètre, une erreur s'affiche. Pour plus d'informations sur les fichiers `useropts` et `localopts`, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

-prod Met à jour le plan de préproduction avec les informations sur les flots de travaux à l'état COMPLETE en production. En procédant ainsi le plan de préproduction est maintenu à jour avec les toutes dernières informations de traitement. Ceci évite le risque que le nouveau plan de production n'exécute à nouveau les flots de travaux déjà complétés dans la période de production précédente.

-minmax {*temps_écoulé* | UC}

Définit le mode de journalisation et de consignation des temps d'exécution de travail minimum et maximum. Les valeurs possibles sont les suivantes :

temps_écoulé

Base les temps d'exécution minimum et maximum sur le temps écoulé.

UC

Base les temps d'exécution minimum et maximum sur le temps UC.

Cette valeur est utilisée lorsque la commande **logman** est exécutée à partir de la ligne de commande et non à partir du script **JnextPlan**. Lorsque la commande **logman** est exécutée par **JnextPlan**, la valeur utilisée est celle spécifiée dans l'option globale *logmanMinMaxPolicy*.

-smooth *pondération*

Utilise un facteur de pondération qui favorise l'exécution du travail le plus récent lors du calcul du temps d'exécution normal (moyen) pour un travail. Le résultat de ce calcul est exprimé sous la forme d'un pourcentage. Par exemple, **-smooth 40** applique un facteur de pondération de 40% à l'exécution du travail le plus récent et de 60% à la moyenne existante. La valeur par défaut est **-smooth 10**. Ce paramètre est utilisé lorsque la commande **logman** est exécutée à partir de la ligne de commande, comme vous pouvez être amené à le faire en cas de reprise des travaux si vous souhaitez remplacer les statistiques de travaux dans la base de données.

Lorsque la commande **logman** est exécutée par **JnextPlan**, la valeur utilisée est celle spécifiée dans l'option globale *logmanSmoothPolicy*.

fichier-Symphony

Le nom d'un fichier Symphony archivé d'où sont extraites des statistiques de travaux.

Commentaires

Les travaux déjà consignés ne peuvent pas être consignés une nouvelle fois. Si vous tentez de le faire, le message d'erreur 0 travaux consignés est généré.

Exemples

Consignez les statistiques de travaux à partir du fichier journal M201403170935 :
logman schedlog/M201403170935

Durée estimée d'un travail et facteur de fiabilité associé

La **durée estimée** de l'exécution d'un travail et le **facteur de fiabilité** associé sont fournis par la commande logman dans le cadre du cycle de planification quotidien. La durée estimée de l'exécution d'un travail est basée sur la moyenne de ses exécutions précédentes, calculée à partir de l'analyse de cinq séries temporelles différentes : GLOBAL, WEEK_DAY, MONTH_DAY, MONTH_DAY_REVERSE, RUN_CYCLE. Pour calculer le temps d'exécution moyen d'un travail, la commande logman divise le temps total d'exécution pour toutes les exécutions réussies par le nombre d'exécutions réussies. Si un grand nombre d'exécutions est utilisé pour calculer la moyenne, le changement du temps d'exécution d'un travail n'est pas immédiatement reflété dans la moyenne. Pour répondre plus rapidement à ces changements, vous pouvez utiliser l'option smooth afin que la moyenne puisse être pondérée en faveur des exécutions les plus récentes. Utilisez l'option -smooth pour saisir un facteur de pondération, tel qu'un pourcentage, pour les exécutions de travaux en cours. Par exemple, la commande logman -smooth 40 oblige la commande logman à utiliser un facteur de pondération de 40 % pour les exécutions les plus récentes du travail et de 60 % pour la moyenne existante. La commande logman -smooth 100 oblige les exécutions du travail les plus récentes à remplacer la moyenne existante. La valeur par défaut de l'option -smooth est 10.

La commande logman conserve les statistiques des exécutions du travail dans la base de données IBM Workload Scheduler. Le nombre d'instances de travail conservées dans l'historique des travaux n'est pas limité.

Pour chaque travail critique, une valeur de pourcentage indique le niveau de fiabilité du travail critique pour respecter son échéance. Le facteur de fiabilité est calculé comme fonction de densité cumulative normale : c'est la probabilité que le travail se terminera dans l'échéance, calculée à l'aide d'une fonction de Gauss, où l'heure de fin prévue est la moyenne et la variance de fin prévue est l'écart type. Pour réduire l'effort de performance à un minimum, les heures de début et de fin prévues sont mises à jour seulement lorsque les heures prévues sont mises à jour. En conséquence, la valeur du facteur de fiabilité correspond toujours au dernier calcul.

Un facteur de fiabilité de 95 % est associé à la durée estimée d'un travail. Le facteur de fiabilité fournit une plage estimée de valeurs, qui est censée inclure la durée du travail. Par exemple, si la durée estimée d'un travail est 00:03:00, avec un facteur de fiabilité de 00:00:05, cela signifie que vous pouvez être certain à 95 %

que la durée réelle du travail est comprise entre (00:03:00 - 00:00:05) et (00:03:00 + 00:00:05), c'est-à-dire entre 00:02:55 et 00:03:05.

Lorsqu'un nouveau plan de production est généré, vous pouvez afficher la durée estimée d'un travail et le facteur de fiabilité associé en exécutant la commande suivante à partir de la ligne de commande **conman** :

```
showjobs nom_travail;props
```

Vous pouvez afficher les mêmes informations à partir de la requête **Surveiller la charge de travail** de Dynamic Workload Console.

Dans certaines conditions, vous pouvez décider de substituer la durée estimée d'un travail, par exemple :

- Pour un nouveau travail dont la durée estimée est définie, par défaut, sur 00:00:00. Vous pouvez décider de définir la durée estimée sur une valeur initiale plus réaliste en fonction de votre expérience.
- Pour un travail dont la durée estimée est associée à un facteur de fiabilité élevé. Vous pouvez décider de modifier la planification du travail et, par conséquent, sa durée estimée.

À l'issue de l'exécution d'un travail, le facteur de fiabilité est défini sur 0 % lorsque la durée estimée est dépassée et sur 100 % lorsque la durée estimée n'est pas dépassée. Pendant que les données statistiques sont collectées, la durée estimée que vous avez définie est remplacée par logman, en fonction de l'algorithme décrit ci-dessus.

Vous pouvez substituer la durée estimée d'un travail dans un flot de travaux à partir du **Concepteur de charge de travail** de Dynamic Workload Console.

Un outil alternatif qui fournit des prévisions de durées de travaux estimées et des intervalles de confiance à travers le même support se base sur les statistiques **SPSS**. Pour plus d'informations, voir Chapitre 13, «Utilisation de statistiques avancées pour prévoir la durée estimée d'un travail», à la page 569.

Démarrage du traitement du plan de production

Pour démarrer un cycle de production, procédez comme suit :

1. Ouvrez une session en tant qu'*utilisateur_TWS* sur le gestionnaire de domaine maître.
2. À l'invite, exécutez la commande de script `./racine_TWS/tws_env.sh` sous UNIX ou `racine_TWS\tws_env.cmd` sous Windows pour configurer l'environnement, puis exécutez le travail `JnextPlan` en saisissant, par exemple sur un poste de travail UNIX, la commande suivante :

```
JnextPlan.sh -from 05/03/06 0400 tz Europe/Rome -to 06/06/06
```

Elle crée un nouveau plan de production qui démarre le 3 mai 2006 à 04:00 (fuseau horaire Europe/Rome) et qui s'arrête le 6 juin 2006 à 03:59. La journée de traitement commencera à l'heure définie sur le gestionnaire de domaine maître par la variable **startOfDay**.

3. Lorsque le travail `JnextPlan` est terminé, vérifiez l'état de IBM Workload Scheduler :

```
conman status
```

Si IBM Workload Scheduler a démarré correctement, le statut est

```
Batchman=ACTIF
```

Si mailman poursuit l'exécution d'un processus sur le poste de travail distant, vous pouvez constater que ce dernier ne s'initialise pas immédiatement. Cela se produit parce que le poste de travail doit terminer toutes les activités en cours impliquant le processus mailman avant de pouvoir être réinitialisé. Une fois que l'intervalle défini dans le paramètre *mm retry link*, lui-même défini dans le fichier de configuration *racine_TWS/localopts*, s'est écoulé, le gestionnaire de domaine tente à nouveau d'initialiser le poste de travail. Dès que les activités en cours sont terminées, les activités du jour suivant sont initialisées. Pour plus d'informations sur le fichier de configuration *localopts*, voir *IBM Workload Scheduler - Guide d'administration*.

4. Augmentez le nombre maximal de travaux de sorte que les travaux s'exécutent. Par défaut, le nombre maximal de travaux est nul après installation. Cela signifie qu'aucun travail ne s'exécutera.

```
conman "limit;10"
```

Automatisation du traitement du plan de production

Si vous étendez votre plan de production à intervalle régulier, par exemple chaque semaine, vous pouvez automatiser cette extension. La présente section explique la procédure à suivre pour cela.

Dans IBM Workload Scheduler versions 9.1 et ultérieures, le fichier *Sfinal* a été modifié pour inclure deux exemples de flot de travaux nommés *FINAL* et *FINALPOSTREPORTS* qui vous aident à automatiser la gestion du plan. Une copie de ces flots de travaux se trouvent dans le fichier *Sfinal* du répertoire *rép_base_TWS*. De plus, une copie des scripts de travaux se trouve également au même emplacement.

Vous pouvez utiliser le fichier *Sfinal* ou créer votre propre fichier, que vous pourrez personnaliser à votre guise.

Important : Dans tous les cas, pour pouvoir exécuter ces flots de travaux avec succès, *utilisateur_TWS* doit détenir les droits d'accès *Write* au répertoire temporaire *tmp* par défaut.

Le flot de travaux *FINAL* exécute la suite de fichiers scripts décrite dans *JnextPlan* pour générer le nouveau plan de production. Pour plus de détails, voir «Création et extension d'un plan de production», à la page 91.

Le flot de travaux *FINALPOSTREPORTS*, chargé de l'impression des rapports de post-production, suit le flot de travaux *FINAL* et démarre seulement si le dernier travail répertorié dans le flot de travaux *FINAL* (*SWITCHPLAN*) a abouti avec succès. Le flot de travaux *FINALPOSTREPORTS* comprend aussi un travail nommé *CHECKSYNC* qui surveille le progrès et le résultat de la commande **planman resync**. La commande **planman resync** charge les données de plan depuis le fichier *Symphony* vers la base de données.

Dans IBM Workload Scheduler versions 9.1 et ultérieures, les données de plan sont désormais complètement répliquées dans la base de données. A chaque fois que le plan est étendu, le plan dans la base de données est recréé, ce qui rend les informations les plus récentes accessibles à tous les utilisateurs. Si vous effectuez une nouvelle installation de IBM Workload Scheduler, cette synchronisation du fichier *Symphony* est alors automatiquement activée lorsque vous ajoutez le fichier *Sfinal* à la base de données avec la commande **composer add Sfinal**. Etant donné que le fichier *Sfinal* n'est pas écrasé lors d'une mise à niveau, vous devez ajouter

les flots de travaux FINAL et FINALPOSTREPORTS mis à jour à la base de données en exécutant la commande `composer add Sfinal`. Ceci permet de s'assurer que le travail CHECKSYNC qui est responsable de la réplication des données du plan se trouve dans la base de données. Le fichier Sfinal final mis à jour se trouve alors dans le répertoire `TWA_home/config/`. Vous devez ensuite exécuter `JnextPlan` pour inclure les flots de travaux FINAL et FINALPOSTREPORTS dans le plan de production en cours. Voir la section concernant la personnalisation et la soumission du flot de travaux final facultatif dans *Planification et installation*

Par défaut, le flot de travaux FINAL est paramétré pour s'exécuter une fois par jour, suivi par le flot de travaux FINALPOSTREPORTS. Vous pouvez modifier l'heure à laquelle les flots de travaux s'exécutent en modifiant deux paramètres dans la définition du flot de travaux. Pour ce faire (par exemple, pour que les flots de travaux s'exécutent tous les trois jours), procédez comme suit :

- Planifiez le flot de travaux de sorte qu'il s'exécute tous les trois jours en modifiant le cycle d'exécution dans la définition du flot de travaux.
- Dans l'instruction qui appelle **MakePlan** dans le flot de travaux FINAL, indiquez que le plan de production doit durer trois jours en spécifiant **-for 72**.

Vous devez ensuite ajouter les flots de travaux à la base de données en procédant comme suit :

1. Ouvrez une session en tant qu'*utilisateur_TWS*.
2. Exécutez le script **twc_env** pour installer ainsi l'environnement IBM Workload Scheduler :
 - UNIX : dans un shell C, lancez `./rep_base_TWS/twc_env.csh`
 - Sous UNIX : dans un shell Korn, lancez `./rep_base_TWS/twc_env.sh`
 - A partir d'une ligne de commande Windows, lancez `rep_base_TWS\twc_env.cmd`où *racine_TWS* représente le répertoire d'installation du produit.
3. Ajoutez les définitions des flots de travaux FINAL et FINALPOSTREPORTS à la base de données en exécutant la commande suivante :
`composer add Sfinal`

Si vous avez créé un nouveau fichier Sfinal plutôt que d'utiliser celui fourni avec le produit, utilisez le nom du nouveau fichier au lieu du nom Sfinal.

4. Démarrez le cycle de production en exécutant le script **JnextPlan**. Ainsi, les flots de travaux FINAL et FINALPOSTREPORTS seront inclus dans le plan de production en cours.

Remarque : Même si vous décidez d'automatiser l'extension du plan de production, vous pouvez toujours exécuter **JnextPlan** à tout moment.

Chapitre 5. Assurance de service de charge de travail

L'assurance de service de charge de travail est une fonction facultative permettant d'identifier les travaux critiques pour votre activité et de s'assurer qu'ils sont traités à temps. L'utilisation de cette fonction permet à votre personnel de planification des opérations d'améliorer sa capacité à atteindre les niveaux de service définis.

Lorsque la fonction d'assurance de service de charge de travail est activée, vous pouvez identifier les travaux critiques et vérifier qu'un délai de réalisation est spécifié pour la définition ou la soumission de ces travaux. Deux autres unités d'exécution, Time Planner et Plan Monitor, qui s'exécutent dans WebSphere Application Server, sont ensuite lancées pour garantir la réalisation des travaux critiques dans les délais.

La définition d'un travail critique et de son échéance déclenche le calcul des heures de début de tous les autres travaux qui sont des prédécesseurs du travail critique. L'ensemble de prédécesseurs d'un travail critique représente son *réseau critique*. Cela peut inclure des travaux provenant d'autres flots de travaux. A partir de l'échéance et de la durée du travail critique, Time Planner calcule son *heure de début critique*, qui est la dernière heure de début permettant au travail de respecter son échéance. En remontant à partir de l'heure de début critique, il calcule la dernière heure à laquelle chaque prédécesseur dans le réseau critique peut démarrer, de manière à pouvoir réaliser à temps le travail critique à la fin de la chaîne.

Pendant l'exécution du plan, Plan Monitor vérifie en permanence le réseau critique pour s'assurer que l'échéance peut être respectée. Lorsque des modifications ayant un impact sur les délais sont apportées au niveau du réseau critique (l'ajout ou la suppression de travaux ou de dépendances de prédécesseur/successeur, par exemple) Plan Monitor demande à Time Planner de recalculer les heures de début critiques. De même, à la fin d'un travail réseau critique, les délais des travaux lui succédant sont recalculés de façon à prendre en compte la durée réelle du travail.

Dans un réseau critique, l'ensemble de prédécesseurs les plus susceptibles de retarder directement l'heure de début critique est appelé *chemin critique*. Le chemin critique est mis à jour de manière dynamique, à mesure des modifications apportées à la réalisation ou aux risques de réalisation des prédécesseurs.

Le planificateur (batchman) s'exécute automatiquement de façon à corriger les retards en donnant priorité aux travaux représentant un risque réel ou potentiel pour l'échéance cible, bien que certaines conditions à l'origine de retards puissent faire l'objet d'une intervention de l'opérateur. Une série de vues de travaux critiques spécialisés, disponible sur la console Dynamic Workload Console, permet à l'opérateur de rechercher des travaux critiques, d'afficher leurs prédécesseurs et les chemins critiques qui leur sont associés, d'identifier les travaux entraînant des problèmes et d'explorer en aval de façon à identifier et corriger les problèmes.

Pour obtenir des informations détaillées, voir :

- «Activation et configuration de l'assurance de service de charge de travail», à la page 118
- «Planification de travaux critiques», à la page 122
- «Traitement et contrôle des travaux critiques», à la page 123
- «Scénario d'assurance de service de charge de travail», à la page 126

Pour plus d'informations sur l'identification et la résolution des problèmes et sur les problèmes habituels liés à l'assurance de service de charge de travail, voir le chapitre Assurance de service de charge de travail dans *IBM Workload Scheduler : Identification et résolution des problèmes*.

Activation et configuration de l'assurance de service de charge de travail

Un certain nombre d'options globales et locales permettent de contrôler la gestion des travaux critiques. Le fichier de sécurité IBM Workload Scheduler doit également autoriser les utilisateurs à accéder à tous leurs travaux, flots de travaux et postes de travail associés aux travaux critiques.

Options globales

La fonction d'assurance de service de charge de travail est activée et désactivée par l'option globale `enWorkloadServiceAssurance`. Elle est activée par défaut. D'autres options globales et locales permettent de contrôler les différents aspects du traitement des travaux critiques et de leurs prédécesseurs.

Le tableau 14 présente les options globales utilisées par l'assurance de service de charge de travail. Si vous souhaitez personnaliser les valeurs, modifiez les options globales dans le gestionnaire de domaine maître à l'aide de la ligne de commande **optman**. Dans la plupart des cas, les modifications prennent effet après l'exécution suivante de JnextPlan.

Tableau 14. Options globales de l'assurance de service de charge de travail

Option	Description
<code>enWorkloadServiceAssurance</code> <code>wa</code>	Active ou désactive le traitement privilégié des travaux critiques et de leurs prédécesseurs. La valeur par défaut est YES. Entrez NO pour la désactiver.
<code>promotionOffset</code> <code>po</code>	L'assurance de service de charge de travail calcule l'heure de début critique du travail critique lui-même et de chacun de ses prédécesseurs. Il s'agit de la dernière heure à laquelle peut démarrer le travail sans menacer le déroulement opportun du travail critique. Lorsque l'heure de début critique d'un travail approche et que le travail n'a pas démarré, le mécanisme de promotion est utilisé. Des ressources de système d'exploitation sont affectées au travail promu et sa soumission est priorisée. L'option globale <code>promotionoffset</code> détermine la longueur de temps avant l'heure de début critique au cours de laquelle le travail devient admissible pour promotion. La valeur par défaut est de 120 secondes.

Tableau 14. Options globales de l'assurance de service de charge de travail (suite)

Option	Description
longDurationThreshold 1d	Le calcul des heures de début critiques des travaux composant un réseau critique repose sur l'échéance définie pour le travail critique et les durées estimées du travail critique et de chacun de ses prédécesseurs. Si le déroulement d'un travail prend plus de temps que prévu, les travaux qui le suivent risquent de manquer leurs heures de début critiques et, par conséquent, de menacer le bon déroulement du travail critique. L'option globale longDurationThreshold est une valeur de pourcentage. La valeur par défaut est 150. Grâce à la valeur par défaut, si la durée réelle d'un travail représente 150 % de la durée estimée ou plus, le travail est considéré comme de longue durée. Il est donc ajouté à la liste d'accès direct qui peut s'afficher dans Dynamic Workload Console.
approachingLateOffset a1	L'heure de début critique d'un travail du réseau critique est la dernière heure à laquelle le travail peut commencer sans obliger le travail critique à se terminer après l'échéance. Dans la plupart des cas, un travail commence bien avant l'heure de début critique afin que, s'il se prolonge plus longtemps que sa durée estimée, la situation ne devienne pas immédiatement critique. En conséquence, si un travail n'a pas commencé et que l'heure de début critique n'est qu'à quelques minutes, l'achèvement du travail dans les délais est considéré comme potentiellement menacé. L'option approachingLateOffset permet de déterminer combien de temps avant l'heure de début critique d'un travail du réseau critique vous devez être informé de ce risque potentiel. Si un travail n'a pas encore démarré dans le nombre spécifié de secondes avant l'heure de début critique, le travail est ajouté à une liste d'accès direct qui peut s'afficher dans la console Dynamic Workload. La valeur par défaut est 120 secondes. Remarque : La valeur de ce paramètre est vérifiée régulièrement. Il n'est pas utile d'exécuter JnextPlan pour que les modifications prennent effet.

Tableau 14. Options globales de l'assurance de service de charge de travail (suite)

Option	Description
<code>deadlineOffset</code> <code>do</code>	En général, une échéance doit être précisée pour un travail marqué <code>critical</code>. Dans le cas contraire, le planificateur utilise l'échéance définie pour le flot de travaux. L'option <code>deadlineOffset</code> offre un décalage permettant de calculer l'heure de début critique en cas d'absence d'échéance pour le travail critique et le flot de travaux. La fin du plan plus le décalage sont considérés comme l'échéance du travail critique. Le décalage est exprimé en minutes. La valeur par défaut est de 2 minutes. Important : Si le plan est étendu, l'heure de début des travaux critiques dont l'échéance est calculée avec ce mécanisme est automatiquement modifiée en raison du fait qu'elle doit à présent concorder avec la nouvelle heure de fin du plan.

Pour de plus amples informations sur les options globales, voir *IBM Workload Scheduler - Guide d'administration*.

Options locales

L'assurance de service de charge de travail utilise des options locales afin de contrôler l'allocation de priorité des ressources système aux travaux du réseau critique qui doivent être promus pour maintenir l'échéance critique. Le tableau 15, à la page 121 présente les options locales utilisées par la fonction d'assurance de service de charge de travail. Pour définir les options locales, éditez le fichier `twshome\localopts` de chaque poste de travail sur lesquels vont être exécutés les travaux critiques. Exécutez `JnextPlan` ou redémarrez l'agent pour que les modifications apportées aux options locales prennent effet.

Tableau 15. Options locales de l'assurance de service de charge de travail

Option	Description
jm promoted nice	Définissez la valeur nice à attribuer aux travaux critiques ou aux prédécesseurs du travail critique devant être promus sous UNIX et Linux, de manière à les attribuer à plusieurs ressources et à les traiter avant les autres travaux. Les valeurs spécifiques varient selon les plateformes, mais d'une manière générale, le paramètre doit être un entier négatif. La valeur par défaut est -1, des valeurs inférieures représentant des priorités plus élevées. Si vous indiquez un entier positif, la valeur par défaut est utilisée. Le rôle de l'option locale jm nice est analogue, donnant la priorité aux travaux ayant été soumis par le superutilisateur. Un travail critique qui a été soumis par le superutilisateur peut être admissible pour les mécanismes de priorisation. Dans ce cas, les valeurs sont ajoutées. Par exemple, si les valeurs -4 et -2 sont respectivement attribuées à jm promoted nice et jm nice, la priorité du travail critique soumis par le superutilisateur est de -6.
jm promoted priority	Définissez la valeur de priorité des travaux critiques ou des prédécesseurs du travail critique qui doivent être promus, de sorte que le système d'exploitation Windows puisse leur affecter d'autres ressources et les traiter avant les autres travaux. Les valeurs possibles sont : • High • AboveNormal • Normal • BelowNormal • Low ou Idle La valeur par défaut est AboveNormal. Notez que si vous définissez une valeur de priorité inférieure à celle qui peut être attribuée aux travaux non critiques, aucun avertissement n'est envoyé et aucun mécanisme comme celui disponible pour jm promoted nice ne rétablit la valeur par défaut.

Exigences relatives au fichier de sécurité

Les utilisateurs en possession des instances IBM Workload Scheduler qui exécutent les travaux critiques doivent obligatoirement être autorisés à utiliser tous les travaux, flots de travaux et postes de travail associés à ces travaux. Ces utilisateurs

doivent par conséquent disposer des droits DISPLAY, MODIFY et LIST, dans leur fichier de sécurité, pour tous les objets associés JOB, SCHEDULE et CPU.

Voir IBM Workload Scheduler - *Guide d'administration* pour plus d'informations sur le fichier de sécurité.

Planification de travaux critiques

L'assurance de service de charge de travail permet d'identifier des travaux critiques, définir des échéances et de calculer les délais de l'ensemble des travaux devant précéder le travail critique.

Si un travail doit absolument être terminé avant une heure spécifique, vous pouvez le signaler comme critique lorsque vous l'ajoutez à un flot de travaux à l'aide des fonctions du concepteur de charge de travail sur Dynamic Workload Console. Vous pouvez définir l'échéance au niveau du travail ou du flot de travaux.

Les travaux peuvent également être signalés comme critiques. Pour ce faire, indiquez le mot clé **critique** lors de la création ou de la modification d'un flot de travaux à l'aide de la ligne de commande **composer**.

Lorsque la commande **JnextPlan** est exécutée afin d'inclure le nouveau travail dans le plan de production, tous les travaux correspondants à des prédécesseurs directs ou indirects du travail critique sont identifiés. Ces travaux, associés au travail critique lui-même, forment un réseau critique.

Puisque le délai des travaux du réseau critique doit être étroitement contrôlé, Time Planner calcule les tests de performances des délais suivants pour chaque travail réseau critique :

Démarrage critique

S'applique aux systèmes répartis uniquement et représente la dernière heure à laquelle le travail peut démarrer sans que le travail critique dépasse son échéance.

Les heures de début critiques sont calculées à partir de l'échéance définie pour le travail critique et fonctionnent de façon rétroactive à l'aide de la durée estimée pour chaque travail afin de déterminer sa durée critique. Par exemple, si l'échéance du travail critique s'opère à 19:00 et que la durée estimée du travail critique est de 30 minutes, le travail critique ne sera pas terminé au moment de l'échéance, à moins qu'il n'ait commencé à 18:30. Si le prédécesseur immédiat du travail critique dispose d'une durée estimée de 20 minutes, il doit démarrer au plus tard à 18:10.

Remarque : Seule l'échéance du travail critique est prise en compte lors du calcul des heures de début critiques des travaux dans le réseau critique. Si des échéances ont été définies pour d'autres travaux, leurs heures de début critiques peuvent être ultérieures à leurs échéances.

Premier démarrage

Représente l'heure la plus tôt à laquelle un travail du réseau critique peut démarrer, en prenant en compte toutes les dépendances et les besoins en ressources.

Heures de début et de fin estimées

Les heures de début estimées sont calculées à partir de l'heure la plus tôt à laquelle le premier travail ou les travaux du réseau critique peuvent

démarrer et fonctionnent de façon rétroactive à l'aide de la durée estimée pour chaque travail afin de déterminer l'heure de début du travail suivant.

Heures de début et de fin planifiées

Pour les calculs initiaux, ces valeurs sont définies sur les heures de début et de fin estimées. Elles sont ensuite recalculées afin de prendre en compte toute modification ou retard au niveau du plan.

Durée estimée

La durée estimée d'un travail est basée sur les statistiques collectées depuis des exécutions antérieures du travail. Prenez ceci en compte lorsque vous considérerez la précision des délais calculés pour les réseaux de travaux critiques incluant les travaux s'exécutant pour la première fois. Dans le cas d'un travail reflet, la durée estimée est toujours définie sur la valeur par défaut d'une minute. Ceci s'applique aux travaux reflets qui s'exécutent pour la première fois, aussi bien que pour toute exécution suivante de travail reflet.

Facteur de fiabilité

Pour chaque travail critique, un pourcentage indique le niveau de fiabilité du travail critique pour respecter son échéance. A l'issue de l'exécution d'un travail, le facteur de fiabilité est écrasé et défini sur 0 % lorsque l'estimation de l'échéance est dépassée et sur 100 % lorsque la date d'échéance n'a pas été dépassée.

Les délais de chaque travail du réseau critique sont ajoutés au fichier Symphony, qui inclut toutes les informations relatives au plan et qui sont réparties sur l'ensemble des postes de travail sur lesquels les travaux doivent être exécutés.

Lors de l'exécution du plan, Plan Monitor examine tous les réseaux critiques : les modifications ultérieures apportées au réseau critique ayant une incidence sur le délai des travaux entraîneront le calcul de l'heure de début estimée et critique. Les modifications peuvent inclure des modifications manuelles, par exemple la publication de dépendances ou la réexécution de travaux, et les modifications automatiquement effectuées par le système en réponse à un risque réel ou potentiel concernant l'achèvement ponctuel d'un travail critique.

Les vues spécifiques des travaux critiques et de leurs prédécesseurs, disponibles dans Dynamic Workload Console, vous permettent de suivre le traitement du réseau critique. Les vues permettent immédiatement d'identifier des problèmes concernant votre planification du travail critique. Par exemple, si l'heure de début estimée d'un travail du réseau critique est postérieure à l'heure de début critique, cela est immédiatement signalé comme risque potentiel au niveau du travail critique.

Traitement et contrôle des travaux critiques

L'assurance de service de charge de travail offre un suivi automatique et permet de prioriser les travaux réseau critiques ainsi que les fonctions en ligne vous permettant de surveiller et d'intervenir au cours du traitement des travaux réseau critiques.

Suivi automatique et priorisation

Pour veiller à ce que les délais critiques soient respectés, l'assurance de service de charge de travail offre les services automatisés suivants aux travaux critiques et aux travaux prédécesseurs formant leurs réseaux critiques :

Promotion

Lorsque l'heure de début critique d'un travail approche et que le travail n'a pas démarré, le mécanisme de promotion est utilisé. Des ressources de système d'exploitation sont affectées au travail promu et sa soumission est priorisée.

Le délai des promotions est contrôlé par l'option globale `promotionoffset`. Les travaux promus sont sélectionnés pour être soumis après des travaux possédant les priorités "Elevée" et "Aller", mais avant tous les autres travaux. La priorisation des ressources du système d'exploitation est contrôlée par les options locales `jm promoted nice` (UNIX et Linux) et `jm promoted priority` (Windows).

Calcul du chemin critique

Le chemin critique correspond à la chaîne des dépendances, menant au travail critique, le plus à même de dépasser l'échéance à un moment donné. Le chemin critique est calculé à l'aide des heures de fin des prédécesseurs du travail critique. Issu du travail critique, le chemin est construit en sélectionnant le prédécesseur avec l'heure de fin estimée la plus récente. Si l'heure de fin réelle diffère de façon substantielle de l'heure de fin estimée, le chemin critique est automatiquement recalculé.

La figure 22, à la page 125 présente le chemin critique via un réseau critique à un moment spécifique de cohérence lors du traitement du plan.

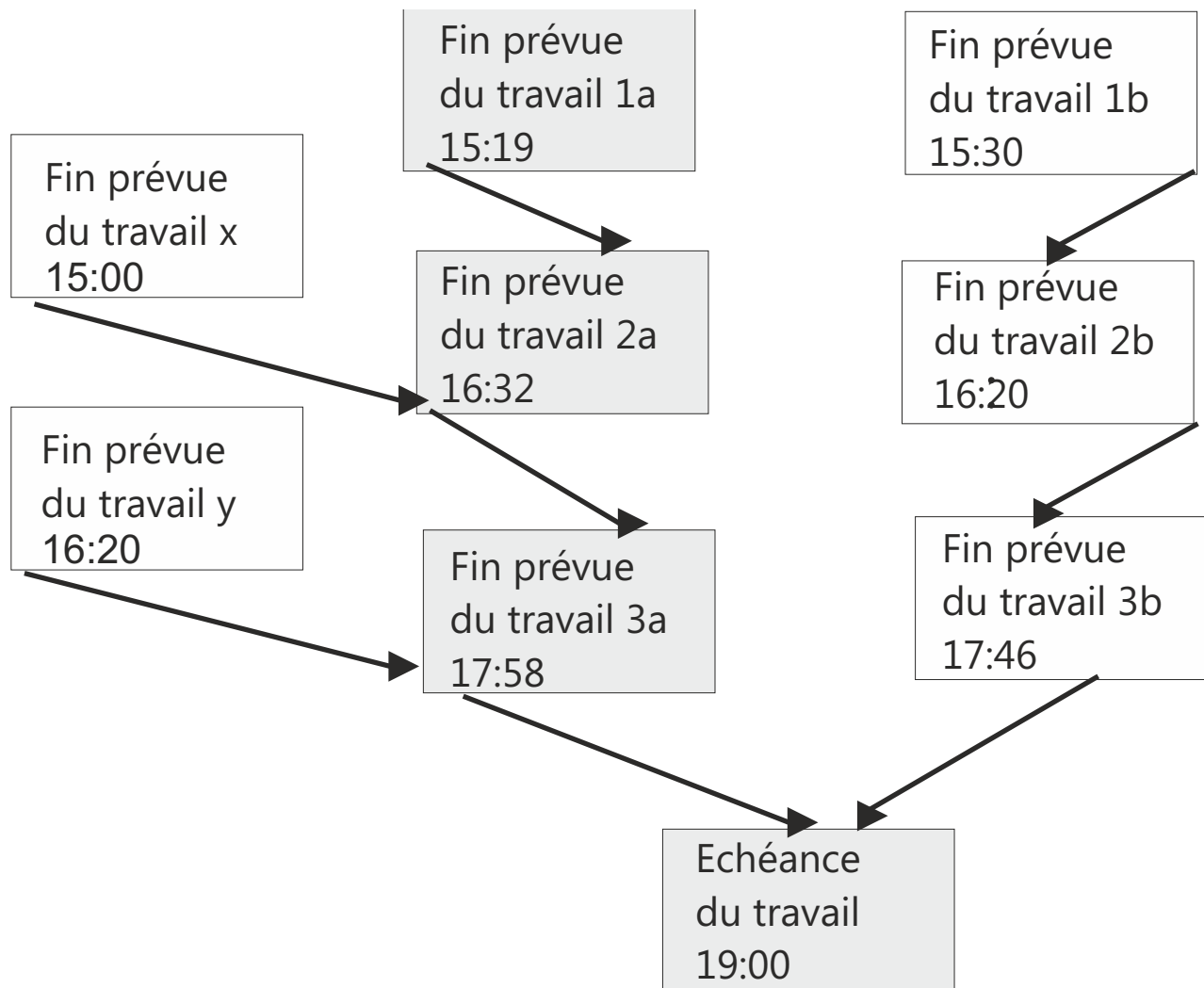


Figure 22. Chemin critique

A ce point de cohérence, le chemin critique inclut Job3a, Job2a et Job1a. Job3a et Job3b correspondent aux prédécesseurs immédiats du travail critique, Job4, et Job3a possède la date de fin estimée la plus récente. Job3a possède deux prédécesseurs immédiats, Job2a et Job_y. Job2a possède l'heure de fin estimée la plus récente, etc.

Ajout de travaux à la liste d'accès direct

Les travaux inclus dans le réseau critique sont ajoutés à la liste d'accès direct associée au travail critique lui-même. La liste d'accès direct inclut les travaux réseau critiques possédant un impact réel ou potentiel sur l'achèvement ponctuel du travail critique. Les travaux sont ajoutés à la liste d'accès direct pour une ou plusieurs des raisons répertoriées ci-après. Notez que seuls les travaux commençant le réseau critique en cours, pour lesquels il n'y a pas de prédécesseur, peuvent être inclus dans la liste d'accès direct.

- Le travail a été arrêté avec une erreur. La durée comprise avant l'heure de début critique est déterminée par l'option globale `approachingLateOffset`.
- Le travail s'est exécuté pendant une durée supérieure à celle estimée par un facteur défini dans l'option globale `longDurationThreshold`.

- Le travail n'a toujours pas démarré, bien que toutes ses dépendances de prédécesseur ont été résolues ou publiées, et au moins l'une des conditions suivantes est vraie :
 - L'heure de début critique a pratiquement été atteinte.
 - Le travail est programmé pour s'exécuter sur un poste de travail pour lequel la limite est définie sur zéro.
 - Le travail appartient à un flot de travaux pour lequel la limite est définie sur zéro.
 - Le travail ou son flot de travaux a été supprimé.
 - Le travail ou son flot de travaux dispose actuellement d'une priorité inférieure à la priorité minimale ou est défini sur zéro.

Définition d'un statut de risque élevé ou potentiel pour un travail critique

Une statut de risque peut être défini pour le travail critique, comme suit :

Risque élevé

Les délais calculés indiquent que le travail critique se terminera après son échéance.

Risque potentiel

Les travaux prédécesseurs critiques ont été ajoutés à la liste d'accès direct.

Suivi en ligne des travaux critiques

Dynamic Workload Console fournit des vues spécialisées afin de suivre le déroulement des travaux critiques et de leurs prédécesseurs. Vous pouvez accéder aux vues depuis le tableau de bord ou créer une tâche pour surveiller les tâches critiques à l'aide de la requête Surveiller la charge de travail.

La vue initiale répertorie tous les travaux critiques associés au moteur et affichant l'état : normal, risque potentiel ou risque élevé. Vous pouvez parcourir cette vue et consulter :

- La liste d'accès direct des travaux représentant un risque pour l'échéance critique.
- Le chemin critique.
- Les détails relatifs à l'ensemble des prédécesseurs critiques.
- Les détails relatifs aux prédécesseurs critiques terminés.
- Les fichiers journaux des travaux déjà exécutés.
- Le facteur de fiabilité exprimé sous forme de pourcentage. La probabilité qu'un travail critique respecte son échéance.

Ces vues vous permettent de suivre le déroulement du réseau critique, rechercher des problèmes actuels ou potentiels, publier des dépendances ou réexécuter des travaux.

Scénario d'assurance de service de charge de travail

Ce scénario illustre l'utilisation de l'assurance de service de charge de travail, en vérifiant que les échéances de production importantes peuvent être respectées.

Fine Cola utilise IBM Workload Scheduler pour gérer les délais et les interdépendances de sa production et de son processus d'approvisionnement.

Fine Cola dispose de contrats de service avec de nombreux clients prenant en charge le réassortiment "en flux tendu", ce qui signifie qu'en cas de retard sur un trajet de livraison, les produits Fine Cola n'arriveront sans doute pas à temps sur les étagères.

Le travail qui génère les commandes de chargement des camions doit être terminé au plus tard à 5h30 du matin. Ce travail dépend de la réussite des autres travaux. Par exemple, même si les commandes sont traitées à l'avance, une modification de dernière minute arrive souvent après le retour de livraison des camions. Fine Cola fournit également des factures avec des notes de livraison. Toute modification de commande doit donc être répercutée dans le tarif et peut être à l'origine d'offres spéciales.

Planification du travail critique

A l'aide du concepteur de charge de travail sur Dynamic Workload Console, le planificateur Fine Cola signale le travail de commande de chargement comme critique et définit l'échéance sur 5h du matin.

Lorsque **JnextPlan** est exécuté, les dates de début critiques de ce travail et de tous ceux qui sont signalés comme ses prédécesseurs sont calculées.

Suivi du travail critique

1. L'opérateur IBM Workload Scheduler vérifie les tableaux de bord et note que des travaux critiques sont planifiés sur l'un des moteurs.
2. Il voit qu'un travail critique est en état de risque potentiel. Il clique sur le lien du risque potentiel pour obtenir la liste des travaux critiques dans cet état.
Le statut du travail de commandes de chargement est défini sur "Risque potentiel".
3. Il sélectionne le travail et clique sur **Accès direct** pour afficher le ou les travaux à l'origine du risque auquel est exposé le travail critique.
Une erreur est indiquée pour le travail d'ajustement des commandes.
4. Il sélectionne le travail et clique sur **Journal des travaux**.
Le journal indique que le travail a échoué en raison de données d'identification incorrectes fournies à la base de données des commandes.
5. Lorsqu'il s'aperçoit que le mot de passe de la base de données a été changé ce jour-là, il modifie la définition du travail dans le fichier Symphony, puis réexécute le travail.
6. Lorsqu'il revient au tableau de bord, il voit qu'il n'y a plus de travaux en état de risque potentiel. De plus, la liste des travaux critiques, ouverte en cliquant sur le lien du risque potentiel, n'affiche plus le travail critique une fois qu'il a été réexécuté.
7. Le travail est en cours d'exécution et il a fait l'objet d'une promotion automatique afin d'avoir priorité pour la soumission et les ressources système.
8. Il n'y a pas besoin de résoudre d'autres incidents et le travail critique s'achève à 4h45 du matin.

Chapitre 6. Personnalisation de votre charge de travail à l'aide des tables de variables

Ce chapitre présente le concept de table de variables permettant de regrouper les paramètres globaux, désormais appelés variables, afin de personnaliser votre charge de travail.

Dans IBM Workload Scheduler version 8.5, les paramètres globaux auxquels faisaient référence les versions précédentes sont désormais appelés variables. Les définitions de variable sont contenues dans les tables de variables. Une table de variables est un objet regroupant plusieurs variables. Grâce à ces tables, vous pouvez attribuer des valeurs différentes à la même variable et les utiliser dans les définitions de travail et de flot de travaux dans JCL, l'ouverture de session, les dépendances d'invite, les dépendances de fichier et les invites de reprise. Elles sont particulièrement utiles lorsqu'une définition de travail est utilisée en tant que modèle pour un travail appartenant à plusieurs flots de travaux. Par exemple, vous pouvez affecter différentes valeurs à la même variable et réutiliser la même définition de travail dans plusieurs flots de travaux, ce qui vous permet de gagner du temps et de l'argent.

Lorsque vous définissez une variable, vous l'attribuez à une table de variables étant donné que la même variable peut être définie dans différentes tables de variables avec différentes valeurs. Sinon, vous pouvez créer une ou plusieurs tables de variables, en spécifiant une liste de noms et de valeurs de variables pour chaque table. De plus, vous pouvez ajouter le même nom de variable avec différentes valeurs dans différentes tables. Lorsque vous demandez une liste de variables, vous obtenez des paires *table_variables.nom_variable* permettant d'identifier aisément à quelle table de variables appartient la variable.

Par exemple, la variable VAR1 définie dans la table de variables REP1_TABLE1 se présente de la manière suivante :

```
REP1_TABLE1.VAR1
```

Vous pouvez affecter des tables de variables aux cycles d'exécution, aux flots de travaux et aux postes de travail.

Grâce aux tables de variables, vous pouvez modifier le comportement de la charge de travail en fonction du moment, de la raison et de l'endroit où vous souhaitez exécuter votre planification, ce qui donne plus de souplesse quant à la personnalisation de votre charge de travail et satisfait le contrat de service. Plus précisément :

Quand

Pour modifier le comportement des travaux et des flots de travaux en fonction du moment de la planification de leur exécution, c'est-à-dire du jour de leur exécution. Utilisation des tables de variables avec les cycles d'exécution.

Pourquoi

Pour modifier le comportement des travaux et des flots de travaux selon la raison qui explique la planification de leur exécution (pour créer un travail qui exécute différentes commandes, par exemple). Utilisation des tables de variables avec des flots de travaux.

Où Pour modifier le comportement des travaux et des flots de travaux selon l'emplacement de leur exécution (sur différents postes de travail, par exemple). Utilisation des tables de variables avec des postes de travail.

Migration des paramètres globaux à partir de versions précédentes

Remarques lors de la migration des paramètres globaux à partir de versions précédentes

Lors de la mise à niveau à partir de versions antérieures à la version 8.5, les définitions de paramètre global (désormais appelées définitions de variable) dont dispose la base de données sont automatiquement migrées vers la table de variables par défaut appelée MAIN_TABLE. Après la mise à niveau :

- Toutes les variables sont précédées par le nom de table par défaut. Par exemple, après la migration, la variable *REP_PATH* porte le nom suivant :

MAIN_TABLE.REP_PATH

Lorsque vous demandez une liste de variables, vous obtenez des paires *table_variable.nom_variable* permettant d'identifier aisément à quelle table de variables appartient la variable.

- Votre charge de travail est résolue de la même manière qu'avant la migration, puisque tous les objets IBM Workload Scheduler contenant des variables font référence à MAIN_TABLE pour la résolution de variable.
- Pour toute section utilisateur incluant le mot clé parameter, la ligne suivante est ajoutée dans le fichier de sécurité :

```
var table name=@ access=add,delete,display,modify,list,use,unlock
```

Pour plus d'informations sur le processus de mise à niveau, voir *IBM Workload Scheduler: Planification et installation*.

Lors de la mise à niveau à partir de la version 8.3 ou ultérieure, ne modifiez pas les variables tant que vous n'avez pas migré le gestionnaire de domaines maîtres et tous ses maîtres de sauvegarde. En effet, lors de cette phase de transition, vous disposez de deux versions différentes de la base de données. Si vous devez ajouter ou modifier des variables au cours de cette phase de transition, vérifiez que les modifications ont été effectuées dans la version 8.3 ou 8.4 et la version 8.5 des gestionnaires de domaine maître.

Le fonctionnement des paramètres locaux qui ont été créés et gérés avec la commande d'utilitaire **parms** dans la base de données de paramètre global sur les postes de travail ne change pas.

Dans la version 8.3 et 8.4, les paramètres sont stockés dans la table MDL.VAR_VARIABLES de la base de données. Après avoir effectué une mise à niveau vers la version 8.5 ou ultérieure, les mêmes paramètres sont stockés dans la MAIN_TABLE qui se trouve dans la table de base de données MDL.VAR_VARIABLES2. Si vous avez commencé à migrer votre environnement, mais n'avez pas terminé la migration, et si la version du gestionnaire de domaine maître est 8.3 ou 8.4 et que celle du gestionnaire de domaine maître de sauvegarde est 8.5 ou ultérieure, ou inversement, le gestionnaire de domaine maître version 8.3 ou 8.4 ne reconnaît pas la table MDL.VAR_VARIABLES2. Ainsi, si vous modifiez un paramètre dans le gestionnaire de domaine maître version 8.3 ou 8.4, cette modification est stockée dans l'ancienne table (MDL.VAR_VARIABLES). Si vous modifiez le paramètre dans le gestionnaire de domaine maître version 8.5, la modification est stockée dans la table MDL.VAR_VARIABLES2.

Table de variables par défaut

Cette rubrique décrit la table de variables par défaut et son fonctionnement.

La table de variables par défaut est celle contenant toutes les variables que vous avez définies sans préciser un nom de table de variables. Le nom par défaut de la table de variables par défaut est **MAIN_TABLE**. Vous pouvez le modifier à tout moment ou définir une autre table de variables comme table de variables par défaut. Vous ne pouvez pas supprimer la table de variables par défaut. Si vous définissez une autre table de variables par défaut, l'originale n'est plus la table par défaut. Vous pouvez utiliser la table de variables par défaut comme toute autre table de variables. Vous pouvez aisément identifier la table de variables par défaut sur l'interface utilisateur car elle est marquée d'un **Y** dans la zone de **valeur par défaut**.

Exemple

L'exemple ci-dessous présente une liste de tables de variables

Variable	Table Name	Default	Updated On	Locked By
MAIN_TABLE	Y	05/07/2008	-	
VT_1		05/07/2008	-	
VT_2		05/07/2008	-	
VARTABLE3		05/07/2008	-	
V4		05/07/2008	-	
VT5		05/07/2008	-	

AWSBIA291I Total objects: 6

Intégrité des données des tables de variables

Cette rubrique explique comment assurer l'intégrité des données à l'aide des tables de variables.

Comme pour les autres objets, IBM Workload Scheduler assure l'intégrité des données de la table de variables à chaque exécution des commandes permettant de créer, de modifier ou de supprimer la définition d'une table de variables.

Lors du référencement d'une table de variable à partir d'un cycle d'exécution, d'un flot de travaux ou d'un poste de travail, IBM Workload Scheduler vérifie que la table de variable existe et sauvegarde son lien avec le cycle d'exécution, le flot de travaux et le poste de travail. Cela signifie que vous ne pouvez pas supprimer une définition de table de variables s'il existe une référence à un cycle d'exécution, un flot de travaux ou un poste de travail.

L'intégrité référentielle est assurée au niveau de la table de variables et non à celui de la variable. En d'autres termes, si un cycle d'exécution, un flot de travaux et un poste de travail font référence à une table de variables, IBM Workload Scheduler s'assure de l'existence de la table de variables et non de la variable référencée.

Mécanisme de verrouillage des tables de variables

Cette rubrique explique le fonctionnement du mécanisme de verrouillage des tables de variables.

Le verrouillage est défini au niveau de la table de variables afin de s'assurer que les définitions présentes dans la base de données ne sont pas écrasées par

différents utilisateurs qui accèdent en même temps à la même table de variables. Cela signifie que lorsque vous verrouillez une table de variables et une variable, vous détenez les droits d'accès exclusifs à toutes les variables de cette table de variables. En d'autres termes, vous pouvez exécuter des commandes sur la table de variables verrouillée et sur les variables qu'elle contient. Les autres utilisateurs ne détiennent qu'un accès en lecture seule à cette table de variables et aux variables qu'elle contient.

Cela empêche tous les autres utilisateurs de modifier la même table de variables que vous. Si un autre utilisateur tente de verrouiller une table de variables ou une variable déjà verrouillée, un message d'erreur s'affiche.

Sécurité de la table de variables

Cette rubrique explique comment définir les paramètres de sécurité des tables de variables.

L'accès utilisateur aux tables de variables doit être autorisé dans le fichier de sécurité IBM Workload Scheduler. Comme pour les autres objets, le connecteur vérifie l'existence de l'autorisation avant d'exécuter une action requérant un accès à une table de variables. Le nouveau mot clé est disponible dans le fichier de sécurité dans ce but :

```
var table name=@ access=add,delete,display,modify,list,use,unlock
```

Vous avez besoin de l'accès `use` pour référencer une table de variables à partir d'autres objets (flots de travaux, cycles d'exécution et postes de travail). Les filtres de sécurité sont uniquement basés sur l'attribut `name`, mais votre administrateur IBM Workload Scheduler peut utiliser le mot clé **\$default** pour spécifier des autorisations de sécurité dans la table par défaut, quel que soit son nom.

L'autorisation de travailler sur une variable n'est plus basée sur la variable individuelle mais sur la table correspondante. L'accès à une variable est uniquement autorisé si l'action correspondante de la table de variables est autorisée. Le tableau suivant présente les autorisations correspondantes pour les variables et les tables de variables :

Tableau 16. Relation entre les tables de variables et leurs variables correspondantes dans le fichier de sécurité IBM Workload Scheduler

Accès défini à var table	Action autorisée sur les variables correspondantes
modify	add
	delete
	modify
display	display
unlock	unlock

Avec la version 8.5, le mot clé `parameter` du fichier de sécurité s'applique aux paramètres locaux uniquement.

Voir *IBM Workload Scheduler - Guide d'administration* pour plus d'informations sur le fichier de sécurité.

Résolution de variable

Cette rubrique explique comment les variables sont résolues lors de la génération d'un plan et de la soumission d'un travail et d'un flot de travaux à exécuter.

Le format utilisé pour spécifier une variable peut également déterminer le moment où une variable est résolue avec une valeur. Voir «Définition des variables et des paramètres», à la page 206 pour plus d'informations sur les formats que vous pouvez utiliser.

Lorsque vous générez un plan, IBM Workload Scheduler analyse les tables de variables dans l'ordre présenté ci-dessous pour la résolution de variable :

1. Dans le cycle d'exécution. Le cycle d'exécution du flot de travaux est vérifié en premier, puis le cycle d'exécution d'un groupe de cycle d'exécution, et enfin la table de variable définie au niveau du groupe de cycle d'exécution.
2. Dans le flot de travaux.
3. Dans le poste de travail. Voir «Poste de travail considéré pour la résolution de variable».
4. Dans la table de variables par défaut mais uniquement quand les variables sont spécifiées dans le format `^nomvariable^` dans la définition du travail. Les variables spécifiées dans le format `${nomvariable}` ne sont pas résolues.

Au moment de la résolution du plan, chaque niveau est analysé dans l'ordre décrit ci-dessus. Si vous spécifiez une variable qui ne figurent dans aucune table de variables, y compris la table de variables par défaut, un message d'avertissement contenant le nom de la variable non résolue est écrit dans le fichier journal `<chemin_profil_WAS>\logs\twaserver\SystemOut.log`, tandis que le nom de variable est conservé dans le plan, où le chemin d'accès par défaut de `<chemin_profil_WAS>` est `<rép_principale_TWA>/WAS/TWSpofile`.

Lors de la soumission d'un flot de travaux, IBM Workload Scheduler résout les variables en analysant les tables de variables dans l'ordre ci-dessous :

1. Spécifiée lors de l'opération de soumission.
2. Dans le flot de travaux.
3. Dans le poste de travail. Voir «Poste de travail considéré pour la résolution de variable».
4. Dans la table de variables par défaut mais uniquement quand les variables sont spécifiées dans le format `^nomvariable^` dans la définition du travail. Les variables spécifiées dans le format `${nomvariable}` ne sont pas résolues.

Lors de la soumission d'un travail, IBM Workload Scheduler résout les variables en analysant les tables de variables dans l'ordre ci-dessous :

1. Spécifiées durant l'opération de soumission mais uniquement quand les variables sont spécifiées dans le format `^nomvariable^` dans la définition du travail. Les variables spécifiées dans le format `${nomvariable}` ne sont pas résolues.
2. Dans le poste de travail. Voir «Poste de travail considéré pour la résolution de variable».
3. Dans la table de variables par défaut mais uniquement quand les variables sont spécifiées dans le format `^nomvariable^` dans la définition du travail. Les variables spécifiées dans le format `${nomvariable}` ne sont pas résolues.

Poste de travail considéré pour la résolution de variable

Lorsque la variable est résolue par la table de variables spécifiée dans le poste de travail, le poste de travail pris en compte est :

Pour la variable d'une dépendance de fichier

Celui sur lequel réside le fichier.

Pour la variable d'un travail

Celui sur lequel est défini le travail.

Pour la variable d'une dépendance d'invite

Invite globale

Aucun poste de travail n'est pris en compte. Les variables des invites globales sont toujours résolues à l'aide de la table de variables par défaut. En effet, l'invite globale est utilisée par tous les travaux et flots de travaux de manière à n'utiliser qu'une seule valeur pour la résolution des variables.

Invite Ad hoc

Celui sur lequel est défini le travail ou le flot de travaux qui dépend de la dépendance d'invite.

Chapitre 7. Automatisation conditionnelle de la charge de travail

L'automatisation conditionnelle de la charge de travail permet de faire démarrer vos flux de travaux juste au bon moment de manière simple et immédiate. Dans votre flot de travaux, vous pouvez définir une condition de démarrage qui, une fois remplie, le libère pour son exécution comme planifié.

Par exemple, si vous disposez d'un flot de travaux contenant des travaux qui analysent un ou plusieurs fichiers, vous pouvez le faire démarrer une fois seulement qu'un ou plusieurs fichiers ont été modifiés ou créés. Il vous est également possible de conditionner le flot de travaux à la condition de sortie d'un travail donné : le flot de travaux ne démarre qu'une fois remplie la condition de sortie indiquée. Par exemple, si le travail contient des travaux qui traitent les données dans une base de données, vous pouvez juger souhaitable de démarrer le flot de travaux après qu'une nouvelle ligne a été écrite dans la base de données.

Vous pouvez démarrer le flux de travaux en vous basant sur l'une des conditions suivantes :

- la création d'un ou plusieurs fichiers,
- la modification d'un ou plusieurs fichiers,
- l'aboutissement d'un travail dont la condition de sortie est remplie.

Lorsque les conditions reposent sur la création ou la modification de fichiers, lorsque vous sauvegardez un flot de travaux, un travail de surveillance est créé automatiquement pour surveiller la condition spécifiée. Ce travail devient le premier travail du flot de travaux et reste à l'état EXEC jusqu'à ce que la condition qu'il surveille soit remplie ou que l'échéance du travail soit atteinte. Ce type de travail est identifié par le libellé +AUTOGEN+ sur la ligne de commande et par une icône comportant le caractère A dans Dynamic Workload Console.

Lorsque les conditions reposent sur le résultat d'un travail spécifié, lorsque vous sauvegardez le flot de travaux, le travail devient le premier travail du flot de travaux et redémarre jusqu'à ce que la condition soit satisfaite ou que l'échéance du travail soit atteinte. Cela s'applique également si le travail se termine à l'état Succès. Il s'agit du travail de surveillance. Lorsque vous spécifiez ce type de condition, IBM Workload Scheduler définit automatiquement une condition de sortie de réussite sur le travail de surveillance. Ce dernier se termine avec succès lorsqu'une de ses conditions de sortie est satisfaite, y compris celle sur le travail de surveillance lui-même. Vous pouvez appliquer cette logique au flot de travaux ou à des travaux spécifiques dans le flot de travaux. Pour plus d'informations sur les conditions de sortie, voir la section Application d'une logique de branchement conditionnelle du *Guide d'utilisation et de référence*.

Lorsque la condition est remplie, le travail se termine et libère la partie restante du flot de travaux. Lorsque le travail arrive à expiration alors que la condition n'est pas remplie, le flot de travaux est supprimé.

Dans le Concepteur de charge de travail, vous définissez l'automatisation conditionnelle de la charge de travail dans l'onglet **Condition de démarrage** lors de la création d'un flot de travaux. Vous pouvez sélectionner le type de condition à surveiller et spécifier les paramètres associés. Pour plus d'informations, voir

= «Scénario métier», à la page 138 ainsi que l'aide en ligne. A partir de la ligne de
= commande **composer**, vous pouvez définir la condition de démarrage dans la
= définition de flot de travaux. Pour plus d'informations, voir «Définition de flot de
= travaux», à la page 226.

= Par défaut, IBM Workload Scheduler continue à surveiller la condition même après
= que celle-ci a été remplie une première fois. Pour que cela soit possible, un travail
= Soumission d'un flot de travaux est créé automatiquement et ajouté au flot de
= travaux en tant que successeur du travail de surveillance. Ce type de travail est
= identifié par le libellé +AUTOGEN+ sur la ligne de commande et par une icône
= comportant le caractère A dans Dynamic Workload Console. Pour qu'IBM
= Workload Scheduler vérifie la condition une seule fois et s'arrête une fois celle-ci
= remplie, cochez la case **Démarrer une fois** dans le Concepteur de charge de travail,
= ou omettez le mot clé **rerun** sur la ligne de commande **composer**.

= Le travail Soumission d'un flot de travaux est défini sur un poste de travail de
= pool spécifique, MASTERAGENTS. Ce dernier contient l'agent dynamique installé
= sur le gestionnaire de domaine maître et sur le gestionnaire de domaine de
= sauvegarde, s'il est présent. L'agent dynamique installé sur le gestionnaire de
= domaine maître et le gestionnaire de domaine de sauvegarde (s'il est présent) sont
= automatiquement ajoutés lors de l'installation à ce poste de travail de pool. Si vous
= supprimez le poste de travail de pool MASTERAGENTS et que vous le recréez
= ensuite, vous devez arrêter et redémarrer l'agent dynamique pour l'ajouter à
= nouveau au poste de travail de pool MASTERAGENTS

= Le travail Soumission d'un flot de travaux crée une instance du flot de travaux
= dans lequel la condition de démarrage est définie. Par défaut, la nouvelle instance
= de flot de travaux démarre également si l'instance précédente est toujours en cours
= d'exécution et les deux instances sont exécutées simultanément. Pour modifier ce
= comportement, dans le Concepteur de charge de travail, passez à l'onglet **Options**
= **de planification** et sélectionnez **Mettre en file d'attente une nouvelle instance**
= dans la section **Actions**. A partir de la ligne de commande **composer**, utilisez le
= mot clé **onoverlap**. Pour plus d'informations, voir «onoverlap», à la page 274. La
= nouvelle instance générée est identique à la précédente et définie pour répéter la
= vérification de la condition. Une série de nouvelles instances est donc créée jusqu'à
= ce que l'échéance du flot de travaux soit atteinte.

= Le figure 23, à la page 137 illustre les détails du processus par défaut. Si vous
= cochez la case **Démarrer une fois**, le processus s'arrête une fois la condition
= remplie pour la première fois, et les étapes représentées sur le trait pointillé dans le
= graphique ne sont pas effectuées.

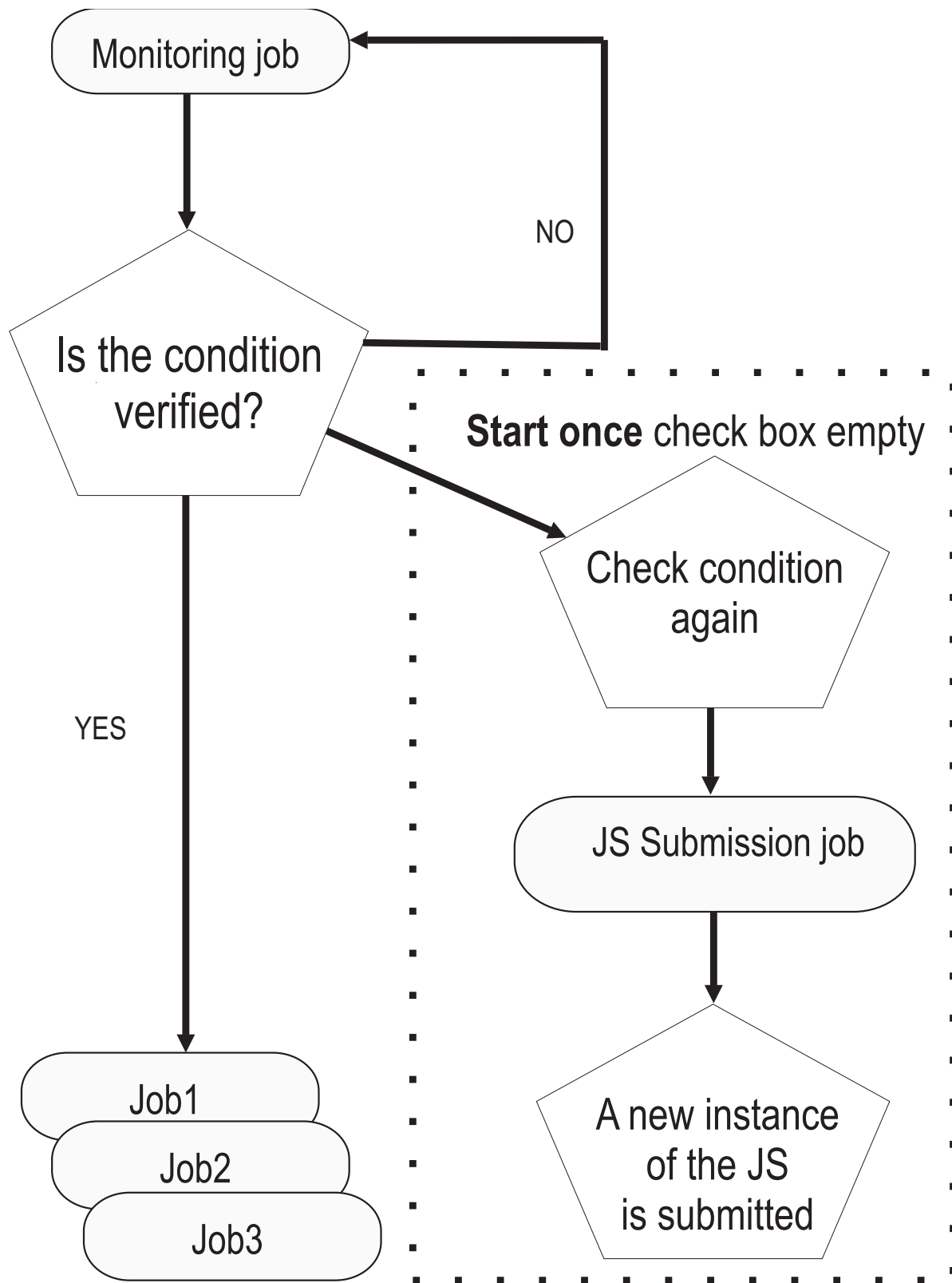


Figure 23. Automatisation conditionnelle de la charge de travail

Le travail de surveillance, le travail Soumission d'un flot de travaux et les instances supplémentaires du flot de travaux ne sont pas visibles dans la base de données, mais ils le sont dans le plan, ce qui vous permet de vérifier la progression du plan.

Dans Dynamic Workload Console, vous pouvez vérifier la progression du plan par le biais de la vue **Surveiller la charge de travail** et, sur la ligne de commande de **conman**, vous pouvez utiliser la commande **showjobs**.

Scénario métier

Ce scénario décrit l'un des nombreux moyens par le biais desquels l'automatisation conditionnelle de la charge de travail peut vous aider à améliorer l'automatisation de la charge de travail

IBM Workload Scheduler fournit plusieurs solutions efficaces destinées à vous aider à rendre l'automatisation plus efficace. Par exemple, par le biais de la fonction d'automatisation de flux de travaux itératifs, vous pouvez implémenter le traitement itératif d'une séquence de travaux faisant partie d'un flot de travaux.

Pour une société de vente au détail en ligne, cette fonction est très utile car elle itère le traitement de chaque article dans chaque commande client et chaque commande dans la base de données, en planifiant un seul flot de travaux.

Ce scénario permet également d'aller plus loin et d'adapter encore mieux l'automatisation à vos besoins. Par exemple, vous pouvez estimer nécessaire de traiter les commandes dès qu'elles sont insérées dans la base de données pour pouvoir répondre plus rapidement aux demandes des clients.

Vous pouvez maintenant créer le flot de travaux qui traite les commandes et ajouter à celui-ci une condition de démarrage afin qu'il vérifie qu'une commande est insérée dans la base de données avant de démarrer. Dès que la condition est remplie, le flot de travaux démarre.

Dans l'exemple suivant, le flot de travaux ORDERS commence à s'exécuter dès qu'une commande est créée dans la base de données.

Pour définir ce type de traitement de travaux, procédez comme suit :

1. Créez la définition du flot de travaux.
2. Définissez la condition de démarrage.

Pour créer le flot de travaux ORDERS, procédez comme suit :

1. Dans le Concepteur de charge de travail, créez la définition du flot de travaux ORDERS.
2. Dans l'onglet **Condition de démarrage**, sélectionnez **Condition de travail remplie**.
3. Dans la zone **Définition du travail**, sélectionnez le travail dont vous souhaitez surveiller la condition de sortie, c'est-à-dire le travail QUERY_DB_ORDERS qui recherche les commandes dans la base de données. Il s'agit du travail de surveillance.
4. Dans la zone **Poste de travail**, spécifiez le poste de travail sur lequel l'exécution du travail de surveillance est planifiée.
5. Dans la zone **Valeur de la condition de sortie**, spécifiez la condition de sortie qui, lorsqu'elle est remplie, entraîne le démarrage de l'exécution du flot de travaux. Dans le scénario en cours, la condition à remplir est que le nombre de lignes dans la base de données soit supérieur à zéro. Pour surveiller cette condition, spécifiez la chaîne suivante : `${this.NumberOfRows} > 0`. Par conséquent, le flot de travaux ORDERS démarre lorsque le travail de surveillance se termine avec la condition de sortie spécifiée.

- =
- =
- =
- =
- =
- =
6. Laissez la case à cocher **Démarrer une fois** désélectionnée pour continuer la surveillance de la condition une fois celle-ci remplie une première fois. Il s'agit du comportement par défaut.
 7. Renseignez les onglets et zones restants comme il convient et ajoutez les travaux appropriés.
 8. Sauvegardez le flot de travaux.

=

=

=

=

Le travail de surveillance lance la recherche des commandes dans la base de données de manière répétée. Dès qu'une commande est trouvée, le travail de surveillance libère la partie restante du flot de travaux et se termine ensuite avec succès.

=

=

=

=

=

=

=

=

Une nouvelle instance du flot de travaux est soumise à nouveau automatiquement, avec la même condition de démarrage, afin que le cycle itère et vérifie la commande suivante à insérer dans la base de données. Par défaut, les instances de flot de travaux s'exécutent en parallèle, en traitant la commande dès qu'elle est insérée dans la base de données, et de manière itérative pour chaque élément qu'elle contient. Si vous ne souhaitez pas que les instances soient exécutées en parallèle, sélectionnez **Mettre en file d'attente une nouvelle instance** dans l'onglet **Options de planification**. Lorsque cette option est sélectionnée, la nouvelle instance démarre uniquement lorsque l'instance précédente est terminée.

=

=

=

=

=

=

En combinant deux fonctions - l'automatisation de flux de travaux itératifs et l'automatisation conditionnelle de la charge de travail - vous pouvez mieux contrôler les conditions de démarrage de votre flux de travaux et traiter les commandes en temps réel. Pour plus d'informations sur les flux de travaux itératifs, voir la section relative à l'automatisation des flux de travaux itératifs dans le document *Présentation*.

=

=

Vous pouvez définir le même scénario en utilisant la ligne de commande Composer comme suit :

=

```
SCHEDULE NewYork#ORDERS
ON RUNCYCLE RC_DST "FREQ=DAILY;INTERVAL=1;BYWORKDAY"
STARTCOND JOB S_AGT#QUERY_DB_ORDERS OUTCOND "${this.NumberOfRows} > 0" INTERVAL 300
( RERUN )
:
Dallas#DBUpdate
:
Denver#OrderProcess
:
Lexington#WarehouseInfo
END
```

=

=

=

=

Pour plus d'informations sur la définition des conditions de démarrage à l'aide de la ligne de commande Composer, voir «Définition de flot de travaux», à la page 226.

Chapitre 8. Exécution de l'automatisation de la charge de travail gérée par événements

L'automatisation de la charge de travail gérée par événement ajoute la capacité d'exécuter l'automatisation de la charge de travail à la demande en complément de la planification de travaux basée sur un plan. Elle offre la possibilité de définir des règles pouvant déclencher l'automatisation à la demande de la charge de travail.

L'objet de l'automatisation de la charge de travail commandée par les événements dans IBM Workload Scheduler est de réaliser un ensemble d'actions prédéfinies en réponse aux événements se produisant sur les nœuds sur lesquels s'exécute IBM Workload Scheduler (ainsi que sur les systèmes non liés à IBM Workload Scheduler lors de l'utilisation de la ligne de commande sendevent). Ceci implique la capacité à soumettre la charge de travail et les commandes d'exécution à la volée, à avertir les utilisateurs par courrier électronique ou à envoyer des messages à Tivoli Enterprise Console.

Les tâches principales de l'automatisation de la charge de travail commandée par les événements sont les suivantes :

- Déclencher l'exécution de travaux et de flots de travaux par lots en fonction de la réception ou de la combinaison d'événements en temps réel.
- répondre à des invites ;
- Avertir les utilisateurs en cas d'anomalies dans l'environnement de planification IBM Workload Scheduler ou dans l'activité de planification par lots.
- Appeler un produit externe lorsqu'une condition d'événement particulière se produit.

L'automatisation de la charge de travail gérée par événement est basée sur le concept de la règle d'événement. Dans IBM Workload Scheduler, une règle d'événement est un objet de planification qui inclut les éléments suivants :

- Événements
- Conditions de corrélation des événements
- Actions

Lorsque vous définissez une règle d'événement, vous devez indiquer un ou plusieurs événements, une règle de corrélation et la ou les actions déclenchées par ces événements. En outre, vous pouvez spécifier des dates de validité, un intervalle horaire journalier d'activité, ainsi qu'un fuseau horaire commun à toutes les restrictions de temps définies.

Les événements que IBM Workload Scheduler peut détecter pour déclenchement d'actions peuvent être :

Événements internes

Il s'agit d'événements impliquant le statut de l'application interne IBM Workload Scheduler et les changements de statut des objets IBM Workload Scheduler. Les événements de cette catégorie peuvent être les changements de statut des travaux ou flots de travaux, le retard de travaux critiques ou de flots de travaux, ainsi que les changements de statut des postes de travail.

Événements externes

Il s'agit d'événements n'impliquant pas directement IBM Workload

Scheduler mais qui peuvent néanmoins affecter la soumission de la charge de travail. Les événements de cette catégorie peuvent être des messages inscrits dans les fichiers journaux, des événements envoyés par des applications tierces ou encore un fichier créé, mis à jour ou supprimé.

Il est possible d'établir une corrélation entre deux ou trois événements contenus dans une règle grâce à des attributs de corrélation tels qu'un poste de travail ou un travail commun. Les attributs de corrélation permettent de diriger la règle pour créer une instance de règle séparée (ou une copie de cette règle) pour chaque groupe d'événements partageant des caractéristiques communes. Généralement, chaque règle active a une seule instance qui s'exécute sur le serveur de traitement d'événements. Cependant, il arrive que la même règle soit requise pour différents groupes d'événements, souvent liés à différents groupes de ressources. Utiliser un ou plusieurs attributs de corrélation permet de diriger une règle afin de créer une copie de règle séparée pour chaque groupe d'événements ayant des caractéristiques communes.

Les actions que IBM Workload Scheduler peut exécuter lorsqu'il détecte l'un de ces événements peuvent être :

Actions opérationnelles

Il s'agit d'actions qui provoquent le changement de statut des objets de planification. Les actions de cette catégorie sont la soumission d'un travail, d'un flot de travaux, une commande ou la réponse à une invite.

Actions de notification

Il s'agit d'actions qui n'ont pas d'impact sur le statut des objets de planification. Les actions appartenant à cette catégorie sont l'envoi d'un courrier électronique, la journalisation d'un événement dans une base de données d'audit interne, la transmission d'un événement à Tivoli Enterprise Console, ou l'exécution d'une commande non IBM Workload Scheduler.

Cette classification des événements et des actions est conceptuelle. Elle n'a pas d'impact sur la façon dont ils sont gérés par le mécanisme commandé par les événements.

Scénarios de règles d'événement simples

Cette section répertorie certains scénarios simples impliquant l'utilisation de règles d'événement. Le codage XML correspondant est décrit dans la section «Exemples de règles d'événement», à la page 152.

Scénario 1 : Envoi d'une notification par courrier électronique

1. L'administrateur définit la règle d'événement suivante :

- Lorsqu'un travail job123 se termine par une erreur et renvoie le message d'erreur suivant :

AWSBHT001E Le travail "MYWORKSTATION#JOBS.JOB1234" dans le fichier "ls" a échoué suite à l'erreur : AWSBDW009E L'erreur système suivante est survenue en récupérant la structure de mot de passe pour l'accès utilisateur...

envoi d'un courrier électronique à l'opérateur john.smith@mycorp.com. Le sujet du courrier électronique inclut les noms de l'instance de travail et du poste de travail associé.

La règle d'événement est valide du 1er au 31 décembre dans la fenêtre de temps EST 12:00 à 16:00.

2. L'administrateur enregistre la règle en tant que version non provisoire dans la base de données et elle est immédiatement déployée par IBM Workload Scheduler.
3. Le planificateur commence à surveiller les travaux et à chaque fois que l'un d'eux se termine par une erreur, John Smith reçoit un courrier électronique lui permettant de vérifier le travail et de prendre les mesures appropriées.

Scénario 2 : Surveillance du rétablissement de la liaison d'un poste de travail

1. L'administrateur définit la règle d'événement suivante :
 - Si le poste de travail CPU1 perd sa liaison et si la liaison n'est pas rétablie dans les 10 minutes, envoi d'une notification par courrier électronique à `chuck.derry@mycorp.com`.
2. L'administrateur enregistre la règle en tant que version non provisoire dans la base de données et elle est immédiatement déployée par IBM Workload Scheduler.
3. Le planificateur commence à surveiller CPU1.
Si le statut du poste de travail passe à `unlinked`, IBM Workload Scheduler fait démarrer le délai d'attente de 10 minutes. Si l'événement d'établissement de liaison CPU1 `linked` n'est pas reçu dans les 10 minutes, le planificateur envoie le courrier électronique de notification à Chuck Derry.
4. Chuck Derry reçoit le courrier électronique, émet une requête pour connaître les actions/règles qui ont été déclenchées dans les 10 dernières minutes et à partir de là, va jusqu'à l'instance CPU1 et exécute une première identification de problème.

Scénario 3 : Soumission d'un flot de travaux après transmission FTP

1. L'administrateur définit la règle d'événement suivante :
 - Lorsque le fichier `daytransac*` est créé dans le répertoire `SFoperation` du poste de travail `system1`, et lorsque les modifications apportées au fichier ont été effectuées, soumet le flot de travaux `calmonthlyrev`.

La règle d'événement est valide toute l'année dans la fenêtre de temps EST 18:00 - 22:00.
2. L'administrateur enregistre la règle en tant que version non provisoire dans la base de données et elle est immédiatement déployée par IBM Workload Scheduler.
3. Le planificateur commence à surveiller le répertoire `SFoperation`. Dès que le fichier `daytransac*` est créé et n'est plus utilisé, il soumet le flot de travaux `calmonthlyrev`.
4. L'opérateur peut vérifier les journaux pour déterminer si la règle d'événement ou le flot de travaux ont été exécutés.

Scénario 4 : Lancer des travaux de longue durée en fonction de délai d'attente

1. L'administrateur définit la règle d'événement suivante :
 - Si les événements `job-x=exec` et `job-x=succ/abend` sont reçus dans les 5 minutes, le planificateur doit répondre `Yes` à `prompt-1` et démarrer le flot de travaux `jobstream-z`, sinon il doit envoyer un courrier électronique à `twsooper@mycompany.com` prévenant que le travail est en retard.

2. L'administrateur enregistre la règle d'événement au statut provisoire. Après quelques jours, il modifie la règle, change le destinataire du courrier électronique et l'enregistre en tant que version non provisoire. La règle est déployée.
3. A chaque fois que le statut de job-x passe à exec, IBM Workload Scheduler démarre le délai d'attente de 5 minutes.
Si l'état interne de job-x ne passe pas à succ ou abend dans les 5 minutes et si l'événement correspondant n'est pas reçu, IBM Workload Scheduler envoie le courrier électronique, sinon il répond Yes à l'invite et soumet jobstream-z.

Scénario 5 : Statuts des processus de surveillance et exécution d'un script de lots

L'administrateur crée une règle pour surveiller le statut du processus IBM Workload Scheduler et exécute un script de lots.

Scénario 6 : Intégration à SAP R/3 (en combinaison avec IBM Workload Scheduler)

1. L'administrateur définit la règle d'événement suivante :
 - Lorsqu'un événement appelé ID3965 est généré sur le serveur SAP R/3 Billing, IBM Workload Scheduler doit :
 - a. Exécuter la commande :


```
"/usr/apps/helpDesk -openTicket -text  
'Processing error $parameter  
on SAP system $wsname'"
```

pour ouvrir un ticket de service de maintenance
 - b. Envoyer un événement à Tivoli Enterprise Console.
2. L'administrateur enregistre la règle en tant que version non provisoire et elle est immédiatement déployée.
3. IBM Workload Scheduler commence à surveiller le statut des événements SAP R/3 activés sur le système Billing.
Lorsque l'événement ID3965 est détecté, IBM Workload Scheduler exécute la commande de service de maintenance spécifiée et envoie un événement à TEC.
4. Après un certain temps, l'administrateur ramène la règle d'événement au statut draft. La règle est automatiquement désactivée. Elle peut être à nouveau déployée lorsque nécessaire.

Scénario 7 : Surveillance du statut du fichier Symphony et consignation de l'occurrence d'un enregistrement endommagé

L'administrateur crée une règle permettant de surveiller le statut du fichier Symphony dans l'instance de IBM Workload Scheduler et consigne l'occurrence d'un enregistrement de dépendance Symphony endommagé dans la base de données d'audit interne.

Processus de gestion des règles d'événements

L'automatisation de la charge de travail gérée par événements est un processus continu et peut se résumer aux étapes ci-dessous :

1. Une définition de règles d'événement est créée ou modifiée avec Dynamic Workload Console ou avec le programme de ligne de commande composer et enregistrée dans la base de données d'objets. Les définitions de règle peuvent être enregistrées selon le statut draft ou non-draft.
2. Toutes les règles nouvelles et modifiées, autres qu'au format brouillon, qui sont enregistrées dans la base de données, sont périodiquement (par défaut toutes

les cinq minutes) détectées, construites et déployées par un processus interne nommé *rule builder*. A ce stade, ils sont activés. Dans l'intervalle, un serveur de traitement d'événements, qui est normalement situé dans le gestionnaire de domaine maître, reçoit tous les événements des agents et les traite.

3. Les configurations de surveillance mises à jour sont téléchargées vers les agents IBM Workload Scheduler et activées. Chaque agent IBM Workload Scheduler exécute un composant nommé *monman*, qui gère deux services intitulés *moteur de surveillance* et *ssmagent*, dont le rôle est d'intercepter les événements se produisant sur l'agent et de leur appliquer une action de filtrage provisoire.
4. Chaque instance de *monman* détecte et envoie ces événements au serveur de traitement d'événements.
5. Le serveur de traitement d'événements reçoit les événements et vérifie leur concordance avec toute règle d'événement déployée.
6. Si une règle d'événement concorde, le serveur de traitement d'événements appelle un auxiliaire d'actions pour effectuer les actions.
7. Celui-ci crée une instance de règles d'événement et journalise la sortie de l'action dans la base de données.
8. L'administrateur ou l'opérateur vérifie le statut des instances de règles d'événement et des actions dans les journaux et dans la base de données.

La fonction d'automatisation de la charge de travail gérée par événements est installée automatiquement avec le produit. Vous pouvez, à tout moment, changer la valeur de l'option globale *enEventDrivenWorkloadAutomation* si vous ne souhaitez pas ajouter cette fonctionnalité à votre réseau IBM Workload Scheduler.

L'automatisation de la charge de travail gérée par événements est basée sur un grand nombre de services, sous-systèmes et mécanismes internes. Les éléments suivants sont déterminants car ils peuvent être gérés :

monman

Installé sur chaque agent IBM Workload Scheduler où il vérifie tous les événements locaux. Tous les événements détectés sont transmis au serveur de traitement d'événements. Les commandes *conman* suivantes sont disponibles pour gérer *monman*:

Tableau 17. Commandes conman pour gérer les moteurs de surveillance

Commande	Action
<i>deployconf</i>	Met à jour le fichier de configuration pour le moteur de surveillance d'événements sur un agent. Cette commande est optionnelle, puisque la configuration est normalement déployée de façon automatique.
<i>showcpus getmon</i>	Renvoie la liste des règles d'événement définies pour le moniteur s'exécutant sur un agent. Cette commande peut être exécutée à distance pour extraire les informations du fichier de configuration situé sur un autre agent du réseau.
<i>startmon</i>	Démarre <i>monman</i> sur un agent. Cette commande peut être exécutée à partir d'un autre agent.
<i>stopmon</i>	Arrête <i>monman</i> sur un agent. Cette commande peut être exécutée à partir d'un autre agent.

monman démarre automatiquement à chaque fois qu'un nouveau fichier Symphony est activé. Ce comportement est déterminé par la définition de

l'option locale autostart monman sur la valeur yes par défaut. Vous pouvez désactiver cette fonction si vous ne souhaitez pas surveiller les événements d'un agent particulier.

À la suite de chaque cycle de déploiement de règles, les configurations de surveillance mises à jour sont automatiquement réparties vers les agents qui hébergent des règles ayant subi des modifications depuis le déploiement précédent. Il est à noter que certaines situations provisoires peuvent survenir durant le déploiement. Si une règle se trouve par exemple en attente de désactivation, il se peut que les agents envoient des événements dans la fraction d'intervalle durant laquelle les nouveaux fichiers de configuration n'ont pas encore été déployés, alors que le processeur d'événements a déjà ignoré ces événements.

Si un agent ne parvient pas à envoyer des événements à l'évent processing server durant une période de temps spécifiée, le statut de surveillance de l'agent est automatiquement désactivé. La période de temps peut être personnalisée (en secondes) avec le paramètre `edwa connection timeout` du fichier `localopts`. Par défaut, ce paramètre est défini à 300 secondes (5 minutes).

Les événements suivants peuvent être configurés dans le fichier `BMEvents.conf` afin d'envoyer les états de surveillance d'un agent :

- `TWS_Stop_Monitoring` (261) : envoyé lorsque l'état de surveillance d'un agent est désactivé (à cause d'une commande `stopmon` ou parce que l'agent ne parvient pas à envoyer d'événements à l'évent processing server).
- `TWS_Start_Monitoring` (262) : envoyé lorsque l'état de surveillance d'un agent est activé (grâce à la commande `startmon` ou parce que l'agent a recommencé à envoyer des événements à l'évent processing server).

Ces événements ont les zones positionnels suivantes :

1. Numéro d'événement
2. Poste de travail concerné
3. Réservé, actuellement toujours réglé sur 1

Serveur de traitement d'événements

Peut être installé sur le gestionnaire de domaine maître, le maître de sauvegarde ou tout agent tolérant aux pannes installé en tant que maître de sauvegarde. L'exécution a lieu sur le serveur d'applications. Le serveur peut être actif sur un seul noeud du réseau. Il crée les règles, génère les fichiers de configuration des agents et signale le téléchargement de nouvelles configurations aux agents. Il assure la réception et la corrélation des événements envoyés par les moteurs de surveillance et exécute les actions. Les commandes `conman` suivantes sont disponibles pour gérer le serveur de traitement d'événements :

Tableau 18. Commandes `conman` pour gérer le serveur de traitement d'événements

Commande	Action
<code>starteventprocessor</code>	Démarre le serveur de traitement d'événements.
<code>stopeventprocessor</code>	Arrête le serveur de traitement d'événements.
<code>switcheventprocessor</code>	Bascule le serveur de traitement d'événements du gestionnaire de domaine maître vers le maître de sauvegarde ou l'agent tolérant aux pannes installé comme maître de sauvegarde, ou inversement

Le serveur de traitement d'événements démarre automatiquement avec le gestionnaire de domaine maître. Un seul processeur d'événement peut s'exécuter à tout moment dans le réseau. Si vous souhaitez exécuter le processeur d'événement installé sur un poste de travail différent du maître (sur le maître de sauvegarde ou sur un agent tolérant aux pannes installé comme tel), vous devez en premier lieu utiliser la commande `switcheventprocessor` pour en faire un serveur de traitement d'événement actif.

Remarque : Si vous définissez le mot clé **ignore** sur la définition de poste de travail de l'agent (installé comme maître de sauvegarde) qui, à cet instant, héberge le processeur d'événement actif, la première occurrence de **JnextPlan** suivante reconnaît que cet agent particulier n'entre pas dans le cadre du plan. En conséquence, elle ne peut pas redémarrer le processeur d'événement hébergé. C'est la raison pour laquelle le planificateur renvoie un message d'avertissement et démarre le processeur d'événement hébergé par le gestionnaire de domaine maître.

Utilisation des interfaces et commandes impliquées

L'exécution et la gestion de l'automatisation de la charge de travail gérée par événements nécessite les tâches suivantes :

- Modifier les paramètres de configuration
- Modéliser les règles d'événement
- Déployer manuellement ou annuler le déploiement de règles d'événement
- Gérer les systèmes de surveillance et de traitement d'événements
- Surveiller et gérer les instances de règles d'événement

Vous devez être prêt à utiliser plusieurs interfaces et commandes IBM Workload Scheduler pour exécuter ces tâches. Le tableau 19, à la page 148 résume celles dont vous avez besoins :

Tableau 19. Interfaces et commandes pour gérer l'automatisation de charge de travail gérée par événements

Interface ou commande	Utilisée pour...
optman	changer les valeurs par défaut des options globales associées à la gestion d'événements. Les options globales sont utilisées pour configurer les éléments suivants : • Fréquence à laquelle les définitions de règles sont vérifiées pour détecter les mises à jour (deploymentFrequency). Les définitions modifiées sont déployées dans le domaine IBM Workload Scheduler • Numéro de port EIF sur lequel le serveur de traitement d'événements reçoit les événements (eventProcessorEIFPort, ou eventProcessorEIFSSLPort en cas de protection SSL). • Gestion des stratégies de nettoyage des données liées aux instances de règles, exécutions d'actions et consignations de messages (logCleanupFrequency). • Propriétés du serveur SMTP si vous déployez des règles mettant en oeuvre des actions qui envoient des courriers électroniques via un serveur SMTP (smtpServerName, smtpServerPort, smtpUseAuthentication, smtpUserName, smtpUserPassword, smtpUseSSL, smtpUseTLS). • Propriétés du serveur Tivoli Enterprise Console si vous déployez des règles mettant en oeuvre des actions qui transmettent des événements à la console TEC (TECServerName, TECServerPort). • Possibilité de désactiver le mécanisme de gestion des règles d'événement (enEventDrivenWorkloadAutomation), qui est installé par défaut avec le produit. Voir le <i>Guide d'administration</i> pour la liste des options globales.
composer	Exécuter la modélisation et les tâches de gestion des définitions de règles d'événement telles que add, create, delete, display, extract, list, lock, modify, new, print, unlock, validate. Les règles d'événement sont définies au format XML. Emettre des requêtes vers la base de données relationnelle IBM Workload Scheduler pour : • les définitions de règles d'événement filtrées par : – propriétés de règle, d'événement et d'action – travaux et flots de travaux impliqués par l'action associée à la règle • instances de règles d'événement, actions exécutées et enregistrement du journal des messages Voir «Définition de règle d'événement», à la page 291 pour la définition des règles d'événement. Voir Chapitre 10, «Gestion des objets dans la base de données - composer», à la page 321 pour les références des commandes.

Tableau 19. Interfaces et commandes pour gérer l'automatisation de charge de travail gérée par événements (suite)

Interface ou commande	Utilisée pour...
Dynamic Workload Console	Lancer une interface utilisateur graphique pour : • Modéliser et gérer les définitions de règles d'événement (browse, create, delete, modify, query, unlock) • Emettre des requêtes vers la base de données relationnelle IBM Workload Scheduler pour : - les définitions de règles d'événement filtrées par : - propriétés de règle, d'événement et d'action - travaux et flots de travaux impliqués par l'action associée à la règle - instances de règles d'événement, actions exécutées et enregistrement du journal des messages • Gérer le serveur de traitement d'événements et les moteurs de surveillance tels que décrit dans le tableau 17, à la page 145 et le tableau 18, à la page 146 Voir la documentation Dynamic Workload Console : Dynamic Workload Console - Guide d'utilisation, section "Création d'une règle d'événement". Dynamic Workload Console - Guide d'utilisation, section "Tâches de gestion des événements".
conman	Gérer les systèmes de surveillance, à savoir le serveur de traitement d'événements et les moteurs de surveillance tels que décrits dans le tableau 17, à la page 145 et le tableau 18, à la page 146. Voir Chapitre 12, «Gestion des objets dans le plan - conman», à la page 411 pour les références des commandes.
commandes d'utilitaire	Créer des définitions d'événements personnalisées et envoyer manuellement ces événements au serveur de traitement d'événements. Voir «evtdef», à la page 692 et «sendevent», à la page 719 pour plus de détails sur ces commandes.
planman	Déployer manuellement des règles nouvelles et modifiées. Pour plus d'informations, voir «Déploiement de règles», à la page 102.
fichier de sécurité	Définir des autorisations de sécurité pour gérer les règles d'événement, les événements, les actions et leurs instances. Voir <i>IBM Workload Scheduler - Guide d'administration</i> pour plus d'informations sur la configuration du fichier de sécurité IBM Workload Scheduler.

Important : Si vous utilisez un pare-feu de sécurité, assurez-vous que les ports définis dans l'option globale eventProcessorEIFPort et dans l'option locale nm port sur chaque agent sont ouverts pour les connexions entrantes et sortantes.

Définition de règles d'événement

Lorsque vous définissez une règle d'événement, vous spécifiez un ou plusieurs événements, une règle de corrélation et une ou plusieurs actions. Pour définir les règles d'événement, vous pouvez utiliser :

- La ligne de commande de composer
- Dynamic Workload Console
- Un ensemble d'interfaces API décrites dans un document séparé

Dans composer, vous modifiez les règles avec un éditeur XML de votre choix (option préférable mais facultative), car les définitions de règles d'événement sont créées avec la syntaxe XML.

La méthode d'utilisation de composer pour définir les règles d'événement est expliquée dans «Définition de règle d'événement», à la page 291, alors que la méthode d'utilisation de Dynamic Workload Console est décrite dans : Dynamic Workload Console - Guide d'utilisation, section Création d'une règle d'événement.

Les définitions de règles d'événement sont enregistrées dans la base de données IBM Workload Scheduler comme tous les autres objets de planification. Les options de sauvegarde disponibles sont les suivantes :

Brouillon

La règle est enregistrée dans la base de données, mais n'est pas prête à être déployée ou activée.

Cet état est déterminé par l'attribut `isDraft=yes`.

Version non provisoire

Cette règle est déployée ou est prête à être déployée dans l'environnement de planification.

Cet état est déterminé par l'attribut `isDraft=no`.

Les règles de version non provisoire sont prêtes à être activées. Le générateur de règles évalue le statut de chaque règle. L'état peut être :

- actif
- inactif
- mise à jour en attente
- erreur de mise à jour
- activation en attente
- erreur d'activation
- désactivation en attente
- erreur de désactivation

Périodiquement (toutes les cinq minutes ou conformément à une heure définie via l'option de configuration globale `deploymentFrequency`), le planificateur analyse la base de données pour détecter les règles de version non provisoire, puis crée les fichiers de configuration de règles pour le déploiement. Les nouvelles configurations de surveillance ne sont téléchargées sur les agents (chaque agent obtient son propre fichier de configuration contenant exclusivement les règles qu'il est censé exécuter) que si des modifications ont été apportées par rapport aux fichiers de configuration précédents.

La commande complémentaire `planman deploy` permet à tout moment de déployer manuellement les règles actives.

Le temps requis par le déploiement augmente de manière proportionnelle avec le nombre de règles actives à déployer. Si vous avez besoin de déployer

manuellement un grand nombre de règles nouvelles ou modifiées et que vous souhaitez réduire le temps de déploiement, exécutez `planman deploy -scratch`.

Vous pouvez déployer ou annuler le déploiement de règles selon vos besoins en définissant l'attribut `isDraft` sur `no` ou sur `yes` avec `composer` ou avec Dynamic Workload Console.

En fonction de leurs caractéristiques, les règles sont classifiées comme suit :

filter La règle est activée sur détection d'un événement spécifique unique.

sequence

La règle est activée lorsqu'un groupe ordonné d'événements est détecté ou intervient dans un intervalle de temps spécifique.

set La règle est activée lorsqu'un groupe non ordonné d'événements est détecté ou intervient dans un intervalle de temps spécifique.

Les règles de filtrage reposent sur la détection d'un événement unique tel que le retard d'un travail, l'arrêt d'un poste de travail IBM Workload Scheduler, la modification d'un fichier, l'achèvement avec succès d'un flot de travaux, etc.

Les règles 'Set' et 'Sequence' reposent sur la détection d'un plus grand nombre d'événements. Ces règles peuvent également, en variante, dépendre d'une condition de dépassement de délai d'attente. Le dépassement du délai d'attente d'une règle a lieu lorsque le ou les premiers événements d'une règle Sequence, ou une partie des événements d'une règle Set sont reçus, mais que tous les événements ne sont pas reçus dans un intervalle de temps spécifié, lequel est calculé à partir de la réception du dernier événement.

Les définitions de règles peuvent inclure des attributs qui spécifient une période de validité et une fenêtre de temps d'activité pour chaque jour de validité. Si vous ne spécifiez pas ces attributs, la règle est active de façon perpétuelle à tout moment une fois déployée et jusqu'à ce qu'elle soit ramenée au statut provisoire ou supprimée de la base de données.

Vous pouvez utiliser la substitution de variable. Lorsque vous définissez des paramètres d'action, vous pouvez utiliser les attributs d'occurrences d'événements qui déclenchent la règle d'événement selon l'une des deux formes suivantes :

- `${event.property}`

Remplace la valeur telle quelle. Cette option est utile pour transmettre les informations à une action liée à celles-ci par voie de programme, par exemple l'option `schedTime` d'un flot de travaux.

- `%{événement.propriété}`

Remplace la valeur formatée en syntaxe lisible. Cette option est utile pour transmettre les informations à une action chargée de faire suivre les informations à un utilisateur, par exemple, pour formater le `schedTime` d'un flot de travaux dans le corps d'un courrier électronique.

où :

event est le nom défini pour le déclenchement `eventCondition`.

property

est le nom de `attributeFilter` dans le prédicat de filtrage de la condition d'événements déclenchante. La valeur prise par le filtre d'attribut lorsque la règle est déclenchée est remplacée en tant que valeur de paramètre dans la définition d'action avant l'exécution.

Il est à noter que vous pouvez également recourir à une substitution de variable si aucune valeur de attributeFilter n'a été spécifiée pour un attribut, ainsi que lorsque l'attribut est accessible en lecture seule.

Vous pouvez voir l'utilisation de la substitution de variables dans certains des exemples de définitions suivants où les valeurs de filtre d'attributs sont remplacées dans l'objet et le corps d'un courrier électronique.

Exemples de règles d'événement

Vous trouverez ci-dessous des exemples de définitions de règles d'événement qui s'appliquent aux scénarios décrits dans «Scénarios de règles d'événement simples», à la page 142.

Définition de règles d'événement pour le scénario 1

Lorsqu'un travail job123 se termine par une erreur et renvoie le message d'erreur suivant :

```
AWSBHT001E The job "MYWORKSTATION#JOBS.JOB1234" in file "ls" has failed with
the error: AWSBDW009E The following operating system error occurred retrieving
the password structure for either the logon user...
```

envoi d'un courrier électronique à l'opérateur john.smith@mycorp.com. Le sujet du courrier électronique inclut les noms de l'instance de travail et du poste de travail associé.

La règle d'événement est valide du 1er au 31 décembre dans la fenêtre de temps EST 12:00 à 16:00.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="scenario1_rule" ruleType="filter" isDraft="no">
    <description>This is the definition for scenario1</description>
    <timeZone>America/Indianapolis</timeZone>
    <validity from="2015-12-01" to="2015-12-31" />
    <activeTime start="12:00:00" end="16:00:00" />
    <eventCondition name="event1"
      eventProvider="TWSObjectsMonitor"
      eventType="JobStatusChanged">
      <filteringPredicate>
        <attributeFilter name="JobStreamWorkstation" operator="eq">
          <value>*</value>
        </attributeFilter>
        <attributeFilter name="JobStreamName" operator="eq">
          <value>*</value>
        </attributeFilter>
        <attributeFilter name="JobName" operator="eq">
          <value>job123*</value>
        </attributeFilter>
        <attributeFilter name="Status" operator="eq">
          <value>Error</value>
        </attributeFilter>
        <attributeFilter name="ErrorMessage" operator="eq">
          <value>*AWSBDW009E*</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="MailSender"
      actionType="SendMail"
      responseType="onDetection">
```

```

        <description>Send email to John Smith including names of job
            and associated workstation</description>
        <parameter name="To">
            <value>john.smith@mycorp.com</value>
        </parameter>
        <parameter name="Subject">
            <value>Job %{event1.JobName} on agent %{event1.Workstation}
                ended in error</value>
        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

Important : Le message d'erreur expliquant pourquoi un travail s'est terminé par une erreur peut figurer dans le fichier journal TWSMERGE. Dans ce scénario, le fichier journal TWSMERGE contient l'instruction suivante :

```

BATCHMAN:+
BATCHMAN:+ AWSBHT001E The job "MYWORKSTATION#JOBS.JOB1234" in file "ls"
has failed with the error: AWSBDW009E The following operating system
error occurred retrieving the password structure for either the logon
user, or the user who owns a file or external dependency
BATCHMAN:+

```

où le message d'erreur est constitué de tout ce qui suit la chaîne :
has failed with the error:

Définition de règles d'événement pour le scénario 2

Si le poste de travail CPU1 perd sa liaison et si la liaison n'est pas rétablie dans 1 heure, envoi d'une notification par courrier électronique à chuck.derry@mycorp.com.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
    xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
        event-management/rules http://www.ibm.com/xmlns/prod/tws/1.0/
        event-management/rules/EventRules.xsd">
    <eventRule name="SCENARIO02_RULE" ruleType="sequence" isDraft="no">
        <description>This is the definition for scenario2</description>
        <timeZone>America/Anchorage</timeZone>
        <timeInterval amount="10" unit="minutes" />
        <eventCondition name="WSevent"
            eventProvider="TWSObjectsMonitor"
            eventType="ChildWorkstationLinkChanged">
            <scope>
                * TO CPU1
            </scope>
            <filteringPredicate>
                <attributeFilter name="Workstation" operator="eq">
                    <value>CPU1</value>
                </attributeFilter>
                <attributeFilter name="LinkStatus" operator="eq">
                    <value>Unlinked</value>
                </attributeFilter>
            </filteringPredicate>
        </eventCondition>
        <eventCondition name="childWksLnkChgEvt1"
            eventProvider="TWSObjectsMonitor"
            eventType="ChildWorkstationLinkChanged">
            <scope>
                * TO CPU1
            </scope>
            <filteringPredicate>
                <attributeFilter name="Workstation" operator="eq">
                    <value>CPU1</value>

```

```

        </attributeFilter>
        <attributeFilter name="LinkStatus" operator="eq">
            <value>Linked</value>
        </attributeFilter>
    </filteringPredicate>
</eventCondition>
<action actionProvider="MailSender" actionType="SendMail"
        responseType="onTimeOut">
    <scope>
        CHUCK.DERRY@MYCORP.COM : AGENT CPU1 HAS BEEN UNLINKED FOR AT LEAST 10 MINUTES
    </scope>
    <parameter name="Subject">
        <value>Agent CPU1 has been unlinked for at least 10 minutes</value>
    </parameter>
    <parameter name="To">
        <value>chuck.derry@mycorp.com</value>
    </parameter>
    <parameter name="Body">
        <value>The cause seems to be: %{WSevent.UnlinkReason}</value>
    </parameter>
</action>
</eventRule>
</eventRuleSet>

```

Définition de règles d'événement pour le scénario 3

Lorsque le fichier daytransac est créé dans le répertoire SFOperation du poste de travail system1, et lorsque les modifications apportées au fichier ont été effectuées, soumet le flot de travaux calmonthlyrev.

La règle d'événement est valide toute l'année dans la fenêtre de temps EST 18:00 - 22:00.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
    xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
        event-management/rules/EventRules.xsd">
    <eventRule name="scenario3_rule"
        ruleType="filter"
        isDraft="no">
        <description>This is the definition for scenario3</description>
        <timeZone>America/Louisville</timeZone>
        <validity from="2015-01-01" to="2015-12-31" />
        <activeTime start="18:00:00" end="22:00:00" />
        <eventCondition eventProvider="FileMonitor"
            eventType="ModificationCompleted">
            <filteringPredicate>
                <attributeFilter name="FileName" operator="eq">
                    <value>daytransac</value>
                </attributeFilter>
                <attributeFilter name="Workstation" operator="eq">
                    <value>EVIAN1</value>
                </attributeFilter>
            </filteringPredicate>
        </eventCondition>
        <action actionProvider="TWSAction"
            actionType="sbs"
            responseType="onDetection">
            <description>Submit the calmonthlyrev job stream.</description>
            <parameter name="JobStreamName">
                <value>calmonthlyrev</value>
            </parameter>
            <parameter name="JobStreamWorkstationName">
                <value>act5cpu</value>
            </parameter>
        </action>
    </eventRule>
</eventRuleSet>

```



```

        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

Définition de règles d'événement pour le scénario 4

Lorsque les événements job-x=exec et job-x=succ/abend sont reçues dans les 500 secondes, le planificateur doit répondre Yes à prompt-1 et démarrer le flot de travaux jobstream-z, sinon il doit envoyer un courrier électronique à twsoper@mycompany.com prévenant que le travail est en retard.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="scenario4_rule"
    ruleType="sequence"
    isDraft="yes">
    <description>This is the definition for scenario4</description>
    <timeZone>America/Buenos_Aires</timeZone>
    <timeInterval amount="500" unit="seconds" />
    <eventCondition eventProvider="TWSObjectsMonitor"
      eventType="JobStatusChanged">
      <filteringPredicate>
        <attributeFilter name="JobName" operator="eq">
          <value>job-x</value>
        </attributeFilter>
        <attributeFilter name="InternalStatus" operator="eq">
          <value>EXEC</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <eventCondition eventProvider="TWSObjectsMonitor"
      eventType="JobStatusChanged">
      <filteringPredicate>
        <attributeFilter name="JobName" operator="eq">
          <value>job-x</value>
        </attributeFilter>
        <attributeFilter name="InternalStatus" operator="eq">
          <value>ABEND</value>
          <value>SUCC</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="MailSender"
      actionType="SendMail"
      responseType="onTimeOut">
      <description>Send email to operator saying that job-x
        is late</description>
      <parameter name="To">
        <value>twsoper@mycorp.com</value>
      </parameter>
      <parameter name="Subject">
        <value>Job-x is late by at least 5 minutes</value>
      </parameter>
    </action>
    <action actionProvider="TWSAction"
      actionType="Reply"
      responseType="onDetection">
      <description>Reply Yes to prompt-1</description>
      <parameter name="PromptName">
        <value>prompt-1</value>
      </parameter>
      <parameter name="PromptAnswer">

```

```

        <value>Yes</value>
    </parameter>
</action>
<action actionProvider="TWSAction"
        actionType="sbs"
        responseType="onDetection">
    <description>Submit jobstream-z</description>
    <parameter name="JobStreamName">
        <value>jobstream-z</value>
    </parameter>
    <parameter name="JobStreamWorkstationName">
        <value>act23cpu</value>
    </parameter>
</action>
</eventRule>
</eventRuleSet>

```

Définition de règles d'événement pour le scénario #5

Contrôlez l'état des processus IBM Workload Scheduler répertoriés dans ProcessName et exécutez le script par lots RUNCMD.FM.BAT situé dans E:\production\eventRules.

Le mot clé TWSPATH indique le chemin qualifié complet où l'instance IBM Workload Scheduler contrôlée est installée, incluant le suffixe /TWS.

Sur les systèmes d'exploitation Windows, la règle d'événement est déclenchée chaque fois que l'agent est arrêté à l'aide de la commande ShutDownLwa et manuellement. Sur les systèmes d'exploitation UNIX, la règle d'événement est déclenchée lorsque vous arrêtez le processus manuellement, mais pas lorsque la commande ShutDownLwa est utilisée.

Si vous spécifiez ProcessName=agent, le composant agent est contrôlé, tandis que le processus TWS JobManager n'est pas contrôlé.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
    xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
        event-management/rules/EventRules.xsd">
    <eventRule name="scenario5rule" ruleType="filter" isDraft="no">
        <eventCondition name="twsProcMonEvt1"
            eventProvider="TWSApplicationMonitor"
            eventType="TWSProcessMonitor">
            <scope>
                AGENT, NETMAN DOWN ON WIN86MAS
            </scope>
            <filteringPredicate>
                <attributeFilter name="ProcessName" operator="eq">
                    <value>agent</value>
                    <value>appservman</value>
                    <value>batchman</value>
                    <value>jobman</value>
                    <value>mailman</value>
                    <value>monman</value>
                    <value>netman</value>
                </attributeFilter>
                <attributeFilter name="TWSPATH" operator="eq">
                    <value>E:\Program Files\IBM\TWA\TWS</value>
                </attributeFilter>
                <attributeFilter name="Workstation" operator="eq">
                    <value>win86mas</value>
                </attributeFilter>
                <attributeFilter name="SampleInterval" operator="eq">

```

```

        <value>5</value>
      </attributeFilter>

    </filteringPredicate>
  </eventCondition>
  <action actionProvider="GenericActionPlugin" actionType="RunCommand"
responseType="onDetection">
    <scope>
      RUNCMD.FM.BAT
    </scope>
    <parameter name="Command">
      <value>runCmdFM.bat</value>
    </parameter>
    <parameter name="WorkingDir">
      <value>E:\production\eventRules</value>
    </parameter>
  </action>
</eventRule>
</eventRuleSet>

```

Définition de règles d'événement pour le scénario #6

Lorsqu'un événement spécifique nommé ID3965 est généré, une commande est exécutée pour ouvrir un ticket de service et un événement est envoyé à Tivoli Enterprise Console.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
    rules http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
    rules/EventRules.xsd">
  <eventRule name="SCENARIO6RULE" ruleType="filter" isDraft="no">
    <eventCondition name="R3EventRaised1" eventProvider="SapMonitor"
      eventType="R3EventGenerated">
      <scope>
        ID3965 ON SAP_WKS
      </scope>
      <filteringPredicate>
        <attributeFilter name="Workstation" operator="eq">
          <value>SAP_WKS</value>
        </attributeFilter>
        <attributeFilter name="SAPEventId" operator="eq">
          <value>ID3965</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="TWSAction" actionType="sbd" responseType="onDetection">
      <scope>
        SBD "/USR/APPS/HELPDESK -OPENTICKET -TEXT 'PROCESSING ERROR
          %{R3EVENTRAISED1.SAPEVENTID} ON SAP SYSTEM %{R3EVE
      </scope>
      <parameter name="JobType">
        <value>Script</value>
      </parameter>
      <parameter name="JobTask">
        <value>"/usr/apps/helpDesk -openTicket -text 'Processing error
          %{R3EventRaised1.SAPEventId} on SAP system
          %{R3EventRaised1.Workstation}&apos;"</value>
      </parameter>
      <parameter name="JobLogin">
        <value>userLogin</value>
      </parameter>
      <parameter name="JobWorkstationName">
        <value>AGENT1</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>

```

```

        <parameter name="JobUseUniqueAlias">
            <value>true</value>
        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

Définition de règle d'événement pour le scénario #7

Surveiller l'état du fichier Symphony du poste de travail *IT041866-9088* et consigner, dans la base de données d'audit, l'occurrence d'un enregistrement Symphony endommagé.

Sur chaque poste de travail, le processus batchman surveille le fichier Symphony. Lorsqu'il détecte un enregistrement endommagé, il envoie l'événement d'endommagement dans la file d'attente de messages du processus monman, puis au processeur d'événement sur le poste de travail maître.

La règle d'événement est déclenchée chaque fois que le processus batchman trouve un enregistrement de dépendance endommagé sur le poste de travail spécifié dans la définition de règle d'événement.

Si vous définissez la valeur du poste de travail sur *IT041866-9088*, le fichier Symphony de ce poste de travail est surveillé et la règle d'événement est déclenchée lorsque le processus batchman détecte un enregistrement endommagé dans le fichier Symphony.

L'occurrence de l'enregistrement endommagé est écrite dans le fichier d'audit de consignation de message.

```

?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules/EventRules.xsd">
<eventRule name="TEST1" ruleType="filter" isDraft="no">
<eventCondition name="productAlertEvt1" eventProvider="TWSObjectsMonitor"
eventType="ProductAlert">
<scope>
IT041866-9088
</scope>
<filteringPredicate>
<attributeFilter name="Workstation" operator="eq">
<value>IT041866-9088</value>
</attributeFilter>

</filteringPredicate>
</eventCondition>
<action actionProvider="MessageLogger" actionType="MSGLOG"
responseType="onDetection">
<scope>
OBJECT=%{PRODUCTALERTEVT1.WORKSTATION} MESSAGE=%{PRODUCTALERTEVT1.WORKSTATION}
corruption
</scope>
<parameter name="ObjectKey">
<value>%{productAlertEvt1.Workstation}</value>
</parameter>
<parameter name="Message">
<value>%{productAlertEvt1.Workstation} corruption</value>
</parameter>
<parameter name="Severity">
<value>Info</value>

```

```
</parameter>  
</action>  
</eventRule>  
</eventRuleSet>
```

Remarques sur le fonctionnement des règles

Le paragraphe suivant contient des informations essentielles sur le comportement des règles d'événement, qui peuvent vous aider à mieux comprendre l'origine de certains résultats inattendus :

Notes sur les règles d'état

- En fonction de ses dates de validité from et to, l'état d'une règle change à mesure du déploiement :
 - Si vous créez une règle dont les dates de validité from et to ont déjà expiré, la règle est sauvegardée à l'état activation en instance. Si la règle est déployée, elle reste à l'état activation en instance.
 - Si vous attribuez une date ultérieure à la zone de validité to, la règle est déployée à l'état actif. Si vous lui attribuez une date antérieure, la règle est déployée à l'état inactif.
- Les périodes d'activité de la règle (start et end) n'affectent pas son état. Tant que la règle se trouve dans les bonnes dates de validité, elle reste à l'état actif, qu'elle soit ou non dans sa période d'activité définie. Si le planificateur reçoit des événements définis par une règle hors de sa période d'activité, les événements sont annulés mais la règle reste à l'état actif.

Chemins d'accès des règles d'événement

Uniquement sur les systèmes d'exploitation Windows, lorsque vous définissez un chemin d'accès pour une règle d'événement, vous ne pouvez pas utiliser de barre oblique /, mais uniquement une barre oblique inverse simple \ ou double \\.

Règles d'événement contraintes par des fichiers

Lorsqu'une règle d'événement de surveillance de fichier est contrainte par la création ou l'existence d'un fichier, si ce fichier est supprimé puis recréé dans le même intervalle de temps entre deux vérifications consécutives, ce fichier est inchangé pour IBM Workload Scheduler et la règle d'événement n'est pas déclenchée. L'intervalle de temps entre les vérifications peut être réduit à un minimum de 5 secondes, mais cela implique une dégradation significative des performances. Pour éviter ce problème, n'effectuez pas d'actions sur les fichiers au cours de cet intervalle.

Règles d'événement de surveillance de fichier pour des fichiers sur des systèmes

NFS Pour activer une règle d'événement de surveillance de fichier pour des fichiers sur des systèmes NFS, vous devez activer la propriété MonitorRemoteFS dans le fichier `<rép_install_tws>/ssm/Config/init.cfg` (pour les agents tolérants aux pannes), ou dans le fichier `<rép_install_tws>/EDWA/ssm/Config/init.cfg` (pour les agents légers), où `<rép_install_tws>` est le chemin du répertoire où l'agent IBM Workload Scheduler est installé. Pour activer cette propriété, procédez comme suit :

1. Editez le fichier `init.cfg`.
2. Remplacez `MonitorRemoteFS=off` par `MonitorRemoteFS=on`.
3. Arrêtez l'agent System Service Monitor (SSM).
4. Démarrez l'agent SSM.

Remarque : Sur les systèmes d'exploitation Windows, l'adresse du poste de travail distant doit être incluse au début du chemin d'accès complet du fichier à surveiller. Par exemple : \\123.123.123.123\images\Automation\myFile.txt. Cela n'est pas nécessaire sur les systèmes d'exploitation UNIX.

Manque de persistance du statut qualifiant l'instance de règle d'événement

Tout arrêt ou défaillance du processeur d'événements entraîne la perte du statut des instances de règles pour lesquelles la correspondance est incomplète.

Répétition des instances de règles set et sequence

De façon générale, chaque règle active a une, et seulement une copie qui s'exécute sur le serveur de traitement d'événements. Les règles Set et sequence utilisent le mécanisme expliqué dans l'exemple suivant :

1. Vous définissez une règle sequence avec deux événements, A et B.
2. Lorsque le premier événement qui correspond à la séquence se produit (événement A), il active la règle et attend le second événement (événement B).

Une fois la règle active, tout événement A additionnel pouvant survenir est ignoré. Aucune instance supplémentaire n'est créée pour aucun événement A nouvellement détecté tant que la règle attend l'événement B.

3. Lorsque l'événement B se produit, la règle est terminée et se réinitialise, attendant que l'événement A se produise à nouveau.

Le mécanisme des règles set et sequence est tel que toute occurrence additionnelle d'un événement déjà détecté est ignorée et rejetée si l'autre événement défini n'est pas arrivé.

Vous pouvez éviter ce problème en utilisant des attributs de corrélation. L'utilisation d'un ou plusieurs attributs de corrélation permet de diriger une règle de manière à créer une copie de règle séparée à la réception d'un événement A.

Les types de règle Set et sequence avec des fenêtres de temps d'activité inférieures à 24 heures

Les occurrences des règles set ou sequence qui ont été définies pour être actives pendant uniquement quelques heures chaque jour, ne sont pas purgées lorsque la période d'activité à l'intérieur de chaque jour expire si uniquement une partie des événements se produit. Elles restent en état de veille, attendant de recevoir les événements restants dans les jours qui suivent.

Par exemple, vous définissez une règle set qui inclut deux événements. La règle est valide du 1er janvier au 10 janvier et est active quotidiennement de 6 heures à 10 heures du matin.

Si le 1er janvier, le premier événement est reçu à 8 heures du matin, la règle attend le second événement et reste *active* au-delà de 10 heures du matin si l'événement n'est pas détecté.

Si le second événement se produit à 11 heures du matin (ce qui est en dehors de la fenêtre de temps d'activité), il est rejeté mais la règle reste *active*. Si le second événement est reçu à nouveau à 7 heures du matin le 2 janvier, la règle est déclenchée et les actions sont mises en oeuvre.

Si vous ne souhaitez pas que la règle soit *reportée* au jour suivant, vous devez la purger.

Éléments de règles déclenchés

Chaque fois que les conditions d'événements répertoriées dans une règle d'événement déployée sont remplies ou qu'un dépassement de délai a lieu, une instance de règle d'événement est créée. Une instance de règles d'événement inclut des informations telles que le nom de la règle d'événement, la date et l'heure à laquelle la concordance a eu lieu et la liste des instances d'actions, ainsi que leur statut, dont l'exécution a eu lieu à la suite d'un correspondance avec les conditions d'événement. L'objet `RuleInstance` est utilisé pour rassembler les informations sur les règles déclenchées dans un journal d'historique des instances de règles.

Les actions effectuées par une règle déclenchée sont collectées dans un journal d'historique des exécutions d'action. Les informations fournies incluent les exécutions d'action qui se sont terminées par une erreur ou par un avertissement par opposition à celles qui se sont terminées sans problème. Si au moins une action s'achève en générant une erreur, l'instance totale de la règle est consignée comme une erreur. Dans les informations suivies au niveau des exécutions d'action, les zones de règle sont également conservés et des requêtes peuvent être exécutées pour rechercher des exécutions d'action en fonction de ces zones (l'identificateur d'instance de règle est également disponible).

Définition d'événements personnalisés

Outre les types et les classes d'événements déjà définis (également appelés "fournisseurs") et répertoriés en détail à la section «Fournisseurs d'événements et définitions», à la page 873, IBM Workload Scheduler fournit un modèle générique de création d'événements intitulé `GenericEventPlugIn`, modifiable par les programmeurs ayant des compétences spécifiques dans certaines applications et en langage XML, afin de définir des types d'événements personnalisés utiles à l'organisation.

Les outils servant à définir des types d'événements personnalisés sont les suivants :

- Le fournisseur d'événements `GenericEventPlugIn` en langage XML
- La commande d'utilitaire `evtdef`, avec laquelle un programmeur peut télécharger le fournisseur d'événements `GenericEventPlugIn` sous forme de fichier local afin de définir des événements personnalisés
- Les fichiers de définition de schéma XML (fichiers XSD) nécessaires à la validation du fournisseur d'événements génériques modifiés. Ces fichiers contiennent également des directives en ligne pour faciliter le travail des programmeurs.
- La commande d'utilitaire `sendevent`, qui permet d'envoyer les événements personnalisés au serveur de traitement d'événements afin de déclencher des règles à partir de n'importe quel poste de travail simplement via le client de ligne de commande à distance de IBM Workload Scheduler.

La procédure de définition et d'utilisation des événements personnalisés est la suivante :

1. La commande `evtdef` permet aux programmeurs d'effectuer les tâches suivantes :
 - a. Télécharger le fournisseur d'événements générique sous forme de fichier local.
 - b. Suivre les définitions de schéma afin d'ajouter des types d'événements personnalisés et définir leurs propriétés et attributs dans le fichier via un éditeur XML.

- c. Charger le fichier local en tant que fournisseur d'événements génériques modifié contenant les nouvelles définitions de types d'événements personnalisés. Le fournisseur d'événements génériques modifié est sauvegardé sous forme de fichier XML sur le gestionnaire de domaine maître.
2. Le générateur de règles ou l'administrateur définit, via le composeur ou Dynamic Workload Console, les règles d'événement devant être déclenchées par ces événements personnalisés, en spécifiant :
 - Le fournisseur d'événements génériques en tant que fournisseur d'événements
 - Les types d'événements personnalisés en tant que types d'événements
 - Les propriétés (ou attributs) des types d'événements personnalisés défini(e)s pour les événements personnalisés dans le fournisseur d'événements génériques, avec les valeurs particulières qui déclencheront les règles.
3. Déployer les règles.
4. Lorsque l'occurrence d'un événement personnalisé survient, cet événement peut être envoyé vers le serveur de traitement d'événements selon l'une des méthodes suivantes :
 - Via la commande sendevent exécutée à partir d'un script ou d'une ligne de commande
 - Via une autre application, telle que Tivoli Enterprise Console ou Tivoli Monitoring

Dès la réception de l'événement par le serveur de traitement d'événements, le déclenchement de la règle a lieu.

Chapitre 9. Définition d'objets dans la base de données

IBM Workload Scheduler exploite un ensemble de définitions d'objets qui servent à mapper votre environnement de planification dans la base de données. Les objets de planification sont :

- les agendas
- les domaines
- les règles d'événement
- les travaux
- les flots de travaux
- les invites
- les paramètres
- les ressources
- groupes de cycle d'exécution
- tables de variables
- applications de charge de travail
- postes de travail
- classes de postes de travail
- utilisateurs

Un ensemble spécial d'objets de planification est composé d'objets de sécurité. Les objets de sécurité mappent la sécurité basée sur les rôles dans votre environnement de planification. Les objets de sécurité sont :

- les listes de contrôle d'accès
- les domaines de sécurité
- les rôles de sécurité

Définition des objets de planification

Les objets de planification sont gérés par le programme de ligne de commande **composer** et stockés dans la base de données IBM Workload Scheduler. Le programme de ligne de commande **composer** peut être installé et utilisé sur toute machine connectée via TCP/IP au système sur lequel le gestionnaire de domaine maître est installé. Il ne nécessite pas l'installation d'un poste de travail IBM Workload Scheduler comme condition préalable. Il communique via HTTP/HTTPS avec le gestionnaire de domaine maître sur lequel la base de données DB2 est installée. La configuration de la communication HTTP/HTTPS et la vérification d'authentification sont gérées par l'infrastructure WebSphere Application Server. Le programme **composer** utilise des fichiers d'édition pour mettre à jour la base de données de planification. Le format du fichier d'édition utilisé pour définir un objet spécifique est décrit plus loin dans le présent chapitre. Par exemple, pour créer un objet, vous devez d'abord entrer sa définition dans un fichier d'édition, puis utiliser **composer** pour l'ajouter à la base de données en indiquant le fichier d'édition qui contient la définition. Le programme de ligne de commande **composer** vérifie la syntaxe dans le fichier d'édition, et si elle est correcte, il convertit la définition de l'objet en langage XML, puis envoie la demande au gestionnaire de domaine maître via HTTP/HTTPS.

Sur le gestionnaire de domaine maître, la définition XML est analysée, des contrôles de sémantique et d'intégrité sont effectués, puis la mise à jour est stockée dans la base de données.

Dans cette version du produit, toutes les entrées sont gérées individuellement. Le mécanisme de verrouillage des objets est une autre fonction disponible dans cette nouvelle version. Les objets de planification définis dans la base de données sont verrouillés lorsqu'un utilisateur les utilise afin d'empêcher les accès simultanés. Cela signifie que seul l'utilisateur qui verrouille l'objet possède un droit d'accès en écriture pour cet objet et que les autres utilisateurs ne possèdent un droit d'accès en lecture uniquement. Pour plus d'informations, voir «lock», à la page 365 et «unlock», à la page 383.

Vous pouvez utiliser des mots clés longs et courts lorsque vous émettez des commandes à partir de **composer**, comme indiqué dans tableau 20 et tableau 21. Les deux premières colonnes sur la gauche répertorient les formats de mots clés longs et courts actuellement supportés par IBM Workload Scheduler. La colonne la plus à droite répertorie les formats compatibles avec les versions antérieures à utiliser si votre réseau inclut des installations antérieures à la version 8.3.

Tableau 20. Liste des mots clés d'objet de planification pris en charge

Mots clés longs	Mots clés courts	Mots clés compatibles avec les installations antérieures à la version 8.3
calendar	cal	calendars
domain	dom	UC
eventrule	erule er	—
jobdefinition	jd	jobs
jobstream	js	sched
parameter	parm	parms
prompt	prom	prompts
resource	res	resources
user	user	users
variabletable	vt	—
wat	wat	—
workstation	ws	UC
workstationclass	wscl	UC

Remarque : Le mot clé **cpu** est conservé pour la compatibilité des domaines, stations de travail et classes de station de travail avec les versions antérieures.

Tableau 21. Liste des mots clés d'objet de sécurité pris en charge

Mots clés longs	Mots clés courts	Mots clés compatibles avec les installations antérieures à la version 8.3
accesscontrollist	acl	—
securitydomain	sdom	—
securityrole	srol	—

Le programme **composer** n'émet pas d'avertissement particulier lorsque des mots clés du langage de planification sont utilisés comme noms d'objet de planification. Toutefois, l'utilisation de ces mots clés peut entraîner des erreurs ; il est donc

conseillé d'éviter d'utiliser les mots clés répertoriés dans le tableau 22 lorsque vous définissez des travaux et des flots de travaux :

Tableau 22. Liste des mots réservés lors de la définition de travaux et de flots de travaux

Liste des mots réservés :				
abendprompt	after	as	at	autodocoff
autodocon	canc	carryforward	confirmed	continuer
dateval	day(s)	day_of_week	deadline	description
docommand	draft	end	every	everyday
except	extraneous	fdignore	fdnext	fdprev
filename	follows	freedays	from	go
hi	i18n_id	i18n_priority	interactive	isdefault
isuserjob	jobfilename	jobs	join	keyjob
keysched	limit	matching	members	needs
nextjob	nop	notempty	number	on
onuntil	op	opens	order	outputcond
previous	priority	prompt	qualifier	rccondsucc
reprise	relative	request	rerun	runcycle
sa	sameday	schedtime	schedule	scriptname
stop	streamlogon	su	succoutputcond	tasktype
timezone	to	token_in	until	validfrom
validto	vartable	vt	weekday(s)	workday(s)

Evitez d'utiliser les mots clés répertoriés dans le tableau 23 lors de la définition de postes de travail, de classes de postes de travail et de domaines :

Tableau 23. Liste des mots réservés lors de la définition de postes de travail

Liste des mots réservés :				
access	AIX	agent_type	autolink	behindfirewall
command	cpuclass	cpuname	description	domain
enabled	end	extraneous	for	force
fta	fullstatus	host	hpux	ibm i
ignore	isdefault	linkto	maestro	manager
master	members	mpeix	mpev	mpexl
mpix	node	number	off	on
os	other	parent	posix	server
secureaddr	securitylevel	tcpaddr	timezone	type
tz	tzid	UNIX	using	vartable
wnt				

Evitez d'utiliser les mots clés répertoriés dans le tableau 24, à la page 166 lors de la définition d'utilisateurs :

Tableau 24. Liste des mots réservés lors de la définition d'utilisateurs

Liste des mots réservés :		
username	mot de passe	end

Utilisation des modèles de définition d'objet

Les modèles de définition d'objet de planification sont à votre disposition dans le répertoire *racine_TWS\templates*. Vous pouvez les utiliser comme point de départ pour la définition des objets de planification.

Notez que les dates des modèles sont présentées au format indiqué dans l'option locale date format.

Définition de poste de travail

Dans un réseau IBM Workload Scheduler, un poste de travail est un objet de planification qui exécute des travaux. Vous créez et gérez le poste de travail d'objet de planification dans la base de données IBM Workload Scheduler à l'aide d'une définition de poste de travail. Une définition de poste de travail est requise pour tout objet qui exécute les travaux. En général, une définition de poste de travail permet de représenter un poste de travail physique. Cependant, en présence d'agents étendus, par exemple, elle représente une définition logique qui doit être hébergée par un poste de travail physique.

/ Lors de la définition d'un nouveau poste de travail, vous pouvez choisir le type de
/ licence à appliquer à ce dernier. Le type de licence détermine les critères par
/ rapport auxquels l'exécution de la charge de travail par IBM Workload Scheduler
/ sur le poste de travail est comptabilisée et la consommation des licences évaluée.
/ Pour plus d'informations sur la gestion des licences, voir la section relative à la
/ **Gestion des licences dans IBM License Metric Tool** dans le *Guide d'administration*.

Vous pouvez faire figurer plusieurs définitions de poste de travail dans un même fichier texte, avec des définitions de classes de postes de travail et des définitions de domaines. Une définition de poste de travail a la syntaxe suivante :

Syntaxe

```
/
cpuname poste_de_travail [description "description"]
[licensetype type]
[variable nom_table]
os type_sys_expl
[node nom_hôte] [tcpaddr port]
[secureaddr port][timezone | tz fuseau_horaire]
[domain nom_domaine]
[for maestro [host host-workstation [access method | agentID agentID ]]
    [type fta | s-agent | x-agent | manager | broker | agent | rem-eng |
    pool | d-pool]
    [ignore]
    [autolink on | off]
    [behindfirewall on | off]
    [securitylevel enabled | on | force]
    [fullstatus on | off]
    [server id_serveur]]
[protocol http | https]
```

```

    [membres [poste_travail] [...]]
    [exigences définition_jsdl]]
end

```

```

[cpuname ...]

```

```

[cpuclass ...]

```

```

[domain
...]

```

Arguments

Tableau 25. Paramètres des attributs pour la gestion des types de poste de travail

Attributs	Gestionnaire de domaine maître	Gestionnaire de domaine	Gestionnaire de domaine de sauvegarde
cpuname	Nom du poste de travail.		
description	Description du poste de travail entre guillemets. Cet attribut est facultatif.		
licensetype	Indique le type de licence à appliquer au poste de travail.		
vartable	Nom d'une table de variables associée à ce poste de travail. Les variables utilisées avec le poste de travail sont définies dans cette table. Cet attribut est facultatif.		
os	Système d'exploitation installé sur le système. Spécifiez l'une des valeurs suivantes : UNIX WNT OTHER IBM_i		
node	Nom d'hôte ou adresse IP du système.		
tcpaddr	Valeur affectée à <i>nm port</i> dans le fichier <i>localopts</i> . Pour plusieurs postes de travail sur un système, entrez un numéro de port non utilisé. La valeur par défaut est 3111.		
secureaddr	Valeur affectée à <i>nm ssl port</i> dans le fichier <i>localopts</i> . Spécifiez-la si securitylevel est défini sur <i>on</i> , <i>force</i> ou <i>enabled</i> .		
timezone tz	Fuseau horaire du système. La valeur doit correspondre à celle définie sur le système d'exploitation.		
domain	MASTERDM	Nom du domaine géré.	
host	Non applicable		
access	Non applicable		
type	manager		fta
ignore	Utilisez cet attribut si vous souhaitez que ce poste de travail ne s'affiche pas sur le prochain plan de production.		
autolink	Indique si un lien entre les postes de travail s'ouvre automatiquement au démarrage. Spécifiez l'une des valeurs suivantes : ON OFF Il s'agit d'un attribut facultatif. La valeur par défaut est ON.		

Tableau 25. Paramètres des attributs pour la gestion des types de poste de travail (suite)

Attributs	Gestionnaire de domaine maître	Gestionnaire de domaine	Gestionnaire de domaine de sauvegarde
behindfirewall	Ce paramètre est ignoré.	Il indique si un pare-feu existe entre le poste de travail et le gestionnaire de domaine maître. Spécifiez l'une des valeurs suivantes : ON OFF La valeur par défaut est OFF.	
securitylevel	Type de l'authentification SSL à utiliser : enabled on force		
fullstatus	ON		
server	Non applicable		Ce paramètre est ignoré.
protocol	Non applicable		
members	Non applicable		
requirements	Non applicable		

Le tableau 26 décrit les valeurs que vous avez définies pour chaque attribut des types de poste de travail cible. Vous trouverez ci-dessous un tableau avec des détails supplémentaires sur chaque attribut.

Tableau 26. Paramètres des attributs pour les types de poste de travail cible

Attribut	Agent tolérant aux pannes et agent standard	Poste de travail courtier de charge de travail	Agent étendu	Agent	Poste de travail de moteur distant	Pool	Pool dynamique
cpuname	Nom du poste de travail.						
description	Description du poste de travail entre guillemets. Cet attribut est facultatif.						
licensetype	Indique le type de licence à appliquer au poste de travail. Cette propriété s'applique aux types de poste de travail suivants : • Agent tolérant aux pannes • Agent standard • Agent						
variable	Nom d'une table de variables associée à ce poste de travail. Les variables utilisées avec le poste de travail sont définies dans cette table. Cet attribut est facultatif.						

Tableau 26. Paramètres des attributs pour les types de poste de travail cible (suite)

Attribut	Agent tolérant aux pannes et agent standard	Poste de travail courtier de charge de travail	Agent étendu	Agent	Poste de travail de moteur distant	Pool	Pool dynamique
os	Système d'exploitation installé sur le système. Spécifiez l'une des valeurs suivantes : UNIX WNT OTHER IBM_i Spécifiez OTHER pour les systèmes IBM i qui s'exécutent en tant qu'agents tolérants aux pannes limités.	OTHER	Système d'exploitation installé sur la machine. Spécifiez l'une des valeurs suivantes : UNIX WNT OTHER IBM_i	Ce paramètre de valeur est reconnu sur le système.	Système d'exploitation installé sur la machine. Spécifiez l'une des valeurs suivantes : UNIX WNT ZOS	Système d'exploitation installé sur la machine. Spécifiez l'une des valeurs suivantes : UNIX WNT OTHER IBM_i	
node	Nom d'hôte ou adresse IP du système.		Nom d'hôte ou adresse IP du système. Spécifiez NULL lorsque host est défini sur \$MASTER, ou lorsque vous définissez un agent étendu pour PeopleSoft, SAP ou Oracle.	Nom d'hôte ou adresse IP de l'agent.	Nom d'hôte ou adresse IP du moteur distant.	Non applicable	
tcpaddr	Valeur affectée à <i>nm port</i> dans le fichier localopts. Lorsque vous définissez plusieurs postes de travail sur un système, entrez un numéro de port non utilisé. La valeur par défaut est 31111.	Valeur affectée à <i>nm port</i> dans le fichier localopts. Lorsque vous définissez plusieurs postes de travail sur un système, entrez un numéro de port non utilisé. La valeur par défaut est 41114.	Voir les spécifications de la méthode d'accès sélectionnée.	Numéro de port permettant de communiquer avec l'agent lorsque le protocole est http.	Numéro de port permettant de communiquer avec le moteur distant lorsque le protocole est http.	Non applicable	
secureaddr	Valeur affectée à <i>nm ssl port</i> dans le fichier localopts. Spécifiez-la si securitylevel est défini sur on, force ou enabled.	Non applicable	Non applicable	Numéro de port permettant de communiquer avec l'agent lorsque le protocole est https.	Numéro de port permettant de communiquer avec le moteur distant lorsque le protocole est https.	Non applicable	

Tableau 26. Paramètres des attributs pour les types de poste de travail cible (suite)

Attribut	Agent tolérant aux pannes et agent standard	Poste de travail courtier de charge de travail	Agent étendu	Agent	Poste de travail de moteur distant	Pool	Pool dynamique
timezone tz	Fuseau horaire du système. La valeur doit correspondre à celle définie sur le système d'exploitation.		Le fuseau horaire défini sur le poste de travail spécifié dans l'attribut host .	Le fuseau horaire défini sur l'agent.	Le fuseau horaire défini sur le moteur distant.	Le fuseau horaire défini sur les agents du pool.	Le fuseau horaire défini sur les agents du pool dynamique.
domain	Spécifiez un domaine existant. La valeur par défaut pour les agents tolérants aux pannes est MASTERDM. Ce paramètre est obligatoire pour les agents standard.	Spécifiez un domaine existant. Ce paramètre est obligatoire.	Ce paramètre est nécessaire uniquement si la valeur affectée à host est \$MANAGER.	Non applicable			
host	Non applicable		Poste de travail hôte. Il peut être défini sur \$MASTER ou \$MANAGER.	Le poste de travail courtier.			
access	Non applicable			Sélectionnez le nom du fichier de méthode d'accès approprié.	Non applicable		
agentID				Identificateur unique de l'agent			
type	fta s-agent La valeur par défaut est fta. Spécifiez fta pour les systèmes IBM i qui s'exécutent en tant qu'agents tolérants aux pannes limités.	broker	x-agent	agent	rem-eng	pool	d-pool
ignore	Utilisez cet attribut si vous ne souhaitez pas que ce poste de travail apparaisse dans le plan de production suivant.						
autolink	Indique si un lien entre les postes de travail est automatiquement ouvert au démarrage. Spécifiez l'une des valeurs suivantes : ON OFF Il s'agit d'un attribut facultatif. La valeur par défaut est ON.		OFF	Non applicable			

Tableau 26. Paramètres des attributs pour les types de poste de travail cible (suite)

Attribut	Agent tolérant aux pannes et agent standard	Poste de travail courtier de charge de travail	Agent étendu	Agent	Poste de travail de moteur distant	Pool	Pool dynamique
behindfirewall	Il indique si un pare-feu existe entre le poste de travail et le gestionnaire de domaine maître. Spécifiez l'une des valeurs suivantes : ON OFF La valeur par défaut est OFF.		OFF	Non applicable			
securitylevel	Type de l'authentification SSL à utiliser : enabled on force Non applicable pour les systèmes IBM i qui s'exécutent en tant qu'agents tolérants aux pannes limités.	Non applicable					
fullstatus	Indique si le poste de travail est mis à jour pour l'état de traitement des travaux dans son domaine et ses sous-domaines. Spécifiez l'une des valeurs suivantes : ON OFF Spécifiez OFF pour les agents standard.	OFF		Non applicable			
server	0-9, A-Z. Lorsqu'il est spécifié, requiert la création de processus mailman dédiés sur le poste de travail parent.		Non applicable				
protocol	Non applicable			Spécifiez l'une des valeurs suivantes : http https Cet attribut est facultatif. S'il n'est pas spécifié, il est automatiquement déterminé à partir des paramètres spécifiés pour tcpaddr et secureaddr.		Non applicable	
members	Non applicable					Valeur requise	Non applicable
requirements	Non applicable						Valeur requise

La liste suivante fournit des détails supplémentaires sur les attributs de définition de poste de travail :

cpuname *poste_de_travail*

Indique le nom du poste de travail. Les noms de poste de travail doivent être uniques et ne peuvent pas être identiques à des noms de classes de postes de travail.

Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut comporter jusqu'à 16 caractères.

N'utilisez pas dans cette zone l'un des mots réservés spécifiés dans le tableau 22, à la page 165.

description *"description"*

Fournit une description du poste de travail. La description peut contenir jusqu'à 120 caractères alphanumériques. Ce texte doit être délimité par des guillemets.

licensetype *type*

Indique le type de licence à appliquer au poste de travail. Utilisez ce mot clé lorsque vous définissez le mot clé **licenseType** optman sur **byWorkstation**. Le type de licence détermine les critères par rapport auxquels l'exécution de la charge de travail par IBM Workload Scheduler sur le poste de travail est comptabilisée et la consommation des licences évaluée. Pour plus d'informations sur la gestion des licences, voir la section relative à la **Gestion des licences dans IBM License Metric Tool** dans le *Guide d'administration*. Les valeurs prises en charge sont les suivantes :

perJob

Votre consommation de licences est évaluée en fonction du nombre de travaux qui sont exécutés chaque mois sur le poste de travail.

perServer

Votre consommation de licences est évaluée en fonction du nombre de composants IBM Workload Scheduler facturables installés sur le poste de travail.

Si vous installez un agent dynamique et un agent tolérant aux pannes sur le même poste de travail dans le même chemin, les agents partagent le même fichier de licence. Par conséquent, seul le type de licence défini le plus récemment s'applique. Pour cette raison, il est recommandé de définir le même type de licence pour les deux postes de travail d'agent.

variable *nom_table*

Indique le nom de la table de variables que vous souhaitez associer au poste de travail. Les variables utilisées avec le poste de travail sont définies dans cette table.

Il doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut contenir jusqu'à 80 caractères.

os *type_système-exploitation*

Indique le système d'exploitation du poste de travail. Lorsqu'il est utilisé dans les définitions de poste de travail de moteur distant, il correspond au système d'exploitation du moteur distant IBM Workload Scheduler.

Les valeurs valides sont les suivantes :

UNIX Pour les systèmes d'exploitation pris en charge sur les systèmes basés sur UNIX, y compris les systèmes LINUX.

WNT Pour les systèmes d'exploitation Windows pris en charge.

OTHER

Valeur obligatoire pour : les postes de travail Dynamic Workload Broker et les systèmes IBM i qui s'exécutent en tant qu'agents tolérants aux pannes limités. Valeur facultative pour d'autres types de postes de travail.

ZOS Utilisé avec les postes de travail distants qui sont définis pour communiquer avec le moteur distant IBM Workload Scheduler for z/OS.

IBM_i Pour les systèmes d'exploitation IBM i pris en charge.

Remarque : Pour la liste des systèmes d'exploitation pris en charge, voir le document *IBM Workload Scheduler System Requirements* à l'adresse <http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048858>.

node *nom_hôte*

Indiquez le nom d'hôte ou l'adresse TCP/IP du poste de travail. Les noms de domaines complets sont admis.

Pour les noms d'hôtes, les caractères alphanumériques sont autorisés, y compris le tiret (-). La longueur est limitée à 51 caractères.

Spécifiez NULL lorsque :

- Vous définissez un agent étendu pour PeopleSoft, SAP ou Oracle.
- **host** est défini sur \$MASTER

Si vous définissez un poste de travail distant, spécifiez le nom d'hôte du système sur lequel le moteur distant est installé.

tcpaddr *port*

Indique le numéro de port TCP/IP **netman** que IBM Workload Scheduler utilise pour la communication entre les postes de travail.

Pour les postes de travail courtiers de charge de travail

Spécifiez la valeur de la propriété **TWS.Agent.Port** du fichier `TWSAgentConfig.properties`.

Pour les postes de travail distants utilisant le protocole HTTP pour communiquer avec le moteur distant

Spécifiez le numéro de port HTTP du moteur distant.

Pour les autres types de postes de travail

Spécifiez la valeur affectée à la variable *nm port* dans le fichier `localopts`.

La valeur par défaut de cette zone est 31111. Indiquez une valeur comprise entre 1 et 65535.

secureaddr

Définit le port utilisé pour détecter les connections SSL entrantes. Cette valeur est lue lorsque l'attribut **securitylevel** est défini.

Pour les postes de travail courtiers de charge de travail

Ignorez cet attribut.

Pour les postes de travail de moteur distant utilisant le protocole HTTPS pour communiquer avec le moteur distant

Spécifiez le numéro de port HTTPS du moteur distant.

Pour les autres types de postes de travail

Spécifiez la valeur affectée à la variable *nm ssl port* dans le fichier *localopts*. La valeur doit être différente de celle affectée à la variable *nm port* dans le fichier *localopts*.

Si **securitylevel** est spécifié, mais que cet attribut est manquant, la valeur par défaut de cette zone est 31113. Indiquez une valeur comprise entre 1 et 65535. Voir *IBM Workload Scheduler - Guide d'administration* pour plus d'informations sur l'authentification SSL et les options locales définies dans le fichier de configuration *racine_TWS/localopts*.

timezone | *tz fuseau_horaire*

Indique le fuseau horaire du poste de travail. Pour garantir l'exactitude des heures de planification, ce fuseau horaire doit être identique à celui du système d'exploitation de l'ordinateur.

Lorsqu'il est utilisé dans les définitions de poste de travail de moteur distant, il correspond au fuseau horaire défini sur le moteur distant IBM Workload Scheduler.

Pour plus d'informations sur les noms de fuseau horaire valides, voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799.

domain *nom_domaine*

Indique le nom du domaine IBM Workload Scheduler auquel appartient le poste de travail. La valeur par défaut pour un poste de travail tolérant aux pannes est MASTERDM.

IBM Workload Scheduler ignore le paramètre de domaine lorsqu'il est défini pour les agents étendus, excepté lorsque l'attribut **host** est défini par \$MASTER.

Ce paramètre est obligatoire pour les postes de travail d'agent standard et de courtier de charge de travail dynamiques.

host *poste_de_travail_hôte*

Cet attribut est obligatoire pour les agents étendus et les postes de travail de moteur distant et spécifie :

Pour les postes de travail de moteur distant, les agents, les pools et les pools dynamiques :

Le poste de travail courtier hébergeant le poste de travail. Cette zone ne peut pas être mise à jour après la création du poste de travail de moteur distant.

Pour les agents étendus

Le poste de travail avec lequel l'agent étendu communique et sur lequel sa méthode d'accès est installée. Le poste de travail hébergeant ne peut pas être un autre agent étendu.

Si le poste de travail hébergeant est un gestionnaire de domaine pour lequel vous avez défini une sauvegarde, vous pouvez spécifier une des valeurs suivantes pour garantir que l'agent étendu n'est pas isolé si le poste de travail hébergeant devient indisponible :

\$MASTER

Indiquer que le poste de travail hôte de l'agent étendu est le gestionnaire de domaine maître.

\$MANAGER

Indiquer que le poste de travail hôte de l'agent étendu est le gestionnaire de domaine. Dans ce cas, vous devez spécifier le domaine dans lequel est situé l'agent.

Dans ce cas, vérifiez que les méthodes d'agent étendu sont également installées sur le poste de travail de sauvegarde. Vous pouvez activer et désactiver la résolution automatique de la clé \$MASTER à l'aide de l'option *mm resolve master* dans le fichier *localopts*.

Pour plus d'informations sur les options disponibles dans le fichier *localopts*, voir *IBM Workload Scheduler - Guide d'administration*.

access *méthode*

Indique une méthode d'accès pour des agents étendus et des agents réseau. Il doit s'agir du nom d'un fichier situé dans le répertoire *racine_TWS/methods* du poste de travail hébergeant.

Spécifiez NULL lorsque vous définissez un agent étendu pour PeopleSoft, SAP ou Oracle.

agentID *agentID*

Identificateur unique de l'agent.

type Indique le type de poste de travail. Si vous planifiez la modification des types de postes de travail, prenez en compte les règles suivantes :

- Vous pouvez modifier les postes de travail d'agent tolérant aux pannes, d'agent standard, d'agent étendu, de gestionnaire de domaine et de courtier de charge de travail dynamiques par tout type de poste de travail, à l'exception de l'agent dynamique, du pool, du pool dynamique et du moteur distant.
- Vous ne pouvez pas modifier le type de l'agent dynamique, du pool, du pool dynamique et du moteur distant.

Entrez l'une des valeurs suivantes :

fta Si vous définissez un *agent tolérant aux pannes*, il s'agit d'un poste de travail d'agent qui lance les travaux et résout les dépendances locales sans gestionnaire de domaine. Il s'agit de la valeur de la valeur par défaut de cet attribut.

Vous devez spécifier *fta* si vous souhaitez affecter au poste de travail le rôle de gestionnaire de domaine de sauvegarde ou de gestionnaire de domaine maître de sauvegarde.

Spécifiez *fta* pour les systèmes IBM i qui s'exécutent en tant qu'agents tolérants aux pannes limités.

s-agent

Si vous définissez un *agent standard*, il s'agit d'un poste de travail d'agent qui lance les travaux uniquement sous l'ordre de son gestionnaire de domaine.

x-agent

Si vous définissez un *agent étendu*, il s'agit d'un poste de travail d'agent qui lance les travaux uniquement sous l'ordre de son poste

de travail hébergeant. Les agents étendus peuvent servir d'interface entre IBM Workload Scheduler et des systèmes ou des applications non IBM.

Pour plus d'informations, voir *IBM Workload Scheduler for Applications*.

manager

Si vous définissez un *gestionnaire de domaine*, il s'agit d'un poste de travail qui gère un domaine. Lors de la définition de ce type de poste de travail, spécifiez :

Server NULL

Domaine, zone

Nom du domaine géré par le poste de travail, s'il est différent du domaine MASTERDM.

Vous spécifiez qu'un poste de travail est un gestionnaire également dans la zone *manager* de la «Définition de domaine», à la page 185. IBM Workload Scheduler vérifie automatiquement que les valeurs spécifiées dans ces zones sont cohérentes.

broker

Si vous définissez un poste de travail *courtier de charge de travail dynamique*, il s'agit d'un poste de travail qui exécute à la fois les types de travaux existants et les types de travaux avec options avancées. Il s'agit du serveur courtier installé avec le gestionnaire de domaine maître et le gestionnaire de domaine dynamique. Il héberge les postes de travail suivants :

- Agent étendu
- Moteur distant
- Pool
- Pool dynamique
- Agent. Cette définition comprend les agents suivants :
 - agent
 - agent IBM Workload Scheduler for z/OS
 - agent pour z/OS

Pour plus d'informations sur l'agent et l'agent IBM Workload Scheduler for z/OS, voir *Scheduling Workload Dynamically*. Pour plus d'informations sur le agent pour z/OS, voir *Planification avec l'agent pour z/OS*.

agent

Si vous définissez un *agent dynamique*, il s'agit d'un poste de travail qui gère une grande variété de types de travaux, par exemple, des travaux de base de données spécifiques ou de transfert de fichiers, en plus des types de travaux traditionnels. Il est hébergé par le poste de travail Workload Broker. La définition de poste de travail est automatiquement créée et enregistrée lorsque vous installez le composant agent dynamique. Dans sa définition, vous pouvez éditer uniquement les attributs suivants :

- Description
- Vartable

Remarque : Si l'option globale `enAddWorkstation` vaut "yes", la définition de poste de travail d'agent dynamique est

automatiquement ajoutée au plan après que le processus d'installation a créé le poste de travail d'agent dynamique dans la base de données.

rem-eng

Si vous définissez un *poste de travail de moteur distant*, il s'agit d'un poste de travail utilisé pour communiquer avec un moteur distant lors de la liaison d'un travail défini localement, appelé *travail reflet*, à un travail spécifique qui s'exécute sur le moteur distant, appelé *travail distant*. Lorsque les deux travaux sont liés, la transition d'état du travail reflet est mappée vers la transition d'état du travail distant. Ce mappage permet également de définir et de surveiller les dépendances des travaux locaux sur les travaux qui s'exécutent sur le moteur distant ; ces dépendances sont appelées *dépendances croisées*.

Pour plus d'informations sur les travaux reflet et les dépendances croisées, voir Chapitre 23, «Définition et gestion des dépendances croisées», à la page 845.

Lors de la définition de ce type de poste de travail, renseignez les zones suivantes :

- os** Le système d'exploitation du moteur distant.
- hôte** Le nom du poste de travail courtier hébergeant.
- node** Le nom d'hôte ou l'adresse IP du moteur distant.

Lorsque vous spécifiez le numéro de port à utiliser pour communiquer avec le moteur distant, utilisez **secureaddr** si le protocole utilisé est HTTPS ou **tcpaddr** si le protocole utilisé est HTTP. Il est recommandé de spécifier dans la zone **fuseau horaire** le fuseau horaire défini sur le moteur distant.

pool

Si vous définissez un *pool*, il s'agit d'un ensemble d'agents dynamiques avec des caractéristiques matérielles ou logicielles identiques sur lequel vous soumettez les travaux. Ce poste de travail est hébergé par le poste de travail Workload Broker. Dans sa définition, vous pouvez éditer uniquement les attributs suivants :

- Description
- Vartable
- Members

d-pool

Si vous définissez un *pool dynamique*, il s'agit d'un ensemble d'agents dynamiques qui est basé dynamiquement sur les exigences répertoriées dans le fichier JSDL spécifié dans l'attribut **resources**. Ce poste de travail est hébergé par le poste de travail Workload Broker. Dans sa définition, vous pouvez éditer uniquement les attributs suivants :

- Description
- Vartable
- Requirements

ignore Indique que la définition de poste de travail ne doit pas être ajoutée au plan de production. Si vous spécifiez ce paramètre, les flots de travaux planifiés pour s'exécuter sur ce poste de travail ne sont pas ajoutés au plan de production.

autolink

Indique si les postes de travail doivent être connectés au démarrage. Selon le type du poste de travail, lorsque vous définissez sa valeur sur on :

Sur un agent tolérant aux pannes ou un agent standard

Cela implique que le gestionnaire de domaine ouvre le lien à l'agent lorsque le gestionnaire de domaine est démarré.

Sur un gestionnaire de domaine

Cela implique que ses agents ouvre des liens au gestionnaire de domaine à leur démarrage.

Ce paramètre est particulièrement utile lorsqu'un nouveau plan de production est créé sur le gestionnaire de domaine maître : dans le cadre de la génération du plan de production, tous les postes de travail sont arrêtés puis redémarrés. Pour tous les agents pour lesquels l'option **autolink** est activée, le gestionnaire de domaine envoie automatiquement une copie du nouveau plan de production et démarre l'agent. Si l'option **autolink** est également activée pour le gestionnaire de domaine, l'agent, à son tour, se connecte au gestionnaire de domaine.

Si la valeur de l'option **autolink** est off pour un agent, vous pouvez ouvrir un lien à partir de son gestionnaire de domaine en exécutant la commande **conman link** sur le gestionnaire de domaine ou le gestionnaire de domaine maître de l'agent.

behindfirewall

Si cette option est définie sur on, cela signifie qu'il existe un pare-feu entre le poste de travail et le gestionnaire de domaine maître. Dans ce cas, seule une connexion directe entre le poste de travail et son gestionnaire de domaine est autorisée et les commandes **start poste_de_travail**, **stop poste_de_travail** et **showjobs** sont envoyées selon la hiérarchie du domaine. Il n'est pas nécessaire de faire établir une connexion directe avec le poste de travail par le gestionnaire de domaine maître ou le gestionnaire de domaine.

Définissez cet attribut par off si vous spécifiez un poste de travail de type broker.

fullstatus

Spécifiez ce paramètre lorsque vous définissez un poste de travail d'agent tolérant aux pannes. Pour les gestionnaires de domaine, ce mot clé est automatiquement défini sur on. Spécifiez l'une des valeurs suivantes :

- le** Si vous souhaitez que le poste de travail d'agent tolérant aux pannes fonctionne en mode *statut intégral*. Cela signifie que le poste de travail est mis à jour avec l'état des travaux et flots de travaux s'exécutant sur tous les postes de travail de son domaine et dans les domaines subordonnés, mais pas dans les domaines homologues ou parents. La copie du plan de production sur l'agent est conservée avec le même niveau de détails que la copie du plan de production sur son gestionnaire de domaine.
- off** Si vous souhaitez que le poste de travail d'agent tolérant aux pannes soit informé de l'état des travaux et flots de travaux uniquement sur les postes de travail qui affectent ses propres travaux et flots de travaux. En spécifiant "off", vous pouvez améliorer les performances en réduisant l'activité du réseau.

securitylevel

Indique le type d'authentification SSL pour le poste de travail. Ne spécifiez pas cet attribut pour un poste de travail de type broker. Il peut prendre l'une des valeurs suivantes :

- activé** Le poste de travail utilise l'authentification uniquement si le poste de travail de son gestionnaire de domaine ou un autre agent tolérant aux pannes situé en-dessous dans la hiérarchie des domaines l'exige.
- le** Le poste de travail utilise une authentification SSL lorsqu'il établit une connexion avec son gestionnaire de domaine. Le gestionnaire de domaine l'utilise lorsqu'il se connecte à son gestionnaire de domaine parent. L'agent tolérant aux pannes refuse toute connexion entrante autre que SSL à partir de son gestionnaire de domaine.
- force** Le poste de travail utilise l'authentification SSL pour toutes ses connexions et accepte les connexions à la fois du gestionnaire de domaine parent et du gestionnaire de domaine subordonné. Il refuse toute connexion entrante non-SSL.

Si cet attribut fait défaut, le poste de travail n'est pas configuré pour les connexions SSL et toute valeur de **secureaddr** est ignorée. Dans ce cas, assurez-vous que l'option locale *nm ssl port* est définie sur 0 pour garantir que le processus **netman** ne tente pas d'ouvrir le port spécifié dans **secureaddr**. Voir *IBM Workload Scheduler - Guide d'administration* pour plus d'informations sur l'authentification SSL.

Le tableau suivant décrit le type de communication utilisé pour chaque type de configuration **securitylevel**.

Tableau 27. Type de communication en fonction de la valeur *securitylevel*

Valeur définie sur l'agent tolérant aux pannes (ou le gestionnaire de domaine)	Valeur définie sur son gestionnaire de domaine (ou sur son gestionnaire de domaine parent)	Type de connexion établie
Non spécifié	Non spécifié	TCP/IP
Activé	Non spécifié	TCP/IP
On	Non spécifié	Pas de connexion
Force	Non spécifié	Pas de connexion
Non spécifié	On	TCP/IP
Activé	On	TCP/IP
Actif	On	SSL
Force	On	SSL
Non spécifié	Activé	TCP/IP
Activé	Activé	TCP/IP
On	Activé	SSL
Force	Activé	SSL
Non spécifié	Force	Pas de connexion
Activé	Force	SSL
On	Force	SSL
Force	Force	SSL

La valeur de **securitylevel** n'est pas spécifiée pour les postes de travail

courtiers de charge de travail dynamiques ou pour les postes de travail comportant l'agent IBM Workload Scheduler version 8.2 ou antérieure.

server *ID_serveur*

Utilisez l'attribut **server** dans la définition de poste de travail d'agent tolérant aux pannes pour réduire le temps requis pour initialiser des agents et améliorer l'opportunité des messages. Par défaut, les communications avec les agent tolérant aux pannes sont gérés par un processus **mailman** exécuté sur le gestionnaire de domaine. L'attribut **server** vous permet de démarrer le processus **mailman** sur le gestionnaire de domaine afin de traiter les communications avec ce poste de travail d'agent tolérant aux pannes uniquement.

Si vous définissez un agent tolérant aux pannes qui peut fonctionner en tant que gestionnaire de domaine de sauvegarde, l'*ID_serveur* est utilisé uniquement lorsque le poste de travail fonctionne en tant qu'agent tolérant aux pannes ; le paramètre est ignoré lorsque le poste de travail fonctionne en tant que gestionnaire de domaine de sauvegarde.

Dans *ID_serveur*, l'ID correspond à une seule lettre ou à un chiffre (A-Z et 0-9). Les ID sont uniques à chaque gestionnaire de domaine, de sorte qu'il est possible d'utiliser les mêmes ID dans d'autres domaines sans créer de conflit. Un *ID_serveur* spécifique peut être dédié à plusieurs postes de travail d'agent tolérant aux pannes.

Les valeurs recommandées sont les suivantes :

- Si vous avez besoin de plus de 36 ID serveur dans un domaine, pensez à diviser ce dernier en deux domaines ou plus.
- Si le même ID est utilisé pour plusieurs agents, un seul serveur est créé pour gérer leurs communications. Vous devez définir des serveurs supplémentaires afin d'éviter qu'un seul serveur gère plus de huit agents.

Si aucun *ID_serveur* n'est indiqué, les communications avec l'agent sont traitées par le processus **mailman** du gestionnaire de domaine.

protocol **http** | **https**

Indique le protocole à utiliser pour communiquer avec :

Le poste de travail courtier

Si le poste de travail est un poste de travail d'agent.

Le moteur distant

Si le poste de travail est un poste de travail de moteur distant.

members [*poste de travail*] [...]

Utilisez cette valeur pour un poste de travail de pool afin de spécifier les agents que vous souhaitez ajouter au pool.

requirements *définition_jsdl*

Utilisez cette valeur pour un poste de travail de pool dynamique afin de spécifier les exigences, au format JSDL, que les agents doivent satisfaire pour appartenir automatiquement au pool dynamique. Vous utilisez la syntaxe suivante :

```
jsdl_definition:
<jsdl:resources>
<jsdl:logicalResource subType="MyResourceType" />
</jsdl:resources>
```

Pour plus d'informations sur la syntaxe JSDL, voir *Scheduling Workload Dynamically*.

Remarque : Vous pouvez ajouter des définitions de poste de travail à la base de données à tout moment mais vous devez réexécuter **JnextPlan -for 0000** afin de pouvoir exécuter des travaux sur les postes de travail récemment créés. Chaque fois que vous exécutez **JnextPlan**, tous les postes de travail sont arrêtés et redémarrés.

Exemples

Dans l'exemple suivant, vous allez créer un gestionnaire de domaine maître baptisé **hdq-1** et un agent tolérant aux pannes baptisé **hdq-2** dans le gestionnaire de domaine. Notez que l'argument **domain** est facultatif dans cet exemple, car la valeur par défaut du domaine maître est **masterdm**.

```
cpuname hdq-1 description "Headquarters master DM"
  os unix
  tz America/Los_Angeles
  node sultan.ibm.com
  domain masterdm
  for maestro type manager
    autolink on
    fullstatus on
  end
cpuname hdq-2
  os wnt
  tz America/Los_Angeles
  node opera.ibm.com
  domain masterdm
  for maestro type fta
    autolink on
  end
```

L'exemple suivant illustre la création d'un domaine nommé **distr-a** avec un gestionnaire de domaine nommé **distr-a1** et un agent standard nommé **distr-a2** :

```
domain distr-a
  manager distr-a1
  parent masterdm
end
cpuname
distr-a1 description "District A domain mgr"
  os wnt
  tz America/New_York
  node pewter.ibm.com
  domain distr-a
  for maestro type manager
    autolink on
    fullstatus on
  end
cpuname distr-a2
  os wnt
  node quatro.ibm.com
  tz America/New_York
  domain distr-a
  for maestro type s-agent
end
```

L'exemple suivant crée un poste de travail tolérant aux pannes avec l'authentification SSL. La définition de sécurité **securitylevel** indique que la connexion entre le poste de travail et son gestionnaire de domaine peut être uniquement de type SSL. Le port 32222 est réservé pour la détection des connexions SSL entrantes.

```

cpuname ENNETI3
  os WNT
  node apollo
  tcpaddr 30112
  secureaddr 32222
  for maestro
    autolink off
    fullstatus on
    securitylevel on
  end
end

```

Dans l'exemple suivant, un poste de travail de courtier est créé. Ce poste de travail gère le cycle de vie des travaux de type IBM Workload Scheduler Workload Broker dans Dynamic Workload Broker.

```

cpuname ITDWBAGENT
  vartable TABLE1
  os OTHER
  node itdwbst11.ibm.com TCPADDR 41114
  timezone Europe/Rome
  domain MASTERDM
  for MAESTRO
    type BROKER
    autolink OFF
    behindfirewall OFF
    fullstatus OFF
  end
end

```

L'exemple suivant crée un poste de travail de moteur distant à utiliser pour gérer les dépendances croisées et communiquer avec un moteur distant installé sur un système de nom d'hôte London-hdq utilisant le port HTTPS 31116 par défaut. Le poste de travail de moteur distant est hébergé par le poste de travail courtier ITDWBAGENT

```

cpuname REW_London
  description "Remote engine workstation to communicate with London-hdq"
  os WNT
  node London-hdq secureaddr 31116
  timezone Europe/London
  for maestro host ITDWBAGENT
    type rem-eng
    protocol HTTPS
  end
end

```

L'exemple suivant montre comment créer un pool dynamique d'agents. Tous les agents du pool dynamique doivent disposer du système d'exploitation HP-UX ou Linux installé :

```

CPUNAME DPOOLUNIX
  DESCRIPTION "Sample Dynamic Pool Workstation"
  VARIABLE table1
  OS OTHER
  TIMEZONE Europe/Rome
  FOR MAESTRO HOST MAS86MAS_DWB
  TYPE D-POOL
  REQUIREMENTS
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:resourceRequirements
  xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:resources>
    <jsdl:candidateOperatingSystems>
      <jsdl:operatingSystem type="HPUX"/>
      <jsdl:operatingSystem type="LINUX"/>
    </jsdl:candidateOperatingSystems>
  </jsdl:resources>
</jsdl:resourceRequirements>

```

```

        </jsdl:candidateOperatingSystems>
    </jsdl:resources>
</jsdl:resourceRequirements>
FIN

```

L'exemple suivant montre comment créer un pool dynamique d'agents. Sur tous les agents dans le pool dynamique, le système d'exploitation Windows 2000 doit être installé :

```

CPUNAME DPOOLWIN
DESCRIPTION "Sample Dynamic Pool Workstation"
OS WNT
TIMEZONE Europe/Rome
FOR MAESTRO HOST MAS86MAS_DWB
TYPE D-POOL
REQUIREMENTS
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:resourceRequirements
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
    <jsdl:resources>
        <jsdl:candidateOperatingSystems>
            <jsdl:operatingSystem type="Windows 2000"/>
        </jsdl:candidateOperatingSystems>
    </jsdl:resources>
</jsdl:resourceRequirements>
FIN

```

L'exemple suivant montre comment créer un pool d'agents nommé POOLUNIX et contenant deux agents : NC121105 et NC117248 :

```

CPUNAME POOLUNIX
DESCRIPTION "Sample Pool Workstation"
OS OTHER
TIMEZONE Europe/Rome
FOR MAESTRO HOST MAS86MAS_DWB
TYPE POOL
MEMBERS
    NC121105
    NC117248
FIN

```

L'exemple suivant indique comment créer un agent standard sur lequel la consommation des licences est évaluée en fonction du nombre de travaux exécutés chaque mois sur le poste de travail :

```

CPUNAME SA1
DESCRIPTION "Sample Standard Agent"
LICENSETYPE PERJOB
VARIABLE TABLE1
OS UNIX
NODE sa1.mycompany.com TCPADDR 31111
SECUREADDR 31113
TIMEZONE Europe/Rome
DOMAIN MASTERDM
FOR MAESTRO
TYPE S-AGENT
IGNORE
AUTOLINK ON
BEHINDFIREWALL ON
SECURITYLEVEL FORCE
FULLSTATUS OFF
SERVER B
FIN

```

Voir aussi

Pour savoir comment effectuer la même tâche à partir de Dynamic Workload Console, voir :

Dynamic Workload Console - Guide d'utilisation, section "Création de postes de travail distribués".

Définition de classe de postes de travail

Une classe de postes de travail correspond à un groupe de postes de travail pour lequel des travaux et des flots de travaux communs peuvent être écrits. Vous pouvez faire figurer plusieurs définitions de classes de travail dans un même fichier texte, avec des définitions de postes de travail et des définitions de domaines.

Lors de la définition des classes de poste de travail, assurez-vous que les postes de travail dans la classe prennent en charge les types de travail que vous prévoyez d'exécuter dessus. Les règles suivantes s'appliquent :

- Les types de travail avec options avancées s'exécutent uniquement sur les agents dynamiques, les pools et les pools dynamiques.
- Les travaux reflet s'exécutent uniquement sur les moteurs distants.

Remarque : Lors de la définition d'une classe de poste de travail contenant des agents tolérants aux pannes antérieurs au groupe de correctifs 1 version 9.3, les problèmes suivants peuvent survenir : gestion incorrecte des objets planifiés sur les agents tolérants aux pannes, signalisation incohérente des états des travaux et des flux de travaux, signalisation incohérente du nombre et des états des dépendances non satisfaites liées aux dépendances conditionnelles.

Chaque définition de classe de postes de travail utilise le format et les arguments suivants :

Syntaxe

```
cpuclass classes_poste_de_travail
  [description "description"]
  [ignore]
    members [poste_de_travail | @] [...]
end
```

[cpuname ...]

[cpuclass ...]

[domain ...]

Arguments

```
cpuclass classes_poste_de_travail
```

Nom de la classe de postes de travail. Il doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut comporter jusqu'à 16 caractères.

Remarque : Vous ne pouvez pas utiliser de noms identiques pour des postes de travail et des classes de postes de travail.

description *"description"*

Fournit une description de la classe de postes de travail. La description peut contenir jusqu'à 120 caractères alphanumériques. Ce texte doit être délimité par des guillemets.

ignore Indique que IBM Workload Scheduler doit ignorer tous les postes de travail appartenant à cette classe lors de la génération du plan de production.

members *poste_de_travail*

Présente une liste de noms de poste de travail, séparés par des espaces, membres de la classe. Le caractère générique @ signifie que la classe de postes de travail inclut tous les postes de travail.

Exemples

Dans l'exemple suivant, nous définissons une classe de postes de travail appelée backup :

```
cpuclass backup
  members
    main
    site1
    site2
end
```

Dans l'exemple suivant, nous définissons une classe de postes de travail appelée allcpus, qui contient chaque poste de travail :

```
cpuclass allcpus
  members
    @
end
```

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Définition de domaine

Un domaine correspond à un groupe de postes de travail comportant un ou plusieurs agents ainsi qu'un gestionnaire de domaine. Le gestionnaire de domaine se comporte de la même manière que le concentrateur de gestion pour les agents dans le domaine. Vous pouvez faire figurer plusieurs définitions de domaines dans un même fichier texte, avec des définitions de postes de travail et des définitions de classes de postes de travail. Chaque définition de domaine utilise le format et les arguments suivants :

Syntaxe

```
domain nom_domaine[description "description"]
  * manager poste_de_travail
  [parent nom_domaine | ismaster]
end

[cpuname ...]
```

[cpuclass ...]

[domain ...]

Arguments

domain *nom_domaine*

Nom du domaine. Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut comporter jusqu'à 16 caractères.

description "*description*"

Fournit une description du domaine. Ce texte doit être délimité par des guillemets.

*** manager** *poste_de_travail*

Il s'agit d'une zone commentée utilisée uniquement pour montrer, lors de l'affichage de la définition du domaine, le nom du poste de travail qui occupe le rôle de gestionnaire de domaine pour ce domaine. Assurez-vous que cette zone reste commentée. Elle est conservée à des fins de compatibilité avec les versions antérieures. Avec la IBM Workload Scheduler version 8.3, l'information indiquant si un poste de travail est un gestionnaire de domaine est spécifiée dans la zone **type** de «Définition de poste de travail», à la page 166.

parent *nom_domaine*

Nom du domaine parent auquel le gestionnaire de domaine est connecté. Il s'agit par défaut du domaine maître, qui ne requiert pas de définition de domaine. Il est défini par les options globales **master** et **master domain**.

ismaster

S'il est défini, indique que le domaine est le domaine supérieur du réseau IBM Workload Scheduler. S'il est défini, il ne peut pas être supprimé ultérieurement.

Exemples

Dans l'exemple suivant, nous définissons un domaine appelé east utilisant le domaine maître comme parent et deux domaines subordonnés baptisés northeast et southeast :

```
domain east description "The Eastern domain"
  * manager cyclops
end
domain northeast description "The Northeastern domain"
  * manager boxcar
  parent east
end
domain southeast description "The Southeastern domain"
  * manager sedan
  parent east
end
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Création d'un domaine".

Définition de travail

Un travail correspond à une commande, à un fichier exécutable ou à un programme planifié et exécuté par IBM Workload Scheduler. Vous pouvez écrire des définitions de travaux dans des fichiers d'édition, puis les ajouter à la base de données IBM Workload Scheduler à l'aide du programme **composer**. Vous pouvez faire figurer plusieurs définitions de travaux dans un même fichier d'édition.

Deux types de travaux différents sont disponibles : le travail IBM Workload Scheduler standard est un fichier exécutable générique, un programme ou une commande que vous exécutez de façon statique, tandis que les types de travaux avec options avancées sont des travaux prédéfinis que vous pouvez utiliser pour exécuter des tâches spécifiques, de façon statique ou dynamique, telles que les opérations de transfert de fichier ou l'intégration à d'autres bases de données.

Les types de travaux avec options avancées ne s'exécutent que sur les agents dynamiques, les pools, les pools dynamiques et les moteurs distants.

Pour définir des travaux standard dans la ligne de commande **composer**, vous utilisez les arguments **script** et **docommand**. Pour définir des types de travaux avec options avancées, vous utilisez l'argument **task**.

Pour plus d'informations sur types de travaux avec options avancées, voir Chapitre 14, «Extension des fonctions IBM Workload Scheduler», à la page 575.

Pour plus d'informations sur la transmission de variables entre des travaux appartenant à la même instance de flot de travaux, voir «Transmission de variables entre travaux», à la page 650.

Remarque : A partir de la version 9.4, groupe de correctifs 1 du produit, la ligne de commande **composer** permettant de créer des définitions de travail utilise les API REST. Cela signifie que lorsque vous créez un travail à l'aide de **composer**, de nouvelles API sont utilisées et celles-ci ne sont pas compatibles avec les API installées sur les maîtres avec les versions de produit précédentes. Vous ne pouvez pas donc utiliser un **composer** au niveau de version 9.4, groupe de correctifs 1 pour créer une définition de travail sur un maître sur lequel une version précédente du produit est installée.

Chaque définition de travail utilise le format et les arguments suivants :

Syntaxe

\$jobs

```
[poste_de_travail#]nom_travail
    {scriptname nom_fichier streamlogon nom_utilisateur |
      docommand "commande" streamlogon nom_utilisateur |
      task définition_travail }
    [description "description"]
    [tasktype type_tâche]
    [interactive]

    [succoutputcond nom_condition "valeur_condition"]
    [outputcond nom_Condition "valeur_Condition"]
```

```
[recovery
{stop
```

=
=

```
[after [poste_de_travail#]nom_travail]
[abendprompt "texte"]]
| continue
[after [poste_de_travail#]nom_travail]
[abendprompt "texte"]] | rerun [same_workstation]
[[repeatevery hmmm] [for nombre attempts]]
[after [poste_de_travail#]nom_travail]
| [after [poste_de_travail#]nom_travail]
[abendprompt "texte"]}
```

Dans la mesure où un travail proprement dit ne comporte pas de paramètres pour les dépendances, ces paramètres doivent être ajoutés au travail lorsqu'il est inclus dans une définition de flot de travaux.

Vous pouvez ajouter ou modifier des définitions de travail à partir des définitions de flot de travaux. Les modifications apportées aux définitions de travail dans les définitions de flot de travaux sont prises en compte dans les définitions de travail stockées dans la base de données. En d'autres termes, si vous modifiez la définition de travail de job1 dans la définition de flot de travaux js1 et si job1 est également utilisé dans le flot de travaux js2, la définition de job1 dans la définition de js2 est modifiée en conséquence.

Remarque : Les mots clés saisis incorrectement dans les définitions de travail entraînent le stockage de définitions de travaux tronquées dans la base de données. En fait, le mot clé erroné est considéré comme externe à la définition de travail et il est interprété comme le nom de travail d'une définition de travail supplémentaire. Généralement, cette mauvaise interprétation entraîne également une erreur de syntaxe ou une erreur de définition de travail inexistant pour la définition de travail supplémentaire.

Soyez vigilant dans le cas où un alias a été affecté à un travail. Vous pouvez décider d'utiliser un nom différent pour faire référence à une instance de travail particulière au sein d'un flot de travaux, mais le nom de l'alias ne doit pas entrer en conflit avec le nom d'un autre travail dans le même flot de travaux. Si une définition de travail est renommée, les travaux portant le même nom modifient leur nom conformément au nom de la définition de travail. Les exemples ci-dessous vous permettront de comprendre le comportement des travaux lorsque le nom de la définition de travail est modifié :

Tableau 28. Exemples : changement de nom de la définition de travail

Noms de définition de travail initiaux dans le flot de travaux	Changement de nom de la définition de travail	Résultat
SCHEDULE WKS#JS : FTA1#A FTA1#B as C FIN	Renommez le travail A en D	SCHEDULE WKS#JS : FTA1#D FTA1#B as C FIN
	Renommez le travail B en D	SCHEDULE WKS#JS : FTA1#A FTA1#D as C FIN
	Renommez le travail A en C	Une erreur se produit lors du changement de nom du travail A en C car le travail C existe déjà comme alias du travail B.

Pour savoir comment écrire des définitions de flots de travaux, voir «Définition de flot de travaux», à la page 226.

Arguments

poste_de_travail#

Indique le nom du poste de travail ou de la classe de postes de travail sur lequel le travail est exécuté. La valeur par défaut est le poste de travail spécifié pour *defaultws* lors du démarrage de la session de **composer**.

Pour plus d'informations sur le démarrage d'une session de composer, voir «Exécution du programme Composer», à la page 324. Le signe dièse (#) est un délimiteur obligatoire. Si une classe de postes de travail est indiquée, celle-ci doit correspondre à la classe de postes de travail d'un flot de travaux contenant le travail.

Si vous définissez un travail qui gère un travail Workload Broker, spécifiez le nom du poste de travail sur lequel le poste de travail Workload Broker est installé. Avec le poste de travail Workload Broker, IBM Workload Scheduler peut soumettre un travail dans l'environnement Dynamic Workload Broker grâce à la soumission de travail dynamique.

nomtravail

Indique le nom du travail. Il doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut contenir jusqu'à 40 caractères.

scriptname *nom_fichier*

Indique le nom du fichier exécuté par le travail. Utilisez le mot clé **scriptname** pour exécuter les travaux UNIX et Windows. Pour un fichier exécutable, entrez le nom du fichier et les éventuels arguments et options. La longueur du *nom_fichier* additionnée à la longueur de la *Condition de succès* (du mot clé **rccondsucc**) ne doit pas dépasser 4095 caractères. Vous pouvez également utiliser des paramètres IBM Workload Scheduler.

Utilisez cet argument pour définir des travaux IBM Workload Scheduler standard.

Voir «Utilisation de variables et de paramètres dans les définitions de travail», à la page 200 pour plus d'informations.

Pour les travaux Windows, inclut les extensions de fichier. Les noms UNC sont admis. Ne spécifiez pas de fichiers résidant sur des unités mappées.

Si vous définissez un travail qui gère un travail Workload Broker, spécifiez le nom du travail Workload Broker. De plus, vous pouvez spécifier des variables et le type d'affinité existant entre le travail IBM Workload Scheduler et le travail Workload Broker à l'aide de la syntaxe indiquée dans la liste ci-dessous. Pour identifier un travail avec affinité à l'aide :

Nom de travail IBM Workload Scheduler

```
jobName [-var var1Name=var1Value,...,varNName=varNValue]
[-twAffinity jobname=twJobName]
```

de l'ID du travail Dynamic Workload Broker

```
jobName [-var var1Name=var1Value,...,varNName=varNValue]
[-affinity jobid=jobid]
```

de l'alias du travail Dynamic Workload Broker

```
jobName [-var var1Name=var1Value,...,varNName=varNValue]
[-affinity alias=alias]
```

Pour plus d'informations, voir *IBM Workload Scheduler : Planification dynamique de la charge de travail*.

Si le chemin d'accès du fichier ou le nom du fichier de l'argument **scriptname** contient des espaces, la chaîne toute entière doit être placée entre "\"" et "\" " comme illustré ci-dessous :

```
scriptname "\"C:\Program Files\tws\myscript.cmd\""
```

Si des caractères spéciaux sont inclus, autres que des barres obliques (/) et des barres obliques inversées (\), la chaîne toute entière doit être placée entre guillemets (").

Le travail échoue si le script spécifié dans l'option **scriptname** est introuvable ou ne possède pas de droit d'exécution. Il s'interrompt si le script qui est introuvable ou ne possède pas de droit d'exécution comprend des paramètres.

docommand *commande*

Indique une commande exécutée par le travail. Entrez une commande valide et les éventuels arguments et options placés entre guillemets ("). La longueur de la *commande* additionnée à la longueur de la *Condition de succès* (du mot clé **rccondsucc**) ne doit pas dépasser 4095 caractères. Vous pouvez également utiliser des paramètres IBM Workload Scheduler.

Utilisez cet argument pour définir des travaux IBM Workload Scheduler standard.

Le travail s'interrompt si le fichier spécifié avec l'option **docommand** est introuvable ou ne possède pas de droit d'exécution.

Pour plus d'informations, voir «Utilisation de variables et de paramètres dans les définitions de travail», à la page 200.

task *définition_travail*

Spécifie la syntaxe XML des types de travaux avec options avancées et des travaux reflet.

Pour définir des types de travaux standard, utilisez les arguments **docommand** ou **scriptname**. Cet argument s'applique uniquement aux postes de travail des types suivants :

- agent
- pool
- d-pool
- rem-eng

La syntaxe du travail dépend du travail que vous définissez.

Pour obtenir la liste complète des types de travail pris en charge, voir «Création de définitions de travaux avancés», à la page 581.

streamlogon *nom_utilisateur*

Nom d'utilisateur sous lequel le travail s'exécute. Cet attribut est obligatoire lorsque **scriptname** ou **docommand** est spécifié. Ce nom peut comporter jusqu'à 47 caractères. Si ce nom comporte des caractères spéciaux, il doit être placé entre guillemets ("). Indiquez un utilisateur qui peut se connecter au poste de travail sur lequel le travail s'exécute. Vous pouvez également utiliser des paramètres IBM Workload Scheduler.

Voir «Utilisation de variables et de paramètres dans les définitions de travail», à la page 200 pour plus d'informations.

Pour les travaux Windows, l'utilisateur doit également posséder une définition d'utilisateur.

Pour obtenir des informations relatives aux besoins de l'utilisateur, voir «Définition d'utilisateur», à la page 200.

Si vous définissez un travail qui gère un travail de courtier de charge de travail dynamique, spécifiez le nom de l'utilisateur que vous avez utilisé pour installer Dynamic Workload Broker.

Le travail échoue si l'utilisateur spécifié dans l'option **streamlogon** n'existe pas.

description "*description*"

Fournit une description du travail. Ce texte doit être délimité par des guillemets. Le nombre maximum de caractères autorisés est 120.

tasktype *type_tâche*

Indique le type de travail. Il peut prendre l'une des valeurs suivantes :

UNIX Pour les travaux qui s'exécutent sur les plateformes UNIX.

WINDOWS

Pour les travaux qui s'exécutent sur les systèmes d'exploitation Windows.

OTHER

Pour les travaux qui s'exécutent sur des agents étendus. Voir *IBM Workload Scheduler for Applications - Guide d'utilisation* pour plus d'informations sur les types de tâche personnalisées pour les applications prises en charge par le fournisseur.

COURTIER

Pour les travaux qui gèrent le cycle de vie d'un travail Dynamic Workload Broker. Reportez-vous à *IBM Workload Scheduler : Planification dynamique de la charge de travail* pour des informations sur l'utilisation de Dynamic Workload Broker.

Lorsque vous définissez un travail, IBM Workload Scheduler enregistre son type dans la base de données sans effectuer d'autres vérifications. Cependant, quand le travail est soumis, IBM Workload Scheduler vérifie le système d'exploitation sur le poste de travail cible et définit le type de travail en conséquence.

interactive

Si vous définissez un travail qui gère un travail Dynamic Workload Broker, ignorez cet argument. Spécifiez que le travail s'exécute de façon interactive sur votre bureau. Cette fonction est disponible uniquement dans les environnements Windows.

succoutputcond *Nom_Condition "Valeur_Condition"*

Condition qui, lorsque satisfaite, désigne un travail comme terminé avec succès, et lui attribue le statut SUCC. La condition est utilisée lorsque vous souhaitez qu'un travail successeur ne s'exécute qu'une fois le travail ou flot de travaux prédécesseur terminé avec succès. Ces conditions permettent également de définir des flux alternatifs dans un flot de travaux à partir d'un travail ou d'un flot de travaux prédécesseur. Le travail successeur est déterminé en fonction des conditions que le travail ou flot de travaux prédécesseur satisfait.

Lorsque le prédécesseur est un flot de travaux, la dépendance conditionnelle ne peut être qu'une condition de statut, comme suit : abend, succ et suppr. Le travail successeur s'exécute lorsque le statut du flot de travaux prédécesseur satisfait le statut de travail indiqué à l'aide de ces arguments. Vous pouvez définir un statut, une combinaison de statuts, ou l'ensemble des statuts. Lorsque vous indiquez plusieurs statuts ou noms de condition, séparez-les d'une barre verticale (|). Vous pouvez définir autant de conditions de sortie à satisfaire que vous le souhaitez. La condition peut s'exprimer de la manière suivante :

Code retour

Sur les postes de travail tolérants aux pannes et les postes de travail d'agents dynamiques exclusivement, vous pouvez définir le code retour qui déterminera si un travail s'est correctement terminé. Les codes retours de travail peuvent être définis de plusieurs façons :

Expression de comparaison

La syntaxe à utiliser pour définir un code retour de travail est la suivante :

(RC opérateur opérande)

RC Mot clé RC.

opérateur

Opérateur de comparaison. Il peut prendre l'une des valeurs suivantes :

Tableau 29. Opérateurs de comparaison

Exemple	Opérateur	Description
RC<valeur	<	Inférieur à
RC<=valeur	<=	Inférieur ou égal à
RC>valeur	>	Supérieur à
RC>=valeur	>=	Supérieur ou égal à
RC=valeur	=	Egal à

Tableau 29. Opérateurs de comparaison (suite)

Exemple	Opérateur	Description
RC!=valeur	!=	Différent de
RC<>valeur	<>	Différent de

opérande

Un entier entre -2147483647 et 2147483647.

Par exemple, vous pouvez définir un travail réussi comme un travail qui prend fin avec un code retour inférieur ou égal à 3. Exemple :

```
succoutputcond UPDATE_OK "(RC <= 3)"
```

expression booléenne

Indique une combinaison logique d'expressions de comparaison. La syntaxe est la suivante :

```
expression_de_comparaison opérateur expression_de_comparaison
```

expression_de_comparaison

Cette expression est analysée de gauche à droite.

Vous pouvez utiliser des parenthèses pour attribuer une priorité à l'évaluation de l'expression.

opérateur

Opérateur logique. Il peut prendre l'une des valeurs suivantes :

Tableau 30. Opérateurs logiques

Exemple	Opérateur	Résultat
expr_a and expr_b	And	TRUE si expr_a ET expr_b sont TRUE.
expr_a or expr_b	Or	TRUE si expr_a OU expr_b est TRUE.
Not expr_a	Not	TRUE si expr_a n'est PAS TRUE.

Par exemple, vous pouvez définir un travail réussi comme un travail qui termine par un code retour inférieur ou égal à 3 ou par un code retour différent de 5 et inférieur à 10. Exemple :

```
succoutputcond "(RC<=3) OR ((RC!=5) AND (RC<10))"
```

Etat de travail

Sur les postes de travail tolérants aux pannes et les postes de travail d'agents dynamiques exclusivement, vous pouvez définir l'état de travail qui déterminera si un travail s'est correctement terminé.

Variable de sortie

Sur les postes de travail d'agents dynamiques exclusivement, qualifie un travail comme étant correctement terminé à l'aide de variables de sortie.

- Vous pouvez définir une condition de succès ou d'autres conditions pour le travail en analysant les propriétés de ce travail.

Par exemple, pour le cas d'un travail de transfert de fichiers, vous pouvez saisir l'expression suivante :

```
${this.File.1.Size}>0
```

si vous voulez qualifier un travail de transfert de fichier de réussi quand la taille du fichier transféré est supérieure à zéro.

- Vous pouvez définir une condition de succès ou d'autres conditions pour le travail en analysant les propriétés ou la sortie d'un autre travail dans le même flot de travaux.

Par exemple, pour un travail de transfert de fichiers, vous pouvez saisir l'expression suivante :

```
${this.NumberOfTransferredFiles}=
${job.DOWNLOAD.NumberOfTransferredFiles}
```

Si vous voulez qualifier un travail de transfert de fichier comme étant réussi quand le nombre de fichiers téléchargés vers l'amont dans le travail est égal au nombre de fichiers téléchargés vers l'aval dans un autre travail nommé DOWNLOAD, dans le même flot de travaux.

- Toutes les fonctions et les expressions Xpath (XML Path Language) sont prises en charge, pour les conditions décrites ci-dessus, dans la zone **succoutputcond** :
 - Comparaisons de chaîne (contient, commence par, correspond, etc.)
 - Manipulations de chaîne (concat, sous-chaîne, majuscule, etc.)
 - Comparaison numérique (=, !=, >, etc.)
 - Fonctions sur des valeurs numériques (abs, plancher, arrondi, etc.)
 - Opérateurs sur des valeurs numériques (addition, somme, division, etc.)
 - Opérateurs booléens

Contenu du journal de travail

Sur les postes de travail d'agents dynamique exclusivement, vous pouvez déterminer si un travail est correctement terminé en analysant le contenu du journal de travail.

Vous pouvez définir une condition de succès ou d'échec pour le travail en analysant la sortie de ce travail. Pour effectuer cette analyse, vous devez vérifier la variable `this.stdlist`. Par exemple, entrez l'expression suivante :

```
contains(${this.stdlist},"error")
```

La condition est satisfaite si le mot "error" est contenu dans la sortie du travail.

outputcond *Nom_Condition* "**Valeur_Condition**"

Condition de sortie qui, lorsqu'elle est satisfaite, détermine le travail successeur qui s'exécute. La condition s'exprime de la manière suivante : *Nom_Condition* "**Valeur_Condition**". Le format d'expression de la condition est identique à celui des conditions **succoutputcond**. Voici quelques exemples de conditions de sortie. Par exemple, pour créer une condition qui indique que le travail prédécesseur s'est terminé avec des erreurs, vous pouvez définir la condition de sortie de la manière suivante :

- **outputcond** STATUS_ERR1 "RC=1" crée une condition nommée STATUS_ERR1 selon laquelle si un travail prédécesseur s'est terminé avec un code retour = 1, alors il s'est terminé avec des erreurs.

- outputcond BACKUP_FLOW "RC != 5 and RC > 2" pour créer une condition nommée BACKUP_FLOW. Si le travail prédécesseur satisfait la condition, le travail successeur qui est lié à ce travail prédécesseur selon ces termes de dépendance s'exécute.

recovery

Options de reprise du travail. La valeur par défaut est **stop** sans aucun travail de reprise et aucune invite de reprise. Entrez l'une des options de reprise suivantes : **stop**, **continue** ou **rerun**. Elle peut être suivie d'un travail de reprise et/ou d'une invite de reprise.

stop En cas de fin anormale du travail, le traitement ne se poursuit pas avec le travail suivant.

continue

En cas de fin anormale du travail, le traitement se poursuit avec le travail suivant. Le travail n'est pas répertorié comme interrompu dans les propriétés du flot de travaux. Si aucun autre problème ne survient, le flot de travaux aboutit.

rerun En cas de fin anormale du travail, le traitement est réexécuté. Vous pouvez l'utiliser en association avec les options «after [poste_de_travail#]nom_travail», à la page 196 et «repeatevery hhmm», ou avec les options «after [poste_de_travail#]nom_travail», à la page 196 et «abendprompt "texte"», à la page 196. Vous pouvez, le cas échéant, spécifier une ou plusieurs des options suivantes et définir une séquence de réexécution :

same_workstation

Indique si le travail doit être réexécuté sur le même poste de travail que le travail parent. Cette option s'applique uniquement aux postes de travail en pool et en pool dynamique.

repeatevery hhmm

Indique la fréquence à laquelle IBM Workload Scheduler tente de réexécuter le travail ayant échoué. La valeur implicite est de 0. La valeur maximale prise en charge est de 99 heures et 59 minutes. Le compte à rebours pour le début des tentatives de réexécution est lancé après le travail parent ou une fois le travail de reprise éventuel terminé.

for nombre attempts

Indique le nombre maximal de tentatives de réexécution à effectuer. La valeur par défaut est de 1. La valeur maximale prise en charge est de 10 000 tentatives.

Si vous spécifiez un travail de reprise et que ce dernier et le travail parent échouent, les dépendances du travail parent ne sont pas libérées et ses successeurs éventuels ne sont pas exécutés. Si vous avez défini l'option rerun, la réexécution n'est pas effectuée. Dans ce cas, vous devez effectuer les étapes suivantes manuellement :

1. Confirmez manuellement que le travail de reprise est à l'état SUCC.
2. Nettoyez l'environnement en effectuant manuellement les opérations qui étaient exécutées par le travail de reprise.
3. Soumettez une réexécution du travail parent.

after [poste_de_travail#]nom_travail

Indique le nom d'un travail de reprise à lancer si le travail parent prend fin de façon anormale. Les travaux de reprise ne sont lancés qu'une seule fois pour chaque instance du travail parent qui prend fin de façon anormale.

Vous pouvez indiquer le poste du travail de reprise s'il est différent du poste du travail parent. Par défaut, il s'agit du poste du travail parent. Tous les travaux ne remplissent pas les conditions nécessaires pour permettre l'exécution de travaux de reprise sur un poste de travail différent. Suivez les directives ci-après :

- Si l'un des postes de travail est un agent étendu, il doit être hébergé par un gestionnaire de domaine ou un agent tolérant aux pannes pour lequel la valeur du mode **fullstatus** a été définie sur **on**.
- Le poste de travail du travail de reprise peut se trouver dans le même domaine que le poste de travail du travail parent ou dans un domaine supérieur.
- Si le poste de travail du travail de reprise est un agent tolérant aux pannes, la valeur **fullstatus** de cet agent doit être définie sur **on**.

abendprompt "texte"

Indique le texte d'une invite de reprise, figurant entre guillemets, qui doit être affiché dans le cas d'une fin anormale du travail. Ce texte peut comporter jusqu'à 64 caractères. S'il commence par un signe deux-points (:), l'invite s'affiche mais aucune réponse n'est requise pour la poursuite du processus. Si le texte commence par un point d'exclamation (!), l'invite s'affiche, mais il n'est pas enregistré dans le fichier journal. Vous pouvez également utiliser des paramètres IBM Workload Scheduler.

Pour plus d'informations, voir «Utilisation de variables et de paramètres dans les définitions de travail», à la page 200.

Le tableau 31 résume toutes les combinaisons possibles d'options de reprise et d'actions.

Ce tableau est basé sur les critères suivants d'un flot de travaux appelé sked1 :

- Le flot de travaux sked1 a deux travaux : job1 et job2.
- S'il est sélectionné pour job1, le travail de reprise est jobr.
- job2 dépend de job1 et ne démarre pas avant que job1 ne soit terminé.

Tableau 31. Options et actions de reprise

	Stop	Continuer	Rerun
Invite de reprise : Non Travail de reprise : Non	Une intervention est requise.	Exécutez job2.	Réexécutez job1. En cas de fin anormale de job1, émettez une invite. Si la réponse est <i>oui</i> , répétez ce qui précède. Si job1 réussit, exécutez job2.

Tableau 31. Options et actions de reprise (suite)

	Stop	Continuer	Rerun
Invite de reprise : Oui Travail de reprise : Non	Emettez une invite de reprise. Une intervention est requise.	Emettez une invite de reprise. Si la réponse est <i>oui</i> , exécutez job2.	Emettez une invite de reprise. Si la réponse est <i>oui</i> , exécutez de nouveau job1. En cas de fin anormale de job1, répétez ce qui précède. Si job1 réussit, exécutez job2.
Invite de reprise : Non Travail de reprise : Oui	Exécutez jobr. Si celui-ci se termine de façon anormale, une intervention est requise. S'il s'exécute avec succès, exécutez job2.	Exécutez jobr. Exécutez job2.	Exécutez jobr. Si jobr se termine de façon anormale, une intervention est requise. Si jobr réussit, réexécutez job1. En cas de fin anormale de job1, émettez une invite. Si la réponse est <i>oui</i> , répétez ce qui précède. Si job1 réussit, exécutez job2.
Invite de reprise : Oui Travail de reprise : Oui	Emettez une invite de reprise. Si la réponse est <i>oui</i> , exécutez jobr. Si celui-ci se termine de façon anormale, une intervention est requise. S'il s'exécute avec succès, exécutez job2.	Emettez une invite de reprise. Si la réponse est <i>oui</i> , exécutez jobr. Exécutez job2.	Emettez une invite de reprise. Si la réponse est <i>oui</i> , exécutez jobr. Si jobr se termine de façon anormale, une intervention est requise. Si jobr réussit, réexécutez job1. En cas de fin anormale de job1, répétez ce qui précède. Si job1 réussit, exécutez job2.

Remarques :

1. L'expression "une intervention est requise" signifie que job2 n'est pas libéré de sa dépendance par rapport à job1, et que c'est à l'utilisateur de le faire.
2. L'option de reprise **continue** se substitue à l'état de fin anormale, ce qui peut faire apparaître le flot de travaux contenant le travail en échec comme ayant réussi. Cela empêche d'éviter le report du flot de travaux au plan de production suivant.
3. Si vous sélectionnez l'option **rerun** sans indiquer d'invite de reprise, IBM Workload Scheduler génère sa propre invite.
4. Pour référencer un travail de reprise dans **conman**, utilisez le nom du travail d'origine (job1 dans le scénario ci-dessus et non jobr). Un seul travail de reprise est exécuté par fin anormale.

Exemples

L'exemple suivant présente un fichier comportant deux définitions de travail :

```
$jobs
cpu1#g11
    scriptname "/usr/acct/scripts/g11"
```

```

streamlogon acct
description "general ledger job1"
bkup
scriptname "/usr/mis/scripts/bkup"
streamlogon "^mis^"
recovery continue after recjob1

```

L'exemple ci-dessous illustre la définition du travail IBM Workload Scheduler TWSJOB qui gère le travail Workload Broker broker_1 s'exécutant sur le même agent Workload Broker que TWSJOB2 :

```

ITDWBAGENT#TWSJOB
SCRIPTNAME "broker_1 -var var1=name,var2=address
-twsaffinity jobname=TWSJOB2"
STREAMLOGON brkuser
DESCRIPTION "Added by composer."
TASKTYPE BROKER
RECOVERY STOP

```

L'exemple suivant montre comment définir un travail qui est affecté à un pool dynamique d'agents UNIX et exécute le script df :

```

DPOOLUNIX#JOBDEF7
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition
  xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>df</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP

```

L'exemple suivant montre comment définir un travail qui est affecté à un pool dynamique d'agents Windows et exécute le script dir :

```

DPOOLWIN#JOBDEF6
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition
  xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>dir</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP

```

L'exemple suivant montre comment définir un travail qui est affecté aux agent NC115084 et exécute le script dir :

```

NC115084#JOBDEF3
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition
  xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>dir</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>

```

```

    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP

```

L'exemple suivant montre comment définir un travail qui est affecté à un pool d'agents UNIX et exécute le script défini dans la balise script :

```

POOLUNIX#JOBDEF5
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jsdl:jobDefinition
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
    xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
      <jsdle:executable interactive="false">
        <jsdle:script>#!/bin/sh
sleep 60
dir</jsdle:script>
      </jsdle:executable>
    </jsdl:application>
  </jsdl:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP

```

L'exemple suivant montre comment définir un travail qui est affecté à un pool d'agents Windows et exécute le script défini dans la balise script :

```

POOLWIN#JOBDEF4
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jsdl:jobDefinition
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
    xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
    <jsdl:application name="executable">
      <jsdle:executable interactive="false">
        <jsdle:script>ping -n 120 localhost</jsdle:script>
      </jsdle:executable>
    </jsdl:application>
  </jsdl:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Utilisation de variables et de paramètres dans les définitions de travail

Une variable est un objet de planification qui fait partie d'une table de variables et qui est défini dans la base de données IBM Workload Scheduler. Elle peut être utilisée par tous les agents du domaine tant que les utilisateurs peuvent accéder au fichier de sécurité.

Un paramètre est défini et utilisé en local sur un agent (avec la commande d'utilitaire parms).

Les utilisations et restrictions suivantes s'appliquent aux variables et aux paramètres des définitions de travail :

- Des variables et des paramètres sont admis dans les valeurs des mots clé **streamlogon**, **scriptname**, **docommand** et **abendprompt**.
- Une variable ou un paramètre peut correspondre à une chaîne entière ou partielle.
- Plusieurs variables et paramètres sont admis dans une seule zone.
- Les noms de variables doivent figurer entre carets (^) et la chaîne entière entre guillemets. Assurez-vous que les caractères caret ne sont pas précédés d'une barre oblique inversée dans la chaîne. Si nécessaire, incluez la barre oblique inversée dans la définition de la variable ou du paramètre.
- Sous UNIX, les noms de paramètres doivent figurer entre des apostrophes (') et la chaîne entière entre guillemets.
- Voir «Définition des variables et des paramètres», à la page 206 pour plus d'informations et d'exemples.

Dans l'exemple ci-dessous, une variable nommée **mis** est utilisée dans la valeur **streamlogon** :

```
$jobs
cpu1#bkup
    scriptname "/usr/mis/scripts/bkup"
    streamlogon "^mis^"
    recovery continue after recjob1
```

Pour d'autres exemples, voir «Définition des variables et des paramètres», à la page 206.

Pour plus d'informations sur la transmission de variables entre des travaux appartenant au même flot de travaux, voir «Transmission de variables entre travaux», à la page 650.

Définition d'utilisateur

Les noms d'utilisateur utilisés comme valeur **streamlogon** dans les définitions de travail Windows doivent posséder des définitions utilisateur. Cette règle n'est pas obligatoire pour les utilisateurs qui exécutent des travaux sur d'autres systèmes d'exploitation. Si vous utilisez destypes de travaux avec options avancées, vous pouvez exploiter ces valeurs quel que soit le système d'exploitation. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204.

Remarque : Si l'option globale enAddUser vaut "yes", la définition d'utilisateur est automatiquement ajoutée au plan après la création ou la modification de la définition d'utilisateur dans la base de données.

Syntaxe

Chaque définition d'utilisateur utilise le format et les arguments suivants :

```
username[poste_de_travail#][domaine\]nom_utilisateur[@domaine_internet]  
password "mot_de_passe"  
end
```

Arguments

username

[poste_de_travail#]nom_utilisateur

Poste de travail

Indique le poste de travail sur lequel l'utilisateur peut exécuter des travaux. Le signe dièse est obligatoire. La valeur par défaut est le caractère blanc qui signifie que tous les postes de travail sont concernés.

nom_utilisateur

Indique le nom de l'utilisateur Windows. La zone *nom_utilisateur* peut comporter jusqu'à 47 caractères.

[poste_de_travail#]domaine\nom_utilisateur

Poste de travail

Indique le poste de travail sur lequel l'utilisateur peut exécuter des travaux. Le signe dièse est obligatoire. La valeur par défaut est le caractère blanc qui signifie que tous les postes de travail sont concernés.

domaine\nom_utilisateur

Indique le nom de l'utilisateur du domaine Windows. La zone *domaine\nom_utilisateur* peut comporter jusqu'à 47 caractères.

[poste_de_travail#]nom_utilisateur@domaine_internet

Poste de travail

Indique le poste de travail sur lequel l'utilisateur peut exécuter des travaux. Le signe dièse est obligatoire. La valeur par défaut est le caractère blanc qui signifie que tous les postes de travail sont concernés.

nom_utilisateur@domaine_interne

Indique le nom de l'utilisateur au format UPN (nom principal de l'utilisateur). Le format UPN est le nom d'un utilisateur système dans un format d'adresse électronique. Le nom d'utilisateur est suivi du symbole @, lui-même suivi du nom du domaine Internet auquel l'utilisateur est associé. La zone *nom_utilisateur@domaine_internet* peut comporter jusqu'à 47 caractères.

Remarque :

Si vous définissez un utilisateur pour les **systèmes d'exploitation Windows** :

- Les majuscules et les minuscules sont différenciées dans les noms d'utilisateur. En outre, l'utilisateur doit être autorisé à se connecter au

poste de travail sur lequel IBM Workload Scheduler lance les travaux et disposer de l'autorisation d'accès requise pour **se connecter par lot**.

- Si le nom d'utilisateur n'est pas unique, il est considéré comme étant respectivement un utilisateur local, un utilisateur de domaine ou un utilisateur de domaine sécurisé.

password

Indique le mot de passe de l'utilisateur. Le mot de passe peut comporter jusqu'à 31 caractères et doit figurer entre guillemets. Pour indiquer un mot de passe vide, utilisez deux guillemets consécutifs qui ne soient pas séparés par des espaces, "". Une fois une définition d'utilisateur enregistrée, il est impossible de lire le mot de passe. Les utilisateurs disposant des droits d'accès appropriés peuvent modifier ou supprimer un utilisateur, mais les informations relatives aux mots de passe ne sont jamais affichées.

Exemples

L'exemple suivant permet de définir quatre utilisateurs :

```
username joe
    password "okidoki"
end
#
username server#jane
    password "okitay"
end
#
username dom1\jane
    password "righto"
end
#
username jack
    password ""
end
#
username administrator@twsbvt.com
    password "internetpwd"
end
#
username serverA#dom1\jack
    password "righto"
end
#
username serverB#user1@twsbvt.com
    password "internetpwd"
end
#
```

Commentaires

Les mots de passe extraits avec la commande `composer extract` ont un usage limité. Lorsque vous exécutez la commande `composer extract` sur une définition d'utilisateur, le mot de passe est masqué avec le mot clé réservé "*****". Si vous essayez d'exécuter les commandes `composer import`, `replace` ou `modify` sur un mot de passe utilisateur extrait, le remplacement du mot de passe n'a aucun effet et l'ancien mot de passe est conservé. De plus, si vous tentez d'utiliser les commandes `composer create`, `new` ou `add` sur un utilisateur où le mot de passe correspond au mot clé réservé "*****", l'erreur suivante est renvoyée :

AWSJCL521E Le mot de passe spécifié pour l'utilisateur Windows "USER_NAME" n'est pas conforme aux exigences de règles de sécurité du mot de passe.

Notez que le mot clé réservé est une chaîne de 10 astérisques (*). Vous ne pouvez pas entrer une séquence de dix astérisques comme mot de passe mais vous pouvez la définir avec n'importe quel autre nombre d'astérisques.

Pour corriger ce problème, assurez-vous d'exécuter `composer extract` avec l'option `;password`.

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Utilisation des définitions d'utilisateur et streamlogon IBM Workload Scheduler

Sur les postes de travail Windows, vous devez spécifier les définitions d'utilisateur à l'aide de **composer** sous la forme `[poste_de_travail#]nom_utilisateur`. L'instance `[poste_de_travail#]nom_utilisateur` identifie de façon unique l'utilisateur Windows dans l'environnement IBM Workload Scheduler. Le nom du poste de travail est facultatif ; si vous ne l'indiquez pas, cela signifie que l'utilisateur nommé `nom_utilisateur` est défini sur tous les postes de travail Windows du réseau IBM Workload Scheduler. Si l'utilisateur nommé `nom_utilisateur` est défini uniquement sur certains postes de travail Windows du réseau IBM Workload Scheduler, vous devez créer, afin d'éviter les incohérences, une définition d'utilisateur au format `[poste_de_travail#]nom_utilisateur` pour chaque poste de travail Windows où l'utilisateur `nom_utilisateur` est défini.

Si vous planifiez un travail sur un agent, sur un pool ou un pool dynamique, le travail s'exécute avec l'utilisateur défini sur le pool ou le pool dynamique. Cependant, l'utilisateur doit exister sur tous les postes de travail du pool ou du pool dynamique où vous planifiez d'exécuter le travail.

Si vous définissez un travail à l'aide du programme **composer**, vous devez spécifier à la fois un poste de travail et lui affecter un nom de connexion correct. Le nom de connexion est simplement un nom d'utilisateur valide pour Windows, sans le nom du poste de travail. Par exemple, dans la définition de travail suivante :

```
$JOB
poste_de_travail#job01 docommand "dir"
streamlogon nom_utilisateur
```

la valeur de l'option `streamlogon` est `nom_utilisateur` et non `poste_de_travail#nom_utilisateur`.

Si vous utilisez la commande **altpass**, rappelez-vous que vous devez toutefois utiliser la définition d'utilisateur sous la forme

```
poste_de_travail#nom_utilisateur
```

Dans cette commande, vous pouvez omettre le nom du poste de travail seulement si vous modifiez son mot de passe sur la machine sur laquelle vous exécutez la commande.

Utilisateur de domaine sécurisé

Si IBM Workload Scheduler doit lancer des travaux pour un utilisateur de domaine sécurisé, respectez les instructions suivantes lors de la définition des comptes utilisateur. En supposant qu'IBM Workload Scheduler est installé dans le domaine Domaine1 pour le compte utilisateur maestro et que le compte utilisateur sue dans le domaine Domaine2 doit lancer un travail, les conditions suivantes doivent être remplies :

- Une sécurisation réciproque doit subsister entre Domaine1 et Domaine2.
- Dans Domaine1, sur les ordinateurs sur lesquels des travaux sont lancés, sue doit posséder le droit de se **connecter par lot**.
- Dans Domaine1, maestro doit correspondre à un utilisateur de domaine.
- Dans Domaine2, maestro doit détenir les droits pour **accéder à l'ordinateur à partir du réseau**.

Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées

Avec les types de travaux avec options avancées, quel que soit le système d'exploitation de l'agent dynamique qui exécutera le travail, vous pouvez fournir le nom d'utilisateur d'une définition d'utilisateur dans la section des droits d'accès du travail et résoudre la zone mot de passe lors de l'exécution avec la valeur de mot de passe stockée dans la base de données.

Par exemple, lorsque vous définissez le travail avec Dynamic Workload Console, entrez le nom d'utilisateur d'une définition d'utilisateur et cliquez sur les points de suspension (...) situés en regard de la zone du mot de passe pour afficher le widget Type de mot de passe :

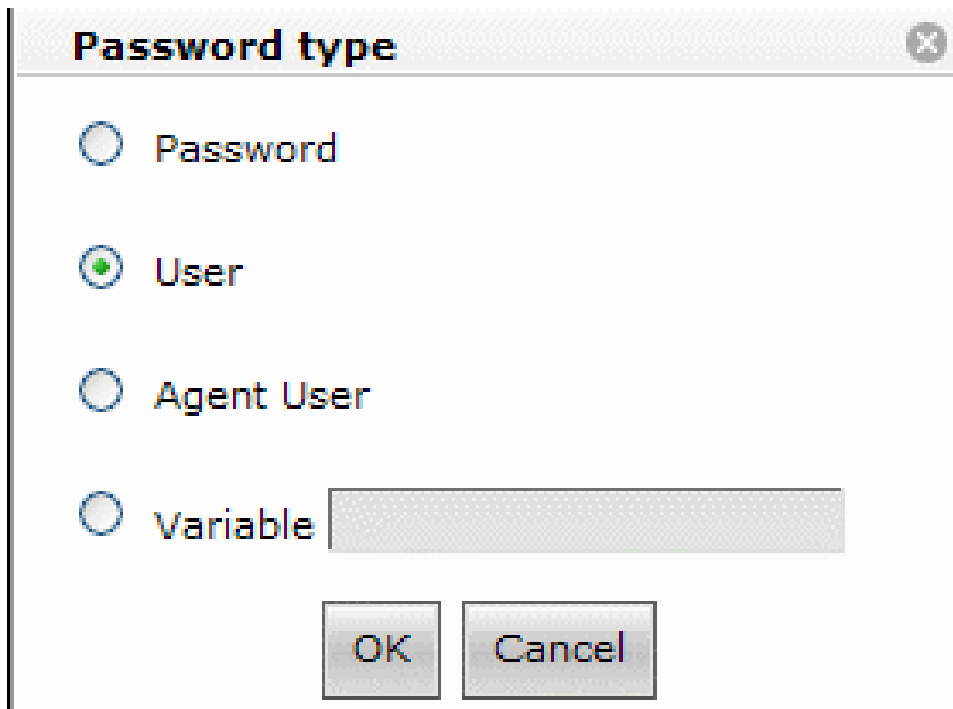


Figure 24. Définition d'utilisateur

où vous sélectionnez Utilisateur comme indiqué. Vous pouvez de la même façon coder cette option dans la section tâche (JSDL) de la définition de travail dans le compositeur. Pour plus d'informations, consultez les sections connexes.

Pour pouvoir utiliser cette option lorsque vous définissez un travail, vous devez être autorisé dans le fichier de sécurité avec le mot clé use pour le type d'objet userobj :

```
userobj access=use
```

IBM Workload Scheduler suit cette séquence lorsqu'il est appelé pour résoudre le nom d'utilisateur et le mot de passe lors de l'exécution :

- Si le poste de travail n'est pas spécifié (par exemple, `${password:myuser}`) :
 1. Recherche myuser sur le poste de travail qui exécute le travail qui applique une politique sensible à la casse.
 2. Recherche myuser sur le poste de travail qui exécute le travail qui applique une politique insensible à la casse.
 3. Recherche myuser sans poste de travail associé qui applique une politique sensible à la casse.
 4. Recherche myuser sans poste de travail associé qui applique une politique insensible à la casse.
- Si le poste de travail est spécifié (par exemple, `${password:agent#myuser}`) :
 1. Recherche myuser sur l'agent du poste de travail qui applique une politique sensible à la casse.
 2. Recherche myuser sur l'agent du poste de travail qui applique une politique insensible à la casse.
 3. Recherche myuser sans poste de travail associé qui applique une politique sensible à la casse.
 4. Recherche myuser sans poste de travail associé qui applique une politique insensible à la casse.
- Si le poste de travail est spécifié mais vide (par exemple, `${password:#myuser}`) :
 1. Recherche myuser sans poste de travail associé qui applique une politique sensible à la casse.
 2. Recherche myuser sans poste de travail associé qui applique une politique insensible à la casse.

Avertissement : Les définitions d'utilisateur manquent d'intégrité référentielle. Cela implique que, si une définition d'utilisateur référencée dans la section des données d'identification d'un type de travail avec des options avancées est modifiée ou supprimée, aucun message d'avertissement ou d'erreur n'est renvoyé jusqu'à l'exécution du travail.

Définition d'agenda

Un agenda est une liste de dates qui définissent si un flot de travaux s'exécute et le moment où il s'exécute. Chaque définition d'agenda utilise le format et les arguments suivants :

Syntaxe

\$calendar

```
nom_agenda ["description"]  
date [...]
```

```
[nom_agenda ...]
```

Arguments

nom_agenda

Nom de l'agenda. Ce nom peut comporter jusqu'à huit caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre.

"description"

Fournit une description de l'agenda. La description peut contenir jusqu'à 120 caractères alphanumériques. Elle doit figurer entre guillemets. Son nom peut comporter des caractères alphanumériques à condition de commencer par une lettre. Il peut contenir les caractères suivants : virgule (,), point (.), tiret (-), plus (+), apostrophe (') et égal (=). Il ne peut pas comporter les caractères suivants : guillemet (") (autres que ceux encadrant la description), deux-points (:), point-virgule(;) et la perluète (&).

date [...]

Indique une ou plusieurs dates, séparées par des espaces. Le format est *mm/jj/aa*.

Exemples

L'exemple ci-après permet de définir trois agendas intitulés *monthend*, *paydays* et *holidays* :

```
$calendar
monthend "Month end dates 1st half 2005"
    01/31/2005 02/28/2005 03/31/2005 04/30/2005 05/31/2005 06/30/2005
paydays
    01/15/2005 02/15/2005
    03/15/2005 04/15/2005
    05/14/2005 06/15/2005
holidays
    01/01/2005 02/15/2005 05/31/2005
```

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Définition des variables et des paramètres

Les variables et les paramètres sont des objets auxquels vous attribuez différentes valeurs.

Les variables et les paramètres sont utiles lorsque des valeurs changent en fonction des flots de travaux et des travaux. Les définitions de flot de travaux, de travail et d'invite qui les utilisent sont mises à jour automatiquement soit au début du cycle de production, soit lors de l'exécution du travail en fonction du format utilisé lorsque la variable est spécifiée.

Utilisez les variables et les paramètres en remplacement des valeurs répétitives lors de la définition d'invites, de travaux et de flots de travaux. Par exemple, si vous utilisez des variables pour les ouvertures de session utilisateur et les noms de fichiers script dans les définitions de travaux, ainsi que pour les dépendances de fichier et d'invite, vous utilisez des valeurs permettant une gestion centralisée dans la base de données du gestionnaire maître.

Lorsque les variables sont des objet de planification définis dans la base de données IBM Workload Scheduler et qu'elles peuvent être utilisées par tout utilisateur autorisé du domaine, les paramètres sont définis et utilisés en local sur des agents individuels.

Les sections suivantes décrivent en détail les variables et les paramètres.

Variables

Les variables sont définies en tant qu'objets de planification dans la base de données. Les variables peuvent être définies de manière individuelle avec la commande suivante :

\$parm

[*nom_table*.]*nom_variable* "*valeur_variable*"

...

Où :

nom_table

Nom de la table de variables contenant la nouvelle variable. Elle doit déjà être définie. Si vous n'indiquez pas de nom de table de variables, la variable est ajoutée à la table par défaut.

nom_variable

Nom de la variable. Ce nom peut comporter jusqu'à 64 caractères alphanumériques, y compris des tirets (-) et des traits de soulignement (_), et doit commencer par une lettre.

valeur

Valeur affectée à la variable. La valeur peut comporter jusqu'à 1024 caractères alphanumériques. N'incluez pas les noms d'autres variables.

Toutefois, le meilleur moyen de définir une variables est d'utiliser une «Définition de table de variables», à la page 211. Dans tous les cas, toutes les variables sont placées dans une table de variables. Si vous définissez une variable sans préciser le nom de la table de variables, elle est incluse dans la table de variables par défaut.

Vous pouvez utiliser des variables dans les définitions de travail et de flot de travaux. Ils sont résolus, c'est-à-dire, remplacés par la valeur qui leur a été affectée, lorsque le plan de production est généré ou étendu, ou que vous soumettez l'exécution d'un travail ou d'un flot de travaux. Le format utilisé pour spécifier une variable détermine également le moment où la variable est résolue avec une valeur. Les formats suivants peuvent être utilisés lors de la spécification d'une variable :

^*nom_variable*^

Spécifiez la variable à ce format si vous souhaitez qu'elle soit résolue lorsque le plan est généré ou étendu.

\${*nom_variable*}

Spécifiez la variable à ce format si vous souhaitez qu'elle soit résolue ou substituée lorsque l'exécution du travail ou du flot de travaux est soumise. Une option dans la définition de travail indiquant de résoudre les variables au moment de l'exécution du travail doit également être spécifiée. Si cette variable n'est présente que dans la table des variables par défaut, elle ne peut pas être résolue. Pour obtenir un exemple d'application de ce type de variable, voir la section «Exemples», à la page 210.

Pour plus d'informations sur la résolution des variables, voir «Résolution de variable», à la page 133.

Les noms de variables spécifiés dans ces définitions sont d'abord résolus pour les définitions de tables de variables, puis pour les paramètres locaux si les variables sont introuvables.

Lorsque vous spécifiez une variable, placez la chaîne contenant la variable entre guillemets.

Si la variable contient une partie d'un chemin, vérifiez que les accents circonflexes ne sont pas immédiatement précédés d'une barre oblique inversée (\). En effet, la séquence \^ pourrait être interprétée comme une séquence d'échappement et serait alors résolue par le programme d'analyse syntaxique comme un accent circonflexe. Si nécessaire, placez la barre oblique inversée dans la définition de la variable entre carets afin d'éviter une interprétation incorrecte. Par exemple, le tableau ci-dessous illustre le meilleur moyen de définir et d'utiliser une variable nommée MYDIR dans la table de variables par défaut :

Tableau 32. Traitement d'une barre oblique inversée dans la substitution de variable

Mauvais moyen	Bon moyen
1. Définissez la variable MYDIR de la manière suivante : \$PARM MYDIR "scripts"	1. Définissez la variable MYDIR de la manière suivante : \$PARM MYDIR "\scripts"
2. Utilisez-la de cette manière : job01 scriptname "c:\operid\^MYDIR\test.cmd"	2. Utilisez-la de cette manière : job01 scriptname "c:\operid^MYDIR^test.cmd"
3. Utilisez-la de cette manière : job01 scriptname "c:\operid\\${MYDIR}\test.cmd"	3. Utilisez-la de cette manière : job01 scriptname "c:\operid\${MYDIR}\test.cmd"

Cela est vrai pour toutes les commandes de ligne de commande, les interfaces graphiques et les API grâce auxquelles vous utilisez la substitution de variable.

Paramètres

Les paramètres locaux sont définis dans une base de données locale sur le poste de travail sur lequel est exécuté le travail qui les utilise. Pour les définir, n'utilisez pas cette commande **composer** mais la commande «parms», à la page 710.

Les paramètres locaux peuvent être utilisés dans les :

- JCL
- ouvertures de session
- dépendances d'invite
- dépendances de fichier
- invites de reprise

Un paramètre local est défini dans ces mots clés ou dans le script de travail appelé à l'aide de la syntaxe suivante :

'bin\parms PARAMETERNAME'

Les paramètres locaux sont résolus à l'aide des définitions stockées dans la base de données PARMS locale comme suit :

- Au moment de l'exécution, sur le poste de travail où le traitement du travail a lieu.
- Au moment de la soumission, sur le poste de travail sur lequel le travail ou le flot de travaux est soumis à partir de la ligne de commande **conman**. Le tableau 33 récapitule les commandes **submit** dans lesquelles vous pouvez utiliser les paramètres.

Tableau 33. Mots clés auxquels peuvent être associés des paramètres locaux dans les commandes **submit**

Mot clé	submit docommand (commande sbd)	submit file (commande sbf)	submit job (commande sbj)	submit job stream (commande sbs)
abendprompt	✓	✓	✓	
scriptname		✓		
docommand	✓			
logon	✓	✓		
opens	✓	✓	✓	✓
prompt	✓	✓	✓	✓

Pour plus d'informations sur la soumission de travaux et de flots de travaux en production à partir de la ligne de commande **conman**, voir Chapitre 12, «Gestion des objets dans le plan - conman», à la page 411.

Sous UNIX, lorsque vous définissez un travail ou un flot de travaux dans la base de données, vous devez placer la chaîne

chemin/parms nom_paramètre

entre les caractères ' ' pour que le paramètre soit résolu au moment de l'exécution sur le poste de travail, même si un paramètre ayant le même nom est défini en tant que paramètre global dans la base de données IBM Workload Scheduler. Par exemple, si vous ajoutez la définition de travail suivante à la base de données :

```
$jobs
myjob
  docommand "ls ^MYDIR^"
  streamlogon "^MYUSER^"
```

et que deux paramètres nommés MYDIR et MYUSER sont définis dans la base de données, les deux paramètres sont résolus, lors de la création ou de l'extension du plan de production, à l'aide des définitions contenues dans la base de données et leurs valeurs correspondantes sont contenues dans le fichier Symphony. Si vous définissez myjob dans la base de données comme suit :

```
$jobs
myjob
  docommand "ls 'bin/parms MYDIR'"
  streamlogon "'bin MYUSER'"
```

la seule action effectuée lors de la création ou de l'extension du plan de production sur les deux paramètres dans la définition de myjob est la suppression des caractères ' '. Les paramètres sont contenus dans le fichier Symphony, non résolus, puis ils sont résolus au moment de l'exécution en local sur le poste de travail cible à l'aide de la valeur stockée dans la base de données PARMS.

Exemples

Deux paramètres, `glpah` et `gllogon`, sont définis de la manière suivante :

```
$parm
glpath    "/glfiles/daily"
gllogon   "gluser"
```

Dans un flot de travaux intitulé `glsched`, les paramètres `glpath` et `gllogon` sont utilisés dans un travail intitulé `gljob2` :

```
schedule glsched on weekdays
:
gljob2
  scriptname "/usr/gl^glpath^"
  streamlogon "^gllogon^"
  opens "^glpath^/datafile"
  prompt " :^glpath^ started by ^gllogon^"
end
```

Voici un exemple de variable utilisée avec le mot clé **docommand** :

```
docommand "ls ^MY_HOME^"
```

L'exemple ci-dessous décrit comment la spécification de variables dans différents formats permet aux variables d'avoir plusieurs valeurs, étant donné qu'elles sont résolues à différents moments. Il explique également comment les variables peuvent être transmises d'un travail à un autre dans un flot de travaux. La variable `SWITCH_VAR` est définie dans la table de variable `STATETABLE` avec la valeur initiale par défaut `on`. Le travail `UPDATE1` est chargé de modifier la valeur de la variable `SWITCH_VAR` dans la table de variable `STATETABLE` en `off`. Le flot de travaux `PROCJS` contient deux travaux identiques, `PROC1` et `PROC2`, dans lesquels la variable `SWITCH_VAR` a été spécifiée dans deux formats différents. Le premier place la variable entre carets (^) (`^nom_var^`) et le second utilise le format `${nom_var}` :

```
<jsdle:script>echo ^SWITCH_VAR^:${SWITCH_VAR}</jsdle:script>
```

L'ordre dans lequel ces travaux s'exécutent est le suivant :

```
SCHEDULE NC117126#PROCJS
VARIABLE STATETABLE
:
NC117126_1# PROC1

NC117126_1# PROC2
FOLLOWS UPDATE1

NC117126_1# UPDATE1
FOLLOWS PROC1
FIN
```

Lorsque le flot de travaux est ajouté au plan, `SWITCH_VAR`, défini dans `PROC1` et `PROC2`, prend immédiatement la valeur par défaut `on` attribuée dans la table de variable. Lorsque le flot de travaux est soumis pour être exécuté, le premier travail à être soumis est `PROC1` et la variable définie comme `SWITCH_VAR` est résolue en `on` de sorte que les variables du travail `PROC1` soient résolues comme suit :

```
<jsdle:script>echo on:onjsdle:script>echo on:on</jsdle:script>
```

`UPDATE1` paramètre ensuite la valeur de `SWITCH_VAR` dans la table de variable sur `off` de sorte que, lorsque `PROC2` s'exécute, les variables soient résolues comme suit :

```
<jsdle:script>echo on:onjsdle:script>echo on:off</jsdle:script>
```


La variable spécifiée comme `^SWITCH_VAR^` dans le travail conserve la valeur on car les variables à ce format sont résolues lorsque le flot de travaux est ajouté au plan et ne sont pas actualisées lorsque l'exécution du travail est soumise. La variable spécifiée au format `$(SWITCH_VAR)`, précédemment définie sur on est désormais mise à jour avec la nouvelle valeur dans la table de variable off.

Création d'une définition de variable à l'aide de Dynamic Workload Console

Pour créer une définition de variable dans Dynamic Workload Console, vous devez l'ajouter dans une définition de table de variable :

1. Cliquez sur **IBM Workload Scheduler**→**Charge de travail**→**Conception**→**Créer des définitions de charge de travail**
2. Sélectionnez un nom de moteur et cliquez sur **Go (Aller)**
3. Ouvrez en mode édition une table de variables existante à partir du panneau Ouverture rapide, ou créez une nouvelle table de variables comme décrit dans «Définition de table de variables»
4. Dans le panneau Propriétés - Table de variables, cliquez sur l'onglet Variables et ajoutez de nouvelles définitions de variables en cliquant sur l'icône "+" (Ajouter) et en indiquant les noms et valeurs des variables.

Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

Définition de table de variables

Une table de variables est un objet regroupant plusieurs variables. Tous les paramètres globaux (désormais appelés *variables*) que vous utilisez dans une planification de charge de travail sont contenues dans au moins une table de variables. Il existe deux méthodes pour définir des variables :

- Lors de la définition d'une table de variables de la manière présentée ici. Il s'agit de la méthode recommandée.
- Définissez-les individuellement avec la commande **composer \$parm** au format `[nom_table.]nom_variable "valeur_variable"`. Si vous n'indiquez pas de nom de table, la nouvelle variable est placée dans la table de variables par défaut.

Vous n'êtes pas obligé de créer des tables de variables pour créer et utiliser des variables. Vous pouvez ne jamais créer de table et ne jamais en utiliser de manière explicite. Dans tous les cas, le planificateur fournit une table par défaut et à chaque fois que vous créez et gérez une variable sans préciser le nom de la table, il la stocke ou la recherche ici.

Vous pouvez définir plusieurs variables avec le même nom mais différentes valeurs, et les placer dans des tables différentes. Grâce à ces tables, vous attribuez différentes valeurs à la même variable, et réutilisez donc la même variable dans les définitions de travail et lors de la définition des invites et des dépendances de fichier. Les tables de variables peuvent être affectées au niveau des cycles d'exécution, des flots de travaux et des postes de travail.

Les tables de variables peuvent s'avérer particulièrement utiles pour les définitions de travail, notamment lorsque vous utilisez une définition de travail en tant que modèle pour un travail appartenant à plusieurs flots de travaux. Par exemple, vous pouvez affecter différentes valeurs à la même variable et réutiliser la même définition de travail dans plusieurs flots de travaux.

Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

Syntaxe

```
variable nom_table
[description "description"]
[isdefault]
members
[nom_variable "valeur_variable"]
...
[nom_variable "valeur_variable"]
end
```

Arguments

variable *nom_table*

Nom de la table de variables. Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut contenir jusqu'à 80 caractères.

description "*description_table*"

Description de la table de variables. Ce texte doit être délimité par des guillemets. La description peut contenir jusqu'à 120 caractères alphanumériques. Il ne peut pas comporter les caractères suivants : guillemet (") (autres que ceux encadrant la description), deux-points (:), point-virgule(;) et la perluète (&).

isdefault

Si ce mot clé est indiqué, il s'agit de la table par défaut. Vous ne pouvez pas indiquer plusieurs tables comme table par défaut. Si vous indiquez qu'une table de variables est celle par défaut, la table de variables par défaut en cours ne l'est plus. Lors de la migration de la base de données à partir d'une version précédente, le produit crée la table de variables par défaut avec toutes les variables déjà définies.

members *nom_variable* "*valeur_variable*"

Liste des variables et de leurs valeurs séparées par des espaces. Ce nom peut comporter jusqu'à 64 caractères alphanumériques, y compris des tirets (-) et des traits de soulignement (_), et doit commencer par une lettre. La valeur peut comporter jusqu'à 1024 caractères alphanumériques. Une valeur doit être délimitée par des guillemets.

Exemple

L'exemple ci-dessous illustre une table de variables et son contenu.

```
VARIABLE TEST1
MEMBERS
  DEVBATCH "DOMD\IMSBATCH\SAME"
  PARAM_01 "date"
  PARAM_02 "root"
  PARM_01 "PARM_001"
  PRPT_02 "PARM_002"
  PRPT_03 "PARM_003"
  PRPT_04 "PARM_004"
  PRPT_05 "PARM_005"
  SAME17 "test/for/variable with samename > variable/table"
  SLAV10 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82STGGDWHSCART"
  SLAV11 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82CDMGALLBCV"
  SLAV12 "/nfsdir/billingprod/crmb/MAESTRO_JOB/AG82CDMGRISCTRAF"
```

```

SLAV13 "/opt/crm/DWH_OK/Business_Copy_ok"
SLAV14 "/opt/crm/DWH_OK/DW_Canc_Cust_Gior_ok_"
TRIGGER "/usr/local/samejobtriggers"
VFILE2 "testforvarwithsamename2.sh"
VUSER2 "same_user2"
WRAPPER "/usr/local/sbin/same/phi_job.ksh"
FIN

```

Considérations relative au fichier de sécurité

Du point de vue des autorisations du fichier de sécurité, les droits d'intervention sur les entrées de variables contenues dans une table de variables dépendent des droits globaux accordés à la table de variables, comme indiqué dans le tableau suivant.

Tableau 34. Mot clé d'accès nécessaire de la table de variables dans le fichier de sécurité (objet variable) et actions autorisées sur ses entrées de variable.

Mot clé d'accès au fichier de sécurité requis sur la table de variables	Action autorisée sur les entrées de variable répertoriées
Modifier	Add
	Suppression
	Modifier
	Renommer
Afficher	Afficher
Déverrouiller	Déverrouiller

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Définition d'invite

Une invite est un texte s'affichant sous forme de message et interrompant le traitement d'un travail ou flot de travaux jusqu'à ce qu'une réponse affirmative soit envoyée (la réponse peut être indiquée manuellement par l'opérateur ou automatiquement par une action de règle d'événement). Une fois la réponse à l'invite obtenue, le traitement reprend. Les invites peuvent servir de dépendances dans les travaux ou les flots de travaux. Vous pouvez utiliser des variables dans les invites.

Il existe deux types d'invites :

Invites locales ou non nommées

Une invite non nommée est une invite définie dans une définition de travail ou de flot de travaux à l'aide du mot clé **prompt**. Aucun nom ne lui est affecté et elle n'est pas définie en tant qu'objet de planification dans la base de données. Elle ne peut donc pas être utilisée par d'autres travaux ou flots de travaux.

Invites globales ou nommées

Une invite globale est définie dans la base de données en tant qu'objet de

planification, identifiée par un nom unique et utilisable par n'importe quel travail ou flot de travaux. Les variables des invites globales sont toujours résolues à l'aide de la table de variables par défaut. En effet, l'invite globale est utilisée par tous les travaux et flots de travaux de manière à n'utiliser qu'une seule valeur pour la résolution des variables.

La présente section décrit les invites globales. Pour plus d'informations sur les invites locales, voir «Travail», à la page 940 et «Définition de flot de travaux», à la page 226.

Remarque : Chaque fois que le travail **JnextPlan** est exécuté, les définitions d'invite globale ou prédéfinie sont réinitialisées.

Syntaxe

\$prompt

promptname "[[: | !]texte"

[*nom_invite* ...]

Arguments

nom_invite

Nom de l'invite. Ce nom peut comporter jusqu'à 8 caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre.

texte

Fournit le texte de l'invite. Le texte de l'invite peut contenir jusqu'à huit caractères alphanumériques. Selon le caractère précédant son libellé, l'invite peut se comporter de manières différentes :

- S'il commence par un signe deux-points (:), l'invite s'affiche mais aucune réponse n'est requise pour la poursuite du processus.
- Si le texte commence par un point d'exclamation (!), l'invite s'affiche, mais il n'est pas enregistré dans le fichier journal.

Un ou plusieurs paramètres peuvent être utilisés pour une partie ou l'ensemble de la chaîne de texte de l'invite. Si un paramètre est utilisé, celui-ci doit être placé entre carets (^). Pour un exemple, voir «Définition des variables et des paramètres», à la page 206.

Remarque : Dans les invites locales, les carets (^) ne désignant pas de paramètre doivent être précédés d'une barre oblique inversée (\) pour ne pas provoquer d'erreurs. Dans les invites globales, il n'est pas nécessaire de faire précéder les carets d'une barre oblique inversée. Pour créer une ligne, vous pouvez inclure une barre oblique inversée suivie de la lettre n (\n) dans le texte.

Exemples

L'exemple suivant permet de définir trois invites :

```
$prompt
  prmt1 "prêt pour job4? (y/n)"
  prmt2 " :job4 lancé"
  prmt3 "!poursuivre?"
```

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Définition de ressource

Les ressources correspondent à des ressources de planification physiques ou logiques qui peuvent servir de dépendances pour les travaux et les flots de travaux. Les ressources ne peuvent être utilisées comme dépendances que par les travaux ou les flots de travaux exécutés sur le poste de travail où la ressource est définie.

En raison du mécanisme de résolution des dépendances de ressources, une dépendance de ressource au niveau du flot de travaux peut être considérée comme 'locale' (et son utilisation est prise en charge) et non comme 'globale', lorsqu'à la fois le flot de travaux et tous ses travaux sont définis sur le même poste de travail que la ressource. Cependant, un agent standard et son hôte peuvent faire référence aux mêmes ressources. Pour plus d'informations, voir le mot clé «needs», à la page 265.

Syntaxe

\$resource

poste_de_travail#nom_ressource unités ["description"]

[poste_de_travail#nom_ressource ...]

Arguments

Poste de travail

Indique le nom du poste de travail ou de la classe de postes de travail sur lequel la ressource est utilisée.

nom_ressource

Indique le nom de la ressource. Ce nom peut comporter jusqu'à huit caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre.

unités Indique le nombre d'unités de ressources disponibles. Les valeurs peuvent être comprises entre **0** et **1024**.

"description"

Fournit une description de la ressource. La description peut contenir jusqu'à 120 caractères alphanumériques. Elle doit figurer entre guillemets.

Les unités de ressource impliquées dans les dépendances needs pour un travail ou un flot de travaux restent occupées jusqu'à ce que le travail ou le flot de travaux soit terminé (qu'il ait ou non abouti). Les unités de ressource sont libérées dès que le travail ou le flot de travaux est terminé.

Lorsque plusieurs travaux et flots de travaux dépendent de la même ressource et que le nombre d'unités actuellement disponibles n'est pas suffisant pour tous ces travaux et flots, la ressource est affectée en fonction de la priorité du travail ou du

flot de travaux. Le statut d'un travail ou d'un flot de travaux devient READY (Prêt) dès que toutes ses dépendances sont résolues. Si la quantité limite d'UC définie sur le poste de travail ne lui permet pas de s'exécuter pour le moment, il attend à l'état READY. La seule exception à ce comportement est lorsque le travail ou le flot de travaux a le statut GO ou HI, auquel cas il démarre, quelle que soit la valeur définie pour la quantité limite d'UC.

Exemples

L'exemple suivant permet de définir quatre ressources :

```
$resource
  ux1#tapes 3 "unités de bande"
  ux1#jobslots 24 "emplacements de travaux"
  ux2#tapes 2 "unités de bande"
  ux2#jobslots 16 "emplacements de travaux"
```

Voir aussi

Pour plus d'informations sur la façon d'effectuer la même tâche à partir de Dynamic Workload Console, voir :

la section "Conception de votre charge de travail" dans le document Dynamic Workload Console - Guide d'utilisation.

Définition d'un groupe de cycle d'exécution

Un groupe de cycle d'exécution est un objet de base de données dans lequel un ou plusieurs cycles d'exécution sont définis. Les cycles d'exécution combinés ensemble produisent un ensemble de dates d'exécution pour un flot de travaux. Un groupe de cycle d'exécution peut contenir :

- Des cycles d'exécution *inclusifs* qui indiquent le moment où un flot de travaux doit s'exécuter. Les mots clé définissant le cycle d'exécution suivent le mot clé *on*.
- Des cycles d'exécution *exclusifs* qui indiquent le moment où un flot de travaux ne doit pas s'exécuter. Les mots clé définissant le cycle d'exécution suivent le mot clé *except*. Généralement, un cycle d'exécution *exclusif* est mis en correspondance avec un cycle d'exécution *inclusif* de façon à définir les dates précises où le flot de travaux est exempté d'exécution.

Chacun des cycles d'exécution comprend ses propres mots clés de définition. Certains de ces mots clés peuvent également être définis au niveau du groupe de cycle d'exécution. Lorsqu'un cycle d'exécution et le groupe de cycle d'exécution incluent le même mot clé, la valeur définie dans la définition du cycle d'exécution est utilisée pour le cycle d'exécution. Lorsque la définition du cycle d'exécution omet un mot clé qui est défini au niveau du groupe de cycle d'exécution, elle hérite de la valeur.

Chaque définition de groupe de cycle d'exécution possède le format et les arguments suivants :

Syntaxe

```
$runcyclegroup
  nom_groupe_cycle_exécution ["description"]
    vartable nom_table
      [freedays nom_agenda [-sa] [-su]]
    [on [runcycle nom
```

=
|

```

[validfrom date] [validto date]
[description "texte"]
[variable nom_table]]
{date | jour | agenda | demande | "icalendar"}
[,...]
[fdignore | fdnext | fdprev][subset nom_sous-ensemble AND | OR]
[({at heure [+n day[s]] |
schedtime heure [+n day[s]]
[until | jsuntil heure [timezone | tz fuseau_horaire][+n day[s]] [onuntilaction]]
[every rate {everyendtime heure[+n day[s]]
[deadline heure [+n day[s]]}]]
[,...]
[except [runcycle nom]
[validfrom date] [validto date]
[description "texte"]
{date | jour | agenda | demande | "icalendar"}
[,...]
[fdignore | fdnext | fdprev][subset nom_sous-ensemble AND | OR]
[({at heure [+n day[s]] |
(schedtime heure [+n day[s]]))
[,...]
[({at heure [timezone | tz fuseau_horaire] [+n day[s]] |
schedtime heure [timezone | tz fuseau_horaire] [+n day[s]]}
[until | jsuntil heure [timezone | tz fuseau_horaire][+n day[s]] [onuntilaction]]
[every rate {everyendtime heure[+n day[s]]
[deadline heure [timezone | tz fuseau_horaire] [+n day[s]]
end

```

=
|

Arguments

nom_groupe_cycle_exécution

Indique le nom du groupe de cycle d'exécution. Ce nom peut comporter jusqu'à huit caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre.

"description"

Fournit une description du groupe de cycle d'exécution. La description peut contenir jusqu'à 120 caractères alphanumériques. Elle doit figurer entre guillemets. Son nom peut comporter des caractères alphanumériques à condition de commencer par une lettre. Il peut contenir les caractères suivants : virgule (,), point (.), tiret (-), plus (+), apostrophe (') et égal (=). Il ne peut pas comporter les caractères suivants : guillemet (") (autres que ceux encadrant la description), deux-points (:), point-virgule(;) et la perluète (&).

variable nom_table

Nom de la table de variables. Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut contenir jusqu'à 80 caractères.

[freedays nom_agenda [-sa] [-su]]

Indique un agenda des jours chômés pour le calcul des jours ouvrés correspondant au flot de travaux. Les samedis et dimanches peuvent également être définis comme jours ouvrés.

Nom_agenda

Nom de l'agenda devant être utilisé comme agenda des jours chômés pour le flot de travaux. Si *Nom_agenda* ne se trouve pas

dans la base de données, IBM Workload Scheduler génère un message d'erreur lorsque vous sauvegardez le flot de travaux. Si *Nom_agenda* ne se trouve pas dans la base de données lors de l'exécution de la commande **schedulr**, IBM Workload Scheduler génère un message d'erreur et utilise l'agenda par défaut **holidays** à la place. N'utilisez pas les noms des jours de la semaine pour nommer un agenda.

-sa Le samedi est considéré comme un *jour ouvré*.

-su Le dimanche est considéré comme un *jour ouvré*.

Voir «freedays», à la page 252 pour obtenir des détails et des exemples.

runcycle *nom*

Indique un libellé avec un nom usuel pour le cycle d'exécution spécifié dans les lignes suivantes.

valid from *date* ... **valid to** *date*

Délimite la période au cours de laquelle le flot de travaux est actif, c'est-à-dire pendant laquelle il est ajouté au plan de production. Notez que la date spécifiée comme valeur **valid to** n'est pas incluse dans le cycle d'exécution et le flot de travaux n'est donc pas actif à cette date.

description "*texte*"

Contient une description du cycle d'exécution.

variable

Indique le nom de la table de variables que le cycle d'exécution doit utiliser.

date Indique un cycle d'exécution qui s'exécute à des dates spécifiques. La syntaxe utilisée pour ce type de cycle d'exécution est la suivante :

aaaammjj [**aaaammjj**][**l**,...] Par exemple, pour un flot de travaux dont l'exécution est prévue les 25 mai 2009 et 12 juin 2009, la valeur est la suivante :

on
20090525,20090612

jour Indique un cycle d'exécution qui s'exécute des jours spécifiques. La syntaxe utilisée pour ce type de cycle d'exécution est la suivante :

{mo|tu|we|th|fr|sa|su} Par exemple, pour un flot de travaux qui est planifié pour s'exécuter chaque lundi, la valeur est la suivante :

on
mo

agenda Dates indiquées dans un agenda portant ce nom. Le nom de l'agenda peut être suivi d'un décalage au format suivant :

{+ | -}n {day[s] | weekday[s] | workday[s]}

Où :

n Indique le nombre de jours, jours de la semaine ou jours ouvrés.

days Chaque jour de la semaine.

weekdays

Tous les jours de la semaine sauf le samedi et le dimanche.

workdays

Tous les jours de la semaine excepté les samedis et dimanches (sauf

indication contraire par le mot clé **freedays**) et les dates figurant dans un agenda désigné des jours chômés ou dans l'agenda **holidays**.

demande

Sélectionne le flot de travaux uniquement lorsque celui-ci est demandé. Ce paramètre est utilisé pour les flots de travaux qui sont sélectionnés par nom et non par date. Pour empêcher un flot de travaux planifié d'être sélectionné pour le travail **JnextPlan**, redéfinissez-le ON REQUEST.

Remarque : Si vous tentez d'exécuter un flot de travaux contenant plusieurs heures "à la demande" (on request), tenez compte des points suivants :

- Les indicateurs "On request" ont la priorité sur les indicateurs "at".
- Les indicateurs "On request" n'ont jamais la priorité sur les indicateurs "On".

icalendar

Représente une norme utilisée pour spécifier une règle récurrente qui décrit le moment où un flot de travaux s'exécute.

La syntaxe utilisée pour un cycle d'exécution de type *icalendar* est la suivante :

FREQ={DAILY|WEEKLY|MONTHLY|YEARLY}

[:INTERVAL=[-]*n*]

[:{BYFREEDAY|BYWORKDAY|BYDAY=*liste_jours_semaine* |

BYMONTHDAY=*liste_jours_mois*}]

où la valeur par défaut du mot clé **INTERVAL** est 1.

A l'aide du mot clé *icalendar*, vous pouvez spécifier qu'un flot de travaux s'exécute :

tous les *n* jours

en utilisant le format suivant :

FREQ=DAILY[:INTERVAL=*n*]

où la valeur définie pour **valide de** est le premier jour des dates résultantes.

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque jour, la valeur est la suivante :

FREQ=DAILY

Pour un flot de travaux planifié pour s'exécuter un jour sur deux, la valeur est la suivante :

FREQ=DAILY; INTERVAL=2

chaque jour chômé ou ouvré

en utilisant le format suivant :

FREQ=DAILY[:INTERVAL=*n*]

[:BYFREEDAY|BYWORKDAY

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque jour chômé, la valeur est la suivante :

FREQ=DAILY; BYFREEDAY

Pour un flot de travaux planifié pour s'exécuter un jour ouvré sur deux, la valeur est la suivante :

FREQ=DAILY; INTERVAL=2; BYWORKDAY

toutes les *n* semaines, à des jours de la semaine spécifiques
en utilisant le format suivant :

FREQ=WEEKLY[;INTERVAL=*n*]

;BYDAY=*liste_jours_semaine*

où la valeur définie pour *liste_jours_semaine* est

[SU] [,MO] [,TU] [,WE] [,TH] [,FR] [,SA]

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque semaine le vendredi et le samedi, la valeur est la suivante :

FREQ=WEEKLY; BYDAY=FR,SA

Pour un flot de travaux planifié pour s'exécuter toutes les trois semaines, le vendredi, la valeur est la suivante :

FREQ=WEEKLY; INTERVAL=3; BYDAY=FR

tous les *n* mois, à des dates spécifiques
en utilisant le format suivant :

FREQ=MONTHLY[;INTERVAL=*n*]

;BYMONTHDAY=*liste_jours_mois*

où la valeur définie pour *liste_jours_mois* est représentée par une liste de

[*+nombre_de_jours_à_partir_début_du_mois*]

[*-nombre_de_jours_à_partir_fin_du_mois*]

[*nombre_de_jours_du_mois*]

Par exemple, pour un flot de travaux planifié pour s'exécuter le 27^{ème} jour de chaque mois, la valeur est la suivante :

FREQ=MONTHLY; BYMONTHDAY=27

Pour un flot de travaux planifié pour s'exécuter tous les six mois, le 15 du mois et le dernier jour du mois, la valeur est la suivante :

FREQ=MONTHLY; INTERVAL=6; BYMONTHDAY=15, -1

tous les *n* mois, à des jours spécifiques de la semaine
en utilisant le format suivant :

FREQ=MONTHLY[;INTERVAL=*n*]

;BYDAY=*numéro_de_la_semaine_du_mois_jour_semaine*

où la valeur définie pour

numéro_de_la_semaine_du_mois_jour_semaine est représentée par une liste de

[*+numéro_semaine_à_partir_début_du_mois*]

[*-numéro_semaine_à_partir_fin_du_mois*]

[*jour_semaine*]

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque mois, le premier lundi et le dernier vendredi du mois, la valeur est la suivante :

FREQ=MONTHLY; BYDAY=1MO, -1FR

Pour un flot de travaux planifié pour s'exécuter tous les six mois, le deuxième mardi du mois, la valeur est la suivante :

FREQ=MONTHLY; INTERVAL=6; BYDAY=2TU

toutes les n années

en utilisant le format suivant :

FREQ=YEARLY[;INTERVAL= n]

où la valeur définie pour **valide de** est le premier jour des dates résultantes.

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque année, la valeur est la suivante :

FREQ=YEARLY

Pour un flot de travaux planifié pour s'exécuter tous les deux ans, la valeur est la suivante :

FREQ=YEARLY; INTERVAL=2

Remarque : Les limitations suivantes s'appliquent :

- L'intervalle maximal pris en charge pour un cycle d'exécution quotidien est de 31 jours.
- L'intervalle maximal pris en charge pour un cycle d'exécution hebdomadaire est de 8 jours.
- L'intervalle maximal pris en charge pour un cycle d'exécution mensuel est de 12 mois. Pour des cycles d'exécution indiquant le jour de la semaine en fonction du mois, par exemple le troisième samedi ou le deuxième vendredi, l'intervalle maximal pris en charge est de 5 jours.
- L'intervalle maximal pris en charge pour un cycle d'exécution annuel est de 10 ans.

fdignore | fdnext | fdprev

Indique la règle à appliquer si la date sélectionnée pour l'exécution du travail ou du flot de travaux coïncide avec un jour chômé. Les valeurs possibles sont les suivantes :

fdignore

La date n'est pas ajoutée.

fdnext Le jour ouvré le plus proche qui suit le jour chômé est ajouté.

fdprev

Le jour ouvré le plus proche précédant le jour chômé est ajouté.

[subset nom_sous-ensemble AND | OR]

subset *nom_sous-ensemble*

Indique le nom du sous-ensemble. Si vous ne spécifiez pas de nom, SUBSET_1 est utilisé par défaut.

AND | OR

Par défaut, les cycles d'exécution dans un sous-ensemble sont dans une relation logique **OR** mais vous pouvez la changer en relation logique **AND** tant que le résultat du groupe de cycle d'exécution est une date ou un ensemble de dates positif (inclusif).

at heure [timezone | tz fuseau_horaire][+ n jour[s]]

Indique une dépendance horaire

heure Indique une heure. Les valeurs admises sont comprises entre **0000** et **2359**.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure de début. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

Remarque : Si des heures **at** et **until** ou **deadline** sont indiquées, les fuseaux horaires doivent être identiques.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

schedtime *heure* [**timezone** | **tz** *fuseau_horaire*][**+n jour[s]**]

Représente l'heure à laquelle le flot de travaux est positionné dans le plan.

heure Indique une heure de la journée au format : HHHHmm. Les valeurs admises sont comprises entre **0000** et **320000**.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure de début. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

until *heure* [**timezone** | **tz** *fuseau_horaire*][**+n day[s]**] [**onuntil** *action*]

Selon la définition d'objet à laquelle appartient le mot clé **until**, spécifie la dernière heure à laquelle un flot de travaux doit être terminé ou la dernière heure à laquelle un travail peut être lancé. Ce mot clé et le mot clé **jsuntil** s'excluent mutuellement.

heure Indique l'heure. Les valeurs admises sont comprises entre **0000** et **2359**.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

Remarque : Si une heure **until** et une heure **at** ou **deadline** sont spécifiées, les fuseaux horaires doivent être identiques.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

onuntil *action*

Selon la définition d'objet à laquelle appartient le mot clé **until**, spécifie :

- l'action à effectuer sur un travail ou flot de travaux dont l'heure **until** a été atteinte mais qui n'a pas encore démarré.
- L'action à prendre sur un flot de travaux dont le temps **until** a expiré alors que ce flot de travaux ne s'est pas encore terminé à l'état SUCC.

Les valeurs possibles du paramètre de l'action sont :

suppr Le travail ou le flot de travaux et tous les travaux ou flots de travaux dépendants ne s'exécutent pas. Il s'agit du comportement par défaut.

Une fois que l'heure **until** est arrivée à expiration pour un flot de travaux, l'état de ce dernier est calculé en fonction des règles habituelles ; les travaux supprimés ne sont pas pris en compte dans ce calcul. Dans le cas où le flot de travaux contient au moins un travail **every**, il est à l'état **HOLD**.

Une fois que l'heure **until** arrive à expiration pour un travail, ce dernier passe à l'état **HOLD** ou conserve tout état antérieur qui soit un état final.

Si l'heure **until** est transmise avec les options **onuntil suppr** et **carryforward**, le flot de travaux est *reporté* par **JnextPlan** uniquement si la date **until** correspond à la date d'exécution de **JnextPlan**. Si les dates d'exécution de **until** et **JnextPlan** ne sont pas identiques, le flot de travaux n'est pas *reporté*.

cont Le travail ou flot de travaux s'exécute lorsque toutes les conditions requises sont remplies. Un message apparaît alors dans le journal lorsque l'heure définie par **until** est dépassée.

Si l'heure **until** est transmise avec les options **onuntil cont** et **carryforward**, le flot de travaux est toujours *reporté* par **JnextPlan**.

canc Un travail ou flot de travaux est annulé lorsque l'heure **until** définie est atteinte. Lorsque le mot clé **onuntil canc** est utilisé sur des travaux, l'opération d'annulation sur le travail est émise par l'agent FTA sur lequel ce travail s'exécute. Tout travail ou flot de travaux qui dépendait de l'accomplissement d'un travail ou flot de travaux qui a été annulé, s'exécute car la dépendance n'est plus appliquée.

Si l'heure **until** est transmise avec les options **onuntil canc** et **carryforward**, le flot de travaux n'est pas *reporté* par **JnextPlan** car il est déjà annulé.

Remarque : Lorsque le mot clé **onuntil canc** est utilisé au niveau d'un flot de travaux, définissez comme propriétaire du flot de travaux le poste de travail le plus haut placé dans la hiérarchie de l'environnement de planification parmi tous les postes de travail qui possèdent des travaux appartenant à ce flot de travaux.

[jsuntil heure [+n day[s]]] [onuntilaction]

Définit la dernière heure de début possible d'un flot de travaux. Il détermine également le comportement des travaux dans le flot de travaux lorsque ce dernier approche de sa dernière heure de début. Ce mot clé et le mot clé **until** s'excluent mutuellement. Utilisez le mot clé **jsuntil** pour éviter que votre flot de travaux ne soit supprimé s'il démarre juste avant sa dernière heure de début et que la durée d'un ou de plusieurs travaux qu'il contient dépasse cet horaire. Par exemple, si vous avez un flot de travaux

avec **jsuntil** défini sur 10:00 et que l'un des travaux commence son exécution à 09:59, le travail et ses successeurs sont exécutés conformément à leur planification.

heure Indique l'heure. Les valeurs admises sont comprises entre 0000 et 2359.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

onuntil *action*

Selon la définition d'objet à laquelle appartient le mot clé until, spécifie :

- l'action à effectuer sur un travail ou flot de travaux dont l'heure until a été atteinte mais qui n'a pas encore démarré.
- L'action à prendre sur un flot de travaux dont le temps until a expiré alors que ce flot de travaux ne s'est pas encore terminé à l'état SUCC.

Les valeurs possibles du paramètre de l'*action* sont :

suppr Le travail ou le flot de travaux et tous les travaux ou flots de travaux dépendants ne s'exécutent pas. Il s'agit du comportement par défaut.

Une fois que l'heure until est arrivée à expiration pour un flot de travaux, l'état de ce dernier est calculé en fonction des règles habituelles ; les travaux supprimés ne sont pas pris en compte dans ce calcul. Dans le cas où le flot de travaux contient au moins un travail **every**, il est à l'état HOLD.

Une fois que l'heure until arrive à expiration pour un travail, ce dernier passe à l'état HOLD ou conserve tout état antérieur qui soit un état final.

Si l'heure **until** est transmise avec les options **onuntil** **suppr** et **carryforward**, le flot de travaux est *reporté* par **JnextPlan** uniquement si la date **until** correspond à la date d'exécution de JnextPlan. Si les dates d'exécution de **until** et JnextPlan ne sont pas identiques, le flot de travaux n'est pas *reporté*.

cont Le travail ou flot de travaux s'exécute lorsque toutes les conditions requises sont remplies. Un message apparaît alors dans le journal lorsque l'heure définie par until est dépassée.

Si l'heure **until** est transmise avec les options **onuntil** **cont** et **carryforward**, le flot de travaux est toujours *reporté* par **JnextPlan**.

canc Un travail ou flot de travaux est annulé lorsque l'heure **until** définie est atteinte. Lorsque le mot clé **onuntil** **canc** est utilisé sur des travaux, l'opération d'annulation sur le

travail est émise par l'agent FTA sur lequel ce travail s'exécute. Tout travail ou flot de travaux qui dépendait de l'accomplissement d'un travail ou flot de travaux qui a été annulé, s'exécute car la dépendance n'est plus appliquée.

Si l'heure **until** est transmise avec les options **onuntil canc** et **carryforward**, le flot de travaux n'est pas *reporté* par **JnextPlan** car il est déjà annulé.

Remarque : Lorsque le mot clé *onuntil canc* est utilisé au niveau d'un flot de travaux, définissez comme propriétaire du flot de travaux le poste de travail le plus haut placé dans la hiérarchie de l'environnement de planification parmi tous les postes de travail qui possèdent des travaux appartenant à ce flot de travaux.

every *fréquence* {**everyendtime** *heure*[+*n* **day**[*s*]]}

Définit la fréquence de répétition à laquelle les instances du même flot de travaux sont exécutées au cours d'un intervalle de temps. Le flot de travaux est lancé à plusieurs reprises à la fréquence spécifiés jusqu'au moment indiqué dans **everyendtime**.

fréquence

La fréquence de répétition, exprimée en heures et en minutes (*hhmm*), à laquelle les instances du flot de travaux sont lancées.

time L'heure de fin, exprimée en heures et en minutes (*hhmm*), à laquelle l'intervalle de répétition s'arrête. Après cette heure, aucune instance du flot de travaux ne s'exécute. L'utilisation du mot clé **everyendtime** est obligatoire.

n Le nombre de jours auquel la valeur de **everyendtime** peut être étendue. Par exemple, si vous tapez :

everyendtime 1550 + 1

et que le flot de travaux est planifié pour démarrer son exécution aujourd'hui à 10:00, l'heure de fin à laquelle les instances du flot de travaux arrêteront d'être lancées est demain à 15:50.

Exemple

L'exemple suivant définit un groupe de cycle d'exécution, RCG2, qui contient un cycle d'exécution inclusif, RUN_CYCLE1, et deux cycles d'exécution exclusifs, RUN_CYCLE2 et RUN_CYCLE3. Pour déterminer le calendrier d'exécution du flot de travaux associé à ce groupe de cycle d'exécution, l'intersection des deux cycles de diffusion et en exclusivité (les deux cycles d'exécution exclusifs ont une relation ET logique entre elles) est soustrait du cycle d'exécution inclusif. Voici les caractéristiques du groupe de cycle d'exécution :

Cycle d'exécution inclusif RUN_CYCLE1

où :

- Le calendrier CAL1 définit les jours considérés comme chômés pour le flot de travaux. Samedi et dimanche sont considérés comme des jours ouvrés.
- Le flot de travaux ne s'exécute pas avant deux jours après le 31 mars 2008 (2 avril) et pas plus tard que deux jours après le 12 avril 2008 (14 avril). Chaque jour, le flot de travaux est retardé de deux jours.

- Les flots de travaux s'exécutent tous les jours (après un retard de deux jours) à partir de 7 h du matin mais pas après 9 h, auquel cas le flot de travaux est supprimé sans être exécuté. Le flot de travaux doit se terminer à 10 h du matin.

Cycle d'exécution exclusif RUN_CYCLE2

Si le flot de travaux tombe un jour chômé, le jour ouvrable le plus proche avant le jour chômé est exclu.

Si le flot de travaux tombe le 1er avril 2008 et que ce jour se trouve être chômé, le jour ouvrable le plus proche après le jour chômé est exclu.

Cycle d'exécution exclusif RUN_CYCLE3

Si le flot de travaux tombe le 1er avril 2008 et que ce jour se trouve être chômé, le jour ouvrable le plus proche après le jour chômé est exclu.

```
RUNCYCLEGROUP RCG2
DESCRIPTION "Exemple de groupe de cycle d'exécution"
VARIABLE TABLE1
FREEDAYS CAL1 -SA -SU
  ON    RUNCYCLE RUN_CYCLE1 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
        "Inclusive Run Cycle" VARIABLE TABLE1 "FREQ=DAILY;" FDIGNORE
        (AT 0700 +2 DAYS UNTIL 0900 +2 DAYS ONUNTIL SUPPR DEADLINE 1000 +2 DAYS)

  EXCEPT RUNCYCLE RUN_CYCLE2 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
        "Exclusive Run Cycle" CAL1 FDPREV SUBSET SUBSET_A AND
        (AT 0700 +2 DAYS)

  EXCEPT RUNCYCLE RUN_CYCLE3 VALIDFROM 03/31/2008 VALIDTO 04/12/2008 DESCRIPTION
        "Exclusive Run Cycle" 04/01/2008 FDNEXT SUBSET SUBSET_A AND
        (SCHEDTIME 0700 +2 DAYS)
SCHEDTIME 0700 TZ Europe/Berlin +2 DAYS UNTIL 0900 TZ Europe/Berlin +2 DAYS ONUNTIL
  CONT DEADLINE 1000 TZ Europe/Berlin +2 DAYS
FIN
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de flot de travaux".

Définition de flot de travaux

Un flot de travaux comprend une séquence des travaux à exécuter, ainsi que des horaires, des priorités et d'autres dépendances qui déterminent l'ordre de traitement.

Un flot de travaux commence avec un mot clé **schedule** suivi par des attributs et des dépendances. Le signe deux-points (:) en tant que délimiteur introduit les travaux appelés par le flot de travaux. Chaque travail possède ses attributs et dépendances.

Syntaxe

```
/
schedule [poste_de_travail#][folder/]nom_flot_travaux
          # commentaire
          [validfrom date]
          [timezone|tz fuseau_horaire]
          [description "texte"]
```



```

[draft]
[vartable nom_table]
[freedays nom_agenda [-sa] [-su]]
[on [runcycle nom
    [validfrom date] [validto date]
    [description "texte" ]
    [vartable nom_table]]
    {date|jour|agenda|demande|"icalendar"|groupe_cycle_exécution} [...]
    [fdignore|fdnext|fdprev]
    [(at heure [+n day[s]] |
    schedtime heure [+n day[s]])]
    [until | jsuntil heure [+n day[s]] [onuntil action]]
    [every fréquence {everyendtime heure[+n day[s]]]
    [deadline heure [+n day[s]]]]]
[...]]
startcond filecreated | filemodified nom_poste_de_travail#nom_fichier user nom_utilisateur interval
secondes [(alias nom_travail_condition_démarrage rerun batch outfile nom_fichier_sortie
params "paramètres filemonitor supplémentaires") |
startcond job nom_poste_travail#nom_travail outcond condition_sortie_travail interval
secondes [(alias nom_travail_condition_démarrage rerun)]
:
    [except [runcycle nom]
        [validfrom date] [validto date]
        [description "texte" ]
        {date|jour|agenda|demande|"icalendar"|groupe_cycle_exécution} [...]
        [fdignore|fdnext|fdprev]
        [(at heure [+n day[s]])] |
        (schedtime heure [+n day[s]])]
    [...]]
    [(at heure [timezone|tz fuseau_horaire] [+n day[s]] |
    schedtime heure [timezone|tz fuseau_horaire] [+n day[s]])]
    [until | jsuntil heure [timezone|tz fuseau_horaire] [+n day[s]] [onuntil action]]
    [deadline heure [timezone|tz fuseau_horaire] [+n day[s]]]
    [carryforward]
    [matching {previous|sameday|relative from [+ | -] heure to [+ | -] heure |
        from heure [+ | -n day[s]] to heure [+ n day[s]] [...] }
    / [follows {{netagent::}[poste_de_travail#][folder/]nom_flot_travaux[nom_travail |
        @] [previous|
        sameday|relative from [+|-] heure to [+|-] heure |
        from heure [+|-n day[s]] to heure [+|-n day[s]]
        ][if <condition> [| <condition>...]]
        } | [...] [...]]
    [join nom_condition [nombre | nombre_conditions | all] of
    description "..."]
    .....
    endjoin
    [keysched]
    [limit nombre_max_travaux]
    [needs { [n] [poste_de_travail#]nom_ressource } [...] ] [...]
    [opens { [poste_travail#]"nom_fichier" [ (qualificatif) ] [...] } ] [...]
    [priority nombre | hi | go]
    [prompt {nom_invite|":!!texte" } [...] ] [...]
    / [onoverlap {parallel|enqueue|donotstart}]
    :
instruction-travail
    # commentaire

```

```

= nom_travail [alias_travail]
= [outcond condition_sortie_travail interval secondes]
    [[at heure [timezone|tz fuseau_horaire] [+n day[s]] |
    schedtime heure [timezone|tz fuseau_horaire] [+n jour[s]]][,...]
    [until heure [timezone|tz fuseau_horaire] [+n day[s]] [onuntil action]
    = [deadline heure [timezone|tz fuseau_horaire] [+n day[s]] [onlate action] ]
    [maxdur heure | pourcentage % onmaxdur action]
    [mindur heure | pourcentage % onmindur action]
    [every fréquence]
    [follows {[agent_réseau::][poste_de_travail#]nom_flot_travaux{.nom_travail @} [previous |
    sameday|relative from [+|-] heure to [+|-] heure |
    from heure [+|-n day[s]] to heure [+|-n day[s]]
    }][if <condition> [ | <condition>...]] [...]] [...]
[join nom_condition [nombre | nombre_conditions | all] of
    description "..."]
    .....
    endjoin
    [confirmed]
    [critical]
    [keyjob]
    [needs { [n] [poste_de_travail#]nom_ressource } [...]] [...]
    [opens { [poste_travail#"nom_fichier" [ (qualificatif) ] [...]] } [...]
    [priority nombre | hi | go]
    [prompt {nom_invite|":|!|texte"} [...]] [...]
    [nop]
    [statistictype custom]

```

```

[instruction-travail...]
end

```

Arguments

Le tableau 35 contient une brève description des mots clés des définitions de flots de travaux. Vous trouverez une description détaillée de chaque mot clé de planification dans les sous-sections suivantes.

Tableau 35. Liste des mots clés de planification

Mot clé	Description	Page
at	Définit la première heure à laquelle l'exécution d'un travail ou d'un flot de travaux peut être lancée. Lorsqu'il est défini dans un cycle d'exécution, ce mot clé spécifie la première heure à laquelle un travail ou un flot de travaux peut être lancé pour ce cycle d'exécution spécifique.	«at», à la page 233
carryforward	Reporte le flot de travaux lorsque celui-ci n'est pas terminé.	«carryforward», à la page 235
comment	Inclut des commentaires dans la définition d'un flot de travaux ou dans un travail appartenant au flot de travaux.	«comment», à la page 235
confirmed	Indique qu'une confirmation de l'état d'achèvement du travail est nécessaire.	«confirmed», à la page 236
critical	Permet d'indiquer que le travail a une importance critique et doit donc être géré en priorité.	«critical», à la page 236

Tableau 35. Liste des mots clés de planification (suite)

Mot clé	Description	Page
deadline	Indique le temps nécessaire pour l'achèvement d'un travail ou d'un flot de travaux. Lorsqu'il est défini dans un cycle d'exécution, ce mot clé spécifie le délai dans lequel un travail ou un flot de travaux doit être terminé pour ce cycle d'exécution spécifique.	«deadline», à la page 237
description	Contient une description du flot de travaux. La longueur maximale de cette zone est 120 caractères.	«description», à la page 238
draft	Indique que le processus de génération du plan doit ignorer ce flot de travaux.	«draft», à la page 238
end	Marque la fin d'un flot de travaux.	«end», à la page 239
every	Lance un flot de travaux ou un travail à plusieurs reprises à une fréquence donnée.	«every», à la page 239
except	Spécifie les dates on où le flot de travaux ne sera pas exécuté. Il peut être suivi d'une définition de cycle d'exécution.	«except», à la page 244
fdignore fdnext fdprev	Indique une règle qui doit être appliquée lorsque la date devant être exclue correspond à un jour chômé.	«except», à la page 244
follows	Indique les travaux ou flots de travaux qui doivent aboutir ou satisfaire un ou plusieurs conditions de sortie avant le lancement du travail ou du flot de travaux en cours de définition.	«follows», à la page 248
freedays	Indique un agenda des jours chômés pour le calcul des <i>jours ouvrés</i> correspondant au flot de travaux. Les samedis et dimanches peuvent également être définis comme <i>jours ouvrés</i> .	«freedays», à la page 252
interval	Indique la fréquence à laquelle IBM Workload Scheduler vérifie si la condition est remplie.	«startcond», à la page 283
job statement	Définit un travail et les dépendances qui lui sont associées.	«job statement», à la page 254
join	Définit un ensemble de dépendances conditionnelles sur un travail ou un flot de travaux.	«Jointure», à la page 256
jsuntil	Indique que l'exécution du flot de travaux continue également si l'un de ses travaux commence à s'exécuter avant que le délai indiqué dans le mot clé jsuntil soit écoulé. Ce mot clé est activé par défaut à partir de la version 9.4, groupe de correctifs 1. Ce mot clé et le mot clé until s'excluent mutuellement. Pour plus d'informations sur le mot clé until , voir «until», à la page 287.	«jsuntil», à la page 257

Tableau 35. Liste des mots clés de planification (suite)

Mot clé	Description	Page
keyjob	Qualifie un travail comme étant clé dans la base de données et dans le plan, afin qu'il soit surveillé par différentes applications, notamment IBM Tivoli Business Systems Manager ou IBM Tivoli Enterprise Console.	«keyjob», à la page 259
keysched	Qualifie un flot de travaux comme étant clé dans la base de données et dans le plan, afin qu'il soit surveillé par différentes applications, notamment IBM Tivoli Business Systems Manager ou IBM Tivoli Enterprise Console.	«keysched», à la page 260
limit	Définit le nombre maximal de travaux qui peuvent être lancés simultanément à partir de ce flot de travaux.	«limite», à la page 260
matching	Définit les critères de correspondance utilisés lorsqu'aucun critère de correspondance n'est spécifié dans les spécifications prédécesseur/successeur de la définition du flot de travaux ou de la définition de travail dans le flot de travaux.	«matching», à la page 261
maxdur	Indique la durée maximale d'exécution d'un travail. Vous pouvez exprimer cette durée en minutes, ou sous la forme d'un pourcentage de la durée estimée du travail la plus récente.	«maxdur», à la page 262
mindur	Indique la durée la plus courte au cours de laquelle un travail s'exécute normalement et se termine.	«mindur», à la page 264
needs	Définit le nombre d'unités d'une ressource requises par ce travail ou ce flot de travaux pour que celui-ci puisse être lancé. Le plus grand nombre de ressources dont le flot de travaux peut être dépendant est 1024.	«needs», à la page 265
nop	Indique qu'un travail ne doit pas être exécuté lorsque le plan s'exécute. Le travail est inclus dans le plan mais, lorsque le plan est mis en oeuvre, il prend le statut Annulation en instance et n'est pas exécuté.	«nop», à la page 266
on	Définit les dates d'exécution du flot de travaux. Il peut être suivi d'une définition de cycle d'exécution.	«on», à la page 267
onlate	Définit l'action à effectuer sur un travail dans le flot de travaux lorsque l'échéance arrive.	«onlate», à la page 273
onoverlap	Indique comment gérer une instance de flot de travaux qui est programmée pour démarrer bien que l'instance précédente ne se soit pas encore terminée.	«onoverlap», à la page 274

Tableau 35. Liste des mots clés de planification (suite)

Mot clé	Description	Page
opens	Définit les ressources qui doivent être disponibles avant le lancement d'un travail ou d'un flot de travaux.	«opens», à la page 274
onuntil	Indique l'action à effectuer sur un travail ou flot de travaux dont l'heure until a été atteinte.	«until», à la page 287
outcond	Condition de sortie, qui lorsqu'elle est remplie, libère la partie restante du flot de travaux ou le travail dans lequel elle est spécifiée.	«startcond», à la page 283
priority	Définit la priorité d'un travail ou d'un flot de travaux.	«priority», à la page 277
prompt	Définit les invites qui doivent avoir reçu une réponse pour que le travail ou le flot de travaux puisse être lancé.	«prompt», à la page 278
runcycle	Indique un libellé avec un nom usuel pour le cycle d'exécution. Il est utilisé avec les mots clés suivants : except Pour les cycles d'exécution <i>exclusifs</i> , qui définissent les moments où le flot de travaux ne doit pas s'exécuter. on Pour les cycles d'exécution <i>inclusifs</i> , qui définissent les moments où le flot de travaux doit s'exécuter.	• «except», à la page 244 • «on», à la page 267
schedule	Attribue un nom au flot de travaux.	«schedule», à la page 281
schedtime	Indique l'heure utilisée pour définir le flot de travaux dans le diagramme chronologique du plan afin de déterminer les successeurs et les prédécesseurs.	«schedtime», à la page 279
statisticstype custom	Marque un travail pour l'utilisation de l'analyse statistique basée sur SPSS afin de prévoir sa durée estimée.	«statisticstype custom», à la page 282
startcond	Génère dans le flot de travaux un mécanisme qui recherche des événements et des conditions spécifiques, et libère le flot de travaux lorsque ceux-ci se produisent.	«startcond», à la page 283
timezone tz	Indique le fuseau horaire à utiliser lors du calcul de l'heure de début.	«timezone», à la page 286
until	Définit la dernière heure à laquelle un travail ou un flot de travaux peut être lancé. Lorsqu'il est défini dans un cycle d'exécution, ce mot clé spécifie la dernière heure à laquelle un travail ou un flot de travaux peut être lancé pour ce cycle d'exécution spécifique. Ce mot clé et le mot clé jsuntil s'excluent mutuellement. Pour plus d'informations sur le mot clé jsuntil , voir «jsuntil», à la page 257.	«until», à la page 287

Tableau 35. Liste des mots clés de planification (suite)

Mot clé	Description	Page
validfrom	Définit la date à laquelle l'instance du flot de travaux démarre.	«validfrom/validto», à la page 290
validto	Définit la date de fin de l'instance de flot de travaux.	«validfrom/validto», à la page 290
vartable	Définit la table de variables que le flot de travaux et le cycle d'exécution doivent utiliser.	«table de variables», à la page 291

Remarque :

1. Les flots de travaux planifiés pour être exécutés sur des postes de travail marqués comme *ignorés* ne sont pas ajoutés au plan de production lorsque le plan est créé ou étendu.
2. Les mots clés saisis incorrectement dans les définitions de travail entraînent le stockage de définitions de travaux tronquées dans la base de données. En fait, le mot clé erroné est considéré comme externe à la définition de travail et il est interprété comme le nom de travail d'une définition de travail supplémentaire. Généralement, cette mauvaise interprétation entraîne également une erreur de syntaxe ou une erreur de définition de travail inexistant pour la définition de travail supplémentaire.
3. Le fait d'accorder l'accès à une classe ou un domaine de poste de travail signifie qu'on accorde l'accès à l'objet lui-même, et non un accès aux postes de travail de cet objet.

Règles de spécification du fuseau horaire

Vous pouvez indiquer un fuseau horaire à différents niveaux de mot clé dans une définition de flot de travaux. En d'autres termes :

- Pour l'ensemble du flot de travaux (y compris toutes ses spécifications de mot clé)
- Au niveau de restriction temporelle (avec les mots clés at, deadline, schedtime et until)
- Pour chaque instruction de travail incluse

Les règles ci-dessous s'appliquent lors de la résolution des fuseaux horaires spécifiés dans une définition de flot de travaux :

- Lorsque vous indiquez le fuseau horaire au niveau du flot de travaux, cela s'applique aux définitions temporelles du cycle d'exécution (définies avec le mot clé on) et à celles des restrictions temporelles.
- Si vous indiquez un fuseau horaire au niveau du flot de travaux et de la restriction temporelle, ils doivent être identiques. Si vous n'indiquez aucun fuseau horaire, tant au niveau du flot de travaux que de la restriction temporelle, le fuseau horaire spécifié sur le poste de travail est utilisé.
- Le fuseau horaire indiqué au niveau du travail peut être différent de celui indiqué au niveau du flot de travaux et peut le remplacer. Si vous n'indiquez aucun fuseau horaire, tant au niveau du flot de travaux que du travail, le fuseau horaire spécifié sur le poste qui exécute le travail est utilisé.

Règles de spécification de la restriction temporelle

Dans une définition de flot de travaux, vous pouvez spécifier des restrictions temporelles (avec les mots clés *at*, *deadline*, *schedtime* et *until*) au niveau du flot de travaux et du cycle d'exécution. Si ces deux restrictions sont spécifiées, les restrictions temporelles au niveau du cycle d'exécution remplacent celles spécifiées au niveau du flot de travaux.

Exemple

Voici un exemple de définition de flot de travaux :

```
SCHEDULE M235062_99#SCHED_FIRST1 VALIDFROM 06/30/2005
ON RUNCYCLE SCHED1_PRESIMPLE VALIDFROM 07/18/2005 "FREQ=DAILY;INTERVAL=1"
( AT 1010 )
ON RUNCYCLE SCHED1_PRED_SIMPLE VALIDFROM 07/18/2005 "FREQ=DAILY;INTERVAL=1"
CARRYFORWARD
PROMPT "parto o no?"
PRIORITY 55
:
M235062_99#JOBMDM
PRIORITY 30
NEEDS 16 M235062_99#JOBSLOTS
PROMPT PRMT3

B236153_00#JOB_FTA
FOLLOWS JOBMDM
END
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de flot de travaux".

Détail des mots clés pour la définition des flots de travaux

Cette section décrit les mots clés des définitions de flots de travaux répertoriés dans le tableau 35, à la page 228.

at

Indique une dépendance horaire. Si le mot clé **at** est utilisé, le travail ou le flot de travaux ne peut pas démarrer avant l'heure définie dans ce mot clé.

Syntaxe

at *heure* [*timezone* | *tz fuseau_horaire*][*+n jour[s]*] [*absolute* | *abs*]

Arguments

heure Indique une heure. Les valeurs admises sont comprises entre 0000 et 2359.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure de début. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

Remarque : Si des heures **at** et **until** ou **deadline** sont indiquées, les fuseaux horaires doivent être identiques.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

absolute

Indique que la date de début est fonction du jour calendaire plutôt que du jour de production.

Commentaires

Si l'heure n'est pas spécifiée à l'aide du mot clé **at** pour un travail ou un flot de travaux, l'heure de lancement est déterminée par ses dépendances et sa priorité. En outre, sa position dans le plan de préproduction est déterminée par la valeur affectée au mot clé **schedtime**. Pour plus d'informations sur le mot clé **schedtime**, voir «schedtime», à la page 279.

Si les heures de début du cycle d'exécution et du flot de travaux sont définies, celle du cycle d'exécution l'emporte lorsque le flot de travaux est planifié avec **JNextPlan**. Si le flot de travaux est lancé avec la commande **submit**, l'heure de début du cycle d'exécution n'est pas utilisée.

La valeur de temps dans l'option **at** est considérée comme suit :

- Si la valeur de temps est inférieure à l'heure définie dans l'option globale **startOfDay**, elle est prise en compte pour le jour suivant.
- Si la valeur de temps est supérieure à l'heure définie dans l'option globale **startOfDay**, elle est prise en compte pour le jour courant.

Si le gestionnaire de domaine maître de votre réseau s'exécute lorsque la valeur **yes** est attribuée aux options **enLegacyStartOfDayEvaluation** et **enTimeZone** afin de convertir l'heure **startOfDay** du gestionnaire de domaine maître en fuseau horaire local sur chaque poste de travail du réseau, vous devez ajouter le mot clé **absolute** pour le faire fonctionner lors de la soumission d'un travail ou d'un flot de travaux.

Si aucun des mots clés **at** et **schedtime** n'est spécifié dans la définition du flot de travaux, l'instance de travail ou de flot de travaux est positionnée par défaut dans le plan à l'heure indiquée dans l'option globale **startOfDay**.

Exemples

Dans les exemples suivants, on considère que le jour de traitement IBM Workload Scheduler commence à 6 heures du matin.

- Comme le flot de travaux suivant doit être exécuté tous les mardis, il est lancé au plus tôt le mercredi à 3 heures du matin. Les deux travaux qu'il contient sont lancés dès que possible après cette heure.

```
schedule sked7 on tu at 0300:  
  job1  
  job2  
end
```

- L'exemple ci-dessous lance le flot de travaux **mysked** tous les dimanches à 8:00. Les travaux **job1**, **job2** et **job3** sont tous lancés le dimanche.

```
schedule mysked on fr at 0800 + 2 days  
:  
  job1  
  job2 at 0900  
  job3 follows job2 at 1200  
end
```


- Le fuseau horaire du poste de travail sfran correspond à l'heure du pacifique America/Los_Angeles et le fuseau horaire du poste de travail nycity correspond à l'heure de la côte Est America/New_York. Le flot de travaux suivant est sélectionné pour s'exécuter le vendredi. Il est lancé sur le poste de travail sfran à 10 heures du matin sur le fuseau horaire America/Los_Angeles le samedi. Le travail job1 est lancé sur sfran dès que possible après cette heure. Le travail job2 est lancé sur sfran à 14 heures sur le fuseau horaire America/New_York (11:00 a.m. America/Los_Angeles) le samedi. Le travail job3 est lancé sur le poste de travail nycity à 16 heures sur le fuseau horaire America/New_York (1:00 p.m. America/Los_Angeles) le samedi.

```
sfran#schedule sked8 on fr at 1000 + 1 day:
job1
job2 at 1400 tz America/New_York
nycity#job3 at 1600
end
```

carryforward

Autorise le report d'un flot de travaux au plan de production suivant, s'il n'est pas terminé avant la fin du plan de production courant.

Syntaxe

carryforward

Exemples

Le flot de travaux suivant est reporté si les travaux qu'il contient ne sont pas terminés avant le début du processus de préproduction de la période de production suivante.

```
schedule sked43 on th
carryforward
:
job12
job13
job13a
end
```

comment

Inclut des commentaires dans une définition de flot de travaux et dans les travaux contenus dans un flot de travaux.

Syntaxe

*texte*

Commentaires

Insère une ligne de commentaire. Le premier caractère de la ligne doit être un signe dièse (#).

Vous pouvez ajouter des commentaires à une définition de flots de travaux immédiatement après la ligne qui contient le mot clé **schedule** ou dans un travail contenu dans une définition de flot de travaux immédiatement après la ligne *job statement*.

Exemples

L'exemple suivant contient les deux types de commentaire :

```
schedule wkend on fr at 1830
#####
# Les travaux de nettoyage hebdomadaires
#####
#
carryforward
:
job1
# totaux et rapports finals
job2
# mettre à jour la base de données
end
```

confirmed

Indique que la fin d'un travail doit être confirmée.

Pour confirmer la fin d'un travail, exécutez la commande **conman confirm**. Pour plus d'informations, voir «confirm», à la page 450.

Syntaxe

confirmed

Exemples

Dans le flot de travaux suivant, la confirmation de la fin du travail job1 doit être reçue avant le lancement de job2 et de job3.

```
schedule test1 on fr:
job1 confirmed
job2 follows job1
job3 follows job1
end
```

critical

Permet d'indiquer que le travail a une importance critique et doit donc être traité en conséquence.

Un travail critique bénéficie d'un traitement spécifique. Compte tenu de l'échéance et de la durée estimée, le planificateur :

- lors de la conception du plan, ou à chaque exécution de la commande **submit**, calcule l'heure de début au plus tard pour le lancement de chacun de ses prédécesseurs afin que le travail respecte l'échéance fixée. Il s'agit de l'*heure de début critique*. Une heure de début critique est attribuée au travail critique et à chacun de ces prédécesseurs.

L'ensemble des prédécesseurs du travail critique est appelé *réseau critique* du travail.

- recalcule de manière dynamique, lors de l'exécution du plan, les heures de début critiques dans le réseau critique.

Lorsqu'un prédécesseur risque de compromettre la réalisation opportune du travail critique, il est *promu*. En d'autres termes, grâce à différents mécanismes du système d'exploitation (implémentation de la commande **nice** sous UNIX ou modification du niveau de priorité sous Windows, par exemple), des ressources supplémentaires lui sont attribuées et sa soumission est prioritaire sur les autres

/ travaux ne se trouvant pas dans le réseau critique. Cette action est exécutée de
/ manière récurrente sur tous les prédécesseurs à l'intérieur du réseau critique et,
/ le cas échéant, sur le travail critique tant qu'il n'existe aucun risque de retard de
/ ce travail.

/ **Important :** Une échéance doit être spécifiée pour les travaux critiques au niveau
/ du travail, du flot de travaux ou du cycle d'exécution, notamment dans le plan car
/ ils sont associés à un cycle d'exécution. Alors que les travaux critiques soumis à la
/ demande dans le plan peuvent ne pas avoir d'échéance, dans ce cas, l'option
/ globale `deadlineOffset` est utilisée.

/ **Syntaxe**

/ **critical**

/ **deadline**

/ Indique le temps nécessaire pour l'achèvement d'un travail ou d'un flot de travaux.
/ Les travaux et flots de travaux qui n'ont pas encore démarré ou qui sont encore en
/ cours d'exécution lorsque la date d'échéance arrive à expiration sont considérés
/ comme étant *en retard* dans le plan. Lorsqu'un travail (ou flot de travaux) est en
/ retard, les actions suivantes sont exécutées :

- / • Le travail apparaît en retard dans **conman**.
- / • Le programme envoie un événement à Tivoli Enterprise Console et IBM Tivoli
/ Business Systems Manager.
- / • Un message est envoyé à *stdlist* et aux journaux de la console.

/ Lorsqu'un travail ne se termine pas avant son délai, un message d'avertissement
/ est affiché. Si ce travail ne fait pas partie d'un flot de travaux réacheminé et que
/ vous exécutez `JnextPlan` alors qu'il est encore en cours d'exécution, le travail est
/ inséré dans `USERJOBS`. Dans ce cas, un autre message d'avertissement concernant
/ l'expiration du délai est ajouté dans le fichier `TWS_home/stdlist/logs/`
/ `aaaammjj_TWSMERGE.log`.

/ **Remarque :** Lorsque vous utilisez le mot clé **deadline**, veillez à ce qu'une valeur
/ supérieure à 0 soit attribuée à l'option **bm check deadline** dans le fichier de
/ configuration `localopts` sur les postes de travail que vous utilisez. Définissez
/ l'option **bm check deadline** sur chaque poste de travail sur lequel vous souhaitez
/ être notifié de l'expiration du délai, ou, si vous souhaitez obtenir des informations
/ à jour sur l'ensemble de l'environnement, définissez l'option dans le gestionnaire
/ de domaine maître. Les échéances des travaux critiques sont évaluées
/ automatiquement, indépendamment de l'option **bm check deadline**. Pour plus
/ d'informations sur l'option **bm check deadline**, voir le *Guide d'administration*.

/ **Syntaxe**

/ **deadline** *heure* [**timezone** | *tz fuseau_horaire*][**+n day[s]**]

/ **Arguments**

/ *heure* Indique une heure. Les valeurs admises sont comprises entre **0000** et **2359**.

/ *tzname* Indique le fuseau horaire à utiliser lors du calcul de l'heure d'échéance.
/ Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les

/ noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire
/ correspondant au poste de travail sur lequel le travail ou le flot de travaux
/ est lancé.

/ *n* Indique un décalage en jours à partir de l'heure d'échéance planifiée.

/ **Remarque :** Si des heures **deadline** et **until** ou **at** sont indiquées, les fuseaux
/ horaires doivent être identiques.

/ **Exemples**

/ L'exemple suivant lance le flot de travaux sked7 chaque jour, ainsi que le travail
/ jobc. Ils doivent démarrer à 14h30 et être terminés à 16h00.

/ schedule sked7 on everyday :
/ jobc at 1430 deadline 1600
/ end

/ **description**

/ Inclut une description du flot de travaux.

/ **Syntaxe**

/ **description** "*texte*"

/ **Commentaires**

/ La longueur maximale de cette zone est 120 caractères.

/ **Exemples**

/ schedule test1
/ description "Revenue at the end of the month"
/ on monthend
/ :
/ job1
/ job2
/ job3
/ end

/ **draft**

/ Marque un flot de travaux comme épreuve. Un flot de travaux au stade d'épreuve
/ n'est pas ajouté au plan de production.

/ **Syntaxe**

/ **draft**

/ **Commentaires**

/ Un flot de travaux à l'état de brouillon n'est pas pris en compte lors de la
/ résolution des dépendances et n'est pas ajouté au plan de production. Après avoir
/ supprimé le mot clé provisoire d'un flot de travaux, vous devez exécuter la
/ commande **JnextPlan** pour ajouter le flot de travaux au plan de préproduction,
/ puis au plan de production.

/ **Exemples**

/ schedule test1 on monthend
/ draft
/ :

```

/      job1
/      job2
/      job3
/      end

/      end
/      Signale la fin d'une définition de flot de travaux.

```

/ **Syntaxe**

```
/ end
```

/ **Exemples**

```

/      schedule test1 on monthend
/      :
/      job1
/      job2
/      job3
/      end << end of job stream >>

```

| **every**

| Définit la fréquence de répétition pour un flot de travaux ou pour un travail. La
 | syntaxe et l'emplacement du mot clé varient en fonction de l'objet pour lequel le
 | mot clé est utilisé. Lorsqu'il est utilisé pour un flot de travaux, il est indiqué dans
 | la définition d'un cycle d'exécution pour ce flot de travaux. Lorsqu'il est utilisé
 | pour un travail, il est indiqué dans la définition du travail dans le flot de travaux.

| Pour plus d'informations, voir :

- | • «every (utilisé pour les flots de travaux)»
- | • «every (utilisé pour les travaux)», à la page 240

| **every (utilisé pour les flots de travaux)**

| Définit la fréquence de répétition d'un flot de travaux. Le flot de travaux est lancé
 | de façon répétitive à la fréquence indiquée. Le mot clé ne peut être indiqué que
 | dans la définition d'un cycle d'exécution pour le flot de travaux.

| L'utilisation de ce mot clé entraîne la création au moment de la génération du plan
 | d'un certain nombre d'instances du flot de travaux qui dépend de la fréquence de
 | répétition, de l'heure de début (définie soit avec *at*, soit avec *schedtime*, ou basée
 | sur le début de journée), et de l'heure définie pour *everyendtime*.

| **Syntaxe**

| **every** *fréquence* {**everyendtime** *heure*[*+n day[s]*]}

| **Arguments**

| **fréquence**

| La fréquence de répétition, exprimée en heures et en minutes (*hhmm*), à
 | laquelle les instances du flot de travaux sont lancées.

| **heure** L'heure de fin, exprimée en heures et en minutes (*hhmm*), à laquelle
 | l'intervalle de répétition s'arrête. Après cette heure, aucune instance du flot
 | de travaux ne s'exécute. L'utilisation du mot clé *everyendtime* est
 | obligatoire.

| **n** Le nombre de jours auquel la valeur de *everyendtime* peut être étendue.
 | Par exemple, si vous tapez :

everyendtime 1550 + 1

et que le flot de travaux est planifié pour démarrer son exécution aujourd'hui à 10:00, l'heure de fin à laquelle les instances du flot de travaux arrêteront d'être lancées est demain à 15:50.

Commentaires

En fonction des valeurs utilisées, les instances du flot de travaux sont créées au moment de la génération du plan et elles sont exécutées en fonction des dates de début définies pour le flot de travaux. Elles ne sont pas créées de façon dynamique au cours de l'exécution du plan.

Exemples

1. Dans l'exemple suivant, une instance du flot de travaux JS_151439298 est exécutée toutes les 2 heures jusqu'à 18:00 à partir du 15 juin 2014 à 14:00.

```
SCHEDULE MDM021#JS_151439298
ON RUNCYCLE RC1 06/15/2014
( AT 1400 EVERY 0200 EVERYENDTIME 1800 )
:
MDM021#J_151439298
FIN
```

2. Dans l'exemple suivant, une instance du flot de travaux JS_0415 est exécutée toutes les 10 minutes jusqu'à 15:15 à partir du 16 juin 2014 à 15:45.

```
SCHEDULE MDM005#JS_0415
ON RUNCYCLE ACCTRC 06/16/2014
( SCHEDTIME 1445 DEADLINE 1530 EVERY 0010 EVERYENDTIME 1515 )
:
MDM005#JS_0415
FIN
```

every (utilisé pour les travaux)

Définit la fréquence de répétition d'un travail. Le travail est lancé de façon répétitive selon la fréquence indiquée. Si une dépendance du travail n'est pas satisfaite, l'itération n'a lieu qu'une fois la dépendance satisfaite. Le mot clé est indiqué dans la définition d'un travail.

Syntaxe

every *fréquence*

Arguments

fréquence

Fréquence de répétition exprimée en heures et en minutes, au format *hhmm*. La fréquence peut dépasser 24 heures. La valeur maximale prise en charge est de 99 heures et 59 minutes.

Commentaires

- L'itération **every** d'un travail ne s'arrête pas, même si l'une des répétitions du travail s'arrête de manière anormale.
- Si vous avez utilisé l'option **every** sans la dépendance **at**, les travaux réexécutés sont planifiés selon la fréquence **every** définie, à partir de l'heure de démarrage réel du travail.
- Dans le cas particulier où l'option **every** est utilisée avec la dépendance **at** et qu'une réexécution est retardée (pour une dépendance ou toute autre cause) alors, pendant que IBM Workload Scheduler se réaligne sur l'heure **at**, il est

possible qu'une ou deux itérations ne respectent pas la fréquence **every**. Dans tous les autres cas, la fréquence **EVERY** est toujours respectée.

L'exemple 2 explique comment IBM Workload Scheduler se réaligne sur l'heure **at** si le travail commence après l'heure **at** définie et si des itérations sont perdues.

- Si une instance **every** ne démarre pas à l'heure de début attendue, utilisez l'option **bm late every** pour définir le nombre maximal de minutes qui peuvent s'écouler avant que IBM Workload Scheduler n'ignore le travail. La valeur de l'option doit être définie dans le fichier <TWSHOME>/localopts :

bm late every = xx

Où **xx** est le nombre de minutes.

Cette option est locale pour chaque agent, c'est pourquoi elle doit être définie sur chaque agent tolérant aux pannes qui a **every** travaux dont le jeu d'options est **bm late every**.

L'option **bm late every** s'applique uniquement aux travaux pour lesquels sont définies à la fois l'option **every** et la dépendance de temps **at** et n'a aucun impact sur les travaux pour lesquels seule l'option **every** est définie. Seuls les travaux pour lesquels la fréquence **every** est supérieure à la valeur **bm late every** seront concernés.

L'exemple 4, à la page 243 montre le comportement de IBM Workload Scheduler lorsque le retard d'une instance **every** ne dépasse pas la valeur de l'option **bm late every**.

L'exemple 5, à la page 243 montre le comportement de IBM Workload Scheduler lorsque le retard d'une instance **every** dépasse la valeur de l'option **bm late every**.

L'exemple 6, à la page 243 montre le comportement de IBM Workload Scheduler lorsque la première instance d'un travail ne s'exécute pas à l'heure de début attendue et dépasse la valeur de l'option **bm late every**.

- Si le mot clé **every** est défini pour un travail lorsque l'heure d'été (DST) est désactivée, c'est-à-dire lorsque l'horloge est reculée d'une heure, la commande **every job** est avertie de ce changement et s'exécute également lors du deuxième intervalle de temps répété.

Exemples

1. L'exemple ci-dessous exécute le travail **testjob** toutes les heures :

testjob every 100

2. Dans l'exemple ci-dessous, le travail **testjob1** doit s'exécuter toutes les 15 minutes entre 18 et 20 heures :

testjob1 at 1800 every 15 until 2000

Le travail est censé être exécuté à 18:00, 18:15, 18:30, et ainsi de suite toutes les 15 minutes.

Si le travail est soumis adhoc à 18:33, il est exécuté à 18:33, 18:34, 18:45, etc. La raison est la suivante :

Notez tout d'abord que dans un travail, deux valeurs de temps doivent être prises en compte :

- **start_time**. Il s'agit de l'heure à laquelle il a été prévu d'exécuter le travail. Elle est définie sur l'heure **at** spécifiée pour le travail ou sur l'heure où la réexécution doit être lancée. Cette valeur peut être affichée à l'aide de **conman showjobs** avant que l'itération du travail ne démarre.

```

/      • time_started. Il s'agit de l'heure à laquelle le travail a réellement démarré
/      (18:33, par exemple). Cette valeur peut être affichée à l'aide de conman
/      showjobs une fois que l'itération du travail a démarré.

/      Etant donné que testjob1 a été soumis adhoc à 18:33, cette information s'affiche
/      tout de suite après la soumission :

/      avec conman showjobs
/      TESTJOB1 HOLD 1800

/      dans le fichier Symphony
/      start_time=1800 (parce que le travail est planifié pour s'exécuter à (at)
/      18:00)
/      time_started=NULL (parce que le travail n'a pas encore démarré)

/      Puisque l'heure de début, définie par start_time (1800), est plus petite que
/      l'heure en cours (1833), testjob1 démarre immédiatement et les informations
/      mises à jour sont alors :

/      avec conman showjobs
/      TESTJOB1 SUCC 1833

/      dans le fichier Symphony
/      start_time=1800 (parce que le travail était planifié pour s'exécuter à
/      (at) 18:00)
/      time_started=1833 (parce que le travail a démarré à 18:33)

/      Lorsque batchman calcule l'heure de la prochaine itération, il utilise les
/      données suivantes :
/      start_time=1800
/      rate=0015
/      current_time=1833
/      Etant donné que l'heure de la prochaine itération (1800+0015=1815) serait
/      encore antérieure à la valeur current_time (1833), batchman identifie la dernière
/      itération planifiée qui n'a pas été exécutée en ajoutant à l'heure start_time la
/      valeur de la fréquence every_rate autant de fois que possible sans dépasser
/      l'heure current_time
/      1800 + 0015 + 0015 = 1830 < 1833
/      puis lance la commande pour exécuter cette itération. En supposant que cette
/      itération est exécutée à 18:34, les informations sont les suivantes une fois que le
/      travail a commencé :

/      avec conman showjobs
/      TESTJOB1 SUCC 1834

/      dans le fichier Symphony
/      start_time=1830 (parce que cette itération du travail était planifiée
/      pour s'exécuter à (at) 18:30)
/      time_started=1834 (parce que cette itération du travail a démarré à
/      1834)

/      Une fois cette itération du travail terminée, batchman calcule à nouveau l'heure
/      à laquelle l'itération suivante doit démarrer en utilisant les valeurs mises à jour
/      suivantes :
/      start_time=1830

```



```

/      rate=0015
/      current_time=1834
/      Le fait que l'heure d'itération suivante (1830+0015=1845) est ultérieure à la
/      valeur current_time (1834) indique à batchman que l'itération a repris. L'heure
/      de l'itération, à partir de 18:45, peut à présent être réalignée sur les heures
/      d'itération prévues dans la définition du travail par les mots clés at et every.
/
/ 3. L'exemple suivant ne démarre pas l'itération du travail testjob2 tant que le
/      travail testjob1 n'a pas abouti.
/      testjob2 every 15 follows testjob1
/
/ 4. Dans l'exemple suivant, le retard d'une instance de travail every ne dépasse pas
/      la valeur de l'option bm late every :
/      bm late every = 10
/      JOB AT 1400 EVERY 0030
/
/      Ce travail est censé être exécuté à 14h00, à 14h30, à 15h00, et ainsi de suite
/      toutes les trente minutes.
/
/      Si le serveur est en panne de 14h35 à 16h05, les instances de 15h00, 15h30 et
/      16h00 ne s'exécutent pas. A 16h05, IBM Workload Scheduler redémarre.
/      Lorsqu'il analyse le fichier Symphony, il détermine que l'heure potentielle la
/      plus propice pour la prochaine instance de travail every est 16h00. IBM
/      Workload Scheduler vérifie si l'heure potentielle la plus propice (16h00) ne
/      dépasse pas le retard maximal autorisé pour un travail every (10 minutes).
/
/      Dans ce cas, le retard n'a pas dépassé l'option bm late every, c'est pourquoi
/      IBM Workload Scheduler adopte un comportement habituel et crée une
/      instance de travail every dont l'heure de début est 16h00. Les instances
/      suivantes sont à 16h30, 17h00 et ainsi de suite, toutes les trente minutes.
/
/ 5. Dans l'exemple suivant, le retard d'une instance d'un travail every dépasse la
/      valeur de l'option bm late every :
/      bm late every = 10
/      JOB AT 1400 EVERY 00030
/
/      Ce travail est censé être exécuté à 14h00, 14h30, 15h00 et ainsi de suite, toutes
/      les trente minutes.
/
/      Si le serveur est en panne de 14h35 à 16h20, les instances de 15h00, 15h30 et
/      16h00 ne s'exécutent pas. A 16h20, IBM Workload Scheduler redémarre.
/      Lorsqu'il analyse le fichier Symphony, il détermine que l'heure potentielle la
/      plus propice pour la prochaine instance de travail every est 16h00. IBM
/      Workload Scheduler vérifie si l'heure potentielle la plus propice (16h00) ne
/      dépasse pas le retard maximal autorisé pour une instance de travail every (10
/      minutes).
/
/      Dans ce cas, le retard dépasse l'option bm late every, c'est pourquoi IBM
/      Workload Scheduler adopte un nouveau comportement, il ne lance pas
/      l'instance de travail every à 16h00 et il crée une instance de travail every dont
/      l'heure de début est 16h30.
/
/ 6. L'exemple suivant montre le comportement de IBM Workload Scheduler
/      lorsque la première instance d'un travail ne s'exécute pas à l'heure de début
/      attendue et dépasse la valeur de l'option bm late every :
/      bm late every = 10
/      JOB AT 1400 EVERY 00030
/
/      Ce travail est censé être exécuté à 14h00, 14h30, 15h00 et ainsi de suite, toutes
/      les trente minutes.
/
/      Si le serveur est en panne de 10h00 à 14h15, la première instance de travail ne
/      s'exécute pas. A 14h15, IBM Workload Scheduler redémarre. Lorsqu'il analyse le

```

```

/      fichier Symphony, il détermine que la première instance de ce travail every ne
/      s'est pas exécutée. Dans ce cas, IBM Workload Scheduler lance le travail à
/      14h15.

```

```

/
/      except
/      Définit les dates qui constituent des exceptions par rapport aux dates définies par
/      le mot clé on d'un flot de travaux. Pour plus d'informations, voir «on», à la page
/      267.

```

Syntaxe

```

/      except [runcycle nom]
/      [validfrom date] [validto date]
/      [description "texte"]
/      {date | jour | agenda | demande | "icalendar" }
/      [...]
/      [fdignore | fdnext | fdprev][subset nom_sous-ensemble AND | OR]

```

Arguments

```

/      runcycle nom
/      Indique un libellé avec un nom usuel pour le cycle d'exécution spécifié
/      dans les lignes suivantes.

```

```

/      valid from date ... valid to date
/      Délimite la période au cours de laquelle le flot de travaux est actif,
/      c'est-à-dire pendant laquelle il est ajouté au plan de production. Notez que
/      la date spécifiée comme valeur valid to n'est pas incluse dans le cycle
/      d'exécution et le flot de travaux n'est donc pas actif à cette date.

```

```

/      description "texte"
/      Contient une description du cycle d'exécution.

```

```

/      date      Indique un cycle d'exécution qui s'exécute à des dates spécifiques. La
/      syntaxe utilisée pour ce type de cycle d'exécution est la suivante :

```

```

/      aaaammjj [,aaaammjj][...] Par exemple, pour un flot de travaux dont
/      l'exécution est prévue les 25 mai 2009 et 12 juin 2009, la valeur est la
/      suivante :

```

```

/      on
/      20090525,20090612

```

```

/      jour      Indique un cycle d'exécution qui s'exécute des jours spécifiques. La syntaxe
/      utilisée pour ce type de cycle d'exécution est la suivante :

```

```

/      {mo | tu | we | th | fr | sa | su} Par exemple, pour un flot de travaux qui est
/      planifié pour s'exécuter chaque lundi, la valeur est la suivante :

```

```

/      on
/      mo

```

```

/      agenda    Dates indiquées dans un agenda portant ce nom. Le nom de l'agenda peut
/      être suivi d'un décalage au format suivant :

```

```

/      {+ | -}n {day[s] | weekday[s] | workday[s]}

```

```

/      Où :

```

```

/      n          Indique le nombre de jours, jours de la semaine ou jours ouvrés.

```

```

/      days      Chaque jour de la semaine.

```

```

/      weekdays

```

```

/      Tous les jours de la semaine sauf le samedi et le dimanche.

```

```

/
/      workdays
/      Tous les jours de la semaine excepté les samedis et dimanches (sauf
/      indication contraire par le mot clé freedays) et les dates figurant
/      dans un agenda désigné des jours chômés ou dans l'agenda
/      holidays.
/
/      demande
/      Sélectionne le flot de travaux uniquement lorsque celui-ci est demandé. Ce
/      paramètre est utilisé pour les flots de travaux qui sont sélectionnés par
/      nom et non par date. Pour empêcher un flot de travaux planifié d'être
/      sélectionné pour le travail JnextPlan, redéfinissez-le ON REQUEST.
/
/      Remarque : Si vous tentez d'exécuter un flot de travaux contenant
/      plusieurs heures "à la demande" (on request), tenez compte des points
/      suivants :
/      • Les indicateurs "On request" ont la priorité sur les indicateurs "at".
/      • Les indicateurs "On request" n'ont jamais la priorité sur les indicateurs
/      "On".
/
/      icalendar
/      Représente une norme utilisée pour spécifier une règle récurrente qui
/      décrit le moment où un flot de travaux s'exécute.
/
/      La syntaxe utilisée pour un cycle d'exécution de type icalendar est la
/      suivante :
/
/      FREQ={DAYLY | WEEKLY | MONTHLY | YEARLY}
/
/      [:INTERVAL=[-]n]
/
/      [:{BYFREEDAY | BYWORKDAY | BYDAY=liste_jours_semaine |
/
/      BYMONTHDAY=liste_jours_mois}]
/
/      où la valeur par défaut du mot clé INTERVAL est 1.
/
/      A l'aide du mot clé icalendar, vous pouvez spécifier qu'un flot de travaux
/      s'exécute :
/
/      tous les n jours
/      en utilisant le format suivant :
/
/      FREQ=DAILY[:INTERVAL=n]
/
/      où la valeur définie pour valide de est le premier jour des dates
/      résultantes.
/
/      Par exemple, pour un flot de travaux planifié pour s'exécuter
/      chaque jour, la valeur est la suivante :
/
/      FREQ=DAILY
/
/
/      Pour un flot de travaux planifié pour s'exécuter un jour sur deux,
/      la valeur est la suivante :
/
/      FREQ=DAILY; INTERVAL=2
/
/      chaque jour chômé ou ouvré
/      en utilisant le format suivant :
/
/      FREQ=DAILY[:INTERVAL=n]
/
/      ;BYFREEDAY | BYWORKDAY
/
/      Par exemple, pour un flot de travaux planifié pour s'exécuter
/      chaque jour chômé, la valeur est la suivante :

```

```

/                                     FREQ=DAILY;BYFREEDAY

/                                     Pour un flot de travaux planifié pour s'exécuter un jour ouvré sur
/                                     deux, la valeur est la suivante :
/                                     FREQ=DAILY;INTERVAL=2;BYWORKDAY

/                                     toutes les n semaines, à des jours de la semaine spécifiques
/                                     en utilisant le format suivant :
/                                     FREQ=WEEKLY[;INTERVAL=n]
/                                     ;BYDAY=liste_jours_semaine
/                                     où la valeur définie pour liste_jours_semaine est
/                                     [SU] [,MO] [,TU] [,WE] [,TH] [,FR] [,SA]
/                                     Par exemple, pour un flot de travaux planifié pour s'exécuter
/                                     chaque semaine le vendredi et le samedi, la valeur est la suivante :
/                                     FREQ=WEEKLY;BYDAY=FR,SA

/                                     Pour un flot de travaux planifié pour s'exécuter toutes les trois
/                                     semaines, le vendredi, la valeur est la suivante :
/                                     FREQ=WEEKLY;INTERVAL=3;BYDAY=FR

/                                     tous les n mois, à des dates spécifiques
/                                     en utilisant le format suivant :
/                                     FREQ=MONTHLY[;INTERVAL=n]
/                                     ;BYMONTHDAY=liste_jours_mois
/                                     où la valeur définie pour liste_jours_mois est représentée par une
/                                     liste de
/                                     [+nombre_de_jours_à_partir_début_du_mois]
/                                     [-nombre_de_jours_à_partir_fin_du_mois]
/                                     [nombre_de_jours_du_mois]
/                                     Par exemple, pour un flot de travaux planifié pour s'exécuter le
/                                     27ème jour de chaque mois, la valeur est la suivante :
/                                     FREQ=MONTHLY;BYMONTHDAY=27

/                                     Pour un flot de travaux planifié pour s'exécuter tous les six mois,
/                                     le 15 du mois et le dernier jour du mois, la valeur est la suivante :
/                                     FREQ=MONTHLY;INTERVAL=6;BYMONTHDAY=15,-1

/                                     tous les n mois, à des jours spécifiques de la semaine
/                                     en utilisant le format suivant :
/                                     FREQ=MONTHLY[;INTERVAL=n]
/                                     ;BYDAY=numéro_de_la_semaine_du_mois_jour_semaine
/                                     où la valeur définie pour
/                                     numéro_de_la_semaine_du_mois_jour_semaine est représentée par une
/                                     liste de
/                                     [+numéro_semaine_à_partir_début_du_mois]
/                                     [-numéro_semaine_à_partir_fin_du_mois]
/                                     [jour_semaine]
/                                     Par exemple, pour un flot de travaux planifié pour s'exécuter
/                                     chaque mois, le premier lundi et le dernier vendredi du mois, la
/                                     valeur est la suivante :
/                                     FREQ=MONTHLY;BYDAY=1MO,-1FR

```

```

/
/      Pour un flot de travaux planifié pour s'exécuter tous les six mois,
/      le deuxième mardi du mois, la valeur est la suivante :
/      FREQ=MONTHLY; INTERVAL=6; BYDAY=2TU
/
/      toutes les n années
/      en utilisant le format suivant :
/      FREQ=YEARLY[;INTERVAL=n]
/
/      où la valeur définie pour valide de est le premier jour des dates
/      résultantes.
/
/      Par exemple, pour un flot de travaux planifié pour s'exécuter
/      chaque année, la valeur est la suivante :
/      FREQ=YEARLY
/
/      Pour un flot de travaux planifié pour s'exécuter tous les deux ans,
/      la valeur est la suivante :
/      FREQ=YEARLY; INTERVAL=2
/
|
|      Remarque : Les limitations suivantes s'appliquent :
|
|      • L'intervalle maximal pris en charge pour un cycle d'exécution quotidien
|      est de 31 jours.
|
|      • L'intervalle maximal pris en charge pour un cycle d'exécution
|      hebdomadaire est de 8 jours.
|
|      • L'intervalle maximal pris en charge pour un cycle d'exécution mensuel
|      est de 12 mois. Pour des cycles d'exécution indiquant le jour de la
|      semaine en fonction du mois, par exemple le troisième samedi ou le
|      deuxième vendredi, l'intervalle maximal pris en charge est de 5 jours.
|
|      • L'intervalle maximal pris en charge pour un cycle d'exécution annuel est
|      de 10 ans.
|
/
/      fdignore | fdnext | fdprev
/      Indique une règle qui doit être appliquée lorsque la date devant être exclue
/      correspond à un jour chômé. Il peut s'agir de l'une des règles suivantes :
/
/      fdignore
/      La date n'est pas exclue.
/
/      fdnext Le jour ouvré le plus proche suivant le jour chômé est exclu.
/
/      fdprev
/      Le jour ouvré le plus proche précédant le jour chômé est exclu.
/
/      subset nom_sous-ensemble
/      Indique le nom du sous-ensemble. Si vous ne spécifiez pas de nom,
/      SUBSET_1 est utilisé par défaut.
/
/      AND | OR
/      Par défaut, les cycles d'exécution dans un sous-ensemble sont dans une
/      relation logique OR mais vous pouvez la changer en relation logique AND
/      tant que le résultat du groupe de cycle d'exécution est une date ou un
/      ensemble de dates positif (inclusif).
/
/      Pour plus d'informations sur les autres mots clés contenus dans la syntaxe except,
/      voir «on», à la page 267.

```

Commentaires

Vous pouvez définir plusieurs instances du mot clé **except** pour le même flot de travaux. Chaque instance correspond à un cycle d'exécution auquel vous pouvez associer une règle de jour chômé.

Si une définition de flot de travaux contient plusieurs instances **except**, celles-ci doivent être consécutives.

Chaque instance du mot clé doit contenir l'une des valeurs autorisées par la syntaxe **except**.

Exemples

Dans l'exemple suivant, le flot de travaux testskd2 est sélectionné pour être exécuté tous les jours de la semaine, à l'exception des dates figurant dans les agendas intitulés monthend et holidays :

```
schedule testskd2 on weekdays  
except monthend,holidays
```

Dans l'exemple suivant, le flot de travaux testskd3 est sélectionné pour être exécuté tous les jours de la semaine à l'exception du 15 mai 2005 et du 23 mai 2005 :

```
schedule testskd3 on weekdays  
except 05/15/2005,05/23/2005
```

Dans l'exemple suivant, le flot de travaux testskd4 est sélectionné pour être exécuté tous les jours sauf pendant les deux jours précédant toute date figurant dans un agenda intitulé monthend :

```
schedule testskd4 on everyday  
except monthend-2 weekdays
```

Sélectionnez le flot de travaux sked4 de sorte qu'il s'exécute les lundis, mardis et 2 jours de la semaine avant chaque date répertoriée dans l'agenda monthend. Si la date d'exécution correspond à un jour chômé, exécutez le flot de travaux le jour ouvré suivant le plus proche. N'exécutez pas le flot de travaux le mercredi.

```
schedule sked4  
on mo  
on tu, MONTHEND -2 weekdays fdnext  
except we
```

Sélectionnez le flot de travaux testskd2 pour qu'il s'exécute chaque jour de la semaine sauf les jours répertoriés dans monthend. Si une date figurant dans monthend correspond à un jour chômé, excluez le jour de la semaine le plus proche précédant cette date. Dans cet exemple, les jours chômés sont les samedis, les dimanches et toutes les dates répertoriées dans l'agenda holidays par défaut.

```
schedule testskd2  
on weekdays  
except MONTHEND fdprev
```

follows

Définit les autres travaux et flots de travaux qui doivent aboutir avant le lancement d'un travail ou d'un flot de travaux.

Commentaires

Utilisez la syntaxe suivante pour les flots de travaux :

[follows {[*netagent::*][*Poste de travail*#]}*nom_flot_travaux*

[previous | sameday | relative from [+/-] *heure* **to** [+/-] *heure* **| from** *heure* [+/-*n* day[s]] **to** *heure* [+/-*n* day[s]]

Utilisez la syntaxe suivante pour les travaux :

[follows {[*netagent::*][*Poste de travail*#]}*nom_flot_travaux*{*nom_travail*}

[previous | sameday | relative from [+/-] *heure* **to** [+/-] *heure* **| from** *heure* [+/-*n* jour[s]] **to** *heure* [+/-*n* jour[s]] **[if** <condition> [**|** <condition>...]]

Arguments

agent_réseau

Nom de l'agent réseau dans lequel la dépendance interréseau est définie.

Poste de travail

Le poste de travail sur lequel s'exécute le travail ou le flot de travaux qui doit être terminé. Par défaut, il s'agit du même poste de travail que le travail ou le flot de travaux dépendant.

Si aucun *poste de travail* n'est spécifié avec l'*agent réseau*, le poste de travail auquel l'agent réseau est connecté est utilisé par défaut.

nom_flot_travaux

Le nom du flot de travaux qui doit être terminé. Pour un travail, il s'agit par défaut du même flot de travaux que le travail dépendant.

heure Indique une heure. Les valeurs admises sont comprises entre 0000 et 2359.

nomtravail

Le nom du travail qui doit être terminé. Un signe at (@) peut être utilisé pour indiquer que toutes les tâches du flot de travaux doivent se terminer correctement.

previous | sameday | relative from [+/-] *heure* **to** [+/-] *heure* **| from** *heure* [+/-*n* day[s]] **to** *heure* [+/-*n* day[s]]

Définit la manière dont le flot de travaux ou un travail référencé par une dépendance de prédécesseur/successeur externe correspond à un flot de travaux particulier ou à une instance de travail dans le plan. Pour une explication détaillée de ces options, voir «Commentaires», à la page 250.

[if <condition> [**|** <condition>...]]

Les dépendances conditionnelles sont utilisées lorsque vous souhaitez qu'un travail ou un flot de travaux successeur ne démarre qu'une fois que le travail ou flot de travaux prédécesseur a satisfait certaines conditions. Ces conditions permettent également de définir des flux alternatifs dans un flot de travaux à partir d'un travail ou d'un flot de travaux prédécesseur. Le travail successeur est déterminé en fonction des conditions que le travail ou flot de travaux prédécesseur satisfait. Dans le cas où les conditions de sortie ne sont pas satisfaites, les travaux du flot qui ne s'exécutent pas se retrouvent dans l'état SUPPR. Cet état se distingue des dépendances classiques, pour lesquelles les travaux se retrouvent mis en attente jusqu'à ce que le prédécesseur se retrouve dans l'état SUCC.

Les conditions exprimées par [if <condition> [! <condition>...]] peuvent être de différents types : basées sur le statut d'exécution du travail ou du flot de travaux prédécesseur, le statut du travail et d'autres conditions basées sur la sortie ou le résultat du travail ou du flot de travaux prédécesseur. Vous pouvez indiquer plusieurs conditions, séparées d'une barre verticale (|), mais toutes du même type ; vous ne pouvez pas indiquer une combinaison de types. Par exemple, IF ABEND | FAIL | SUPPR.

Lorsque le prédécesseur est un flot de travaux, la dépendance conditionnelle ne peut être qu'une condition de statut, comme suit : abend, succ et suppr. Le travail successeur s'exécute lorsque le statut du flot de travaux prédécesseur satisfait le statut de travail indiqué à l'aide de ces arguments. Vous pouvez définir un statut, une combinaison de statuts, ou l'ensemble des statuts. Lorsque vous indiquez plusieurs statuts ou noms de condition, séparez-les d'une barre verticale (|).

if nom_condition

Où *nom_condition* correspond à un statut ou un nom réel attribué à une condition que vous définissez.

if exec Le travail successeur s'exécute quand le travail prédécesseur a démarré.

if fail | abend | succ | suppr

Le travail successeur s'exécute quand le statut du travail prédécesseur correspond au statut défini à l'aide de ces arguments. Vous pouvez définir un statut, une combinaison de statuts, ou l'ensemble des statuts. Pour définir plus d'un statut, séparez chaque statut à l'aide d'une barre verticale (|).

if nom_condition]

Le travail successeur s'exécute lorsque le travail prédécesseur satisfait les conditions de sortie définies par la variable *Nom_Condition*. Vous pouvez indiquer un nom de condition, ou une combinaison de noms. Lorsque vous indiquez plusieurs noms de condition, séparez-les d'une barre verticale (|). Ces conditions de sortie sont initialement définies dans la définition de travail.

Commentaires

Les critères de résolution des dépendances permettent de définir la manière dont le flot de travaux ou un travail référencé par une dépendance de prédécesseur/successeur externe correspond à un flot de travaux particulier ou à une instance de travail dans le plan. Parce que le plan autorise l'inclusion de plusieurs instances du même travail ou flot de travaux, vous pouvez identifier l'instance qui résout la dépendance de prédécesseur/successeur externe en fonction du critère de résolution suivant :

previous

La plus proche qui précède : l'instance de travail ou de flot de travaux qui résout la dépendance est la plus proche précédant l'instance qui inclut la dépendance.

sameday

Même jour : l'instance de travail ou de flot de travaux qui résout la

/ dépendance est la plus proche dans le temps planifiée pour commencer le
/ jour où l'instance qui inclut la dépendance est planifiée pour exécution.

/ **relative from [+/-] heure to [+/-] heure**

/ Présente dans un intervalle relatif : l'instance de travail ou de flot de
/ travaux qui résout la dépendance est la plus proche, dans un intervalle de
/ temps de votre choix, définie relativement à l'heure de début planifié de
/ l'instance dépendante.

/ **from heure [+/-n day[s]] to heure [+/-n day[s]]**

/ Présente dans un intervalle absolu : l'instance de travail ou de flot de
/ travaux qui résout la dépendance est la plus proche dans un intervalle de
/ temps de votre choix. L'intervalle de temps n'est pas lié à l'heure de début
/ planifié de l'instance dépendante.

/ Quels que soient les critères de correspondance utilisés, si plusieurs instances d'un
/ flot de travaux predecessor potentielle existent dans l'intervalle de temps spécifié,
/ la règle utilisée par le produit pour identifier l'instance predecessor correcte est la
/ suivante :

- / 1. IBM Workload Scheduler recherche l'instance la plus proche précédant l'heure
/ de début du travail ou du flot de travaux dépendant. Si une telle instance
/ existe, il s'agit de l'instance prédécesseur.
- / 2. S'il n'y a pas d'instance précédente, IBM Workload Scheduler considère
/ l'instance prédécesseur correcte comme étant l'instance la plus proche qui
/ commence après l'heure de début du travail ou du flot de travaux dépendant.

/ L'agenda classe les dépendances de prédécesseur/successeur comme *internes*
/ lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il
/ les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le
/ format *jobStreamName.workstationName.jobName*.

/ Lorsqu'un flot de travaux inclut un travail possédant une dépendance de
/ prédécesseur/successeur qui partage le même nom de flot de travaux (par
/ exemple, le flot de travaux schedA inclut un travail appelé job6 qui possède une
/ dépendance de prédécesseur/successeur sur schedA.job2), la dépendance est
/ ajoutée au plan en tant que dépendance de prédécesseur/successeur *externe*.
/ Depuis la version 8.3, contrairement aux versions précédentes, étant donné que
/ l'agenda utilise le critère de correspondance *sameday* pour résoudre des
/ dépendances externes, les dépendances générées de cette manière ne sont jamais
/ ajoutées lors de la première soumission de l'objet.

/ Pour davantage d'informations et d'exemples sur la façon dont les dépendances de
/ prédécesseur/successeur externes sont résolues dans le plan, voir «Gestion des
/ dépendances externes de prédécesseur/successeur des travaux et des flots de
/ travaux», à la page 71.

/ **Exemples**

/ L'exemple suivant indique que le flot de travaux skedc ne doit pas être lancé tant
/ que l'instance de flot de travaux sked4 précédente la plus proche sur le poste de
/ travail site1 n'a pas abouti :

/ `schedule skedc on fr follows site1#sked4 previous`

/ L'exemple suivant indique que le flot de travaux skedc ne doit pas être lancé tant
/ que l'instance de flot de travaux sked4 sur le poste de travail site1 qui s'exécute
/ entre 12:00, 3 jours avant, jusqu'à 3:00 un jour après n'a pas abouti :

```

/      schedule skedc on fr follows site1#sked4 from 1200 -3 days to 0300 1 day
/
/      L'exemple suivant indique que le flot de travaux skedc ne doit pas être lancé tant
/      que le flot de travaux sked4 sur le poste de travail site1 et le travail joba du flot
/      de travaux sked5 sur le poste de travail site2 n'ont pas abouti :
/
/      schedule skedc on fr
/      follows site1#sked4,site2#sked5.jobx
/
/      Ne lancez pas sked6 tant que le travail jobx du flot de travaux skedx sur l'agent
/      réseau cluster4 n'a pas abouti :
/
/      sked6 follows cluster4::site4#skedx.jobx
/
/      L'exemple suivant indique que le travail jobd ne doit pas être lancé tant que le
/      travail joba du même flot de travaux et le travail job3 du flot de travaux skeda
/      n'ont pas abouti :
/
/      jobd follows joba,skeda.job3
/
/      L'exemple suivant fait exécuter le travail LOADDATA_INFO après le travail CHECKJOB
/      dans le flot de travaux CHECKDATA uniquement si le travail CHECKJOB aboutit à l'état
/      FAIL ou ABEND et si le travail CHECKJOB1 du flot de travaux CHECKDATA1 satisfait la
/      condition STATUS_OK définie dans la définition de travail CHECKJOB1 :
/
/      WK1#LOADDATA_INFO
/      FOLLOWS WK2#CHECKDATA.CHECKJOB IF FAIL|ABEND
/      FOLLOWS W32#CHECKDATA1.CHECKJOB1 IF STATUS_OK
/
/      freedays
/
/      Utilisez freedays pour indiquer le nom d'un agenda des jours chômés dans lequel
/      figurent les jours non travaillés dans votre entreprise. Les conditions d'exécution
/      d'un flot de travaux pendant ces jours particuliers sont définies dans une règle
/      freedays lors de la configuration du cycle d'exécution. IBM Workload Scheduler
/      utilise ce type d'agenda comme agenda de base pour calculer les jours ouvrés pour
/      le flot de travaux.
/
/      Le mot clé affecte uniquement la planification des flots de travaux pour lesquels il
/      est indiqué.
/
/      Syntaxe
/
/      freedays Nom_agenda [-sa] [-su]
/
/      Arguments
/
/      Nom_agenda
/
/      Nom de l'agenda devant être utilisé comme agenda des jours chômés pour
/      le flot de travaux. Si Nom_agenda ne se trouve pas dans la base de données,
/      IBM Workload Scheduler génère un message d'erreur lorsque vous
/      sauvegardez le flot de travaux. Si Nom_agenda ne se trouve pas dans la
/      base de données lors de l'exécution de la commande schedulr, IBM
/      Workload Scheduler génère un message d'erreur et utilise l'agenda par
/      défaut holidays à la place. N'utilisez pas les noms des jours de la semaine
/      pour nommer un agenda.
/
/      -sa      Le samedi est considéré comme un jour ouvré.
/
/      -su      Le dimanche est considéré comme un jour ouvré.

```

Commentaires

Si vous spécifiez un agenda des jours chômés dans la définition du flot de travaux, le concept *workdays* prend la valeur suivante : *workdays* = tous les jours sauf le samedi et le dimanche (à moins que vous n'ayez indiqué -sa ou -su avec *freedays*) et toutes les dates de *Nom_agenda*

Si vous ne précisez pas **freedays** dans la définition du flot de travaux, alors : *workdays* = tous les jours sauf le samedi, le dimanche et les congés

Par défaut, le *samedi* et le *dimanche* sont considérés comme des jours chômés, sauf si vous indiquez le contraire en ajoutant -sa, -su ou les deux après *Nom_agenda*.

Exemples

Sélectionnez le flot de travaux sked2 pour qu'il s'exécute le 01/01/2005 ainsi que tous les jours ouvrés ne figurant pas sur l'agenda des jours chômés nommé GERMHOL.

```
schedule sked2
freedays GERMHOL
on 01/01/2005, workdays
```

Sélectionnez le flot de travaux sked3 pour qu'il s'exécute deux jours ouvrés avant chaque date de l'agenda PAYCAL. Les jours ouvrés incluent tous les jours de la semaine, du lundi au samedi, non répertoriés dans l'agenda des jours chômés nommé USAHOL.

```
schedule sked3
freedays USAHOL -sa
on PAYCAL -2 workdays
```

Sélectionnez le flot de travaux sked3 pour les dates répertoriées dans l'agenda APDATES. Si une date sélectionnée correspond à un jour chômé, n'exécutez pas le flot de travaux. Dans le présent exemple, les dimanches et toutes les dates répertoriées dans l'agenda GERMHOL sont considérés comme jours chômés. A l'exception des dates spécifiques répertoriées dans GERMHOL, tous les jours de la semaine du lundi au samedi sont des jours ouvrés.

```
schedule sked3
freedays GERMHOL -sa
on APDATES fdignore
```

Sélectionnez le flot de travaux testsked3 pour qu'il s'exécute chaque jour de la semaine sauf le 15/5/2005 et le 23/5/2006. Si le 23/05/2006 correspond à un jour chômé, ne l'excluez pas. Dans le présent exemple, les samedis, dimanches et toutes les dates répertoriées dans GERMHOL doivent être considérés comme des jours chômés. A l'exception des dates spécifiques répertoriées dans GERMHOL, tous les jours de la semaine du lundi au vendredi sont des jours ouvrés.

```
schedule testskd3
freedays GERMHOL
on weekdays
except 5/15/2005 fdignore
except 5/23/2006
```

Sélectionnez le flot de travaux testsked4 pour qu'il s'exécute tous les jours sauf 2 jours de la semaine avant chaque date répertoriée dans l'agenda MONTHEND. Si la date à exclure correspond à un jour chômé, ne l'excluez pas, mais excluez le jour ouvré suivant le plus proche. Dans l'exemple suivant, les jours chômés

```

/ correspondent à toutes les dates répertoriées dans USAH0L, tandis que les jours
/ ouverts sont tous les jours de la semaine du lundi au dimanche ne figurant pas
/ dans USAH0L.
/
/ schedule testskd4
/ freedays USAH0L -sa -su
/ on everyday
/ except MONTHEND -2 weekdays fdnext

```

job statement

Les travaux peuvent être définis dans la base de données indépendamment (voir «Travail», à la page 940) ou dans le cadre des flots de travaux. Dans l'un ou l'autre cas, les modifications sont apportées dans la base de données et n'ont pas d'incidence sur le plan de production jusqu'au début d'un nouveau plan de production.

Syntaxe

Pour définir un travail comme faisant partie d'un flot de travaux, utilisez la syntaxe suivante dans la définition du flot de travaux :

```

/ [poste_de_travail#]nom_travail [as nomveau_nom]
/ {scriptname nom_fichier | docommand "commande"}
/ streamlogon nom_utilisateur
/ [description "description"]
/ [tasktype type_tâche]
/ [interactive]
/ [succoutputcond nom_condition "valeur_condition"]
/ [outputcond nom_Condition "valeur_Condition"]
/ [recovery
/ {stop
/ [after [poste_de_travail#]nom_travail]
/ [abendprompt "texte"]}
/ | continue
/ [after [poste_de_travail#]nom_travail]
/ [abendprompt "texte"]}
/ | rerun [même_poste_de_travail]
/ [[repeatevery hhmm] [for nombre tentatives]]
/ [after [poste_de_travail#]nom_travail]
/ | [after [poste_de_travail#]nom_travail]
/ [abendprompt "texte"]}
=
=
=
=

```

Pour utiliser un travail déjà défini dans la base de données dans la définition du flot de travaux, définissez *instruction_travail* en utilisant la syntaxe suivante :

```

/ [poste_de_travail#]nom_travail [as nomveau_nom]

```

Arguments

as Nom à utiliser pour faire référence à l'instance de travail dans ce flot de travaux.

Pour les autres mots clés, voir «Définition de travail», à la page 187.

Commentaires

Lors de la définition d'un travail comme faisant partie d'un flot de travaux, l'ajout de la définition du flot de travaux à la base de données entraîne celui de la définition du nouveau travail, laquelle peut désormais être référencée depuis d'autres flots de travaux.

Remarque : Les mots clés saisis incorrectement dans les définitions de travail entraînent le stockage de définitions de travaux tronquées dans la base de données. En fait, le mot clé erroné est considéré comme externe à la définition de travail et il est interprété comme le nom de travail d'une définition de travail supplémentaire. Généralement, cette mauvaise interprétation entraîne également une erreur de syntaxe ou une erreur de définition de travail inexistant pour la définition de travail supplémentaire.

Lors de l'ajout ou de la modification d'un flot de travaux, les attributs ou les options de reprise de ses travaux sont également ajoutés ou modifiés. N'oubliez pas que lors de l'ajout ou du remplacement d'un flot de travaux, les éventuelles modifications apportées aux travaux ont une incidence sur tous les autres flots de travaux qui les utilisent. Le rapport de références croisées, *xref*, peut être utilisé pour déterminer les noms des flots de travaux qui contiennent un travail particulier.

Pour plus d'informations sur le rapport de références croisées, voir «xref», à la page 763.

Remarque : Les travaux planifiés pour être exécutés sur des postes de travail marqués comme étant *ignorés*, et appartenant à des flots de travaux planifiés pour s'exécuter sur des postes de travail actifs, sont ajoutés au plan bien qu'ils ne soient pas traités.

Exemples

L'exemple suivant définit un flot de travaux comportant trois travaux précédemment définis :

```
schedule bkup on fr at 20:00:
  cpu1#jbk1
  cpu2#jbk2
  needs 1 tape
  cpu3#jbk3
  follows jbk1
end
```

La définition de flot de travaux suivante contient des instructions de travail qui ajoutent ou modifient les définitions de deux travaux de la base de données :

```
schedule sked4 on mo:
  job1 scriptname "d:\apps\maestro\scripts\jcljob1"
  streamlogon jack
  recovery stopabendprompt "continue production"
  site1#job2 scriptname "d:\apps\maestro\scripts\jcljob2"
  streamlogon jack
  follows job1
end
```

/

/ / /

/

/

/

/

/

/

/

/ /

/ /

/

/ /

/

/ /

/ /

/

/ /

=

$$=$$
$$=$$
$$=$$

=

/

/ /

1. Une dépendance de prédécesseur/successeur sur un flot de travaux à l'état ABEND ou SUPPRESS.
2. Un ensemble de dépendances jointes qui contient deux dépendances, la première sur le travail nc007108#COND_DEP_JS1.COND_DEP_J1, qui doit se terminer avec succès pour que la dépendance soit satisfaite, la seconde sur le travail nc007108#COND_DEP_JS2.COND_DEP_J2, qui doit satisfaire la condition EXT_STATUS_PREREQ_SUCC_2. Cette condition est définie par l'utilisateur lors de la création de la définition de travail. L'une de ces conditions au moins doit être remplie pour que l'ensemble de dépendances conditionnelles jointes soit satisfait.
3. Une dépendance de prédécesseur/successeur sur un flot de travaux externe à l'état EXEC.
4. Un second ensemble de dépendances jointes qui contient deux dépendances, la première sur le travail CDJ_PRED_2_163955532, qui doit être à l'état SUCC, et la seconde sur le travail CDJ_PRED_1_163955532, qui doit être à l'état INT_STATUS_PREDEC_SUCC_1. La seconde condition est définie par l'utilisateur lors de la création de la définition de travail. Ces conditions doivent être remplies pour que l'ensemble de dépendances conditionnelles jointes soit satisfait.

```

:
SCHEDULE nc007108#CDJS_163955532
FOLLOWS nc007108#COND_DEP_JS1.@ IF ABEND | SUPPRESS

JOIN JOINDEP_COV_01 1 OF
DESCRIPTION "Description for join JOINDEP_COV_01"
FOLLOWS nc007108#COND_DEP_JS1.COND_DEP_J1 IF SUCC
FOLLOWS nc007108#COND_DEP_JS2.COND_DEP_J2 IF EXT_STATUS_PREREQ_SUCC_2
ENDJOIN

:
nc007108#CDJ_PRED_1_163955532
nc007108#CDJ_PRED_2_163955532

FTA_nc007108#CDJ_FTA_163955532
FOLLOWS nc007108#COND_DEP_JS1.COND_DEP_J1 IF EXEC

JOIN JOINDEP_COV_02 ALL OF
FOLLOWS CDJ_PRED_2_163955532 IF SUCC
FOLLOWS CDJ_PRED_1_163955532 IF INT_STATUS_PREDEC_SUCC_1
ENDJOIN

END

```

jsuntil

Le mot clé **jsuntil** définit la dernière heure de lancement d'un flot de travaux. Il détermine également le comportement des travaux dans le flot de travaux lorsque ce dernier approche de sa dernière heure de lancement. Utilisez **jsuntil** pour éviter que le flot de travaux soit supprimé, annulé ou défini sur continue (en fonction de l'action indiquée dans le mot clé **onuntil**) s'il commence avant sa dernière heure de lancement. Par exemple, si vous avez un flot de travaux avec **jsuntil** défini sur 10h00 et que l'un des travaux commence son exécution à 9h59, le travail et ses successeurs sont exécutés conformément à leur planification.

Ce mot clé et le mot clé **until** s'excluent mutuellement.

Il existe également une différence importante entre les mots clés **until** et **jsuntil** :

Si vous spécifiez le mot clé until dans votre définition de flot de travaux

Ce mot clé est évalué également après que le flot de travaux a été lancé.

Par conséquent, si la dernière heure de début expire avant que le flot de travaux se termine avec succès, l'action indiquée dans le mot clé **onuntil** associé est effectuée sur le flot de travaux et ses travaux qui n'ont pas encore été lancés.

Si vous spécifiez le mot clé **jsuntil** dans votre définition de flot de travaux

Ce mot clé n'est évalué qu'une fois, dès que toutes les dépendances du flot de travaux sont satisfaites et que ce dernier passe à l'état READY. Si la dernière heure de début définie à l'aide du mot clé **jsuntil** n'a pas encore expiré, elle n'est plus évaluée et le flot de travaux est exécuté indépendamment d'elle. Toutefois, pour empêcher que le flot de travaux ne reste indéfiniment à l'état READY, deux jours après que l'heure spécifiée dans le mot clé **jsuntil** ait été atteinte, le flot de travaux est supprimé par défaut.

Pour plus d'informations sur le mot clé **until**, voir «until», à la page 287.

Syntaxe

[jsuntil *heure* [**timezone** | **tz** *fuseau_horaire*] [**+nday[s]**] [**onuntil***action*]

Arguments

heure Indique l'heure. Les valeurs admises sont comprises entre 0000 et 2359.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

onuntil *action*

L'action à prendre sur un flot de travaux dont le temps until a expiré alors que ce flot de travaux ne s'est pas encore terminé à l'état SUCC. Les valeurs possibles du paramètre de l'*action* sont :

suppr Le travail ou le flot de travaux et tous les travaux ou flots de travaux dépendants ne s'exécutent pas. Il s'agit du comportement par défaut.

Une fois que l'heure until est arrivée à expiration pour un flot de travaux, l'état de ce dernier est calculé en fonction des règles habituelles ; les travaux supprimés ne sont pas pris en compte dans ce calcul. Dans le cas où le flot de travaux contient au moins un travail **every**, il est à l'état HOLD.

Une fois que l'heure until arrive à expiration pour un travail, ce dernier passe à l'état HOLD ou conserve tout état antérieur qui soit un état final.

Si l'heure **until** est transmise avec les options **onuntil suppr** et **carryforward**, le flot de travaux est *reporté* par **JnextPlan** uniquement si la date **until** correspond à la date d'exécution de JnextPlan. Si les dates d'exécution de **until** et JnextPlan ne sont pas identiques, le flot de travaux n'est pas *reporté*.

cont Le travail ou flot de travaux s'exécute lorsque toutes les conditions requises sont remplies. Un message apparaît alors dans le journal lorsque l'heure définie par until est dépassée.

Si l'heure **until** est transmise avec les options **onuntil cont** et **carryforward**, le flot de travaux est toujours *reporté* par **JnextPlan**.

canc Un travail ou flot de travaux est annulé lorsque l'heure **until** définie est atteinte. Lorsque le mot clé **onuntil canc** est utilisé sur des travaux, l'opération d'annulation sur le travail est émise par l'agent FTA sur lequel ce travail s'exécute. Tout travail ou flot de travaux qui dépendait de l'accomplissement d'un travail ou flot de travaux qui a été annulé, s'exécute car la dépendance n'est plus appliquée.

Si l'heure **until** est transmise avec les options **onuntil canc** et **carryforward**, le flot de travaux n'est pas *reporté* par **JnextPlan** car il est déjà annulé.

Remarque : Lorsque le mot clé **onuntil canc** est utilisé au niveau d'un flot de travaux, définissez comme propriétaire du flot de travaux le poste de travail le plus haut placé dans la hiérarchie de l'environnement de planification parmi tous les postes de travail qui possèdent des travaux appartenant à ce flot de travaux.

Commentaires

Le mot clé **jsuntil** est pris en charge sur les composants aux niveaux de version 9.4 groupe de correctifs ou ultérieurs, à l'exception des agents dynamiques connectés à un maître à un niveau de version 9.4 groupe de correctifs 1 ou ultérieur. Sur tous les autres types d'agent où une version antérieure du produit est installée, le mot clé **jsuntil** est ignoré.

Exemples

Pour planifier le flot de travaux BankReports afin qu'il soit lancé à 08 h 00 et continue à s'exécuter si l'un de ses travaux commence son exécution à 9 h 59 et que celle-ci se prolonge au-delà de 10 h 00, spécifiez la syntaxe suivante :

```
schedule BankReports
on everyday
AT 0800 JSuntil 1000
end
```

keyjob

Le mot clé **keyjob** est utilisé pour indiquer qu'un travail est une clé dans la base de données, ainsi que dans le plan, et qu'il doit être surveillé par des applications telles que Tivoli Business Systems Manager ou Tivoli Enterprise Console. Pour plus d'informations sur l'activation du mécanisme des indicateurs clés, voir le guide *IBM Workload Scheduler - Intégration à d'autres produits*.

Syntaxe

keyjob

Exemples

Dans l'exemple suivant, le travail et le flot de travaux sont tous deux indiqués comme clé.

```

/      SCHEDULE cpu1#sched1
/      ON everyday
/      KEYSCHED
/      AT 0100
/      cpu1#myjob1 KEYJOB
/      FIN

```

keysched

Le mot clé **keysched** est utilisé pour indiquer qu'un travail est une clé dans la base de données, ainsi que dans le plan, et qu'il doit être surveillé par des applications telles que Tivoli Business Systems Manager. Pour plus d'informations sur l'activation du mécanisme des indicateurs clés, voir le guide *IBM Workload Scheduler - Intégration à d'autres produits*.

Syntaxe

keysched

Exemples

Voici un exemple :

```

/      SCHEDULE cpu1#sched1
/      ON everyday
/      KEYSCHED
/      AT 0100
/      cpu1#myjob1 KEYJOB
/      FIN

```

limite

Le mot clé **limit** limite le nombre maximal de travaux pouvant être exécutés simultanément dans un flot de travaux. Ce mot clé fonctionne seulement si tous les travaux du flot de travaux sont définis sur des postes de travail qui sont gérés par le même processus batchman. Trois cas sont possibles :

Le poste de travail sur lequel le batchman s'exécute est un agent tolérant aux pannes

Dans ce cas, le mot clé **limit** limite le nombre de travaux, définis dans l'agent tolérant aux pannes ou dans les agents étendus définis sur cet agent tolérant aux pannes, pouvant s'exécuter simultanément dans le flot de travaux. Le flot de travaux doit contenir uniquement des travaux définis dans l'agent tolérant aux pannes ou dans les agents étendus qui s'exécutent sur l'agent tolérant aux pannes.

Le poste de travail sur lequel batchman s'exécute est un gestionnaire de domaine.

Dans ce cas, outre les postes de travail du cas précédent, le flot de travaux peut contenir des travaux s'exécutant sur des agents standard connectés au gestionnaire de domaine, ou à n'importe quel agent tolérant aux pannes géré par ce gestionnaire de domaine, configurés avec le paramètre FULLSTATUS défini sur **on**.

Le poste de travail sur lequel le batchman s'exécute est un gestionnaire de domaine dynamique.

En plus des postes de travail des cas précédents, le flot de travaux peut contenir des travaux définis sur le courtier et ses agents dynamiques.

Dans d'autres scénarios, l'utilisation du mot clé **limit** pour limiter le nombre de travaux pouvant s'exécuter simultanément peut ne pas fonctionner comme prévu.

Syntaxe

limit *nombre_max_travaux*

Arguments

nombre_max_travaux

Indique le nombre maximal de travaux qui peuvent être exécutés en même temps dans le flot de travaux. Les valeurs admises sont comprises entre 0 et 1024. Si vous indiquez 0, vous empêchez le lancement de tous les travaux, même de ceux pour lesquels la priorité **GO** ou **HI** est définie.

Exemples

Dans l'exemple suivant, le nombre de travaux pouvant être exécutés simultanément dans le flot de travaux sked2 est limité à cinq :

```
schedule sked2 on fr
  limit 5;
```

matching

Définit une valeur par défaut pour les critères de correspondance à utiliser dans toutes les dépendances de prédécesseur/successeur lorsqu'aucun critère de correspondance n'est défini dans la définition du flot de travaux ou dans les travaux contenus dans le flot de travaux.

Syntaxe

matching {**previous** | **sameday** | **relative from** [+/-] *heure* **to** [+/-] *heure*}

Arguments

Pour plus d'informations sur le mot clé utilisé avec **matching** voir le mot clé «follows», à la page 248.

Exemples

L'exemple suivant montre la définition du flot de travaux SCHED2 qui :

- contient un travail job1 pouvant s'exécuter aujourd'hui uniquement s'il a été exécuté la veille ;
- nécessite que l'instance du flot de travaux SCHED1 qui s'exécute le même jour ait abouti avant de pouvoir s'exécuter.

```
SCHEDULE PDIVITA1#SCHED2
ON RUNCYCLE RULE1 "FREQ=DAILY;"
ON RUNCYCLE CALENDAR2 CAL1
MATCHING PREVIOUS
FOLLOWS PDIVITA1#SCHED1.@ SAMEDAY
FOLLOWS PDIVITA1#SCHED2.JOB1
:
PDIVITA1#JOB1

PDIVITA1#JOB2
FIN
```

Dans cet exemple, la dépendance de prédécesseur/successeur externe de PDIVITA1#SCHED2.JOB1 hérite des critères de correspondance spécifiés dans le mot clé **matching**.

Commentaires

Remarque : si vous supprimez un flot de travaux, puis l'ajoutez à nouveau à la base de données, le flot de travaux obtient un autre identificateur. Par conséquent, si le flot de travaux contient des dépendances **FOLLOWS** avec des critères **PREVIOUS**, ces dépendances ne sont pas mises en correspondance lorsque **JnextPlan** s'exécute à nouveau, car il s'agit de dépendances entre les plans qui font référence à un ancien identificateur.

Dans l'exemple suivant, si vous supprimez le flot de travaux JS01 afin de garantir l'intégrité référentielle de la base de données, **FOLLOWS TWS851MASTER#JS01.@** est également supprimé de la définition de JS02 et du plan de préproduction.

Si vous supprimez le flot de travaux JS03 afin de garantir l'intégrité référentielle de la base de données, **FOLLOWS TWS851MASTER#JS03.@** est également supprimé de la définition de JS02 et du plan de préproduction.

Si vous supprimez le flot de travaux JS02, puis l'ajoutez à nouveau au plan, ses dépendances **FOLLOWS** sont également ajoutées à nouveau. Lorsque le plan est étendu, la dépendance **FOLLOWS TWS851MASTER#JS03.@ PREVIOUS** du flot de travaux JS02 ne correspond pas à l'instance du flot de travaux JS03 provenant du plan précédent et cette dépendance n'est pas ajoutée.

A l'extension de plan suivante, le processus fonctionne à nouveau.

```
SCHEDULE TWS851MASTER#JS01
ON RUNCYCLE RULE1 "FREQ=DAILY;"
:
TWS851MASTER#J02
END
SCHEDULE TWS851MASTER#JS03
ON RUNCYCLE RULE1 "FREQ=DAILY;"
SCHEDTIME 1000
CARRYFORWARD
:
TWS851MASTER#J03
END
SCHEDULE TWS851MASTER#JS02
ON RUNCYCLE RULE1 "FREQ=DAILY;"
:
TWS851MASTER#J01
FOLLOWS TWS851MASTER#JS01.@
FOLLOWS TWS851MASTER#JS03.@ PREVIOUS
END
```

Pour éviter ce problème, utilisez la commande Composer **replace**. Dans ce cas, les identificateurs du flot de travaux ne sont pas modifiés.

maxdur

Indique la durée maximale d'exécution d'un travail. Vous pouvez exprimer cette durée en minutes, ou sous la forme d'un pourcentage de la durée estimée du travail la plus récente. Si un travail est en cours d'exécution et que la durée maximale a été dépassée, les actions suivantes sont réalisées :

- L'une des actions suivantes est déclenchée : Kill ou Continue.
- La durée du travail apparaît comme étant dépassée dans les cas suivants :
 - Lors de l'exécution de **showjob** à partir de la ligne de commande **conman**, **MaxDurationExceeded** s'affiche.
 - A partir de Dynamic Workload Console dans les propriétés du travail.

/ – Un message d'information est consigné dans le fichier *rép_base_TWS/stdlist/*
/ *logs/aaaammjj_TWSMERGE.log*.

/ Si ce travail est toujours en cours d'exécution lorsque JnextPlan s'exécute, il est
/ inséré dans le flot de travaux USERJOBS. Le paramètre de durée maximale n'est
/ pas conservé pour le travail dans le flot de travaux USERJOBS et ne sera pas
/ surveillé. Pour que le flot de travaux soit reporté et pour éviter que le travail ne
/ soit déplacé vers le flot de travaux USERJOBS, marquez le flot de travaux initial où
/ le paramètre de durée maximale a été spécifié sous la forme d'un flot de travaux
/ carryforward (en paramétrant le mot clé carryforward dans le flot de travaux) si
/ l'option globale enCarryForward est définie sur yes ; sinon, définissez l'option
/ globale enCarryForward sur all.

/ **Syntaxe**

/ **maxdur** *heure* | *pourcentage* % **onmaxdur** *action*

/ **Arguments**

/ *heure* Indique une durée exprimée à l'aide de la syntaxe *HHHMM*, où :

/ **HHH** Représente le nombre d'heures sous la forme d'un nombre compris
/ entre 000 et 500.

/ **MM** Représente le nombre de minutes sous la forme d'un nombre
/ compris entre 00 et 59.

/ *pourcentage*

/ Indique le pourcentage de la durée estimée la plus récente. Il peut s'agir
/ d'un nombre compris entre 0 et 1000000.

/ **onmaxdur** *action*

/ Indique l'action à déclencher sur un travail toujours en cours d'exécution
/ lorsque la durée maximale spécifiée pour ce travail a été dépassée. Les
/ valeurs possibles du paramètre de l'*action* sont :

/ **Kill** Indique d'arrêter le travail en cours d'exécution. A la suite de la
/ commande kill, les travaux sont à l'état ABEND (arrêt anormal).
/ Les travaux ou les flots de travaux qui dépendent de ces travaux
/ ne sont pas libérés. Ces travaux ne peuvent pas être exécutés à
/ nouveau.

/ **Continue**

/ Indique de poursuivre l'exécution du travail en cours, même s'il a
/ dépassé la durée maximale.

/ Lors de la soumission d'une commande **conman** pour définir ou modifier
/ l'action **onmaxdur**, vous devez également spécifier le mot clé **maxdur** associé
/ à l'argument **onmaxdur**.

/ **Exemples**

/ L'exemple suivant indique de poursuivre l'exécution d'un travail en cours s'il
/ s'exécute toujours après une heure et 20 minutes :

/ MAXDUR 80 ONMAXDUR CONT

/ L'exemple suivant indique d'arrêter un travail en cours si celui-ci s'exécute plus
/ d'une heure et 20 minutes :

/ MAXDUR 80 ONMAXDUR KILL

L'exemple suivant indique de poursuivre l'exécution d'un travail en cours s'il s'exécute toujours après avoir dépassé 120 % de sa durée maximale, basée sur la durée estimée la plus récente :

```
MAXDUR 120 % ONMAXDUR KILL
```

mindur

Indique la durée la plus courte au cours de laquelle un travail s'exécute normalement et se termine. Si un travail se termine avant que cette durée minimale soit atteinte, les actions suivantes sont exécutées :

- L'une des actions suivantes est déclenchée : Abend, Confirm ou Continue.
- Le travail apparaît comme ne pas avoir atteint sa durée minimale dans les cas suivants, uniquement si le travail se termine avec le statut SUCCESS :
 - Lors de l'exécution de **showjobs** et **showjobs ;props** à partir de la ligne de commande **conman**, MinDurationNotReached s'affiche.
 - A partir de Dynamic Workload Console dans les propriétés du travail.
 - Un message d'information est consigné dans le fichier *rep_base_TWS/stdlist/logs/aaaammjj_TWSMERGE.log*.

Si ce travail est toujours en cours d'exécution lorsque JnextPlan s'exécute, il est inséré dans le flot de travaux USERJOBS. Le paramètre de durée minimale n'est pas conservé pour le travail dans le flot de travaux USERJOBS et ne sera pas surveillé. Pour que le flot de travaux soit reporté et pour éviter que le travail ne soit déplacé vers le flot de travaux USERJOBS, marquez le flot de travaux initial où le paramètre de durée minimale a été spécifié sous la forme d'un flot de travaux carryforward (en paramétrant le mot clé carryforward dans le flot de travaux) si l'option globale enCarryForward est définie sur yes ; sinon, définissez l'option globale enCarryForward sur all.

Syntaxe

```
mindur heure | pourcentage % onmindur action
```

Arguments

heure Indique une durée exprimée à l'aide de la syntaxe *HHHMM*, où :

HHH Représente le nombre d'heures sous la forme d'un nombre compris entre 000 et 500.

MM Représente le nombre de minutes sous la forme d'un nombre compris entre 00 et 59.

pourcentage

Indique le pourcentage de la durée estimée la plus récente. Il peut s'agir d'un nombre compris entre 0 et 1000000.

onmindur *action*

Indique l'action à déclencher sur un travail qui se termine avant sa durée minimale. Les valeurs possibles du paramètre de l'*action* sont :

Abend

Le travail est défini sur le statut **ABEND**.

Confirm

Le travail est défini sur le statut **CONFIRM**. La charge de travail nécessite une confirmation de l'utilisateur pour se poursuivre.

Continue

L'exécution de la charge de travail se poursuit sans qu'aucune action ne soit à prendre.

Exemples

L'exemple suivant indique de poursuivre l'exécution d'une charge de travail même si le travail n'atteint pas une durée minimale d'au moins 80 minutes :

```
MINDUR 120 ONMINDUR CONT
```

L'exemple suivant indique de définir le statut du travail sur Error si le travail ne s'exécute pas pendant 80 minutes au minimum :

```
MINDUR 120 ONMINDUR ABEND
```

L'exemple suivant nécessite qu'un utilisateur confirme le travail lorsqu'il a atteint 50 % ou la moitié de sa durée minimale estimée la plus récente :

```
MINDUR 50 % ONMINDUR CONFIRM
```

needs

Le mot clé **needs** définit les ressources qui doivent être disponibles avant le lancement d'un travail ou d'un flot de travaux. Vous pouvez utiliser le mot clé **needs** soit dans une définition de flot de travaux, soit dans la définition des travaux contenus, mais pas dans les deux à la fois.

Syntaxe

```
needs [n] [poste_de_travail#]nom_ressource [...]
```

Arguments

n Indique le nombre d'unités de ressources requises. Les valeurs admises sont comprises entre **1** et **1024** pour chaque instruction **needs**. La valeur par défaut est 1.

poste de travail

Indique le nom du poste de travail sur lequel la ressource est définie localement. Si elle n'est pas spécifiée, la valeur par défaut est le poste de travail sur lequel le travail ou flot de travaux dépendant s'exécute. Les ressources ne peuvent être utilisées comme dépendances que par les travaux ou les flots de travaux exécutés sur le poste de travail où la ressource est définie.

En raison du mécanisme de résolution des dépendances de ressources, une dépendance de ressource au niveau du flot de travaux peut être considérée comme 'locale' (et son utilisation est prise en charge) et non comme 'globale', lorsqu'à la fois le flot de travaux et tous ses travaux sont définis sur le même poste de travail que la ressource.

Cependant, un agent standard et son hôte peuvent faire référence aux mêmes ressources.

nom_ressource

Indique le nom de la ressource.

Commentaires

Un travail ou flot de travaux peut nécessiter un maximum de 1024 unités d'une ressource dans une instruction **needs**. Au moment de l'exécution, chaque instruction **needs** est convertie en *réserve*, chacune contenant un maximum de 32 unités d'une ressource spécifique. 32 réservations au maximum peuvent exister pour une seule ressource, quelle que soit sa quantité d'unités disponible. Si 32 réservations sont déjà définies pour une ressource, le travail ou flot de travaux suivant qui attend cette ressource attend jusqu'à ce qu'une réservation en cours se termine ET que la quantité de ressource requise soit disponible.

Exemples

Dans l'exemple suivant, le lancement du flot de travaux sked3 est interdit tant que trois unités de CPUtime et deux unités de tapes ne sont pas disponibles :

```
schedule sked3 on fr
  needs 3 cputime, 2 tapes :
```

La ressource jlimit a été définie avec deux unités disponibles. Dans l'exemple suivant, l'exécution simultanée d'un maximum de deux travaux est autorisée dans le flot de travaux sked4 :

```
schedule sked4 on mo,we,fr:
  joba needs 1 jlimit
  jobb needs 1 jlimit
  jobc needs 2 jlimit <<runs alone>>
  jobd needs 1 jlimit
end
```

nop

Le mot clé nop indique que le travail ne doit pas être exécuté lorsque le plan s'exécute. Le travail est inclus dans le plan en tant qu'élément du flot de travaux dont il fait partie, mais lorsque le plan est mis en oeuvre, il prend le statut Annulation en instance et n'est pas exécuté. Ses dépendances, le cas échéant, sont toutefois libérées et ses successeurs sont exécutés.

Cette option constitue un moyen rapide et facile de *désactiver* provisoirement l'exécution d'un travail sans qu'il soit nécessaire de le supprimer du flot de travaux et de modifier ses dépendances. Par exemple, si vous devez désactiver temporairement l'exécution d'un travail d'impression parce que l'imprimante est hors service, vous pouvez marquer le travail comme étant NOP (non opérationnel) dans la définition de flot de travaux jusqu'à ce que l'imprimante soit réparée ou remplacée.

Syntaxe

```
job-statement
  [...]
  [nop]
  [...]
```

Exemples

Dans l'exemple de travail suivant, jbk1 est défini dans le flot de travaux bkup avec l'option NOP. Il sera marqué comme ayant le statut Annulation en instance dans tous les fichiers Symphony à venir, aussi longtemps que l'option sera définie. Le travail successeur jbk3 sera exécuté.


```

/      schedule bkup on fr at 20:00:
/      cpu1#jbk1
/      nop
/      cpu2#jbk2
/      needs 1 tape
/      cpu3#jbk3
/      follows jbk1
/      end

```

/ **on**

/ Ce mot clé de flot de travaux définit le moment et la fréquence de sélection d'un
 / flot de travaux en vue de son exécution. S'il est omis, le flot de travaux n'est pas
 / ajouté au plan de préproduction. Le mot clé **on** doit suivre le mot clé **schedule**.
 / Pour plus d'informations, voir «except», à la page 244.

/ **Syntaxe**

/ **on** [**runcycle** *nom*

/ [**valide de** *date*] [**valide jusqu'à** *date*]

/ [**description** "*texte*"]

/ [**variable** *nom_table*]

/ {*date* | *jour* | *agenda* | *demande* | "*icalendar*" } [...]

/ [**fdignore** | **fdnext** | **fdprev**][**subset** *nom_sous-ensemble* **AND** | **OR**]

/ **Arguments**

/ **runcycle** *nom*

/ Indique un libellé avec un nom usuel pour le cycle d'exécution spécifié
 / dans les lignes suivantes.

/ **valid from** *date* ... **valid to** *date*

/ Délimite la période au cours de laquelle le flot de travaux est actif,
 / c'est-à-dire pendant laquelle il est ajouté au plan de production. Notez que
 / la date spécifiée comme valeur **valid to** n'est pas incluse dans le cycle
 / d'exécution et le flot de travaux n'est donc pas actif à cette date.

/ **description** "*texte*"

/ Contient une description du cycle d'exécution.

/ **variable**

/ Indique le nom de la table de variables que le cycle d'exécution doit
 / utiliser.

/ *date* Indique un cycle d'exécution qui s'exécute à des dates spécifiques. La
 / syntaxe utilisée pour ce type de cycle d'exécution est la suivante :

/ *aaaammjj* [*aaaammjj*][,...] Par exemple, pour un flot de travaux dont
 / l'exécution est prévue les 25 mai 2009 et 12 juin 2009, la valeur est la
 / suivante :

/ on
 / 20090525,20090612

/ *jour* Indique un cycle d'exécution qui s'exécute des jours spécifiques. La syntaxe
 / utilisée pour ce type de cycle d'exécution est la suivante :

```

/ {mo|tu|we|th|fr|sa|su} Par exemple, pour un flot de travaux qui est
/ planifié pour s'exécuter chaque lundi, la valeur est la suivante :
/ on
/ mo
/
/ agenda Dates indiquées dans un agenda portant ce nom. Le nom de l'agenda peut
/ être suivi d'un décalage au format suivant :
/
/ {+ | -}n {day[s] | weekday[s] | workday[s]}
/
/ Où :
/
/ n Indique le nombre de jours, jours de la semaine ou jours ouvrés.
/
/ days Chaque jour de la semaine.
/
/ weekdays
/ Tous les jours de la semaine sauf le samedi et le dimanche.
/
/ workdays
/ Tous les jours de la semaine excepté les samedis et dimanches (sauf
/ indication contraire par le mot clé freedays) et les dates figurant
/ dans un agenda désigné des jours chômés ou dans l'agenda
/ holidays.
/
/ demande
/ Sélectionne le flot de travaux uniquement lorsque celui-ci est demandé. Ce
/ paramètre est utilisé pour les flots de travaux qui sont sélectionnés par
/ nom et non par date. Pour empêcher un flot de travaux planifié d'être
/ sélectionné pour le travail JnextPlan, redéfinissez-le ON REQUEST.
/
/ Remarque : Si vous tentez d'exécuter un flot de travaux contenant
/ plusieurs heures "à la demande" (on request), tenez compte des points
/ suivants :
/
/ • Les indicateurs "On request" ont la priorité sur les indicateurs "at".
/
/ • Les indicateurs "On request" n'ont jamais la priorité sur les indicateurs
/ "On".
/
/ icalendar
/ Représente une norme utilisée pour spécifier une règle récurrente qui
/ décrit le moment où un flot de travaux s'exécute.
/
/ La syntaxe utilisée pour un cycle d'exécution de type icalendar est la
/ suivante :
/
/ FREQ={DAILY|WEEKLY|MONTHLY|YEARLY}
/
/ [:INTERVAL=[-]n]
/
/ [:{BYFREEDAY|BYWORKDAY|BYDAY=liste_jours_semaine|
/
/ BYMONTHDAY=liste_jours_mois}]
/
/ où la valeur par défaut du mot clé INTERVAL est 1.
/
/ A l'aide du mot clé icalendar, vous pouvez spécifier qu'un flot de travaux
/ s'exécute :
/
/ tous les n jours
/ en utilisant le format suivant :
/
/ FREQ=DAILY[:INTERVAL=n]
/
/ où la valeur définie pour valide de est le premier jour des dates
/ résultantes.

```

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque jour, la valeur est la suivante :

FREQ=DAILY

Pour un flot de travaux planifié pour s'exécuter un jour sur deux, la valeur est la suivante :

FREQ=DAILY; INTERVAL=2

chaque jour chômé ou ouvré
en utilisant le format suivant :

**FREQ=DAILY[;INTERVAL=*n*]
;BYFREEDAY | BYWORKDAY**

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque jour chômé, la valeur est la suivante :

FREQ=DAILY; BYFREEDAY

Pour un flot de travaux planifié pour s'exécuter un jour ouvré sur deux, la valeur est la suivante :

FREQ=DAILY; INTERVAL=2; BYWORKDAY

toutes les *n* semaines, à des jours de la semaine spécifiques
en utilisant le format suivant :

**FREQ=WEEKLY[;INTERVAL=*n*]
;BYDAY=*liste_jours_semaine***

où la valeur définie pour *liste_jours_semaine* est
[SU] [,MO] [,TU] [,WE] [,TH] [,FR] [,SA]

Par exemple, pour un flot de travaux planifié pour s'exécuter chaque semaine le vendredi et le samedi, la valeur est la suivante :

FREQ=WEEKLY; BYDAY=FR,SA

Pour un flot de travaux planifié pour s'exécuter toutes les trois semaines, le vendredi, la valeur est la suivante :

FREQ=WEEKLY; INTERVAL=3; BYDAY=FR

tous les *n* mois, à des dates spécifiques
en utilisant le format suivant :

**FREQ=MONTHLY[;INTERVAL=*n*]
;BYMONTHDAY=*liste_jours_mois***

où la valeur définie pour *liste_jours_mois* est représentée par une liste de
[+nombre_de_jours_à_partir_début_du_mois]
[-nombre_de_jours_à_partir_fin_du_mois]
[nombre_de_jours_du_mois]

Par exemple, pour un flot de travaux planifié pour s'exécuter le 27ème jour de chaque mois, la valeur est la suivante :

FREQ=MONTHLY; BYMONTHDAY=27

Pour un flot de travaux planifié pour s'exécuter tous les six mois, le 15 du mois et le dernier jour du mois, la valeur est la suivante :

FREQ=MONTHLY; INTERVAL=6; BYMONTHDAY=15,-1

```

/      tous les n mois, à des jours spécifiques de la semaine
/      en utilisant le format suivant :
/
/      FREQ=MONTHLY[;INTERVAL=n]
/
/      ;BYDAY=numéro_de_la_semaine_du_mois_jour_semaine
/
/      où la valeur définie pour
/      numéro_de_la_semaine_du_mois_jour_semaine est représentée par une
/      liste de
/      [+numéro_semaine_à_partir_début_du_mois]
/      [-numéro_semaine_à_partir_fin_du_mois]
/      [jour_semaine]
/
/      Par exemple, pour un flot de travaux planifié pour s'exécuter
/      chaque mois, le premier lundi et le dernier vendredi du mois, la
/      valeur est la suivante :
/      FREQ=MONTHLY;BYDAY=1MO,-1FR
/
/      Pour un flot de travaux planifié pour s'exécuter tous les six mois,
/      le deuxième mardi du mois, la valeur est la suivante :
/      FREQ=MONTHLY;INTERVAL=6;BYDAY=2TU
/
/      toutes les n années
/      en utilisant le format suivant :
/
/      FREQ=YEARLY[;INTERVAL=n]
/
/      où la valeur définie pour valide de est le premier jour des dates
/      résultantes.
/
/      Par exemple, pour un flot de travaux planifié pour s'exécuter
/      chaque année, la valeur est la suivante :
/      FREQ=YEARLY
/
/      Pour un flot de travaux planifié pour s'exécuter tous les deux ans,
/      la valeur est la suivante :
/      FREQ=YEARLY;INTERVAL=2
/
/      Remarque : Les limitations suivantes s'appliquent :
/      • L'intervalle maximal pris en charge pour un cycle d'exécution
/      quotidien est de 31 jours.
/      • L'intervalle maximal pris en charge pour un cycle d'exécution
/      hebdomadaire est de 8 jours.
/      • L'intervalle maximal pris en charge pour un cycle d'exécution
/      mensuel est de 12 mois. Pour des cycles d'exécution indiquant le
/      jour de la semaine en fonction du mois, par exemple le troisième
/      samedi ou le deuxième vendredi, l'intervalle maximal pris en
/      charge est de 5 jours.
/      • L'intervalle maximal pris en charge pour un cycle d'exécution
/      annuel est de 10 ans.
/
/      fdignore | fdnext | fdprev
/      Indique la règle à appliquer si la date sélectionnée pour l'exécution du
/      travail ou du flot de travaux coïncide avec un jour chômé. Les valeurs
/      possibles sont les suivantes :
/
/      fdignore
/      La date n'est pas ajoutée.

```

/ **fdnext** Le jour ouvré le plus proche qui suit le jour chômé est ajouté.
 / **fdprev**
 / Le jour ouvré le plus proche précédant le jour chômé est ajouté.
 /
 / **[subset nom_sous-ensemble AND|OR]**
 / **subset nom_sous-ensemble**
 / Indique le nom du sous-ensemble. Si vous ne spécifiez pas de nom,
 / SUBSET_1 est utilisé par défaut.
 /
 / **AND|OR**
 / Par défaut, les cycles d'exécution dans un sous-ensemble sont dans
 / une relation logique **OR** mais vous pouvez la changer en relation
 / logique **AND** tant que le résultat du groupe de cycle d'exécution
 / est une date ou un ensemble de dates positif (inclusif).

/ **Commentaires**

/ Vous pouvez définir plusieurs instances du mot clé **on** pour le même flot de
 / travaux. Si une définition de flot de travaux contient plusieurs instances **on**,
 / celles-ci doivent être consécutives. Chaque instance correspond à un cycle
 / d'exécution auquel vous pouvez associer une règle de jour chômé.

/ Chaque instance du mot clé doit contenir l'une des valeurs autorisées par la
 / syntaxe **on**.

/ Si les heures de début du cycle d'exécution et du flot de travaux sont définies, celle
 / du cycle d'exécution l'emporte lorsque le flot de travaux est planifié avec
 / **JNextPlan**. Si le flot de travaux est lancé avec la commande **submit**, l'heure de
 / début du cycle d'exécution n'est pas utilisée.

/ **Exemples**

/ Dans l'exemple suivant, le flot de travaux sked1 est sélectionné les lundis et les
 / mercredis :
 / `schedule sked1 on mo,we`

/ Dans l'exemple suivant, le flot de travaux sked3 est sélectionné le 15 juin 2008 et
 / aux dates répertoriées dans l'agenda apdates :
 / `schedule sked3 on 6/15/08,apdates`

/ Dans l'exemple suivant, le flot de travaux sked4 est sélectionné deux jours avant
 / chaque date figurant dans l'agenda monthend :
 / `schedule sked4 on monthend -2 weekdays`

/ Dans l'exemple suivant, le flot de travaux testskd1 est sélectionné tous les jours de
 / la semaine sauf les mercredis :
 / `schedule testskd1 on weekdays`
 / `except we`

/ Dans l'exemple suivant, le flot de travaux testskd3 est sélectionné tous les jours de
 / la semaine à l'exception des 15 et 24 mai 2008 :
 / `schedule testskd3 on weekdays`
 / `except 05/16/2008,05/24/2008`

```

/ Dans l'exemple suivant, le flot de travaux testskd4 est sélectionné tous les jours
/ sauf pendant les deux jours précédant toute date figurant dans un agenda intitulé
/ monthend :
/
/ schedule testskd4 on everyday
/ except monthend -2 weekdays
/
/ Sélectionnez le flot de travaux sked1 pour qu'il s'exécute les lundis et vendredis
/ ainsi que le 29/12/2009. Si les lundis et le 29/12/2009 correspondent à des jours
/ chômés, exécutez le flot de travaux le jour ouvré suivant le plus proche. Si les
/ vendredis correspondent à des jours chômés, exécutez le flot de travaux le jour
/ ouvré précédent le plus proche. Dans cet exemple, les jours chômés sont les
/ samedis, les dimanches et toutes les dates répertoriées dans l'agenda HOLIDAYS par
/ défaut. Les jours ouvrés incluent tous les jours de la semaine du lundi au vendredi
/ ne figurant pas dans l'agenda HOLIDAYS.
/
/ schedule
/ sked1
/ on mo, 12/29/2009 fdnext
/ on fr fdprev
/
/ L'exemple suivant montre le résultat de la commande display du flot de travaux
/ testcli défini pour s'exécuter lors de cycles d'exécution différents sur le poste de
/ travail site2 :
/
/ display js=site2#testcli
/
/ obtenu dans un format à 120 colonnes par la commande
/ MAESTROCOLUMNS=120 avant l'accès à la ligne de commande composer :
/
/ JobstreamName Workstation Draft Valid From Valid To UpdatedBy UpdatedOn LockedBy
/ -----
/ TESTCLI SITE2 Y 08/25/2008 - mdmDBE4 08/25/2008 mdmDBE4
/
/ SCHEDULE W5#TESTCLI VALID FROM 08/25/2008 TIMEZONE ACT
/ DESCRIPTION "Job stream with several run cycle settings."
/ DRAFT
/ ON RUNCYCLE M5 VALID FROM 08/25/2008
/ DESCRIPTION "monthly"
/ "FREQ=MONTHLY;INTERVAL=5;BYMONTHDAY=-3,1"
/ ( AT 0000 )
/ ON RUNCYCLE W4 VALID FROM 08/25/2008
/ DESCRIPTION "weekly"
/ "FREQ=WEEKLY;INTERVAL=5;BYDAY=MO,WE"
/ FDNEXT ( AT 0000 )
/ ON RUNCYCLE D3 VALID FROM 08/25/2008
/ DESCRIPTION "daily"
/ "FREQ=DAILY;INTERVAL=2"
/ FDPREV ( AT 0000 )
/ ON RUNCYCLE C2 VALID FROM 08/25/2008
/ DESCRIPTION "calendar"
/ ITALY +2 DAYS
/ ( AT 0000 )
/ ON RUNCYCLE M6 VALID FROM 08/25/2008
/ DESCRIPTION "monthly"
/ "FREQ=MONTHLY;INTERVAL=2;BYDAY=1MO,1TH,2WE"
/ ( AT 0000 +2 DAYS )
/ ON RUNCYCLE Y7 VALID FROM 08/25/2008
/ DESCRIPTION "yearly"
/ "FREQ=YEARLY;INTERVAL=7"
/ ( AT 0100 )
/ ON RUNCYCLE SS1 VALID FROM 08/25/2008
/ 08/10/2008,08/18/2008,08/20/2008,08/25/2008
/ ( AT 0000 UNTIL 0000 +1 DAYS ONUNTIL SUPPR DEADLINE 0000 +2 DAYS )
/ EXCEPT RUNCYCLE S1 VALID FROM 08/25/2008
/ DESCRIPTION "simple"

```

```

/          08/26/2008,08/28/2008,08/30/2008,09/13/2008
/          ( AT 0000 )
/
/          CARRYFORWARD
/          MATCHING SAMEDAY
/          FOLLOWS LAB235004#SROBY2.@
/          FOLLOWS X8#COPYOFJS2.RR
/          FOLLOWS XA15::TPA
/          KEYSCHED
/          LIMIT 22
/          PRIORITY 15
/          :
/          X8#PIPP0 AS JOBTC
/          CONFIRMED
/          PRIORITY 13
/          KEYJOB
/          FOLLOWS W5#POPO.@
/          FOLLOWS X8#JS2.F3
/          END
/
/          AWSBIA291I Total objects: 1

```

/ L'agenda ITALY est un agenda personnalisé défini dans la base de données qui
/ détermine les jours ouvrés et chômés de l'agenda en vigueur en Italie.

= **onlate**
= Le mot clé **onlate** définit l'action à effectuer sur un travail dans le flot de travaux
= lorsque l'échéance arrive.

= **Syntaxe**

= **onlate** *action*

= **Arguments**

= *action* Indique l'action à effectuer sur le travail lorsque l'échéance arrive. L'action
= prise en charge est **kill**. Si le travail est en cours d'exécution lorsque
= l'échéance arrive, il est arrêté. A la suite de la commande kill, les travaux
= sont à l'état ABEND (arrêt anormal). Les travaux ou les flots de travaux
= qui dépendent de ces travaux ne sont pas libérés. Si la dépendance sur le
= travail est conditionnelle et que celui-ci se termine à l'état de ABEND (fin
= anormale), cette dépendance est libérée.

= **Commentaires**

= Ce mot clé s'applique uniquement aux travaux définis dans les flots de travaux.

= Si vous ne spécifiez pas une heure **until** pour le travail et que vous spécifiez une
= heure **deadline** avec une action **onlate kill**, le mot clé **until** est automatiquement
= défini sur la même heure que le mot clé **deadline**. Par conséquent, si le travail n'a
= pas encore démarré, il est supprimé de toute manière.

= Le mot clé **until** est défini uniquement au niveau du plan et ne modifie donc pas
= la définition du travail.

Exemples

L'exemple suivant lance le flot de travaux ABSENCES chaque jour, ainsi que le travail calc_job. Ils doivent démarrer à 14h30 et être terminés à 16h00. Si le travail calc_job n'est pas terminé à l'échéance de 16 h 00, il est arrêté et son exécution s'arrête.

```
schedule ABSENCES on everyday :  
    ABSENCES at 1430 deadline 1600 onlate kill  
end
```

onoverlap

Définit l'action que le planificateur doit exécuter sur une instance de flot de travaux qui est sur le point de démarrer alors que l'instance précédente n'est pas encore terminée. Les options possibles sont les suivantes :

- Démarrer tout de même l'instance de flot de travaux
- Attendre que l'instance précédente soit terminée
- Annuler complètement l'exécution de la nouvelle instance

Syntaxe

onoverlap parallel | enqueue | donotstart

Arguments

parallel

L'instance suivante est tout de même démarrée et les deux instances s'exécutent simultanément. Il s'agit du comportement par défaut pour le flot de travaux si vous n'utilisez pas le mot clé onoverlap.

enqueue

L'instance suivante ne démarre pas tant que l'exécution de l'instance précédente n'est pas terminée. Lors de la planification, une nouvelle dépendance est ajoutée à l'instance précédente. La nouvelle instance démarrera lorsque la dépendance sera libérée, sous réserve qu'elle est libérée dans les quatre minutes après l'heure de début de l'instance précédente. Si ce délai est dépassé, la nouvelle instance ne démarre pas.

donotstart

L'instance suivante n'est pas du tout démarrée.

Exemple

Dans l'exemple suivant, une instance de flot de travaux JS_0415 est exécutée toutes les 10 minutes. Dans le cas où une instance ne s'est pas terminée lorsque l'instance suivante doit démarrer, cette dernière attend que l'exécution de l'instance précédente soit terminée.

```
SCHEDULE MDM005#JS_0415  
ON RUNCYCLE ACCTRC 06/16/2014  
( SCHEDTIME 1445 DEADLINE 1530 EVERY 0010 EVERYENDTIME 1515 )  
ONOVERLAP ENQUEUE  
:  
MDM005#JS_0415  
FIN
```

opens

Indique les fichiers qui doivent être disponibles avant le lancement d'un travail ou d'un flot de travaux.

Syntaxe

opens [*poste_de_travail*#]"*nom_fichier*" [(*qualifiant*)] [...]

Arguments

poste de travail

Indique le nom du poste de travail ou de la classe de postes de travail qui contient le fichier. Par défaut, il s'agit du poste de travail ou de la classe de postes de travail du travail ou du flot de travaux dépendant. Si une classe de postes de travail est utilisée, celle-ci doit être identique à celle du flot de travaux qui contient cette instruction.

nom_fichier

Indique le nom du fichier, y compris son chemin d'accès complet, placé entre guillemets. Vous pouvez utiliser les paramètres de IBM Workload Scheduler dans tout ou partie de la chaîne de noms de fichier. Vous pouvez également utiliser les variables définies dans la table des variables du poste de travail sur lequel se trouve le fichier. Voir «Définition des variables et des paramètres», à la page 206 pour plus d'informations et d'exemples.

qualifier

Indique une condition de test valide. Sous UNIX, le qualifiant est transmis à une commande **test**, qui s'exécute en tant que **racine** dans bin/sh. Cependant, sur des agents dynamiques, si une installation d'agent non root est effectuée, vous devez vérifier si la commande de test est disponible pour l'utilisateur de l'installation et le test s'exécutera en tant qu'utilisateur IBM Workload Scheduler (par exemple, l'utilisateur de l'installation).

Pour les pools et les pools dynamiques, étant donné qu'il est impossible de savoir à l'avance sur quel agent de membre le test s'exécutera et qu'il n'existe aucune affinité entre l'agent qui satisfait la condition et l'agent qui exécute le travail, il est recommandé de définir une dépendance de fichier uniquement dans le cas où une condition doit être évaluée sur le système de fichiers partagé.

Si vous voulez qu'au moins un agent satisfasse la condition et exécuter le travail, voir la surveillance du fichier d'automatisation de la charge de travail commandée par les événements avec les actions de soumission TWSAction, en utilisant une variable comme poste de travail. Voir les actions TWSAction dans la section consacrée aux définitions d'action et d'événement d'automatisation de la charge de travail commandée par les événements du manuel *IBM Workload Scheduler - Guide d'utilisation et de référence*. Voir aussi la rubrique concernant l'utilisation des dépendances de fichier dans la planification dynamique pour plus d'informations sur la gestion des dépendances de fichier avec des agents dynamiques, des pools et des pools dynamiques dans *IBM Workload Scheduler - Guide d'utilisation et de référence*.

Avertissement :

- Sur les systèmes d'exploitation UNIX, la liste des qualifiants pris en charge dépend du type du système d'exploitation. Vous pouvez vérifier les qualifiants pris en charge en exécutant la commande **bin/sh/test**.
- Sous Windows, la fonction de test est effectuée en tant qu'utilisateur IBM Workload Scheduler (par exemple, l'utilisateur de l'installation).
- Le qualifiant **-e %p** n'est pas pris en charge sur les systèmes d'exploitation Solaris.

Les qualifiants acceptés sont les suivants :

-d %p Si le fichier existe et qu'il s'agit d'un répertoire.

-e %p Si le fichier existe.

-f %p Si le fichier existe et qu'il est ordinaire.

-r %p Si le fichier existe et qu'il peut être lu.

-s %p Si le fichier existe et que sa taille n'est pas nulle.

-w %p Si le fichier existe et qu'il peut être écrit.

-a Opérateur booléen AND.

-o Opérateur booléen OR.

Sous UNIX et Windows, l'expression **%p** est utilisée pour transmettre la valeur attribuée à *nom_fichier* à la fonction de test.

Entrer (**notempty**) est équivalent à (**-s %p**). Si aucun qualifiant n'est indiqué, la valeur par défaut est (**-f %p**).

Commentaires

Combinaison entre le *chemin de fichier* et les *qualifiants* (limitée à 120 caractères) et le *nom de fichier* (limité à 28 caractères).

Exemples

L'exemple suivant permet de vérifier, avant le lancement de `ux2#sked6`, que le fichier `c:\users\fred\datafiles\file88` du poste de travail `nt5` est accessible en lecture :

```
schedule ux2#sked6 on tu opens nt5#"c:\users\fred\datafiles\file88"
```

L'exemple suivant permet de vérifier, avant le lancement du travail `jobr2`, l'existence des trois répertoires `/jean`, `/marie` et `/roger` sous le répertoire `/users` :

```
jobr2 opens "/users"(-d %p/jean -a -d %p/marie -a -d %p/roger)
```

L'exemple suivant permet de vérifier si `cron` a créé son fichier FIFO avant de lancer le travail `job6` :

```
job6 opens "/usr/lib/cron/FIFO"(-p %p)
```

L'exemple suivant permet de vérifier, avant le lancement du travail `jobt2`, que le fichier `d:\work\john\execit1` du poste de travail `dev3` existe et qu'il n'est pas vide :

```
jobt2 opens dev3#"d:\work\john\execit1"(notempty)
```

L'exemple suivant permet de vérifier, avant le lancement du travail jobt2, que le fichier c:\tech\checker\startf du poste de travail nyc existe et qu'il n'est pas vide :

```
job77 opens nyc#"C:\tech\checker\startf"(-s %p -a -w %p)
```

Sécurité des commandes test(1) :

Sous UNIX, une fonction de sécurité spéciale empêche toute utilisation non autorisée d'autres commandes dans le qualifiant. Par exemple, le fichier ci-dessous contient une commande dans le qualifiant :

```
/users/xpr/hp3000/send2(-n "`ls /users/xpr/hp3000/m*`" -o -r %p)
```

Si le qualifiant ne contient aucune autre commande, les vérifications suivantes sont effectuées :

- L'option locale *jm no root* doit avoir la valeur no.
- Dans le fichier de sécurité, l'utilisateur qui documente le programme ou qui ajoute la dépendance Open Files avec une commande **conman adddep**, doit avoir soumis l'accès à un travail avec les attributs suivants :

name=cmdstest.fileeq

logon=root

jcl=chemin d'accès des fichiers OPEN

cpu=poste de travail sur lequel résident les fichiers OPEN

Notez que cmdstest et fileeq n'existent pas.

priority

Définit la priorité d'un travail ou d'un flot de travaux. En affectant une priorité différente aux travaux ou flots de travaux, vous déterminez lequel démarre en premier, à condition que les dépendances soient résolues.

En supposant que les travaux et flots de travaux soient prêts à être lancés, si vous définissez une priorité pour les flots de travaux et pour les travaux des flots de travaux :

- Le flot de travaux qui commence le premier est celui dont la priorité est maximale.
- Parmi les travaux du flot de travaux ayant la priorité maximale, le travail qui démarre le premier est celui dont la priorité est maximale.

Syntaxe

priority *nombre* | **hi** | **go**

Arguments

nombre Indique la priorité. Les valeurs admises sont comprises entre 0 et 99. Une priorité de 0 empêche le lancement du travail ou du flot de travaux. La valeur par défaut est 10 et n'apparaît pas lors de l'affichage de la définition de flot de travaux.

hi Représente une valeur supérieure à toute valeur pouvant être spécifiée à l'aide d'un nombre. Une fois défini, le travail ou le flot de travaux est immédiatement lancé lorsqu'il ne contient aucune dépendance.

go Représente la plus haute priorité pouvant être définie. Une fois défini, le travail ou le flot de travaux est immédiatement lancé lorsqu'il ne contient aucune dépendance.

Commentaires

Les travaux et les flots de travaux avec des niveaux de priorité **hi** ou **go** sont lancés dès que toutes leurs dépendances sont résolues. Dans ce cas :

- Les flots de travaux remplacent le nombre maximal de travaux de l'unité centrale.
- Les travaux remplacent le nombre maximal de travaux de l'unité centrale, mais ils ne remplacent ni le nombre maximal de travaux planifiés ni la priorité minimale de travail de l'unité centrale.

Exemples

L'exemple suivant illustre la relation entre les priorités des flots de travaux et celles des travaux. Les deux flots de travaux, sked1 et sked2 ont les définitions suivantes dans la base de données :

```
schedule sked1 on tu
priority 50
:
job1 priority 15
job2 priority 10
end

schedule sked2 on tu
priority 10
:
joba priority 60
jobb priority 50
end
```

Etant donné que le flot de travaux sked1 possède la priorité maximale, les travaux sont lancés dans l'ordre suivant : job1, job2, joba, jobb.

Si les priorités de flot de travaux étaient identiques, les travaux seraient lancés dans l'ordre suivant : joba, jobb, job1, job2.

Si job2 possède une dépendance **A** et job1 une dépendance **B** et si la dépendance **A** est résolue (tandis que **B** reste non résolue), alors job2 démarre avant job1, même si la priorité de job2 est plus faible que celle de job1.

prompt

Définit les invites auxquelles une réponse affirmative doit être donnée avant le lancement d'un travail ou d'un flot de travaux.

Syntaxe

prompt *nom_invite* [...]

prompt "[: | !]*texte*" [...]

Arguments

nom_invite

Indique le nom d'une invite de la base de données. Vous pouvez spécifier plusieurs valeurs *nom_invite* séparées par des virgules, mais vous ne pouvez pas mélanger sous le même mot clé **prompt** des invites définies dans la base de données avec des invites littérales.

texte Indique une invite littérale sous forme de chaîne de caractères placée entre

/ guillemets (""). Plusieurs chaînes séparées par une barre oblique inversée et
 / **n** (\n) peuvent être utilisées pour des messages longs. Si la chaîne
 / commence par un signe deux-points (:), le message s'affiche mais aucune
 / réponse n'est requise. Si la chaîne commence par un point d'exclamation
 / (!), le message s'affiche, mais il n'est pas enregistré dans le fichier journal.
 / Vous pouvez entrer une barre oblique suivie d'un **n** ("\n") dans le texte
 / pour les retours à la ligne.

/ Un ou plusieurs paramètres peuvent être utilisés dans tout ou partie de la
 / chaîne de caractères. Pour utiliser un paramètre, placez son nom entre
 / carets (^). Voir «Définition des variables et des paramètres», à la page 206
 / pour plus d'informations et d'exemples.

/ **Remarque :** Dans une invite locale, lorsqu'ils ne désignent pas de
 / paramètre, les carets (^) doivent être précédés d'une barre oblique inversée
 / pour ne pas provoquer d'erreur. Dans les invites globales, il n'est pas
 / nécessaire de faire précéder les carets d'une barre oblique inversée.

/ Exemples

/ L'exemple suivant illustre des invites littérales et nommées. La première
 / correspond à une invite littérale qui utilise un paramètre intitulé sys1. Lorsqu'une
 / seule réponse affirmative est reçue pour l'invite nommée apmsg, les dépendances
 / des deux travaux job1 et job2 sont respectées.

```
/ schedule sked3 on tu,th
/   prompt "All ap users logged out of ^sys1^? (y/n)"
/   :
/     job1 prompt apmsg
/     job2 prompt apmsg
/ end
```

/ L'exemple suivant définit une invite littérale qui apparaît sur plusieurs lignes. Il
 / contient une barre oblique inversée suivie de la lettre **n** (\n) à la fin de chaque
 / ligne :

```
/ schedule sked5 on fr
/   prompt "Les travaux du flot de travaux occupent \n
/   un temps UC énorme.\n
/   Voulez-vous les lancer maintenant ? (o/n)"
/   :
/     j1
/     j2 follows j1
/ end
```

/ schedtime

/ Représente l'heure à laquelle le flot de travaux est positionné dans le plan. La
 / valeur affectée à **schedtime** ne représente pas une dépendance pour le flot de
 / travaux. Pendant que le plan de production est en cours, l'instance de travail ou de
 / flot de travaux démarre parfois son traitement avant l'heure définie dans le mot clé
 / **schedtime** si toutes ses dépendances sont résolues et si sa priorité lui permet de
 / démarrer.

/ Syntaxe

/ **schedtime** *heure* [**timezone** | **tz fuseau_horaire**][**+n jour[s]**] [...]

/ Arguments

/ *heure* Indique une heure de la journée au format : HHHH:mm. Les valeurs
 / admises sont comprises entre **0000** et **320000**.

tzname Indique le fuseau horaire à utiliser lors du calcul de l'heure de début. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

n Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

Commentaires

A la différence du mot clé **at**, le mot clé **schedtime** ne représente pas une dépendance horaire : il n'indique pas d'heure avant laquelle un travail ou flot de travaux ne peut pas démarrer. Au lieu de cela, la valeur spécifiée dans le mot clé **schedtime** est utilisée uniquement pour positionner l'instance de travail ou de flot de travaux spécifique dans le plan de préproduction. Pendant que le plan de production est en cours, l'instance de travail ou de flot de travaux démarre parfois son traitement avant l'heure définie dans le mot clé **schedtime** si toutes ses dépendances sont résolues et si sa priorité lui permet de démarrer.

Pour plus d'informations sur l'identification des prédécesseurs dans le plan de préproduction à l'aide du mot clé **schedtime**, voir «Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux», à la page 71.

Les mots clés **at** et **schedtime** s'excluent mutuellement. Si **schedtime** n'est pas spécifié et si le mot clé **at** est spécifié dans le travail ou le flot de travaux, sa valeur est utilisée pour positionner l'instance dans le plan de préproduction.

Si aucun des mots clés **at** et **schedtime** n'est spécifié dans la définition du travail ou du flot de travaux, le système adopte par défaut la valeur affectée à l'option globale *startOfDay* définie sur le gestionnaire de domaine maître.

Pour les flots de travaux comportant une définition **schedtime**, la valeur de la zone Heure de début affichée sur Dynamic Workload Console dépend du paramétrage de l'option globale *enPreventStart* (qui détermine si les flots de travaux sans dépendance **at** peuvent démarrer immédiatement, sans attendre le cycle d'exécution spécifié dans le flot de travaux) :

- Si *enPreventStart* est défini sur la valeur *yes*, l'heure de démarrage est "12:00 AM" convertie dans le fuseau horaire spécifié dans l'interface graphique.
- Si *enPreventStart* n'est pas défini sur la valeur *no*, la zone de l'heure de démarrage est vide.

Exemples

Dans les exemples suivants, on considère que le jour de traitement IBM Workload Scheduler commence à 6 heures du matin.

- Le flot de travaux suivant qui doit être exécuté tous les mardis est planifié pour démarrer à 3:00 le mercredi matin. Les deux travaux qu'il contient sont lancés dès que possible après le début du traitement du flot de travaux.

```
schedule sked7 on tu schedtime 0300:  
  job1  
  job2  
end
```

- Le fuseau horaire du poste de travail *sfran* correspond à l'heure du pacifique America/Los_Angeles et le fuseau horaire du poste de travail *nycity* correspond à l'heure de la côte Est America/New_York. Le flot de travaux *sked8* est

```

/ sélectionné pour s'exécuter le vendredi. Il est planifié pour démarrer sur le poste
/ de travail sfran à 10 heures du matin, heure de Los Angeles, le samedi (tel que
/ spécifié par le décalage + 1 day). Le travail job1 est lancé sur sfran dès que
/ possible après le début du traitement du flot de travaux. Le travail job2 est
/ lancé sur sfran à 14 heures America/New_York (11:00 a.m.
/ America/Los_Angeles) le samedi. Le travail job3 est lancé sur le poste de travail
/ nycity à 16 heures sur le fuseau horaire America/New_York (1:00 p.m.
/ America/Los_Angeles) le samedi.
/
/ sfran#schedule sked8 on fr schedtime 1000 + 1 day:
/ job1
/ job2 at 1400 tz America/New_York
/ nycity#job3 at 1600
/ end

```

/ **schedule**

/ Indique le nom du flot de travaux. A l'exception des commentaires, il doit
/ correspondre au premier mot clé d'un flot de travaux et être suivi du mot clé **on**.

/ **Syntaxe**

/ **schedule** [*poste_de_travail#*]*nom_flot_travaux*

/ [**timezone** | **tz** *fuseau_horaire*]

/ **Arguments**

/ *poste de travail*

/ Indique le nom du poste de travail sur lequel le flot de travaux est lancé.
/ Par défaut, il s'agit du poste de travail sur lequel **composer** s'exécute pour
/ ajouter le flot de travaux.

/ *nom_flot_travaux*

/ Indique le nom du flot de travaux. Ce nom doit commencer par une lettre
/ et peut contenir des caractères alphanumériques, des tirets et des traits de
/ soulignement. Il peut comporter jusqu'à 16 caractères.

/ **timezone** | **tz** *fuseau_horaire*

/ Indique le fuseau horaire à utiliser lors de la gestion du flot de travaux. Ce
/ paramètre est ignoré si l'option globale *enTimeZone* est définie sur **no** sur le
/ gestionnaire de domaine maître. Pour plus d'informations sur les
/ paramètres de fuseau horaire, voir Chapitre 19, «Gestion des fuseaux
/ horaires», à la page 799.

/ **Commentaires**

/ Dans une définition de flot de travaux, vous pouvez définir un fuseau horaire pour
/ l'ensemble du flot de travaux en utilisant le mot clé **timezone** dans l'intervalle de
/ validité, ou en spécifiant des restrictions temporelles à l'aide des mots clés **at**, **until**
/ ou **deadline**.

/ Vous pouvez également définir un fuseau horaire pour un travail contenu dans un
/ flot de travaux en définissant les mots clés **at**, **until** ou **deadline** pour ce travail.

/ Que vous soyez en train de définir un travail ou un flot de travaux, si vous utilisez
/ un fuseau horaire dans une restriction temporelle, par exemple, **at**, vous devez
/ utiliser le même fuseau horaire lorsque vous spécifiez les autres restrictions
/ temporelles, telles que **deadline** et **until**.

Dans une définition de flot de travaux, vous pouvez définir un fuseau horaire pour l'ensemble du flot de travaux et pour les travaux qu'il contient. Ces fuseaux horaires peuvent être différents, auquel cas le fuseau horaire défini pour le travail est converti dans le fuseau horaire défini pour le flot de travaux.

Pour gérer tous les paramètres de fuseau horaire possibles, la conversion de fuseau horaire effectuée lors du traitement des travaux et des flots de travaux dans l'ensemble du réseau IBM Workload Scheduler respecte les critères suivants :

1. Si aucun fuseau horaire n'est défini pour un travail contenu dans un flot de travaux, ce travail hérite du fuseau horaire défini sur le poste de travail où il est censé s'exécuter.
2. Si aucun fuseau horaire n'est défini pour un flot de travaux, le fuseau horaire utilisé correspond à celui déjà défini sur le poste de travail où le flot de travaux est censé s'exécuter.
3. Si aucun des fuseaux horaires mentionnés n'est défini, le fuseau horaire utilisé est celui défini sur le gestionnaire de domaine maître.

Exemples

Voici la définition d'un fuseau horaire du travail sked8 exécuté sur le poste de travail sfran sur lequel le fuseau horaire America/New_York est défini. Le fuseau horaire défini pour l'exécution de job2 sur le poste de travail LA est défini sur America/Los_Angeles.

```
schedule sfran#sked8
tz America/New_York
on fr at 1000 + 1 day:
job1
LA#job2 at 1400 tz America/Los_Angeles
end
```

statisticstype custom

Marque un travail de façon à ce que sa durée estimée soit calculée par une sous-ensemble d'IBM SPSS Statistics et non par le processus logman par défaut (automatique).

L'outil statistique basé sur SPSS calcule des prévisions à cinq jours de la durée estimée du travail en utilisant des données de l'historique remontant jusqu'à quatre-vingt-dix jours. La durée estimée et l'intervalle de confiance associé sont renvoyés dans la sortie de la commande conman showjobs et dans les vues suivantes de Dynamic Workload Console :

- Propriétés de travail
- Surveiller la charge de travail
- Chemin critique
- Analyse par hypothèses

Cet outil peut être particulièrement utile pour voir au-delà des incidences acceptées des événements cycliques déjà connus, et pour comprendre quels sont les conflits apparemment masqués qui peuvent affecter la durée d'un travail. Il peut être pertinent de mesurer et de prévoir les durées des travaux le long d'un chemin critique qui ne respecte pas son échéance occasionnellement.

Pour en savoir plus sur les mesures à suivre pour utiliser les statistiques basées sur SPSS en plus de l'utilisation de ce mot clé, voir Chapitre 13, «Utilisation de statistiques avancées pour prévoir la durée estimée d'un travail», à la page 569.

Syntaxe

statisticstype custom

Si vous omettez ce mot clé, la durée estimée pour le travail ainsi que son intervalle de confiance associé sont calculés par défaut par le processus exécuté par la commande logman et deviennent disponibles sur le même support.

startcond

Utilisez ce mot clé pour générer dans le flot de travaux un mécanisme qui recherche des événements et des conditions spécifiques, et libère le flot de travaux lorsque ces éléments se produisent.

Vous pouvez rechercher l'un des types d'événement suivants :

filecreated

Vous pouvez vérifier si un ou plusieurs fichiers ont été créés sur le poste de travail indiqué.

filemodified

Vous pouvez rechercher des modifications dans un plusieurs fichiers présents sur le poste de travail indiqué.

job Vous pouvez définir un travail dont la condition de sortie doit être surveillée.

Lorsque les conditions reposent sur la création ou la modification de fichiers, lorsque vous sauvegardez un flot de travaux, un travail de surveillance est créé automatiquement pour surveiller la condition spécifiée. Ce travail devient le premier travail du flot de travaux et reste à l'état EXEC jusqu'à ce que la condition qu'il surveille soit remplie ou que l'échéance du travail soit atteinte. Ce type de travail est identifié par le libellé +AUTOGEN+ sur la ligne de commande et par une icône comportant le caractère A dans Dynamic Workload Console.

Lorsque les conditions reposent sur le résultat d'un travail spécifié, lorsque vous sauvegardez le flot de travaux, le travail devient le premier travail du flot de travaux et redémarre jusqu'à ce que la condition soit satisfaite ou que l'échéance du travail soit atteinte. Cela s'applique également si le travail se termine à l'état Succès. Il s'agit du travail de surveillance. Lorsque vous spécifiez ce type de condition, IBM Workload Scheduler définit automatiquement une condition de sortie de réussite sur le travail de surveillance. Ce dernier se termine avec succès lorsqu'une de ses conditions de sortie est satisfaite, y compris celle sur le travail de surveillance lui-même. Vous pouvez appliquer cette logique au flot de travaux ou à des travaux spécifiques dans le flot de travaux. Pour plus d'informations sur les conditions de sortie, voir la section Application d'une logique de branchement conditionnelle du *Guide d'utilisation et de référence*.

Si vous ne spécifiez pas de nom pour le travail de surveillance, le nom **file_StartCond** est utilisé par défaut. Cette valeur est définie à l'aide de l'option optman **fileStartConditionJobName**. Pour plus d'informations sur l'option **fileStartConditionJobName**, voir la description des options globales dans Guide d'administration.

Par défaut, IBM Workload Scheduler continue à surveiller la condition même après que celle-ci a été remplie une première fois. Pour que cela soit possible, un travail Soumission d'un flot de travaux est créé automatiquement et ajouté au flot de travaux en tant que successeur du travail de surveillance. Ce type de travail est identifié par le libellé +AUTOGEN+ sur la ligne de commande et par une icône

comportant le caractère A dans Dynamic Workload Console. Pour qu'IBM Workload Scheduler vérifie la condition une seule fois et s'arrête une fois celle-ci remplie, cochez la case **Démarrer une fois** dans le Concepteur de charge de travail, ou omettez le mot clé **rerun** sur la ligne de commande **composer**.

Le travail Soumission d'un flot de travaux est défini sur un poste de travail de pool spécifique, MASTERAGENTS. Ce dernier contient l'agent dynamique installé sur le gestionnaire de domaine maître et sur le gestionnaire de domaine de sauvegarde, s'il est présent. L'agent dynamique installé sur le gestionnaire de domaine maître et le gestionnaire de domaine de sauvegarde (s'il est présent) sont automatiquement ajoutés lors de l'installation à ce poste de travail de pool. Si vous supprimez le poste de travail de pool MASTERAGENTS et que vous le recréez ensuite, vous devez arrêter et redémarrer l'agent dynamique pour l'ajouter à nouveau au poste de travail de pool MASTERAGENTS

Le travail Soumission d'un flot de travaux crée une instance du flot de travaux dans lequel la condition de démarrage est définie. Par défaut, la nouvelle instance de flot de travaux démarre également si l'instance précédente est toujours en cours d'exécution et les deux instances sont exécutées simultanément. Pour modifier ce comportement, dans le Concepteur de charge de travail, passez à l'onglet **Options de planification** et sélectionnez **Mettre en file d'attente une nouvelle instance** dans la section **Actions**. A partir de la ligne de commande **composer**, utilisez le mot clé **onoverlap**. Pour plus d'informations, voir «onoverlap», à la page 274. La nouvelle instance générée est identique à la précédente et définie pour répéter la vérification de la condition. Une série de nouvelles instances est donc créée jusqu'à ce que l'échéance du flot de travaux soit atteinte.

Le nom du travail Soumission d'un flot de travaux est défini à l'aide de la valeur indiquée pour l'option **optman resubmitJobname**. Par défaut, la valeur attribuée à cette option est **restart_StartCond**. Pour plus d'informations sur l'option **resubmitJobname**, voir la description des options globales dans le Guide d'administration.

Syntaxe

startcond filecreated | filemodified | job

startcond filecreated *nom_poste_de_travail#nom_fichier* **user** *nom_utilisateur*
interval *secondes* [**alias** *nom_travail_condition_démarrage* **rerun** **batch** **outfile**
nom_fichier_sortie **params** "*paramètres supplémentaires filemonitor*"]

startcond filemodified *nom_poste_de_travail#nom_fichier* **user** *nom_utilisateur*
interval *secondes* [**alias** *nom_travail_condition_démarrage* **rerun** **batch** **outfile**
nom_fichier_sortie **params** "*paramètres supplémentaires filemonitor*"]

startcond job *nom_poste_travail#nom_travail* **outcond** *condition_sortie_travail*
interval *secondes* [**alias** *nom_travail_condition_démarrage* **rerun**]

Arguments

filecreated *nom_poste_travail#nom_fichier*

Vérifie si le ou les fichiers indiqués sont créés, où :

nom_poste_travail#nom_fichier

Indique le nom du poste de travail et le chemin d'accès qualifié complet au(x) fichier(s) à surveiller.

filemodified *nom_poste_travail#nom_fichier*

Vérifie si le ou les fichiers indiqués sont modifiés, où :

nom_poste_travail#nom_fichier

Indique le nom du poste de travail et le chemin d'accès qualifié complet au(x) fichier(s) à surveiller.

job *nom_poste_travail#nom_travail*

Vérifie si le travail indiqué a rempli avec succès la condition indiquée avec le mot clé **outcond**.

user *nom_utilisateur*

Informations de connexion permettant d'accéder au poste de travail où se trouvent le ou les fichiers à surveiller. Applicable uniquement aux mots clés **filecreated** et **filemodified**.

interval *secondes*

Indique la fréquence à laquelle IBM Workload Scheduler vérifie si la condition est remplie, exprimée en secondes. Pour la condition de démarrage **job** uniquement, la valeur est arrondie à 60 secondes, si elle est inférieure à 60 secondes. Si la valeur est supérieure à 60 secondes et n'est pas divisible par 60, elle est évaluée à la valeur la plus proche qui est également divisible par 60.

alias *nom_travail_condition_démarrage*

Il s'agit du nom du travail qui est ajouté automatiquement au plan pour exécuter la tâche de surveillance.

Pour les conditions de type filecreated ou filemodified,

si vous n'indiquez pas de nom, **file_StartCond** est utilisé par défaut. Le nom par défaut est extrait de l'option globale **fileStartConditionJobName**. Pour plus d'informations, voir la section relative aux options globales dans le document Guide d'administration

Pour les conditions de type job,

si vous n'indiquez pas de valeur, le nom de la définition de travail est utilisé.

rerun Permet à IBM Workload Scheduler de créer automatiquement un travail Soumission d'un flot de travaux qui est ajouté en tant que successeur du travail de surveillance. Le travail Soumission d'un flot de travaux soumet une instance du flot de travaux dans lequel la condition de démarrage est définie.

batch Si le processus renvoie simultanément plusieurs fichiers, une même instance de flot de travaux est utilisée pour les traiter par lots. Si vous ne spécifiez pas ce paramètre, une instance de flot de travaux est lancée pour chaque fichier extrait. Applicable uniquement aux mots clés **filecreated** et **filemodified**.

outfile *nom_fichier_sortie*

Les noms du ou des fichiers extraits sont renvoyés sous forme de variable. Vous pouvez éventuellement spécifier un fichier de sortie pour le stockage des noms de fichier. Vérifiez que le répertoire dans lequel le fichier de sortie doit être créé existe déjà. Applicable uniquement aux mots clés **filecreated** et **filemodified**.

params "*paramètres filemonitor supplémentaires*"

En option, spécifiez des paramètres **filemonitor** supplémentaires.

Applicable uniquement aux mots clés **filecreated** et **filemodified**. Pour plus d'informations, voir «Filemonitor», à la page 698.

outcond

Condition de sortie, qui lorsqu'elle est remplie, libère la partie restante du flot de travaux. Applicable uniquement au mot clé **job**. Vous pouvez spécifier ce mot clé aux niveaux du flot de travaux et du travail. Lorsque vous sauvegardez le flot de travaux, le travail redémarre jusqu'à ce que la condition soit remplie ou que l'échéance du travail soit atteinte.

Commentaires

Assurez-vous que tous les composants de l'environnement IBM Workload Scheduler sont à la version 9.4, groupe de correctifs 1, ou version ultérieure.

Les types de poste de travail suivants ne sont pas pris en charge si vous spécifiez une condition de démarrage basée sur des fichiers créés ou modifiés :

- Agent étendu
- Courtier de charge de travail
- Moteur distant

Exemples

L'exemple suivant illustre un flot de travaux qui démarre uniquement lorsque le fichier Reports.txt est créé sur le poste de travail S_MDM dans le chemin /Reports. L'utilisateur FileMngr est utilisé pour se connecter au poste de travail spécifié et la vérification de la condition est effectuée toutes les 100 secondes. Le travail de surveillance est nommé ReportCheck et les informations extraites sont stockées dans le fichier /logs/ReportsOutput.txt. A l'aide du mot clé **params**, deux paramètres **filemonitor (-recursive -maxEventsThreshold)** ont été insérés pour indiquer que la vérification est effectuée également sur les sous-dossiers et tous les événements doivent être retournés.

```
SCHEDULE S_MDM#JS1
VARIABLE MAIN_TABLE
ON RUNCYCLE RC2 08/04/2017
( AT 0800 +1 DAYS )
STARTCOND FILECREATED S_MDM#/Reports/Report.txt USER FileMngr INTERVAL 100
( ALIAS ReportCheck PARAMS "-recursive -maxEventsThreshold all" OUTFILE /logs/ReportsOutput.txt )
LIMIT 5
OPENS S_MDM#"/my file . txt" (-f %p)
:
S_MDM#NATIVE
NOP
FOLLOWS DYNAMIC

S_AGT#DYNAMIC

S_AGT#SLEEP3
FOLLOWS NATIVE
FOLLOWS S_MDM#JS1_EXT.SLEEP3

S_AGT#JOB_MGMT
FOLLOWS SLEEP3
END
```

timezone

Indique (au niveau du flot de travaux) le fuseau horaire utilisé pour calculer l'heure à laquelle le flot de travaux doit démarrer le traitement.

Syntaxe

timezone | **tz** *fuseau_horaire*

Arguments

tzname Indique le nom du fuseau horaire. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires.

Commentaires

Le fuseau horaire spécifié au niveau du flot de travaux s'applique à toutes les définitions temporelles des cycles d'exécution et des restrictions temporelles (définies par les mots clés **at**, **deadline**, **schedtime** et **until**).

Si vous indiquez un fuseau horaire pour le flot de travaux et un autre pour la restriction temporelle, ils doivent être identiques.

Si vous n'indiquez aucun fuseau horaire, tant au niveau du flot de travaux que de la restriction temporelle, le fuseau horaire spécifié sur le poste de travail est utilisé.

until

Selon la définition d'objet à laquelle appartient le mot clé **until**, spécifie la dernière heure à laquelle un flot de travaux doit être terminé ou la dernière heure à laquelle un travail peut être lancé. Ce mot clé et le mot clé **jsuntil** s'excluent mutuellement. Pour plus d'informations sur le mot clé **jsuntil**, voir «jsuntil», à la page 257.

Syntaxe

[**until** *heure* [**timezone** | **tz** *fuseau-horaire*][**+n** **day**[**s**]][**absolute** | **abs**][**onuntil** *action*]]

Arguments

heure Indique l'heure. Les valeurs admises sont comprises entre 0000 et 2359.

timezone *fuseau_horaire*

Indique le fuseau horaire à utiliser lors du calcul de l'heure. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms des fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

Remarque : Si une heure **until** et une heure **at** ou **deadline** sont spécifiées, les fuseaux horaires doivent être identiques.

day *n* Indique un décalage, en jours, à partir de la date et de l'heure de début planifiées.

absolute

Indique que la date **until** est fonction du jour calendaire plutôt que du jour de production.

onuntil *action*

Selon la définition d'objet à laquelle appartient le mot clé **until**, spécifie :

- l'action à effectuer sur un travail ou flot de travaux dont l'heure **until** a été atteinte mais qui n'a pas encore démarré.
- L'action à prendre sur un flot de travaux dont le temps **until** a expiré alors que ce flot de travaux ne s'est pas encore terminé à l'état SUCC.

Les valeurs possibles du paramètre de l'*action* sont :

/

/ **suppr** Le travail ou le flot de travaux et tous les travaux ou flots de

/ travaux dépendants ne s'exécutent pas. Il s'agit du comportement

/ par défaut.

/

/ Une fois que l'heure **until** est arrivée à expiration pour un flot de

/ travaux, l'état de ce dernier est calculé en fonction des règles

/ habituelles ; les travaux supprimés ne sont pas pris en compte

/ dans ce calcul. Dans le cas où le flot de travaux contient au moins

/ un travail **every**, il est à l'état HOLD.

/

/ Une fois que l'heure **until** arrive à expiration pour un travail, ce

/ dernier passe à l'état HOLD ou conserve tout état antérieur qui soit

/ un état final.

/

/ Si l'heure **until** est transmise avec les options **onuntil suppr** et

/ **carryforward**, le flot de travaux est *reporté* par **JnextPlan**

/ uniquement si la date **until** correspond à la date d'exécution de

/ JnextPlan. Si les dates d'exécution de **until** et JnextPlan ne sont

/ pas identiques, le flot de travaux n'est pas *reporté*.

/

/ **cont** Le travail ou flot de travaux s'exécute lorsque toutes les conditions

/ requises sont remplies. Un message apparaît alors dans le journal

/ lorsque l'heure définie par **until** est dépassée.

/

/ Si l'heure **until** est transmise avec les options **onuntil cont** et

/ **carryforward**, le flot de travaux est toujours *reporté* par **JnextPlan**.

/

/ **canc** Un travail ou flot de travaux est annulé lorsque l'heure **until**

/ définie est atteinte. Lorsque le mot clé **onuntil canc** est utilisé sur

/ des travaux, l'opération d'annulation sur le travail est émise par

/ l'agent FTA sur lequel ce travail s'exécute. Tout travail ou flot de

/ travaux qui dépendait de l'accomplissement d'un travail ou flot de

/ travaux qui a été annulé, s'exécute car la dépendance n'est plus

/ appliquée.

/

/ Si l'heure **until** est transmise avec les options **onuntil canc** et

/ **carryforward**, le flot de travaux n'est pas *reporté* par **JnextPlan** car

/ il est déjà annulé.

/

/ **Remarque :** Lorsque le mot clé **onuntil canc** est utilisé au niveau

/ d'un flot de travaux, définissez comme propriétaire du flot de

/ travaux le poste de travail le plus haut placé dans la hiérarchie de

/ l'environnement de planification parmi tous les postes de travail

/ qui possèdent des travaux appartenant à ce flot de travaux.

/

/ **Remarque :**

/

/ • Les mots clés **until** et **deadline** peuvent être utilisés dans la même définition,

/ mais ils doivent être exprimés avec le même paramètre de fuseau horaire.

|

| • Si une heure **until** (dernière heure de début) n'est pas indiquée pour un flot de

| travaux, alors une heure **until** par défaut peut être définie à l'aide de l'option

| globale **untilDays**. L'heure **until** est calculée en ajoutant la valeur de l'option

| globale **untilDays**, exprimée en nombre de jours, à l'heure planifiée pour le flot

| de travaux. Si l'option *enCarryForward* est définie sur **all**, et que le nombre de

| jours spécifié pour *untilDays* est atteint, alors toutes les instances de flot de

| travaux du plan qui ont échoué sont automatiquement supprimées du plan et ne

| sont pas ajoutées au nouveau plan de production. Si la valeur par défaut de

| **untilDays** est conservée (0), alors aucune heure par défaut n'est définie pour

| l'heure **until**.

Exemples

Dans l'exemple suivant, le programme interdit tout lancement du processus sked1 après 17 heures les mardis :

```
schedule sked1 on tu until 1700:
```

Le programme lance sked1 à 17:00, lorsque son heure **until** est atteinte :

```
schedule sked1 until 1700 onuntil cont
```

Le programme lance le travail job1 entre 13 et 17 heures pendant les jours de la semaine :

```
schedule sked2 on weekdays:
  job1 at 1300 until 1700
end
```

Dans l'exemple suivant, le programme lance le travail joba toutes les 15 minutes entre 22:30 et 23:30 les lundis :

```
schedule sked3 on mo:
  joba at 2230 every 0015 until 2330
end
```

Dans l'exemple suivant, le programme lance le flot de travaux sked4 les dimanches entre 8 et 13 heures. Les travaux doivent être lancés selon l'intervalle suivant :

```
schedule sked4 on fr at 0800 + 2 days
  until 1300 + 2 days
:
  job1
  job2 at 0900 <<launched on sunday>>
  job3 follows job2 at 1200 <<launched on sunday>>
end
```

Dans l'exemple suivant, le programme lance le flot de travaux sked8 tous les jours de la semaine à 16:00 et il doit être terminé à 17:00. Si le flot de travaux n'est pas terminé à 17:00, il est considéré comme étant en retard. Les travaux doivent être lancés comme suit : job1 démarre à 16:00 ou au plus tard à 16:20. Si le travail job1 n'a pas encore démarré à 16:20, le programme consigne un message dans le journal et job1 commence son exécution. Quant au travail job2, il s'exécute à 16:30 ou au plus tard à 16:50. Si le travail job2 n'a pas encore démarré à 16:50, il est annulé.

```
schedule sked8 on weekdays at 1600 deadline 1700 :
  job1 at 1600 until 1620 onuntil cont
  job2 at 1630 until 1650 onuntil canc
end
```

Dans l'exemple suivant, le programme lance le flot de travaux sked01. Si l'événement **until** intervient, le programme exécute le flot de travaux sked02 car le flot de travaux sked01 passe à l'état SUCC. Par contre, le flot de travaux sked03 n'est pas exécuté car il possède une dépendance horaire ponctuelle sur le travail job01, qui n'a pas été libérée.

```
SCHEDULE sked01 on everyday:
  job01 until 2035 onuntil suppr
end

SCHEDULE sked02 on everyday follows sked01.@
:
  job02
end
```

```

/ SCHEDULE sked03 on everyday follows sked01.job01
/ :
/ job03
/ FIN

```

validfrom/validto

Vous pouvez définir pour un flot de travaux un intervalle de validité qui correspond à une période du plan de préproduction et qui comprend le flot de travaux. L'intervalle de validité est défini à l'aide du mot clé **validfrom** dans la définition du flot de travaux.

Syntaxe

validfrom *date*

Arguments

validfrom *date*

Définit la date à partir de laquelle le flot de travaux est actif. Il doit faire partie d'un plan de production si celui-ci inclut cette date.

Commentaires

Différentes *versions* du même flot de travaux peuvent être définies via la création de différents flots de travaux ayant le même nom et le même poste de travail mais pas le même intervalle de validité. Le concept de versions du même flot de travaux partageant les mêmes valeurs *nom_flot_travaux* et *nom_poste_travail* est essentiel lors de la gestion de la dépendance par rapport à ce flot de travaux. En fait, lorsque vous définissez une dépendance de prédécesseur/successeur externe par rapport à un flot de travaux, vous identifiez le flot de travaux remplacé en utilisant les valeurs *nom_flot_travaux* et *nom_poste_travail*. Le flot de travaux identifié comme dépendance est celui dont l'intervalle de validité se situe dans la période d'activité de la dépendance.

Si vous modifiez la valeur *nom_flot_travaux* ou *nom_poste_travail* dans une version du flot de travaux, la modification est répercutée dans toutes ses versions.

Si vous verrouillez une version du flot de travaux, toutes les versions de ce flot de travaux le sont également.

Si vous modifiez le nom d'un travail défini dans une version du flot de travaux, le nouveau nom de travail est propagé dans toutes les versions du flot de travaux. Par conséquent, si vous modifiez une autre valeur que *nom_flot_travaux* ou *nom_poste_travail*, les associations entre les flots de travaux internes et externes restent identiques.

Lors de la définition d'une version d'un flot de travaux, il vous suffit d'entrer la date **validfrom** pour que la date **validto** soit automatiquement définie sur la valeur de la date **validfrom** de la version suivante. La date **validto** s'affiche lors de l'exécution des commandes **list** et **display** lorsque la valeur 120 est attribuée à *MAESTROCOLUMNS*. Les différentes versions du même flot de travaux continuent à partager les zones du nom et du poste de travail après leur création. Si vous modifiez le nom d'une version ou le poste de travail sur lequel il a été défini, la modification s'applique à toutes les versions de ce flot de travaux.

Remarque : Si les mots clés utilisés dans la définition du flot de travaux sont **validfrom** et **validto**, les mots clés de filtrage correspondants utilisés lors de

/ l'émission de commandes sur des définitions stockées dans la base de données sont
/ **validfrom** and **validto**. Pour plus d'informations, voir Chapitre 10, «Gestion des
/ objets dans la base de données - composer», à la page 321.

/ La date spécifiée comme valeur **validto** n'est pas incluse dans le cycle d'exécution
/ et le flot de travaux n'est donc pas actif à cette date.

table de variables

Grâce à ces tables, vous attribuez différentes valeurs à la même variable, et réutilisez donc la même variable dans les définitions de travail et lors de la définition des invites et des dépendances de fichier.

Syntaxe

vartable *nom_table*

Arguments

vartable *nom_table*

Nom de la table de variables. Ce nom peut comporter jusqu'à 80 caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre.

Définition de règle d'événement

Une règle d'événement de planification définit un ensemble d'actions qui doivent être exécutées à l'occurrence de conditions d'événement spécifiques. La définition d'une règle d'événement établit une corrélation entre des événements et déclenche des actions.

Une définition de règle d'événement est identifiée par un nom de règle et par un ensemble d'attributs qui spécifient si la règle est en mode brouillon ou non, l'intervalle de temps pendant lequel elle est active, la durée de sa validité et d'autres informations nécessaires pour décider du moment où les actions sont déclenchées. Elle inclut des informations liées aux événements spécifiques (eventCondition) que la règle doit détecter et aux actions spécifiques qu'elle doit déclencher en cas de détection ou de dépassement du délai d'attente de ces événements(action). Des règles complexes peuvent inclure plusieurs événements et plusieurs actions.

Pour une présentation de la planification des règles d'événement, voir Chapitre 8, «Exécution de l'automatisation de la charge de travail gérée par événements», à la page 141.

Syntaxe

Vous définissez des règles d'événement directement en XML avec tout éditeur XML. Vous pouvez configurer une variable d'environnement sur votre ordinateur pour ouvrir automatiquement un éditeur XML de votre choix afin de travailler avec les définitions de règle d'événement. Pour de plus amples informations, voir «Editeur Composer», à la page 323. Le langage XML décrivant la règle d'événement doit correspondre au schéma de langage de règle défini dans les fichiers EventRules.xsd et FilteringPredicate.xsd.

Les schémas linguistiques des règles définies dans les fichiers eventRules.xsd et FilteringPredicate.xsd permettent de valider les définitions de règles et, selon

l'éditeur XML dont vous êtes équipé, de fournir une aide à la syntaxe. Les fichiers sont situés dans le sous-répertoire schemas du répertoire d'installation de IBM Workload Scheduler.

Le tableau suivant répertorie tous les éléments de langage utilisés pour définir une règle d'événement. Le tableau 36 explique la signification de la notation qui suit chaque élément de langage. *n* représente un nombre non limité.

Tableau 36. Explication de la notation définissant le nombre d'occurrences pour un élément de langage.

Notation	Signification
(0, 1)	0 indique que l'élément de langage est facultatif. 1 indique que s'il est codé, une seule occurrence est autorisée.
(0, n)	0 indique que l'élément de langage est facultatif. n indique que s'il est codé, plusieurs occurrences sont autorisées.
(1, 1)	Le premier 1 indique que l'élément de langage est obligatoire. Le second 1 indique qu'une seule occurrence est autorisée.
(1, 2)	1 indique que l'élément de langage est obligatoire. 2 indique que 2 occurrences sont obligatoires.
(1, n)	1 indique que l'élément de langage est obligatoire. n indique que plusieurs occurrences sont autorisées.

- eventRule name=" " ruleType=" " isDraft=" " (1, 1)
 - description (0, 1)
 - timeZone (0, 1)
 - validity from=" " to=" " (0, 1)
 - activeTime start=" " end=" " (0, 1)
 - timeInterval amount=" " unit=" " (0, 1)
 - eventCondition eventProvider=" " eventType=" " (1, n)
 - scope (0, 1)
 - filteringPredicate (0, 1)
 - attributeFilter name=" " operator="eq" (0, n)
 - value (1, n)
 - attributeFilter name=" " operator="ne" (0, n)
 - value (1, n)
 - attributeFilter name=" " operator="le" (0, n)
 - value (1, 1)
 - attributeFilter name=" " operator="ge" (0, n)
 - value (1, 1)
 - attributeFilter name=" " operator="range" (0, 1)
 - value (1, 2)
 - correlationAttributes (0, 1)
 - attribute name=" " (1, n)
 - action actionProvider=" " actionType=" " responseType=" " (0, n)
 - description (0, 1)
 - scope (0, 1)
 - parameter name=" " (1, n)
 - value (1, 1)

Les définitions de règle d'événement sont regroupées en ensembles de règle d'événement.

- eventRuleSet (1, 1)

- eventRule (1, n)

Utilisez l'élément de langage eventRuleSet également si vous devez entourer une seule définition de règle.

Remarque : Aucun commentaire que vous écrivez en XML, selon la forme `<!--texte-->`, n'est enregistré dans la base de données. A l'ouverture suivante d'une définition de règle, les commentaires que vous avez écrits la première fois n'apparaissent pas. Préférez l'attribut description pour écrire des informations additionnelles.

Arguments

Les mots clés décrivant une règle d'événement correspondent aux balises XML suivantes :

eventRule

L'objet de planification qui englobe la définition de plusieurs conditions d'événement et de plusieurs actions de règle en complément à un ensemble d'attributs qui définissent le moment où la règle est activée. Un élément eventRule inclut généralement :

- Un grand nombre d'attributs de règle obligatoires et facultatifs
- Une ou plusieurs conditions d'événement
- Une ou plusieurs actions de règle, bien que des règles sans action soient également autorisées.

Les attributs de règle sont :

- Attributs obligatoires :

name Le nom de la règle d'événement. Il peut avoir jusqu'à 40 caractères alphanumériques et doit commencer par une lettre. Il ne peut pas contenir d'espaces. Les caractères de soulignement (`_`) et le tiret (`-`) sont autorisés.

ruleType

Le type de règle est basé sur le nombre d'événements et sur leur corrélation que la règle est définie pour détecter. Il peut s'agir de l'une des règles suivantes :

filter La règle est activée sur détection d'un événement spécifique unique.

sequence

La règle est activée lorsqu'une séquence ordonnée d'événements survient dans un intervalle de temps spécifique.

set La règle est activée lorsqu'une séquence non ordonnée d'événements survient dans un intervalle de temps spécifique.

Les règles de type set et sequence peuvent également être activées sur **timeout**, lorsque un ou plusieurs événements se produisent mais que la séquence complète ne se produit pas dans la fenêtre de temps spécifiée.

isDraft

Indique si la définition de règle est actuellement activée. Les valeurs peuvent être **yes** ou **no**. La valeur par défaut est **no**.

- Attributs facultatifs :

description

Une description de la règle. Il peut contenir jusqu'à 120 caractères.

Vous devez écrire tout caractère spécial XML que vous pourriez utiliser selon la notation XML spéciale, tel que :

- `&` pour `&`
- `>` pour `>`
- `<` pour `<`
- `"` pour `"`

etc.

timeZone

Indique un fuseau horaire différent pour l'exécution de la règle. Le fuseau horaire par défaut est celui défini sur le poste de travail où réside le serveur de traitement d'événements.

L'application de l'heure d'été (DST) a un impact sur l'intervalle `activeTime` (décrit plus loin) des règles d'événement :

- Au jour d'activation de l'heure d'été (l'horloge étant alors avancée d'une heure), l'exécution des règles comprises dans l'heure absorbée par l'application de l'heure d'été est retardée d'une heure. Exemple : 2:10 devient 3:10.
- Au jour de désactivation de l'heure d'été (l'horloge étant alors retardée d'une heure), les règles comprises dans l'heure dupliquée par l'application de l'heure d'été sont considérées comme devant être exécutées sans décalage.

validity

Indique la période de validité de la règle en termes de :

from *aaaa-mm-jj*

La période de validité commence à minuit (du fuseau horaire de la règle) le jour spécifié.

to *aaaa-mm-jj*

La période de validité se termine à minuit (du fuseau horaire de la règle) le jour spécifié.

activeTime

Spécifie la fenêtre de temps d'activité de la règle pour chaque jour de validité en termes de :

start *hh:mm:ss*

Le début de la fenêtre de temps lorsque la règle est active en heures, minutes et secondes.

end *hh:mm:ss*

La fin de la fenêtre de temps pendant laquelle la règle est active en heures, minutes et secondes. Si l'heure est antérieure à **start** *hh:mm:ss*, elle fait référence au jour suivant.

timeInterval

S'applique aux règles qui incluent des actions de dépassement de délai (timeout). Indique l'intervalle de temps à l'intérieur duquel tous les événements spécifiés dans la règle doivent avoir été reçus avant qu'une action de délai de dépassement corrective ne soit démarrée. L'intervalle de temps commence au moment

où le premier événement spécifié dans la règle est détecté.
Indique l'intervalle de temps en termes de :

quantité

La durée de l'intervalle de temps en unités de temps.

unit L'unité de temps est l'une des suivantes :

- heures
- secondes
- millisecondes

Cet attribut est obligatoire lorsqu'il y a des actions de dépassement de délai dans la définition de règle d'événement.

eventCondition

La condition d'événement est composée des attributs suivants :

- Attributs obligatoires :

eventProvider

Identifie le fournisseur de moniteur d'événements qui peut capturer un type d'événement. Les fournisseurs d'événements fournis à l'installation sont :

TWSObjectsMonitor

Surveille le statut des objets du plan IBM Workload Scheduler. Ce fournisseur d'événements s'exécute sur chaque agent IBM Workload Scheduler et envoie les événements au serveur de traitement d'événement.

TWSApplicationMonitor

Surveille les processus IBM Workload Scheduler, les systèmes de fichiers et la boîte de message.

FileMonitor

Surveille les événements affectant les fichiers.

eventType

Indique le type d'événement à surveiller. Chaque événement peut être associé à un fournisseur d'événements. Les tableaux suivants répertorient les types d'événement par fournisseur d'événements.

Pour afficher les propriétés de chaque type d'événement, accédez à l'Annexe A, «Automatisation de la charge de travail commandée par les événements, définitions des événements et des actions», à la page 873.

Le tableau 37 répertorie les événements TWSObjectsMonitor.

Tableau 37. Événements TWSObjectsMonitor.

Type d'événement	Heure d'envoi de l'événement.
JobStatusChanged	Modification du statut d'un travail
JobUntil	La dernière heure de début possible d'un travail est dépassée
JobSubmit	Un travail est soumis
JobCancel	Un travail est annulé
JobRestart	Un travail est relancé
JobLate	Un travail prend du retard

Tableau 37. Evénements *TWSObjectsMonitor*. (suite)

Type d'événement	Heure d'envoi de l'événement.
JobPromoted	Un travail dans un réseau critique approche l'heure de début critique et n'a pas encore démarré
JobRiskLevelChanged	• Un nouveau travail critique est ajouté au plan avec le niveau de risque défini sur élevé ou potentiel • Le niveau de risque d'un travail est défini sur risque élevé ou risque potentiel et évolue • Un travail critique avec le niveau de risque défini sur élevé ou potentiel est supprimé du plan Un événement n'est pas envoyé si le travail critique se trouve dans le flot de travaux nommé JOBS.
JobExceededMaximumDuration	Un travail dépasse la durée maximale établie pour le travail
JobDidnotReachMinimumDuration	Un travail ne s'exécute pas assez longtemps pour atteindre la durée minimale établie pour le travail
JobStreamStatusChanged	Le statut d'un flot de travaux change
JobStreamCompleted	Un flot de travaux s'est achevé
JobStreamUntil	La dernière heure de début possible d'un flot de travaux est dépassée
JobStreamSubmit	Un flot de travaux est soumis
JobStreamCancel	Un flot de travaux est annulé
JobStreamLate	Un flot de travaux prend du retard
WorkstationStatusChanged	Un poste de travail est démarré ou arrêté
ApplicationServerStatusChanged	L'instance WebSphere Application Server s'est arrêtée ou redémarre
ChildWorkstationLinkChanged	Le poste de travail parent défini dans la règle d'événement établi ou supprime un lien avec son poste de travail parent (le poste de travail parent envoie l'événement)
ParentWorkstationLinkChanged	Le poste de travail parent établit ou supprime un lien avec le poste de travail défini dans la règle d'événement (le poste de travail enfant envoie l'événement)
PromptStatusChanged	Une invite est émise ou reçoit une réponse
ProductAlert	Le fichier Symphony du poste de travail spécifié dans l'événement contient un enregistrement endommagé

Remarque :

Toute modification effectuée sur un poste de travail référencé dans une règle n'est pas signalée dans la règle. Par exemple, si vous modifiez, mettez à jour ou supprimez un poste de travail qui est référencé dans une règle, la règle ne tient pas compte du changement et continue de considérer le poste de travail comme il était quand il a été inclus dans la règle. Une règle dont le type d'événement est **ParentWorkstationLinkChanged** n'est pas applicable lorsque le poste de travail des filtres est défini comme agent, pool, pool dynamique ou moteur distant et que l'attribut **ParentWorkstation** est défini sur courtier. Pour

surveiller une modification changement de statut de lien entre le serveur courtier de la charge de travail et un poste de travail géré par un serveur courtier d'une charge de travail, définissez une règle sans type d'événement égal à **ChildWorkstationLinkChanged**.

Une règle avec type d'événement égal à **ChildWorkstationLinkChanged** fonctionne uniquement lorsque le poste de travail courtier est associé, dissocié, arrêté ou démarré. Si le changement de statut du lien est dû à une opération d'arrêt ou de démarrage sur le poste de travail agent avec les commandes **StartupLwa** et **ShutdownLwa**, aucune action n'est initialisée. Pour surveiller ou démarrer des opérations sur les postes de travail agent, définissez une règle avec type d'événement égal à **WorkstationStatusChanged**.

Le tableau 38 répertorie les événements TWSApplicationMonitor.

Tableau 38. Événements TWSApplicationMonitor.

Type d'événement	Heure d'envoi de l'événement.
MessageQueuesFilling	Une boîte aux lettres dépasse la valeur totale du pourcentage.
TivoliWorkloadSchedulerFileSystemFilling	Le système de fichiers où l'instance IBM Workload Scheduler est installée dépasse la valeur totale du pourcentage.
TivoliWorkloadSchedulerProcessNotRunning	Un processus spécifié n'est pas en cours d'exécution.

Le tableau 39 répertorie les événements FileMonitor.

Tableau 39. Événements FileMonitor.

Type d'événement	Heure d'envoi de l'événement.
FileCreated	Un fichier est créé
FileDeleted	Un fichier est supprimé
ModificationCompleted	Un fichier est modifié (l'événement est envoyé uniquement si deux cycles de surveillance consécutifs se sont écoulés depuis la création ou la modification du fichier sans changement additionnel détecté)
LoggedMessageWritten	Une chaîne spécifique est trouvée dans un fichier (cet événement peut être utilisé pour surveiller les journaux d'application ou système)

- Attributs facultatifs :

scope Un ou plusieurs attributs de qualification qui décrivent l'événement. Il peut contenir jusqu'à 120 caractères. La portée est automatiquement générée à partir de ce qui est défini dans le fichier XML. Elle ne peut pas être définie par les utilisateurs.

filteringPredicate

Le prédicat de filtrage définit les conditions d'événement qui doivent être surveillées pour chaque type d'événement. Il est composé de :

attributeFilter

Le filtre d'attribut est un attribut particulier du type d'événement qui doit être surveillé :

- est défini par les éléments suivants :

name L'attribut ou nom de propriété du type d'événement qui doit être surveillé. Voir «Fournisseurs d'événements et définitions», à la page 873 pour une liste des noms de propriété pour chaque type d'événement.

opérateur

peut être :

- eq (égal à)
- ne (différent de)
- ge (égal ou supérieur à)
- le (égal ou inférieur à)
- range (plage)

- Inclut une ou plusieurs :

value La valeur avec laquelle l'opérateur doit concorder.

Remarque : Chaque type d'événement a un grand nombre d'attributs obligatoires ou noms de propriété. Les noms de propriété obligatoires n'ont pas tous des valeurs par défaut. Tous les noms de propriété obligatoires sans valeur par défaut doivent avoir un prédicat de filtrage.

correlationAttributes

Les attributs de corrélation permettent de diriger la règle pour créer une instance de règle séparée pour chaque groupe d'événements qui partagent des caractéristiques communes. Généralement, chaque règle active a une seule instance de règle qui s'exécute dans le serveur de traitement d'événements. Cependant, il arrive que la même règle soit requise pour différents groupes d'événements, souvent liés à différents groupes de ressources. Utiliser un ou plusieurs attributs de corrélation permet de diriger une règle afin de créer une copie de règle séparée pour chaque groupe d'événements ayant des caractéristiques communes. A utiliser avec les types de règle set et sequence.

Vous pouvez spécifier un ou plusieurs attributs. Chaque attribut est défini par :

attribute name=" "

L'attribut objet que vous corrélerez.

action L'action qui doit être déclenchée si l'événement est détecté. Les définitions de règle d'événement avec les événements mais sans les actions sont syntaxiquement acceptées bien qu'elles puissent ne pas avoir de signification pratique. Vous pouvez enregistrer de telles règles à titre de version provisoire et ajouter des actions ultérieurement avant qu'elles ne soient déployées.

- Est défini par les attributs obligatoires suivants

actionProvider

Le nom du fournisseur d'actions qui peut mettre en œuvre une ou plusieurs actions configurables. Les fournisseurs d'actions disponibles à l'installation sont :

GenericAction

Exécute les commandes non IBM Workload Scheduler.
Les commandes sont exécutées sur le même ordinateur que celui où s'exécute le processeur d'événements.

MailSender

Connecte à un serveur de protocole SMTP et envoie un courrier électronique.

MessageLogger

Journalise l'occurrence d'une situation dans une base de données d'audit interne.

TECEventForwarder

Transmet l'événement à un serveur TEC (Tivoli Enterprise Console) externe ou à toute autre application capable d'écouter les événements au format TEC.

TWSAction

Exécute l'une des commandes conman suivantes :

- submit job (sbj)
- submit job stream (sbs)
- submit command (sbd)
- reply to prompt (reply)

TWSForZosAction

Ajoute une occurrence d'application (flot de travaux) au plan actuel sur IBM Workload Scheduler for z/OS. Ce fournisseur est à utiliser dans des configurations de planification de bout en bout IBM Workload Scheduler.

La description d'application de l'occurrence à ajouter doit exister dans la base de données AD de IBM Workload Scheduler for z/OS.

actionType

Indique le type d'action qui doit être déclenché lorsqu'un événement spécifié est détecté. Chaque action peut être associée à un fournisseur d'action. Le tableau suivant répertorie les types d'action par fournisseur d'actions.

Pour afficher les propriétés de chaque type d'événement, accédez à l'Annexe A, «Automatisation de la charge de travail commandée par les événements, définitions des événements et des actions», à la page 873.

Tableau 40. Types d'action par fournisseur d'actions.

Fournisseur d'actions	Types d'action
GenericAction	RunCommand
MailSender	SendMail
MessageLogger	PostOperatorMessage
TECEventForwarder	TECFWD
TWSAction	reply (ReplyPrompt)
	sbd (SubmitAdHocJob)
	sbj (SubmitJob)
	sbs (SubmitJobStream)
TWSForZosAction	AddJobStream

responseType

Indique le moment où l'action doit être exécutée. Les valeurs peuvent être :

onDetection

L'action démarre dès que tous les événements définis dans la règle ont été détectés. S'applique à tous les types de règle. Voir aussi «Remarques sur le fonctionnement des règles», à la page 159.

onTimeOut

L'action démarre lorsque le temps spécifié dans `timeInterval` a expiré mais que tous les événements définis dans la règle n'ont pas été reçus. S'applique uniquement aux règles `set` et `sequence`.

Notez que les actions de dépassement de délai ne sont pas exécutées si vous ne spécifiez pas un intervalle de temps. Le planificateur vous permettra cependant d'enregistrer des règles d'événement pour lesquelles des actions de dépassement de délai ont été définies sans spécifier d'intervalle de temps parce que vous pouvez définir l'intervalle de temps ultérieurement. Jusqu'à ce point, seules les actions ayant le type de réponse `onDetection` sont traitées.

Les actions de dépassement de délai d'attente pour lesquelles un intervalle de temps n'a pas été défini sont exécutées uniquement lorsque les règles sont désactivées. Une règle d'événement est désactivée dans l'un des deux cas ci-après :

- La commande `planman deploy -scratch` est exécutée
- La règle est modifiée (elle est alors désactivée dès l'exécution de la commande `planman deploy`)

Dans tous les cas, la règle est en premier lieu désactivée, puis réactivée. A cet instant, toutes les actions en attente sont exécutées.

- Inclut les attributs facultatifs suivants :

description

Une description de l'action. Il peut contenir jusqu'à 120 caractères.

Vous devez écrire tout caractère spécial XML que vous pourriez utiliser selon la notation XML spéciale, tel que :

- `&` pour `&`
- `>` pour `>`
- `<` pour `<`
- `"` pour `"`

etc.

scope Un ou plusieurs attributs de qualification qui décrivent l'action. Il peut contenir jusqu'à 120 caractères. La portée est automatiquement générée à partir de ce qui est défini dans le fichier XML. Elle ne peut pas être définie par les utilisateurs.

- Inclut une liste d'un ou plusieurs paramètres ou noms de propriété. Tous les types d'action ont au moins un paramètre obligatoire. Chaque paramètre est défini par :

parameter name=" "

Voir «Fournisseurs d'actions et définitions», à la page 885 pour une liste des paramètres ou noms de propriété disponibles pour chaque type d'action.

value Voir «Fournisseurs d'actions et définitions», à la page 885 pour une liste des valeurs possibles ou types de valeurs.

Vous pouvez utiliser la substitution de variable. Lorsque vous définissez des paramètres d'action, vous pouvez utiliser les noms de propriété des événements qui déclenchent la règle d'événement pour remplacer la valeur du nom de propriété de l'action. Pour cela, inscrivez la valeur du paramètre d'action que vous souhaitez substituer, de l'une des manières suivantes :

– `${event.property}`

Remplace la valeur telle quelle. Cette option est utile pour transmettre les informations à une action liée à celles-ci par voie de programme, par exemple l'option `schedTime` d'un flot de travaux.

– `%{événement.propriété}`

Remplace la valeur formatée en syntaxe lisible. Cette opération est utile pour transmettre les informations à une action chargée de faire suivre les informations à un utilisateur, par exemple, de formater le `schedTime` d'un flot de travaux dans le corps d'un courrier électronique.

Où :

événement

est le nom défini pour le déclenchement `eventCondition`.

property

est le nom de filtre d'attribut dans le prédicat de filtrage de la condition d'événements déclenchante. La valeur prise par le filtre d'attribut lorsque la règle est déclenchée est remplacée en tant que valeur de paramètre dans la définition d'action avant l'exécution.

Il est à noter que vous pouvez également recourir à une substitution de variable si aucune valeur de `attributeFilter` n'a été spécifiée pour un attribut, ainsi que lorsque l'attribut est accessible en lecture seule.

Par exemple, la tâche d'une règle d'événement doit détecter le moment où l'un des travaux dont le nom commence par `job15` se termine par une erreur, et lorsque ceci se produit, doit soumettre à nouveau ce travail. La section `eventCondition` de la règle est codée comme suit :

```
<eventCondition
  name="event1"
  eventProvider="TWSObjectsMonitor"
  eventType="JobStatusChanged">
  <filteringPredicate>
    <attributeFilter
      name="JobName"
      operator="eq">
      <value>job15*</value>
    </attributeFilter>
    <attributeFilter
      name="Workstation"
      operator="eq">
      <value>*</value>
    </attributeFilter>
    <attributeFilter
      name="Status"
```

```

        operator="eq">
        <value>Error</value>
    </attributeFilter>
</filteringPredicate>
</eventCondition>

```

Des caractères génériques (* pour plusieurs caractères ou ? pour un seul caractère) sont utilisés pour généraliser les conditions d'événement à appliquer à toutes les instances de travail dont le nom commence par job15 et à leur poste de travail associé. La substitution de variable est utilisée dans la section action pour soumettre à nouveau le travail spécifique qui s'est terminé par une erreur sur le même poste de travail. La section action est :

```

<action
    actionProvider="TWSAction"
    actionType="sbj"
    responseType="onDetection">
    <description>Soumet de nouveau le travail qui s'est terminé
    par une erreur</description>
    <parameter name="JobDefinitionName">
        <value>${event1.JobName}</value>
    </parameter>
    <parameter name="JobDefinitionWorkstationName">
        <value>${event1.Workstation}</value>
    </parameter>
</action>

```

Exemples

JOB7 a une dépendance de fichier sur DAILYOPS.XLS. Dès que le fichier est reçu, JOB7 doit commencer à le traiter. La règle suivante contrôle que JOB7 commence dans la minute qui suit le transfert de DAILYOPS.XLS. Sinon, un courrier électronique est envoyé à l'opérateur d'événement. Ceci est réalisé en définissant deux conditions d'événement séquentielles qui doivent être surveillées :

1. Le premier événement qui déclenche la règle est la création du fichier DAILYOPS.XLS sur le poste de travail vers lequel il doit être transféré. Dès que cet événement est détecté, une instance de règle est créée et un décompte d'une minute est démarré pour détecter la condition d'événement suivante.
2. Le second événement est la soumission de JOB7. Si cet événement n'est pas détecté dans l'intervalle de temps spécifié, la règle expire et l'action SendMail est démarrée.

```

<?xml version="1.0"?>
<eventRuleSet
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
    xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules
    EventRules.xsd">
    <eventRule
        name="sample_rule"
        ruleType="sequence"
        isDraft="no">
        <description>An email is sent if job JOB7 does not start within
        one minute après que le fichier DAILYOPS.XLS a été créé</description>
        <timeZone>America/Indianapolis</timeZone>
        <validity
            from="2007-01-01"
            to="2007-12-31" />
        <activeTime
            start="20:00:00"
            end="22:00:00" />
        <timeInterval
            amount="60"
            unit="seconds" />
    </eventRule>
</eventRuleSet>

```

```

<eventCondition
  name="DAILYOPS_FTPed_event"
  eventProvider="FileMonitor"
  eventType="FileCreated">
<filteringPredicate>
  <attributeFilter
    name="FileName"
    operator="eq">
    <value>c:/dailybus/DAILYOPS.XLS</value>
  </attributeFilter>
  <attributeFilter
    name="Workstation"
    operator="eq">
    <value>ACCREC03</value>
  </attributeFilter>
  <attributeFilter
    name="SampleInterval"
    operator="eq">
    <value>300</value>
  </attributeFilter>
</filteringPredicate>
</eventCondition>
<eventCondition
  name="job_JOB7_problem_event"
  eventProvider="TWSObjectsMonitor"
  eventType="JobSubmit">
<filteringPredicate>
  <attributeFilter
    name="JobStreamWorkstation"
    operator="eq">
    <value>ACCREC03</value>
  </attributeFilter>
  <attributeFilter
    name="Workstation"
    operator="eq">
    <value>ACCREC03</value>
  </attributeFilter>
  <attributeFilter
    name="JobStreamName"
    operator="eq">
    <value>JSDAILY</value>
  </attributeFilter>
  <attributeFilter
    name="JobName"
    operator="eq">
    <value>JOB7</value>
  </attributeFilter>
</filteringPredicate>
</eventCondition>
<action
  actionProvider="MailSender"
  actionType="SendMail"
  responseType="onTimeOut">
<description>Envoi d'un courrier électronique à l'opérateur du soir pour
  signaler que le travail n'a pas
  démarré</description>
<parameter name="To">
  <value>eveoper@bigcorp.com</value>
</parameter>
<parameter name="Subject">
  <value>Job JOB7 failed to start within scheduled time on
  workstation ACCREC03.</value>
</parameter>
</action>
</eventRule>
</eventRuleSet>

```

Il est à noter que la portée n'est pas indiquée lors de la définition initiale de la règle.

Pour d'autres exemples de règles d'événement, voir «Exemples de règles d'événement», à la page 152.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section Création d'une règle d'événement.

Définition des applications de charge de travail

Vous pouvez utiliser une applications de charge de travail pour normaliser une solution d'automatisation de charge de travail de sorte que la solution puisse être réutilisée dans un ou plusieurs environnements IBM Workload Scheduler automatisant de ce fait des processus métier.

Les Applications de charge de travail sont définies dans un environnement IBM Workload Scheduler également appelé environnement source, à l'aide de la ligne de commande **composer** ou de l'interface graphique du Concepteur de charge de travail accessible à partir de Dynamic Workload Console. Vous pouvez créer un modèle d'application de charge de travail, puis lui ajouter un ou plusieurs flots de travaux. Pour reproduire la solution d'automatisation de charge de travail dans un autre environnement IBM Workload Scheduler, le modèle d'application de charge de travail est exporté, puis après l'avoir personnalisé manuellement, il peut être importé dans l'environnement cible.

Le processus d'exportation génère un fichier compressé contenant tous les fichiers et informations requises pour exécuter la charge de travail dans un autre environnement IBM Workload Scheduler. Le fichier compressé contient :

Un fichier de définitions

Fichier XML, *nom du modèle d'application de charge de travail*_Definitions.UTF8.xml, qui contient les définitions de tous les objets exportés. Ces définitions seront déployées dans l'environnement cible de sorte à remplir la base de données cible avec les mêmes objets qui existent dans l'environnement source.

Fichier de mappage

Fichier de mappage, *nom du modèle d'application de charge de travail*_Mapping.UTF8.properties, requis pour traiter les objets qui sont dépendants de la topologie de l'environnement et qui ne peuvent pas être reproduits sans avoir été personnalisés manuellement. L'utilisateur cible modifie le fichier en remplaçant les noms des objets dans l'environnement source par les noms que ces objets portent dans l'environnement cible.

Un fichier d'informations de référence

Fichier d'informations de référence, *nom du modèle d'application de charge de travail*_SourceEnv_reference.txt, contenant les définitions des postes de travail utilisées dans le modèle d'application de charge de travail et d'autres informations pouvant être utiles pour mapper correctement l'environnement source dans l'environnement cible afin d'exécuter l'application de charge de travail.

Vous pouvez importer le module compressé à partir de Dynamic Workload Console en suivant une procédure intuitive décrite dans la section relative à

l'importation d'un modèle d'application de charge de travail dans le document *Dynamic Workload Console - Guide d'utilisation*.

Vous pouvez également importer manuellement le module compressé dans l'environnement cible où l'application de charge de travail doit être déployée, et créer ainsi tous les objets nécessaires dans l'environnement cible. Dans l'environnement cible, le fichier *nom d'application de charge de travail*_Mapping.UTF8.properties doit être édité manuellement en indiquant les noms des objets tels qu'ils sont définis dans l'environnement cible (par exemple, les noms des postes de travail sur lesquels s'exécutent les flots de travaux). L'opération d'importation doit être effectuée dans l'environnement cible à l'aide de la ligne de commande. Pour plus de détails, voir les sections sur les applications de charge de travail et la commande wappman dans *Guide d'utilisation et de référence*.

Lorsque vous utilisez la ligne de commande **composer**, chaque définition d'application de charge de travail possède le format et les arguments suivants :

Syntaxe

```
wat nom_wat
    [description "description"]
    [vendor "vendor"]
    jstreams
    [poste_travail#flot_travaux [poste_travail#flot_travaux]...]
end
```

Arguments

wat Zone obligatoire qui contient le nom du modèle d'application de charge de travail. La longueur est limitée à 80 caractères. Dans les versions du produit antérieures à IBM Workload Scheduler version 94, groupe de correctifs 1, ce mot clé était nommé **bapplication**.

description

Texte descriptif facultatif pour aider les utilisateurs à comprendre l'objectif et les caractéristiques de l'application de charge de travail. La longueur est limitée à 120 caractères.

vendor

Zone facultative qui indique le créateur du modèle d'application de charge de travail. Elle peut être utile pour indiquer aux utilisateurs de l'application de charge de travail qui a créé le modèle et qui l'a fourni. La longueur est limitée à 120 caractères.

jstreams

Flots de travaux que vous souhaitez ajouter au modèle d'application de charge de travail.

Exemples

Pour créer deux modèles d'application de charge de travail, WAT_NAME1 et WAT_NAME2, exécutez :

```
new wat
WAT WAT_NAME1
  DESCRIPTION "Description"
  VENDOR "Provider"
  JSTREAMS
    FTA1#JS_1_1
```

$$=$$

11

1
1
1

1
1
1

— — — — —

1
1
1
1
1
1

11

1

— — — — —

1

1

1

—

|||

306 IBMWorkload Scheduler : Guide d'utilisation et de référence

Exemples

L'exemple suivant définit une liste de contrôle d'accès sur le domaine SECDOM1 et une autre sur le domaine SECDOM2 :

```
ACCESSCONROLLIST FOR SECDOM1
  USER1 SECR0LE1, SECR0LE2, SECR0LE3
  USER2 SECR0LE4
  USER3 SECR0LE2, SECR0LE4
END

ACCESSCONROLLIST FOR SECDOM2
  USER1 SECR0LE1, SECR0LE2
  USER2 SECR0LE3
FIN
```

Définition de domaine de sécurité

Dans un modèle de sécurité basée sur les rôles, chaque domaine représente l'ensemble des objets que les utilisateurs ou les groupes peuvent gérer. Par exemple, vous pouvez définir un domaine qui contient tous les objets nommés avec un préfixe 'AA'. Si vous voulez spécifier des attributs de sécurité différents pour tout ou partie de vos utilisateurs, vous pouvez créer des domaines de sécurité supplémentaires en fonction de critères de correspondance spécifiques. Vous pouvez filtrer des objets en indiquant un ou plusieurs attributs pour chaque type d'objet de sécurité. Vous pouvez inclure ou exclure chaque attribut de la sélection. Par exemple, vous pouvez restreindre l'accès à un ensemble d'objets ayant le même nom et/ou définis sur le même poste de travail.

Vous inclure plusieurs définitions de domaine de sécurité dans le même fichier texte, avec des définitions de rôles de sécurité et des définitions de listes de contrôle d'accès.

Chaque définition de domaine de sécurité utilise le format et les arguments suivants :

Syntaxe

Chaque attribut peut être inclu ou exclu de la sélection en utilisant les signes plus (+) et tilde (~).

```
securitydomain
nom_domaine_sécurité
  [description "description"]
  [common [[+|~]attribut_objet [=
valeur |
@[, valeur | @]...]]]
  [type_objet [[+|~]attribut_objet [=
valeur | @[, valeur | @]...]]
  [type_objet [[+|~]attribut_objet [=
valeur | @[, valeur | @]...]]]...
end
```

[**securityrole** ...]

[**accesscontrollist** ...]

Arguments

securitydomain *nom_domaine_sécurité*

Indique le nom du domaine de sécurité. Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut comporter jusqu'à 16 caractères.

description *"description"*

Donne une description du domaine de sécurité. La description peut contenir jusqu'à 120 caractères alphanumériques. Ce texte doit être délimité par des guillemets.

common *[[+|~]attribut_objet [= valeur | @[, valeur | @]...]]*

Fournit tous les attributs d'objet communs à tous les types d'objet de sécurité.

type_objet *[[+|~]attribut_objet [= valeur | @[, valeur | @]...]]*

Pour chaque type d'objet, spécifie les attributs qui s'appliquent au type d'objet en question, ainsi que les valeurs associées. Chaque attribut peut être inclu ou exclu de la sélection en utilisant les signes plus (+) et tilde (~). Le caractère générique (@) est pris en charge pour la valeur d'attribut : *attribut_objet* =@ signifie que tous les objets correspondant à l'attribut d'objet doivent être inclus dans le domaine. Pour l'utilisation du caractère générique (@), voir les exemples ci-dessous.

Pour les attributs que vous pouvez spécifier pour chaque type d'objet de sécurité, voir la section relative à la gestion de la sécurité avec Dynamic Workload Console, dans *Dynamic Workload Console - Guide d'utilisation*.

Pour les valeurs que vous pouvez spécifier pour chaque attribut d'objet, voir la section relative à la gestion de la sécurité avec Dynamic Workload Console, dans *Dynamic Workload Console - Guide d'utilisation*.

Exemples

L'exemple suivant définit un domaine de sécurité nommé SECDOM1 et un autre nommé SECDOM2:

```
securitydomain SECDOM1
description "Sample Security Domain1"
job      cpu = $THISCPU, # The workstation where the user logs on
          $MASTER, # The master workstation
          $SLAVES, # Any fault tolerant agent
          $REMOTES # Any standard agent
          cogs@    # Any workstation whose name starts with "cogs"
+ name =  A@      # Any job whose name starts with "A"
△ name =  A2@     # but doesn't start with A2
+ jcltype = SCRIPTNAME # Allow only SCRIPTNAME type of job definition
+ jcltype = DOCOMMAND  # Allow only DOCOMMAND type of job definition
+ logon =  $USER, # Streamlogon is the conman/composer user
          $OWNER, # Streamlogon is the job creator
          $JCLOWNER, # Streamlogon is the OS owner of the file
          $JCLGROUP # Streamlogon is the OS group of the file
△ logon =  root, twsuser # The job cannot logon as "root" or "twsuser"
+ jcl =    "/usr/local/bin/@" # The jobs whose executable file that is present in /usr/local/bin
△ jcl =    "@rm@" # but whose JSDCL definition does not contain the string "rm"
end

securitydomain SECDOM2
description "Sample Security Domain2"
common     cpu=@+name=@
userobj    cpu=@
job        cpu=@
```

```

|         schedule    cpu=@+name=AP@
|         resource    cpu=@
|         invite
|         file        name=@
|         cpu         cpu=@
|         parameter   cpu=@
|         calendar
|         report      name=@
|         entrule     name=@
|         action      provider=@
|         event       provider=@
|         vartable    name=@
|         wkldapp     name=@
|         runcygrp    name=@
|         lob         name=@
|
|     end

```

Définition du rôle de sécurité

Dans un modèle de sécurité basée sur les rôles, un rôle de sécurité représente un certain niveau d'autorisation et inclut l'ensemble des actions que les utilisateurs ou les groupes peuvent exécuter. Vous pouvez inclure plusieurs définitions de rôles de sécurité dans le même fichier texte, avec des définitions de domaines de sécurité et des définitions de listes de contrôle d'accès.

Chaque définition de rôles de sécurité utilise le format et les arguments suivants :

Syntaxe

```

| securityrole nom_rôle_sécurité
|     [description "description"]
|         type_objet
| access[=action[,action]...]
|         [type_objet
| access[=action[,action]...]]...
|     end

```

[securitydomain ...]

[accesscontrollist ...]

Arguments

securityrole *securityrolename*

Spécifie le nom du rôle de sécurité. Ce nom doit commencer par une lettre et peut contenir des caractères alphanumériques, des tirets et des traits de soulignement. Il peut comporter jusqu'à 16 caractères.

description *"description"*

Donne une description du rôle de sécurité. La description peut contenir jusqu'à 120 caractères alphanumériques. Ce texte doit être délimité par des guillemets.

type_objet **access**[=action[,action]...]

Pour chaque type d'objet, spécifie une liste des actions que les utilisateurs ou les groupes peuvent exécuter sur ce type d'objet spécifique.

Le tableau 41, à la page 310 présente les différents types d'objet et la manière dont ils sont référencés avec **composer** et Dynamic Workload Console :

Tableau 41. Types d'objet de sécurité

Type d'objet - composer	Type d'objet - Dynamic Workload Console	Description
action	Actions	Actions définies dans les règles d'événement de planification
calendar	Agendas	Agendas de l'utilisateur
UC	Postes de travail	Postes de travail, domaines et classes de poste de travail
event	Événements	Conditions d'événement dans les règles d'événement de planification
eventrule	Règles d'événement	Définitions de règles d'événement de planification
fichier	Fichiers	Fichiers de base de données IBM Workload Scheduler
travail	Travaux	Travaux planifiés et définitions de travaux
lob	IBMApplication Lab	IBMApplication Lab
parameter	Paramètres	Paramètres locaux
prompt	Invites	Invites globales
rapport	Rapports	Les rapports suivants dans Dynamic Workload Console : RUNHIST Historique d'exécution du travail RUNSTATS Statistiques d'exécution du travail WWS Récapitulatif de la charge de travail du poste de travail WWR Temps d'exécution de la charge de travail du poste de travail SQL SQL personnalisé ACTPROD Détails de la production réelle (pour les plans en cours et archivés) PLAPROD Détails de la production planifiée (pour les plans d'essai et les plans prévisionnels)
Ressource	Ressources	Ressources de planification
runcygrp	Groupes de cycle d'exécution	Groupes de cycle d'exécution
schedule	Flots de travaux	Flots de travaux
userobj	Objets utilisateur	Objets utilisateur
table de variables	Tables de variables	Tables de variables
wkldappl	Application de charge de travail	Application de charge de travail

Le tableau 42 présente les actions que les utilisateurs ou les groupes peuvent exécuter sur différents objets.

Tableau 42. Actions que les utilisateurs ou les groupes peuvent exécuter sur les différents objets

Actions que les utilisateurs ou les groupes peuvent exécuter sur les différents objets			
add	deldep	manage	shutdown
adddep	delete	modify	start
altpass	display	release	stop
altpri	fence	reply	submit
build	kill	rerun	submitdb
cancel	limit	resetfta	unlink
confirm	link	Ressource	unlock
console	liste	run	use

Pour les actions que les utilisateurs ou les groupes peuvent exécuter sur un type d'objet spécifique, pour chaque tâche IBM Workload Scheduler, voir la section relative à la gestion des rôles de sécurité avec Dynamic Workload Console, dans *Dynamic Workload Console - Guide d'utilisation*.

Exemples

L'exemple suivant définit les rôles de sécurité SECROLE1 et SECROLE2 :

```
SECURITYROLE SECROLE1
DESCRIPTION "Sample Security Role"
SCHEDULE ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,MODIFY,
RELEASE
RESOURCE ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
PROMPT ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
FILE ACCESS=BUILD,DELETE,DISPLAY,MODIFY,UNLOCK
CPU ACCESS=LIMIT,LINK,MODIFY,SHUTDOWN,START,STOP,UNLINK,LIST,UNLOCK,RUN
PARAMETER ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
CALENDAR ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
REPORT ACCESS=DISPLAY
EVENTRULE ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
ACTION ACCESS=DISPLAY,SUBMIT,USE,LIST
EVENT ACCESS=USE
VARIABLE ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
WKLDAPPL ACCESS=ADD,DELETE,DISPLAY,MODIFY,LIST,UNLOCK
RUNCYGRP ACCESS=ADD,DELETE,DISPLAY,MODIFY,USE,LIST,UNLOCK
LOB ACCESS=USE
END

SECURITYROLE SECROLE2
DESCRIPTION "Sample Security Role"
SCHEDULE ACCESS=ADD,ADDDEP,ALTPRI,CANCEL,DELDEP,DELETE,DISPLAY,LIMIT,MODIFY,
RELEASE
RESOURCE ACCESS=ADD,DELETE,DISPLAY,MODIFY,RESOURCE,USE,LIST,UNLOCK
PROMPT ACCESS=ADD,DELETE,DISPLAY,MODIFY,REPLY,USE,LIST,UNLOCK
END
```

Actions sur les objets de sécurité

Les tableaux suivants affichent les actions que les utilisateurs ou les groupes peuvent réaliser sur les différents types d'objet pour chaque tâche IBM Workload Scheduler. Consultez les valeurs correspondantes des *actions* et *objets* entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande **composer**.

Tableau 43. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la conception et de la surveillance de la charge de travail

Concevoir et surveiller la charge de travail	
Actions que les utilisateurs ou les groupes peuvent réaliser	Types d'objet de sécurité
Lister (list)	Travaux (job)
Afficher (display)	Flots de travaux (schedule)
Créer (add)	Objets utilisateurs (userobj)
Supprimer (delete)	Invites (prompt)
Modifier (modify)	Ressources (resource)
Utiliser (use)	Agendas (calendar)
Déverrouiller (unlock)	Groupe de cycle d'exécution (runcygrp)
Actions sur des postes de travail à distance lors de la modélisation de travaux (cpu-run)	Tables de variables (vartable)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .	Application de charge de travail (wkldappl)
	Paramètres (parameter)

Tableau 44. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la modification du plan courant

Modifier le plan courant
Actions que les utilisateurs ou les groupes peuvent réaliser sur le plan courant
Ajouter une dépendance de flot de travaux (schedule - adddep)
Ajouter une dépendance de travail (job - adddep)
Supprimer une dépendance de travail (job - deldep)
Supprimer une dépendance de flot de travaux (schedule - deldep)
Modifier la priorité du travail (job - altpri)
Modifier la priorité du flot de travaux (schedule - altpri)
Annuler le travail (job - cancel)
Annuler le flot de travaux (schedule - cancel)
Réexécuter le travail (job - rerun)
Confirmer le travail (job - confirm)
Libérer le travail (job - release)
Libérer le flot de travaux (schedule - release)
Arrêter les travaux (job - kill)
Répondre aux invites (prompt - reply)
Répondre aux invites de travail (job - reply)
Répondre aux invites de flot de travaux (schedule - reply)
Modifier le mot de passe utilisateur (userobj - altpass)
Modifier la limite de travaux (schedule - limit)
Actions sur le système distant de travaux (job - run)
Modifier la quantité de ressources (resource - resource)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Tableau 45. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la soumission de la charge de travail

Soumettre la charge de travail
Définitions de charge de travail qui peuvent être ajoutées au plan courant
Seulement les définitions de travail existantes (job - submitdb)
Définitions de travaux et travaux ad hoc existants (job - submit)
Définitions de flot de travaux existant (schedule - submit)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Tableau 46. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la gestion de l'environnement de charge de travail

Gérer l'environnement de charge de travail
Actions que les utilisateurs ou les groupes peuvent réaliser sur les postes de travail, les domaines et les classes de poste de travail
Lister les postes de travail (cpu - list)
Afficher les détails des postes de travail (cpu - display)
Créer des postes de travail (cpu - add)
Supprimer des postes de travail (cpu - delete)
Modifier des postes de travail (cpu - modify)
Utiliser des postes de travail (cpu - use)
Déverrouiller des postes de travail (cpu - unlock)
Démarrer un poste de travail (cpu - start)
Arrêter un poste de travail (cpu - stop)
Modifier la limite (cpu - limit)
Modifier la priorité minimale (cpu - fence)
Arrêter (cpu - shutdown)
Réinitialiser l'agent tolérant aux pannes (cpu - resetfta)
Lier (cpu - link)
Supprimer le lien (cpu - unlink)
Utiliser la commande 'console' à partir de conman (cpu - console)
Mettre le poste de travail à niveau (cpu - manage)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Tableau 47. Actions que les utilisateurs ou les groupes peuvent réaliser lors de la gestion des règles d'événement

Gérer les règles d'événement
Actions que les utilisateurs ou les groupes peuvent réaliser sur les règles d'événement
Lister les règles d'événement (eventrule - list)
Afficher les détails des règles d'événement (eventrule - display)
Créer des règles d'événement (eventrule - add)
Supprimer des règles d'événement (eventrule - delete)
Modifier des règles d'événement (eventrule - modify)
Utiliser des règles d'événement (eventrule - use)
Déverrouiller des règles d'événement (eventrule - unlock)
Afficher des actions dans les règles d'événement (action - display)
Surveiller les actions déclenchées (action - list)
Utiliser des types d'action dans les règles d'événement (action - use)
Soumettre une action (action - submit)
Utiliser des événements dans les règles d'événement (event - use)
Utilisez un événement de surveillance de fichier sur le poste de travail où se trouve le fichier. (event - display)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Tableau 48. Tâches administratives que les utilisateurs ou les groupes peuvent réaliser

Tâches administratives
Tâches administratives que les utilisateurs ou les groupes peuvent réaliser
Afficher la configuration (sécurité et options globales de vidage) (file - display)
Modifier la configuration (makesec, optman add) (file - modify)
Supprimer des définitions d'objets (file - delete)
Déverrouiller les définitions d'objets (file - unlock)
Autoriser les commandes planman deploy, prodsked et stageman (file - build)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Tableau 49. Actions que les utilisateurs ou les groupes peuvent réaliser sur les rapports de charge de travail

Rapports de charge de travail	
Actions que les utilisateurs ou les groupes peuvent réaliser sur les rapports de charge de travail	
Générer des rapports de charge de travail (display report)	Rapports dans Dynamic Workload Console RUNHIST Historique d'exécution du travail RUNSTATS Statistiques d'exécution du travail WWS Récapitulatif de la charge de travail du poste de travail WWR Temps d'exécution de la charge de travail du poste de travail SQL SQL personnalisé ACTPROD Détails de la production réelle (pour les plans en cours et archivés) PLAPROD Détails de la production planifiée (pour les plans d'essai et les plans prévisionnels)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .	

Tableau 50. Actions réalisables par les utilisateurs ou les groupes sur Application Lab

Application Lab
Actions réalisables par les utilisateurs ou les groupes sur Application Lab
Accès à Application Lab (use lob)
Remarque : Consultez les valeurs correspondantes des <i>actions</i> et <i>objets</i> entre parenthèses que vous devez utiliser lors de la définition de la sécurité basée sur les rôles avec l'interface de ligne de commande composer .

Attributs des types d'objet de sécurité

Le tableau 51 présente les attributs que vous pouvez indiquer pour chaque type d'objet de sécurité (prenez connaissance du type d'objet et de l'attribut d'objet correspondants indiqués entre parenthèses et que vous devez utiliser lors de la définition d'objets de sécurité à l'aide de l'interface de ligne de commande **composer**).

Tableau 51. Attributs des types d'objet de sécurité

Attribut										
Type d'objet de sécurité	Nom (name)	Poste de travail (cpu)	Personnalisé (custom)	JCL (jcl)	Type de JCL (jcltype)	Connexion (logon)	Fournisseur (provider)	Type (type)	Hôte (host)	Port (port)
Actions (action)							✓	✓	✓	✓
Agendas (calendar)	✓									
Postes de travail (cpu)								✓		
Événements (event)			✓				✓	✓		
Règles d'événement (event)	✓									
Fichiers (file)	✓									
Travaux (job)	✓	✓		✓	✓	✓				
Application Lab (lob)	✓									
Paramètres (parameter)	✓	✓								

Tableau 51. Attributs des types d'objet de sécurité (suite)

Attribut										
Type d'objet de sécurité	Nom (name)	Poste de travail (cpu)	Personnalisé (custom)	JCL (jcl)	Type de JCL (jcltype)	Connexion (logon)	Fournisseur (provider)	Type (type)	Hôte (host)	Port (port)
Invites (prompt)	✓									
Rapports (report)	✓									
Ressource (resource)	✓	✓								
Groupes de cycle d'exécution (runcygrp)	✓									
Flots de travaux (schedule)	✓	✓								
Objets utilisateurs (userobj)		✓				✓				
Tables de variables (vartable)	✓									
Applications de charge de travail (wkldappl)	✓									

Pour connaître les valeurs admises pour chaque attribut d'objet, voir «Spécification des valeurs d'attribut des objets».

Spécification des valeurs d'attribut des objets

Les valeurs suivantes sont autorisées pour chaque attribut d'objet (prenez connaissance du type d'objet et de l'attribut d'objet correspondants indiqués entre parenthèses pour l'interface de ligne de commande **composer**) :

Nom (name)

Spécifie un ou plusieurs noms pour le type d'objet.

- Pour le type d'objet **Fichiers (file)**, les valeurs suivantes s'appliquent :
globalopts

Permet à l'utilisateur de définir les options globales avec la commande `optman`. Les types d'accès suivants sont autorisés :

- Accès en consultation pour `optman ls` et `optman show`
- Accès en modification pour `optman chg`

prodsked

Permet à l'utilisateur de créer, d'étendre ou de réinitialiser le plan de production.

security

Permet à l'utilisateur de gérer le fichier de sécurité.

Symphony

Permet à l'utilisateur d'exécuter **stageman** et **JnextPlan**.

trialsked

Permet à l'utilisateur de créer des plans d'essai et des plans prévisionnels ou d'étendre des plans d'essai.

Remarque : Les utilisateurs disposant d'un accès restreint aux fichiers doivent disposer au moins du droit d'accès suivant pour pouvoir afficher d'autres types d'objet, à savoir Agendas (calendar) et Postes de travail (cpu) :

```
file name=globalopts action=display
```

- Pour le type d'objet **Tables de variables (vartable)**, vous pouvez utiliser la valeur `$DEFAULT` pour l'attribut **Nom (name)** afin d'indiquer la table de variables par défaut. Cela permet de sélectionner la table définie avec l'attribut `isdefault`.

Poste de travail (cpu)

Spécifie un ou plusieurs noms de poste de travail, de domaine ou de classe

de poste de travail. Si vous omettez cet attribut, tous les postes de travail et les domaines définis restent accessibles. Des variables de poste de travail peuvent être utilisées :

\$MASTER

De IBM Workload Scheduler gestionnaire de domaine maître.

\$SLAVES

Tout agent tolérant aux pannes.

\$REMOTES

Tout agent standard.

\$THISCPU

Poste de travail sur lequel l'utilisateur exécute la commande ou le programme IBM Workload Scheduler

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

cpu=workstation[,workstation]...

Personnalisé (custom)

Utilisez cet attribut pour attribuer des droits d'accès aux événements définis dans les plug-in d'événement. La syntaxe précise de la valeur dépend du plug-in. Par exemple :

- Spécifiez différents droits pour différents utilisateurs en fonction de noms d'événement SAP R/3 lors de la définition de règle d'événement pour les événements SAP R/3.
- Définissez votre propre attribut de sécurité pour vos fournisseurs d'événements personnalisés.
- Spécifiez le type d'événement à surveiller. Chaque événement peut se rapporter à un fournisseur d'événements.

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

custom=value[,value]...

JCL (jcl)

Spécifie la commande ou le nom du chemin du fichier exécutable d'un objet travail. Si vous omettez cet attribut, tous les fichiers de travail et toutes les commandes définis sont éligibles.

Vous pouvez également spécifier une chaîne contenue dans la chaîne de tâches d'une définition JSDL à utiliser pour la mise en correspondance de modèle.

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

jcl="path" | "command" | "jSDL"

JCL Type (jcltype)

Indique que l'utilisateur est autorisé à agir sur les définitions de travaux exécutant uniquement des scripts (si défini avec **scriptname**) ou des commandes (si défini avec **docommand**). Utilisez cet attribut facultatif pour limiter les autorisations utilisateur sur les actions portant sur les définitions de travaux d'un seul type. Les actions sont octroyées pour les scripts et les commandes lorsque le paramètre **Type de JCL (jcltype)** est manquant.

Un utilisateur auquel serait refusée l'autorisation de travailler sur les définitions de travaux exécutant une commande ou un script obtient un message d'erreur de sécurité s'il tente d'exécuter une action sur ces définitions.

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

jc1type=[scriptname | docommand]

Connexion (logon)

Spécifie les ID utilisateur. Si vous omettez cette option, tous les ID utilisateur sont qualifiés.

Vous pouvez utiliser les valeurs suivantes pour l'attribut **Connexion (logon)** afin d'indiquer la connexion par défaut :

\$USER

Streamlogon est l'utilisateur conman/composer.

\$OWNER

Streamlogon est le créateur du travail.

\$JCOWNER

Streamlogon est le propriétaire du système d'exploitation du fichier.

\$JCLGROUP

Streamlogon est le groupe du système d'exploitation du fichier.

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

logon=username[,username]...

Fournisseur (provider)

Pour les types d'objet **Actions (action)**, spécifie le nom du fournisseur d'action.

Pour les types d'objet **Événements (event)**, indique le nom du fournisseur d'événements.

Si **Fournisseur (provider)** n'est pas spécifié, aucun objet défini n'est accessible.

Si vous utilisez la ligne de commande **composer** pour définir des domaines de sécurité, la syntaxe suivante s'applique :

provider=provider_name[,provider_name]...

Type (type)

Pour les types d'objet **Actions (action)**, il s'agit du type d'action `actionType`.

Pour les types d'objet **Événements (event)**, il s'agit du type d'action `eventType`.

Pour des types d'objet **Postes de travail (cpu)**, les valeurs admises sont celles utilisées dans **composer** ou Dynamic Workload Console lors de la définition des postes de travail (manager, broker, fta, agent, s-agent, x-agent, rem-eng, pool et d-pool, par exemple).

Remarque : La valeur de master, utilisée dans **conman**, est mappée aux attributs de sécurité manager.

| Si **Type (type)** n'est pas indiqué, tous les objets définis sont accessibles
| pour les fournisseurs spécifiés (c'est toujours le cas après une installation
| ou une mise à niveau, car l'attribut de type n'est pas fourni par défaut).

| Si vous utilisez la ligne de commande **composer** pour définir des domaines
| de sécurité, la syntaxe suivante s'applique :

| **type=type[, type]...**

| **Hôte (host)**

| Pour les types d'objet **Actions (action)**, spécifie le nom d'hôte TEC ou
| SNMP (utilisé pour certains types d'actions tels que l'envoi d'événements
| TEC ou l'envoi SNMP). Si cela ne s'applique pas, cette zone doit être vide.

| Si vous utilisez la ligne de commande **composer** pour définir des domaines
| de sécurité, la syntaxe suivante s'applique :

| **host=host_name**

| **Port (port)**

| Pour les types d'objet **Actions (action)**, spécifie le numéro de port TEC ou
| SNMP (utilisé pour certains types d'actions tels que l'envoi d'événements
| TEC ou l'envoi SNMP). Si cela ne s'applique pas, cette zone doit être vide.

| Si vous utilisez la ligne de commande **composer** pour définir des domaines
| de sécurité, la syntaxe suivante s'applique :

| **port=port_number**

Chapitre 10. Gestion des objets dans la base de données - composer

La présente section décrit comment utiliser le programme de ligne de commande **composer** pour gérer les objets de planification dans la base de données IBM Workload Scheduler. Il comprend les sections suivantes :

- «Configuration du programme de ligne de commande composer»
- «Exécution de commandes depuis composer», à la page 327
- «Commandes Composer», à la page 331

Configuration du programme de ligne de commande composer

Le programme de ligne de commande **composer** gère les objets de planification dans la base de données.

Le programme de ligne de commande **composer** gère les objets de planification.

Vous devez installer la fonction de client de ligne de commande de IBM Workload Scheduler sur des agents et sur des systèmes tolérants aux pannes à l'extérieur du réseau IBM Workload Scheduler pour pouvoir utiliser le programme de ligne de commande **composer**.

Les utilisateurs peuvent décider de gérer une trace d'audit qui consigne tous les changements qu'ils effectuent et les justifications associées. Pour activer l'option de justification, configurez les variables d'environnement IBM Workload Scheduler ci-dessous dans un shell système avant d'exécuter des commandes **composer** :

IWS_DESCRIPTION

Indiquez la description à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 512 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_CATEGORY

Indiquez la catégorie à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_TICKET

Indiquez le ticket à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

Pour plus d'informations sur l'option de justification, voir la section décrivant le suivi des changements dans le document *IBM Dynamic Workload Console - Guide d'utilisation*.

Remarque : Sur les systèmes AIX, si vous prévoyez de définir un grand nombre d'objets, le segment de données doit être augmenté pour assurer la fiabilité du produit.

Configuration de l'environnement composer

La présente section décrit la configuration de votre environnement composer.

Sortie sur le terminal

Les variables de shell nommées *MAESTROLINES* et *MAESTROCOLUMNS* définissent la sortie de votre ordinateur. Les variables sont définies dans le script *tw5_env* que vous lancez avant d'exécuter les commandes de ligne de commande. Si l'une ou l'autre de ces variables n'est pas définie, les variables de shell standard *LINES* et *COLUMNS* sont utilisées. A la fin de chaque page d'écran, le programme composer ne fait pas de pause à la fin d'une page. Si *MAESTROLINES* (ou *LINES*) est défini sur une valeur positive, le programme composer vous invite à continuer.

Selon la valeur attribuée à la variable locale *MAESTROCOLUMNS*, deux ensembles d'informations s'affichent concernant l'objet sélectionné. Il existe deux possibilités :

- *MAESTROCOLUMNS* < 120 caractères
- *MAESTROCOLUMNS* >= 120 caractères

La valeur définie dans la variable locale *MAESTROCOLUMNS* ne peut pas être supérieure à 1024.

Voir tableau 58, à la page 349 et tableau 59, à la page 362 pour en savoir plus sur les différents formats de sortie.

Sortie offline

Dans les commandes du programme composer, l'option **;offline** permet d'imprimer la sortie d'une commande. Si vous la définissez, les variables suivantes contrôlent la sortie :

Variables Windows :

MAESTROLP

Indique le fichier dans lequel la sortie d'une commande est écrite. La valeur par défaut est **stdout**.

MAESTROLPLINES

Indique le nombre de lignes par page. La valeur par défaut est 60.

MAESTROLPCOLUMNS

Indique le nombre de caractères par ligne. La valeur par défaut est 132.

Variables UNIX :

Dans les commandes du programme composer, l'option **;offline** permet d'imprimer la sortie d'une commande. Lorsqu'elle est insérée, la sortie est contrôlée par les variables de shell ci-après :

MAESTROLP

Indique la destination de la sortie d'une commande. Vous pouvez indiquer :

> *fichier*

La sortie est redirigée vers un fichier dont le contenu est remplacé. Si le fichier n'existe pas, il est créé.

>> *fichier*

La sortie est redirigée vers un fichier et ajoutée à la fin de celui-ci. Si le fichier n'existe pas, il est créé.

| *commande*

La sortie est dirigée vers une commande système ou un processus.
Le programme exécute la commande système, que la sortie soit
générée ou non.

|| *commande*

La sortie est dirigée vers une commande système ou un processus.
La commande système n'est pas exécutée en l'absence de résultat.

La valeur par défaut de **MAESTROL** est | **lp -tCONLIST** qui dirige la
sortie de la commande vers l'imprimante et insère le titre "CONLIST" sur
la page bannière de la sortie imprimée.

MAESTROLPLINES

Indique le nombre de lignes par page. La valeur par défaut est **60**.

MAESTROLPCOLUMNS

Indique le nombre de caractères par ligne. La valeur par défaut est **132**.

Vous devez exporter les variables avant d'exécuter le programme **composer**.

Editeur Composer

Plusieurs commandes du programme **composer** permettent d'ouvrir
automatiquement un éditeur de texte. Vous pouvez sélectionner l'éditeur que
composer doit utiliser.

De plus, aussi bien sous Windows que sous UNIX, vous pouvez définir la variable
d'environnement **XMLEDIT** pour pointer vers un éditeur XML de votre choix afin
d'éditer les définitions de règles d'événement. L'éditeur XML s'ouvre
automatiquement à chaque exécution des commandes **composer add**, **new** ou **modify**
sur une règle d'événement.

Windows :

Sous Windows, **Notepad** est l'éditeur utilisé par défaut. Pour changer d'éditeur,
définissez la variable d'environnement **EDITOR** avec le chemin d'accès et le nom
du nouvel éditeur avant d'exécuter **composer**.

UNIX :

Plusieurs des commandes que vous pouvez émettre à partir de **composer** ouvrent
automatiquement un éditeur de texte. Le type d'éditeur utilisé est déterminé par la
valeur de deux variables du shell. Si la variable **VISUAL** est définie, elle définit
l'éditeur ; sinon, c'est la variable **EDITOR** qui définit l'éditeur. Si aucune de ces
variables n'est définie, l'éditeur **vi** apparaît.

Sélection de l'invite de commande sous UNIX

L'invite de commande **composer** est définie dans le fichier *racine_TWS/locopts*.
L'invite de commande par défaut correspond à un tiret (-). Pour sélectionner une
autre invite, éditez l'option **composer prompt** dans le fichier *locopts* et modifiez
le tiret. L'invite peut comporter jusqu'à dix caractères, le signe dièse (#) de fin
requis n'étant pas inclus :

```
#-----  
# Custom format attributes  
#  
date format =          1      # The possible values are 0-ymd, 1-mdy,  
2-dmy, 3-NLS.  
composer prompt =      -  
conman prompt =        %  
switch sym prompt =    <n>%  
#-----
```

Pour plus d'informations sur le fichier de configuration `localopts`, voir *IBM Workload Scheduler - Guide d'administration*.

Exécution du programme Composer

Pour configurer l'environnement afin d'utiliser **composer**, définissez les variables `PATH` et `TISDIR_TWS` en exécutant l'un des scripts suivants :

Sous UNIX :

- `./inst_TWS/TWS/tws_env.sh` pour les interpréteurs de commandes Bourne et Korn
- `./inst_TWS/TWS/tws_env.csh` pour les interpréteurs de commandes C

Sous Windows :

- `inst_TWS\TWS\tws_env.cmd`

Utilisez ensuite la syntaxe suivante pour exécuter les commandes à partir de l'interface utilisateur **composer** :

composer [-file *nom_fichier*][*paramètres_connexion*] ["commande[&[commande]][...]"]

où :

-file *nom_fichier*

Indique un autre fichier de propriétés personnalisé contenant la configuration des paramètres de connexion, utilisé à la place des fichiers **useropts** et **localopts**.

paramètres_connexion

Si vous utilisez **composer** à partir du gestionnaire de domaine maître, les paramètres de connexion ont été configurés lors de l'installation et n'ont pas besoin d'être fournis, sauf si vous ne voulez pas utiliser les valeurs par défaut.

Si vous utilisez **composer** à partir du client de ligne de commande sur un autre poste de travail, les paramètres de connexion peuvent être fournis par l'une ou plusieurs de ces méthodes :

- Ils sont stockés dans le fichier `localopts`
- Ils sont stockés dans le fichier `useropts`
- Ils sont fournis à la commande dans un fichier de paramètres
- Ils sont fournis à la commande au sein d'une chaîne de commande

Pour une présentation de ces options, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65. Pour des détails complets sur les paramètres de configuration, voir la rubrique relative à la configuration de l'accès du client de ligne de commande dans *IBM Workload Scheduler - Guide d'administration*.

Remarque : Si vous utilisez **composer** à partir du client de ligne de commande sur un autre poste de travail pour le sous-ensembles d'objets de planification suivant :

- jobs
- flots de travaux
- groupes de cycle d'exécution
- applications de charge de travail
- les listes de contrôle d'accès

- les domaines de sécurité
- les rôles de sécurité

la ligne de commande **composer** se connecte au serveur à l'aide d'une connexion TTPS. Dans ce cas, le client de ligne de commande assemble l'ensemble des paramètres de connexion dans l'ordre suivant :

1. Paramètres spécifiés dans la chaîne de commande
2. Paramètres spécifiés dans le fichier des propriétés personnalisées
3. Paramètres spécifiés dans le fichier useropts
4. Paramètres spécifiés dans le fichier localopts
5. Paramètres spécifiés dans le fichier jobmanager.ini

Les valeurs possibles sont les suivantes :

[-username <nom_utilisateur>]

Utilisateur IBM Workload Scheduler doté des droits suffisants pour effectuer l'opération.

[-password <mot_de_passe>]

Mot de passe de l'utilisateur IBM Workload Scheduler.

[-host <nom_hôte>]

Nom de l'hôte auquel vous souhaitez accéder à l'aide de la ligne de commande wappman.

[-port <numéro_port>]

Numéro de port TCP/IP utilisé pour se connecter à l'hôte indiqué.

[-protocol {http | https}]

Protocole utilisé pour se connecter à l'hôte indiqué.

[-file <fichier_propriétés_personnalisées>]

Fichier de propriétés personnalisées dans lequel vous pouvez spécifier des paramètres de connexion qui prévalent sur les valeurs indiquées dans les fichiers useropts, localopts et jobmanager.ini. Les paramètres de connexion spécifiés dans le fichier de propriétés personnalisé doivent avoir la syntaxe suivante :

```
HOST=<nom_hôte>
PORT=<port>
PROTOCOL=<http/https>
USERNAME=<nom_utilisateur>
PASSWORD=<mot_de_passe>
```

Si les paramètres host, port et protocol sont spécifiés dans un fichier, ils doivent tous l'être dans le même fichier.

Le programme de ligne de commande **composer** est automatiquement installé lors de l'installation du gestionnaire de domaine maître. Il doit être installé séparément sur un poste de travail d'agent IBM Workload Scheduler ou de façon autonome sur un noeud en dehors du réseau IBM Workload Scheduler. La fonction qui installe le programme de ligne de commande **composer** est nommée *client de ligne de commande*. Pour plus d'informations sur l'installation de la fonction *Command Line Client*, voir *IBM Workload Scheduler - Guide de planification et d'installation*.

Vous pouvez utiliser la ligne de commande **composer** en mode de *traitement par lots* et en mode *interactif*.

Lorsque vous exécutez **composer** en mode *interactif*, vous lancez d'abord le programme de ligne de commande **composer**, puis vous exécutez les commandes l'une après l'autre à partir de l'invite de ligne de commande **composer**, par exemple :

```
composer -username admin2 -password admin2pwd
add myjobs.txt
create myjobs.txt from jobs=@
```

Lorsque vous exécutez **composer** en mode de *traitement par lots*, vous lancez le programme de ligne de commande **composer** en spécifiant comme paramètre d'entrée la commande à émettre. Lorsqu'elle est exécutée, le programme de ligne de commande **composer** s'arrête comme dans l'exemple suivant :

```
composer -f "c:\TWS\network\mylocalopts" add myjobs.txt
```

Remarque : Si vous utilisez le mode de traitement par lots pour exécuter plusieurs commandes à partir de **composer**, veillez à gérer le point-virgule (;) de l'une des façons suivantes :

- Utilisation des guillemets, par exemple :

```
composer "delete dom=ancien_domaine; noask"
```
- Utilisation d'un espace, par exemple :

```
composer delete dom=ancien_domaine noask
```
- Utilisation d'une barre oblique inversée (\), puis d'un point-virgule (;) par exemple :

```
composer delete dom=ancien_domaine \; noask
```

Voici d'autres exemples de l'utilisation de la commande avec les paramètres de connexion définis dans les scripts de la configuration locale :

- Exécute les commandes **print** et **version** avant de quitter :

```
composer "p parms&v"
```
- Exécute les commandes **print** et **version** avant d'inviter l'utilisateur à taper une commande :

```
composer "p parms&v&"
```
- Lit les commandes à partir de cmdfile :

```
composer < cmdfile
```
- Transmet les commandes de cmdfile au programme **composer**:

```
cat cmdfile | composer
```

Remarque : Sur les postes de travail Windows, si le contrôle de compte utilisateur (UAC) est activé et que la liste d'exceptions UAC ne contient pas le fichier cmd.exe, vous devez ouvrir l'interpréteur de commandes DOS avec l'option "Exécuter en tant qu'administrateur" pour exécuter **composer** sur votre poste de travail en tant qu'utilisateur générique différent de l'administrateur ou utilisateur IBM Workload Scheduler.

Caractères de commande

Vous pouvez taper les caractères de commande suivants en mode conversationnel pour interrompre le programme **Composer**, si les paramètres *stty* sont configurés à cette fin.

Ctrl+c Le programme **composer** interrompt l'exécution de la commande dès que l'étape courante le permet et renvoie une invite de commande.

Ctrl+d Le programme **composer** s'arrête après avoir exécuté la commande courante.

Exécution de commandes depuis composer

Les commandes du programme **composer** se décomposent de la manière suivante :

nom_commande sélection arguments

où :

nom_commande

Indique le nom de la commande.

sélection

Indique l'objet ou le groupe d'objets sur lequel sera exécutée la commande.

arguments

Indique les arguments de la commande.

Filtres et caractères génériques

Dans le programme **composer** de IBM Workload Scheduler, vous pouvez utiliser des caractères génériques et des filtres lorsque vous exécutez certaines commandes spécifiques afin de filtrer les objets de planification définis dans la base de données. vous pouvez utiliser les caractères génériques suivants à partir de **composer** :

@ Remplace un ou plusieurs caractères alphanumériques.

? Remplace un caractère alphanumérique.

Pour rechercher les occurrences de noms contenant soit @ soit ?, utilisez la barre oblique inversée \ devant @ ou ? de façon à ce qu'ils ne soient pas interprétés comme des caractères génériques. De la même façon, la barre oblique inversée doit être précédée d'une autre barre oblique inversée pour être interprétée comme une occurrence à trouver. Les exemples suivantes expliquent ces règles qui s'appliquent également lorsque vous spécifiez des chaînes de recherche à l'aide du mot clé **filter**.

S@E Recherche de toutes les chaînes commençant par S et se terminant par E, quelle que soit leur longueur.

S?E Recherche de toutes les chaînes commençant par S, se terminant par E et dont la longueur est de trois caractères.

S\\@E Recherche de la chaîne S@E.

S\\?E Recherche de la chaîne S?E.

S\\E Recherche de la chaîne S\E.

Les commandes que vous pouvez exécuter à partir de composer et qui prennent en charge le filtrage sont :

- display
- create
- delete
- list
- lock
- modify
- print

- unlock
- Mise à jour

La syntaxe utilisée pour filtrer les objets lorsque vous émettez l'une de ces commandes est la suivante :

nom_commande *type_objet*=selection; [option;] [**filter** *mot_clé_filtre*=selection [...]]

Le tableau 52 montre les objets de planification que vous pouvez filtrer lorsque vous émettez les commandes répertoriées ci-dessus et, pour chaque objet, les zones qui peuvent être filtrées (en *italique*) ou la clé (en **gras**) utilisée pour filtrer les zones :

Tableau 52. Critères de filtrage des objets de planification

Objet de planification	Mots clés de filtre ou zones qui peuvent être filtrées	Description	Exemple
poste de travail	<i>nom_poste_travail</i>	Applique la commande aux postes de travail dont le nom respecte les critères.	list ws=p@
	domaine	Applique la commande aux postes de travail qui appartiennent à un domaine.	mod ws=@; filter domain =dom1
	table de variables	Applique la commande aux postes de travail qui font référence à la table de variables spécifiée.	mod ws=@; filter variable =table2
domaine	<i>nom_domaine</i>	Applique la commande aux domaines dont le nom respecte les critères.	display dom=dom?
	parent	Applique la commande aux domaines dont le domaine parent respecte les critères.	list dom=@; filter parent =rome
prompt	<i>nom_invite</i>	Applique la commande aux invites globales dont le nom respecte les critères.	lock prompt=p@
user	<i>nom_poste_travail# nom_utilisateur</i>	Applique la commande aux utilisateurs dont l'identificateur respecte les critères.	list users=cpu1#operator?
resource	<i>nom_poste_travail# nom_ressource</i>	Applique la commande aux ressources dont l'identificateur respecte les critères.	print res=cpu?#operator?
variable	<i>nom_variable</i>	Applique la commande aux paramètres dont le nom respecte les critères.	delete vb=mytable.myparm@

Tableau 52. Critères de filtrage des objets de planification (suite)

Objet de planification	Mots clés de filtre ou zones qui peuvent être filtrées	Description	Exemple
définition de travail	<i>nomtravail</i>	Applique la commande aux définitions de travail dont le nom respecte les critères.	mod jd=mycpu#myjob@
	RecoveryJob	Applique la commande aux travaux dont la définition contient la définition de travail de reprise spécifiée.	list jobdefinition=@; filter RecoveryJob =CPUA#job01
flot de travaux	<i>nom_poste_travail# nom_flot_travaux</i>	Applique la commande aux définitions de flot de travaux dont le nom respecte les critères.	mod js=mycpu#myjs@
	Agenda	Applique la commande aux flots de travaux contenant l'agenda spécifié dans le filtre.	list js=@#@; filter Calendar =cal1
	Jobdefinition	Applique la commande aux flots de travaux contenant la définition de travail spécifiée dans le filtre.	list js=@#@; filter jobdefinition =CPUA#job01
	Ressource	Applique la commande aux flots de travaux qui font référence à la ressource spécifiée dans le filtre.	list js=@#@; filter Resource =cpu1#disk1
	Prompt	Applique la commande aux flots de travaux qui font référence à l'invite spécifiée dans le filtre.	list js=@#@; filter Prompt =myprompt
	Vartable	Applique la commande aux flots de travaux qui font référence à la table de variables spécifiée dans le filtre. La table de variables peut être spécifiée dans la section du cycle d'exécution ou du le flot de travaux.	list js=@#@; filter Vartable =table1
	Rcvartable	Applique la commande aux cycles d'exécution des flots de travaux qui font référence à la table de variables spécifiée dans le filtre.	list js=@#@; filter Rcvartable =table1
	Jsvartable	Applique la commande aux flots de travaux qui font référence à la table de variables spécifiée dans le filtre, quels que soient les éléments spécifiés dans le cycle d'exécution.	list js=@#@; filter Jsvartable =table1

Tableau 52. Critères de filtrage des objets de planification (suite)

Objet de planification	Mots clés de filtre ou zones qui peuvent être filtrées	Description	Exemple
règle d'événement	<i>nom_règle_évenmt</i>	Applique la commande aux règles d'événement comprenant une action appliquée à un travail ou flot de travaux spécifique.	list er=@; filter js=accrecjs5
table de variables	<i>nom_table_var</i>	Applique la commande aux tables de variables dont le nom satisfait les critères.	list vartable=A@
	isdefault	Applique la commande à la table de variables par défaut.	list vartable=A@; filter isdefault

Vous pouvez associer plusieurs filtres pour le même type d'objet (voir exemple suivant) :

```
list js=@#@; filter Calendar=call jobdefinition=CPUA#job01
```

La commande génère une liste des flots de travaux utilisant l'agenda call et contenant un travail avec la définition CPUA#job01.

Délimiteurs et caractères spéciaux

Le tableau 53 répertorie les caractères qui ont une signification particulière dans les commandes composer.

Tableau 53. Délimiteurs et caractères spéciaux pour le programme composer

Caractère	Description
&	Délimiteur de commande. Voir «Exécution du programme Composer», à la page 324.
;	Délimiteur d'argument. Par exemple : ;info;offline
=	Délimiteur de valeur. Par exemple : sched=sked5
: !	Préfixes de commande qui transmettent la commande au système. Ces préfixes sont facultatifs. Si le programme composer ne reconnaît pas une commande, elle est transmise automatiquement au système. Par exemple : !ls ou :ls
<< >>	Crochets de mise en commentaire. Des commentaires peuvent figurer sur une seule ligne n'importe où dans un flot de travaux. Par exemple : schedule foo <<commentaire>> on everyday
*	Préfixe de commentaire. Lorsque ce préfixe correspond au premier caractère d'une ligne, la ligne entière correspond à un commentaire. Lorsque ce préfixe suit une commande, seul le reste de la ligne correspond au commentaire. Par exemple : *commentaire ou print& *commentaire

Tableau 53. Délimiteurs et caractères spéciaux pour le programme composer (suite)

Caractère	Description
>	La sortie de la commande est redirigée vers un fichier dont le contenu est remplacé. Si le fichier n'existe pas, il est créé. Par exemple : display parms > parmlist
>>	Dirige la sortie de la commande vers un fichier et celle-ci est ajoutée à la fin du fichier. Si le fichier n'existe pas, il est créé. Par exemple : display parms >> parmlist
	La sortie de la commande est dirigée vers une commande système ou un processus. Le programme exécute la commande système, que la sortie soit générée ou non. Par exemple : display parms grep alparm
	La sortie de la commande est dirigée vers une commande système ou un processus. La commande système n'est pas exécutée en l'absence de résultat. Par exemple : display parms grep alparm

Codes retour de composer

Codes retour de composer

Gestion des codes retour de composer

Lorsque vous exécutez une commande composer, la ligne de commande peut afficher un code retour de résultat. Pour trouver le code retour, procédez comme suit :

Sur les systèmes d'exploitation Windows :

```
echo %ERRORLEVEL%
```

Sur les systèmes d'exploitation UNIX :

```
echo $?
```

La ligne de commande Composer renvoie les codes retour suivants :

- 0 Commande terminée.
- 4 La commande a été exécutée avec un avertissement.
- 8 La commande a été exécutée avec une erreur.
- 16 La commande échoue.
- 32 La commande comporte une erreur de syntaxe.

Commandes Composer

Le tableau 54, à la page 332 répertorie les commandes **composer**.

Remarque : Les noms de commande et les mots clés peuvent être entrés en majuscules ou en minuscules et peuvent être abrégés au nombre minimal de caractères suffisant pour les distinguer les uns des autres de façon unique. Certains noms de commande ont également une forme abrégée.

Cependant, certaines abréviations (par exemple **v** pour **version**) désignent une commande spécifique, même si elles n'identifient pas uniquement la commande

dans la liste. Ce cas se produit lorsque l'abréviation est incorporée dans le produit, ce qui permet d'éviter l'appel d'une la commande incorrecte pouvant générer des problèmes de non-concordance.

Tableau 54. Liste des commandes composer

Commande	Nom abrégé	Description	Voir page
add	a	Ajoute une définition d'objets de planification à la base de données à partir d'un fichier texte.	«add», à la page 338
authenticate	au	Modifie les droits d'accès de l'utilisateur exécutant composer .	«authenticate», à la page 340
continue	c	Ignore l'erreur suivante.	«continue», à la page 340
create	cr	Extrait une définition d'objet à partir de la base de données et l'écrit dans un fichier texte. Synonyme de la commande extract .	«extract», à la page 352
delete	de	Supprime des objets de planification.	«delete», à la page 341
display	di	Affiche les détails de l'objet de planification spécifié.	«display», à la page 345
edit	ed	Permet d'éditer un fichier.	«edit», à la page 351
exit	e	Quitte le programme composer .	«exit», à la page 352
extract	ext	Extrait une définition d'objet à partir de la base de données et l'écrit dans un fichier texte.	«extract», à la page 352
help	h	Appelle l'aide en ligne d'une commande.	«help», à la page 357
list	l	Répertorie les objets de planification.	«list», à la page 358
lock	lo	Verrouille l'accès aux objets de la base de données.	«lock», à la page 365
modify	m	Modifie les objets de planification.	«modify», à la page 370
new		Crée un objet de planification à l'aide d'un fichier texte dans lequel la définition d'objet est insérée en ligne. Si vous indiquez le type d'objet de planification que vous souhaitez définir après la commande <i>new</i> , une définition d'objet prédéfinie est écrite dans le fichier texte.	«new», à la page 375
print	p	Imprime des objets de planification.	«display», à la page 345
redo	red	Modifie et exécute à nouveau la commande précédente.	«redo», à la page 377
rename	rn	Modifie le nom d'objet.	«rename», à la page 379
replace	rep	Permet de remplacer des objets de planification.	«replace», à la page 381

Tableau 54. Liste des commandes composer (suite)

Commande	Nom abrégé	Description	Voir page
<i>commande système</i>		Appelle une commande de système d'exploitation.	«Commande système», à la page 383
unlock	u	Déverrouille l'objet de planification défini dans la base de données.	«unlock», à la page 383
update	up	Met à jour les paramètres d'attribut de l'objet de planification dans la base de données.	«update», à la page 388
validate	val	Valide la syntaxe, la sémantique et l'intégrité des données d'une définition d'objet.	«validate», à la page 390
version	v	Affiche la bannière du programme de ligne de commande composer .	«version», à la page 391

Vérification de l'intégrité référentielle

IBM Workload Scheduler effectue automatiquement des vérifications référentielles pour préserver l'intégrité des définitions d'objet dans la base de données lorsque vous exécutez des commandes qui créent, modifient ou suppriment la définition d'un objet référencé. Le produit effectue les vérifications suivantes :

- Chaque fois que vous utilisez une commande qui crée un nouvel objet dans la base de données, IBM Workload Scheduler vérifie les points suivants :
 - Aucun objet du même type et avec le même identificateur n'existe déjà.
 - Les objets référencés par cet objet existent déjà dans la base de données.
- Chaque fois que vous exécutez une commande qui modifie une définition d'objet dans la base de données, IBM Workload Scheduler vérifie les points suivants :
 - La définition d'objet à modifier existe dans la base de données.
 - Les objets référencés par cet objet existent dans la base de données.
 - Pour éviter les incohérences dans l'intégrité, la définition d'objet ne s'affiche pas dans la définition d'un objet appartenant à la chaîne de ses ancêtres.
- Chaque fois que vous exécutez une commande qui supprime une définition d'objet dans la base de données, IBM Workload Scheduler vérifie les points suivants :
 - La définition d'objet à supprimer existe dans la base de données.
 - La définition d'objet à supprimer n'est pas référencée par d'autres objets définis dans la base de données.

Il est à noter que les règles d'événement ne font l'objet d'aucun contrôle d'intégrité référentielle.

Le tableau 55 montre, pour chaque type d'objet, les identificateurs utilisés pour identifier de manière unique l'objet dans la base de données lors de la création ou de la modification des définitions d'objet :

Tableau 55. Identificateurs d'objet pour chaque type d'objet défini dans la base de données

Type d'objet	Identificateurs d'objet
domaine	<i>nom_domaine</i>
Poste de travail	<i>poste_de_travail</i> (vérifié sur les postes de travail et les classes de poste de travail)

Tableau 55. Identificateurs d'objet pour chaque type d'objet défini dans la base de données (suite)

Type d'objet	Identificateurs d'objet
classe de poste de travail	<i>nom_classe_poste_de_travail</i> (vérifié sur les postes de travail et les classes de postes de travail)
calendar	<i>nom_agenda</i>
définition de travail	<i>nom_poste_travail</i> et <i>nom_travail</i>
utilisateur	<i>nom_poste_travail</i> et <i>nom_utilisateur</i>
flot de travaux	<i>nom_poste_de_travail</i> et <i>nom_flot_travaux</i> et <i>début_validité</i> si ce paramètre est défini
travail dans un flot de travaux	<i>nom_poste_de_travail</i> , <i>nom_flot_travaux</i> , <i>nom_travail</i> et <i>début_validité</i> si ce paramètre est défini
ressource	<i>nom_poste_travail</i> et <i>nom_ressource</i>
invite	<i>nom_invite</i>
table de variables	<i>nom_table_variables</i>
variable	<i>nom_table_variables.nom_variable</i>
règle d'événement	<i>nom_règle_événement</i>
liste de contrôle d'accès	<i>nom_domaine_sécurité</i>
domaine de sécurité	<i>nom_domaine_sécurité</i>
rôle de sécurité	<i>nom_rôle_sécurité</i>

En règle générale, l'intégrité référentielle empêche la suppression des objets lorsqu'ils sont référencés par d'autres objets dans la base de données. Toutefois, dans certains cas où la suppression d'un objet (par exemple un poste de travail) implique uniquement la mise à jour d'un objet de référence (par exemple, une classe de postes de travail qui le comprend), la suppression peut être autorisée. Le tableau 56 montre tous les cas où un objet référencé peut être supprimé même si d'autres objets y font référence :

Tableau 56. Mise à jour de la définition d'objet lors de la suppression de l'objet référencé

Objet	Références	Lors de la suppression de l'objet référencé ...
Dépendance interréseau	Poste de travail	... suppression de la dépendance du travail ou du flot de travaux
Dépendance de prédécesseur/successeur externe	Flot de travaux	... suppression de la dépendance du travail ou du flot de travaux
	Travail	... suppression de la dépendance du travail ou du flot de travaux
Dépendance interne	Travail	... suppression de la dépendance du travail ou du flot de travaux
Classe de poste de travail	Poste de travail	... suppression du poste de travail de la classe de postes de travail

Le tableau 57, à la page 335 décrit le comportement du produit lorsqu'il doit supprimer un objet désigné par un autre objet à l'aide d'une relation spécifique :

Tableau 57. Vérification de l'intégrité référentielle lors de la suppression d'un objet de la base de données

Objet à supprimer	Désigné par l'objet	Relation	Règle de suppression
Domaine A	Domaine B	Le domaine A est parent du domaine B	Un message d'erreur spécifiant la relation existante s'affiche.
	Poste de travail B	Le poste de travail B appartient au domaine A	Un message d'erreur spécifiant la relation existante s'affiche.
Poste de travail A	Poste de travail B	Le poste de travail A est l'hôte du poste de travail B	Un message d'erreur spécifiant la relation existante s'affiche.
	Travail B	Le travail B est défini sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Flot de travaux B	Le flot de travaux B est défini sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Utilisateur B	L'utilisateur B est défini sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Flot de travaux B	Le poste de travail A fonctionne comme agent réseau pour les dépendances interréseaux définies dans le flot de travaux B	Le poste de travail A et la dépendance interréseau sont supprimés
	Flot de travaux B	Le flot de travaux B possède une dépendance de fichier à partir d'un fichier défini sur le poste de travail A	Le poste de travail A et la dépendance de fichier sont supprimés
	Travail B dans le flot de travaux B	Le poste de travail A fonctionne comme agent réseau pour les dépendances interréseaux définies dans le travail B	Le poste de travail A et la dépendance interréseau sont supprimés
	Travail B dans le flot de travaux B	Le travail B possède une dépendance de fichier à partir d'un fichier défini sur le poste de travail A	Le poste de travail A et la dépendance de fichier sont supprimés
	Ressource B	La ressource B est définie sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Fichier B	Le fichier B est défini sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Classe de postes de travail B	Le poste de travail A appartient à la classe de postes de travail B	Le poste de travail A et son entrée dans la classe de postes de travail B sont supprimés.
	Travail B dans le flot de travaux B	Le travail B contenu dans le flot de travaux B est défini sur le poste de travail A	Un message d'erreur spécifiant la relation existante s'affiche.

Tableau 57. Vérification de l'intégrité référentielle lors de la suppression d'un objet de la base de données (suite)

Objet à supprimer	Désigné par l'objet	Relation	Règle de suppression
Travail A	Travail B	Le travail A est un travail de reprise du travail B	Un message d'erreur spécifiant la relation existante s'affiche.
	Flot de travaux B	Le travail A est contenu dans le flot de travaux B	Un message d'erreur spécifiant la relation existante s'affiche.
	Flot de travaux B	Le flot de travaux B suit le travail A	Le travail A et la dépendance de prédécesseur/successeur dans le flot de travaux B sont supprimés.
	Travail B dans le flot de travaux B	Le flot de travaux B suit le travail A	Le travail A et la dépendance de prédécesseur/successeur dans le travail B sont supprimés.
	règle d'événement B	Le travail A est dans la définition d'action de la règle d'événement B (et ne fait appel à aucune substitution de variable)	Un message d'erreur spécifiant la relation existante s'affiche.
Agenda A	Flot de travaux B	Le flot de travaux B utilise l'agenda A	Un message d'erreur spécifiant la relation existante s'affiche.
Classe de postes de travail A	Travail B	Le travail B est défini sur la classe de postes de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Flot de travaux B	Le flot de travaux B est défini sur la classe de postes de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Ressource B	La ressource B est définie sur la classe de postes de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
	Fichier B	Le fichier B est défini sur la classe de postes de travail A	Un message d'erreur spécifiant la relation existante s'affiche.
Ressource A	Flot de travaux B	Nécessite la dépendance définie dans le flot de travaux B	Un message d'erreur spécifiant la relation existante s'affiche.
	Travail B dans le flot de travaux B	Nécessite la dépendance définie dans le travail B	Un message d'erreur spécifiant la relation existante s'affiche.
Invite A	Flot de travaux B	Dépendance d'invite définie dans le flot de travaux B	Un message d'erreur spécifiant la relation existante s'affiche.
	Travail B dans le flot de travaux B	Dépendance d'invite définie dans le travail B	Un message d'erreur spécifiant la relation existante s'affiche.

Tableau 57. Vérification de l'intégrité référentielle lors de la suppression d'un objet de la base de données (suite)

Objet à supprimer	Désigné par l'objet	Relation	Règle de suppression
variable A	Flot de travaux B	variable A est utilisé dans le flot de travaux B dans : - le texte d'une invite ad hoc - ou le nom de fichier spécifié dans une dépendance de fichier	variable A est supprimé sans aucune vérification
	Travail B	variable A est utilisé dans le flot de travaux B dans : - le texte d'une invite ad hoc - ou le nom de fichier spécifié dans une dépendance de fichier - ou la valeur spécifiée pour connexion_flot - ou la valeur spécifiée pour nom_script	variable A est supprimé sans aucune vérification
	Invite B	variable A est utilisé dans le texte de l'invite B	variable A est supprimé sans aucune vérification
table de variables A	Flot de travaux B	table de variables A est référencé dans le flot de travaux B	table de variables A n'est pas supprimé
	Travail B	table de variables A est référencé dans le travail B	table de variables A n'est pas supprimé
	Invite B	table de variables A est référencé dans le texte de l'invite B	table de variables A n'est pas supprimé
Flot de travaux A	Flot de travaux B	Le flot de travaux B suit le flot de travaux A	Le flot de travaux A et la dépendance de prédécesseur/successeur dans le flot de travaux B sont supprimés.
	Travail B dans un flot de travaux B	Le travail B suit le flot de travaux A	Le flot de travaux A et la dépendance de prédécesseur/successeur dans le travail B sont supprimés.
	règle d'événement B	Le flot de travaux A est dans la définition d'action de la règle d'événement B (et ne fait appel à aucune substitution de variable)	Un message d'erreur spécifiant la relation existante s'affiche.
domaine de sécurité A	liste de contrôle d'accès B	la liste de contrôle d'accès B est définie sur le domaine de sécurité A	la liste de contrôle d'accès B est supprimée.
rôle de sécurité A	liste de contrôle d'accès B	le rôle de sécurité A est référencé dans la liste de contrôle d'accès B	le rôle de sécurité A n'est pas supprimé.

add

Ajoute ou met à jour les objets de planification dans la base de données.

Autorisation

Vous disposer des droits d'accès **add** pour ajouter un nouvel objet de planification. Si l'objet existe déjà dans la base de données, vous devez avoir les droits suivants :

- droit d'accès **modify** sur l'objet si ce dernier n'est pas verrouillé.
- droit d'accès **modify** et **unlock** sur l'objet s'il est verrouillé par un autre utilisateur.

Pour **ajouter** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{add | a} nom_fichier [;unlock]
```

Arguments

nom_fichier

Indique le nom du fichier texte qui contient les définitions d'objet. Pour les règles d'événement, *nom_fichier* spécifie le nom du fichier XML contenant les définitions des règles d'événement que vous souhaitez ajouter (voir «Définition de règle d'événement», à la page 291 pour les références XML et «Editeur Composer», à la page 323 pour des informations sur la configuration d'un éditeur XML).

;unlock

Indique que les définitions d'objet doivent être déverrouillées si elles ont été verrouillées par le même utilisateur dans la même session. Si vous n'avez pas verrouillé l'objet et que vous utilisez l'option **;unlock**, vous recevez un message d'erreur et l'objet n'est pas remplacé.

Commentaires

Le fichier texte est validé sur le client et, s'il est correct, les objets sont insérés dans la base de données sur le gestionnaire de domaine maître. Le programme **composer** transforme les définitions d'objet en définition XML utilisée sur le serveur. Sinon, la commande est interrompue et un message d'erreur s'affiche. Ceci ne s'applique pas aux définitions de règles d'événement, car elles sont fournies directement au format XML.

Avec la commande **add**, le programme vous demande si vous souhaitez remplacer les objets existants. Ce comportement ne concerne pas les définitions de travail existantes dans les flots de travaux et les définitions de travail sont automatiquement mises à jour sans aucun message d'invite. Vous pouvez utiliser l'option **unlock** pour mettre à jour des objets existants que vous avez précédemment verrouillés à l'aide d'une seule commande. Pour tous les nouveaux objets insérés, l'option est ignorée. Si vous modifiez le nom d'un objet, il sera interprété par **composer** comme un nouvel objet et il sera inséré. Une commande **rename** est recommandée dans ce cas.

La commande add vérifie la présence de dépendances de boucle à l'intérieur des flots de travaux. Par exemple, si job1 suit job2 et si job2 suit job1, il y a une dépendance de boucle. Lorsqu'une dépendance de boucle à l'intérieur d'un flot de travaux est détectée, une erreur est affichée.

La commande add ne vérifie pas les dépendances de boucle entre les flots de travaux parce que selon la complexité des activités de planification, cette vérification pourrait prendre trop de temps.

Exemples

Pour ajouter les travaux du fichier myjobs, exécutez la commande suivante :

```
add myjobs
```

Pour ajouter les flots de travaux du fichier mysked, exécutez la commande suivante :

```
a mysked
```

Pour ajouter les postes de travail, les classes de postes de travail et les domaines du fichier cpus.src, exécutez la commande suivante :

```
a cpus.src
```

Pour ajouter les définitions d'utilisateur du fichier users_nt, exécutez la commande suivante :

```
lata users_nt
```

Pour ajouter les définitions de règles d'événement, vous éditez un fichier nommé newrules.xml, exécutez :

```
a newrules.xml
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

le manuel *Dynamic Workload Console - Guide d'utilisation*.

- Pour ajouter des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section "Création de postes de travail distribués".
- Pour ajouter des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Création d'une règle d'événement.
- Pour ajouter des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour ajouter d'autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".
- Pour ajouter des modèles d'application de charge de travail, voir
la section relative à la création d'un modèle d'application de charge de travail dans le manuel *Dynamic Workload Console - Guide d'utilisation*.

authenticate

Bascule vers d'autres droits d'accès d'utilisateur lors de l'exécution de **composer**.

Autorisation

Tout utilisateur autorisé à exécuter **composer** est autorisé à émettre cette commande.

Syntaxe

```
{authenticate | au} [username=nom_utilisateur password=mot_de_passe]
```

Arguments

username=*nom_utilisateur*

Nom d'utilisateur de l'utilisateur vers lequel vous souhaitez basculer.

password=*mot_de_passe*

Mot de passe de l'utilisateur vers lequel vous souhaitez basculer.

Commentaires

Un message indique l'échec ou la réussite de l'authentification. Cette commande est uniquement utilisée en mode interactif.

Exemples

Pour basculer vers l'utilisateur **utilisateur_tws1** avec le mot de passe **mon_mot_de_passe1** à partir du programme de ligne de commande **composer**, exécutez la commande suivante :

```
au username=tws_user1 password=mypasswd1
```

continue

Indique que l'erreur de commande suivante doit être ignorée.

Autorisation

Tout utilisateur autorisé à exécuter **composer** est autorisé à émettre cette commande.

Syntaxe

```
{continue | c}
```

Commentaires

Cette commande est utile lorsque plusieurs commandes sont entrées sur la ligne de commande ou redirigées à partir d'un fichier. Elle indique au programme **composer** de continuer à exécuter les commandes même si celle qui a été lancée après **continue** génère une erreur. Elle n'est pas nécessaire lorsque vous entrez des commandes de manière interactive dans la mesure où le programme **composer** ne s'arrête pas sur une erreur.

Exemples

Si vous souhaitez que le programme composer continue à traiter la commande **print** si la commande **delete** génère une erreur, exécutez la commande suivante :
composer "c&delete cpu=site4&print cpu=@"

delete

Permet de supprimer des définitions d'objet dans la base de données.

Autorisation

Pour **supprimer** des objets de planification, vous devez détenir l'accès *delete* aux objets à supprimer.

Pour **supprimer** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{delete | de}
[[calendars | calendar | cal=nom_agenda] |
[domain | dom=nom_domaine] |
[eventrule | erule | er=nom_règle_événmt] |
[parms | parm | vb=[nom_table.]nom_variable] |
[prompts | prom=nom_invite] |
[resources | resource | res=[poste_de_travail#]nom_ressource] |
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |
[table | vt=nom_table] |
[wat=workloadapplicationtemplatename]
[cpu={workstationname [:force] | workstationclassname [:force] | domainame}]
[workstation | ws=workstationname] [:force] |
[workstationclass | wscl=workstationclassname] [:force] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
[début de validité date | fin de validité date | valid
in date date]] |
[users |
user=[nom_poste_travail#]nom_utilisateur] |
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain | sdom=nom_domaine_sécurité] |
[securityrole | srol=nom_rôle_sécurité]}
[:noask]
```

Arguments

calendars | calendar | cal

Si aucun argument ne suit, permet de supprimer toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet de supprimer l'agenda spécifié. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de supprimer toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet de supprimer le domaine spécifié. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de supprimer toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événement* suit, permet de supprimer la règle d'événement spécifiée. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de supprimer toutes les définitions de variable globale trouvées dans la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet de supprimer la variable *nom_variable* de la table *nom_table*. Si *nom_table* est ignoré, **composer** recherche la définition de variable dans la table de variables par défaut. Les caractères génériques sont acceptés pour *nom_table* et *nom_variable*. Par exemple :

```
delete parms=@.@"
```

Permet de supprimer toutes les variables de toutes les tables.

```
delete parms=@
```

Permet de supprimer toutes les variables de la table par défaut.

```
delete parms=@.acct@"
```

Permet de supprimer toutes les variables dont le nom commence par acct de toutes les tables existantes.

A faire : Lors de la suppression d'une variable, la table de variables la contenant est verrouillée. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle ou sur les variables qu'elle contient.

prompts | prom

Si aucun argument ne suit, permet de supprimer toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet de supprimer l'invite spécifiée. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de supprimer toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de supprimer la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de supprimer toutes les définitions de groupe de cycle d'exécution.

Si l'argument *nom_groupe_cycle_exécution* suit, permet de supprimer le groupe de cycle d'exécution spécifié. Les caractères génériques sont acceptés.

vartable | vt

Si aucun argument ne suit, permet de supprimer toutes les définitions de table de variables.

Si la table de variables *nom_table* de l'argument suit, permet de supprimer la table de variables spécifiée. Les caractères génériques sont acceptés.

wat

Si aucun argument ne suit, permet de supprimer toutes les définitions de modèle d'application de charge de travail.

Si l'argument *modèle_application_charge_travail* suit, permet de supprimer le modèle d'application de charge de travail spécifié. Les caractères génériques sont acceptés.

cpu

Supprime les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

Si vous indiquez l'argument **force**, la définition de poste de travail est supprimée de la base de données IBM Workload Scheduler.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés. Si vous indiquez l'argument **force**, la classe de poste de travail est supprimée de la base de données IBM Workload Scheduler.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de supprimer toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet de supprimer le poste de travail spécifié. Les caractères génériques sont acceptés. Si vous indiquez l'argument **force**, la définition de poste de travail est supprimée de la base de données IBM Workload Scheduler.

workstationclass | wscl

Si aucun argument ne suit, permet de supprimer toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_travail* suit, permet de supprimer la classe de poste de travail spécifiée. Les caractères génériques sont acceptés. Si vous indiquez l'argument **force**, la définition de classe de poste de travail est supprimée de la base de données IBM Workload Scheduler.

jobs | **jobdefinition** | jd

Si aucun argument ne suit, permet de supprimer toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de supprimer le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | **jobstream** | js

Si aucun argument ne suit, permet de supprimer toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de supprimer le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

users | user

Si aucun argument ne suit, permet de supprimer toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de supprimer l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*. La zone du mot de passe n'est pas copiée pour des raisons de sécurité.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de supprimer les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de supprimer les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de supprimer toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de supprimer la définition de domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de supprimer toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet de supprimer la définition de rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

;noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque objet éligible.

Commentaires

Si vous utilisez des caractères génériques pour spécifier plusieurs définitions, le programme **composer** demande confirmation avant de supprimer une définition répondant aux critères. Une confirmation est requise avant la suppression de chaque définition correspondante si vous ne spécifiez pas l'option **noask**.

Les objets à supprimer ne doivent pas être verrouillés. Si certains objets correspondants sont verrouillés lors du traitement de la commande, un message d'erreur s'affiche avec la liste de ces objets.

Exemples

Pour supprimer le travail job3 qui a été lancé sur le poste de travail site3, exécutez la commande suivante :

```
delete jobs=site3#job3
```

Pour supprimer tous les postes de travail dont le nom commence par ux, exécutez la commande suivante :

```
de cpu=ux@
```

Pour supprimer tous les flots de travaux dont le nom commence par test sur tous les postes de travail, exécutez la commande suivante :

```
de sched=@#test@
```

Pour supprimer toutes les règles d'événement nommées de rulejs320 à rulejs329, exécutez la commande suivante :

```
de erule=rulejs32?
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour supprimer des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section Edition de définitions de poste de travail.
- Pour supprimer des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Edition d'une règle d'événement.
- Pour supprimer des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour supprimer tous les autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

display

Permet d'afficher les détails d'une ou plusieurs définitions d'objet du même type enregistrées dans la base de données. La définition complète de l'objet s'affiche.

Autorisation

Pour **afficher** des objets de planification, vous devez détenir l'accès *display* à l'objet à afficher. Si vous souhaitez utiliser le mot clé **full**, vous devez également avoir le droit d'accès *display* aux travaux contenus dans la définition du flot de travaux. Si vous ne possédez pas l'accès requis, le composeur ne parvient pas à trouver les objets.

Pour **afficher** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **display** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{display | di}
[[calendars | calendar | cal=nom_agenda] |
[eventrule | erule | er=nom_règle_événmt] |
[parms | parm | vb=nom_variable.]nom_variable] |
[vartable | vt=nom_table] |
[prompts | prom=nom_invite] |
[resources | resource | res=[poste_de_travail#]nom_ressource] |
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |
[cpu={nom_poste_de_travail | nom_classe_poste_de_travail | nom_domaine}]
[wat=nom_modèle_application_charge_travail]
[workstation | ws=nom_poste_travail] |
[workstationclass | wscl=nom_classe_poste_de_travail] |
[domain | dom=nom_domaine] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
    [valid from date | valid to date | valid in date date]
    [,full]] |
[users |
user=[nom_poste_travail#]nom_utilisateur] |
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain | sdom=nom_domaine_sécurité] |
[securityrole | srol=nom_rôle_sécurité]}
[:offline]
```

Arguments

calendars | calendar | cal

Si aucun argument ne suit, permet d'afficher toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet d'afficher l'agenda *nom_agenda*. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet d'afficher toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événmt* suit, permet d'afficher la règle d'événement *nom_règle_événmt*. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet d'afficher toutes les définitions de variable globale trouvées dans la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet d'afficher la variable *nom_variable* de la table spécifiée. Si la table de variables *nom_table* est ignorée, **composer** recherche la définition de variable dans la table de variables par défaut. Les caractères génériques sont utilisés dans la table de variables *nom_table* et dans la variable *nom_variable*. Par exemple :

```
display parms=@.@"
```

Permet d'afficher toutes les variables dans toutes les tables.

```
display parms=@
```

Permet d'afficher toutes les variables dans la table par défaut.

```
display parms=@.acct@"
```

Permet d'afficher toutes les variables dont le nom commence par *acct* dans toutes les tables existantes.

variable | vt

Si aucun argument ne suit, permet d'afficher toutes les définitions de table de variables.

Si la table de variables *nom_table* de l'argument suit, permet d'afficher la table de variables *nom_table*. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet d'afficher toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet d'afficher l'invite *nom_invite*. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet d'afficher toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet d'afficher la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet d'afficher toutes les définitions de groupe de cycle d'exécution.

Si l'argument *runcyclegroupname* suit, permet d'afficher le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu

Affiche les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

wat

Si aucun argument ne suit, permet d'afficher toutes les définitions de modèle d'application de charge de travail.

| Si l'argument *workloadapplicationtemplate* suit, permet d'afficher le modèle
| d'application de charge de travail spécifié. Les caractères génériques sont
| acceptés.

workstation | ws

Si aucun argument ne suit, permet d'afficher toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet d'afficher le poste de travail *nom_poste_travail*. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet d'afficher toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet d'afficher le domaine *nom_domaine*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet d'afficher toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_de_travail* suit, permet d'afficher la classe de poste de travail *nom_classe_poste_de_travail*. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet d'afficher toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet d'afficher le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet d'afficher toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet d'afficher le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

full Affiche également toutes les définitions de travail contenues dans le flot de travaux.

users | user

Si aucun argument ne suit, permet d'afficher toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet d'afficher l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet d'afficher les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet d'afficher les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet d'afficher toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet d'afficher la définition de domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet d'afficher toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet d'afficher la définition de rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

;offline

Envoie la sortie de la commande au périphérique de sortie de **composer**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 322.

Résultats

La commande **display** vous renvoie les informations suivantes sur l'objet à afficher :

- une ligne récapitulative contenant des informations sur l'objet sélectionné
- la définition d'objet sélectionnée

En fonction de la valeur définie dans la variable locale **MAESTROCOLUMNS**, la ligne récapitulative montre différents ensembles d'information sur l'objet sélectionné.

Le tableau 58 illustre la sortie produite en fonction de la valeur définie pour la variable **MAESTROCOLUMNS**.

Tableau 58. Formats de sortie de l'affichage des objets de planification

Type d'objet	Format de sortie si MAESTROCOLUMNS<120	Format de sortie si MAESTROCOLUMNS 120
Agenda	"CalendarName : UpdatedOn : UpdatedBy : LockedBy"	"CalendarName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Tableau 58. Formats de sortie de l'affichage des objets de planification (suite)

Type d'objet	Format de sortie si MAESTROCOLUMNS<120	Format de sortie si MAESTROCOLUMNS 120
Domaine	"DomainName : ParentDomain : Master : UpdatedOn : LockedBy"	"DomainName : ParentDomain : Master : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Règle d'événement	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy"	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy : LockedOn"
Travail	"Workstation : JobDefinitionName : UpdatedOn : LockedBy"	"Workstation : JobDefinitionName : TaskType : UpdatedBy : LockedBy : LockedOn"
Flot de travaux	"Workstation : JobstreamName : Validfrom : UpdatedOn : LockedBy"	"Workstation : JobstreamName : Draft : ValidFrom : ValidTo : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Paramètre	"VariableTableName : VariableName : UpdatedOn : LockedBy"	"VariableTableName : VariableName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Invite	"PromptName : UpdatedOn : LockedBy "	"PromptName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Ressource	"Workstation : ResourceName : Quantity : UpdatedOn : LockedBy "	"Workstation : ResourceName : Quantity : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Table de variables	"VariableTableName : Default : UpdatedOn : LockedBy "	"VariableTableName : Default : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Utilisateur	"Workstation : UserName : UpdatedOn : LockedBy"	"UserName : Workstation : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Poste de travail	"WorkstationName : Type : Domain : Ignored : UpdatedOn : LockedBy"	"WorkstationName : Type : Domain : OsType : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Classe de poste de travail	"WorkstationClassName : Ignored : UpdatedOn : LockedBy"	"WorkstationClassName : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Voir «Sortie offline», à la page 322 pour plus d'informations sur la définition de **MAESTROCOLUMNS**.

Exemples

Pour afficher tous les agendas, exécutez la commande suivante :

```
display calendars=@
```

Voici un exemple de sortie :

```
Calendar Name   Updated On Locked By
-----
HOLIDAYS        12/31/2005   tws83
HOLIDAYS
01/01/2006 02/15/2006 05/31/2006

Calendar Name   Updated On Locked By
-----
```

```

MONTHEND          01/01/2006  -

MONTHEND
  "Month end dates 1st half 2006"
  01/31/2006 02/28/2006 03/31/2006 04/30/2006 05/31/2006 06/30/2006

Calendar Name    Updated On Locked By
-----
PAYDAYS          01/02/2006  -

PAYDAYS
  01/15/2006 02/15/2006 03/15/2006 04/15/2006 05/14/2006 06/15/2006

```

Pour imprimer la sortie de la commande d'affichage sur tous les flots de travaux lancés sur le poste de travail site2, exécutez la commande suivante :

```
di sched=site2#@;offline
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour afficher des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section Edition de définitions de poste de travail.
- Pour afficher des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Edition d'une règle d'événement.
- Pour afficher des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour afficher tous les autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

edit

Permet d'éditer un fichier.

Autorisation

Tout utilisateur autorisé à exécuter composer est autorisé à émettre cette commande.

Syntaxe

```
{edit | ed} nom_fichier
```

Arguments

nom_fichier

Nom du fichier à modifier.

Commentaires

Il convient de lancer un éditeur et d'ouvrir le fichier à modifier dans ce programme. Pour plus d'informations, voir «Editeur Composer», à la page 323.

Exemples

Pour modifier le fichier `mytemp`, exécutez la commande suivante :

```
edit mytemp
```

Pour modifier le fichier `resfile`, exécutez la commande suivante :

```
ed resfile
```

exit

Quitte le programme de ligne de commande **composer**.

Autorisation

Tout utilisateur autorisé à exécuter **composer** est autorisé à émettre cette commande.

Syntaxe

```
{exit | e}
```

Commentaires

Si vous exécutez le programme de ligne de commande **composer** en mode Aide, la commande active le programme **composer** en mode Saisie de commande.

Exemples

Pour quitter le programme de ligne de commande **composer**, exécutez la commande suivante :

```
exit
```

ou :

```
e
```

extract

Crée un fichier texte contenant les définitions d'objet extraites de la base de données.

Autorisation

Pour **extraire** un objet de planification, vous devez détenir l'accès *display* aux objets en cours de copie, ainsi que l'accès *modify* si vous souhaitez utiliser le mot-clé **lock**. Si vous ne possédez pas l'accès requis, le compositeur ne parvient pas à trouver les objets.

Pour **extraire** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
[create | cr | extract | ext] nom_fichier from
[[calendars | calendar | cal=nom_agenda] |
[eventrule | erule | er=nom_règle_événmt] |
[parms | parm | vb=[nom_table.]nom_variable] |
[table | vt=nom_table] |
[prompts | prom=nom_invite] |
[resources | resource | res=[poste_de_travail#]nom_ressource] |
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |
[cpu={nom_poste_travail | classe_poste_de_travail | nom_domaine}] |
[workstation | ws=nom_poste_travail] |
[workstationclass | wscl=nom_classe_poste_de_travail] |
[domain | dom=nom_domaine] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
    [valid from date | valid to date | valid in date date]
    [:full]] |
[users |
user=[nom_poste_travail#]nom_utilisateur
[:password]] |
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain | sdom=nom_domaine_sécurité] |
[securityrole | srol=nom_rôle_sécurité]
[:lock]
```

Arguments

nom_fichier

Indique le nom du fichier qui contient les définitions d'objet.

calendars | calendar | cal

Si aucun argument ne suit, permet de copier toutes les définitions d'agenda dans le fichier.

Si l'argument *nom_agenda* suit, permet de copier l'agenda *nom_agenda* dans le fichier. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de copier toutes les définitions de règle d'événement dans le fichier XML.

Si l'argument *nom_règle_événmt* suit, permet de copier la règle d'événement *nom_règle_événmt* dans le fichier. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de copier toutes les définitions de variable globale trouvées dans la table de variables par défaut dans le fichier.

Si l'argument *nom_table.nom_variable* suit, permet de copier la variable *nom_variable* de la variable *nom_table* spécifiée dans le fichier. Si la table de variables *nom_table* est ignorée, **composer** recherche la définition de variable dans la table de variables par défaut. Les caractères génériques sont acceptés dans la table de variables *nom_table* et dans la variable *nom_variable*.

Par exemple :

```
create parmfile from parms=@.@"
```

Permet de copier toutes les variables à partir de toutes les tables.
`create parmfile from parms=@`

Permet de copier toutes les variables à partir de la table par défaut.
`create parmfile from parms=@.acct@`

Permet de copier toutes les variables dont le nom commence par `acct` à partir de toutes les tables existantes.

A faire : L'utilisation de l'option `lock` sur une variable permet de verrouiller la table de variables qui la contient. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle ou sur les variables qu'elle contient.

variable | vt

Si aucun argument ne suit, permet de copier toutes les définitions de table de variables dans le fichier.

Si la table de variables *nom_table* de l'argument suit, permet de copier la table de variables *nom_table* dans le fichier. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet de copier toutes les définitions d'invite dans le fichier.

Si l'argument *nom_invite* suit, permet de copier l'invite *nom_invite* dans le fichier. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de copier toutes les définitions de ressource dans le fichier.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de copier la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie dans le fichier. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_de_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de copier toutes les définitions de groupe de cycle d'exécution dans le fichier.

Si l'argument *runcyclegroupname* suit, permet de copier le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu

Copie les postes de travail, les classes de postes de travail ou les domaines dans le fichier.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de copier toutes les définitions de poste de travail dans le fichier.

Si l'argument *nom_poste_travail* suit, permet de copier le poste de travail *nom_poste_travail* dans le fichier. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de copier toutes les définitions de domaine dans le fichier.

Si l'argument *nom_domaine* suit, permet de copier le domaine *nom_domaine* dans le fichier. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de copier toutes les définitions de classe de poste de travail dans le fichier.

Si l'argument *nom_classe_poste_de_travail* suit, permet de copier la classe de poste de travail *nom_classe_poste_de_travail* dans le fichier. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet de copier toutes les définitions de travail dans le fichier.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de copier le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute dans le fichier. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet de copier toutes les définitions de flot de travaux dans le fichier.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de copier le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini dans le fichier. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

full

Copie également toutes les définitions de travail contenues dans le flot de travaux.

users | user

Si aucun argument ne suit, permet de copier toutes les définitions d'utilisateur dans le fichier.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de copier l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini dans le fichier. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

Si vous n'ajoutez pas l'option **;password**, le mot de passe défini pour l'utilisateur est enregistré dans le fichier de sortie sous la forme d'une séquence de 10 astérisques (*) et ne peut pas être réutilisé.

Si vous ajoutez l'option **;password**, le mot de passe défini pour l'utilisateur est chiffré et enregistré dans le fichier de sortie. Il peut être ainsi réimporté et utilisé à nouveau.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de copier les définitions de listes de contrôle d'accès dans le fichier pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de copier les définitions de listes de contrôle d'accès dans le fichier pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de copier toutes les définitions de domaines de sécurité dans le fichier.

Si un argument *nom_domaine_sécurité* suit, permet de copier la définition de domaine de sécurité *nom_domaine_sécurité* dans le fichier. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de copier toutes les définitions de rôles de sécurité dans le fichier.

Si un argument *nom_rôle_sécurité* suit, permet de copier la définition de rôle de sécurité *nom_rôle_sécurité* dans le fichier. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

;lock Indique que l'objet sélectionné doit toujours être verrouillé.

Commentaires

Vous pouvez utiliser cette commande pour créer un fichier contenant des définitions de paramètres à importer dans la base de données de paramètres définie localement sur un poste de travail. Pour plus d'informations sur la façon d'importer les définitions de paramètres localement, voir «parms», à la page 710.

Vous pouvez appeler la commande avec l'ancien nom "create" ou le nouveau nom "extract". Sans option **lock**, le verrouillage de la base de données n'est pas vérifié et tous les objets correspondants sont extraits dans le fichier. Après avoir créé un fichier, vous pouvez le modifier à l'aide de la commande **edit** et ajouter ou mettre à jour la base de données à l'aide de la commande **add** ou **replace**.

Avec l'option **lock**, vous pouvez préciser si l'utilisateur de la base de données doit laisser verrouillés les objets qui répondent aux critères sélectionnés. Au cours de l'extraction, si **composer** détecte que certains de ces objets sont déjà verrouillés par un autre utilisateur, ces objets ne sont pas insérés dans le fichier et un message

stdout s'affiche pour chaque objet verrouillé.

Exemples

Pour créer un fichier nommé *caltemp* contenant tous les agendas, exécutez la commande suivante :

```
create caltemp from calendars=@
```

Pour créer un fichier nommé *stemp* contenant tous les flots de travaux définis sur le poste de travail sur lequel s'exécute **composer**, exécutez la commande suivante :

```
cr stemp from jobstream=@
```

Pour créer un fichier nommé *alljobs.txt* contenant toutes les définitions de travail, exécutez la commande suivante :

```
extract alljobs.txt from jd=@#@
```

Pour créer un fichier nommé *allrules.xml* contenant toutes les définitions de règles d'événement, exécutez la commande suivante :

```
ext allrules.xml from erule=@
```

Pour créer un fichier intitulé *dbmainadm.txt* avec la définition de l'utilisateur princeps du poste de travail *dbserv349*, y compris le mot de passe chiffré, exécutez la commande :

```
composer extract c:\dbmainadm.txt from user=dbserv349#princeps;password
```

Le contenu du fichier *dbmainadm.txt* sera :

```
USERNAME princeps
      PASSWORD "ENCRYPT:EIu7PP+gvS8="
FIN
```

help

Permet d'afficher l'aide en ligne d'une commande ou la liste des commandes qui peuvent être exécutées à partir de **composer**. Non disponible sous Windows.

Autorisation

Tout utilisateur autorisé à exécuter **composer** est autorisé à émettre cette commande.

Syntaxe

```
{help | h} {commande | mot_clé}
```

Arguments

commande

Indique le nom d'une commande **composer** ou d'une commande système. Pour les commandes **composer**, entrez le nom complet de la commande ; les abréviations et les formats abrégés ne sont pas pris en charge.

mot_clé

Vous pouvez également entrer les mots clés suivants :

COMMANDS

Répertorie toutes les commandes **composer**.

RUNCONPOSER

Comment exécuter composer.

SETUPCOMPOSER

Décrit comment configurer la commande composer.

SPECIALCHAR

Décrit les caractères génériques, les délimiteurs et les autres caractères spéciaux que vous pouvez utiliser.

Exemples

Pour afficher une liste des toutes les commandes **composer**, exécutez la commande suivante :

```
help commands
```

Pour afficher des informations sur la commande **add**, exécutez la commande suivante :

```
help add
```

Pour afficher des informations relatives aux caractères spéciaux que vous pouvez utiliser, exécutez la commande suivante :

```
h specialchar
```

list

Permet de répertorier ou d'imprimer les informations récapitulatives sur des objets définis dans la base de données IBM Workload Scheduler. La commande **list** vous fournit la liste des noms d'objets et de leurs attributs. La commande **Print** envoie la liste des noms d'objets et de leurs attributs au périphérique ou au fichier défini dans la variable locale **MAESTROL**. La commande **print** peut être utilisée pour envoyer la sortie à une imprimante locale, à condition que la variable **MAESTROL** soit définie en conséquence.

Autorisation

Si l'option globale **enListSecChk** est définie sur **yes** sur le questionnaire de domaine maître, vous devez, pour afficher ou imprimer un objet, posséder un accès de type **list** et/ou **list** et **display**.

Pour **lister** les objets de sécurité, vous devez détenir des droits pour exécuter l'action **display** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{list | l}  
[calendars | calendar | cal=nom_agenda] |  
[eventrule | erule | er=nom_règle_événmt] |  
[parms | parm | vb=[nom_table.]nom_variable] |  
[varable | vt=nom_table] |  
[prompts | prom=nom_invite] |  
[resources | resource | res=[poste_de_travail#]nom_ressource] |  
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |  
[cpu={nom_poste_travail | nom_classe_poste_de_travail | nom_domaine}]  
[wat=nom_modèle_application_charge_travail]  
[workstation | ws=nom_poste_travail] |  
[workstationclass | wsc=nom_classe_poste_de_travail] |
```

```

[domain | dom=nom_domaine] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
    [valid from date |
    valid to date | valid in date date] |
[users |
user=[nom_poste_travail#]nom_utilisateur
|
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain |
sdom=nom_domaine_sécurité]
|
[securityrole |
srol=nom_rôle_sécurité]}
[;offline]

```

Arguments

calendars | calendar | cal

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet de répertorier ou d'imprimer l'agenda *nom_agenda*. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événmt* suit, permet de répertorier ou d'imprimer la règle d'événement *nom_règle_événmt*. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de variable globale trouvées dans la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet de répertorier ou d'imprimer la variable *nom_variable* de la table *nom_table*. Si *nom_table* est ignoré, **composer** recherche la définition de variable dans la table de variables par défaut. Les caractères génériques sont utilisés dans *nom_table* et *nom_variable*. Par exemple :

```
list parms=@.@"
```

Répertorie toutes les variables de toutes les tables.

```
list parms=@
```

Répertorie toutes les variables de la table par défaut.

```
list parms=@.acct@"
```

Répertorie toutes les variables dont le nom commence par acct dans toutes les tables existantes.

vartable | vt

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de table de variables.

Si la variable *nom_table* de l'argument suit, permet de répertorier ou d'imprimer la table de variables *nom_table*. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet de répertorier ou d'imprimer l'invite *nom_invite*. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de répertorier ou d'imprimer la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de répertorier ou d'imprimer tous les groupes de cycle d'exécution.

Si l'argument *runcyclegroupname* suit, permet de répertorier ou d'imprimer le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu Répertorie ou imprime les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

wat Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de modèle d'application de charge de travail.

Si l'argument *workloadapplicationtemplate* suit, permet de répertorier ou d'imprimer le modèle d'application de charge de travail spécifié. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet de répertorier ou d'imprimer le poste de travail *nom_poste_travail*. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet de répertorier ou d'imprimer le domaine *nom_domaine*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_de_travail* suit, permet de répertorier ou d'imprimer la classe de poste de travail *nom_classe_poste_de_travail*. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de répertorier ou d'imprimer le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de répertorier ou d'imprimer le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

users | user

Si aucun argument ne suit, permet de répertorier ou d'imprimer toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de répertorier ou d'imprimer l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_de_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

Remarque : Si vous répertoriez les utilisateurs Windows au format UPN *nom_utilisateur@domaine_internet*, insérez le caractère d'échappement '\' avant le caractère '@' dans la valeur *nom_utilisateur@domaine_internet*. Par exemple, si vous faites référence à l'utilisateur *administrator@bvt.com*, exécutez la commande suivante :

```
list users=administrator\@bvt.com
```

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de répertorier ou d'imprimer les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de répertorier ou d'imprimer les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de répertorier ou d'imprimer toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de répertorier ou d'imprimer la définition de domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de répertorier ou d'imprimer toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet de répertorier ou d'imprimer la définition de rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

;offline

Envoie la sortie de la commande au périphérique de sortie de **composer**. Pour plus d'informations sur ce périphérique, voir «Variables UNIX», à la page 322. La commande **list ;offline** est équivalente à la commande **print**.

Résultats

La commande **list** vous fournit la liste des noms d'objets et de leurs attributs. La commande **print** envoie la liste des noms d'objets et de leurs attributs au périphérique ou au fichier défini dans la variable locale *MAESTROL*P. La commande **print** peut être utilisée pour envoyer la sortie vers une imprimante locale si vous avez attribué une valeur à *MAESTROL*P en conséquence. Assurez-vous que la variable *MAESTROL*P est définie dans votre environnement avant d'exécuter la commande **print**.

En fonction de la valeur définie dans la variable locale *MAESTROCOLUMNS*, les différents ensembles d'informations sur l'objet sélectionné peuvent être affichés.

Le tableau 59 illustre la sortie produite en fonction de la valeur définie pour la variable *MAESTROCOLUMNS*.

Tableau 59. Formats de sortie de l'affichage des objets de planification

Type d'objet	Format de sortie si MAESTROCOLUMNS<120	Format de sortie si MAESTROCOLUMNS 120
Agenda	"CalendarName : UpdatedOn : UpdatedBy : LockedBy"	"CalendarName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Domaine, zone	"DomainName : ParentDomain : Master : UpdatedOn : LockedBy"	"DomainName : ParentDomain : Master : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Tableau 59. Formats de sortie de l'affichage des objets de planification (suite)

Type d'objet	Format de sortie si MAESTROCOLUMNS<120	Format de sortie si MAESTROCOLUMNS 120
Règle d'événement	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy"	"EventRuleName : Type : Draft : Status : UpdatedOn : LockedBy : LockedOn"
Travail	"Workstation : JobDefinitionName : UpdatedOn : LockedBy"	"Workstation : JobDefinitionName : TaskType : UpdatedBy : LockedBy : LockedOn"
Flot de travaux	"Workstation : JobstreamName : Validfrom : UpdatedOn : LockedBy"	"Workstation : JobstreamName : Draft : ValidFrom : ValidTo : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Paramètre	"VariableTableName : VariableName : UpdatedOn : LockedBy"	"VariableTableName : VariableName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Invite	"PromptName : UpdatedOn : LockedBy "	"PromptName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Ressource	"Workstation : ResourceName : Quantity : UpdatedOn : LockedBy "	"Workstation : ResourceName : Quantity : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Groupe de cycle d'exécution	"RunCycleGroupName : UpdatedOn : LockedBy "	"RunCycleGroupName : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Table de variables	"VariableTableName : Default : UpdatedOn : LockedBy "	"VariableTableName : Default : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
User	"Workstation : UserName : UpdatedOn : LockedBy"	"UserName : Workstation : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Poste de travail	"WorkstationName : Type : Domain : Ignored : UpdatedOn : LockedBy"	"WorkstationName : Type : Domain : OsType : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"
Classe de poste de travail	"WorkstationClassName : Ignored : UpdatedOn : LockedBy"	"WorkstationClassName : Ignored : UpdatedBy : UpdatedOn : LockedBy : LockedOn"

Voir «Sortie offline», à la page 322 pour plus d'informations sur la définition de **MAESTROL**.

Exemples

- Pour répertorier tous les agendas, exécutez la commande suivante :
list calendars=@

Voici un exemple de sortie :

```
Calendar Name   Updated On Locked By
-----
HOLIDAYS       03/02/2010
PAYDAYS        03/02/2010
HOLIDAYS       03/02/2010
01/01/2010 02/15/2010 05/31/2010
```

```
Calendar Name   Updated On Locked By
-----
```

MONTHEND 01/01/2010 -

MONTHEND

"Month end dates 1st half 2010"

01/31/2010 02/28/2010 03/31/2010 04/30/2010 05/31/2010 06/30/2010

Calendar Name Updated On Locked By

PAYDAYS 01/02/2010 -

PAYDAYS

01/15/2010 02/15/2010 03/15/2010 04/15/2010 05/14/2010 06/15/2010

- Pour répertorier toutes les règles d'événement définies, exécutez la commande suivante :

list er=@

Si MAESTROCOLUMNS=80, la sortie est similaire à ceci :

Event Rule Name	Type	Draft	Status	Updated On	Locked By
EVENT-MULTIPLE1	filter	active	06/06/2009	-	
EVENT-MULTIPLE2	filter	active	06/06/2009	-	
EVENT-MULTIPLE3	filter	active	06/06/2009	-	
M_SUCC_12_S	sequence	Y	inactive	06/07/2009	-
M_SUCC_12_S_A	filter	active	06/07/2009	-	
M_SUCC_12_S_B	filter	Y	inactive	06/07/2009	-
NEWEVENTRULE	filter	active	06/01/2009	administrator	

Si MAESTROCOLUMNS≥120 la sortie est similaire à ceci :

Event Rule Name	Type	Draft	Status	Updated On	Locked By
EVENT-MULTIPLE1	filter	active	06/06/2009	-	
EVENT-MULTIPLE2	filter	active	06/06/2009	-	
EVENT-MULTIPLE3	filter	active	06/06/2009	-	
M_SUCC_12_S	sequence	Y	inactive	06/07/2009	-
M_SUCC_12_S_A	filter	active	06/07/2009	-	
M_SUCC_12_S_B	filter	Y	inactive	06/07/2009	-
NEWEVENTRULE	filter	active	06/01/2009	administrator	

- Pour afficher les propriétés du poste de travail NC1150691 de l'agent, exécutez la commande suivante :

list ws=NC1150691

Le résultat affiché est du type suivant :

Workstation Name	Type	Domain	Ignored	Updated On	Locked By
NC1150691	agent	-		03/31/2010	-

CPUNAME NC1150691

DESCRIPTION "This workstation was automatically created at agent installation time."

OS WNT

NODE nc115069.romelab.it.ibm.com SECUREADDR 22114

TIMEZONE GMT+1

FOR MAESTRO HOST NC115069_DWB

TYPE AGENT

PROTOCOL HTTPS

FIN

- Pour afficher les propriétés du poste de travail du pool POOL_A, y compris de tous ses membres, exécutez la commande suivante :

list ws=POOL_A

Le résultat affiché est du type suivant :

Workstation Name	Type	Domain	Ignored	Updated On	Locked By
POOL_A	pool	-	03/31/2010	-	

```

CPUNAME POOL_A
DESCRIPTION "This is a manually created pool"
VARIABLE TABLE1
OS OTHER
TIMEZONE America/Argentina/Buenos_Aires
FOR MAESTRO HOST NC115069_DWB
TYPE POOL
MEMBERS
NC1150691
NC1150692
FIN

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour répertorier ou imprimer des postes de travail, voir *Dynamic Workload Console - Guide d'utilisation*, section Edition de définitions de poste de travail.
- Pour répertorier ou imprimer des règles d'événement, voir *Dynamic Workload Console - Guide d'utilisation*, section Edition d'une règle d'événement.
- Pour répertorier ou imprimer des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir *Dynamic Workload Console - Guide d'utilisation*, section relative à la gestion de sécurité de charge de travail.
- Pour répertorier ou imprimer tous les autres objets, voir *Dynamic Workload Console - Guide d'utilisation*, section Liste des définitions d'objet dans la base de données.

lock

Permet de verrouiller l'accès aux définitions des objets de planification dans la base de données.

Autorisation

Pour **verrouiller** des objets de planification, vous devez détenir l'accès *modify* à l'objet.

Pour **verrouiller** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```

{lock | lo}
{[calendars | calendar | cal=nom_agenda] |
[eventrule | erule | er=nom_règle_événmt] |
[parms | parm | vb=[nom_table.]nom_variable] |
[table | vt=nom_table] |
[prompts | prom=nom_invite] |
[resources | resource | res=[poste_de_travail#]nom_ressource] |

```

```
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |
[cpu={nom_poste_travail | nom_classe_poste_de_travail | nom_domaine}]
[workstation | ws=nom_poste_travail] |
[workstationclass | wscl=nom_classe_poste_de_travail] |
[domain | dom=nom_domaine] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
    [début de validité date | fin de validité date | valid
    in date date]] |
[users |
user=[nom_poste_travail#]nom_utilisateur] |
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain | sdom=nom_domaine_sécurité] |
[securityrole | srol=nom_rôle_sécurité]}
```

Arguments

calendars

Verrouille toutes les définitions d'agenda.

calendars | calendar | cal

Si aucun argument ne suit, permet de verrouiller toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet de verrouiller l'agenda *nom_agenda*. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de verrouiller toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événement* suit, permet de verrouiller la règle d'événement *nom_règle_événement*. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de verrouiller toute la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet de verrouiller toute la table contenant la variable *nom_variable*. Si *nom_table* est ignoré, **composer** verrouille toute la table de variables par défaut.

Remarque : Le verrouillage d'une table verrouille également toutes la table de variables qui la contient. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle.

Les caractères génériques sont utilisés dans *nom_table* et *nom_variable*. Par exemple :

```
lock parms=@.@
```

Permet de verrouiller toutes les variables sur toutes les tables. En conséquence, toutes les tables de variables sont verrouillées.

```
lock parms=@
```

Permet de verrouiller toutes les variables de la table par défaut. En conséquence, la table de variables est verrouillée.

```
lock parms=@.acct@
```

Permet de verrouiller toutes les variables dont le nom commence par *acct* dans toutes les tables existantes. En conséquence, toutes les tables de variables contenant au moins une variable nommée ainsi sont verrouillées.

variable | vt

Si aucun argument ne suit, permet de verrouiller toutes les définitions de table de variables.

Si la table de variables *nom_table* de l'argument suit, permet de verrouiller la table de variables *nom_table*. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet de verrouiller toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet de verrouiller l'invite *nom_invite*. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de verrouiller toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de verrouiller la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de verrouiller toutes les définitions de groupe de cycle d'exécution.

Si l'argument *runcyclegroupname* suit, permet de verrouiller le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu

Verrouille les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de verrouiller toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet de verrouiller le poste de travail *nom_poste_travail*. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de verrouiller toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet de verrouiller le domaine *nom_domaine*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de verrouiller toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_de_travail* suit, permet de verrouiller la classe de poste de travail *nom_classe_poste_de_travail*. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet de verrouiller toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de verrouiller le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet de verrouiller toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de verrouiller le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

users | user

Si aucun argument ne suit, permet de verrouiller toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de verrouiller l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de verrouiller les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de verrouiller les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de verrouiller toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de verrouiller la définition du domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité* .

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de verrouiller toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet de verrouiller la définition du rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité* .

Commentaires

Les objets sont verrouillés pour que les définitions de la base de données ne soient pas écrasées par différents utilisateurs qui accèdent simultanément aux mêmes objets.

Avec cette commande, l'utilisateur acquiert de façon explicite les serrures des objets de base de données. Lorsqu'un objet de l'utilisateur est verrouillé, tout autre utilisateur possède le droit d'accès en lecture seule jusqu'à ce que l'objet soit libéré ou déverrouillé de façon explicite par l'administrateur. Si un utilisateur essaie de verrouiller un objet déjà verrouillé par un autre utilisateur, un message d'erreur s'affiche.

Les verrous sur les objets de base de données sont acquis par l'utilisateur en utilisant `username` et `session`, où `session` est une chaîne qui peut être définie dans la variable d'environnement **TWS_SESSION** identifiant cette session utilisateur spécifique.

Ceci signifie que sur un ordinateur, l'identificateur **TWS_SESSION** est différent pour :

- un utilisateur connecté dans deux shells différents sur le programme de ligne de commande **composer**.
- un utilisateur connecté, déconnecté, puis reconnecté à la ligne de commande **composer** à partir du même shell.

Si aucune valeur n'est attribuée à la variable **TWS_SESSION**, la valeur par défaut identifiant la session est définie comme suit :

- Si vous utilisez **composer** en mode de traitement par lots, la valeur par défaut est le *nom_utilisateur* utilisé par l'utilisateur lorsqu'il se connecte au gestionnaire de domaine maître.
- Si vous utilisez **composer** en mode interactif, la valeur par défaut correspond à une chaîne alphanumérique que le produit a automatiquement créée.

Remarque : Dans la base de données, le *nom_utilisateur* de l'utilisateur verrouillant une définition d'objet est enregistré en majuscules.

Exemples

Pour verrouiller l'agenda Holidays, exécutez la commande suivante :

```
lock calendar=HOLIDAYS
```

Voir aussi

Dans Dynamic Workload Console, les objets sont automatiquement verrouillés tant que vous, ou un autre utilisateur, ne les aurez pas ouverts à l'aide du bouton **Editer**. Les objets ne sont pas verrouillés si vous, ou un autre utilisateur, les ouvrez avec le bouton **Afficher**.

modify

Permet de modifier ou d'ajouter des objets de planification. Lorsque vous modifiez des objets, la commande **modify** extrait uniquement les objets qui peuvent être verrouillés par l'utilisateur en cours.

Autorisation

Vous devez avoir le droit d'accès **add** pour ajouter un nouvel objet de planification. Si l'objet existe déjà dans la base de données, vous devez avoir le droit d'accès **modify** sur cet objet ; sinon, le compositeur ne parvient pas à trouver les objets.

Pour **modifier** les objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{modify | m}
[calendars | calendar | cal=nom_agenda] |
[eventrule | erule | er=nom_règle_événement] |
[parms | parm | vb=[nom_table.]nom_variable] |
[table | vt=nom_table] |
[prompts | prom=nom_invite] |
[resources | resource | res=[poste_de_travail#]nom_ressource] |
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |
[cpu={nom_poste_travail | nom_classe_poste_de_travail | nom_domaine}]
[wat=nom_modèle_application_charge_travail]
[workstation | ws=nom_poste_travail] |
[workstationclass | wscl=nom_classe_poste_de_travail] |
[domain | dom=nom_domaine] |
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux
    [valid from date | valid to date | valid in date date]
    [:full]] |
[users |
user=[nom_poste_travail#]nom_utilisateur] |
[accesscontrollist | acl
for nom_domaine_sécurité] |
[securitydomain | sdom=nom_domaine_sécurité] |
[securityrole | srol=nom_rôle_sécurité]}
```

Arguments

calendars | calendar | cal

Si aucun argument ne suit, permet de modifier toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet de modifier l'agenda *nom_agenda*. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de modifier toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événement* suit, permet de modifier la règle d'événement *nom_règle_événement*. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de modifier toutes les définitions de variable globale trouvées dans la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet de modifier la variable spécifiée de la table *nom_table*. Si *nom_table* est ignoré, **composer** recherche la définition de variable *nom_variable* dans la table de variables par défaut. Les caractères génériques sont utilisés dans *nom_table* et *nom_variable*. Par exemple :

```
modify parms=@.@"
```

Modifie toutes les variables de toutes les tables.

```
modify parms=@"
```

Modifie toutes les variables de la table par défaut.

```
modify parms=@.acct@"
```

Modifie toutes les variables dont le nom commence par acct dans toutes les tables existantes.

A faire : La modification ou l'ajout d'une variable permet de verrouiller la table de variables qui la contient. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle ou sur les variables qu'elle contient.

variable | vt

Si aucun argument ne suit, permet de modifier toutes les définitions de table de variables.

Si la table de variables *nom_table* de l'argument suit, permet de modifier la table de variables *nom_table*. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet de modifier toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet de modifier l'invite *nom_invite*. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de modifier toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de modifier la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de modifier toutes les définitions de groupe de cycle d'exécution.

Si l'argument *runcyclegroupname* suit, permet de modifier le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu Modifie les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

wat Si aucun argument ne suit, permet de modifier toutes les définitions de modèle d'application de charge de travail.

Si l'argument *workloadapplicationtemplatename* suit, permet de modifier le modèle d'application de charge de travail spécifié. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de modifier toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet de modifier le poste de travail *nom_poste_travail*. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de modifier toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet de modifier le domaine *nom_domaine*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de modifier toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_de_travail* suit, permet de modifier la classe de poste de travail *nom_classe_poste_de_travail*. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet de modifier toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de modifier le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet de modifier toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de modifier le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_travail* est ignoré, la valeur

par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

full Modifie toutes les définitions de travail contenues dans le flot de travaux.

users | user

Si aucun argument ne suit, permet de modifier toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de modifier l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de modifier les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de modifier les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de modifier toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de modifier la définition de domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de modifier toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet de modifier la définition de rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

Commentaires

La commande **modify** effectue la séquence d'actions suivante :

1. Verrouille les objets dans la base de données.
2. Copie les définitions d'objet dans un fichier temporaire.
3. Modifie le fichier.

4. Retransmet la définition contenue dans le fichier temporaire dans la base de données.
5. Si la commande **modify** échoue sur un sous-ensemble d'objets sélectionnés, **composer** demande *"Voulez-vous la rééditer ?"* Le fichier enregistré auparavant est rouvert pour être modifié et les étapes suivantes de la séquence sont répétées.
6. Déverrouille les objets dans la base de données.

Les définitions de règle d'événement sont ouvertes avec un éditeur XML (voir «Définition de règle d'événement», à la page 291 pour la référence XML et voir «Editeur Composer», à la page 323 pour des détails sur la configuration d'un éditeur XML).

Si vous modifiez avec la même commande **modify** deux objets ou plus liés par une relation (par exemple un travail remplaçant et le travail remplacé), l'ordre dans lequel les objets sont répertoriés dans le fichier temporaire a parfois son importance pour la réussite de la commande **modify**. Ceci se produit parce que la commande **modify** lit en séquence les objets contenus dans le fichier temporaire ; par conséquent si l'objet référençant est affiché avant l'objet référencé la commande **modify** peut échouer sur l'objet référençant.

Par exemple, si la commande :

```
modify FTA1#@PROVA
```

produit le fichier temporaire suivant :

```
SCHEDULE FTA1#PROVA VALIDFROM 08/31/2005
MATCHING SAMEDAY
:
FTA2#MY-JOB
FOLLOWS FTA1#COPYOFPROVA.MY-JOB06
END
```

```
SCHEDULE FTA1#COPYOFPROVA VALIDFROM 08/31/2005
MATCHING SAMEDAY
:
FTA1#MY-JOB06
FIN
```

et vous changez le nom du travail prédécesseur de FTA1#MY-JOB06 à FTA1#MY-JOB05 dans les deux flots de travaux FTA1#PROVA et FTA1#COPYOFPROVA, puis la commande **modify** :

1. essaie d'abord de changer la définition du flot de travaux FTA1#PROVA. Elle échoue parce qu'elle trouve une dépendance de prédécesseur/successeur provenant d'un travail FTA1#MY-JOB05 qui est encore inconnu.
2. elle essaie ensuite de modifier la définition de FTA1#COPYOFPROVA et elle réussit.

A la seconde exécution de **modify** pour faire passer le nom du travail précédent de FTA1#MY-JOB06 à FTA1#MY-JOB05 dans le flot de travaux FTA1#PROVA, la commande est exécutée correctement car le travail précédent FTA1#MY-JOB05 existe maintenant dans la base de données.

Si le flot de travaux FTA1#COPYOFPROVA a été répertorié dans le fichier temporaire avant FTA1#PROVA, alors la commande **modify** s'exécute correctement la première fois parce que le nom du travail prédécesseur a été modifié avant de changer la définition de dépendance dans le travail successeur.

Pour les définitions d'utilisateur, si la zone de mot de passe conserve la valeur "*****" lorsque vous quittez l'éditeur, l'ancien mot de passe est conservé. Pour définir un mot de passe vide, insérez deux guillemets ("").

La commande `modify` vérifie la présence de dépendances de boucle à l'intérieur des flots de travaux. Par exemple, si `job1` suit `job2` et si `job2` suit `job1`, il y a une dépendance de boucle. Lorsqu'une dépendance de boucle est détectée à l'intérieur d'un flot de travaux, une erreur est affichée. La commande `modify` ne vérifie pas les dépendances de boucle entre les flots de travaux parce que selon la complexité des activités de planification, cette vérification pourrait prendre trop de temps et de ressources d'UC.

Exemples

Pour modifier tous les agendas, exécutez la commande suivante :

```
modify calendars=@
```

Pour modifier le flot de travaux `sked9` qui a été lancé sur le poste de travail `site1`, exécutez la commande suivante :

```
m sched=site1#sked9
```

Pour modifier toutes les règles d'événement comprenant une action appliquée à l'objet `DPJOB10`, exécutez la commande :

```
mod er=@;filter job=DPJOB10
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour modifier des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section Edition de définitions de poste de travail.
- Pour modifier des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Edition d'une règle d'événement.
- Pour modifier des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour modifier tous les autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

new

Permet d'ajouter une nouvelle définition d'objet dans la base de données.

Autorisation

Vous devez avoir le droit d'accès **add** pour ajouter un nouvel objet de planification. Si l'objet existe déjà dans la base de données, vous devez avoir le droit d'accès **modify** à l'objet.

Pour **ajouter** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
new  
[calendar |  
domain |  
eventrule |  
job |  
jobstream |  
parameter |  
prompt |  
resource |  
runcyclegroup |  
user |  
variable |  
wat |  
workstation |  
workstationclass |  
accesscontrollist |  
securitydomain |  
securityrole]
```

Arguments

Objet que vous souhaitez définir : un agenda, un domaine, une règle d'événement, un travail, un flot de travaux, une variable, une invite, une ressource, un utilisateur, une table de variables, un modèle d'application de charge de travail, un poste de travail ou une classe de poste de travail.

Commentaires

La commande ouvre un modèle prédéfini qui vous aide à modifier la définition d'objets et l'ajoute à la base de données lorsque vous l'enregistrez.

Les modèles d'objets sont situés dans le sous-dossier templates du répertoire d'installation IBM Workload Scheduler. Ils peuvent être personnalisés pour tenir compte de vos préférences.

Les définitions de règle d'événement sont ouvertes avec un éditeur XML (voir «Définition de règle d'événement», à la page 291 pour la référence XML et voir «Editeur Composer», à la page 323 pour des détails sur la configuration d'un éditeur XML).

Lors de la création d'une variable, la table de variables de destination est verrouillée. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle.

Exemples

Pour créer une nouvelle définition d'utilisateurs, exécutez la commande :

```
new user
```

Pour créer une nouvelle définition d'invite, exécutez la commande :

```
new prompt
```

Pour créer une nouvelle définition de règle d'événement, exécutez la commande :
new erule

Pour créer une définition de table de variables, exécutez la commande suivante :
new vartable

Pour créer une définition de variable, exécutez la commande suivante :
new parameter

Pour créer deux modèles d'application de charge de travail, WAT_NAME1 et WAT_NAME2, exécutez :

```
new wat
BAPPLICATION WAT_NAME1
  DESCRIPTION "Description"
  VENDOR "Provider"
  JSTREAMS
    FTA1#JS_1_1
    AGENT1#JS_1_2
END

BAPPLICATION WAT_NAME2
  DESCRIPTION "Description"
  VENDOR "Provider"
  JSTREAMS
    JS_2_1
    JS_2_2
END
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

le manuel *Dynamic Workload Console - Guide d'utilisation*.

- Pour créer des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section "Création de postes de travail distribués".
- Pour créer des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Création d'une règle d'événement.
- Pour créer des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour créer tous les autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

print

Il s'agit d'un synonyme de la commande **display** utilisé avec l'option **;offline**. Voir «display», à la page 345 pour plus de détails.

redo

Modifie et exécute à nouveau la commande précédente.

Remarque : Si la commande précédente était **authenticate**, la commande **redo** n'affiche pas le mot de passe défini.

Autorisation

Tout utilisateur autorisé à exécuter composer est autorisé à émettre cette commande.

Syntaxe

{redo | red}

Contexte

Si vous exécutez la commande **redo**, le programme composer affiche la dernière commande que vous pouvez modifier et réexécuter à votre guise. Utilisez la barre d'espacement pour placer le curseur sous le caractère à modifier, puis entrez les instructions ci-dessous.

Instructions

- d[rép]** Supprime le caractère situé au-dessus de la lettre **d**. Cette commande peut être suivie par d'autres instructions.
- i***texte* Insère du texte avant le caractère situé au-dessus de la lettre **i**.
- r***texte* Remplace un ou plusieurs caractères par du texte en commençant par le caractère au-dessus du **r**. Le remplacement est implicite si aucune autre instruction n'est entrée.
- >***texte* Ajoute du texte à la fin de la ligne.
- >d[rép | *texte*]**
Supprime les caractères en fin de ligne. Cette commande peut être suivie d'une autre instruction ou d'un autre texte.
- >r***texte* Remplace des caractères à la fin de la ligne par du texte.

Exemples d'instruction

- ddd** Supprime les trois caractères au-dessus des **d**.
- iabc** Insère **abc** avant le caractère situé au-dessus de la lettre **i**.
- rabc** Remplace les trois caractères, en commençant par celui qui est situé au-dessus du **r**, par **abc**.
- abc** Remplace les trois caractères situés au-dessus de **abc**, par **abc**.
- d diabc**
Supprime le caractère au-dessus du premier **d**, ignore un caractère, supprime le caractère au-dessus du second **d**, puis le remplace par **abc**.
- >abc** Ajoute **abc** à la fin de la ligne.
- >ddabc**
Supprime les deux derniers caractères de la ligne, puis les remplace par **abc**.
- >rabc** Remplace les trois derniers caractères de la ligne par **abc**.

Exemples

Pour insérer un caractère, exécutez la commande suivante :

```
redo
display site1#sa@
    ip
display site1#sa@
```

Pour remplacer trois caractères, par exemple, pour remplacer site par serv en remplaçant ite par erv, exécutez la commande suivante :

```
redo
display site1#sa@
    erv
display serv1#sa@
```

rename

Permet de renommer un objet de planification existant dans la base de données. Le nouveau nom ne doit pas identifier un objet déjà défini dans la base de données.

Autorisation

Pour **renommer** des objets de planification, vous devez détenir l'accès *delete* à l'objet avec l'ancien nom et l'accès *add* avec le nouveau nom.

Pour **renommer** des objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{rename | rn}
{calendars | calendar | cal |
parms | parm | vb |
    vartable | vt |
prompts | prom |
resorces | resource | res |
runcyclegroup | rcg |
workstation | ws |
workstationclass | wscl |
domain | dom |
jobs | jobdefinition | jd |
jobsched | jb |
eventrule | erule | er |

sched | jobstream | js |
securitydomain | sdom |
securityrole | srol |
users | user }
ancien_identificateur_objet  nouvel_identificateur_objet
```

Arguments

ancien_identificateur_objet

Spécifie l'ancienne clé externe qui identifie l'objet de planification, par exemple le nom d'agenda cal1 comme identificateur d'un objet d'agenda défini pour être renommé.

nouvel_identificateur_objet

Spécifie la nouvelle clé externe qui identifie l'objet de planification, par exemple le nom d'agenda cal2 comme le nouvel identificateur à attribuer à l'objet d'agenda auparavant nommé cal1.

En ce qui concerne les travaux, flots de travaux, ressources et utilisateurs, les valeurs *ancien_identificateur_objet* et *nouvel_identificateur_objet* possèdent les formats suivants :

[nom_poste_travail#]nom_travail

La commande s'applique à cette définition de travail. Ce format est utilisé avec la clé **jobs** | **jobdefinition** | **jd**.

[nom_poste_travail#]nom_flot_travaux

La commande s'applique à toutes les versions de ce flot de travaux. Ce format est utilisé avec la clé **sched** | **jobstream** | **js**.

[nom_poste_travail#]nom_flot_travaux.nom_travail

La commande s'applique à cette instance de travail définie dans ce flot de travaux. Voir le mot clé **js** dans la syntaxe «Définition de flot de travaux», à la page 226 pour des détails supplémentaires. Ce format est utilisé avec la clé **jobsched** | **jb**.

[nom_poste_travail#]nom_ressource

La commande s'applique à cette définition de ressource. Ce format est utilisé avec la clé **resources** | **resource** | **res**.

[nom_poste_travail#][domaine\]nom_utilisateur

La commande s'applique à cette définition d'utilisateur. Ce format est utilisé avec la clé **users** | **user**.

En ce qui concerne les variables (paramètres globaux) :

ancien_identificateur_objet

Doit être spécifié au format *nom_table.nom_variable*. Si *nom_table* est ignoré, composer recherche la définition de variable dans la table de variables par défaut.

nouvel_identificateur_objet

Doit être spécifié au format *nom_variable*. L'ajout du nom de table ici génère une erreur.

Commentaires

Pour pouvoir être renommé, l'objet doit être déverrouillé ou verrouillé par l'utilisateur qui exécute la commande **rename**.

La table de variables contenant la variable est verrouillée, alors que la variable est renommée. Cela implique que, lorsque la table est verrouillée, aucun autre utilisateur ne peut exécuter d'autres commandes de verrouillage sur elle.

Si l'objet nommé comme dans la zone *ancien_identificateur_objet* n'existe pas dans la base de données, un message d'erreur s'affiche.

L'utilisation de caractères génériques n'est pas autorisée avec cette commande.

Lorsque *nom_poste_travail* n'est pas spécifié pour les objets qui ont le nom du poste de travail dans leur identificateur d'objet (par exemple définitions de travail ou de flot de travaux), le planificateur utilise l'un des éléments suivants pour *nom_poste_travail* :

- Le poste de travail par défaut spécifié dans le fichier *localopts*
- Le gestionnaire de domaine maître si le programme de ligne de commande **composer** s'exécute sur un noeud à l'extérieur du réseau IBM Workload Scheduler. Dans ce cas en effet, le poste de travail par défaut défini dans le fichier *localopts* est le gestionnaire de domaine maître.

La commande **rename** est utilisée pour attribuer les nouveaux noms aux objets déjà existants dans la base de données. Le nouveau nom attribué à l'objet est immédiatement utilisé dans la base de données alors qu'il est utilisé dans le plan après la réexécution du script **JnextPlan**. Cela peut entraîner des absurdités lors de la soumission des travaux adhoc avant la régénération du plan de production.

Exemples

Pour remplacer le nom de l'objet de domaine DOMAIN1 par DOMAIN2, exécutez la commande suivante :

```
rename dom=DOMAIN1 DOMAIN2
```

Pour renommer le flot de travaux de LABJST1 en LABJST2 sur le poste de travail CPU1, exécutez la commande suivante :

```
rename js=CPU1#LABJST1 CPU1#LABJST2
```

Pour renommer la variable ACCTOLD (définie dans la table ACCTAB) en ACCTNEW, exécutez la commande suivante :

```
rename parm=ACCTAB.ACCTOLD ACCTNEW
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour renommer des postes de travail, voir
Dynamic Workload Console - Guide d'utilisation, section Edition de définitions de poste de travail.
- Pour renommer des règles d'événement, voir
Dynamic Workload Console - Guide d'utilisation, section Edition d'une règle d'événement.
- Pour renommer les domaines de sécurité et les rôles de sécurité, voir
Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.
- Pour renommer tous les autres objets, voir
Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

replace

Remplace les définitions d'objets de planification dans la base de données.

Autorisation

Vous devez avoir le droit d'accès **add** pour ajouter un nouvel objet de planification. Si l'objet existe déjà dans la base de données, vous devez avoir les droits suivants :

- droit d'accès **modify** sur l'objet si ce dernier n'est pas verrouillé.
- droits d'accès **modify** et **unlock** aux objets si vous souhaitez utiliser l'option **unlock** sur des objets verrouillés par d'autres utilisateurs.

Pour **remplacer** les objets de sécurité, vous devez détenir des droits pour exécuter l'action **modify** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{replace | rep} nom_fichier [;unlock]
```

Arguments

nom_fichier

Indique le nom d'un fichier qui contient les définitions d'objet à remplacer. Le fichier peut contenir tous les types de définition d'objet de planification.

unlock

Permet de mettre à jour les objets auparavant verrouillés, puis de les déverrouiller. Une erreur s'affiche si les objets n'étaient pas auparavant verrouillés. Pour tous les nouveaux objets insérés, cette option est ignorée.

Commentaires

La commande **replace** ressemble à la commande **add** mais elle n'affiche aucune invite de confirmation du remplacement des objets existants. Pour plus d'informations, voir «add», à la page 338.

La commande **replace** vérifie la présence de dépendances de boucle à l'intérieur des flots de travaux. Par exemple, si job1 suit job2 et si job2 suit job1, il y a une dépendance de boucle. Lorsqu'une dépendance de boucle à l'intérieur d'un flot de travaux est détectée, une erreur est affichée.

La commande **replace** ne vérifie pas les dépendances de boucle entre les flots de travaux parce que selon la complexité des activités de planification, cette vérification pourrait prendre trop de temps. Si vous devez vérifier les dépendances de boucle entre les flots de travaux, utilisez l'utilitaire **datamigrate**.

Lorsque vous effectuez une mise à niveau à partir d'instances de IBM Workload Scheduler version 8.3 ou ultérieure, vous pouvez vérifier les dépendances de boucle entre les flots de travaux en procédant comme suit :

1. Extrayez les définitions d'objet à partir de la base de données sur une instance de IBM Workload Scheduler dans un fichier à plat à l'aide de la ligne de commande **composer extract**.
2. Importez le fichier dans la base de données sur le gestionnaire de domaine maître à l'aide de l'utilitaire **datamigrate**.

datamigrate importe les données enregistrées dans le fichier et vérifie les dépendances de boucle à l'intérieur des flots de travaux.

Pour plus d'informations sur l'utilitaire **datamigrate**, voir «datamigrate», à la page 689.

Exemples

Pour remplacer les travaux du fichier `myjobs`, exécutez la commande suivante :

```
replace myjobs
```

Pour remplacer toutes les ressources par celles du fichier `myres`, exécutez la commande suivante :

```
rep myres
```

Vous souhaitez modifier certaines définitions de règles d'événement existantes dans la base de données. Vous souhaitez également ajouter certaines nouvelles définitions. Vous utilisez cette commande de la façon suivante :

1. Vous écrivez les définitions complètes dans un fichier XML que vous nommez `2007rules.xml`.
2. Vous exécutez :

```
rep 2007rules.xml
```

Commande système

Exécute une commande système.

Syntaxe

`[ : | ! ] commande-système`

Arguments

commande système

Désigne une commande système valide. Le préfixe deux-points (:) ou le point d'exclamation (!) sont requis uniquement lorsque la commande est orthographiée comme une commande Composer.

Exemples

Pour exécuter une commande **ps** sous UNIX, exécutez la commande suivante :

```
ps -ef
```

Pour exécuter une commande **dir** sous Windows, exécutez la commande suivante :

```
dir \bin
```

unlock

Permet de déverrouiller l'accès aux objets de planification dans la base de données. Par défaut, pour déverrouiller un objet, ce dernier doit avoir été verrouillé par le même utilisateur lors de la même session.

Autorisation

Vous devez disposer des droits d'accès **unlock** pour déverrouiller les objets de planification verrouillés par d'autres utilisateurs.

Pour **déverrouiller** les objets de sécurité, vous devez détenir des droits pour exécuter l'action **unlock** sur le type d'objet **file** avec l'attribut **name=security**.

Syntaxe

```
{unlock | u}  
[{calendars | calendar | cal=nom_agenda] |  
[eventrule | erule | er=nom_règle_événmt] |  
[parms | parm | vb=[nom_table.]nom_variable] |  
[variable | vt=nom_table] |  
[prompts | prom=nom_invite] |  
[resources | resource | res=[poste_de_travail#]nom_ressource] |  
[runcyclegroup | rcg=nom_groupe_cycle_exécution] |  
[cpu={nom_poste_travail | nom_classe_poste_de_travail | nom_domaine}]  
[workstation | ws=nom_poste_travail] |  
[workstationclass | wscl=nom_classe_poste_de_travail] |  
[domain | dom=nom_domaine] |  
[jobs | jobdefinition | jd=[nom_poste_travail#]nom_travail] |  
[sched | jobstream | js= [nom_poste_travail#]nom_flot_travaux  
  [début de validité date | fin de validité date | valid  
  in date date]] |  
[users |  
  user=[nom_poste_travail#]nom_utilisateur] |  
[accesscontrollist | acl  
  for nom_domaine_sécurité] |  
[securitydomain | sdom=nom_domaine_sécurité] |  
[securityrole | srol=nom_rôle_sécurité}  
[:forced]
```

Arguments

calendars | calendar | cal

Si aucun argument ne suit, permet de déverrouiller toutes les définitions d'agenda.

Si l'argument *nom_agenda* suit, permet de déverrouiller l'agenda *nom_agenda*. Les caractères génériques sont acceptés.

eventrule | erule | er

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de règle d'événement.

Si l'argument *nom_règle_événmt* suit, permet de déverrouiller la règle d'événement *nom_règle_événmt*. Les caractères génériques sont acceptés.

parms | parm | vb

Si aucun argument ne suit, permet de déverrouiller la table de variables par défaut.

Si l'argument *nom_table.nom_variable* suit, permet de déverrouiller toute la table contenant la variable *nom_variable*. Si *nom_table* est ignoré, permet de déverrouiller la table de variables par défaut. Les caractères génériques sont utilisés dans *nom_table* et *nom_variable*. Par exemple :

```
unlock parms=@.@
```

Déverrouille toutes les tables.

```
unlock parms=@
```

Déverrouille la table par défaut.

```
unlock parms=@.acct@
```

Permet de déverrouiller toutes les tables contenant les variables dont le nom commence par acct.

unlock parms=acct@

Déverrouille la table par défaut.

A faire : L'action effectuée sur une seule variable permet de déverrouiller la table de variables la contenant.

variable | vt

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de table de variables.

Si la table de variables *nom_table* de l'argument suit, permet de déverrouiller la table de variables *nom_table*. Les caractères génériques sont acceptés.

prompts | prom

Si aucun argument ne suit, permet de déverrouiller toutes les définitions d'invite.

Si l'argument *nom_invite* suit, permet de déverrouiller l'invite *nom_invite*. Les caractères génériques sont acceptés.

resources | resource | res

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de ressource.

Si l'argument *nom_poste_travail#nom_ressource* suit, permet de déverrouiller la ressource *nom_ressource* du poste de travail *nom_poste_travail* sur lequel la ressource est définie. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_ressource*.

runcyclegroup | rcg

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de groupe de cycle d'exécution.

Si l'argument *runcyclegroupname* suit, permet de déverrouiller le groupe de cycle d'exécution *runcyclegroupname*. Les caractères génériques sont acceptés.

cpu

Déverrouille les postes de travail, les classes de postes de travail ou les domaines.

Poste de travail

Nom du poste de travail. Les caractères génériques sont acceptés.

nom_classe_poste_de_travail

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

domaine

Nom du domaine. Les caractères génériques sont acceptés.

workstation | ws

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de poste de travail.

Si l'argument *nom_poste_travail* suit, permet de déverrouiller le poste de travail *nom_poste_travail*. Les caractères génériques sont acceptés.

domain | dom

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de domaine.

Si l'argument *nom_domaine* suit, permet de déverrouiller le domaine *nom_domaine*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de classe de poste de travail.

Si l'argument *nom_classe_poste_de_travail* suit, permet de déverrouiller la classe de poste de travail *nom_classe_poste_de_travail*. Les caractères génériques sont acceptés.

jobs | jobdefinition | jd

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de travail.

Si l'argument *nom_poste_travail#nom_travail* suit, permet de déverrouiller le travail *nom_travail* du poste de travail *nom_poste_travail* sur lequel le travail s'exécute. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_travail*.

sched | jobstream | js

Si aucun argument ne suit, permet de déverrouiller toutes les définitions de flot de travaux.

Si l'argument *nom_poste_travail#nom_flot_travaux* suit, permet de déverrouiller le flot de travaux *nom_flot_travaux* du poste de travail *nom_poste_travail* sur lequel le flot de travaux est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_flot_travaux*.

valid from

date Limite la sélection aux flots de travaux qui ont une date *valid from* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid to

date Limite la sélection aux flots de travaux qui ont une date *valid to* égale à la valeur indiquée. Le format est *mm/jj/aaaa*.

valid in

date date Intervalle de temps d'exécution du flot de travaux. Le format est *mm/jj/aaaa - mm/jj/aaaa*. L'une des deux dates peut être représentée par @.

users | user

Si aucun argument ne suit, permet de déverrouiller toutes les définitions d'utilisateur.

Si l'argument *nom_poste_travail#nom_utilisateur* suit, permet de déverrouiller l'utilisateur *nom_utilisateur* du poste de travail *nom_poste_travail* sur lequel l'utilisateur est défini. Si *nom_poste_travail* est ignoré, la valeur par défaut est le poste de travail sur lequel **composer** est exécuté. Les caractères génériques sont acceptés pour *nom_poste_travail* et *nom_utilisateur*.

accesscontrollist | acl

Si aucun argument *nom_domaine_sécurité* ne suit, permet de déverrouiller les définitions de listes de contrôle d'accès pour tous les domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de déverrouiller les définitions de listes de contrôle d'accès pour le domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securitydomain | sdom

Si aucun argument *nom_domaine_sécurité* ne suit, permet de déverrouiller toutes les définitions de domaines de sécurité.

Si un argument *nom_domaine_sécurité* suit, permet de déverrouiller la définition de domaine de sécurité *nom_domaine_sécurité*. Les caractères génériques sont admis pour *nom_domaine_sécurité*.

securityrole | srol

Si aucun argument *nom_rôle_sécurité* ne suit, permet de déverrouiller toutes les définitions de rôles de sécurité.

Si un argument *nom_rôle_sécurité* suit, permet de déverrouiller la définition du rôle de sécurité *nom_rôle_sécurité*. Les caractères génériques sont admis pour *nom_rôle_sécurité*.

forced Autorise l'utilisateur qui a verrouillé l'objet à le déverrouiller, quelle que soit la session.

Si cette option est utilisée par le *superutilisateur*, la commande **unlock** peut fonctionner quels que soient l'utilisateur et la session utilisés pour verrouiller l'objet.

Commentaires

Si un utilisateur autre que le *superutilisateur* essaie de déverrouiller un objet verrouillé par un autre utilisateur, un message d'erreur s'affiche.

Exemples

Pour déverrouiller la définition de travail JOBDEF1, exécutez la commande suivante :

```
unlock jd=@#JOBDEF1
```

Pour déverrouiller la définition de règle d'événement ERJS21, exécutez la commande suivante :

```
unlock erule=ERJS21
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour déverrouiller des postes de travail, voir *Dynamic Workload Console - Guide d'utilisation*, section Edition de définitions de poste de travail.
- Pour déverrouiller des règles d'événement, voir

Dynamic Workload Console - Guide d'utilisation, section Edition d'une règle d'événement.

- Pour déverrouiller des listes de contrôle d'accès, des domaines de sécurité et des rôles de sécurité, voir

Dynamic Workload Console - Guide d'utilisation, section relative à la gestion de sécurité de charge de travail.

- Pour déverrouiller tous les autres objets, voir

Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

update

Modifie les paramètres d'attribut d'un type d'objet de planification spécifique qui est défini dans la base de données sans utiliser les commandes **modify** ou **replace**. Vous pouvez mettre à jour certains paramètres d'attribut pour le type d'objet spécifique dans la base de données sans ouvrir ni modifier la définition d'objet dans un éditeur de texte, ou remplacer la définition d'objet en utilisant le nouveau contenu qui est enregistré dans un fichier texte.

Autorisation

Vous devez disposer du droit d'accès *modify* et *display* pour mettre à jour les propriétés d'objet, sinon **composer** sera incapable de mettre à jour les attributs de l'objet.

Syntaxe

```
{update | up}
{[cpu={nom_poste_de_travail | nom_classe_poste_de_travail}] |
[workstation | ws=nom_poste_travail] |
[workstationclass | wscl=nom_classe_poste_de_travail];
[filter critères_filtre_poste_de_travail= selection [...]];
set [ignore= on | off]]
[:noask]
```

Arguments

UC Met à jour les postes de travail ou les classes de poste de travail.

workstationname

Nom du poste de travail. Les caractères génériques sont acceptés.

workstationclassname

Nom de la classe de poste de travail. Les caractères génériques sont acceptés.

Remarque : La commande ne met pas à jour les domaines.

workstation | ws

Si aucun argument ne suit, permet de mettre à jour toutes les définitions de poste de travail.

Si l'argument *workstationname* suit, permet de mettre à jour le poste de travail *workstationname*. Les caractères génériques sont acceptés.

workstationclass | wscl

Si aucun argument ne suit, permet de modifier toutes les définitions de classe de poste de travail.

Si l'argument *workstationclassname*, met à jour la classe de poste de travail *workstationclassname*. Les caractères génériques sont acceptés.

filter Les critères de filtrage des objets de planification de poste de travail à utiliser.

Pour plus d'informations sur les critères de filtrage des objets de planification de poste de travail, voir tableau 52, à la page 328.

set Le critère set est obligatoire et introduit la liste des paramètres d'attribut d'objet à modifier.

ignore=on | off

L'attribut ignore pour une classe de poste de travail, une UC et un poste de travail. Si vous indiquez on, le poste de travail ou la classe de poste de travail est défini sur ignore dans la base de données, et le poste de travail ou tous les postes de travail appartenant à une classe de poste de travail sont exclus du plan de production suivant. Si vous indiquez off, le poste de travail ou tous les postes de travail appartenant à une classe de poste de travail sont inclus du plan de production suivant.

Commentaires

Remarque : Les modifications apportées à la définition de l'objet de planification ne s'appliquent que dans la base de données. Pour appliquer ces modifications au plan, vous devez attendre le plan de production suivant (le flot de travaux Final qui s'exécute) ou vous pouvez exécuter la commande **JnextPlan**.

Exemples

Si vous voulez que tous les postes de travail dont le nom commence par AB soient inclus dans le plan de production suivant et que vous souhaitez mettre à jour l'attribut ignore sur ON dans leur définition, exécutez la commande suivante :

```
update ws=AB@;set ignore=on
```

Si vous voulez que tous les postes de travail du réseau soient inclus dans le plan de production suivant et que vous devez mettre à jour l'attribut ignore sur OFF dans la définition de tous les postes de travail de votre base de données, sans répondre à l'invite de confirmation émise par **composer**, exécutez la commande suivante :

```
update ws;set ignore=off; noask
```

Si vous voulez que tous les postes de travail appartenant à la classe de poste de travail *New York* soient exclus du plan de production suivant et que vous devez mettre à jour l'attribut ignore sur ON dans leur définition, exécutez la commande suivante :

```
update wscl=NewYork;set ignore=on
```

Si vous voulez que tous les postes de travail du domaine DOMWIN1 soient inclus dans le plan de production suivant et que vous devez mettre à jour l'attribut ignore sur OFF dans la définition de tous les postes de travail de votre base de données, sans répondre à l'invite de confirmation émise par **composer**, exécutez la commande suivante :

```
update ws;filter domain=DOMWIN1; set ignore=off; noask
```

Si vous voulez que tous les postes de travail de la table de variable MAIN_TABLE soient inclus dans le plan de production suivant et que vous devez mettre à jour

l'attribut ignore sur ON dans la définition de tous les postes de travail de votre base de données, exécutez la commande suivante :

```
update ws=@; filter vartable=MAIN_TABLE domain=MASTER@; set ignore=on
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer les mêmes tâches que celles décrites dans :

Dynamic Workload Console - Guide d'utilisation.

- Pour supprimer des postes de travail, voir *Dynamic Workload Console - Guide d'utilisation*, section Edition de définitions de poste de travail.
- Pour supprimer des règles d'événement, voir *Dynamic Workload Console - Guide d'utilisation*, section Edition d'une règle d'événement.
- Pour supprimer tous les autres objets, voir *Dynamic Workload Console - Guide d'utilisation*, section Liste des définitions d'objet dans la base de données.

validate

Exécute la validation des définitions d'objets contenus dans un fichier utilisateur.

Autorisation

Vous n'avez pas besoin d'une autorisation spécifique sur les objets pour exécuter cette commande.

Syntaxe

```
{validate | val} nom_fichier [;syntax]
```

Arguments

nom_fichier

Désigne le nom d'un fichier qui contient des agendas, des postes de travail, des classes de postes de travail, des domaines, des travaux, des paramètres, des invites, des ressources, des flots de travaux, des règles d'événement ou des tables de variables. Pour les définitions de règles d'événement, le fichier doit être en XML. Voir «Définition de règle d'événement», à la page 291 pour plus de détails sur l'écriture des définitions de règles d'événement.

syntaxe

Détecte les erreurs de syntaxe dans le fichier.

Commentaires

La sortie de la commande **validate** peut être redirigée vers un fichier de la manière suivante :

```
composer "validate filename" > outfile
```

Pour insérer des messages d'erreur dans le fichier de sortie, utilisez la commande suivante :

```
composer "validate filename" > outfile 2>&1
```

Exemples

Pour vérifier la syntaxe d'un fichier contenant les définitions de poste de travail, exécutez la commande suivante :

```
validate mycpus;syntax
```

version

Affiche la bannière du programme de ligne de commande **composer**.

Autorisation

Tout utilisateur autorisé à exécuter composer est autorisé à émettre cette commande.

Syntaxe

```
{version | v}
```

Exemples

Pour afficher la bannière du programme de ligne de commande **composer**, exécutez la commande suivante :

```
version
```

ou :

```
v
```

Chapitre 11. Gestion des applications de charge de travail

Les applications de charge de travail peuvent être créées puis exportées de sorte à pouvoir être partagées avec d'autres environnements IBM Workload Scheduler. Dans le nouvel environnement, l'application de charge de travail peut être mise à jour, remplacée ou supprimée par la suite.

Le cycle de vie d'une application de charge de travail commence avec la définition du modèle d'application de charge de travail. Le modèle est ensuite exporté pour déploiement dans l'environnement cible.

Vous pouvez définir le modèle d'application de charge de travail à l'aide de la ligne de commande **composer** ou de Dynamic Workload Console.

Vous pouvez exporter le modèle application de charge de travail en utilisant la procédure disponible dans Dynamic Workload Console, comme indiqué dans «Création d'un modèle d'application de charge de travail», à la page 394. Vous pouvez également exporter une définition de flot de travaux à partir du Concepteur de charge de travail et la sauvegarder comme modèle d'application de charge de travail dans un fichier compressé, comme indiqué dans «Exportation d'une définition de flot de travaux comme modèle d'application de charge de travail», à la page 397.

Vous pouvez importer le modèle application de charge de travail en utilisant la procédure disponible dans Dynamic Workload Console, comme indiqué dans «Importation d'un modèle d'application de charge de travail», à la page 398.

Pour exporter et importer l'application de charge de travail, vous pouvez également utiliser la ligne de commande **wappman**, comme indiqué dans les rubriques ci-dessous.

Le processus d'exportation produit un fichier compressé contenant cinq fichiers :

*nom de l'application de charge de travail*_Definitions.UTF8.xml

Un fichier au format XML contenant une définition de tous les objets référencés dans l'application de charge de travail. Les définitions sont déployées dans l'environnement cible pour remplir la base de données cible avec les mêmes objets qui existent dans l'environnement source. N'écrivez pas ce fichier.

*nom de l'application de charge de travail*_Mapping.UTF8.properties

Fichier de mappage modifié par l'utilisateur cible, qui remplace les noms des objets dans l'environnement source par les futurs noms des objets dans l'environnement cible.

Lorsque le processus d'importation est exécuté à partir de la ligne de commande **wappman**, vous pouvez demander en option à ce que le fichier de mappage soit modifié automatiquement, selon des règles définies avec des expressions régulières et indiquées dans un des fichiers ad hoc suivants :

- *application de charge de travail*name_BasicRegExpMapping.UTF8.rules
- *application de charge de travail*name_AdvancedRegExpMapping.UTF8.rules

Ces fichiers sont générés par l'exportation et, s'ils sont utilisés, ils doivent être correctement personnalisés.

nom de l'application de charge de travail **SourceEnv_reference.txt**
Fichier contenant des informations de référence sur les postes de travail utilisés dans l'application de charge de travail ainsi que d'autres informations pouvant être utiles pour mapper correctement l'environnement source à l'environnement cible.

application de charge de travail **name_BasicRegExpMapping.UTF8.rules**
Fichier contenant des règles définies avec des expressions régulières simples et permettant de modifier le fichier de mappage. (Facultatif)
Personnalisez ce fichier en fonction des futurs noms des objets dans l'environnement cible. Le processus d'importation exécuté à partir de la ligne de commande **wappman** applique alors les règles définies pour modifier le fichier de mappage.

application de charge de travail **name_AdvancedRegExpMapping.UTF8.rules**
Fichier contenant des règles définies avec des expressions régulières avancées et permettant de modifier le fichier de mappage. (Facultatif)
Personnalisez ce fichier en fonction des futurs noms des objets dans l'environnement cible. Le processus d'importation exécuté à partir de la ligne de commande **wappman** applique alors les règles définies pour modifier le fichier de mappage.

Création d'un modèle d'application de charge de travail

Comment définir un modèle application de charge de travail à l'aide de Dynamic Workload Console.

Pour s'assurer que la solution d'automatisation de charge de travail puisse être facilement reproduite dans un autre environnement, il existe des valeurs recommandées à prendre en compte lors de la création du modèle application de charge de travail :

Définitions de travaux

Les travaux qui se rapportent aux éléments qui dépendent de l'environnement ou de la topologie dans lesquels ils se trouvent, comme les travaux de service Web, les travaux de transfert de fichier et les travaux de base de données, entre autres, devraient utiliser des variables lorsqu'ils indiquent ces éléments, tels que des accréditations, des chemins d'accès, et des numéros de port. Les variables peuvent être gérées dans le fichier de mappage de sorte que les valeurs correctes puissent être affectées à la variable.

Noms des postes de travail

Quand des travaux et des flots de travaux sont extraits de l'application de charge de travail pendant l'exportation, les noms des postes de travail sont extraits lorsqu'ils sont détectés dans l'environnement source. Les noms significatifs ou une convention de dénomination normalisée peuvent simplifier le procédé de mappage.

Utilisateurs

Les utilisateurs sont également extraits lorsqu'ils sont détectés dans l'environnement source. Si le même utilisateur n'est pas présent dans l'environnement source et cible, des variables devraient être utilisées pour indiquer l'utilisateur.

Fichier de mappage

Le fichier de mappage devrait être mis à jour après le processus d'importation. Dans ce cas, il peut être utile de remplacer une application

application de charge de travail ou de la mettre à jour en apportant les modifications nécessaires au fichier de mappage.

Table de variable du flot de travaux

Toutes les variables utilisées pour représenter génériquement des objets dans application de charge de travail devraient être ajoutées à une table de variable spécifique en relation au flot de travaux dans application de charge de travail. Ceci permet à la personnalisation du flot de travaux de refléter l'environnement cible via le fichier de mappage. Evitez d'associer la table de variable par défaut à un flot de travaux. La table de variable par défaut est extraite comme n'importe quelle autre table et devra être renommée, sinon le processus d'importation échouera car une table du même nom existe déjà. L'environnement cible a déjà une table de variable par défaut, MAIN_TABLE, définie.

Table de variables du cycle d'exécution

Toutes les variables utilisées pour représenter génériquement des objets dans application de charge de travail devraient être ajoutées à une table de variable spécifique en relation au cycle d'exécution dans application de charge de travail. Ceci permet à la personnalisation du cycle d'exécution de refléter l'environnement cible via le fichier de mappage. Evitez d'associer la table de variable par défaut à un cycle d'exécution. La table de variable par défaut est extraite comme n'importe quelle autre table et devra être renommée, sinon le processus d'importation échouera car une table du même nom existe déjà. L'environnement cible a déjà une table de variable par défaut, MAIN_TABLE, définie.

Dans le Concepteur de charge de travail, vous pouvez créer le modèle d'une charge de travail qui peut ensuite être importée et exécutée dans un autre environnement. Vous pouvez créer un modèle de application de charge de travail contenant un ou plusieurs flots de travaux avec tous les travaux connexes et les dépendances internes ou externes (telles que des fichiers, des ressources, des invites) afin d'avoir un flux autonome. Vous pouvez ensuite exporter le modèle de application de charge de travail pour le déployer et l'exécuter dans un autre environnement. Pour créer un modèle de application de charge de travail, procédez comme suit :

1. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de la charge de travail > Gestion des définitions de charge de travail**
2. Indiquez le nom d'un moteur réparti. Concepteur de charge de travail s'ouvre.
3. Dans le panneau Liste de travail, sélectionnez **Nouveau > Modèle d'application de charge de travail**. Le modèle de application de charge de travail est créé dans la vue de détails et sa page de propriétés s'affiche.
4. Dans le panneau des propriétés, spécifiez les attributs pour le modèle de application de charge de travail que vous créez :

Nom Zone obligatoire qui contient le nom du modèle de application de charge de travail. La longueur est limitée à 80 caractères.

Description

Texte descriptif facultatif pour aider les utilisateurs de application de charge de travail à comprendre l'objectif et les caractéristiques de application de charge de travail. La longueur est limitée à 120 caractères.

Fournisseur

Zone facultative qui indique le créateur du modèle de application de

charge de travail. Elle peut être utile pour faire savoir aux utilisateurs de application de charge de travail qui a créé le modèle et qui l'a fourni. La longueur est limitée à 120 caractères.

5. Dans la vue Détails, cliquez avec le bouton droit de la souris sur le modèle de application de charge de travail, puis cliquez sur **Ajouter un flot de travaux** pour ajouter des flots de travaux à celui-ci.
6. Dans la boîte de dialogue de recherche, sélectionnez les flots de travaux que vous souhaitez ajouter. En même temps que les flots de travaux, les dépendances correspondantes sont également automatiquement ajoutées au modèle de application de charge de travail.
7. Cliquez sur **Sauvegarder** pour sauvegarder le modèle de application de charge de travail dans la base de données.
8. Cliquez avec le bouton droit de la souris le modèle de application de charge de travail et cliquez sur **Exporter** pour produire un fichier compressé, nommé *nom de modèle application de charge de travail.zip*, contenant tous les fichiers et informations requises pour permettre à la charge de travail de s'exécuter également dans un autre environnement.

Le fichier compressé contient :

Nom de modèle application de charge de travail **Definitions.UTF8.xml**

Fichier XML qui contient les définitions de tous les objets exportés. Ces définitions seront déployées dans l'environnement cible pour remplir la base de données cible avec les mêmes objets qui existent dans l'environnement source. Les objets dans le fichier de définition peuvent rester tel quel ou vous pouvez les renommer. Si un objet n'a pas de définition dans le fichier de définition, par exemple, un poste de travail, alors, au moment de l'importation, l'objet correspondant ne sera pas créé dans l'environnement cible. Cet objet doit déjà être présent dans l'environnement cible, donc, pour ces types d'objets, vous devez les mapper dans le fichier de mappage.

Nom de modèle application de charge de travail **Mapping.UTF8.properties**

Fichier de mappage utilisé pour remplacer les noms des objets dans l'environnement source avec les noms de ces objets dans l'environnement cible. Les objets qui seront créés dans l'environnement cible peuvent être créés avec les mêmes noms que ceux de l'environnement source ou vous pouvez spécifier un autre nom dans ce fichier.

Lorsque le processus d'importation est exécuté à partir de la ligne de commande **wappman**, vous pouvez demander en option à ce que le fichier de mappage soit modifié automatiquement, selon des règles définies avec des expressions régulières et indiquées dans un des fichiers ad hoc suivants :

- *application de charge de travail nom de modèle*_BasicRegExpMapping.UTF8.rules
- *application de charge de travail nom de modèle*_AdvancedRegExpMapping.UTF8.rules

Ces fichiers sont générés par l'exportation et, s'ils sont utilisés, ils doivent être correctement personnalisés.

Nom de modèle application de charge de travail **SourceEnv_reference.txt**

Informations de référence contenant les définitions des postes de travail utilisés dans le modèle de application de charge de travail et toutes les autres informations qui peuvent être utiles pour mapper correctement

l'environnement source dans l'environnement cible afin de permettre l'exécution de l'application de charge de travail.

application de charge de travail nom de modèle **BasicRegExpMapping.UTF8.rules**

Fichier contenant des règles définies avec des expressions régulières simples et permettant de modifier le fichier de mappage. (Facultatif)
Personnalisez le fichier en fonction des futurs noms des objets dans l'environnement cible. Le processus d'importation exécuté à partir de la ligne de commande **wappman** applique alors les règles définies pour modifier le fichier de mappage.

application de charge de travail nom de modèle **AdvancedRegExpMapping.UTF8.rules**

Fichier contenant des règles définies avec des expressions régulières avancées et permettant de modifier le fichier de mappage. (Facultatif)
Personnalisez le fichier en fonction des futurs noms des objets dans l'environnement cible. Le processus d'importation exécuté à partir de la ligne de commande **wappman** applique alors les règles définies pour modifier le fichier de mappage.

Vous pouvez importer le module compressé à partir de Dynamic Workload Console en suivant une procédure intuitive décrite dans la section relative à l'importation d'un modèle d'application de charge de travail dans le document *Dynamic Workload Console - Guide d'utilisation*.

Vous pouvez également utiliser la ligne de commande **wappman** pour importer manuellement le module compressé dans l'environnement cible où l'application de charge de travail sera déployée, et créer ainsi tous les objets requis dans l'environnement cible. Dans l'environnement cible, le fichier *application de charge de travail nom* **Mapping.UTF8.properties** doit être édité manuellement ou à l'aide de fichiers de mappage d'expressions régulières, en indiquant les noms des objets tels qu'ils sont définis dans l'environnement cible (par exemple, les noms des postes de travail sur lesquels s'exécutent les flots de travaux). Pour plus de détails, voir les sections sur les applications de charge de travail et la commande **wappman** dans IBM Workload Scheduler - Guide d'utilisation et de référence.

Exportation d'une définition de flot de travaux comme modèle d'application de charge de travail

A partir du Concepteur de charge de travail, vous pouvez exporter une définition de flot de travaux et la sauvegarder comme modèle d'application de charge de travail dans un fichier compressé. La définition du flot de travaux peut alors être importée dans un autre environnement.

Pour effectuer cette tâche, vérifiez que la connexion au moteur IBM Workload Scheduler est opérationnelle.

Pour exporter une définition de flot de travaux comme modèle d'application de charge de travail, exécutez la procédure suivante à partir de Concepteur de charge de travail :

1. Ouvrez le flot de travaux dont vous voulez exporter la définition. Pour trouver le flot de travaux, effectuez l'une des opérations suivantes :
 - Recherchez-le à l'aide du menu **Rechercher** dans la **Liste de travail**
 - Sélectionnez l'icône du flot de travaux  et lancez une recherche.

2. Vous pouvez maintenant exporter le flot de travaux à l'aide de la vue **Détails** ou **Graphique**. Sélectionnez l'onglet de la vue souhaitée.
 - Dans la vue **Détails** ou **Graphique**, procédez comme suit à l'aide des menus :
 - a. Cliquez avec le bouton droit de la souris sur le flot de travaux, puis sélectionnez l'option **Télécharger la définition du flot de travaux** dans le menu contextuel, ou sélectionnez le flot de travaux et choisissez la même option dans le menu **Sélectionner une action**.
 - b. Un panneau système dans lequel vous pouvez indiquer l'emplacement du téléchargement s'affiche.

La définition du flot de travaux est exportée comme modèle d'application de charge de travail dans un fichier compressé. Le fichier compressé est appelé par défaut *nom_flot_travaux.zip*. Vous pouvez maintenant importer la définition du flot de travaux dans un autre environnement.

Importation d'un modèle d'application de charge de travail

Comment importer un modèle d'application de charge de travail à l'aide de Dynamic Workload Console.

Vérifiez que vous avez créé un modèle application de charge de travail comme indiqué dans «Création d'un modèle d'application de charge de travail», à la page 394 ou «Exportation d'une définition de flot de travaux comme modèle d'application de charge de travail», à la page 397.

Vous pouvez importer votre modèle d'application de charge de travail pour l'exécuter dans un environnement cible. Pour importer un modèle d'application de charge de travail, procédez comme suit :

1. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de la charge de travail > Importation d'application de charge de travail**
2. Indiquez le nom d'un moteur réparti, puis naviguez vers le modèle d'application de charge de travail.
3. Cliquez sur **Téléchargement amont**.
4. Vous devez maintenant modifier les noms des objets tels qu'ils sont définis dans l'environnement cible (par exemple, les noms des postes de travail où les flots de travaux s'exécutent). Dans la colonne de gauche, sous l'en-tête **Charge de travail de base**, vous pouvez trouver les objets d'origine, tels que les postes de travail, les travaux et les flots de travaux. Dans les zones de la colonne de droite, vous pouvez entrer les noms à attribuer aux objets dans l'environnement cible.
5. Cliquez sur **Importer**.

Le modèle d'application de charge de travail est maintenant importé et prêt à être utilisé dans votre environnement cible.

Résolution du fichier de mappage

Le fichier de mappage généré par le processus d'exportation d'une application de charge de travail, contient une liste d'éléments, dont certains sont dépendants de la topologie de l'environnement dans lequel ils sont utilisés. Ces éléments doivent être personnalisés pour refléter l'environnement cible.

Pour utiliser l'application de charge de travail dans un nouvel environnement, modifiez le fichier de mappage de sorte à refléter l'environnement de destination à l'aide des informations fournies dans le fichier de référence, puis exécutez une opération d'importation. L'opération d'importation est réalisée en transmettant le fichier de mappage et le fichier de définition sous la forme d'une entrée à la commande wappman.

La commande wappman peut être utilisée pour importer, exporter, remplacer, répertorier, afficher et supprimer une application de charge de travail. Voir «wappman», à la page 406 pour obtenir la syntaxe et l'utilisation complète de la ligne de commande afin de réaliser ces actions sur une application de charge de travail et pour prendre connaissance des éventuelles remarques associées.

Au cours du processus d'exportation, les objets contenus dans l'application de charge de travail sont extraits dans le fichier de définitions avec la même définition que dans l'environnement source. Le fichier de définitions peut contenir une définition d'objet complète, ou dans certains cas, un nom ou une référence à l'objet extrait uniquement. Des références simples et une définition d'objet incomplète sont extraites pour ces objets qui doivent être mappés à un objet déjà présent dans l'environnement cible. Pour certains objets extraits par référence, la définition d'objet est écrite dans le fichier de mappage qui nécessite une personnalisation afin de mapper les objets dans l'environnement source IBM Workload Scheduler à l'environnement dans lequel l'application de charge de travail sera déployée.

Le fichier de mappage peut être consulté et édité à l'aide d'un éditeur de texte. Il est organisé en sections et contient des commentaires permettant de vous aider à attribuer les valeurs correctes aux éléments.

De plus, lorsque le processus d'importation est exécuté à partir de la ligne de commande **wappman**, vous pouvez demander en option à ce que le fichier de mappage soit modifié automatiquement, selon des règles définies avec des expressions régulières et indiquées dans un des fichiers ad hoc suivants :

- *application de charge de travail*name_BasicRegExpMapping.UTF8.rules
- *application de charge de travail*name_AdvancedRegExpMapping.UTF8.rules

Ces fichiers sont générés par l'exportation et, s'ils sont utilisés, ils doivent être correctement personnalisés. Pour plus de détails, voir «Utilisation d'expressions régulières pour modifier le fichier de mappage», à la page 404.

Le tableau ci-dessous présente tous les objets qui peuvent se trouver dans une application de charge de travail ou qui sont référencés par un autre élément dans l'application de charge de travail et la manière dont le processus d'exportation les gère.

Tableau 60. Objets extraits au cours du processus d'exportation

Type d'objet	Quels éléments sont exportés vers le fichier de définitions ?	Quels éléments nécessitent d'être personnalisés dans le fichier de mappage ?	Que se passe-t-il au cours de l'importation ?
Flot de travaux	Définition d'objet incluant la condition de démarrage du flot de travaux, si elle est indiquée.	Poste de travail Nom du flot de travaux Alias du travail Flot de travaux condition de démarrage si elle spécifiée	Un flot de travaux est créé dans la base de données. Restriction : Si le flot de travaux possède une dépendance à un flot de travaux ou un travail externe à l'application de charge de travail, le fichier de mappage contient une référence au nom du flot de travaux ou du travail externe et à la définition de poste de travail relative. Toutefois, le fichier de définitions ne contient pas la définition de flot de travaux ou de travail. Le nom du fichier de mappage doit être mappé à un flot de travaux ou un travail existant dans l'environnement cible afin d'importer correctement l'application de charge de travail.
Travail	Définition d'objet	Poste de travail Nom du travail Affinité Variables de mot de passe trouvés dans JSDL	Un travail est créé dans la base de données. Restriction : Si le travail possède une dépendance à un travail qui est défini dans un flot de travaux externe à l'application de charge de travail, le fichier de mappage contient une référence aux objets suivants : nom du flot de travaux externe, travail défini dans le flot de travaux externe, définitions de postes de travail dans le flot de travaux et définitions de poste de travail du travail. Cependant, le fichier de définitions ne contient pas la définition du travail. Le nom du fichier de mappage doit être mappé à un flot de travaux ou un travail existant dans l'environnement cible afin d'importer correctement l'application de charge de travail. Les relations d'affinité déclenchent l'exécution de travaux sur le même poste de travail. Le poste de travail sur lequel le premier travail s'exécute est choisi de façon dynamique, le travail ou les travaux d'affinité s'exécutant sur le même poste de travail. Les travaux doivent appartenir au même flot de travaux. Lorsqu'un travail avec une affinité est exporté, le nom du travail est ajouté au fichier de mappage. Les variables du JSDL utilisent le format \${password:ws#user}. Seuls les postes de travail sont représentés de façon générique. La zone de l'utilisateur est copiée telle quelle dans l'environnement cible. Les variables doivent être utilisées pour les noms d'utilisateur.

Tableau 60. Objets extraits au cours du processus d'exportation (suite)

Type d'objet	Quels éléments sont exportés vers le fichier de définitions ?	Quels éléments nécessitent d'être personnalisés dans le fichier de mappage ?	Que se passe-t-il au cours de l'importation ?
Tables de variable de cycle d'exécution Tables de variable du flot de travaux	Définition d'objet	Nom de la table Nom de la variable	Une table de variable est créée dans la base de données. Les tables de poste de travail et de variable par défaut sont extraites par référence et écrites dans le fichier de mappage. La valeur associée à la variable peut être modifiée, pas les noms de variable. Evitez d'associer la table de variable par défaut aux flots de travaux et aux cycles d'exécution.
User	Aucune action. Aucune définition d'objet ou référence n'est établie dans le fichier de référence.		L'utilisateur doit exister dans l'environnement cible. Des variables doivent être utilisées comme références aux utilisateurs afin de rendre l'application de charge de travail plus flexible pour les futures utilisations.
Agenda	Définition d'objet	Nom de l'agenda	Un agenda est créé dans la base de données
Cycle d'exécution	Définition d'objet	Non de cycle d'exécution	Un cycle d'exécution est créé dans la base de données
Groupe de cycle d'exécution	Définition d'objet	Nom du groupe de cycle d'exécution	Un groupe de cycle d'exécution est créé dans la base de données
Dépendances interréseaux	Le travail ou le flot de travaux référencé n'est pas exporté car il appartient à un moteur différent.	Nom du poste de travail de l'agent de réseau qui gère les dépendances de prédécesseur/successeur entre le réseau local et le réseau distant.	Une dépendance interréseau est ajoutée au travail ou au flot de travaux.
Dépendances externes	Le travail ou le flot de travaux référencé est exporté uniquement s'il appartient au modèle d'application de charge de travail.	Si le travail ou le flot de travaux référencé n'appartient pas au modèle d'application de charge de travail, attribuez un nom au travail ou au flot de travaux qui correspond à un travail ou un flot de travaux qui existe déjà dans l'environnement cible.	Une dépendance externe est ajoutée au travail ou au flot de travaux
Ressources	Définition d'objet	Poste de travail Nom de la ressource	Des ressources sont créées dans la base de données

Tableau 60. Objets extraits au cours du processus d'exportation (suite)

Type d'objet	Quels éléments sont exportés vers le fichier de définitions ?	Quels éléments nécessitent d'être personnalisés dans le fichier de mappage ?	Que se passe-t-il au cours de l'importation ?
Invites globales	Définition d'objet	Variables utilisées dans la définition	Des invites globales sont créées dans la base de données. Vous pouvez utiliser des variables. Etant donné qu'elles sont résolues à l'aide de la table par défaut, la variable utilisée dans une invite globale peut être mappée à une variable dans l'environnement cible.
Postes de travail	Référence uniquement	Nom	Non importée. Les postes de travail sont extraits dans le fichier de définition sous la forme d'une référence. Une définition n'est pas importée car les postes de travail sont déjà définis dans l'environnement cible. Toutefois, leurs noms n'ont pas besoin d'être mappés.
Classe de poste de travail	Référence uniquement	Nom	Non importée. Les classes de poste de travail sont extraites dans le fichier de définition sous la forme d'une référence. Une définition n'est pas importée car les classes de poste de travail sont déjà définies dans l'environnement cible. Toutefois, leurs noms n'ont pas besoin d'être mappés.
Variable	Définition d'objet	Valeur	Importée. Les variables sont utilisées à plusieurs emplacements dans une définition de flot de travaux. Une référence est ajoutée au fichier de définitions.
Règles d'événement	Définition d'objet	Nom	Les règles d'événement sont créées dans la base de données pour les postes de travail et les flots de travaux inclus dans le modèle.

Exemple

L'exemple ci-dessous décrit comment les informations des fichiers contenus dans le fichier compressé créé par l'exportation de l'application de charge de travail sont utilisées pour préparer les fichiers au déploiement dans l'environnement cible.

Tableau 61. Résolution du fichier de mappage

Fichier de définitions	Fichier de mappage	Fichier d'informations de référence
<pre> <model:JobStream carryforward= "true" draft="false" iskey="false" limit="55" name= "\$JOBSTREAM_BADPBN34_JS1_1I\$" onrequest="false" priority="100" workstation="\$WORKSTATION_ BADPBN34_WC1\$"> <model:runcycles/> <model:matching> <model:sameDay/> </model:matching> <model:restrictions/> <model:dependencies/> <model:jobs> <model:job confirmed="false" definition= "\$WORKSTATION_MDM112097\$ #\$JOB_BADPBN34_J1\$" isCritical="false" iskey="false" name="\$JOB_BADPBN34_J1\$" priority="10"> <model:restrictions/> <model:dependencies> <model:predecessor target= "\$WORKSTATION_BADPBN34_ WC1#\$JOBSTREAM_BADPBN34_ JS2\$.@"> <model:matching> <model:previous> </model:matching> </model:predecessor> </model:dependencies> </model:job> </model:jobs> </model:JobStream> </pre>	<pre> #Noms des postes de travail #Remplacez la valeur par le nom d'un poste de travail qui existe déjà dans l'environnement cible #Voir le fichier MAIN_TEMPLATE_ SourceEnv_reference.txt pour obtenir des détails sur le poste de travail. # #Ce poste de travail est de type Gestionnaire WORKSTATION_MDM112097=MDM112097 #Ce poste de travail est de type Agent WORKSTATION_MDM112097_1= MDM112097_1 #Ce poste de travail est de type Courtier WORKSTATION_MDM112097_DWB= MDM112097_DWB </pre>	<pre> CPUNAME \$WORKSTATION_MDM112097\$ DESCRIPTION "Exemple de gestionnaire de domaine maître de sauvegarde" OS UNIX NODE MDM112097.romelab.it.ibm.com TCPADDR 35111 TIMEZONE Europe/Rome DOMAIN MASTERDM FOR MAESTRO TYPE MDM AUTOLINK ON BEHINDFIREWALL ON FULLSTATUS ON SERVER A FIN </pre>
Le fichier d'informations de référence indique que le poste de travail nommé MDM112097 est de type gestionnaire de domaine maître s'exécutant sur un système d'exploitation UNIX. Le fichier de définitions contient des références au nom du poste de travail. L'entrée dans le fichier de mappage WORKSTATION_MDM112097=MDM112097 doit ainsi être mise à jour. Remplacez MDM112097 par le nom d'un poste de travail qui existe déjà dans l'environnement cible et qui est doté des mêmes caractéristiques que dans le fichier d'informations de référence.		

Fichier de définitions

```

<model:JobStream carryforward="true" draft="false"
iskey="false" limit="55" name="$JOBSTREAM_BADPBN34_JS1_1I$"
onrequest="false" priority="100"
workstation="$WORKSTATION_BADPBN34_WC1$">
  <model:runcycles/>
  <model:matching>
    <model:sameDay/>
  </model:matching>
  <model:restrictions/>
  <model:dependencies/>
  <model:jobs>
    <model:job confirmed="false"
defintion="$WORKSTATION_MDM112097$#$JOB_BADPBN34_J1$"
isCritical="false" iskey="false" name="$JOB_BADPBN34_J1$"
priority="10">
      <model:restrictions/>
      <model:dependencies>
        <model:predecessor target="$WORKSTATION_BADPBN34_WC1$#$JOBSTREAM_
BADPBN34_JS2$.@">

          <model:matching>
            <model:previous>
          </model:matching>
        </model:predecessor>
      </model:dependencies>
    </model:job>
  </model:jobs>
</model:JobStream>

```

= Utilisation d'expressions régulières pour modifier le fichier de mappage

Vous pouvez, le cas échéant, demander à ce que le fichier de mappage produit par le processus d'exportation soit modifié automatiquement par le processus d'importation, en fonction des règles définies à l'aide des expressions régulières.

Le fichier de mappage généré par le processus d'exportation d'une application de charge de travail, contient une liste d'éléments, dont certains sont dépendants de la topologie de l'environnement dans lequel ils sont utilisés. Ces éléments doivent être personnalisés pour correspondre à l'environnement cible. Lorsque le processus d'importation est exécuté à partir de la ligne de commande **wappman**, vous pouvez demander en option à ce que le fichier de mappage soit modifié automatiquement, selon des règles définies avec des expressions régulières et indiquées dans un des fichiers ad hoc suivants :

*application de charge de travail*name_**BasicRegExpMapping.UTF8.rules**

Ce fichier contient les règles définies à l'aide des expressions régulières qui incluent uniquement les caractères génériques '*' et '?'.

*application de charge de travail*name_**AdvancedRegExpMapping.UTF8.rules**

Ce fichier contient les règles définies à l'aide d'expressions régulières avancées conformément aux normes Java. Pour plus d'informations sur les expressions régulières avancées, voir la documentation appropriée.

Ces fichiers sont générés par l'exportation et, s'ils sont utilisés, ils doivent être correctement personnalisés.

Sélectionnez le fichier qui correspond le mieux à vos besoins et personnalisez les expressions régulières en fonction des futurs noms des objets dans l'environnement cible. Le processus d'importation exécuté à partir de la ligne de commande **wappman** applique alors les règles définies pour modifier la valeur des éléments inclus dans le fichier de mappage.

Chaque fichier est subdivisé en sections, une pour chaque type d'objet pouvant être contenu dans une application de charge de travail. Pour chaque section, vous trouverez des commentaires et des exemples de règles, définis à l'aide d'expressions régulières, qui s'appliquent à l'objet spécifique. Par exemple :

```
[JOBSTREAM]
## DEV=* _PRD
#A24Y?=B42X?
```

Supprimez la mise en commentaire des exemples fournis ou ajoutez vos propres règles. Par exemple :

```
[JOBSTREAM]
* DEV=* _PRD
A24Y?=B42X?
J*=A*
```

Chaque règle est composée de deux expressions régulières. L'expression régulière située à gauche définit le motif recherché, et celle située à droite définit la valeur de remplacement de chaque élément concordant. Au cours du processus d'importation, les actions de recherche et de remplacement sont effectuées sur la valeur des éléments inclus dans le fichier de mappage.

Si un objet de l'application de charge de travail correspond à plusieurs entrées dans le fichier, seule la règle incluse dans la dernière entrée concordante est appliquée. Par exemple :

Conformément aux règles ci-dessus :

- le flot de travaux JS1_DEV est renommé en AS1_DEV
- le flot de travaux CCB_DEV est renommé en CCB_PRD
- le flot de travaux A24YK est renommé en B42XK

Déploiement d'une application de charge de travail

Le déploiement d'une application de charge de travail se compose de deux étapes. Il commence par la personnalisation du fichier de mappage et se termine par l'importation du fichier de mappage et du fichier de définitions dans le nouvel environnement IBM Workload Scheduler.

1. Extrayez le contenu du modèle d'application de charge de travail à partir du fichier compressé créé par l'opération d'exportation.
2. Personnalisez le fichier de mappage. Attribuez à chaque objet répertorié dans le fichier de mappage le nom d'un objet existant dans l'environnement cible, ou renommez-le avec le nom de votre choix dans l'environnement cible. Pour des informations sur la modification du fichier de mappage, voir «Résolution du fichier de mappage», à la page 398. Vous pouvez, en option, modifier le fichier de mappage conformément aux règles définies à l'aide d'expressions régulières et spécifiées dans les fichiers ad-hoc. Ces fichiers sont générés par l'exportation et, s'ils sont utilisés, ils doivent être correctement personnalisés. Pour des informations sur l'utilisation d'expressions régulières, voir «Utilisation d'expressions régulières pour modifier le fichier de mappage», à la page 404.
3. A partir de la ligne de commande, soumettez la commande **wappman -import** en indiquant le fichier de définition, le fichier de mappage et, en option, le fichier contenant les règles définies à l'aide d'expressions régulières. Voir «wappman», à la page 406 pour obtenir des détails sur l'utilisation et la syntaxe de la commande.

Tous les objets IBM Workload Scheduler définis dans *fichier_XML_définitions* sont créés dans l'environnement cible, s'ils n'existent pas déjà. Vous mettez à jour le

fichier de mappage afin de résoudre les références aux objets qui doivent être personnalisés pour refléter l'environnement cible dans lequel l'application de charge de travail sera déployée.

Vous pouvez mettre à jour une application de charge de travail ultérieurement. Il existe deux cas dans lesquels l'application de charge de travail peut être mise à jour ou modifiée :

Modification du modèle dans l'environnement source

Une version mise à jour du modèle peut être déployée à nouveau dans l'environnement cible. Tout objet déjà présent dans la base de données IBM Workload Scheduler de l'environnement cible est remplacé par les versions mises à jour. Au contraire, tout objet qui n'existe pas déjà dans l'environnement cible est créé et les objets sont supprimés de l'environnement cible si la définition d'objet a été supprimée de l'application de charge de travail mise à jour. Le même fichier de mappage utilisé pour déployer initialement l'application de charge de travail peut être utilisé pour la mettre à jour, en personnalisant tout nouvel objet en cours de déploiement avec la mise à jour.

Modification de l'instance dans l'environnement cible

Après avoir déployé une application de charge de travail, vous pouvez ajouter un nouveau travail à un flot de travaux, modifier une définition de travail ou supprimer un travail ou un flot de travaux. Ces modifications ne sont toutefois pas conservées si l'application de charge de travail est mise à jour avec un modèle d'application de charge de travail révisé.

wappman

Importe, remplace, supprime, exporte, affiche et répertorie une application de charge de travail.

Remarque : Sous Windows 2012, la commande n'est pas prise en charge sur Windows PowerShell.

Autorisation

Vous devez avoir le droit d'accès **add** pour importer une application de charge de travail. Si l'objet existe déjà dans la base de données, vous devez avoir le droit d'accès **modify** à l'objet.

Syntaxe

```
wappman[paramètres_connexion]
[-u] | [-V]
| {-import|-replace} <fichier_XML_définitions> <fichier_propriétés_mappage>
[-translationRules <fichier_règles_traduction>]
[-workloadApplicationName <nom_application_charge_travail>]
-list
-delete <nom_application_charge_travail>
-export <nom_application_charge_travail> [-path <chemin_exportation>] [-zip <nom_zip_exportation>]
-display <nom_application_charge_travail>
```

Arguments

[paramètres_connexion]

Fichier contenant les paramètres de connexion à utiliser pour se connecter

au serveur où vous souhaitez importer, répertorier, supprimer, exporter ou afficher l'application de charge de travail. Vous pouvez utiliser les paramètres de connexion pour remplacer les valeurs spécifiées dans les fichiers useropts et localopts. Pour déterminer les propriétés de connexion à utiliser, une vérification est réalisée dans l'ordre suivant :

1. Paramètres spécifiés dans la chaîne de commande.
2. Paramètres spécifiés dans le fichier des propriétés personnalisées.
3. Fichier useropts.
4. Fichier localopts.
5. Fichier jobmanager.ini.

Les valeurs possibles sont les suivantes :

[-username <nom_utilisateur>]

Utilisateur IBM Workload Scheduler doté des droits suffisants pour effectuer l'opération.

[-password <mot_de_passe>]

Mot de passe de l'utilisateur IBM Workload Scheduler.

[-host <nom_hôte>]

Nom de l'hôte auquel vous souhaitez accéder à l'aide de la ligne de commande wappman.

[-port <numéro_port>]

Numéro de port TCP/IP utilisé pour se connecter à l'hôte indiqué.

[-protocol {http | https}]

Protocole utilisé pour se connecter à l'hôte indiqué.

[-file <fichier_propriétés_personnalisées>]

Fichier de propriétés personnalisées dans lequel vous pouvez spécifier des paramètres de connexion qui prévalent sur les valeurs indiquées dans les fichiers useropts, localopts et jobmanager.ini. Les paramètres de connexion spécifiés dans le fichier de propriétés personnalisé doivent avoir la syntaxe suivante :

```
HOST=<nom_hôte>
PORT=<port>
PROTOCOL=<http/https>
USERNAME=<nom_utilisateur>
PASSWORD=<mot_de_passe>
```

Remarque : Si les paramètres host, port et protocol sont spécifiés dans un fichier, ils doivent tous l'être dans le même fichier.

-u Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

{-import | -replace} <fichier_xml_définitions> <fichier_propriétés_mappage> [-translationRules <fichier_règles_traduction>] [-workloadApplicationName <nom_application_charge_travail>]

Nom du fichier de définitions et du fichier de mappage à utiliser lors de l'importation ou du remplacement d'une application de charge de travail. L'opération d'importation d'une application de charge de travail est identique au déploiement d'une application de charge de travail.

-import

Après avoir personnalisé le fichier de mappage, celui-ci ainsi que le fichier de définitions sont transmis en tant qu'entrée à la

commande **wappman -import** afin de déployer l'application de charge de travail dans un environnement IBM Workload Scheduler autre que l'environnement source où le modèle d'application de charge de travail a été créé à l'origine.

-replace

Une version mise à jour du modèle d'application de charge de travail est réimportée dans l'environnement cible. Les objets qui étaient initialement créés dans la base de données IBM Workload Scheduler au premier déploiement du modèle d'application de charge de travail, sont mis à jour s'ils sont toujours présents dans la base de données, créés s'ils ne s'y trouvent plus et supprimés s'ils n'existent plus dans le modèle d'application de charge de travail mis à jour.

L'opération replace échoue si un objet externe à l'application de charge de travail référence un objet dans l'application de charge de travail. Suite à l'opération replace, l'objet référencé est supprimé car il n'existe plus dans l'application de charge de travail mise à jour.

Si des références ont été établies de l'application de charge de travail à un élément externe, la référence est supprimée à l'aide de l'action replace.

Le même fichier de mappage utilisé pour déployer initialement l'application de charge de travail peut être utilisé pour la mettre à jour, en effectuant les modifications nécessaires pour refléter l'application de charge de travail mise à jour. Le fichier de mappage et le fichier de définitions sont transmis en tant qu'entrée à la commande **wappman -replace**.

Les paramètres supplémentaires facultatifs que vous pouvez spécifier sont les suivants :

-translationRules

Nom du fichier contenant les règles définies à l'aide d'expressions régulières que la commande **wappman -import | -replace** utilisera pour personnaliser le fichier de mappage.

-workloadApplicationName

Nom de l'application de charge de travail, au cas où vous souhaiteriez l'importer avec un nouveau nom.

-list Répertoire toutes les applications de charge de travail présentes dans l'environnement.

-delete <nom_application_charge_travail>

Supprime l'application de charge de travail spécifiée et tous les objets qui ont été ajoutés à l'environnement lorsque l'application de charge de travail a été initialement déployée.

-export <nom_application_charge_travail>

Crée le fichier compressé, appelé modèle d'application de charge de travail, qui contient l'ensemble des fichiers et des informations nécessaires pour déployer l'application de charge de travail dans l'environnement cible.

-path Chemin où le modèle d'application de charge de travail est exporté.

-zip Nom du fichier zip contenant le modèle d'application de charge de travail.

-display <nom_application_charge_travail>

Affiche le contenu de l'application de charge de travail spécifiée.

Affiche tous les objets présents dans l'application de charge de travail qui ont été créés au cours du processus d'importation.

Distributé

Commentaires

Les utilisateurs peuvent décider de gérer une trace d'audit qui consigne tous les changements qu'ils effectuent et les justifications associées. Pour activer l'option de justification, configurez les variables d'environnement IBM Workload Scheduler ci-dessous dans un shell système avant d'exécuter les commandes **wappman** :

IWS_DESCRIPTION

Indiquez la description à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 512 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_CATEGORY

Indiquez la catégorie à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_TICKET

Indiquez le ticket à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

Pour configurer l'environnement, suivez la procédure correspondant à votre système d'exploitation :

Sous Windows

entrez une ou plusieurs commandes ci-dessous, selon la variable que vous souhaitez définir :

```
set IWS_CATEGORY='my category'
set IWS_DESCRIPTION='my desc'
set IWS_TICKET=12345
```

Sous UNIX

entrez une ou plusieurs commandes ci-dessous, selon la variable que vous souhaitez définir :

```
export IWS_CATEGORY='my category'
export IWS_DESCRIPTION='my desc'
export IWS_TICKET=12345
```

Une fois que les variables d'environnement ont été définies, toutes les commandes que vous entrez dans le shell héritent les valeurs que vous avez définies pour le shell. Les informations d'audit provenant du shell sont stockées en l'état dans les rapports d'audit Tivoli Common Reporting. Si les informations que vous fournissez dans Dynamic Workload Console sont dirigées à l'aide d'un menu déroulant, les informations fournies à partir de la ligne de commande ne sont pas filtrées.

Pour plus d'informations sur l'option de justification et les rapports d'audit Tivoli Common Reporting, voir la section relative au suivi des modifications et de Tivoli

Common Reporting dans le document *Dynamic Workload Console - Guide d'utilisation*.

Journaux et fichiers de trace

Vous pouvez configurer la ligne de commande **wappman** en modifiant les paramètres dans le fichier `CLI.ini` situé à l'emplacement `rép_base_TWA/TWS/ITA/cpa/config`. Ce fichier contient les paramètres des journaux de messages et des fichiers de trace associés aux applications de charge de travail. Vous ne devriez modifier les paramètres que si la documentation IBM Workload Scheduler le recommande ou si le service de support logiciel IBM vous invite à le faire.

Exemples

Pour importer une application de charge de travail dans un nouvel environnement, exécutez :

```
wappman -import <fichier_XML_définitions> <fichier_propriétés_mappage>
```

Pour remplacer une application de charge de travail existante, exécutez :

```
wappman -replace <fichier_XML_définitions> <fichier_propriétés_mappage>
```

Pour supprimer une application de charge de travail, exécutez :

```
wappman -delete <nom_application_charge_travail>
```

Pour répertorier toutes les applications de charge de travail disponibles dans un environnement, exécutez :

```
wappman -list
```

Pour afficher une application de charge de travail spécifique, exécutez :

```
wappman -display <nom_application_charge_travail>
```

Pour exporter une application de charge de travail donnée à partir d'un hôte distant, à l'aide d'un fichier de propriétés personnalisé, exécutez :

```
wappman -export <nom_application_charge_travail> -path <chemin_exportation> -host <hôte_distant> -fi
```

Voir aussi

Pour plus d'informations sur la définition d'un modèle d'application de charge de travail dans l'environnement source, voir «Création d'un modèle d'application de charge de travail», à la page 394.

Pour plus d'informations sur la gestion d'une application de charge de travail dans l'environnement cible, voir «Déploiement d'un application de charge de travail», à la page 405

Chapitre 12. Gestion des objets dans le plan - conman

L'environnement du plan de production IBM Workload Scheduler est géré à l'aide du programme de ligne de commande **conman**. Le programme **conman** est utilisé pour démarrer et arrêter le traitement, modifier et afficher le plan de production Symphony et contrôler les liens entre les postes de travail dans un réseau. Il peut être utilisé à partir du gestionnaire de domaine maître ou de tout agent tolérant aux pannes du réseau IBM Workload Scheduler.

Le présent chapitre comprend les sections suivantes :

- «Configuration du programme de ligne de commande conman»
- «Exécution de commandes depuis Conman», à la page 416
- «Sélection de travaux dans les commandes», à la page 419
- «Sélection de flots de travaux dans les commandes», à la page 428
- «Gestion des travaux et des flots de travaux des agents de niveau antérieur», à la page 435
- «Commandes Conman», à la page 436

Configuration du programme de ligne de commande conman

Le programme de ligne de commande **conman** gère l'environnement du plan de production.

Vous pouvez utiliser le programme **conman** à partir du gestionnaire de domaine maître et de tout poste de travail agent tolérant aux pannes du réseau IBM Workload Scheduler.

Les utilisateurs peuvent décider de gérer une trace d'audit qui consigne tous les changements qu'ils effectuent et les justifications associées. Pour activer l'option de justification, configurez les variables d'environnement IBM Workload Scheduler ci-dessous dans un shell système avant d'exécuter des commandes **conman** :

IWS_DESCRIPTION

Indiquez la description à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 512 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_CATEGORY

Indiquez la catégorie à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

IWS_TICKET

Indiquez le ticket à consigner pour chaque changement effectué par des commandes dans le shell. La longueur maximale pour cette valeur est 128 caractères. Un message d'avertissement s'affiche si vous dépassez la valeur maximale et les caractères en trop sont tronqués.

Pour plus d'informations sur l'option de justification, voir la section décrivant le suivi des changements dans le document *IBM Dynamic Workload Console - Guide d'utilisation*.

Configuration de l'environnement Conman

Vous trouverez dans la présente section des informations sur la configuration que vous pouvez choisir pour votre environnement **conman**.

Remarque : Sur les systèmes Windows, avant d'exécuter **conman**, assurez-vous que la page de codes et les polices de caractères sont définies correctement dans le shell DOS afin d'éviter toute conversion incorrecte des caractères. Pour plus d'informations sur les paramètres obligatoires, voir la section *Remarques relatives à l'internationalisation* du document *IBM Workload Scheduler : Release Notes* à l'adresse <http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048863>.

Sortie sur le terminal

La sortie sur votre ordinateur est déterminée par les variables shell définies dans le script `twc_env` que vous lancez avant d'exécuter les commandes de ligne de commande. Si l'une ou l'autre de ces variables n'est pas définie, les variables de shell standard **LINES** et **COLUMNS** sont utilisées. Les variables peuvent être définies comme suit :

MAESTROLINES

Indique le nombre de lignes par page d'écran. La valeur par défaut est **-1**. A la fin de chaque page d'écran, le programme **conman** ne fait pas de pause à la fin d'une page. Si **MAESTROLINES** (ou **LINES**) est défini sur une valeur positive, le programme **conman** vous invite à continuer.

Il est recommandé d'utiliser **MAESTROLINES** puisque **LINES** est une variable du système d'exploitation shell et que dans la plupart des systèmes d'exploitation elle est automatiquement réinitialisée par le système lui-même.

MAESTROCOLUMNS

Indique le nombre de caractères par ligne. Les options suivantes sont disponibles :

- Inférieur à 120
- Supérieur ou égal à 120

MAESTRO_OUTPUT_STYLE

Indique la façon dont les noms d'objets sont affichés. Si la valeur définie est **LONG**, les noms complets sont affichés. Si la valeur est définie par une valeur différente de **LONG**, les noms longs sont tronqués à huit caractères suivis du signe (+).

Sortie offline

Dans les commandes **conman**, l'option **;offline** sert généralement à imprimer la sortie d'une commande. Lorsqu'elle est insérée, la sortie est contrôlée par les variables de shell ci-après :

MAESTROL

Indique la destination de la sortie d'une commande. Vous pouvez indiquer :

> *fichier*

La sortie est dirigée vers un fichier dont elle remplace le contenu. Si le fichier n'existe pas, il est créé.

>> *fichier*

La sortie est dirigée vers un fichier et est ajoutée à la fin de celui-ci. Si le fichier n'existe pas, il est créé.

| *commande*

La sortie est dirigée vers une commande système ou un processus.
Le programme exécute la commande système, que la sortie soit
générée ou non.

|| *commande*

La sortie est dirigée vers une commande système ou un processus.
La commande système n'est pas exécutée en l'absence de résultat.

La valeur par défaut de **MAESTROL** est | **lp -tCONLIST** qui transmet la
sortie de la commande à l'imprimante et insère le titre "CONLIST" sur la
page bannière de la sortie imprimée.

MAESTROLPLINES

Indique le nombre de lignes par page. La valeur par défaut est **60**.

MAESTROLPCOLUMNS

Indique le nombre de caractères par ligne. La valeur par défaut est **132**.

Les variables doivent être exportées avant l'exécution de **conman**.

Sélection de l'invite conman sous UNIX

Par défaut, l'invite de commande **conman** est figurée par le signe %. Elle est
définie dans le fichier *racine_TWS/localopts*. L'invite de commande par défaut
correspond à un tiret (-). Pour sélectionner une autre invite, éditez l'option d'invite
de **conman** dans le fichier *localopts* et modifiez le tiret. Le message d'invite peut
comporter jusqu'à 10 caractères, le signe dièse (#) de fin requis non inclus.

```
#-----  
# Custom format attributes  
#  
date format =          1      # The possible values are 0-ymd, 1-mdy,  
2-dmy, 3-NLS.  
composer prompt =      -  
conman prompt =        %  
switch sym prompt =    <n>%  
#-----
```

Exécution de conman

Pour configurer l'environnement en vue d'utiliser **conman**, définissez les variables
PATH et **TWS_TISDIR** en exécutant l'un des scripts suivants :

Sous UNIX :

- **./racine_TWS/tws_env.sh** pour les interpréteurs de commandes
Bourne et Korn
- **./racine_TWS/tws_env.csh** pour les interpréteurs de commandes C

Sous Windows :

- **racine_TWS\tws_env.cmd**

Utilisez ensuite la syntaxe suivante pour exécuter des commandes à partir de
l'interface utilisateur **conman** :

conman [*paramètres_connexion*] ["*commande* [&[*commande*]...] [&]"

où :

paramètres_connexion

Si vous utilisez **conman** à partir du gestionnaire de domaine maître, les

paramètres de connexion ont été configurés lors de l'installation et n'ont pas besoin d'être fournis, à moins que vous ne vouliez pas utiliser les valeurs par défaut.

Si vous utilisez **conman** à partir du client de ligne de commande sur un autre poste de travail, les paramètres de connexion peuvent être fournis par l'une ou plusieurs des méthodes suivantes :

- Ils sont stockés dans le fichier `localopts`
- Ils sont stockés dans le fichier `useropts`
- Ils sont fournis à la commande dans un fichier de paramètres
- Ils sont fournis à la commande au sein d'une chaîne de commande

Pour une présentation de ces options, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65. Pour des détails complets sur les paramètres de configuration, voir la rubrique relative à la configuration de l'accès du client de ligne de commande dans *IBM Workload Scheduler - Guide d'administration*.

Remarque : Si vous utilisez **conman** à partir du client de ligne de commande sur un autre poste de travail, pour les commandes suivantes :

- `Rerunsucc`
- `Listsucc`

la ligne de commande **conman** se connecte au serveur à l'aide d'une connexion TTPS. Dans ce cas, le client de ligne de commande assemble l'ensemble des paramètres de connexion dans l'ordre suivant :

1. Paramètres spécifiés dans la chaîne de commande
2. Paramètres spécifiés dans le fichier des propriétés personnalisées
3. Fichier `useropts`
4. Fichier `localopts`
5. Fichier `jobmanager.ini`

Les valeurs possibles sont les suivantes :

[-username <nom_utilisateur>]

Utilisateur IBM Workload Scheduler doté des droits suffisants pour effectuer l'opération.

[-password <mot_de_passe>]

Mot de passe de l'utilisateur IBM Workload Scheduler.

[-host <nom_hôte>]

Nom de l'hôte auquel vous souhaitez accéder à l'aide de la ligne de commande `wappman`.

[-port <numéro_port>]

Numéro de port TCP/IP utilisé pour se connecter à l'hôte indiqué.

[-protocol {http | https}]

Protocole utilisé pour se connecter à l'hôte indiqué.

[-file <fichier_propriétés_personnalisées>]

Fichier de propriétés personnalisées dans lequel vous pouvez spécifier des paramètres de connexion qui prévalent sur les valeurs indiquées dans les fichiers `useropts`, `localopts` et `jobmanager.ini`. Les paramètres de connexion spécifiés dans le fichier de propriétés personnalisé doivent avoir la syntaxe suivante :

```
=
HOST=<nom_hôte>
=
PORT=<port>
=
PROTOCOL=<http/https>
=
USERNAME=<nom_utilisateur>
=
PASSWORD=<mot_de_passe>
```

= Si les paramètres host, port et protocol sont spécifiés dans un fichier, ils
 = doivent tous l'être dans le même fichier.

Vous pouvez appeler la ligne de commande **conman** en mode de *traitement par lots* et en mode *interactif*.

Lorsque vous exécutez **conman** en mode *interactif*, vous lancez d'abord le programme de ligne de commande **conman** puis vous exécutez les commandes l'une après l'autre à partir de l'invite de la ligne de commande **conman**, par exemple :

```
conman -username admin2 -password admin2pwd
ss @+state=hold;deps
dds sked5;needs=2 tapes
```

Lorsque vous exécutez **conman** en mode de *traitement par lots*, vous lancez d'abord le programme de ligne de commande **conman** en indiquant comme paramètre d'entrée la commande à émettre. Une fois la commande traitée, le programme de ligne de commande **conman** s'arrête comme dans l'exemple suivant :

```
conman"sj&sp"
```

Lors de l'émission de commandes à partir de **conman** en mode de traitement par lots, assurez-vous de délimiter les commandes entre guillemets. Voici des exemples de l'utilisation du mode de traitement par lots pour émettre plusieurs commandes depuis **conman** :

- **conman** exécute les commandes **sj** et **sp** avant de s'arrêter :

```
conman"sj&sp"
```
- **conman** exécute les commandes **sj** et **sp**, puis invite l'utilisateur à taper une commande :

```
conman "sj&sp&"
```
- **conman** lit les commandes à partir de **cfile** :

```
conman < cfile
```
- les commandes du fichier **cfile** sont dirigées vers **conman** :

```
cat cfile | conman
```

Remarque : Sur les postes de travail Windows, si le contrôle de compte utilisateur (UAC) est activé et que la liste d'exceptions UAC ne contient pas le fichier cmd.exe, vous devez ouvrir l'interpréteur de commandes DOS avec l'option "Exécuter en tant qu'administrateur" pour exécuter **conman** sur votre poste de travail en tant qu'utilisateur générique différent de l'administrateur ou utilisateur IBM Workload Scheduler.

Caractères de commande

Vous pouvez entrer les caractères de commande suivants pour arrêter **conman**.

Ctrl+c Le programme **conman** interrompt l'exécution de la commande dès que l'étape courante le permet et renvoie une invite de commande.

Ctrl+d **conman** s'arrête après avoir exécuté la commande courante, uniquement sur les postes de travail UNIX.

Exécution des commandes système

Lorsque vous entrez une commande système avec une barre verticale ou un préfixe de commande système (: ou !), elle est exécutée par un processus enfant. Pour garantir la sécurité, l'ID utilisateur effectivement associé au processus enfant est celui de l'utilisateur qui exécute **conman**.

Invites utilisateur

Si vous utilisez des caractères génériques pour sélectionner les objets soumis à l'action d'une commande, **conman** vous demande confirmation après avoir localisé chacun des objets recherchés. Si vous répondez **oui** l'action est effectuée sur l'objet, et si vous répondez **non** l'objet est ignoré et aucune action n'est effectuée.

Lorsque vous exécutez **conman** en mode interactif, les invites de confirmation sont exécutées sur votre ordinateur. La sélection de la touche **Retour** en réponse à une invite est interprétée comme étant une réponse **négative**. La fonction d'invite peut être désactivée dans une commande à l'aide de l'option **;noask**.

Bien qu'aucune invite de confirmation ne soit générée lorsque **conman** est exécuté en mode interactif, celui-ci se comporte comme si la réponse avait été **négative** dans chacun des cas et aucune action n'est effectuée sur les objets. Par conséquent, il est important d'inclure l'option **;noask** dans les commandes lorsque **conman** ne s'exécute pas en mode interactif.

Exécution de commandes depuis Conman

Les commandes du programme **conman** se décomposent de la manière suivante :

nom_commande sélection arguments

Où :

nom_commande

Indique le nom de la commande.

sélection

Indique l'objet ou le groupe d'objets sur lequel sera exécutée la commande.

arguments

Indique les arguments de la commande.

Voici un exemple de commande **conman** :

```
sj sked1(1100 03/05/2006).@+state=hold~priority=0;info;offline
```

où :

sj Forme abrégée de la commande **showjobs**.

sked1(1100 03/05/2006).@+state=hold~priority=0

Sélectionne tous les travaux du flot de travaux **sked1(1100 03/05/2006)** qui sont à l'état HOLD et dont la priorité est différente de zéro.

;info;offline

Arguments de la commande **showjobs**.

Pour les commandes **conman** qui s'exécutent sur les travaux et les flots de travaux dont les spécifications incluent le caractère spécial '#', le délimiteur de valeur '=' doit précéder la sélection d'objet.

Voir l'exemple suivant :

`cj=site3#apwkly(0900 02/19/06).report`

où :

cj Forme abrégée de la commande **cancel job**.

site3#apwkly(0900 02/19/06).report

Sélectionne le travail *report* dans le flot de travaux *apwkly(0900 02/19/06)* sur le poste de travail *site3*.

Caractères génériques

Les caractères génériques acceptés sont les suivants :

@ Remplace un ou plusieurs caractères alphanumériques.

? Remplace un caractère alphanumérique.

% Remplace un caractère numérique.

Délimiteurs et caractères spéciaux

Le tableau 62 répertorie les caractères qui ont une signification spéciale dans les commandes **conman** :

Tableau 62. Délimiteurs et caractères spéciaux pour la commande *conman*

Caractère	Description
&	Délimiteur de commande. Voir «Configuration du programme de ligne de commande conman», à la page 411.
+	Délimiteur servant à sélectionner les objets pour les commandes. Il permet d'ajouter un attribut dont un objet doit être doté. Par exemple : <code>sked1(1100 03/05/2006).@~priority=0</code>
~	Délimiteur servant à sélectionner les objets pour les commandes. Il permet d'ajouter un attribut dont un objet doit être doté. Par exemple : <code>sked1(1100 03/05/2006).@~priority=0</code>
;	Délimiteur d'argument. Par exemple : <code>;info;offline</code>
,	Délimiteur de répétition et de fourchette. Par exemple : <code>state=hold,sked,pend</code>
=	Délimiteur de valeur. Par exemple : <code>state=hold</code>
!:	Préfixes de commande qui transmettent la commande au système. Ces préfixes sont facultatifs. Si conman ne reconnaît pas la commande, celle-ci est transmise automatiquement au système. Par exemple : <code>!!ls</code> ou <code>:ls</code>
*	Préfixe de commentaire. Le préfixe doit être le premier caractère de la ligne de commande ou suivre un délimiteur de commande. Par exemple : <code>*commentaire</code> ou <code>sj& *commentaire</code>
>	Dirige la sortie de la commande vers un fichier dont elle remplace le contenu. Si le fichier n'existe pas, il est créé. Par exemple : <code>sj> joblist</code>

Tableau 62. Délimiteurs et caractères spéciaux pour la commande *conman* (suite)

Caractère	Description
>>	Dirige la sortie de la commande vers un fichier et celle-ci est ajoutée à la fin du fichier. Si le fichier n'existe pas, il est créé. Par exemple : sj>> joblist
	La sortie de la commande est dirigée vers une commande système ou un processus. Le programme exécute la commande système, que la sortie soit générée ou non. Par exemple : sj grep ABEND
	La sortie de la commande est dirigée vers une commande système ou un processus. La commande système n'est pas exécutée en l'absence de résultat. Par exemple : sj grep ABEND

Traitement des commandes Conman

Le programme **conman** exécute les commandes qui modifient l'état des objets, telles que le démarrage ou l'arrêt d'un poste de travail, ainsi que les commandes qui modifient les objets du plan de façon *asynchrone*. Cela explique le fait que vous observerez sans doute un décalage entre le moment où vous soumettez la commande et celui où les informations stockées dans le fichier Symphony sont mises à jour avec le résultat de la commande.

Ce décalage est dû au fait que le programme **conman** ne met pas à jour les informations dans le fichier Symphony ; **conman** soumet les commandes à **batchman** qui est le seul processus capable d'accéder aux informations du fichier Symphony et de les mettre à jour. Pour cette raison, vous devez attendre que **batchman** traite la demande de modification d'objet émise par **conman** et mette à jour les informations relatives à l'objet stockées dans le fichier Symphony avant de voir les informations mises à jour dans la sortie de la commande *showobj*.

Tout changement effectué à l'aide du programme **conman** qui affecte le fichier Symphony s'applique également aux informations du plan répliquées dans la base de données.

Par exemple, si vous demandez la suppression d'une dépendance à l'aide de la commande **conman deldep**, **conman** soumet la commande **deldep** en publiant un événement dans la boîte aux lettres Mailman.msg. Le processus **mailman** reçoit les informations relatives à la demande de suppression à partir de Mailman.msg et les place dans la boîte aux lettres Intercom.msg sur le poste de travail auquel appartient la ressource dont vous supprimez la dépendance. Sur chaque poste de travail, **batchman** reçoit les événements dans sa boîte aux lettres Intercom.msg et les traite selon leur ordre de réception. Dans le cas où **batchman** est occupé, pour quelque raison que ce soit, les événements transmettant des demandes d'exécution de commandes **conman** sont mis en file d'attente dans le fichier Intercom.msg en attendant d'être lus et traités par **batchman**.

En outre, lorsque **batchman** traite l'événement, l'utilisateur ne reçoit aucune notification. En conclusion, il se peut que vous supprimiez une dépendance et que la suppression n'apparaisse pas parce que **batchman** était trop occupé pour exécuter immédiatement l'opération demandée. Si vous réexécutez la commande, il se peut que la suppression soit déjà effective bien qu'un message indiquant que la commande a été transmise à **batchman** s'affiche dans l'invite **conman**.

Sélection de travaux dans les commandes

Pour les commandes qui s'exécutent sur les travaux, les travaux cible sont sélectionnés au moyen d'attributs et de qualificants. La syntaxe de sélection des travaux fournie ci-après est décrite dans les pages suivantes.

Syntaxe

```
[poste_travail_flot_travaux#]{nom_flot_travaux(hhmm[date]).jobname}  
[{|+|~}qualifiant_travail[...]]
```

ou

```
[poste_travail_de_travail#]numéro_travail [|+|~}qualifiant_travail[...]]
```

ou

```
[poste_travail_flot_travaux#]id_flot_travaux.job [|+|~}qualifiant_travail[...]] ;schedid
```

ou :

```
agent_réseau:.[poste_de_travail_flot_travaux#]{nom_flot_travaux(hhmm[date]).nom_travail  
| ID_flot_travaux.nom_travail;schedid}
```

Arguments

poste de travail

Utilisé avec le fichier *flot_travaux.travail*, cet argument indique le nom du poste de travail sur lequel est exécuté le flot de travaux. Utilisé avec l'argument *numéro_travail*, il indique le poste de travail sur lequel est exécuté le travail. Excepté lorsque *schedid* est également utilisé, les caractères génériques sont autorisés. Cet argument peut être obligatoire, selon le poste de travail sur lequel vous lancez la commande, comme suit :

- Si vous lancez la commande sur le poste de travail sur lequel les travaux cible se sont exécutés, l'argument *workstation* est facultatif.
- Si vous lancez la commande sur un poste de travail hébergé, l'argument *workstation* est obligatoire. Les postes de travail hébergés sont :
 - agents étendus
 - agents
 - les pools
 - les pools dynamiques

nom_flot_travaux

Indique le nom du flot de travaux dans lequel s'exécute le travail. Les caractères génériques sont acceptés.

(hhmm [date])

Indique l'heure et la date auxquelles l'instance du flot de travaux est située dans le plan de préproduction. La valeur *hhmm* correspond à la valeur attribuée au mot clé **schedtime** dans la définition du flot de travaux si aucune restriction temporelle **at** n'a été définie. Une fois que le traitement de l'instance du flot de travaux a démarré, la valeur de *hhmm [date]* est définie sur l'heure à laquelle le flot de travaux a démarré. L'utilisation de caractères génériques n'est pas autorisée dans cette zone. Lorsque vous

émettez des commandes **conman** en ligne à partir de l'invite shell, mettez la commande **conman** entre guillemets " ". Par exemple, exécutez la commande suivante comme suit :

```
conman "sj my_workstation#my_js(2101 02/23).@"
```

ID_flot_travaux

Indique l'identificateur unique du flot de travaux. Voir «Arguments», à la page 428 pour plus d'informations sur les identificateurs de flot de travaux.

schedid Indique que l'identificateur du flot de travaux est utilisé pour sélectionner le flot de travaux.

nomtravail

Indique le nom du travail. Les caractères génériques sont acceptés.

numéro_travail

Indique le numéro du travail.

qualifiant_travail

Voir la section suivante .

agent_réseau

Indique le nom de l'agent du réseau IBM Workload Scheduler qui assure l'interface avec le réseau IBM Workload Scheduler distant contenant le travail cible. Les deux-points doubles (::) sont un délimiteur obligatoire. Les caractères génériques sont acceptés. Pour plus d'informations, reportez-vous à Chapitre 21, «Gestion des dépendances interréseaux», à la page 817.

Remarque : IBM Workload Scheduler vous aide à identifier l'instance de flot de travaux correct lorsque la sélection de flot de travaux fournit un résultat ambigu si plus d'une instance répond à vos critères de sélection. Par exemple, lorsque plusieurs instances de WK1#J1 sont incluses dans le plan de production et que, par conséquent, la sélection de flot de travaux fournit un résultat ambigu, l'invite suivante est générée automatiquement pour vous permettre de choisir l'instance appropriée :

```
Process WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAAABTB)]
(tapez "y" pour oui, "n" pour non)?y
Commande réacheminée vers batchman pour WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAAABTB)]
Process WK1#J1[(1010 03/04/06),(0AAAAAAAAAAAAABTC)]
(tapez "y" pour oui, "n" pour non)?n
```

Dans le résultat, seule l'instance du flot de travaux planifiée le (0600 03/04/06) et ayant pour identificateur 0AAAAAAAAAAAAABTB est sélectionnée pour exécuter la commande.

Qualifiants de travaux

Les qualifiants de travaux indiquent les attributs de travaux sur lesquels une commande doit agir. Ils peuvent être précédés d'un signe + ou ~. Si un qualifiant de travail est précédé par un +, alors les travaux contenant cet attribut spécifique sont sélectionnés pour exécuter la commande. Si un qualifiant de travail est précédé par un ~, alors les travaux contenant cet attribut spécifique sont exclus de l'exécution de la commande.

Les mots clés des qualifiants de travaux peuvent être abrégés en conservant autant de premiers caractères que nécessaire pour les distinguer les uns des autres de façon unique.

at[=*heure* | *heure_min*, | *heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut des travaux en fonction de l'heure spécifiée dans la dépendance **at**.

heure

Indique l'heure comme suit :

hhmm[**+n days** | *date*] [**timezone** | **tz** *fuseau_horaire*]

où :

hhmm Heure et minute.

+n days

Occurrence suivante de *hhmm* en *n* nombre de jours.

date Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj[/aa]*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du travail. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure respecte les règles suivantes:

- Lorsque *hhmm* est antérieure à l'heure courante, l'heure de début est le lendemain ; lorsque *hhmm* est postérieure à l'heure courante, l'heure de début est le jour même.
- Lorsque *hhmm* est supérieure à 2400, elle est divisée par 2400. La partie entière du résultat de la division représente le nombre de *jours en +*, tandis que la partie décimale représente l'heure.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les travaux planifiés pour être lancés après cette heure sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les travaux planifiés pour être lancés avant cette heure sont sélectionnés.

Si **at** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sélectionnés sont ceux qui contiennent une dépendance **at**.

Si **at** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sélectionnés sont ceux qui ne contiennent pas de dépendance **at**.

confirmed

Sélectionne ou exclut les travaux qui ont été planifiés à l'aide du mot clé **confirm**.

critical

Permet de sélectionner ou d'exclure les travaux associé au mot clé **critical** dans une définition de flot de travaux.

critnet Permet de sélectionner ou d'exclure les travaux signalés comme **critical** ou qui sont des prédécesseurs de travaux critiques. Ainsi, il s'applique à tous les travaux dont la date de début est critique.

L'heure de début critique d'un travail critique est calculée en soustrayant sa durée estimée de son échéance. L'heure de début critique d'un prédécesseur est calculée en soustrayant sa durée estimée de l'heure de

début critique de son successeur. Dans un réseau critique, les heures de début critiques sont calculées en commençant par le travail critique et en fonctionnant de façon rétroactive le long de l'aligne des prédécesseurs.

deadline[=*heure* | *heure_min*, | ,*heure_max* | *heure_min*,*heure_max*]

Indique le temps nécessaire pour l'achèvement d'un travail.

hhmm[+*n days* | *date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

+*n days*

Décalage en jours à partir de l'heure d'échéance planifiée.

date Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj/aa*.

timezone | **tz** *fuseau_horaire*

Indique le fuseau horaire à utiliser lors du calcul de l'heure d'échéance. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms de fuseaux horaires. Il s'agit par défaut du fuseau horaire correspondant au poste de travail sur lequel le travail ou le flot de travaux est lancé.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les travaux sélectionnés ont une heure d'échéance planifiée qui n'est pas antérieure à cette heure.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les travaux sélectionnés ont une heure d'échéance planifiée qui n'est pas postérieure à cette heure.

every[=*fréquence* | *fréquence_min*, | ,*fréquence_max* | *fréquence_min*,*fréquence_max*]

Sélectionne ou exclut des travaux soumis ou non à une fréquence de répétition.

fréquence

Indique la fréquence d'exécution planifiée, exprimée en heures et en minutes (*hhmm*).

fréquence_min

Indique la limite inférieure d'une plage de fréquence, exprimée au même format que la *fréquence*. Les travaux dont la fréquence de répétition est supérieure ou égale à la fréquence indiquée sont sélectionnés.

fréquence_max

Indique la limite supérieure d'une plage de fréquence, exprimée au même format que la *fréquence*. Les travaux dont la fréquence de répétition est inférieure ou égale à la fréquence indiquée sont sélectionnés.

Si **every** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sélectionnés sont ceux qui contiennent une fréquence de répétition.

Si **every** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sélectionnés sont ceux qui ne contiennent aucune fréquence de répétition.

finished[=*heure* | *heure_min*, | ,*heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut les travaux selon qu'ils sont terminés ou non.

heure Indique l'heure de fin exacte du travail, exprimée comme suit :

hhmm [*date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

date Occurrence suivante de *hhmm* à la date indiquée, exprimée au format *mm/jj/aa*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du travail. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que *l'heure*. Les travaux qui sont terminés à cette heure ou après sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que *l'heure*. Les travaux qui sont terminés à cette heure ou avant sont sélectionnés.

Si **finished** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sélectionnés sont ceux dont l'exécution est terminée.

Si **finished** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sélectionnés sont ceux dont l'exécution n'est pas terminée.

follows=[*agent_réseau::*][*poste_de_travail#*]{*nom_flot_travaux*(*hhmm* [*mm*]/*jj* [*aa*])}[*.travail* | @] | *id_flot_travaux.travail*;**schedid** | *travail*];**nocheck**][,...]

Sélectionne ou exclut les travaux selon qu'ils comportent ou non une dépendance de prédécesseur/successeur.

agent_réseau

Indique le nom de l'agent du réseau IBM Workload Scheduler qui assure l'interface avec le réseau IBM Workload Scheduler distant contenant le travail ou le flot de travaux prérequis. Les caractères génériques sont acceptés. Pour plus d'informations, reportez-vous à Chapitre 21, «Gestion des dépendances interréseaux», à la page 817.

Poste de travail

Indique le nom du poste de travail sur lequel le travail prérequis est exécuté. Les caractères génériques sont acceptés.

nom_flot_travaux

Indique le nom du flot de travaux dans lequel le travail prérequis est exécuté. Les caractères génériques sont acceptés. Si vous entrez *nom_flot_travaux.@*, indiquez que le travail cible vient à la suite de tous les travaux du flot de travaux.

nomtravail

Indique le nom du travail prérequis. Lorsque ce paramètre est entré sans *nom_flot de travaux*, il indique que le travail prérequis se trouve dans le même flot de travaux que le travail cible. Pour une dépendance de prédécesseur/successeur, n'utilisez pas de caractères génériques dans le nom du travail.

ID_flot_travaux

Indique l'identificateur unique du flot de travaux. Voir «Arguments», à la page 428 pour plus d'informations sur les identificateurs de flot de travaux.

schedid

Ce mot clé, s'il est présent, s'applique à tous les identificateurs de flots de travaux spécifiés dans la clause et indique que pour tous les flots de travaux spécifiés, vous utilisez les arguments *ID_flot_travaux* et non les arguments *nom_flot_travaux*. Si vous souhaitez sélectionner des flots de travaux à l'aide de l'argument *ID_flot_travaux* et certains flots de travaux à l'aide de l'argument *nom_flot_travaux*, vous devez utiliser deux clauses **follows** différentes : l'une contenant les flots de travaux identifiés par l'argument *nom_flot_travaux* sans les mots clés **schedid** et l'autre contenant les flots de travaux identifiés par l'argument *ID_flot_travaux* avec le mot clé **schedid**.

nocheck

Valide uniquement pour les commandes de soumission et utilisé en association avec le mot clé **schedid**. Le mot clé **nocheck** indique que **conman** n'a pas besoin de vérifier l'existence du travail prérequis *ID_flot_travaux.travail* dans le fichier Symphony. Le programme considère que *ID_flot_travaux.travail* existe et dans le cas contraire **batchman** imprime un message d'avertissement dans `stdlist`.

Si **follows** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sont sélectionnés s'ils contiennent des dépendances de prédécesseur/successeur.

Si **follows** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sont sélectionnés s'ils ne contiennent aucune dépendance de prédécesseur/successeur.

logon=*nom_utilisateur*

Sélectionnez les travaux en fonction des noms d'utilisateur sous lesquels ils sont exécutés. Si le *nom_utilisateur* contient des caractères spéciaux, il doit être placé entre guillemets ("). Les caractères génériques sont acceptés.

needs[=*poste_de_travail*#]*nom_ressource*]

Sélectionne ou exclut des travaux selon qu'ils comportent ou non une dépendance de ressource.

Poste de travail

Indique le nom du poste de travail sur lequel la ressource est définie. Les caractères génériques sont acceptés.

nom_ressource

Indique le nom de la ressource. Les caractères génériques sont acceptés.

Si **needs** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sont sélectionnés s'ils contiennent des dépendances resource.

Si **needs** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sont sélectionnés s'ils ne contiennent aucune dépendance resource.

opens[=*poste_de_travail*#]*nom_fichier*[(*qualifiant*)]]

Sélectionne des travaux selon qu'ils comportent ou non une dépendance file. Une dépendance file se produit lorsque la présence d'un ou de plusieurs fichiers permet l'exécution d'un travail ou d'un flot de travaux.

Poste de travail

Indique le nom du poste de travail sur lequel se trouve le fichier. Les caractères génériques sont acceptés.

nom_fichier

Indique le nom du fichier. Ce nom doit être placé entre guillemets (") s'il contient des caractères spéciaux autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inversées (\) et traits de soulignement (_). Les caractères génériques sont acceptés.

qualifier

Condition de test valide. Si elle est omise, les travaux sont sélectionnés ou exclus, quel que soit le qualifiant utilisé.

Si **opens** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sont sélectionnés s'ils contiennent des dépendances file.

Si **opens** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sont sélectionnés s'ils ne contiennent aucune dépendance file.

priority=*pri* | *pri_min*, | *pri_max* | *pri_min*,*pri_max*

Sélectionne ou exclut des travaux en fonction de leurs priorités.

pri Indique la valeur de la priorité. Vous pouvez entrer une valeur comprise entre 0 et 99, **hi** ou **go**.

pri_min

Indique la limite inférieure d'une plage de priorité. Les travaux dont la priorité est égale ou supérieure à cette valeur sont sélectionnés.

pri_max

Indique la limite supérieure d'une plage de priorité. Les travaux dont la priorité est inférieure ou égale à cette valeur sont sélectionnés.

prompt[=*nom_invite* | *num_message*]

Sélectionne ou exclut les travaux selon qu'ils comportent ou non une dépendance prompt.

nom_invite

Indique le nom d'une invite globale. Les caractères génériques sont acceptés.

msgnum

Indique le numéro de message d'une invite locale.

Si **prompt** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sont sélectionnés s'ils contiennent des dépendances prompt.

Si **prompt** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sont sélectionnés s'ils ne contiennent aucune dépendance prompt.

recovery=*option_reprise*

Sélectionne ou exclut des travaux en fonction de leurs options de reprise.

recv-option

Affecte à l'option de reprise de travaux la valeur **stop**, **continue** ou **rerun**.

scriptname=*nom_fichier*

Sélectionne ou exclut des travaux en fonction du nom de leurs fichiers exécutables.

nom_fichier

Indique le nom d'un fichier exécutable. Ce nom doit être placé entre guillemets (") s'il contient des caractères spéciaux autres que

les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inversées (\) et traits de soulignement (_). Les caractères génériques sont acceptés.

started[=*heure* | *heure_min*, | *heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut des travaux selon qu'ils ont été lancés ou non.

heure Indique l'heure de début exacte des flots de travaux, exprimée comme suit :

hhmm [*date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

date Occurrence suivante de *hhmm* à la date indiquée, exprimée au format *mm/jj/aa*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du travail. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Uniquement les travaux qui ont été lancés à cette heure ou après sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Uniquement les travaux qui ont été lancés à cette heure ou avant sont sélectionnés.

Si **started** est utilisé seul et qu'il est précédé du caractère +, alors uniquement les travaux dont l'exécution a démarré à cette heure sont sélectionnés.

Si **started** est utilisé seul et qu'il est précédé du caractère ~, alors uniquement les travaux dont l'exécution a démarré à cette heure ou ultérieurement et est toujours en cours sont sélectionnés.

state=*état*[,...]

Sélectionne ou exclut des travaux en fonction de leur état.

état Indique l'état courant du travail. Les états acceptés pour un travail sont les suivants :

ABEND

Le travail s'est terminé avec un code de sortie différent de zéro.

ABENP

Une confirmation **abend** a été reçue, mais le travail n'est pas terminé.

ADD Le travail est en cours de soumission.

CANCL

Ne s'applique qu'aux dépendances interréseau. Le travail distant ou le flot de travaux a été annulé.

DONE

Le travail est terminé avec un état inconnu.

ERROR

Pour les dépendances interréseaux uniquement, une erreur s'est produite lors de la recherche du statut distant.

EXEC

Le travail est en cours. L'indicateur + en regard de l'état EXEC signifie que le travail est géré par le processus **batchman** local.

EXTRN

Pour les dépendances interréseaux uniquement, l'état est inconnu. Une erreur s'est produite, une action de réexécution vient d'être effectuée sur le travail du flot de travaux EXTERNE, ou le travail ou le flot de travaux distant n'existe pas.

FAIL

Impossible de lancer le travail.

FENCE

La valeur de la priorité du travail est inférieure à la priorité minimale.

HOLD

Le travail attend la résolution des dépendances.

INTRO

Le travail est introduit à des fins de lancement par le système. L'indicateur + en regard de l'état INTRO signifie que le travail est géré par le processus **batchman** local.

PEND

Le travail est terminé et attend une confirmation.

READY

Le travail est prêt à démarrer et toutes les dépendances sont résolues.

SCHED

L'heure **at** du travail n'est pas arrivée.

SUCC

Le travail s'est terminé avec un code de sortie nul.

SUCCP

Une confirmation **succ** a été reçue mais le travail n'est pas terminé.

WAIT

Le travail est à l'état WAIT (agent étendu uniquement).

until[=*heure* | *heure_min*, | ,*heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut des travaux en fonction de l'heure de fin planifiée qui leur est associée.

heure Indique l'heure de fin planifiée exprimée comme suit :

hhmm[+*n days* | *date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

+*n days*

Occurrence suivante de *hhmm* en *n* nombre de jours.

date

Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj[/aa]*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du travail. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les travaux dont l'heure de fin planifiée est égale ou postérieure à l'heure indiquée sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les travaux dont l'heure de fin planifiée est égale ou antérieure à l'heure indiquée sont sélectionnés.

Pour sélectionner une liste d'objets attendus, **hightime** nécessite l'option **date** avec l'heure du système d'exploitation. Cela ne fonctionne pas si l'heure du système d'exploitation en cours est définie plus tard que la valeur **until** *hightime*. Pour afficher les flots de travaux à l'état **ready** ou **hold** avec **until** défini à 12:00, vous devez ajouter la valeur **date** à **until=,1200**.

Si **until** est utilisé seul et qu'il est précédé du caractère +, alors les travaux sont sélectionnés s'ils contiennent une heure **until** spécifiée.

Si **until** est utilisé seul et qu'il est précédé du caractère ~, alors les travaux sont sélectionnés s'ils ne contiennent aucune heure **until** spécifiée.

Sélection de flots de travaux dans les commandes

Pour les commandes qui s'exécutent sur les flots de travaux, les flots de travaux cible sont sélectionnés à travers la spécification d'attributs et de qualifiants.

Etant donné que *date_et_heure_calendrier* est spécifié en minutes, la combinaison de **nom_flot_travaux** et de l'heure *date_et_heure_calendrier* peut ne pas être unique. Pour cette raison, l'argument **ID_flot_travaux** est disponible pour l'utilisateur, soit à des fins d'affichage, soit pour effectuer des actions sur une instance spécifique d'un flot de travaux.

La syntaxe de sélection des flots de travaux fournie ci-après est décrite dans les pages suivantes. Vous pouvez choisir l'une des deux syntaxes décrites.

Syntaxe

[*poste_de_travail*#]*nom_flot_travaux*(*hhmm*[*date*]) [{+ | ~}*qualifiant_flot_travaux*[...]]

[*poste_de_travail*#]*ID_flot_travaux* ;**schedid**

Arguments

poste_de_travail

Indique le nom du poste de travail sur lequel le flot de travaux est exécuté. Excepté lorsque **schedid** est également utilisé, les caractères génériques sont autorisés.

nom_flot_travaux

Indique le nom du flot de travaux. Les caractères génériques sont acceptés.

(*hhmm* [*date*])

Indique l'heure et la date auxquelles l'instance du flot de travaux est située

dans le plan de préproduction. Cette valeur correspond à la valeur affectée au mot clé **schedtime** dans la définition du flot de travaux si aucune restriction temporelle **at** n'a été définie. Une fois que le traitement de l'instance du flot de travaux a démarré, la valeur de *hhmm* [date] est définie sur l'heure à laquelle le flot de travaux a démarré. L'utilisation de caractères génériques n'est pas autorisée dans cette zone. Lorsque vous émettez des commandes **conman** en ligne à partir de l'invite shell, mettez la commande **conman** entre guillemets (""). Par exemple, exécutez la commande suivante comme suit :

```
conman "ss my_workstation#my_js(2101 02/23)"
```

qualifiant_flot_travaux

Voir Identificateurs de flot de travaux ci-après.

ID_flot_travaux

Indique l'identificateur de flot de travaux alphanumérique unique généré automatiquement par le planificateur et affecté à ce flot de travaux. Celui-ci permet essentiellement aux processus internes d'identifier l'instance du flot de travaux dans le plan de production, mais il peut souvent être utilisé lors de la gestion du flot de travaux à partir du programme de ligne de commande **conman** en spécifiant l'option **;schedid**.

schedid

Indique que l'identificateur du flot de travaux est utilisé pour sélectionner le flot de travaux.

Remarque : IBM Workload Scheduler vous aide à identifier l'instance de flot de travaux correct lorsque la sélection de flot de travaux fournit un résultat ambigu si plus d'une instance répond à vos critères de sélection. Par exemple, lorsque plusieurs instances de WK1#J1 sont incluses dans le plan de production et que, par conséquent, la sélection de flot de travaux fournit un résultat ambigu, l'invite suivante est générée automatiquement pour vous permettre de choisir l'instance appropriée :

```
Process WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAAABTB)]
(tapez "y" pour oui, "n" pour non)?y
Commande réacheminée vers batchman pour WK1#J1[(0600 03/04/06),(0AAAAAAAAAAAAABTB)]
Process WK1#J1[(1010 03/04/06),(0AAAAAAAAAAAAABTC)]
(tapez "y" pour oui, "n" pour non)?n
```

Dans le résultat, seule l'instance du flot de travaux planifiée le (0600 03/04/06) et ayant pour identificateur 0AAAAAAAAAAAAABTB est sélectionnée pour exécuter la commande.

Qualifiants de flot de travaux

at[=heure | heure_min, | ,heure_max | heure_min,heure_max]

Sélectionne ou exclut des flots de travaux en fonction de l'heure de début planifiée.

heure Indique l'heure de début planifiée exprimée comme suit :

hhmm[+*n days* | *date*] [*timezone* | *tz fuseau_horaire*]

hhmm Heure et minute.

+*n days*

Occurrence suivante de *hhmm* en *n* nombre de jours.

date Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj[/aa]*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du flot de travaux. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les flots de travaux dont l'heure de début planifiée est égale ou postérieure à l'heure indiquée sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les flots de travaux dont l'heure de début planifiée est égale ou antérieure à l'heure indiquée sont sélectionnés.

Si **at** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sélectionnés sont ceux qui contiennent une dépendance **at**.

Si **at** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sélectionnés sont ceux qui ne contiennent pas de dépendance **at**.

carriedforward

Sélectionne les flots de travaux qui ont été reportés s'il est précédé du caractère +, ou les exclut s'il est précédé de ~.

carryforward

Sélectionne les flots de travaux qui ont été planifiés à l'aide du mot clé **carryforward** s'il est précédé du caractère +, ou les exclut s'il est précédé de ~.

finished[=*heure* | *heure_min*, | ,*heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut les flots de travaux selon qu'ils sont terminés ou non.

heure Indique l'heure de fin exacte des flots de travaux, exprimée comme suit :

hhmm [*date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

date Occurrence suivante de *hhmm* à la date indiquée, exprimée au format *mm/jj/aa*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du flot de travaux. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les flots de travaux qui sont terminés à cette heure ou après sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les flots de travaux qui sont terminés à cette heure ou avant sont sélectionnés.

Si **finished** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sélectionnés sont ceux dont l'exécution est terminée.

Si **finished** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sélectionnés sont ceux dont l'exécution n'est pas terminée.

follows=[agent_réseau::][poste_de_travail#]{nom_flot_travaux(hhmm [mm/jj/Uaa])}[travail
| @] | id_flot_travaux.travail;schedid} | travail[;nocheck] [...]

Sélectionne ou exclut les flots de travaux selon qu'ils comportent ou non une dépendance de prédécesseur/successeur.

agent_réseau

Indique le nom de l'agent de réseau qui assure l'interface avec le réseau IBM Workload Scheduler distant contenant le travail ou le flot de travaux prérequis. Les caractères génériques sont acceptés. Pour plus d'informations sur les agents réseau, reportez-vous à Chapitre 21, «Gestion des dépendances interréseaux», à la page 817.

Poste de travail

Indique le nom du poste de travail sur lequel est exécuté le travail ou le flot de travaux prérequis. Les caractères génériques sont acceptés.

nom_flot_travaux

Indique le nom du flot de travaux prérequis. Les caractères génériques sont acceptés.

nomtravail

Indique le nom du travail prérequis. Les caractères génériques sont acceptés.

ID_flot_travaux

Indique l'identificateur unique du flot de travaux. Voir «Arguments», à la page 428 pour plus d'informations sur les identificateurs de flot de travaux.

schedid

Ce mot clé, s'il est présent, s'applique à tous les identificateurs de flots de travaux spécifiés dans la clause et indique que pour tous les flots de travaux spécifiés, vous utilisez les arguments *ID_flot_travaux* et non les arguments *nom_flot_travaux*. Si vous souhaitez sélectionner des flots de travaux à l'aide de l'argument *ID_flot_travaux* et certains flots de travaux à l'aide de l'argument *nom_flot_travaux*, vous devez utiliser deux clauses **follows** différentes : l'une contenant les flots de travaux identifiés par l'argument *nom_flot_travaux* sans les mots clés **schedid** et l'autre contenant les flots de travaux identifiés par l'argument *ID_flot_travaux* avec le mot clé **schedid**.

nocheck

Valide uniquement pour les commandes de soumission et utilisé en association avec le mot clé **schedid**. Le mot clé **nocheck** indique que **conman** n'a pas besoin de vérifier l'existence du travail prérequis *ID_flot_travaux.travail* dans le fichier Symphony. Le programme considère que *ID_flot_travaux.travail* existe et dans le cas contraire **batchman** imprime un message d'avertissement dans `stdlist`.

Si **follows** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sont sélectionnés s'ils contiennent des dépendances de prédécesseur/successeur.

Si **follows** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sont sélectionnés s'ils ne contiennent aucune dépendance de prédécesseur/successeur.

limit[=*limite* | *limite_min*, | ,*limite_max* | *limite_min*,*limite_max*]

Sélectionne ou exclut les flots de travaux selon qu'ils comportent ou non un nombre maximal de travaux.

limit Indique la valeur précise du nombre maximal de travaux.

limite_min

Indique la limite inférieure d'une plage. Les flots de travaux dont le nombre maximal de travaux est égal ou supérieur au nombre maximal indiqué sont sélectionnés.

limite_max

Indique la limite supérieure d'une fourchette. Les flots de travaux dont le nombre maximal de travaux est inférieur ou égal au nombre maximal indiqué sont sélectionnés.

Si **limit** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sont sélectionnés s'ils sont associés à un nombre maximal de travaux.

Si **limit** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sont sélectionnés s'ils ne sont associés à aucun nombre maximal de travaux.

needs[=*poste_de_travail#*]*nom_ressource*]

Sélectionne ou exclut les flots de travaux selon qu'ils comportent ou non une dépendance de ressource.

Poste de travail

Indique le nom du poste de travail sur lequel la ressource est définie. Les caractères génériques sont acceptés.

nom_ressource

Indique le nom de la ressource. Les caractères génériques sont acceptés.

Si **needs** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sont sélectionnés s'ils contiennent des dépendances resource.

Si **needs** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sont sélectionnés s'ils ne contiennent aucune dépendance resource.

opens[=*poste_de_travail#*]*nom_fichier*[(*qualifiant*)]]

Sélectionne ou exclut les flots de travaux selon qu'ils comportent ou non une dépendance de fichier. Une dépendance de fichier se produit lorsque la présence d'un ou de plusieurs fichiers permet l'exécution d'un travail ou d'un flot de travaux.

Poste de travail

Indique le nom du poste de travail sur lequel se trouve le fichier. Les caractères génériques sont acceptés.

nom_fichier

Indique le nom du fichier. Ce nom doit être placé entre guillemets (") s'il contient des caractères spéciaux autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inversées (\) et traits de soulignement (_). Les caractères génériques sont acceptés.

qualifier

Condition de test valide. En cas d'omission, les flots de travaux sont sélectionnés ou exclus, quel que soit le qualifiant utilisé.

Si **opens** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sont sélectionnés s'ils contiennent des dépendances file.

Si **opens** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sont sélectionnés s'ils ne contiennent aucune dépendance file.

priority=*pri* | *pri_min*, | ,*pri_max* | *pri_min*,*pri_max*

Sélectionne ou exclut les flots de travaux en fonction de leurs priorités.

pri Indique la valeur de la priorité. Vous pouvez entrer une valeur comprise entre 0 et 99, **hi** ou **go**.

pri_min

Indique la limite inférieure d'une plage de priorité. Les flots de travaux dont la priorité est égale ou supérieure à cette valeur sont sélectionnés.

pri_max

Indique la limite supérieure d'une plage de priorité. Les flots de travaux dont la priorité est inférieure ou égale à cette valeur sont sélectionnés.

prompt[=*nom_invite* | *num_message*]

Sélectionne ou exclut les flots de travaux selon qu'ils comportent ou non une dépendance d'invite.

nom_invite

Indique le nom d'une invite globale. Les caractères génériques sont acceptés.

msgnum

Indique le numéro de message d'une invite locale.

Si **prompt** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sont sélectionnés s'ils contiennent des dépendances prompt.

Si **prompt** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sont sélectionnés s'ils ne contiennent aucune dépendance prompt.

started[=*heure* | *heure_min*, | ,*heure_max* | *heure_min*,*heure_max*]

Sélectionne ou exclut les flots de travaux selon qu'ils ont été lancés ou non.

heure Indique l'heure de début exacte du flot de travaux, exprimée comme suit :

hhmm [*date*] [**timezone** | **tz** *fuseau_horaire*]

hhmm Heure et minute.

date Occurrence suivante de *hhmm* à la date indiquée, exprimée au format *mm/jj/aa*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du flot de travaux. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les flots de travaux qui sont lancés à cette heure ou après sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les flots de travaux qui sont lancés à cette heure ou avant sont sélectionnés.

Si **started** est utilisé seul et qu'il est précédé du caractère +, alors les flots de travaux sélectionnés sont ceux dont l'exécution a démarré.

Si **started** est utilisé seul et qu'il est précédé du caractère ~, alors les flots de travaux sélectionnés sont ceux dont l'exécution n'a pas démarré.

state=état[,...]

Sélectionne ou exclut les flots de travaux en fonction de leur état.

état Indique l'état courant du flot de travaux. Les états des flots de travaux sont les suivants :

ABEND

Le flot de travaux s'est arrêté de manière anormale.

ADD Le flot de travaux vient d'être soumis.

EXEC Le flot de travaux est en cours d'exécution.

EXTRN

Ne s'applique qu'aux dépendances interréseau. Il s'agit de l'état du flot de travaux EXTERNAL contenant des travaux qui font référence à des travaux ou des flots de travaux dans le réseau distant.

HOLD

Le flot de travaux attend la résolution des dépendances.

READY

Le flot de travaux est prêt à démarrer et toutes les dépendances sont résolues.

STUCK

L'exécution est interrompue. Aucun travail n'est lancé sans l'intervention de l'opérateur.

SUCC Le flot de travaux s'est achevé correctement.

until[=heure | heure_min, | heure_max | heure_min,heure_max]

Sélectionne ou exclut les flots de travaux en fonction de l'heure de fin planifiée.

heure Indique l'heure de fin planifiée exprimée comme suit :

hhmm[+*n days* | *date*] [**timezone** | *tz fuseau_horaire*]

hhmm Heure et minute.

+*n days*

Occurrence suivante de *hhmm* en *n* nombre de jours.

date Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj[/aa]*.

timezone | **tz** *fuseau_horaire*

Nom du fuseau horaire du flot de travaux. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour les noms valides.

heure_min

Indique la limite inférieure d'une plage horaire exprimée au même format que l'*heure*. Les flots de travaux dont l'heure de fin planifiée est égale ou postérieure à l'heure indiquée sont sélectionnés.

heure_max

Indique la limite supérieure d'une plage horaire, exprimée au même format que l'*heure*. Les flots de travaux dont l'heure de fin planifiée est égale ou antérieure à l'heure indiquée sont sélectionnés.

Si **until** est utilisé seul et qu'il est précédé du caractère +, les flots de travaux sélectionnés sont ceux qui contiennent une heure de fin planifiée.

Si **until** est utilisé seul et qu'il est précédé du caractère ~, les flots de travaux sélectionnés sont ceux qui ne contiennent aucune heure de fin planifiée.

Gestion des travaux et des flots de travaux des agents de niveau antérieur

Le changement apporté par la version 8.3 de IBM Workload Scheduler à la convention de nom des instances de flot de travaux nécessite que vous appliquiez le contournement suivant lors de l'émission de commandes de ligne de commande sur un plan généré sur un gestionnaire de domaine maître IBM Workload Scheduler de version 8.3 (ou ultérieure) à partir d'agents IBM Workload Scheduler version 8.1, 8.2, or 8.2.1 :

- Vous devez utiliser le symbole @ comme premier caractère de l'identificateur de l'instance du flot de travaux. Par exemple, le flot de travaux qui s'exécute sur le poste de travail CPU1 avec l'identificateur 0AAAAAAAAAAAAAY3 doit être identifié dans la ligne de commande **conman** comme suit :

CPU1#@AAAAAAAAAAAAAY3

- Vous ne pouvez pas utiliser le mot clé **follows** lorsque vous ajoutez une dépendance à un travail ou un flot de travaux ou lorsque vous soumettez une commande ou un fichier comme travail.
- Vous ne pouvez pas utiliser le mot clé **into** pour définir le flot de travaux dans lequel le travail doit être ajouté lorsque vous soumettez une commande ou un fichier comme travail.

Par exemple, pour afficher les informations sur le travail job2 contenu dans l'instance de flot de travaux avec l'identificateur 0AAAAAAAAAAAAAT1 s'exécutant sur le poste de travail CPU1, exécutez la commande suivante sur les agents de IBM Workload Scheduler version 8.1, 8.2 ou 8.2.1 :

sj CPU1#@AAAAAAAAAAAAAT1.job2

Ces modifications seront également visibles dans les rapports et les journaux, ainsi qu'à tous les autres endroits où les noms de flot de travaux sont imprimés ou affichés.

Codes retour de Conman

Codes retour de Conman

Gestion des codes retour de Conman

Lorsque vous exécutez une commande **conman**, la ligne de commande peut afficher un code retour de résultat. Pour trouver le code retour, procédez comme suit :

Sur les systèmes d'exploitation Windows :

```
echo %ERRORLEVEL%
```

Sur les systèmes d'exploitation UNIX :

```
echo $?
```

La ligne de commande **conman** aboutit aux codes retour suivant pour les commandes **submit sched** et **submit job** :

- 0 La commande a abouti.
- 10 La commande **submit sched** a rencontré une erreur.
- 11 La commande **submit job** a rencontré une erreur.

Toutes les autres commandes renvoient toujours le code retour 0.

Commandes Conman

Le tableau 63 répertorie les commandes **conman**. Les noms de commande et les mots clés peuvent être entrés en majuscules ou en minuscules et peuvent être abrégés au nombre minimal de caractères suffisant pour les distinguer les uns des autres de façon unique. Certains noms de commandes ont également une forme abrégée spécifique.

Remarque : Les types de poste de travail dans le tableau suivant ont les significations suivantes :

- D** Gestionnaires de domaine dynamique, gestionnaires de domaine de sauvegarde
- M** Gestionnaires de domaine maître et maîtres de sauvegarde
- F** Gestionnaires de domaine et agents tolérants aux pannes
- T** Agents tolérants aux pannes
- S** Agents standard (vous pouvez uniquement afficher des fichiers sur un agent standard)

Tableau 63. Liste des commandes Conman

Commande	Forme abrégée	Description	Type	Page
adddep { job sched }	adj ads	Ajoute des dépendances d'un travail ou d'un flot de travaux.	F	«adddep job», à la page 439 «adddep sched», à la page 442
altpass		Modifie un mot de passe de définition d'objet utilisateur.	F	«altpass», à la page 444
altpri	ap	Modifie les priorités d'un travail ou d'un flot de travaux.	F	«altpri», à la page 445
bulk_discovery	bulk	Exécute une reconnaissance avec accès simultané. A utiliser avec IBM Tivoli Monitoring 6.1 (Tivoli Enterprise Portal).	F	«bulk_discovery», à la page 446
cancel { job sched }	cj cs	Annule un travail ou un flot de travaux.	F	«cancel job», à la page 446 «cancel sched», à la page 448

Tableau 63. Liste des commandes Conman (suite)

Commande	Forme abrégée	Description	Type	Page
checkhealthstatus	chs	Appelle le service chkhlst afin de vérifier que la boîte aux lettres peut être lue correctement par mailman ou s'il y a des erreurs dans l'en-tête de boîte aux lettres.	MFS	«checkhealthstatus», à la page 450
confirm	conf	Confirme l'achèvement d'un travail.	F	«confirm», à la page 450
console	cons	Affecte la console IBM Workload Scheduler.	FS	«console», à la page 453
continue	cont	Ignore l'erreur suivante.	FS	«continue», à la page 454
deldep { job sched }	ddj dds	Supprime les dépendances du travail ou du flot de travaux.	F	«deldep job», à la page 455 «deldep sched», à la page 457
deployconf	deploy	Obtient la configuration de surveillance la plus récente pour le moteur de surveillance d'événement sur le poste de travail.	FS	«deployconf», à la page 458
display { file job sched }	df dj ds	Affiche les fichiers, les travaux et les flots de travaux.	FS	«display», à la page 459
exit	e	Quitte le programme conman .	FS	«exit», à la page 461
fence	d	Définit la priorité minimale de travail de IBM Workload Scheduler.	F	«fence», à la page 462
help⁽¹⁾	h	Affiche les informations relatives à la commande.	FS	«help», à la page 463
kill	k	Arrête l'exécution d'un travail.	F	«kill», à la page 464
limit { cpu sched }	lc ls	Modifie le nombre maximal de travaux d'un poste de travail ou d'un flot de travaux.	F	«limit cpu», à la page 465 «limit sched», à la page 467
link	lk	Etablit des liaisons de poste de travail.	FS	«link», à la page 468
listsym	lis	Affiche la liste de fichiers journaux Symphony.	F	«listsym», à la page 470
listsucc		Répertorie les successeurs d'un travail.	F	«Listsucc», à la page 472
recall	rc	Affiche les messages d'invite.	F	«recall», à la page 474
redo	red	Edite la commande précédente.	FS	«redo», à la page 475
release { job sched }	rj rs	Libère les dépendances du travail ou du flot de travaux.	F	«release job», à la page 476 «release sched», à la page 478
reply	rep	Répond au message d'invite.	F	«reply», à la page 480
rerun	rr	Lance une nouvelle exécution d'un travail.	F	«rerun», à la page 481
rerunsucc		Réexécute un travail et exécute ses successeurs.	F	«Rerunsucc», à la page 484

Tableau 63. Liste des commandes Conman (suite)

Commande	Forme abrégée	Description	Type	Page
resetFTA	N/A	Récupère un fichier Symphony endommagé sur l'agent tolérant aux pannes spécifié	T	«resetFTA», à la page 487
resource	res	Modifie le nombre d'unités de ressource.	F	«resource», à la page 488
setsym	set	Sélectionne un fichier journal Symphony.	F	«setsym», à la page 489
showcpus	sc	Affiche les informations relatives au poste de travail et aux liaisons.	FS	«showcpus», à la page 490
showdomain	showd	Affiche les informations relatives au domaine.	FS	«showdomain», à la page 497
showfiles	sf	Affiche les informations relatives aux fichiers.	F	«showfiles», à la page 498
showjobs	sj	Affiche les informations relatives aux travaux.	F	«showjobs», à la page 501
showprompts	sp	Affiche les informations relatives aux invites.	F	«showprompts», à la page 520
showresources	sr	Affiche les informations relatives aux ressources.	F	«showresources», à la page 523
showschedules	ss	Affiche les informations relatives aux flots de travaux.	F	«showschedules», à la page 525
shutdown	shut	Arrête les processus de production de IBM Workload Scheduler.	FS	«shutdown», à la page 531
start		Lance les processus de production de IBM Workload Scheduler.	FS	«start», à la page 532
startappserver		Démarre le processus WebSphere Application Server	DM	«startappserver», à la page 534
startbrokerapp		Démarre l'application Dynamic Workload Broker.	DM	«startbrokerapp», à la page 535
starteventprocessor	startevtp	Démarre le serveur de traitement d'événements.	M ⁽²⁾	«starteventprocessor», à la page 535
startmon	startm	Démarre le processus monman qui active le moteur de surveillance d'événements sur l'agent.	FS	«startmon», à la page 536
status	stat	Affiche l'état de la production de IBM Workload Scheduler.	FS	«status», à la page 537
stop		Arrête les processus de production de IBM Workload Scheduler.	FS	«stop», à la page 538
stop ;progressive		Stoppe les processus de production de IBM Workload Scheduler de façon hiérarchique.		«stop ;progressive», à la page 540
stopappserver	stopapps	Arrête le processus WebSphere Application Server	DM	«stopappserver», à la page 541

Tableau 63. Liste des commandes Conman (suite)

Commande	Forme abrégée	Description	Type	Page
stopbrokerapp		Arrête l'application Dynamic Workload Broker.	DM	«stopbrokerapp», à la page 542
stopeventprocessor	stopevtp	Arrête le serveur de traitement d'événements.	M ⁽²⁾	«stopeventprocessor», à la page 543
stopmon	stopm	Arrête le moteur de surveillance d'événements sur l'agent.	FS	«stopmon», à la page 544
submit { docommand file job sched }	sbd sbf sbj sbs	Soumet une commande, un fichier, un travail ou un flot de travaux.	FS ⁽³⁾	«submit docommand», à la page 545 «submit file», à la page 549 «submit job», à la page 552 «submit sched», à la page 556
switcheventprocessor	switchevtp	Fait passer le serveur de traitement d'événements des gestionnaires de domaine maîtres aux maîtres de sauvegarde ou inversement.	M	«switcheventprocessor», à la page 561
switchmgr	switchm	Change le gestionnaire de domaine.	F	«switchmgr», à la page 562
<i>commande système</i>		Envoie une commande au système.	FS	«Commande système», à la page 563
tellop	to	Envoie un message à la console.	FS	«tellop», à la page 564
unlink		Ferme les liaisons de poste de travail.	FS	«unlink», à la page 565
version	v	Affiche la bannière du programme de ligne de commande conman .	FS	«version», à la page 567

1. Non disponible sur le système d'exploitation Windows pris en charge.
2. Inclut les postes de travail installés comme maîtres de sauvegarde, mais utilisés comme agents tolérants aux pannes ordinaires.
3. Vous pouvez utiliser les commandes **submit job (sbj)** et **submit sched (sbs)** sur un agent standard en utilisant les paramètres de connexion ou en spécifiant les paramètres dans le fichier `useropts` lors de l'appel de la ligne de commande **conman**.

Remarque : Dans les commandes, les termes *sched* et *schedule* se rapportent aux *flots de travaux*, et le terme *CPU* aux *postes de travail*.

adddep job

Ajoute des dépendances à un travail.

Vous devez avoir un accès **adddep** au travail. Pour inclure des dépendances needs et prompt, vous devez avoir un accès **use** aux ressources et aux invites globales.

Syntaxe

```
{adddep job  
| adj} travail_sélectionné  
;dépendance[;...]  
[;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

dépendance

Type de dépendance. Indiquez l'un des éléments suivants. Les caractères génériques ne sont pas admis.

at=*hhmm*[**timezone** | **tz** fuseau_horaire][**+n days** | *mm/jj/aa*] | [**absolute** | **abs**]

confirmed

deadline=*heure* [**timezone** | **tz** fuseau_horaire][**+n day[s** | *mm/jj/aa*]

every=*fréquence*

follows=[*agent_réseau::*][*poste_de_travail#*]{*nom_flot_travaux*[*hhmm* [*mm/dd/yy*]]][*.travail* | @] | *id_flot_travaux.job*;**schedid**] | *travail*[...] [**if** *nom_condition*[| *nom_condition*][| ...]']

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

maxdur=[*hhmm*] [**onmaxdur** *action*]

mindur=[*hhmm*] [**onmindur** *action*]

needs=[*num*] [*poste_de_travail#*]*ressource*[,...]

opens=[*poste_travail#*]"*nom_fichier*"[(*qualificatif*)][,...] **priority**[=*pri* | **hi** | **go**]

prompt="[: | !]*texte*" | *nom_invite*[,...]

until *heure* [**timezone** | **tz** fuseau_horaire][**+n day[s]**] | [**absolute** | **abs**]
[;**onuntil** *action*]

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Remarque :

1. Si vous ajoutez deux fois une dépendance sur un flot de travaux à un travail, les deux dépendances sont traitées.
2. Lorsque vous utilisez le mot clé **deadline**, veillez à ce qu'une valeur supérieure à 0 soit attribuée à l'option **bm check deadline** dans le fichier de configuration *localopts* sur les postes de travail que vous utilisez. Définissez l'option **bm check deadline** sur chaque poste de travail sur lequel vous souhaitez être

notifié de l'expiration du délai, ou, si vous souhaitez obtenir des informations à jour sur l'ensemble de l'environnement, définissez l'option dans le gestionnaire de domaine maître. Les échéances des travaux critiques sont évaluées automatiquement, indépendamment de l'option **bm check deadline**.

Pour plus d'informations sur l'option **bm check deadline**, voir la section relative aux détails de localopts dans *IBM Workload Scheduler - Guide d'administration*.

3. Si vous ajoutez une dépendance à un travail après que celui-ci est terminé, elle n'est pas évaluée. Toutefois, les réexecutions suivantes du travail traiteront la dépendance correctement.

Commentaires

Si aucune valeur n'est indiquée pour **priority**, le travail retourne à sa priorité planifiée initialement. Si aucun poste de travail n'est indiqué dans **follows**, **needs** ou **opens**, le poste de travail sur lequel le travail s'exécute est utilisé par défaut.

Vous ne pouvez pas utiliser cette commande pour ajouter une ressource ou une invite sous la forme de dépendances **resource** ou **prompt**, sauf si ces dernières sont déjà référencées par un travail ou un flot de travaux dans le fichier **Symphony**.

Exemples

Pour ajouter une dépendance de ressource au travail **job3** du flot de travaux **sked9(0900 02/19/06)**, exécutez la commande suivante :

```
adj sked9(0900 02/19/16).job3 ; needs=2 tapes
```

Pour ajouter une dépendance de prédécesseur/successeur externe depuis un travail **JOB022** du flot de travaux **MLN#SCHED_02(0600 02/24)** vers **J0BA** du flot de travaux **MLN#NEW_TEST(0900 02/19/16)**, exécutez la commande suivante :

```
adj=MLN#NEW_TEST(0900 02/19/16).J0BA ; follows MLN#SCHED_02(0600 02/24/16).JOB022
```

Pour ajouter une dépendance de fichier et une échéance **until** au travail **j6** du flot de travaux **JS2(0900 02/19/16)**, exécutez la commande suivante :

```
adj=WK1#JS2(0900 02/19/16).j6 ; opens="/usr/lib/prdata/file5"(-s %p) ; until=2330
```

Pour annuler le travail **PAYROLL_JOB** du flot de travaux **ABSENCES_JS** s'il a été exécuté pendant plus de 9 heures et 1 minute, exécutez la commande suivante :

```
adj DUBAI#ABSENCES_JS.PAYROLL_JOB ;maxdur=901 ;onmaxdur kill
```

Voir aussi

A partir de **Dynamic Workload Console**, vous pouvez effectuer les tâches suivantes :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

6. Dans le tableau qui contient la liste des travaux, sélectionnez le travail auquel vous souhaitez ajouter une dépendance, puis cliquez sur **Dépendances....**
7. Dans le panneau des dépendances, développez une section de dépendance, puis cliquez sur le bouton correspondant à l'action de dépendance que vous souhaitez ajouter.

adddep sched

Ajoute des dépendances à un flot de travaux.

Vous devez avoir un accès *adddep* au flot de travaux. Pour inclure des dépendances needs et prompt, vous devez avoir un accès *use* aux ressources et aux invites globales.

Syntaxe

```
{adddep sched | ads} flot_travaux_sélectionné
;dépendance[;...]
[;noask]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

dépendance

Type de dépendance. Indiquez l'un des éléments suivants. Les caractères génériques ne sont pas admis.

at=*hhmm*[**timezone** | **tz** *fuseau_horaire*][**+n days** | *mm/jj[/aa]*] | [**absolute** | **abs**]

carryforward

deadline=*heure* [**timezone** | **tz** *fuseau_horaire*][**+n day[s** | *mm/jj[/aa]*]

follows=[*agent_réseau::*][*poste_travail#*]{*nom_flot_travaux*[*hhmm* [*mm/dd[/yy]*]] [*.travail* | @] | *id_flot_travaux.job*;**schedid**] | *travail*[,...] [**if** *nom_condition*[| *nom_condition*][| ...]']

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction follows, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

limit=*limite*

needs=[*num*] [*poste_de_travail#*]*ressource*[,...]

opens=[*poste_travail#*] "*nom_fichier*"[(*qualificatif*)][,...] **priority**[=*pri* | **hi** | **go**]

prompt="[: | !]*texte*" | *nom_invite*[,...]

until *heure* [**timezone** | **tz** *fuseau_horaire*][**+n jour[s** | [**absolute** | **abs**]] [**onuntil** *action*]

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Remarque :

1. Si vous ajoutez deux fois une dépendance à un flot de travaux sur un autre flot de travaux, seule une dépendance est prise en compte.
2. Lorsque vous utilisez le mot clé **deadline**, veillez à ce qu'une valeur supérieure à 0 soit attribuée à l'option **bm check deadline** dans le fichier de configuration **localopts** sur les postes de travail que vous utilisez. Définissez l'option **bm check deadline** sur chaque poste de travail sur lequel vous souhaitez être notifié de l'expiration du délai, ou, si vous souhaitez obtenir des informations à jour sur l'ensemble de l'environnement, définissez l'option dans le gestionnaire de domaine maître. Les échéances des travaux critiques sont évaluées automatiquement, indépendamment de l'option **bm check deadline**.
Pour plus d'informations sur l'option **bm check deadline**, voir la section relative aux détails de **localopts** dans *IBM Workload Scheduler - Guide d'administration*.
3. Si vous ajoutez une dépendance à un flot de travaux après que celui-ci est terminé, elle n'est pas évaluée. Toutefois, les réexecutions suivantes du flot de travaux traiteront la dépendance correctement.

Commentaires

- Si aucune valeur n'est indiquée pour **priority**, le flot de travaux retourne à sa priorité planifiée à l'origine.
- Si vous n'indiquez aucune valeur pour **limit**, 0 est la valeur par défaut.
- Si aucun poste de travail n'est indiqué dans **follows**, **needs** ou **opens**, le poste de travail sur lequel le flot de travaux s'exécute est utilisé par défaut.
- Vous ne pouvez pas utiliser cette commande pour ajouter une ressource ou une invite sous la forme de dépendances **resource** ou **prompt**, sauf si ces dernières existent déjà dans le plan de production. Pour voir quelle ressource et quelles invites existent dans le plan, voir «**showresources**», à la page 523 and «**showprompts**», à la page 520.

Exemples

Pour ajouter une dépendance d'invite au flot de travaux **sked9(0900 02/19/06)**, exécutez la commande suivante :

```
ads sked9(0900 02/19/06) ; prompt=msg103
```

Pour ajouter une dépendance de prédécesseur/successeur externe depuis un travail **JOB0** du flot de travaux **CPUA#SCHED_02(0600 02/24/06)** et un nombre maximal de travaux vers le flot de travaux **CPUA#TEST(0900 02/19/06)**, exécutez la commande suivante :

```
ads=CPUA#TEST(0900 02/19/06) ; follows CPUA#SCHED_02(0600 02/24/06).JOB0; limit=2
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.

3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les flots de travaux du plan** ou une autre tâche pour surveiller les flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans le tableau qui contient la liste des flots de travaux, sélectionnez le flot de travaux auquel vous souhaitez ajouter une dépendance, puis cliquez sur **Dépendances....**
7. Dans le panneau des dépendances, développez une section de dépendance, puis cliquez sur le bouton correspondant à l'action de dépendance que vous souhaitez ajouter.

altpass

Modifie le mot de passe d'un objet utilisateur dans le plan de production en cours.

Vous devez avoir un accès *altpass* à l'objet utilisateur.

Syntaxe

altpass

```
[poste_de_travail#]
nom_utilisateur
[:"mot_de_passe"]
```

Arguments

poste de travail

Indique le poste de travail sur lequel l'utilisateur est défini. Utilisez des majuscules pour cette zone même si vous avez utilisé des majuscules et des minuscules lorsque vous avez spécifié la valeur *poste_de_travail* dans la définition de l'utilisateur. Pour plus d'informations, consultez «Définition d'utilisateur», à la page 200. Ne spécifiez pas cette zone si l'utilisateur appartient à un domaine Windows géré par Active Directory. Par défaut, le poste de travail est celui sur lequel vous exécutez **conman**.

nom_utilisateur

Indique le nom d'un utilisateur. Utilisez le même nom d'utilisateur défini dans la base de données IBM Workload Scheduler. Les noms d'utilisateurs sont sensibles à la casse. Pour plus d'informations, voir «Définition d'utilisateur», à la page 200.

password

Indique le nouveau mot de passe. Elle doit figurer entre guillemets. Pour n'indiquer aucun mot de passe pour l'utilisateur, utilisez deux guillemets consécutifs ("").

Commentaires

Si vous n'indiquez aucun *mot de passe*, **conman** vous invite à entrer un mot de passe et une confirmation. Le mot de passe ne s'affiche pas lorsqu'il est saisi et ne doit pas être placé entre guillemets. Il est à noter que la modification s'effectue uniquement dans le plan de production en cours. Elle est donc temporaire. Pour apporter un changement permanent, voir «Définition d'utilisateur», à la page 200.

Exemples

Pour remplacer le mot de passe de l'utilisateur Jim sur le poste de travail mis5 par mynewpw, exécutez la commande suivante :

```
altpass MIS5#JIM;"mynewpw"
```

Pour remplacer le mot de passe de l'utilisateur jim sur le poste de travail Mis5 par mynewpw sans l'afficher, exécutez la commande suivante :

```
altpass MIS5#JIM  
password: xxxxxxxx  
confirm: xxxxxxxx
```

Pour remplacer le mot de passe de l'utilisateur Jim, défini dans un domaine Windows géré par Active Directory nommé twsDom, par mynewpw, exécutez la commande suivante :

```
altpass TWSDOM\JIM;"mynewpw"
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section relative à la modification du mot de passe dans le plan.

altpri

Modifie la priorité d'un travail ou d'un flot de travaux.

Vous devez avoir un accès *altpri* au travail ou au flot de travaux.

Syntaxe

```
{altpri | ap}  
travail_sélectionné | flot_travaux_sélectionné  
[;pri]  
[;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

pri

Indique le niveau de priorité. Vous pouvez entrer une valeur comprise entre 0 et 99, **hi** ou **go**.

noask

Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail ou flot de travaux éligible.

Exemples

Pour modifier la priorité du travail balance du flot de travaux glmonth(0900 02/19/06), exécutez la commande suivante :

```
ap glmonth(0900 02/19/06).balance;55
```

Pour modifier la priorité du flot de travaux `glmonth(0900 02/19/06)`, exécutez la commande suivante :

`ap glmonth(0900 02/19/06);10`

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail** ou **Flot de travaux**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller des travaux ou des flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des résultats, sélectionnez le travail ou le flot de travaux dont vous souhaitez modifier la priorité et cliquez sur l'onglet **Actions supplémentaires** > **Priorité**.
7. Dans le panneau qui s'affiche, définissez la nouvelle priorité et cliquez sur **OK**.

bulk_discovery

Demande une reconnaissance avec accès simultané pour mettre à jour le statut en cours des objets surveillés. Elle est utilisée pour l'intégration à IBM Tivoli Monitoring 6.1 (Tivoli Enterprise Portal).

Vous devez avoir l'accès *display* à l'objet fichier.

Syntaxe

`{bulk_discovery | bulk}`

Commentaires

Lorsque l'intégration avec IBM Tivoli Monitoring 6.1 est activée, la commande **bulk_discovery** vérifie le statut de tous les travaux et flots de travaux surveillés du plan et écrit les événements correspondants dans le fichier journal des événements.

Par défaut, les événements sont écrits dans le fichier **event.log** .

les messages indiquant le début et la fin de l'activité de reconnaissance avec accès simultané sont journalisés dans le fichier `twsmmerge.log`.

cancel job

Windows Annule un travail.

Vous devez avoir un accès *cancel* au travail.

Syntaxe

```
{cancel job |  
cj} travail_sélectionné  
    [;pend]  
    [;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

pend Annule le travail uniquement après la résolution de ses dépendances.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Si vous annulez un travail avant de l'avoir lancé, il n'est pas lancé. Si vous annulez un travail après l'avoir lancé, il continue son exécution. Si vous annulez un travail en cours d'exécution et qu'il se termine sur un état ABEND, aucune tentative de reprise automatique du travail n'est effectuée.

Si l'option **;pend** n'est pas utilisée, les travaux et les flots de travaux qui dépendent du travail annulé sont libérés immédiatement de la dépendance.

Si l'option **;pend** est indiquée et que le travail n'a pas été lancé, l'annulation est différée jusqu'à ce que toutes les dépendances, y compris l'heure d'exécution définie par l'argument **at**, soient résolues. Une fois que toutes les dépendances sont résolues, le travail est annulé et les travaux ou les flots de travaux qui dépendent du travail annulé sont libérés de la dépendance. Lors de la période pendant laquelle l'annulation est différée, la mention **[Cancel Pend]** (annulation en cours) apparaît dans la colonne Dépendances du travail dans un écran **showjobs**.

Si l'option **;pend** est indiquée et que le travail a déjà été lancé, cette option est ignorée et les travaux ou les flots de travaux qui dépendent du travail annulé sont immédiatement libérés de la dépendance.

Vous pouvez utiliser la commande **rerun** pour relancer les travaux qui ont été annulés ou portant la mention **[Cancel Pend]** (annulation en cours). Vous pouvez aussi ajouter et supprimer des dépendances de des travaux associés à la mention **[Cancel Pend]**.

Pour annuler immédiatement un travail associé à la mention **[Cancel Pend]**, entrez une commande de **libération** du travail ou une commande d'**annulation** sans l'option **;pend**.

En ce qui concerne les travaux parvenus à **échéance**, la mention **[Until]** apparaît dans la colonne Dépendances d'un écran **showjobs** et leurs dépendances ne sont plus évaluées. Si un travail de ce type porte également la mention **[Cancel Pend]**, il n'est annulé que lorsque vous libérez ou supprimez l'échéance **until** ou entrez une autre commande **cancel** sans l'option **;pend**.

Pour arrêter l'évaluation des dépendances, affectez à la priorité d'un travail la valeur zéro, à l'aide de la commande **altpri**. Pour reprendre l'évaluation des dépendances, affectez à la priorité une valeur supérieure à zéro.

Remarque : Dans le cas des dépendances interréseau, l'annulation d'un travail du flot de travaux EXTERNE permet de libérer tous les travaux et flots de travaux locaux de la dépendance. Les travaux du flot de travaux EXTERNES correspondent à des travaux et à des flots de travaux qui sont signalés comme étant des dépendances interréseaux. Le statut d'une dépendance interréseau n'est pas vérifié à la suite d'une commande **cancel**. Pour plus d'informations, reportez-vous à «Gestion des dépendances interréseau dans le plan», à la page 822.

Exemples

Pour annuler le travail report du flot de travaux apwkly(0900 02/19/06) sur le poste de travail site3, exécutez la commande suivante :

```
cj=site3#apwkly(0900 02/19/06).report
```

Pour annuler le travail setup du flot de travaux mis5(1100 02/10/06) qui n'est pas à l'état ABEND, exécutez la commande suivante :

```
cj mis5(1100 02/10/06).setup~state=abend
```

Pour annuler le travail job3 du flot de travaux sked3(0900 02/19/03) une fois ses dépendances résolues, exécutez la commande suivante :

```
cj sked3(0900 02/19/06).job3;pend
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
5. A partir de la table contenant la liste des travaux, sélectionnez un travail et cliquez sur **Plus d'actions > Annuler**.

cancel sched

Annule un flot de travaux.

Vous devez avoir un accès *cancel* au flot de travaux.

Syntaxe

```
{cancel sched |  
cs} flot_travaux_sélectionné  
    [;pend]  
    [;noask]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

pend Annule le flot de travaux uniquement après la résolution de ses dépendances.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Commentaires

Si vous annulez un flot de travaux avant de l'avoir lancé, il n'est pas lancé. Si vous annulez un flot de travaux une fois qu'il est lancé, les travaux en cours sont exécutés jusqu'à la fin mais aucun autre travail n'est lancé.

Si l'option **pend** n'est pas utilisée, les travaux et les flots de travaux qui dépendent du flot de travaux annulé sont libérés immédiatement de la dépendance.

Si l'option **pend** est indiquée et que le flot de travaux n'a pas été lancé, l'annulation est différée jusqu'à ce que toutes les dépendances, y compris l'heure d'exécution indiquée par l'argument **at**, soient résolues. Une fois que toutes les dépendances sont résolues, le flot de travaux est annulé et les travaux ou les flots de travaux dépendants sont libérés de la dépendance. Lorsqu'une **annulation** est différée, la mention **[Cancel Pend]** (Annulation en cours) apparaît dans la colonne Dépendances d'un écran **showschedules**.

Si l'option **pend** est indiquée et que le flot de travaux a déjà été lancé, les travaux non encore lancés sont annulés et les travaux et flots de travaux dépendants sont libérés de la dépendance.

Pour annuler immédiatement un flot de travaux marqué **[Cancel Pend]**, entrez une commande de **release** pour le flot de travaux ou une autre commande **cancel** sans l'option **pend**.

Pour arrêter l'évaluation des dépendances, définissez une priorité zéro pour le flot de travaux à l'aide de la commande **altpri**. Pour reprendre l'évaluation des dépendances, affectez à la priorité une valeur supérieure à zéro.

Si le flot de travaux annulé contient des travaux définis avec l'option **every**, seule la dernière instance de ces travaux est signalée comme étant annulée dans un affichage **showjobs**.

Exemples

Pour annuler le flot de travaux sked1(1200 02/17/06) sur le poste de travail site2, exécutez la commande suivante :

```
cs=site2#sked1(1200 02/17)
```

Pour annuler le flot de travaux mis2(0900 02/19/06) s'il est à l'état STUCK, exécutez la commande suivante :

```
cs mis2(0900 02/19)+state=stuck
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.

4. Dans la liste déroulante **Requête**, sélectionnez **Tous les flots de travaux du plan** ou une autre tâche pour surveiller les flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Sélectionnez un flot de travaux et cliquez sur **Plus d'actions** > **Annuler**.

checkhealthstatus

Appelle le service **chkhlst** pour vérifier la connectivité entre le gestionnaire de domaine et les postes de travail. Cette commande vérifie que le fichier Symphony n'est pas endommagé, que les fichiers de boîte aux lettres peuvent être correctement lus par **mailman**, sans erreurs dans l'en-tête de boîte aux lettres, et que la boîte aux lettres n'est pas saturée. Cette commande peut être utile pour diagnostiquer la cause du manque de liaison d'un poste de travail et pour obtenir des suggestions sur la manière de résoudre le problème.

Syntaxe

```
[checkhealthstatus | chs] [poste_de_travail]
```

Commentaires

Si le *poste de travail* n'est pas spécifié, le service est lancé localement.

Exemples

Pour vérifier l'état de santé du poste de travail *site1*, lancez la commande suivante :

```
checkhealthstatus site1
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez le poste de travail pour lequel vous souhaitez vérifier la connectivité et cliquez sur **Plus d'actions** > **Contrôle de l'état de santé...**

confirm

Confirme la fin d'un travail qui a été planifié à l'aide du mot clé **confirmed**. Par défaut, l'évaluation sur le travail est exécutée lorsque le travail se termine. Cependant, si vous confirmez le travail tandis qu'il s'exécute, la confirmation se substitue à l'évaluation exécutée à la fin de l'exécution du travail. Vous pouvez également substituer l'évaluation des conditions de sortie : par exemple, si vous définissez une ou plusieurs conditions de sortie sur true (à l'aide de Confirm SUCC), les conditions de sortie spécifiées sont définies sur true et toutes les autres conditions du travail sont définies sur false.

Vous devez avoir un accès *confirm* au travail.

Syntaxe

```
{confirm | conf} travail_sélectionné  
  ;{succ | abend}  
  [;IF 'nom_condition_sortie[, nom_condition_sortie][, ...]'] [;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

succ Confirme que le travail a abouti.

abend

Confirme que le travail a échoué.

nom_condition_sortie

Confirme le statut SUCC ou ABEND pour une ou plusieurs des conditions de sortie indiquées. Toutes conditions non spécifiées sont définies sur non satisfaite. Ce paramètre remplace toute autre évaluation.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Il n'est pas nécessaire d'utiliser le mot clé **confirmed** lors de la planification du travail pour que ce dernier passe de l'état ABEND à SUCC. Pour plus d'informations sur la confirmation d'un travail, voir «confirmed», à la page 236. Pour plus d'informations sur les travaux EXTERNAL, voir «Gestion des dépendances interréseau dans le plan», à la page 822.

tableau 64 présente l'effet de la commande **confirm** sur les différents états des travaux, avec ou sans conditions de sortie :

Tableau 64. Changement d'état après exécution de la commande *confirm*

Etat initial du travail	Etat après confirm : réussite	Etat après confirm : arrêt anormal
READY	Aucun effet, avec ou sans conditions de sortie	Aucun effet, avec ou sans conditions de sortie
HOLD	Aucun effet, avec ou sans conditions de sortie	Aucun effet, avec ou sans conditions de sortie
EXEC	sans conditions de sortie SUCC avec conditions de sortie SUCC_P et les conditions de sortie sélectionnées sont définis comme étant satisfaits	sans conditions de sortie ABEND avec conditions de sortie ABEND_P et les conditions de sortie sélectionnées sont définis comme étant satisfaits
ABENP	SUCCP , avec ou sans conditions de sortie	Aucun effet, avec ou sans conditions de sortie
SUCCP	Aucun effet, avec ou sans conditions de sortie	Aucun effet, avec ou sans conditions de sortie

Tableau 64. Changement d'état après exécution de la commande confirm (suite)

Etat initial du travail	Etat après confirm : réussite	Etat après confirm : arrêt anormal
PEND	sans conditions de sortie SUCC avec conditions de sortie SUCC et les conditions de sortie sélectionnées sont définis comme étant satisfaits.	sans conditions de sortie ABEND avec conditions de sortie ABEND et les conditions de sortie sélectionnées sont définis comme étant satisfaits.
DONE	SUCC , avec ou sans conditions de sortie.	ABEND , avec ou sans conditions de sortie.
SUCC	sans conditions de sortie L'opération n'est pas prise en charge. avec conditions de sortie SUCC et les conditions de sortie sélectionnées sont définis comme étant satisfaits.	sans conditions de sortie L'opération n'est pas prise en charge. avec conditions de sortie L'opération n'est pas prise en charge.
ABEND	SUCC , avec ou sans conditions de sortie.	Aucun effet, avec ou sans conditions de sortie.
SUPPR	sans conditions de sortie SUCC avec conditions de sortie SUCC et les conditions de sortie sélectionnées sont définis comme étant satisfaits.	L'opération n'est pas prise en charge, avec ou sans conditions de sortie.
FAIL	L'opération n'est pas prise en charge, avec ou sans conditions de sortie.	L'opération n'est pas prise en charge, avec ou sans conditions de sortie.
SCHED	Aucun effet, avec ou sans conditions de sortie	Aucun effet, avec ou sans conditions de sortie
ERROR (pour les travaux reflét uniquement)	SUCC , avec ou sans conditions de sortie	ABEND , avec ou sans conditions de sortie
tout travail du flot de travaux EXTERNE	SUCC , avec ou sans conditions de sortie	ABEND , avec ou sans conditions de sortie

Exemples

Pour générer une confirmation **succ** si le travail job3 du flot de travaux misdly(1200 02/17/06) réussit, exécutez la commande suivante :

```
confirm misdly(1200 02/17/06).job3;succ
```

Pour générer une confirmation **abend** si le travail 234 échoue, exécutez la commande suivante :

```
confirm 234;abend
```

Pour émettre une confirmation ;succ pour le travail job4 et définir MYOUTPUTCOND sur **true** dans le flot de travaux daily(1130 02/17/2016), exécutez la commande suivante :

```
confirm daily(1130 02/17/2016).job4;succ if MYOUTPUTCOND
```

L'exemple suivant montre l'effet de la commande **confirm** sur le statut du travail 79765613 et sur ses conditions de sortie. Le travail 79765613 s'est achevé avec le statut SUCC :

1. Entrez la commande **showjobs;info** pour extraire des informations sur le travail :

```
sj 79765613;info
```

L'exemple de sortie suivant est un sous-ensemble des informations que vous obtenez en exécutant la commande :

Workstation	Job Stream	SchedTime	Job
NC050239	#DUBAIJS1	1908 10/23	(NC050239_1#)JS2

oc: OUT1 false "RC=2"
oc: OUT2 true "RC=1500"

La première condition de sortie, OUT1 a pour valeur **false** et la deuxième, OUT2, a pour valeur **true**.

2. Entrez la commande **confirm** pour confirmer le statut ABEND sur la condition OUT1 :

```
confirm 79765613 ABEND; IF OUT1
```

3. Entrez à nouveau la commande **showjobs;info**. L'exemple de sortie suivant est un sous-ensemble des informations que vous obtenez en exécutant la commande :

Workstation	Job Stream	SchedTime	Job
NC050239	#DUBAIJS1	1908 10/23	(NC050239_1#)JS2

oc: OUT1 true "RC=2"
oc: OUT2 false "RC=1500"

La première condition de sortie, OUT1 est désormais passée à la valeur **true**, et la deuxième, OUT2, est passée à la valeur **false**.

console

Affecte la console IBM Workload Scheduler et définit le niveau de message.

Vous devez avoir un accès *console* au poste de travail.

Syntaxe

```
{console | cons}  
[sess | sys]  
[;level=niveau_message]
```

Arguments

- sess** Envoie des messages et des invites de la console IBM Workload Scheduler à la sortie standard.
- sys** Arrête d'envoyer des messages et des invites de la console IBM Workload Scheduler à la sortie standard. Ce processus se déroule automatiquement lorsque vous arrêtez **conman**.

niveau_message

Le niveau des messages IBM Workload Scheduler envoyés à la console. Indiquez l'un des niveaux suivants :

- 1 Il s'agit de la valeur que le produit affecte automatiquement si vous modifiez un quelconque argument pour la console sans réaffecter une valeur à `msglevel`. Avec cette valeur, le produit envoie tous les messages générés par tous les agents et pour toutes les opérations vers la console.
- 0 Aucun message. Il s'agit de la valeur par défaut sur les agents tolérants aux pannes.
- 1 Messages d'exception, tels que des invites opérateur et des fins anormales de travail.
- 2 Niveau 1 avec messages d'aboutissement du flot de travaux.
- 3 Niveau 2 avec messages d'aboutissement du travail. Il s'agit du niveau par défaut du gestionnaire de domaine maître.
- 4 Niveau 3 avec messages de lancement de travail.

Commentaires

Si une commande **console** est entrée sans aucune option, l'état en cours de la console s'affiche.

Par défaut, les processus de contrôle IBM Workload Scheduler écrivent les messages et les invites de console dans les fichiers de liste standard. Sous UNIX, ils peuvent également être envoyés au daemon **syslog**.

Exemples

Pour lancer l'écriture des invites et des messages de console dans la sortie standard et définir le niveau des messages sur **1**, exécutez la commande suivante :

```
console sess;level=1
```

Pour interrompre l'écriture des invites et des messages de console dans la sortie standard et définir le niveau des messages sur **4**, exécutez la commande suivante :

```
cons sys;l=4
```

Pour afficher l'état courant de la console, exécutez la commande suivante :

```
cons  
Console is #J675, level 2, session
```

675 correspond à l'ID processus du shell de l'utilisateur.

continue

Ignore l'erreur de commande suivante.

Syntaxe

```
{continue | cont}
```

Commentaires

Cette commande est utile lorsque des commandes sont entrées de façon non interactive. Elle indique au programme **conman** de continuer à exécuter les commandes même si celle qui a été lancée après **continue** génère une erreur.

Exemples

Pour demander au programme **conman** de continuer à utiliser la commande **rerun** même si la commande **cancel** échoue, exécutez la commande suivante :

```
conman "cont&cancel=176&rerun job=sked5(1200 02/17/06).job3"
```

deldep job

Supprime des dépendances d'un travail.

Vous devez avoir un accès *deldep* au travail.

Syntaxe

```
{deldep job | ddj} jobselect  
;dépendance[;...]  
[;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

dépendance

Type de dépendance. Indiquez au moins l'un des éléments suivants. Vous pouvez utiliser des caractères génériques dans les arguments

poste_de_travail, *flot_travaux*, *travail*, *ressource*, *nom_fichier* et *nom_invite*.

at[=*heure* | *heure_min* | *heure_max* | *heure_min,heure_max*]

confirmed

deadline[=*heure*[**timezone** | **tz** *fuseau_horaire*][**+n days** | *mm/jj[/aa]]*]

every

follows=[*agent_réseau::*][*poste_de_travail*#]{*nom_flot_travaux*(*hhmm* [*mm/jj[/aa]]*) [*.travail* | @] | *ID_flot_travaux.travail*;**schedid**}| *travail*[,...] [**if** *nom_condition*[| *nom_condition*][| ...]"][**from join** *nom_jointure*]

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez supprimer des conditions de statut ou de sortie. Si la dépendance conditionnelle fait partie d'une jointure, lorsque le nombre de conditions à satisfaire est différent de ALL, ce nombre est automatiquement réduit de un.

maxdur=[*hhmm*] [**onmaxdur** *action*]

`mindur=[hhmm] [onmindur action]`
`needs=[num] [poste_de_travail#]ressource[,...]`
`opens=[poste_travail#"nom_fichier"[(qualificatif)][,...]]`
`priority`
`prompt=["[: |!]texte" | nom_invite[,...]]`
`until=[heure [timezone | tz fuseau_horaire][+n day[s]] ;onuntil action]]`
noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Si vous supprimez `priority`, le travail retourne à sa priorité planifiée initialement. Lors de la suppression d'une dépendance `opens`, vous pouvez inclure uniquement le nom de fichier de base. **conman** recherche ensuite, sans faire de distinction majuscules/minuscules, les fichiers concordants, sans tenir compte des noms de répertoires. Les dépendances sur tous les fichiers concordants sont supprimées.

Les dépendances supprimées ne restent pas actives lors de l'exécution de la commande **rerun**.

Pour supprimer toutes les dépendances `follows` des travaux d'un flot de travaux spécifique, spécifiez le mot clé `follows` comme suit :

`follows=nom_flot_travaux`

Dans ce cas, n'utilisez pas de caractère générique (par exemple, `follows=nom_flot_travaux.*`), sinon la commande est rejetée.

Exemples

Pour supprimer une dépendance de ressource du travail `job3` du flot de travaux `sked9(0900 02/19/06)`, exécutez la commande suivante :

`ddj sked9(0900 02/19/06).job3 ; needs=2 tapes`

Pour supprimer toutes les dépendances de prédécesseur/successeur externes du flot de travaux `CPUA#TEST(0900 02/19/06)`, exécutez la commande suivante :

`ddj=CPUA#TEST(0900 02/19/06).JOBA ; follows`

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans le tableau qui contient la liste des travaux, sélectionnez le travail auquel vous souhaitez supprimer une dépendance, puis cliquez sur **Dépendances...**

7. Dans le panneau Dépendances, sélectionnez la dépendance que vous souhaitez supprimer et cliquez sur **Supprimer**.

deldep sched

Supprime des dépendances d'un flot de travaux.

Vous devez avoir un accès *deldep* au flot de travaux.

Syntaxe

```
{deldep sched | dds} jstreamselect
;dépendance[;...]
[;noask]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

dépendance

Type de dépendance. Indiquez au moins l'un des éléments suivants. Vous pouvez utiliser des caractères génériques dans les arguments *poste_de_travail*, *nom_flot_travaux*, *nom_travail*, *ressource*, *nom_fichier* et *nom_invite*, à l'exception de *poste_de_travail* lors d'une utilisation dans une dépendance de prédécesseur/successeur.

at[=*heure* | *heure_min* | *heure_max* | *heure_min,heure_max*]

carryforward

deadline[=*heure*[*timezone* | **tz** *fuseau_horaire*][**+n days** | *mm/jj/aa*]]

follows=[*agent_reseau::*][*poste_de_travail#*]{*nom_flot_travaux*[*hhmm* [*mm/dd/yy*]]}[*.travail* | **@**] | *id_flot_travaux.job*;**schedid**] | *travail*[,...][**if** '*nom_condition*[| *nom_condition*][| ...]'][**from join** *nom_jointure*]

La variable *nom_condition* indique le nom de la condition spécifié dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction follows, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez supprimer des conditions de statut ou de sortie. Si la dépendance conditionnelle fait partie d'une jointure, lorsque le nombre de conditions à satisfaire est différent de ALL, ce nombre est automatiquement réduit de un.

limit

needs=[*num*] [*poste_de_travail#*]*ressource*[,...]

opens=[*poste_de_travail#*]"*nom_fichier*"[(*qualificatif*)][,...]

priority

prompt=["[: | !]*texte*" | *nom_invite*[,...]]

until[=*heure* [*timezone* | **tz** *fuseau_horaire*][**+n day[s]**] [**onuntil** *action*]]

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Commentaires

Si vous supprimez **priority**, le travail retourne à sa priorité planifiée initialement. Lors de la suppression d'une dépendance **opens**, vous pouvez inclure uniquement le nom de fichier de base. **conman** recherche ensuite, sans faire de distinction majuscules/minuscules, les fichiers concordants, sans tenir compte des noms de répertoires. Les dépendances sur tous les fichiers concordants sont supprimées.

Les dépendances supprimées ne restent pas actives lors de l'exécution de la commande **rerun**.

Exemples

Pour supprimer une dépendance de ressource dans le flot de travaux **sked5(0900 02/19/06)**, exécutez la commande suivante :

```
dds sked5(0900 02/19/06);needs=2 tapes
```

Pour supprimer toutes les dépendances de prédécesseur/successeur du flot de travaux **sked3(1000 04/19/06)**, exécutez la commande suivante :

```
dds sked3(1000 04/19/06);follows
```

Voir aussi

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les flots de travaux du plan** ou une autre tâche pour surveiller les flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans le tableau qui contient la liste des flots de travaux, sélectionnez les flots de travaux dans lesquels vous souhaitez supprimer une dépendance, puis cliquez sur **Dépendances....**
7. Dans le panneau Dépendances, sélectionnez la dépendance que vous souhaitez supprimer et cliquez sur **Supprimer**.

deployconf

Télécharge la configuration de surveillance la plus récente pour le moteur de surveillance d'événement sur un poste de travail.

Syntaxe

```
{deployconf | deploy} [domain!]poste de travail
```

Arguments

domaine

Spécifie le nom du domaine de destination pour l'opération. Les caractères génériques ne sont pas admis.

Si vous n'incluez pas *domain*, le domaine par défaut est celui dans lequel **conman** s'exécute.

poste de travail

Indique le nom du poste de travail sur lequel la configuration va être déployée. Les caractères génériques ne sont pas admis.

Commentaires

Utilisez cette commande pour effectuer le déploiement vers un poste de travail à la fois.

Si la configuration existante est déjà mise à jour, la commande n'a pas d'effet.

Le droit de démarrer des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

display

Affiche un fichier de travail ou une définition de flot de travaux.

Si vous indiquez un fichier par son nom, vous devez avoir un accès en lecture au fichier. En ce qui concerne les fichiers de travail et les définitions de flot de travaux, vous devez avoir un accès **display** au travail ou au flot de travaux.

Syntaxe

```
{display file |  
df} nom_fichier [:offline]
```

```
{display job | dj} jobselect [:offline]
```

```
{display sched | ds} jstreamselect  
[valid {at date | in date date}  
[:offline]
```

Arguments

nom_fichier

Indique le nom du fichier, en général, un fichier script de travail. Ce nom doit être placé entre guillemets (") s'il contient des caractères autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inverses (\) et traits de soulignement (_). Les caractères génériques sont acceptés. Le fichier doit être accessible à partir de votre poste de travail de connexion. Utilisez cette option si vous ne souhaitez montrer que le contenu du fichier du script de travail.

travail_sélectionné

Travail dont le fichier de travail est affiché. Voir «Sélection de travaux dans les commandes», à la page 419. Le fichier de travail doit être accessible à partir de votre poste de travail de connexion. Ce mot clé s'applique uniquement au chemin d'accès et au nom de fichier du fichier de script des travaux définis avec l'option **scriptname**.

flot_travaux_sélectionné

Flot de travaux dont la définition est affichée. Voir «Sélection de flots de travaux dans les commandes», à la page 428.

valid Spécifie le jour ou la période au cours de laquelle les instances de flot de

travaux à afficher doivent être actives. Cela signifie que l'intervalle de validité de ces instances de flot de travaux doivent contenir la période spécifiée dans l'argument **valid**. Le format utilisé pour *date* dépend de la valeur affectée à la variable *format_date* spécifiée dans le fichier localopts. Sauf indication contraire, l'instance sélectionnée est l'instance valide aujourd'hui.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Exemples

Pour afficher le fichier c:\maestro\jclfiles\arjob3, exécutez la commande suivante :

```
df c:\apps\maestro\jclfiles\arjob3
```

Pour afficher hors ligne le fichier script du travail createpostreports du flot de travaux FINALPOSTREPORTS, exécutez la commande suivante :

```
dj FINALPOSTREPORTS(2359 02/14/13).CREATEPOSTREPORTS
```

Voici un exemple de sortie de cette commande :

```
M235062_99#FINALPOSTREPORTS(2359 02/14/13).CREATEPOSTREPORTS /opt/TWA/TWS/
CreatePostReports

#!/bin/sh
#####
# Eléments sous licence - Propriété d'IBM* et HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#####
#@(#) $Id: CreatePostReports.sh,v 1.0

##
## CreatePostReports message catalog definitions.
##

##
## message set id
##
MAE_CREATEPOSTREPORTS_SET=226
MAE_COPYRIGHT_SET=234

##
....
...
....
#
# End
#
```

Pour afficher la définition du flot de travaux mod, exécutez la commande suivante :

```
ds mod
```

Voici un exemple de sortie de cette commande :

Job Stream Name	Workstation	Valid From	Updated On	Locked By
MOD	M235062_99	06/30/2007	03/04/2006	-

```

SCHEDULE M235062_99#MOD VALIDFROM 06/30/2007
ON RUNCYCLE SCHED1_PRESIMPLE VALIDFROM 07/18/2007 "FREQ=DAILY;INTERVAL=1"
( AT 1111 )
CARRYFORWARD
FOLLOWS M235062_99#SCHED_FIRST1.@
FOLLOWS M235062_99#SCHED_FIRST.JOB_FTA
PRIORITY 66
:
M235062_99#JOBMDM
SCRIPTNAME "/usr/acct/scripts/gl1" STREAMLOGON root
DESCRIPTION "general ledger job1"
TASKTYPE UNIX
RECOVERY STOP
PRIORITY 30
NEEDS 16 M235062_99#JOBSLOTS
PROMPT PRMT3

B236153_00#JOB_FTA
FOLLOWS M235062_99#SCHED_FIRST1.@
FOLLOWS M235062_99#SCHED_FIRST.JOB_FTA
PRIORITY 66
:
M235062_99#JOBMDM
SCRIPTNAME "/usr/acct/scripts/gl1" STREAMLOGON root
DESCRIPTION "general ledger job1"
TASKTYPE UNIX
RECOVERY STOP
PRIORITY 30
NEEDS 16 M235062_99#JOBSLOTS
PROMPT PRMT3

B236153_00#JOB_FTA
DOCOMMAND "echo pippo" STREAMLOGON root
DESCRIPTION "general ledger job1"
TASKTYPE UNIX
RECOVERY STOP
FOLLOWS JOBMDM
FIN

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section Liste des définitions d'objet dans la base de données.

Pour plus d'informations sur la création et l'édition d'objets de planification, voir :

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

exit

Quitte le programme de ligne de commande **conman**.

Syntaxe

```
{exit | e}
```

Commentaires

Si vous exécutez le programme de ligne de commande Conman en mode Aide sous UNIX, la commande active le programme **conman** en mode Saisie de commande.

Exemples

Pour quitter le programme de ligne de commande **conman**, exécutez la commande suivante :

`exit`

ou
`e`

fence

Modifie la priorité minimale de travail sur un poste de travail. Les travaux ne sont pas lancés si leur niveau de priorité est inférieur ou égal à la priorité minimale de travail.

Vous devez avoir un accès *fence* au poste de travail.

Syntaxe

```
{fence | f} poste_de_travail  
    ;pri  
    [;noask]
```

Arguments

Poste de travail

Indique le nom du poste de travail. Par défaut, il s'agit de votre poste de travail de connexion.

pri Indique le niveau de priorité. Vous pouvez entrer une valeur comprise entre 0 et 99, **hi**, **go** ou **system**. Si vous entrez **system**, la priorité minimale de travail a la valeur zéro.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

La priorité minimale de travail empêche le lancement des travaux dont le niveau de priorité est faible, quel que soit le niveau de priorité affecté aux flots de travaux dont ils dépendent. Il est donc possible de différer l'exécution des travaux de priorité faible dans des flots de travaux de priorité élevée tout en permettant le lancement des travaux de priorité élevée dans des flots de travaux de priorité faible.

Lors du premier démarrage de IBM Workload Scheduler après installation, la priorité minimale de travail a pour valeur zéro. Lorsque vous modifiez la priorité minimale de travail, celle-ci est différée pendant le traitement préproduction au plan de production du jour suivant.

Pour afficher la valeur en cours affectée à la priorité minimale de travail, utilisez la commande **status**.

Exemples

Pour modifier la priorité minimale de travail du poste de travail `site4`, exécutez la commande suivante :

```
fence site4;20
```

Pour modifier la priorité minimale de travail du poste de travail sur lequel vous exécutez le programme **conman**, exécutez la commande suivante :

```
f ;40
```

Pour empêcher IBM Workload Scheduler de lancer tous les travaux sur le poste de travail `tx3`, exécutez la commande suivante :

```
f tx3;go
```

Pour modifier la priorité minimale de travail en la définissant sur zéro sur le poste de travail sur lequel vous exécutez le programme **conman**, exécutez la commande suivante :

```
f ;system
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans le panneau qui s'affiche, définissez la nouvelle priorité et cliquez sur **OK**.

help

Affiche l'aide relative aux commandes. Non disponible sous Windows.

Syntaxe

```
{help | h} {commande | mot_clé}
```

Arguments

commande

Indique le nom d'une commande système ou **conman**. Pour les commandes **conman**, entrez le nom complet de la commande ; les abréviations et les formats abrégés ne sont pas pris en charge. Pour les commandes comportant deux mots, il suffit d'entrer le premier mot pour afficher l'aide relative à toutes les versions de la commande. Par exemple, si vous entrez **help display**, toutes les informations relatives aux commandes **display file**, **display job**, et **display sched** s'affichent.

mot_clé

Vous pouvez également entrer les mots clés suivants :

COMMANDS

Affiche la liste de toutes les commandes Conman.

SETUPCONMAN

Décrit comment configurer la commande conman.

RUNCONMAN

Comment exécuter conman.

SPECIALCHAR

Décrit les caractères génériques, les délimiteurs et les autres caractères spéciaux que vous pouvez utiliser.

JOBSELECT

Répertoire des informations concernant la sélection des travaux relatifs aux commandes.

JOBSTATES

Répertoire des informations relatives aux états des travaux.

JSSELECT

Répertoire des informations concernant la sélection des flots de travaux relatifs aux commandes.

JSSTATES

Répertoire des informations relatives aux états des flots de travaux.

MANAGEBACKLEVEL

Gestion des travaux et des flots de travaux des agents de niveau antérieur.

Exemples

Pour afficher une liste de toutes les commandes **conman**, exécutez la commande suivante :

```
help commands
```

Pour afficher les informations relatives à la commande **fence**, exécutez la commande suivante :

```
help fence
```

Pour afficher les informations relatives aux commandes **altpri job** et **altpri sched**, exécutez la commande suivante :

```
h altpri
```

Pour afficher des informations relatives aux caractères spéciaux que vous pouvez utiliser, exécutez la commande suivante :

```
h specialchar
```

kill

Arrête un travail qui est en cours d'exécution. Sous UNIX, vous devez utiliser la commande UNIX **kill**. Vous devez avoir un accès *kill* au travail.

Syntaxe

```
{kill | k} jobselect  
[;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Le temps de réponse est légèrement plus long car l'opération **kill** n'est pas réalisée par **conman** mais par un processus de production IBM Workload Scheduler.

A la suite de la commande **kill**, les travaux sont à l'état ABEND (arrêt anormal). Les travaux ou les flots de travaux qui dépendent de ces travaux ne sont pas libérés. Ces travaux ne peuvent pas être exécutés à nouveau.

Exemples

Pour supprimer le travail report du flot de travaux apwkly(0600 03/05/06) sur le poste de travail site3, exécutez la commande suivante :

```
kill site3#apwkly(0600 03/05/06).report
```

Pour arrêter le travail numéro 124 qui s'exécute sur le poste de travail geneva, exécutez la commande suivante :

```
kill geneva#124
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des travaux, sélectionnez le travail à arrêter et cliquez sur **Plus d'actions > Annuler**.

limit cpu

Modifie le nombre maximal de travaux pouvant être exécutés simultanément sur un poste de travail. Vous devez avoir un accès *limit* au poste de travail.

Syntaxe

```
{limit cpu | lc} poste de travail  
;limite  
[;noask]
```

Arguments

poste de travail

Indique le nom du poste de travail. Les caractères génériques sont acceptés. Par défaut, il s'agit de votre poste de travail de connexion.

limit Indique le nombre de travaux qui peuvent s'exécuter simultanément sur le poste de travail. Les valeurs prises en charge vont de **0** à **1024** et **system**.

Si vous définissez *limit cpu* à **0** :

- Pour un flot de travaux à l'état **READY**, seuls les travaux avec des valeurs de priorité **hi** et **go** peuvent être lancés sur le poste de travail.
- Pour un flot de travaux avec une valeur de priorité **hi** ou **go**, tous les travaux avec une valeur de priorité différente de 0 peuvent être lancés sur le poste de travail.

Si vous définissez *limit cpu* à **system**, il n'y a pas de limite au nombre de travaux simultanés sur le poste de travail. Pour l'agent étendu, lorsque la limite est définie à **SYSTEM**, la limite du travail est définie à 0.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

Pour afficher le nombre maximal de travaux autorisés sur votre poste de travail de connexion, utilisez la commande **status**.

Lors du premier démarrage de IBM Workload Scheduler après installation, le nombre maximal de travaux de poste de travail a pour valeur zéro ; vous devez augmenter ce nombre avant de lancer des travaux. Lorsque vous modifiez la limite, le travail est différé lors du processus de préproduction au plan de processus du jour suivant.

IBM Workload Scheduler tente de lancer autant de travaux que possible, dans les limites autorisées. Le nombre de processus qu'il est possible de lancer sur un poste de travail est limité de fait. Si cette limite est atteinte, le système envoie un message indiquant que les ressources système ne sont pas disponibles. Lorsqu'un travail ne peut pas être lancé pour cette raison, il passe à l'état **fail** (échec). Une réduction du nombre maximal de travaux permet d'éviter ces échecs.

Exemples

Pour modifier le nombre maximal de travaux du poste de travail sur lequel vous exécutez le programme **conman**, exécutez la commande suivante :

```
lc ;12
```

Pour modifier le nombre maximal de travaux du poste de travail **rx12**, exécutez la commande suivante :

```
lc rx12;6
```

Pour définir à 10 le nombre maximal de travaux sur tous les postes de travail appartenant au domaine et aux domaines enfant, exécutez la commande suivante :

```
lc @!@;10
```


Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions > Limite...**
7. Définissez la limite selon les besoins.

limit sched

Modifie la valeur du mot clé **limit** déterminée dans la définition d'un flot de travaux. Pour des informations supplémentaires sur la définition d'une limite dans la définition de flot de travaux, reportez-vous à «limite», à la page 260. Vous devez avoir un accès **limit** au flot de travaux.

Syntaxe

```
{limit sched |  
ls } flot_travaux_sélectionné  
    ;limite  
    [;noask]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

limit Indique le nombre maximal de travaux. Vous pouvez entrer une valeur comprise entre **0** et **1024**.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Exemples

Pour modifier le nombre maximal de travaux dans tous les flots de travaux dont le nom comporte la mention *sales*, exécutez la commande suivante :

```
ls sales@;4
```

Pour modifier le nombre maximal de travaux du flot de travaux *CPUA#Job1*, exécutez la commande suivante :

```
ls=CPUA#apwk1y;6
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.
4. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
5. Sélectionnez un flot de travaux et cliquez sur **Plus d'actions** > **Limite....**
6. Définissez la limite selon les besoins.

link

Etablit des liaisons de données entre postes de travail. Dans un réseau IBM Workload Scheduler, les agents standard et tolérants aux pannes sont reliés à leurs gestionnaires de domaine et ces derniers sont reliés à leurs gestionnaires de domaine parent. Les agents étendus ne sont pas reliés ; ils communiquent via un hôte.

Vous devez avoir un accès *link* au poste de travail cible.

La commande nécessite qu'un autre poste de travail soit présent dans votre environnement en plus du gestionnaire de domaine maître.

Syntaxe

```
{link | lk} [domaine!]poste de travail
[;noask]
```

Arguments

domaine

Indique le nom du domaine dans lequel des liaisons sont établies. Les caractères génériques sont acceptés.

Cet argument est utile lors de la connexion de plusieurs postes de travail d'un domaine. Par exemple, pour connecter tous les agents du domaine *stlouis*, utilisez la commande suivante :

```
lk stlouis!@
```

Le domaine n'est pas nécessaire si vous n'incluez pas de caractères génériques dans *poste de travail*.

Si le *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui où **conman** est exécuté.

poste de travail

Indique le nom du poste de travail à connecter. Les caractères génériques sont acceptés.

Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

Si l'option **autolink** a pour valeur **on** dans une définition de poste de travail, ce dernier est connecté automatiquement à chaque démarrage de IBM Workload

Scheduler. Si l'option **autolink** a pour valeur **off**, vous devez utiliser les commandes **link** et **unlink** pour contrôler le processus de connexion. Pour plus d'informations sur **autolink** voir «Définition de poste de travail», à la page 166.

Lorsque l'utilisateur possède un accès **link** aux postes de travail en cours de connexion, les règles ci-dessous s'appliquent :

- Un utilisateur exécutant **conman** sur le gestionnaire de domaine maître peut relier n'importe quel poste de travail du réseau.
- Un utilisateur exécutant **conman** sur un autre gestionnaire que le gestionnaire de domaine maître peut relier n'importe quel poste de travail de son domaine et des domaines subordonnés. L'utilisateur ne peut pas relier des postes de travail dans les domaines homologues.
- Un utilisateur exécutant **conman** sur un agent peut relier n'importe quel poste de travail dans son domaine local à condition que le poste de travail soit un gestionnaire de domaine ou un hôte. Un agent homologue dans le domaine local ne peut pas être relié.
- Pour relier un domaine subordonné lors de l'exécution de **conman** dans un domaine de niveau supérieur, il n'est pas nécessaire que les liaisons intermédiaires soient ouvertes.

Exemples

La figure 25 et le tableau 65, à la page 470 présentent les liaisons établies par les commandes **link** exécutées par les utilisateurs en différents points du réseau. **DM n** sont des gestionnaires de domaine et **A mn** sont des agents.

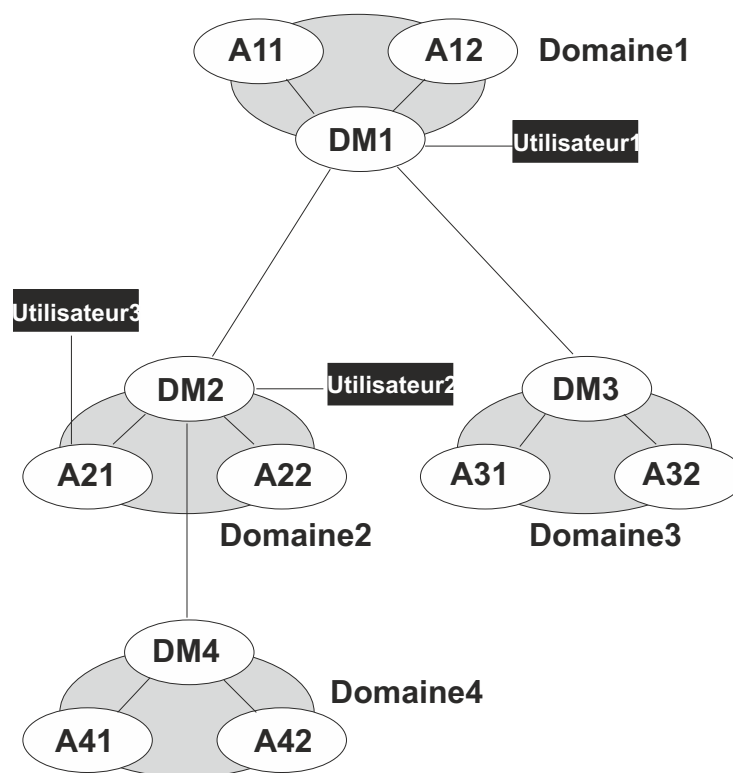


Figure 25. Liaisons réseau

Tableau 65. Liaisons ouvertes

Commande	Liaisons établies par Utilisateur1	Liaisons établies par Utilisateur2	Liaisons établies par Utilisateur3
link @!@	Toutes les liaisons sont ouvertes.	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4 DM4-A41 DM4-A42	DM2-A21
link @	DM1-A11 DM1-A12 DM1-DM2 DM1-DM3	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4	DM2-A21
link DOMAIN3!@	DM3-A31 DM3-A32	Non autorisé	Non autorisé
link DOMAIN4!@	DM4-A41 DM4-A42	DM4-A41 DM4-A42	Non autorisé
link DM2	DM1-DM2	Sans objet	DM2-A21
link A42	DM4-A42	DM4-A42	Non autorisé
link A31	DM3-A31	Non autorisé	Non autorisé

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Lier**.

listsym

Affiche le plan de production (fichiers Symphony) déjà traité.

Lorsque cette commande est utilisée à partir de la ligne de commande d'un agent tolérant aux pannes, elle indique le fichier Symphony le plus récent, enregistré sous le nom MSym01dBackup.

Syntaxe

```
{listsym | lis} [trial | forecast]
                [;offline]
```

Arguments

trial Répertoire les plans d'essai.

forecast
Répertoire les plans prévisionnels.

offline
Envoie la sortie de la commande au périphérique de sortie de **conman**.
Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Résultats

Date du calendrier
Date utilisée pour la sélection des flots de travaux à exécuter.

Date réelle
Date à laquelle **batchman** a lancé l'exécution du fichier Symphony.

Heure de début
Heure à laquelle **batchman** a lancé l'exécution du fichier Symphony.

Date de journalisation
Date de consignation du plan (fichier Symphony) par la commande **stageman**.

Num. d'exécution
Numéro d'exécution attribué au plan (fichier Symphony). Il est utilisé en interne pour la synchronisation de réseau IBM Workload Scheduler.

Taille Nombre d'enregistrements contenus dans le fichier Symphony.

Num. de journalisation
Numéro de consignation indiquant l'ordre chronologique des fichiers journaux. Ce numéro peut être utilisé dans une commande **setsym** pour accéder à un fichier journal spécifique.

Nom du fichier
Nom du fichier journal attribué par la commande **stageman**.

Exemples

Pour afficher les fichiers du plan de production, exécutez la commande suivante :
`listsym`

Voici un exemple de sortie de cette commande :

Job	Stream	Actual	Start	Log	Run	Log		
Date	Date	Time	Date	Num	Size	Num	Filename	
03/05/06	03/05/06	21:06	03/05/06	42	534	1	M200603052111	Exp
03/04/06	03/04/06	15:59	03/05/06	41	463	2	M200603052106	Exp
03/04/06	03/04/06	15:51	03/04/06	40	362	3	M200603041559	Exp
03/04/06	03/04/06	14:31	03/04/06	39	460	4	M200603041551	Exp
03/04/06	03/04/06	14:26	03/04/06	38	436	5	M200603041431	Exp
03/04/06	03/04/06	14:24	03/04/06	37	436	6	M200603041426	Exp
03/04/06	03/04/06	14:19	03/04/06	36	436	7	M200603041424	Exp
03/04/06	03/04/06	14:17	03/04/06	35	436	8	M200603041419	Exp
03/04/06	03/04/06	14:17	03/04/06	34	364	9	M200603041417	Exp

Pour afficher le fichier de plan de production le plus récent qui ait été traité sur un agent tolérant aux pannes, exécutez la commande suivante à partir du programme de ligne de commande **conman** de l'agent tolérant aux pannes :

`listsym`

Voici un exemple de sortie de cette commande :

Job Stream	Actual	Start	Log	Run	Log		
Date	Date	Time	Date	Num	Size	Num	Filename
07/07/14	07/07/14	14:01	07/08/14	19	1607	1	MSymOldBackup (Exp)

Pour afficher les fichiers contenant des plans d'essai, exécutez la commande suivante :

```
listsym trial
```

Voici un exemple de sortie de cette commande :

Job Stream	Actual	Start	Log	Run	Log		
Date	Date	Time	Date	Num	Size	Num	Filename
03/03/06		03/03/06	0	126	1	Tpippo	Exp
03/03/06		03/03/06	0	1850	2	Tangelo2	Exp
03/03/06		03/03/06	0	1838	3	Tangelo1	Exp

Pour afficher les fichiers contenant les plans prévisionnels, exécutez la commande suivante :

```
listsym forecast
```

Voici un exemple de sortie de cette commande :

Job Stream	Actual	Start	Log	Run	Log		
Date	Date	Time	Date	Num	Size	Num	Filename
03/03/06			03/03/06	0	62	1	Fpluto Exp

Voir aussi

Dans Dynamic Workload Console :

1. Dans la barre de navigation, cliquez sur **Planification > Prévion de charge de travail > Gestion des plans disponibles**.
2. Sélectionnez un moteur.
3. Cliquez sur un type de plan ou saisissez le nom de fichier du plan souhaité
4. Cliquez sur **Afficher liste des plans**.

Listsucc

Répertorie les successeurs d'un travail.

Vous devez avoir un accès *rerun* au travail.

Syntaxe

```
listsucc travail_sélectionné
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

Commentaires

Si l'utilisateur qui exécute la commande n'est pas autorisé à visualiser et réexécuter tous les successeurs du travail ayant échoué, la liste qui s'affiche contient uniquement les successeurs qu'il est autorisé à visualiser. Un message d'erreur s'affiche pour indiquer qu'il existe des successeurs supplémentaires qu'il n'est pas autorisé à visualiser ou à exécuter.

Le nombre maximal de travaux successeurs pouvant être retourné est 1000. Pour modifier cette valeur, éditez la propriété **com.hcl.tws.conn.plan.rerun.successors.maxjobs** dans le fichier TWSEConfig.properties situé dans rép_principal_TWA/WAS/TWSEprofile/properties. Pour que cette modification soit prise en compte, redémarrez le gestionnaire de domaine maître. Lorsque vous exécutez la commande, le travail parent est retourné dans la liste des successeurs, mais il n'est pas pris en compte dans le nombre total de travaux successeurs répertoriés. Par exemple, si vous définissez la propriété **com.hcl.tws.conn.plan.rerun.successors.maxjobs** sur dix, le nombre total de successeurs de votre travail parent est dix, alors qu'au total onze travaux seront retournés. Cela est dû au fait que le travail parent est déjà répertorié, car il est planifié pour être réexécuté avec ses successeurs.

L'action est toujours effectuée sur la dernière instance de réexécution du travail indiqué. De plus, si vous indiquez le numéro de travail d'un travail intermédiaire dans la séquence de réexécution, l'action est effectuée sur le dernier travail dans la séquence de réexécution.

Exemples

Lorsque vous lancez la commande listsucc S_LONDON#JS_PAYROLL.ABSENCES_JOB1, une sortie similaire à celle-ci est affichée :

Successors in the same job stream						
Workstation	Job stream	Job	Status	Sched Time		Sched id
S_LONDON	JS_PAYROLL	ABSENCES_JOB1	SUCC	0000	03/08	OAAAAAAAAAABU5
S_LONDON	JS_PAYROLL	ABSENCES_JOB2	SUCC	0000	03/08	OAAAAAAAAAABU5
S_LONDON	JS_PAYROLL	ABSENCES_JOB3	SUCC	0000	03/08	OAAAAAAAAAABU5
S_LONDON	JS_PAYROLL	ABSENCES_JOB4	SUCC	0000	03/08	OAAAAAAAAAABU5
Successors in other job streams						
Workstation	Job stream	Job	Status	Sched Time		Sched id
S_LONDON	JS_BALANCE	ALLSHEET_JOB1	SUCC	0000	03/08	OAAAAAAAAAABT0
S_LONDON	JS_BALANCE	ALLSHEET_JOB2	EXEC	0000	03/08	OAAAAAAAAAABTB
S_LONDON	JS_BALANCE	ALLSHEET_JOB3	EXEC	0000	03/08	OAAAAAAAAAABTB
S_LONDON	JS_BALANCE	ALLSHEET_JOB4	HOLD	0000	03/08	OAAAAAAAAAABU0

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des travaux, sélectionnez un travail, puis cliquez sur **Réexécuter avec les successeurs**.

=

recall

Affiche des invites qui attendent une réponse.

Vous devez avoir un accès *display* aux invites.

Syntaxe

```
{recall | rc}  
[poste de travail]  
[:offline]
```

Arguments

poste de travail

Indique le nom du poste de travail sur lequel l'invite a été émise. Si aucun poste de travail n'est indiqué, seules les invites du poste de travail de connexion et les invites globales s'affichent.

offline

Envoie la sortie de la commande à un fichier ou une commande indiqué dans **conman**. Pour plus d'informations, voir «Sortie offline», à la page 412.

Résultats

Etat Etat de l'invite. Les invites en attente sont toujours à l'état ASKED (demandée).

Message ou invite

Pour les invites nommées, numéro de message, nom de l'invite et texte du message. Pour les invites non nommées, numéro de message, nom du travail ou du flot de travaux et texte du message.

Exemples

Pour afficher les invites en attente générées sur le poste de travail où vous exécutez le programme **conman**, exécutez la commande suivante :

```
recall
```

ou :

```
rc
```

Pour afficher les invites en attente du poste de travail `site3`, exécutez la commande suivante :

```
rc site3
```

Pour afficher les invites en attente sur tous les postes de travail et envoyer les résultats à un fichier ou une commande, exécutez la commande suivante :

```
rc @;offline
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.

2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Invite**.
4. Dans la liste déroulante **Requête**, sélectionnez **Toutes les invites du plan**, qui répertorie toutes les invites, quel que soit leur statut, ou créez et sélectionnez une autre tâche.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

redo

Modifie et exécute à nouveau la commande précédente.

Syntaxe

{redo | red}

Contexte

Si vous exécutez la commande **redo**, le programme **conman** affiche la dernière commande, que vous pouvez modifier et réexécuter à votre guise. Utilisez la barre d'espacement pour placer le curseur sous le caractère à modifier, puis entrez les instructions ci-dessous.

Instructions

- d[rép]** Supprime le caractère situé au-dessus de la lettre **d**. Cette commande peut être suivie par d'autres instructions.
- itexte** Insère du texte avant le caractère situé au-dessus de la lettre **i**.
- rtexte** Remplace un ou plusieurs caractères par du texte en commençant par le caractère au-dessus du **r**. Le remplacement est implicite si aucune autre instruction n'est entrée.
- >texte** Ajoute du texte à la fin de la ligne.
- >d[rép | texte]** Supprime les caractères en fin de ligne. Cette commande peut être suivie d'une autre instruction ou d'un autre texte.
- >rtexte** Remplace des caractères à la fin de la ligne par du texte.

Exemples d'instruction

- ddd** Supprime les trois caractères au-dessus des **d**.
- iabc** Insère **abc** avant le caractère situé au-dessus du **i**.
- rabc** Remplace les trois caractères, en commençant par celui qui est situé au-dessus du **r**, par **abc**.
- abc** Remplace les trois caractères situés au-dessus de **abc**, par **abc**.
- d diabc** Supprime le caractère au-dessus du premier **d**, ignore un caractère, supprime le caractère au-dessus du second **d**, puis le remplace par **abc**.
- >abc** Ajoute **abc** à la fin de la ligne.
- >ddabc** Supprime les deux derniers caractères de la ligne, puis les remplace par **abc**.
- >rabc** Remplace les trois derniers caractères de la ligne par **abc**.

Exemples

Pour insérer un caractère, exécutez la commande suivante :

```
redo
setsm 4
    iy
setsym 4
```

Pour remplacer un caractère, exécutez la commande suivante :

```
redo
setsym 4
    5
setsym 5
```

release job

Libère les travaux des dépendances.

Vous devez avoir un accès *release* au travail.

Syntaxe

```
{release job |
rj} travail_sélectionné
    [;dépendance[;...]]
    [;noask]
```

Arguments

travail_sélectionné

Indique le ou les travaux à libérer. Voir «Sélection de travaux dans les commandes», à la page 419.

dépendance

Type de dépendance. Vous pouvez indiquer l'un des éléments suivants. Vous pouvez utiliser des caractères génériques dans les arguments *poste_de_travail*, *nom_flot_travaux*, *nom_travail*, *ressource*, *nom_fichier* et *nom_invite*.

at[=*heure* | *heure_min* | *heure_max* | *heure_min,heure_max*]

confirmed

deadline[=*heure*[**timezone** | **tz fuseau_horaire**][**+n days** | *mm/jj[/aa]]*]

every

follows=[*agent_réseau*::][*poste_de_travail*#]{*nom_flot_travaux*[*hhmm* [*mm/dd[/yy]]*][*.travail* | @] | *id_flot_travaux.job*;**schedid** | *travail*[,...] [**if** *nom_condition*[| *nom_condition*][| ...]']][**from join** *nom_jointure*]

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction *follows*, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez libérer des conditions de statut ou de sortie. Si

la dépendance conditionnelle fait partie d'une jointure, lorsque le nombre de conditions à satisfaire est différent de ALL, ce nombre est automatiquement réduit de un.

needs=[*num*] [*poste_de_travail#*]*ressource*[,...]]

opens=[*poste_travail#*]"*nom_fichier*"[(*qualificatif*)][,...]]

priority

prompt=["[: | !]*texte*" | *nom_invite*[,...]]

until[=*heure* [**timezone** | *tz fuseau_horaire*][+*n day[s]*] [**onuntil** *action*]]

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

La commande s'applique seulement aux travaux qui sont à l'état HOLD ; c'est-à-dire, les travaux qui attendent la résolution d'une dépendance. Notez que la dépendance est libérée uniquement pour l'exécution en cours du travail mais pas pour les réexecutions ultérieures. (La libération permanente d'une dépendance peut être obtenue à l'aide de la commande `deldep`).

Lors de la libération d'une dépendance `opens`, vous pouvez inclure uniquement le nom de fichier de base. **conman** recherche ensuite, sans faire de distinction majuscules/minuscules, les fichiers concordants, sans tenir compte des noms de répertoires. Les dépendances sur tous les fichiers concordants sont libérées.

En ce qui concerne les dépendances `needs`, le travail libéré reçoit le nombre requis d'unités de la ressource, même si celles-ci risquent de ne pas être disponibles. Cela peut entraîner l'affichage par les unités disponibles d'un nombre négatif dans **showresources**.

Lorsque vous libérez un travail d'une dépendance de type `priority`, celui-ci reprend sa priorité d'origine planifiée.

Les dépendances libérées restent en vigueur lors de l'exécution de la commande **rerun**.

Exemples

Pour libérer le travail `job3` du flot de travaux `ap(1000 03/05/06)` de toutes ses dépendances, exécutez la commande suivante :

```
rj ap(1000 03/05/06).job3
```

Pour libérer tous les travaux du poste de travail `site4` de ses dépendances sur une invite baptisée `glprmt`, exécutez la commande suivante :

```
rj=site4#@. @;prompt=glprmt
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des travaux, sélectionnez un ou plusieurs travaux et cliquez sur **Libérer des dépendances**.

release sched

Libère les flots de travaux des dépendances.

Vous devez avoir un accès *release* au flot de travaux.

Syntaxe

```
{release sched
| rs} flot_travaux_sélectionnés
    [dépendance[:...]]
    [noask]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

dépendance

Type de dépendance. Indiquez l'un des éléments suivants. Vous pouvez utiliser des caractères génériques dans les arguments *poste_de_travail*, *flot_travaux*, *travail*, *ressource*, *nom_fichier* et *nom_invite*.

at[=*heure* | *heure_min* | *heure_max* | *heure_min,heure_max*]

carryforward

deadline[=*heure*[**timezone** | **tz fuseau_horaire**]][**+n days** | *mm/jj[/aa]]]*

follows=[*agent_réseau::*][*poste_de_travail*#]{*nom_flot_travaux*[*hhmm* [*mm/dd[/yy]]*][*.travail* | **@**] | *id_flot_travaux.job*;**schedid**] | *travail*[,...] [**if** *nom_condition*[| *nom_condition*][| ...]']][**from join** *nom_jointure*]

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions doivent être séparées d'une barre verticale (|) et placées entre guillemets simples. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez libérer des conditions de statut ou de sortie. Si la dépendance conditionnelle fait partie d'une jointure, lorsque le nombre de conditions à satisfaire est différent de ALL, ce nombre est automatiquement réduit de un.

limit

```
needs=[num] [poste_de_travail#]ressource[,...]]
opens=[poste_travail#]"nom_fichier"[(qualificatif)][,...]]
priority
prompt=["[: | !]texte" | nom_invite[,...]]
until[=heure [timezone | tz fuseau_horaire][+n day[s]] [onuntil action]]
```

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Commentaires

Lors de la suppression d'une dépendance **opens**, vous pouvez inclure uniquement le nom de fichier de base. **conman** recherche ensuite, sans faire de distinction majuscules/minuscules, les fichiers concordants, sans tenir compte des noms de répertoires. Les dépendances sur tous les fichiers concordants sont libérées.

En ce qui concerne les dépendances **needs**, le flot de travaux libéré reçoit le nombre requis d'unités de la ressource, même si celles-ci risquent de ne pas être disponibles. Cela peut entraîner l'affichage par les unités disponibles d'un nombre négatif dans **showresources**.

Lorsque vous libérez un flot de travaux d'une dépendance de type **priority**, le flot de travaux reprend sa priorité d'origine.

Dans certaines circonstances, la commande **deldep**, une fois soumise, est susceptible de s'exécuter avec succès, même si elle est à nouveau envoyée à **batchman**. Pour plus d'informations, voir «Traitement des commandes Conman», à la page 418.

Exemples

Pour libérer l'instance de flot de travaux ayant l'ID *ID_flot_travaux* 0AAAAAAAAAAAAABSE de toutes ses dépendances, exécutez la commande suivante :

```
rs 0AAAAAAAAAAAAABSE; schedid
```

Pour libérer le flot de travaux sked5(1105 03/07/06) de toutes ses dépendances **opens**, exécutez la commande suivante :

```
rs sked5(1105 03/07/06);opens
```

Pour libérer tous les flots de travaux du poste de travail *site3* de ses dépendances dans le flot de travaux *main#sked23*, exécutez la commande suivante :

```
rs=site3#0;follows=main#sked23
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.

4. Dans la liste déroulante **Requête**, sélectionnez **Tous les flots de travaux du plan** ou une autre tâche pour surveiller les flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des flots de travaux, sélectionnez un ou plusieurs flots de travaux et cliquez sur **Libérer des dépendances**.

reply

Répond à une invite de travail ou de flot de travaux.

Vous devez avoir un accès *reply* à l'invite nommée ou globale. Pour répondre à une invite non nommée, vous devez détenir un accès *reply* aux invites et *reply* au travail ou flot de travaux associés.

Syntaxe

```
{reply | rep}
  { nom_invite | [poste_de_travail#]num_message}
  ;reply
  [:noask]
```

Arguments

nom_invite

Indique le nom d'une invite globale. Les caractères génériques sont acceptés.

poste de travail

Indique le nom du poste de travail sur lequel une invite non nommée a été émise.

msgnum

Indique le numéro de message d'une invite non nommée. Vous pouvez afficher des numéros de message à l'aide des commandes **recall** et **showprompts**.

reply Indique la réponse : **O** pour oui ou **N** pour non.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque invite éligible.

Commentaires

Si la réponse est **O**, les dépendances sur l'invite sont respectées. Si la réponse est **N**, les dépendances ne sont pas respectées et l'invite n'est pas émise de nouveau.

Vous pouvez répondre aux invites avant qu'elles soient émises. La commande **showprompts** permet d'afficher toutes les invites, qu'elles aient été émises ou non.

Exemples

Pour répondre **O** à l'invite globale `arpm`, exécutez la commande suivante :

```
reply arpm;y
```

Pour répondre **N** au message **24** sur le poste de travail `site4`, exécutez la commande suivante :

```
rep site4#24;n
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Invite**.
4. Sélectionnez **Toutes les invites du plan** ou une autre tâche pour surveiller les invites.
5. A partir de la table des résultats, sélectionnez une invite et cliquez sur **Répondre oui** ou **Répondre non**.

rerun

Lance une nouvelle exécution d'un travail.

Vous devez avoir un accès *rerun* au travail.

Syntaxe

```
{rerun | rr}  
travail_sélectionné  
    [;from=[poste_de_travail#]travail  
    [;at=heure]  
    [;pri=pri]]  
    [;sameworkstation=]  
    [;step=étape]  
    [;noask]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419. Les caractères génériques sont autorisés.

from=[*poste_de_travail#*]*travail*

Indique le nom d'un travail défini dans la base de données dont le fichier ou la commande va être exécuté à la place du travail indiqué par l'argument *travail_sélectionné*. Vous pouvez réexécuter des travaux également à l'état SUPPR, à condition qu'ils n'appartiennent pas à des flots de travaux à l'état annulé ou supprimé.

poste_de_travail#

Indique le nom du poste de travail sur lequel le travail **from** s'exécute. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé.

travail

Indique le nom de la définition de travail donnée par l'argument **from**. Les types de noms de travail suivants ne sont pas admis :

- Les noms des travaux soumis à l'aide des commandes **submit file** et **submit docommand**.
- Les noms d'alias des travaux soumis à l'aide de la commande **submit job**.

Les options de reprise éventuelles sont en partie héritées de la définition de travail d'origine et en partie extraites du travail **from**. tableau 66, à la page 482

page 482 décrit les critères en fonction desquels les options de reprise dans la définition de travail d'origine et **from** sont extraites.

Tableau 66. Critères de récupération des options de reprise

option de reprise	héritée du travail d'origine	extraite du travail "from"
stop	Non	Oui
continuer	Non	Oui
rerun	Oui	Non
repeatevery	Oui	Non
for	Oui	Non
after	Oui	Non
abendprompt	Oui	Non

Pour utiliser l'argument **from**, vous devez avoir accès à la base de données à partir de l'ordinateur sur lequel **conman** est en cours d'exécution.

at=heure

Spécifie l'heure de démarrage de réexécution du travail, exprimée comme suit :

hhmm [*timezone* | *tz fuseau_horaire*] [*+n days* | *date*]

Où :

hhmm Heure et minute.

+n days

Occurrence suivante de *hhmm* en *n* nombre de jours.

date Occurrence suivante de *hhmm* à la *date* indiquée, exprimée au format *mm/jj[/aa]*.

timezone | **tz fuseau_horaire**

Nom du fuseau horaire du travail. Voir Chapitre 19, «Gestion des fuseaux horaires», à la page 799 pour connaître les noms valides.

pri=pri

Indique la priorité à affecter au travail réexécuté. Si aucune priorité n'est indiquée, le travail reçoit la même priorité que le travail d'origine.

sameworkstation

Si le travail parent s'exécutait sur un poste de travail faisant partie d'un pool ou d'un pool dynamique, vous pouvez décider s'il doit se réexécuter sur le même poste de travail ou sur un autre poste de travail. Cela est dû au fait que la charge de travail sur les pools et les pools dynamiques est affectée dynamiquement en fonction d'un certain nombre de critères et que le travail peut alors être réexécuté sur un autre poste de travail.

Lorsque vous réexécutez le travail manuellement, le paramètre **sameworkstation** que vous définissez avec la commande **rerun** est appliqué uniquement à l'instance que vous réexécutez et il est ignoré dans les ré exécutions ultérieures de cette instance. Par exemple, si le travail que vous réexécutez contient déjà des informations de réexécution (définies avec les arguments **rerun**, **repeatevery** et **for** dans la définition de travail), le paramètre **sameworkstation** que vous définissez avec la commande **rerun** s'applique uniquement à l'instance spécifique que vous réexécutez. Pour les ré exécutions ultérieures, le paramètre défini dans la définition de

= travail est utilisé. Pour plus d'informations sur les arguments dans la
= définition de travail, voir «Définition de travail», à la page 187.

step=étape

Indique que le travail est réexécuté à l'aide de ce nom, à la place du nom de travail d'origine. Pour plus d'informations, voir "Usage Notes".

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Vous pouvez réexécuter des travaux qui sont à l'état SUCC, FAIL ou ABEND. Un travail réexécuté est placé dans le même flot de travaux que le travail d'origine, et hérite des dépendances de ce dernier. Si vous réexécutez un travail répétitif (**every**), le travail sera réexécuté selon la même fréquence que le travail d'origine.

Remarque : Vous pouvez soumettre la commande **rerun** pour les travaux du flot de travaux EXTERNES qui sont à l'état ERROR. Les travaux du flot de travaux EXTERNES correspondent à des travaux et à des flots de travaux qui sont signalés comme étant des dépendances interréseaux. Après toute exécution de la commande **rerun**, le travail passe immédiatement à l'état **extrn** et **conman** commence à vérifier l'état.

Lorsque l'option **;from** est utilisée, le nom du travail réexécuté est fonction de la valeur de l'option globale **enRetainNameOnRerunFrom**. Si l'option a pour valeur **O**, les travaux réexécutés conservent les noms de travaux d'origine. Si l'option a pour valeur **N**, les noms de travaux **from** sont attribués aux travaux réexécutés. Pour plus d'informations, voir IBM Workload Scheduler - *Guide d'administration*.

Dans les écrans **conman**, les travaux réexécutés qui s'affichent portent la mention **>>rerun as**. Pour faire référence à un travail réexécuté dans une autre commande telle que **altpri**, vous devez utiliser le nom de travail d'origine.

Lors de la réexécution d'un travail à l'aide de l'option **;step**, le travail s'exécute sous le nom **step** à la place de son nom d'origine. Dans un script de travail, vous pouvez utiliser la commande **jobinfo** pour revenir au nom du travail, puis exécuter le script différemment pour chaque itération. Dans le script UNIX suivant, la commande **jobinfo** permet d'affecter à une variable **STEP** le nom qui a été utilisé pour exécuter le travail. Le programme utilise ensuite la variable **STEP** pour déterminer le mode d'exécution du script.

```
...
MPATH=`maestro`
STEP=`$MPATH/bin/jobinfo nom_travail`
if [$STEP = JOB3]
then
...
STEP=JSTEP1
fi
if [$STEP = JSTEP1]
then
...
STEP=JSTEP2
fi
if [$STEP = JSTEP2]
then
...
fi
...
```

Dans les écrans **conman**, les travaux réexécutés avec l'option **;step** s'affichent avec la mention **>>rerun step**.

Pour plus d'informations sur **jobinfo**, voir «**jobinfo**», à la page 701.

Exemples

Pour réexécuter le travail **job4** du flot de travaux **sked1** sur le poste de travail **main**, exécutez la commande suivante :

```
rr main#sked1.job4
```

Pour réexécuter le travail **job5** du flot de travaux **sked2** en utilisant la définition du travail **jobx** dont l'heure d'exécution **at** et la priorité sont respectivement définies sur 18:30 et **25**, exécutez la commande suivante :

```
rr sked2.job5;from=jobx;at=1830;pri=25
```

Pour réexécuter le travail **job3** du flot de travaux **sked4** en utilisant le nom de travail **jstep2**, exécutez la commande suivante :

```
rr sked4.job3;step=jstep2
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des travaux, sélectionnez un travail, puis cliquez sur **Réexécuter....**

Rerunsucc

Réexécute un travail et ses successeurs. Vous pouvez choisir si vous souhaitez réexécuter tous les successeurs dans le même flot de travaux que celui du travail ayant échoué (successeurs internes), ou tous les successeurs, dans le même flot de travaux et dans des flots de travaux externes (successeurs internes et externes).

Vous devez avoir un accès **rerun** au travail.

Syntaxe

```
rerunsucc travail_sélectionné [;internal][;all]
```

Arguments

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

;internal

Indique que tous les successeurs du travail ayant échoué (travail parent) dans le même flot de travaux doivent être réexécutés. Les successeurs éventuels dans les flots de travaux externes ne sont pas réexécutés.

;all

Indique que tous les successeurs du travail ayant échoué (travail parent) figurant à la fois dans le même flot de travaux et dans des flots de travaux externes peuvent être réexécutés.

Commentaires

Si vous entrez la commande sans option, une liste des successeurs internes et externes associés à leur statut est retournée. Un message s'affiche pour vous demander de confirmer si vous souhaitez réexécuter uniquement les successeurs internes ou les successeurs internes et externes à la fois.

Si l'utilisateur qui exécute la commande n'est pas autorisé à visualiser et réexécuter tous les successeurs du travail ayant échoué, la liste qui s'affiche contient uniquement les successeurs qu'il est autorisé à visualiser. Un message d'erreur s'affiche pour indiquer qu'il existe des successeurs supplémentaires qu'il n'est pas autorisé à visualiser ou à exécuter.

Le nombre maximal de travaux successeurs pouvant être retourné est 1000. Pour modifier cette valeur, éditez la propriété **com.hcl.tws.conn.plan.rerun.successors.maxjobs** dans le fichier `TWSConfig.properties` situé dans `rép_principal_TWA/WAS/TWSprofile/properties`. Pour que cette modification soit prise en compte, redémarrez le gestionnaire de domaine maître. Lorsque vous exécutez la commande, le travail parent est retourné dans la liste des successeurs, mais il n'est pas pris en compte dans le nombre total de travaux successeurs répertoriés. Par exemple, si vous définissez la propriété **com.hcl.tws.conn.plan.rerun.successors.maxjobs** sur dix, le nombre total de successeurs de votre travail parent est dix, alors qu'au total onze travaux seront retournés. Cela est dû au fait que le travail parent est déjà répertorié, car il est planifié pour être réexécuté avec ses successeurs.

L'action de réexécution est toujours effectuée sur la dernière instance de réexécution du travail indiqué. De plus, si vous indiquez le numéro de travail d'un travail intermédiaire dans la séquence de réexécution, l'action est effectuée sur le dernier travail dans la séquence de réexécution.

Les successeurs du travail ne peuvent être réexécutés que s'ils sont dans un état déterminé. Par exemple, les successeurs dans des états intermédiaires, tels que EXEC, WAIT, INTRO, ne peuvent pas être réexécutés. Pour une liste complète, voir tableau 67.

Tableau 67. Statut des successeurs

Statut	Comportement attendu
WAIT	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
INTRO	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
EXEC	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
externe	Une erreur est retournée et l'opération de réexécution n'est pas effectuée

Tableau 67. Statut des successeurs (suite)

Statut	Comportement attendu
ABENDP/SUCCP	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
READY	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
PEND	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
SUPPR (flot de travaux)	Une erreur est retournée et l'opération de réexécution n'est pas effectuée
HOLD	Le prédécesseur du travail à l'état HOLD est réexécuté, mais la séquence de réexécution s'arrête au travail à l'état HOLD
BOUND	Le prédécesseur du travail à l'état BOUND est réexécuté, mais la séquence de réexécution s'arrête au travail à l'état BOUND
FENCE	Le prédécesseur du travail à l'état FENCE est réexécuté, mais la séquence de réexécution s'arrête au travail à l'état FENCE
SUPPR (travail)	L'opération de réexécution est effectuée
SUCC	L'opération de réexécution est effectuée
CANCEL	L'opération de réexécution est effectuée

Exemples

Pour retourner une liste de tous les successeurs du travail ayant échoué (travail parent) avec leur statut, entrez la commande suivante :

```
Rerunsucc MDM94FP1#RequestInfo.UpdateData
```

Une sortie similaire à celle-ci est retournée :

```

Successeurs dans le même flot de travaux :
MDM94FP1#RequestInfo.UpdateFunction          SUCC
SUCCMDM04FP1#RequestInfo.NotifyOfTheRequestReived  SUCC

Successeurs à réexécuter dans d'autres flots de travaux :
MDM94FP1#BatchProcessing.UpdateFunction          SUCC
MDM94FP1#BatchProcessing.EvaluateRisk            SUCC
....
MDM94FP1#ReportProcessing.RunReport              SUCC
Voulez-vous exécuter tous les successeurs, internes et externes ? Y
```

Pour exécuter la commande en mode par lots et réexécuter tous les successeurs internes sans confirmation de la part de l'utilisateur, entrez la commande suivante :

```
Rerunsucc MDM94FP1#RequestInfo.UpdateData;internal
```

Le résultat affiché est du type suivant :

```

Successeurs dans le même flot de travaux :

MDM94FP1#RequestInfo.UpdateFunction          SUCC
MDM04FP1#RequestInfo.NotifyOfTheRequestReceived  SUCC
.....
```

Pour exécuter la commande en mode par lots et réexécuter tous les successeurs, internes et externes, sans confirmation de la part de l'utilisateur, entrez la commande suivante :

```
Rerunsucc MDM94FP1#RequestInfo.UpdateData;all
```

Le résultat affiché est du type suivant :

MDM94FP1#RequestInfo.UpdateFunction	SUCC	
MDM04FP1#RequestInfo.NotifyOfTheRequestReceived		SUCC
MDM94FP1#BatchProcessing.UpdateFunction		SUCC
MDM94FP1#BatchProcessing.EvaluateRisk		SUCC
MDM94FP1#ReportProcessing.RunReport		SUCC
.....		
.....		

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. Dans la table contenant la liste des travaux, sélectionnez un travail, puis cliquez sur **Actions supplémentaires > Réexécuter avec les successeurs**. Une boîte de dialogue répertorie tous les successeurs, internes et externes, dans deux tables distinctes. Dans la boîte de dialogue, vous pouvez choisir si vous souhaitez réexécuter tous les successeurs dans le même flot de travaux ou tous les successeurs dans le même flot de travaux et dans d'autres flots de travaux éventuels.

resetFTA

Génère un fichier Sinfonia mis à jour et envoie celui-ci à l'agent tolérant aux pannes sur lequel le fichier Symphony a été endommagé.

Remarque : La suppression et le remplacement du fichier Symphony entraînent la perte de certaines données, par exemple les événements relatifs aux statuts de travail ou le contenu du message Mailbox.msg et des files d'attente de messages tomaster.msg. Si des informations d'état concernant un travail figuraient dans ces files d'attente, ce travail est réexécuté. Il est recommandé d'appliquer cette commande avec précaution.

Lors du processus, les fichiers suivants sont déplacés vers le répertoire *rep_base_TWA/TWS/tmp* :

- Appserverbox.msg
- clbox.msg
- Courier.msg
- Intercom.msg
- Mailbox.msg
- Monbox.msg

- Moncmd.msg
- Symphony
- Sinfonia

Avant l'exécution de la commande, un message d'information s'affiche pour demander une confirmation et s'assurer que la commande n'est pas émise par erreur. Si l'un des fichiers cible ne peut pas être déplacé parce qu'il est utilisé par un autre processus (par exemple, si le processus mailman est toujours en cours d'exécution), l'opération n'est pas effectuée et un message d'erreur s'affiche.

Autorisation

Vous devez avoir l'accès **RESETFTA** à l'agent tolérant aux pannes que vous souhaitez réinitialiser.

Syntaxe

UC **resetFTA**

Arguments

UC Représente l'agent tolérant aux pannes à réinitialiser.

Cette commande n'est pas disponible dans Dynamic Workload Console.

Exemples

Pour réinitialiser l'agent tolérant aux pannes avec le nom omaha, exécutez la commande suivante :

```
resetFTA omaha
```

Voir aussi

Pour plus d'informations concernant la procédure de reconstitution d'un agent tolérant aux pannes, consultez la section relative à la procédure de reconstitution d'un agent tolérant aux pannes dans *IBM Workload Scheduler - Guide d'identification et résolution des problèmes..*

resource

Modifie le nombre total d'unités d'une ressource.

Vous devez avoir un accès **resource** à la ressource.

Syntaxe

```
{resource | reso} [poste_de_travail#]  
ressource;num  
[;noask]
```

Arguments

Poste de travail

Indique le nom du poste de travail sur lequel la ressource est définie. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé.

ressource

Indique le nom de la ressource.

nombre Indique le nombre total d'unités de ressources. Les valeurs correctes sont comprises entre **0** et **1024**.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque ressource éligible.

Exemples

Pour remplacer le nombre d'unités de la ressource `tapes` par **5**, exécutez la commande suivante :

```
resource tapes;5
```

Pour remplacer le nombre d'unités de la ressource `jobslots` du poste de travail `site2` par **23**, exécutez la commande suivante :

```
reso site2#jobslots;23
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Ressource**.
4. Dans la liste déroulante **Requête**, sélectionnez **Toutes les ressources du plan** ou une autre tâche pour surveiller les ressources.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table des résultats, sélectionnez une ressource et cliquez sur **Modifier des unités....**

setsym

Sélectionne un fichier d'archivage de plan de production. Les écrans de commande ultérieurs affichent le contenu du plan de production archivé. Les informations contenues dans un fichier d'archivage de plan de production ne peuvent pas être modifiées.

Syntaxe

```
{setsym | set} [essai | prévision] [num_fichier]
```

Arguments

Essai Répertorie les plans d'essai.

Prévision

Répertorie les plans prévisionnels.

num_fichier

Indique le numéro du fichier d'archivage de plan de production. Si aucun numéro de fichier journal n'est indiqué, le pointeur retourne en position

zéro (plan de production en cours (Symphony)). Utilisez la commande **listsym** pour afficher la liste des numéros de fichiers d'archivage.

Exemples

Pour sélectionner le fichier archive du plan de production 5, exécutez la commande suivante :

```
setsym 5
```

Pour sélectionner le plan de production courant (fichier Symphony), exécutez la commande suivante :

```
set
```

Voir aussi

Dans Dynamic Workload Console :

1. Dans la barre de navigation, cliquez sur **Planification > Prévission de charge de travail > Gestion des Plans disponibles**.
2. Sélectionnez un moteur.
3. Cliquez sur **Plans archivés** ou entrez un nom de fichier pour le plan.
4. Cliquez sur **Afficher liste des plans**.

showcpus

Affiche des informations relatives aux postes de travail et aux liaisons.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute sur les postes de travail. Le fait que **batchman** soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande `conman start`.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
{showcpus | sc} [[domaine!]poste_de_travail]
                [;info |;link]
                [;offline]
```

```
{showcpus | sc} [[domaine!]poste_travail] [:getmon]
```

Arguments

domaine

Indique le nom d'un domaine. Par défaut, le programme utilise le domaine dans lequel la commande est en cours d'exécution.

poste de travail

Indique le nom d'un poste de travail. Par défaut, le programme utilise le poste de travail sur lequel la commande est en cours d'exécution. Si vous omettez le domaine et le poste de travail, le programme génère les résultats suivants :

- La commande suivante affiche tous les postes de travail figurant dans le domaine du poste de travail où la commande a été exécutée, ainsi que les gestionnaire de domaine connectés si le poste de travail est un gestionnaire de domaine.

```
conman "sc"
```

- La commande suivante affiche tous les postes de travail figurant dans le domaine du poste de travail où la commande a été exécutée, sans les gestionnaire de domaine connectés.

```
conman "sc @"
```

info Affiche les informations au format **info**.

link Affiche les informations au format **link**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

getmon

Renvoie la liste des règles d'événement définies pour le programme de surveillance s'exécutant sur le poste de travail spécifié selon le format suivant :

```
<nom_règle>::<eventProvider>#<eventType>:<scope>
```

La portée de la règle est automatiquement générée : informations sur les attributs de règle, tels que les postes de travail où elle est utilisée, un travail ou un nom de fichier, etc.

L'en-tête de la sortie contient également l'horodatage de la dernière génération du module de configuration de règle.

Remarque : Cette option n'est pas valide sur les postes de travail Dynamic Workload Broker (ou les gestionnaires de domaine dynamique). Dans ce cas, vous pouvez récupérer les informations concernant les règles actives définies dans ces postes de travail dans le fichier *TWA_home\TWS\monconf\TWSObjectsMonitor.cfg* sur le gestionnaire de domaine maître.

Résultats

Lorsque le paramètre **getmon** n'est pas utilisé, la sortie de la commande est produite selon trois formats **standard**, **info** et **lien**. La valeur par défaut est **standard**. La signification des caractères affichés dépend du type de format que vous sélectionnez.

Lorsque vous effectuez l'exécution sur un poste de travail dont la version de IBM Workload Scheduler est antérieure à la version 8.6, la commande **sc** présente comme agents tolérants aux pannes les types de postes de travail introduits avec le pool, le pool dynamique, l'agent et le moteur distant IBM Workload Scheduler version 8.6.

Lorsque le paramètre **getmon** est utilisé, la liste des règles est fournie dans une sortie séparée.

Exemples

1. Pour afficher les informations relatives au poste de travail sur lequel vous exécutez le programme **conman** au format **info**, exécutez la commande suivante :

```
showcpus ;info
```

Voici un exemple de sortie de cette commande :

```
CPUID      VERSION  TIME ZONE      INFO
MASTER    8.6.0.0  US/Pacific    Linux 2.6.5-7.191-s390 #1 SM
FTA1       8.6.0.0          Linux 2.4.9-e.24 #1 Tue May
FTA2       8.6.0.0          HP-UX B.11.11 U 9000/785
```

2. Pour afficher les informations de liaison relatives à tous les postes de travail, exécutez la commande suivante :

```
sc @!@;link
```

Voici un exemple de sortie :

```
CPUID      HOST      FLAGS ADDR  NODE
MASTER    MASTER    AF T 51099 9.132.239.65
FTA1      FTA1      AF T 51000 CPU235019
FTA2      FTA2      AF T 51000 9.132.235.42
BROKER1    MASTER    A T 51111 9.132.237.17
```

3. Pour afficher les informations relatives au poste de travail, exécutez la commande suivante :

```
showcpus
```

Si vous exécutez la commande dans un environnement où la liaison principale du poste de travail avec son domaine ou gestionnaire supérieur n'est pas active, le programme génère la sortie suivante :

```
CPUID  RUN  NODE  LIMIT FENCE DATE  TIME  STATE  METHOD DOMAIN
MASTER 360 *WNT MASTER 10 0 03/05/2010 1348 I J E MASTERDM
FTA1 360 WNT FTA 10 0 03/05/2010 1348 FTI JW M MASTERDM
FTA2 360 WNT FTA 10 0 03/05/2010 1348 FTI JW M MASTERDM
FTA3 360 WNT MANAGER 10 0 03/05/2010 1348 LTI JW M DOMAIN1
FTA4 360 WNT FTA 10 0 03/05/2010 1348 F I J M DOMAIN1
FTA5 360 WNT FTA 10 0 03/05/2010 1348 I J M DOMAIN1
SA1 360 WNT S-AGENT 10 0 03/05/2010 1348 F I J M DOMAIN1
XA_FTA4 360 OTHR X-AGENT 10 0 03/05/2010 1348 L I J M DOMAIN1
FTA6 360 WNT MANAGER 10 0 03/05/2010 1348 F I J M DOMAIN2
FTA7 360 WNT FTA 10 0 03/05/2010 1349 F I J M DOMAIN2
FTA7 360 WNT FTA 10 0 03/05/2010 1349 F I J M DOMAIN2
BROKER 360 OTHR BROKER 10 0 03/05/2010 1349 LTI JW MASTERDM
```

Si vous exécutez la commande dans un environnement où la liaison principale du poste de travail avec son domaine ou gestionnaire supérieur n'est pas active où au moins une liaison secondaire est active, le programme génère la sortie suivante :

```
CPUID  RUN  NODE  LIMIT FENCE DATE  TIME  STATE  METHOD DOMAIN
MASTER 360 *WNT MASTER 10 0 03/05/2010 1348 I J E MASTERDM
FTA1 360 WNT FTA 10 0 03/05/2010 1348 FTI JW M MASTERDM
FTA2 360 WNT FTA 10 0 03/05/2010 1348 FTI JW M MASTERDM
FTA3 360 WNT MANAGER 10 0 03/05/2010 1348 FTI JW M DOMAIN1
FTA4 360 WNT FTA 10 0 03/05/2010 1348 F I J M DOMAIN1
FTA5 360 WNT FTA 10 0 03/05/2010 1348 L I M DOMAIN1
SA1 360 WNT S-AGENT 10 0 03/05/2010 1348 F I J M DOMAIN1
XA_FTA4 360 OTHR X-AGENT 10 0 03/05/2010 1348 L I J M DOMAIN1
FTA6 360 WNT MANAGER 10 0 03/05/2010 1348 F I J M DOMAIN2
FTA7 360 WNT FTA 10 0 03/05/2010 1349 F I J M DOMAIN2
```

Si vous exécutez la commande dans un environnement où la liaison principale du poste de travail avec son domaine ou gestionnaire supérieur et toutes les liaisons secondaires sont actives, le programme génère la sortie suivante :

CPUID	RUN	NODE	LIMIT	FENCE	DATE	TIME	STATE	METHOD	DOMAIN
MASTER	360	*WNT MASTER	10	0	03/05/2010	1348 I J E	MASTERDM		
FTA1	360	WNT FTA	10	0	03/05/2010	1348 FTI JW M	MASTERDM		
FTA2	360	WNT FTA	10	0	03/05/2010	1348 FTI JW M	MASTERDM		
FTA3	360	WNT MANAGER	10	0	03/05/2010	1348 FTI JW M	DOMAIN1		
FTA4	360	WNT FTA	10	0	03/05/2010	1348 F I J M	DOMAIN1		
FTA5	360	WNT FTA	10	0	03/05/2010	1348 F I M	DOMAIN1		
SA1	360	WNT S-AGENT	10	0	03/05/2010	1348 F I J M	DOMAIN1		
XA_FTA4	360	OTHR X-AGENT	10	0	03/05/2010	1348 L I J M	DOMAIN1		
FTA6	360	WNT MANAGER	10	0	03/05/2010	1348 F I J M	DOMAIN2		
FTA7	360	WNT FTA	10	0	03/05/2010	1349 F I J M	DOMAIN2		

4. Pour obtenir une liste des moniteurs de règles actifs sur le poste de travail CPU1, exécutez cette commande :

```
sc CPU1 getmon
```

Vous obtenez la sortie suivante :

```
Monitoring configuration for CPU1:
*****
*** Package Date : 04/22/2009 12:00 GMT ***
*****
Rule1::FileMonitor#FileCreated:Workstation=CPU1,CPU2;File="\tmp\filename"
Rule2::FileMonitor#ModificationCompleted:Workstation=CPU1,CPU3;File="\staging\orders"
Rule3::TWSObjectsMonitor#JobSubmit:JobKey=CPU1#JS1.Job1
Rule5::TWSObjectsMonitor#JobLate:JobKey=CPU1#JS1.Job1
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

ID poste de travail

Nom du poste de travail auquel ces informations s'appliquent.

EXECUTION

Numéro d'exécution du fichier Symphony.

NOEUD

Type de noeud et type de poste de travail. Les types de noeud sont les suivants :

- UNIX
- WNT
- OTHER
- ZOS
- IBM i

Les types de poste de travail sont les suivants :

- MASTER
- GESTIONNAIRE
- FTA
- S-AGENT

- X-AGENT
- AGENT
- POOL
- D-POOL
- REM-ENG

LIMIT

Nombre maximal de travaux IBM Workload Scheduler.

FENCE

Priorité minimale de travail IBM Workload Scheduler.

DATE HEURE

Date et heure de début de l'exécution du plan de production courant (fichier Symphony) par IBM Workload Scheduler.

ETAT Affiche les informations suivantes :

- L'état des liens et processus du poste de travail. Jusqu'à cinq caractères sont affichés comme suit. L'explication des caractères est divisée en fonction de la portée des caractères :

[L|F] [T|H|X|B] [I] [J] [W|H|X] [M] [E|e] [D] [A|R]

Où :

L La liaison principale est ouverte ("linked") sur le gestionnaire de domaine/supérieur.

Si le poste de travail est de type agent ou moteur distant, cet indicateur précise que le poste de travail est connecté au serveur Workload Broker.

Si le poste de travail est de type pool ou pool dynamique, cet indicateur précise que le poste de travail de courtier de charge de travail auprès duquel le pool ou pool dynamique est enregistré est lié à son gestionnaire de domaine/supérieur.

F Le poste de travail est complètement connecté via les liaisons principale et secondaires. Cet indicateur ne s'affiche que si l'option globale *enSwfaultTol* a pour valeur YES à l'aide de la ligne de commande **optman** sur le gestionnaire de domaine maître. Il indique que le poste de travail est directement lié à son gestionnaire de domaine, ainsi qu'à tous ses gestionnaires de domaine de sauvegarde intégrale. Pour plus d'informations sur l'utilisation de la ligne de commande **optman**, voir *IBM Workload Scheduler - Guide d'administration*.

T Cet indicateur s'affiche si l'agent tolérant aux pannes est directement connecté au gestionnaire de domaine à partir duquel vous exécutez la commande.

H Le poste de travail est connecté via son hôte.

X Le poste de travail est connecté comme agent étendu (x-agent).

B Le poste de travail communique au moyen du serveur de courtier de charge de travail.

I Si le poste de travail est de type agent, MASTER, MANAGER, FTA, S-AGENT, X-AGENT, cet indicateur précise que le programme **jobman** a terminé l'initialisation du démarrage.

Si le poste de travail est de type agent, pool ou pool dynamique, cet indicateur précise que l'agent est correctement initialisé.

Si le poste de travail est de type moteur distant, cet indicateur précise que la communication entre le poste de travail de moteur distant et le moteur distant s'est initialisée correctement.

- J** Si le poste de travail est de type agent MASTER, MANAGER, FTA, S-AGENT, X-AGENT, cet indicateur précise que le programme **jobman** est en cours d'exécution.
- Si le poste de travail est de type agent, cet indicateur précise que JobManager est en cours d'exécution. Etant donné qu'aucune surveillance n'est effectuée sur les postes de travail du pool dynamique, le caractère J apparaît toujours pour ce type de poste de travail.
- Si le poste de travail est de type pool, cet indicateur précise que le processus JobManager est en cours d'exécution sur au moins un agent enregistré auprès du pool.
- Si le poste de travail est de type moteur distant, cet indicateur précise que la commande ping lancée sur le moteur distant a réussi.
- W** Le poste de travail est lié via le protocole TCP/IP à l'aide du processus **writer**.
- Si le poste de travail exécutant **conman** est directement lié au poste de travail à distance, l'indicateur W apparaît car le processus mailman local est lié au processus writer.
- LTI JW
- Si le poste de travail exécutant **conman** est directement lié au poste de travail à distance, l'indicateur W n'apparaît pas car le processus mailman local n'est pas lié au processus writer.
- L I J
- Pour plus de détails sur le processus **writer**, voir la rubrique relative aux processus réseau dans *IBM Workload Scheduler - Guide d'administration*.

Remarque : Si le poste de travail exécutant **conman** correspond à l'hôte de l'agent étendu, l'agent étendu est à l'état

LXI JX

Si le poste de travail exécutant **conman** n'est pas l'hôte de l'agent étendu, l'agent étendu est à l'état

LHI JH

- L'état de l'agent de surveillance. Jusqu'à trois caractères sont affichés comme suit :

[M] [E|e] [D]

où :

- M** Le processus monman s'exécute. Cet indicateur est affiché pour tous les postes de travail du réseau lorsque la fonction d'automatisation de la charge de travail gérée par des événements est activée (l'option globale de `enEventDrivenWorkloadAutomation` est définie sur `yes`), à l'exception des postes de travail sur lesquels monman a été arrêté manuellement (au moyen de **conman** ou de Dynamic Workload Console).
- E** Le serveur de traitement d'événements est installé et s'exécute sur le poste de travail.
- e** Le serveur de traitement d'événements est installé sur le poste de travail mais ne s'exécute pas.

- D Le poste de travail utilise une configuration de surveillance de module mise à jour. Cet indicateur est affiché pour les postes de travail sur lesquels le module le plus récent de règles d'événement a été déployé (soit manuellement au moyen de la commande `planman deploy` ou automatiquement à la fréquence spécifiée par l'option globale `deploymentFrequency`).
- L'état du serveur WebSphere Application Server. Un indicateur à un caractère est affiché si le serveur d'application est installé.
[A|R]

où :

- A Le serveur WebSphere Application Server a été démarré.
 - R Le serveur WebSphere Application Server redémarre.
- L'indicateur est vide si le serveur d'applications est en panne ou s'il n'a pas été installé.

METHODE

Nom de la méthode d'accès indiquée dans la définition du poste de travail.
Réservé uniquement aux agents étendus.

DOMAINE

Nom du domaine auquel appartient le poste de travail.

Format info

ID poste de travail

Nom du poste de travail auquel ces informations s'appliquent.

VERSION

Version de l'agent IBM Workload Scheduler installée sur le poste de travail.

FUSEAU HORAIRE

Fuseau horaire du poste de travail. Il s'agit de la même valeur que celle de la variable d'environnement TZ. Pour un agent étendu, il s'agit du fuseau horaire de son hôte. Pour un poste de travail de moteur distant, il s'agit du fuseau horaire du moteur distant.

INFO Zone d'informations. Elle contient la version du système d'exploitation et le modèle de matériel de tous les types de poste de travail (à l'exception des postes de travail de l'agent étendu et du courtier). Pour les agents étendus et les postes de travail de moteur distant, aucune information n'est répertoriée. Pour les postes de travail de moteur distant, cette zone affiche le moteur distant.

Format link

ID poste de travail

Nom du poste de travail auquel ces informations s'appliquent.

HOTE Nom du poste de travail servant d'hôte à un agent standard ou à un agent étendu. Pour les gestionnaires de domaine et les agents tolérants aux pannes, ce nom est identique à l'ID du poste de travail. Pour les postes de travail standard d'un agent et d'un courtier, il s'agit du nom du gestionnaire de domaine. Pour les agents étendus, ce nom correspond à celui du poste de travail hôte.

INDICATEURS

Etat des propriétés du poste de travail. Il peut compter jusqu'à cinq caractères et s'affiche comme suit :

[A] [B] [F] [s] [T]

A La liaison automatique est activée dans la définition du poste de travail.

B Cet indicateur est uniquement utilisé dans un environnement complet et précise si l'indicateur **deactivate job launching** est désactivé.

F Le mode Statut intégral est activé dans la définition du poste de travail.

s ID du serveur **mailman** du poste de travail.

T La liaison est définie comme TCP/IP.

ADR Numéro de port TCP/IP du poste de travail.

NOEUD

Nom de noeud du poste de travail.

showdomain

Affiche les informations relatives au domaine.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande conman start.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
{showdomain | showdom | sd} [domaine]
    [;info]
    [;offline]
```

Arguments

domaine

Indique le nom du domaine. Par défaut, le domaine dans lequel **conman** est en cours d'exécution est utilisé. Les caractères génériques sont acceptés.

info Affiche les informations au format **info**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Résultats

La sortie de la commande est générée dans deux formats : **standard** et **info**.

Exemples

Pour afficher les informations relatives au domaine masterdm, exécutez la commande suivante :

```
showdomain masterdm
```

Voici un exemple de sortie :

DOMAIN	MANAGER	PARENT
*MASTERDM	*MASTER	

Pour afficher les postes de travail appartenant à tous les domaines au format **info**, exécutez la commande suivante :

```
showdomain @;info
```

Voici un exemple de sortie :

DOMAIN	MEMBER-CPUs	CPU-Type
MASTERDM	*MASTER	MASTER
DOM1	FTA1	MANAGER
DOM2	FTA2	MANAGER

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Domaine**.
4. Dans la liste déroulante **Requête**, sélectionnez une tâche pour surveiller les domaines.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

DOMAINE

Nom du domaine auquel ces informations s'appliquent.

GESTIONNAIRE

Nom du gestionnaire de domaine.

PARENT

Nom du domaine parent.

Format info

DOMAINE

Nom du domaine auquel ces informations s'appliquent.

POSTES DE TRAVAIL MEMBRES

Noms des postes de travail du domaine.

TYPE POSTE DE TRAVAIL

Type de chaque poste de travail : MASTER, MANAGER, FTA, S-AGENT, X-AGENT ou BROKER.

showfiles

Affiche des informations relatives aux dépendances sur les fichiers. Une dépendance de fichier se produit lorsque la présence d'un ou de plusieurs fichiers permet l'exécution d'un travail ou d'un flot de travaux.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande `conman start`.

Syntaxe

```
{showfiles |  
sf} [[num_poste_de_travail]fichier]  
    [:état[:...]]  
    [:keys]  
    [:offline]
```

```
{showfiles |  
sf} [[poste de travail#]fichier]  
    [:état[:...]]  
    [:deps[:keys | info | logon]]  
    [:offline]
```

Arguments

Poste de travail

Indique le nom du poste de travail sur lequel se trouve le fichier. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé. Les caractères génériques sont acceptés.

fichier Indique le nom du fichier. Ce nom doit être placé entre guillemets (") s'il contient des caractères autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inversées (\) et traits de soulignement (_). Par défaut, toutes les dépendances sur les fichiers s'affichent. Les caractères génériques sont acceptés.

état Indique l'état des dépendances de fichiers à afficher. Par défaut, les dépendances sur les fichiers s'affichent avec tous les états. Les états sont les suivants :

Yes Le fichier existe et est disponible.

Non Le fichier n'est pas disponible ou n'existe pas.

? La disponibilité est en cours de vérification.

<vide>

Le fichier n'a pas encore été vérifié ou était disponible et a été utilisé pour respecter une dépendance de travail ou flot de travaux.

keys Affiche la liste composée d'une seule colonne des objets sélectionnés par la commande.

deps Affiche les informations au format **deps**. Utilisez **keys**, **info** ou **logon** pour modifier l'affichage.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Résultats

La sortie de la commande est générée dans trois formats : **standard**, **keys** et **deps**. Les arguments **keys**, **info** et **logon** modifient l'affichage **deps**.

Exemples

Pour afficher l'état d'une dépendance de fichier sur le fichierd:\apps\mis\lib\data4, exécutez la commande suivante :

```
showfiles d:\apps\mis\lib\data4
```

Pour afficher **hors ligne** l'état de toutes les dépendances de fichier sur tous les postes de travail au format deps, exécutez la commande suivante :

```
sf @#@;deps;offline
```

Pour afficher l'état de toutes les dépendances de fichier sur tous les postes de travail au format deps, exécutez la commande suivante :

```
sf @#@;deps
```

Voici un exemple de sortie :

```

                                (Est) (Est)
Workstation Job Stream SchedTime Job      State Pr Start Elapse ReturnCode Dependencies
MASTER#/test/^\LFILEJOB^ Les dépendances sont les suivantes :
MASTER      #LFILEJOB  0600 11/26 ***** READY 10
                                LFILEJOB HOLD 10 (11/26)
                                ^LFILEJOB^
MASTER#/usr/home/me10_99/~ /usr/home/me10_99/bin/parms FILE_JS1~ Les dépendances sont les suivantes :
MASTER      #FILE_JS1  0600 11/26 ***** HOLD 10 (11/26)
                                FILE_JS1 HOLD 10 (11/26)
                                parms FILE_JS1~
MASTER#/usr/home/me10_99/~ /usr/home/me10_99/bin/parms FILE_JB1~ Les dépendances sont les suivantes :
MASTER      #FILE_JB1  0600 11/26 ***** READY 10
                                FILE_JB1 HOLD 10 (11/26)
                                parms FILE_JB1~
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Fichier**.
4. Dans la liste déroulante **Requête**, sélectionnez une tâche pour surveiller les fichiers.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

Existe Etat de la dépendance de fichier

Nom de fichier

Nom du fichier.

Format Keys

Les fichiers sont répertoriés dans une liste, chacun d'eux sur une ligne différente. Les noms de répertoires ne sont pas inclus. Chaque fichier est répertorié au format suivant :

```
workstation#file
```

Format Deps

Les fichiers sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** standard. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;keys

Les travaux et les flots de travaux qui comportent des dépendances file sont répertoriés, un par ligne, au format suivant :

poste_de_travail#flot_travaux[.job]

Format Deps;info

Les fichiers sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;info**. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;logon

Les fichiers sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;logon**. Les flots de travaux sont répertoriés au format **showschedules** standard.

showjobs

Affiche les informations relatives aux travaux.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande conman start.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
{showjobs |  
sj} [travail_sélectionné]  
    [;keys | info | step | logon | crit | keys retcod]  
    [;short | single]  
    [;offline]  
    [;showid]
```

```
{showjobs |  
sj} [travail_sélectionné]  
    [;deps[;keys | info | logon]]  
    [;short | single]  
    [;offline]  
    [;showid]  
    [;props]
```

```
{showjobs  
| sj} [travail_sélectionné |  
    [poste_de_travail#]jobnumber.hhmm]  
    [;stdlist[;keys]]  
    [;short | single]  
    [;offline]  
    [;showid]  
    [;props]
```

Arguments

crit Affiche des informations au format **crit**.

- deps** Affiche des informations au format **deps**. En d'autres termes, les travaux utilisés dans les dépendances de *prédécesseur/successeur* sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** de base. Les flots de travaux sont répertoriés au format **showschedules** de base. Utilisez "keys", "info" ou "logon" pour modifier l'affichage "deps".
- hmmm** Heure de début du travail. Utilisez cet argument avec les arguments **stdlist** et **single** pour afficher une instance spécifique du travail.
- info** Affiche les informations au format **info**.

Remarque : Lors de l'affichage de la sortie pour un travail, la définition de travail ne s'affiche pas correctement dans **USERJOBS** et les conditions de sortie ne sont pas présentées. Toutefois, les informations correctes apparaissent dans **stdlist**.

numéro_travail

Numéro du travail.

travail_sélectionné

Voir «Sélection de travaux dans les commandes», à la page 419.

- keys** Affiche la liste composée d'une seule colonne des objets sélectionnés par la commande.

- logon** Affiche les informations au format **logon**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

- props** Affiche les informations suivantes sur l'instance de travail spécifiée, vous devez avoir un accès Display aux propriétés de l'instance de travail spécifiée qui est affichée :

Informations générales

- Travail
- Poste de travail
- Tâche
- Type de tâche
- Flot de travaux
- Poste de travail du flot de travaux
- Heure planifiée
- Priorité
- Connexion
- Monitored
- Confirmation requise
- Interactif
- Critique

Informations d'exécution

- Poste de travail réel
- Statut
- Statut interne
- Dépendances non satisfaites

/

- Numéro du travail
- Options de réexécution
- Information
- Promu
- Code retour
- Expression du mappage de code retour
- Conditions de sortie de réussite liées à un statut de travail SUCC
- Autres conditions de sortie

Time Information

- Démarrage réel
- Premier démarrage
- Dernier démarrage
- Dernière action de démarrage
- Durée maximale
- Action durée maximale
- Durée minimale
- Action durée minimale
- Dernier démarrage critique
- Échéance
- Intervalle de répétition
- Durée réelle
- Durée estimée
- Intervalle de confiance

Informations de reprise

- Action
- Message
- Définition de travail
- Poste de travail
- Réessayer après
- Nombre de tentatives
- Tentative en cours
- Exécution sur le même poste de travail

=
=
=
=

Informations supplémentaires

Cette section présente des propriétés supplémentaires spécifiques pour les travaux reflet et les travaux définis par le JSDL. Pour les travaux reflet, elle contient les informations suivantes :

Pour les travaux reflet distribués :

- Heure planifiée du travail distant
- Travail distant
- Flot de travaux distant
- Poste de travail du flot de travaux distant

Pour les travaux reflet z/OS :

- Heure planifiée du travail distant
- Travail distant

- Poste de travail du travail distant
- Code d'erreur du travail distant

Pour plus d'informations, voir «Modification de l'état du travail reflet après que la liaison est établie», à la page 857.

Remarque : Les informations sur les travaux archivés ne peuvent pas être extraites à l'aide de l'option **props**.

Remarque : Lors de l'affichage de la sortie pour un travail, la définition de travail ne s'affiche pas correctement dans USERJOBS et les conditions de sortie ne sont pas présentées. Toutefois, les informations correctes apparaissent dans **stdlist**.

retcod Affiche le code retour du travail. Cet argument doit être utilisé avec l'argument **keys**, par exemple :

```
%sj @; keys retcod
```

short Réduit l'affichage des travaux **réexécutés** et **répétitifs** afin d'inclure uniquement ce qui suit :

- Première répétition
- Travaux avec des états différents
- Travaux concordant exactement

Remarque : Cette zone affiche les propriétés spécifiques si le travail est un travail reflet ou un travail défini par JSDL.

showid

Affiche l'identificateur de chaque flot de travaux.

single Sélectionne uniquement le travail parent d'une chaîne pouvant inclure des ré exécutions, des répétitions et des travaux de reprise. Le travail doit être identifié par son numéro dans *travail_sélectionné*. Cette fonction est utile avec l'option **stdlist**.

stdlist Affiche les informations au format **stdlist**. Utilisez l'argument **keys** pour modifier l'affichage.

Remarque : Les informations sur les travaux archivés ne peuvent pas être extraites à l'aide de l'option **stdlist**.

step Affiche les informations au format **step**.

poste de travail

Nom du poste de travail sur lequel le travail est exécuté. Les caractères génériques sont acceptés.

Commentaires

Si un travail échoue parce que l'agent est indisponible, le travail est automatiquement relancé et défini sur le statut PRÊT, puis attend que l'agent se reconnecte. Dès que l'agent se reconnecte de nouveau, le travail est soumis.

Résultats

La sortie de la commande **showjobs** est générée dans huit formats : **standard**, **keys**, **info**, **step**, **logon**, **deps**, **crit** et **stdlist**. Les arguments **keys**, **info**, **crit** et **logon** modifient l'affichage.

Exemples

Pour afficher le statut de tous les travaux du flot de travaux acctg sur le poste de travail site3, vous pouvez exécuter la commande showjobs selon l'un de ces deux formats :

```
showjobs site3#acctg.@
```

ou :

```
showjobs site3#acctg
```

Pour afficher le statut du travail JBA appartenant au flot de travaux TEST1 sur le poste de travail CPUA, sur lequel vous exécutez **conman**, et demander l'affichage de l'identificateur du flot de travaux, exécutez la commande suivante :

```
sj CPUA#TEST1(0900 02/19/15).JBA
```

Voici un exemple de sortie de cette commande :

```
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies
CPUA          #TEST1    0900 02/19 *** HOLD    0(02/19)                {02/20/15}; -TEST-
                                   JBA HOLD  66(14:30)                J2(0600 02/24/15).JB1
```

- La dépendance **at** est définie sur (14:30) dans la colonne Démarrage et la dépendance de prédécesseur/successeur entre le travail J2(0600 02/24/15).JB1 et le travail J0BA est indiquée dans la colonne Dépendances.
- Dans la colonne Dépendances, la date placée entre accolades, {02/20/15}, indique que l'instance de flot de travaux a été reportée et la date indique le jour où elle a été ajoutée au plan de production pour la première fois.

L'exemple de résultats suivant affiche le statut de tous les travaux, y compris les prédécesseurs, et notamment le travail JOBVACS définit dans le flot de travaux JSHOLIDAYS1, qui a une dépendance conditionnelle sur le travail prédécesseur JOBCHECKCALC spécifiant que le travail JOBVACS s'exécute si le travail JOBCHECKCAL passe à l'état ABEND ou FAIL :

```
S_MDM          #JSHOLIDAYS 0600 09/23 ***** READY 10          (00:01)
                                   (S_AGT#)JOBCHECKCAL      HOLD 19
                                   (S_AGT#)JOBVACS        HOLD 10

JOBCHECKCAL IF ABEND | FAIL
```

Pour afficher le statut des travaux appartenant au flot de travaux JSDOC sur le poste de travail site3, sur lequel vous exécutez **conman**, et demander l'affichage de l'identificateur du flot de travaux, exécutez la commande suivante :

```
%sj JSDOC.@;showid
```

Voici un exemple de sortie de cette commande :

```
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies
site3        #JSDOCOM 0600 11/26 *** SUCC  10 11/26 00:01          {0AAAAAAAAAAACRZ}
                                   JDOC SUCC  10 11/26 00:01          0      #J25565
```

L'identificateur 0AAAAAAAAAAACRZ du flot de travaux JDOCOM est indiqué dans la colonne Dépendances.

Remarque : La date ou l'heure affichée dans la colonne **Démarrage** est convertie en fonction du fuseau horaire défini sur le poste de travail sur lequel le flot de travaux doit s'exécuter.

Pour afficher le statut des travaux appartenant au flot de travaux JSDOCOM sur le poste de travail site3, et demander l'affichage des informations sur l'ID utilisateur sous lequel le travail s'exécute, exécutez la commande suivante :

```
sj site3#JSDOCOM.@;logon
```

Voici un exemple de sortie de cette commande :

```
Workstation Job Stream SchedTime Job State Job# Logon ReturnCode
site3        #JSDOCOM 0600 11/26
                                JSDOCOM SUCC #J25565 me10_99 0
```

Pour afficher le statut de tous les travaux à l'état HOLD sur tous les postes de travail au format **deps**, exécutez la commande suivante :

```
sj @#@.@+state=hold;deps
```

Voici un exemple de sortie :

Poste de travail Flot de travaux Heure calend Travail Etat Pr Démarrage Temps écoulé Code retour Dépendances

CPUA#JS2.JOBB Les dépendances sont les suivantes :

```
CPUA      #JS21      0900 02/19 ***** HOLD 0(02/19)          {02/20/15}; -TEST- JOBA HOLD 66(14:30)
                                           JS22(0600 02/24/15).JOBB
```

CPUA#JS25.JOBC Les dépendances sont les suivantes :

```
CPUA      #JS25      0600 02/24 ***** HOLD 10(02/24)        {02/20/15}
                                           jobaa HOLD 10(02/24)(00:01) TEST1; JOBC          TEST2; JOB1
                                           JS18(0600 02/24/15).@
```

CPUA#JS25.JOB1 Les dépendances sont les suivantes :

```
CPUA      #JS25      0600 02/24 ***** HOLD 10(02/24)        {02/20/15}
                                           JOBC HOLD 10(02/24)(00:01) JOB1
                                           jobaa HOLD 10(02/24)(00:01) TEST1; JOBC          TEST2; JOB1
JS18(0600 02/24/15).@
```

Pour afficher le journal à partir des fichiers de liste standard du travail JOBC du flot de travaux JS25(0600 09/24/15) sur le poste de travail CPUA, qui fonctionne dans un environnement UNIX, exécutez la commande suivante :

```
sj CPUA#JS25 (0600 09/24/15).JOBC;stdlist
```

Le programme génère la sortie suivante :

```
=====
= JOB : CPUA#JS25[(0600 09/24/15),(0AAAAAAAAAABQM)].JOBC
= USER : mdm93mdm
= FICHER JCL : ls
= TWSRCMAP :
= AGENT : CPUA
= Job Number: 987278608
= Thu Sep 24 17:06:27 CEST 2015
=====
AE
CAP
IMShared
InstallationManager
tsamp
TWA
WebSphere
=====
```



```
= Exit Status : 0
= SC STATUS_OK : true
= OC OUTPUTCOND2 : false
= OC OUTPUTCOND1 : true= System Time (Seconds) : 0      Elapsed Time (hh:mm:ss) : 00:00:01
= User Time (Seconds) : 0
= Job CPU usage (ms) : 20
= Job Memory usage (kb) : 1272
= Thu Sep 24 15:16:25 CEST 2015
```

où :

Exit Status

Statut du travail à l'issue de son exécution

OC <nom_condition_sortie>

Résultat de l'évaluation des conditions de sortie qui, lorsqu'elles sont satisfaites, déterminent le travail successeur à exécuter. Les conditions de sortie satisfaites affichent true et les conditions de sortie non satisfaites affichent false. Les conditions de sortie satisfaites sont signalées par l'indicateur sc dans la sortie ;**stdlist**.

System Time

Temps passé par le noyau système pour exécuter le travail

Elapsed Time

Durée écoulée du travail

User Time

Temps passé par l'utilisateur système pour exécuter le travail

Remarque : Les zones **Heure système** et **Temps utilisateur** sont utilisées sur UNIX uniquement. Leur valeur sous Windows est toujours 0, ceci parce que, sous Windows, le processus **joblnch.exe** s'exécute en un temps très court qui peut être considéré comme nul.

Pour les postes de travail qui exécutent une **version de l'agent antérieure à la version 9.3**, la zone **Elapsed Time** est exprimée en **heures et minutes** et correspond au temps d'exécution arrondi à la minute, sans prendre en compte les secondes. Par exemple, un temps d'exécution de 4 minutes 20 secondes sera arrondi à 5 minutes ou un temps d'exécution de 10 minutes sera arrondi à 11 minutes. Pour un poste de travail qui exécute la **version 9.3 ou ultérieure de l'agent**, la zone **Elapsed Time** est exprimée en **heures, minutes et secondes** ; toutefois, si le poste de travail s'exécute sous un gestionnaire de domaine maître d'une version antérieure 9.3, seules les **heures et minutes** sont affichées par **showjob**.

Pour afficher les propriétés du travail avec le numéro de travail 227137038, exécutez la commande suivante :

```
sj 227137038;props
```

Voici un exemple de sortie de cette commande :

```
= sj SMA1964199;props
= General Information
= Job = JOBAUTO
= Workstation = NC005090_1
= Task =
= <?xml version="1.0" encoding="UTF-8"?>
= <jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
=   xmlns:jSDL=
=     "http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL">
= <jSDL:application name="executable">
```

```

=      <jsdle:executable interactive="false">
=      <jsdle:script>ping -n 180 localhost</jsdle:script>
=      </jsdle:executable>
=      </jsdl:application>
=      </jsdl:jobDefinition>
=      Task Type = executable
=      Job Stream = SMA1964199
=      Job Stream Workstation = NC005090_1
=      Scheduled Time = 11/20/2014 16:57:00 TZ CET
=      Priority = 50
=      Login =
=      Monitored = No
=      Requires Confirmation = No
=      Interactive = No
=      Critical = No
=
=      Runtime Information
=      Status = Running
=      Internal Status = EXEC
=      Not Satisfied Dependencies = 0
=      Job Number = 227137038
=      Rerun Options =
=      Information =
=      Promoted = No
=      Return Code =
=      Success Output Conditions
=      STATUS_OK = true "RC=0"
=      Other Output Conditions
=      STATUS_ERR1 = false "RC=2"
=      STATUS_ERR2 = false "RC=3"
=
=      Time Information
=      Actual Start = 11/20/2014 16:57:36 TZ CET
=      Earliest Start =
=      Latest Start =
=      Latest Start Action =
=      Maximum Duration =
=      Maximum Duration Action =
=      Minimum Duration =
=      Minimum Duration Action =
=      Critical Latest Start =
=      Deadline =
=      Repeat Range =
=      Actual Duration =
=      Estimated Duration = 00:03:02 (hh:mm:ss)
=      Confidence Interval = 00:00:00 (hh:mm:ss)
=
=      Recovery Information
=      Action = Stop
=      Message =
=      Job Definition =
=      Workstation =
=      Retry after =
=      Number of attempts=
=      Current attempt =
=      Run on same workstation =
=
=      Extra Information
=      PID = 3132

```

Pour une explication des propriétés du travail Durée estimée et Intervalle de confiance, voir «Commande logman», à la page 110.

Dans l'exemple suivant, le programme affiche le statut du travail dbse1oad en générant le code retour 7 et l'état SUCCESSFUL :

```

$ conman sj workstation#DAILY_DB_LOAD
#####
# Eléments sous licence - Propriété d'IBM* et HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#####
Installation réalisée pour l'utilisateur "tme10_99".
Variable locale LANG définie sur : "fr"
Scheduled for (Exp) 02/20/16 (#35) on CPUA.
Batchman ACTIF. Limit:50,Fence:0,Audit Level:0
sj workstation#DAILY_DB_LOAD
(Est) (Est)
UC Calendrier Travail Etat Pr Démarrage
Temps écoulé Dépendances Code retour
WORKSTATION #DAILY_DB_LOAD ***** SUCC 10 22:11
00:04
DATASPLT SUCC 10 22:11
00:01 #J17922 0
DATAMRGE ABEND 10 22:12
00:01 #J17924 1
CHCKMRGE SUCC 10 22:12
00:01 #J17926 0
DATACLNS SUCC 10 22:12
00:01 #J17932 0
DATARMRG SUCC 10 22:13
00:01 #J18704 0
DBSELOAD SUCC 10 22:13
00:01 #J18706 7
DATAREPT SUCC 10 22:13
00:01 #J18712 0
DATARTRN SUCC 10 22:14
00:01 #J18714 0
$

```

L'exemple suivant affiche le code retour d'un travail spécifique intitulé `workstation#daily_db_load.dbseload` :

```

$ conman sj workstation#daily_db_load.dbseload\;keys\;retcod
#####
# Eléments sous licence - Propriété d'IBM* et HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#####
Installation réalisée pour l'utilisateur "tme10_99".
Variable locale LANG définie sur : "fr"
Scheduled for (Exp) 02/20/16 (#35) on CPUA.
Batchman ACTIF. Limit:50,Fence:0,Audit Level:0
sj workstation#daily_db_load.dbseload\;keys\;retcod 8
$

```

Une fois intégrée à un script, la fonction `retcod` peut se révéler assez puissante.

=
=
=
=
=
=
=
=

L'exemple ci-après présente un flot de travaux qui contient un travail (`RE_ACCOUNTS_JOB`) se terminant à l'état `ABEND`. Après l'échec du travail `RE_ACCOUNTS_JOB` parent, le travail de reprise démarre. Une fois ce dernier passé à l'état de réussite, le travail parent attend une minute, puis est réexécuté. La séquence est répétée trois fois si le travail parent continue à échouer. Si ce dernier aboutit, la séquence de réexécution est interrompue à la première exécution réussie :

```

= SCHEDULE NC053009#RE_JS_04
= ON RUNCYCLE RC1 "FREQ=DAILY;INTERVAL=1"
= :
= NC053009_1#JOB_2
= FOLLOWS RE_ACCOUNTS_JOB
= NC053009_1#RE_ACCOUNTS_JOB
= END
=
= NC053009_1#RE_ACCOUNTS_JOB
= DOCOMMAND "sleep 60; exit 5"
= STREAMLOGON bankmgr
= TASKTYPE UNIX
= RECOVERY RERUN REPEATEVERY 0001 FOR 3
= AFTER NC053009_1#RECOVERY_JOB
=

```

Voici la sortie de la commande sj à la fin de la séquence :

```

= NC053009          #RE_JS_04          0000 01/24 ***** STUCK 10 17:40
=                                     (NC053009_1#)JOB_2          HOLD 10          RE_
=                                     (NC053009_1#)RE_ACCOUNTS_JOB      ABEND 10 17:40 00:01      5 #J2
=                                     (NC053009_1#)RECOVERY_JOB      SUCC 10 17:42 00:01      0
= >>recovery          (NC053009_1#)RE_ACCOUNTS_JOB      ABEND 10 17:44 00:01      5
= >>rerun 1 of 3       (NC053009_1#)RECOVERY_JOB      SUCC 10 17:45 00:01      0
= >>recovery          (NC053009_1#)RE_ACCOUNTS_JOB      ABEND 10 17:47 00:01      5
= >>rerun 2 of 3       (NC053009_1#)RECOVERY_JOB      SUCC 10 17:48 00:01      0
= >>recovery          (NC053009_1#)RE_ACCOUNTS_JOB      ABEND 10 17:50 00:01      5
= >>rerun 3 of 3       (NC053009_1#)RECOVERY_JOB
=                                     [Recovery]
=

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Travail**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les travaux du plan** ou une autre tâche pour surveiller les travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du travail dans le plan.

Travail

Nom du travail. Il se peut que la mention suivante précède un nom de travail :

>> rerun as

Travail réexécuté à l'aide de la commande **rerun** ou par suite d'une reprise automatique.

>> rerun nombre_réexecutions of total_réexecutions

Travail faisant partie d'une séquence de réexécution et sa position dans la séquence

>> rerun step

Travail réexécuté à l'aide de la commande **rerun ;step**.

>> every run

Deuxième exécution et exécutions ultérieures d'un travail répétitif.

>>recovery

Exécution d'un travail de reprise.

State Etat du travail ou du flot de travaux. Les états des travaux sont les suivants :

ABEND

Le travail s'est terminé avec un code de sortie différent de zéro.

ABENP

Une confirmation **abend** a été reçue mais le travail n'est pas terminé.

ADD Le travail est en cours de soumission.

CANCL

Ne s'applique qu'aux dépendances interréseau. Le travail distant ou le flot de travaux a été annulé.

DONE

Le travail est terminé avec un état inconnu.

ERROR

Pour les dépendances interréseaux uniquement, une erreur s'est produite lors de la recherche du statut distant.

EXEC Le travail est en cours.

EXTRN

Pour les dépendances interréseaux uniquement, l'état est inconnu. Une erreur s'est produite, une action de réexécution vient d'être effectuée sur le travail du flot de travaux EXTERNE, ou le travail ou le flot de travaux distant n'existe pas.

FAIL Impossible de lancer le travail.

FENCE

La valeur de priorité du travail est inférieure à la priorité minimale.

HOLD

Le travail attend la résolution des dépendances.

INTRO

Le travail est introduit à des fins de lancement par le système.

PEND Le travail est terminé et attend une confirmation.

READY

Le travail est prêt à démarrer et toutes les dépendances sont résolues.

SCHED

L'heure **at** définie pour le travail n'a pas été atteinte.

SUCC Le travail s'est terminé avec un code de sortie nul.

SUCCP

Une confirmation **SUCC** a été reçue mais le travail n'est pas terminé.

SUPPR

Le travail est supprimé car les dépendances de condition associées à ses prédécesseurs ne sont pas satisfaites.

WAIT Le travail est à l'état WAIT (agent étendu).

Les états de flot de travaux sont les suivants :

ABEND

Le flot de travaux s'est terminé avec un code de sortie différent de zéro.

ADD Le flot de travaux a été ajouté par intervention de l'opérateur.

CANCEL

Le flot de travaux est en cours d'annulation. L'annulation est reportée jusqu'à ce que toutes les dépendances, y compris l'heure at, soient résolues.

ERROR

Pour les dépendances interréseaux uniquement, une erreur s'est produite lors de la recherche du statut distant.

EXEC Le flot de travaux est en cours d'exécution.

EXTRN

Ne s'applique qu'aux dépendances interréseau. Il s'agit de l'état du flot de travaux EXTERNAL contenant des travaux qui font référence à des travaux ou des flots de travaux dans le réseau distant.

HOLD

Le flot de travaux attend la résolution des dépendances.

READY

Le flot de travaux est prêt à démarrer et toutes les dépendances sont résolues.

STUCK

L'exécution du flot de travaux a été interrompue. Aucun travail n'est lancé sans l'intervention de l'opérateur.

SUCC Le flot de travaux s'est achevé correctement.

SUPPR

Le flot de travaux est supprimé car les dépendances de condition associées à ses prédécesseurs ne sont pas satisfaites.

Pr Priorité du travail ou du flot de travaux. Un signe plus (+) précédant la priorité indique que le travail a été lancé.

(Est)Start

Heure de début du travail ou du flot de travaux. Les parenthèses indiquent une estimation de l'heure de début. Si la commande est effectuée le jour où le travail est planifié pour s'exécuter, le paramètre **Start** affiche l'heure en tant que (Est)Start. Si la commande est effectuée un autre jour que le jour où le travail est planifié pour s'exécuter, le paramètre **Start** affiche la date en tant que (Est)Start. Par exemple, si vous avez le travail suivant dont l'heure de début a lieu le jour où le travail est planifié pour s'exécuter :

```
SCHEDULE MASTERB1#JS_B
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1700
```

```

:
MASTERB1#JOB1
  AT 1800
END

```

Vous obtenez la sortie suivante :

```

%sj @#@
                                     (Est) (Est)
CPU      Schedule SchedTime Job   State Pr Start  Elapse RetCode Deps
MASTERB1#JS_B  1700 08/18 ***** HOLD 10(17:00)
                                     JOB1 HOLD 10(18:00)

```

Par exemple, si vous avez le travail suivant dont l'heure de début a lieu un autre jour que celui auquel le travail est planifié pour s'exécuter :

```

SCHEDULE MASTERB1#JS_A
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 0400
:
MASTERB1#JOB_A
  AT 0500
END

```

Vous obtenez la sortie suivante :

```

%sj @#@
                                     (Est) (Est)
CPU      Schedule SchedTime Job   State Pr Start  Elapse RetCode Deps
MASTERB1#JS_A  0400 08/19 ***** HOLD 10(08/19)
                                     JOB_A HOLD 10(08/19)

```

(Est)Elapse

Heure d'exécution du travail ou du flot de travaux. Les parenthèses indiquent une estimation basée sur les statistiques consignées.

dépendances

Liste des commentaires et des dépendances sur les travaux. Une combinaison des éléments suivants peut être répertoriée :

- Pour une dépendance de prédécesseur/successeur, un nom de travail ou de flot de travaux s'affiche.

Si le travail ou le flot de travaux est un prédécesseur suspendu, son nom est suivi d'un [P].

En cas de dépendance orpheline, un [0] est affiché.

Pour les dépendances conditionnelles, le nom du travail ou du flot de travaux prédécesseur s'affiche suivi d'une ou plusieurs conditions de sortie au format IF <nom_condition> où *nom_condition* correspond au statut d'exécution du travail ou du flot de travaux prédécesseur, au statut du travail ou du flot de travaux prédécesseur ou à d'autres conditions basées sur la sortie ou le résultat du travail prédécesseur. Lorsque plusieurs conditions sont spécifiées, elles doivent être séparées d'une barre verticale (|). Voici l'affichage de la sortie de **showjob** dans la colonne Dépendances d'un travail prédécesseur JOBL1 sur lequel ont été définies plusieurs conditions de sortie. La condition satisfaite, quelle qu'elle soit, détermine le travail successeur qui s'exécute :

```

JOBL1 IF EXEC
      | STATUS_OK | STATUS_ERR12

```

Lorsque plusieurs conditions de sortie sont regroupées ou jointes, par exemple, lorsque 2 conditions sur 2 doivent être satisfaites pour qu'un travail successeur s'exécute, la sortie s'affiche comme suit :

```

JOIN MYJOIN      2 OF
JOBL1 IF EXEC
JOBL2 IF ABEND

```

Pour plus d'informations sur les prédécesseurs ou les dépendances orphelines en attente, reportez-vous à «Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux», à la page 71.

- Pour une dépendance **opens**, le nom de fichier s'affiche. Si le fichier réside sur un agent étendu et que son nom comporte plus de 25 caractères, seuls les 25 derniers caractères s'affichent.
- Pour une dépendance **needs**, un nom de ressource placé entre tirets (-) s'affiche. Si le nombre d'unités demandées est supérieur à un, le nombre s'affiche avant le premier tiret.
- Pour une heure d'échéance **deadline**, l'heure précédée du signe inférieur (<) s'affiche.
- Pour une fréquence **every**, la fréquence de répétition précédée par une perluète (&) s'affiche.
- Pour une échéance **until**, l'heure précédée du signe inférieur (<) s'affiche.
- Pour une durée **maximum duration** dépassée, [**MaxDurationExceeded**] s'affiche en plus du paramètre **maxdur=hh:mm**.
- Pour une durée **maximum duration** dépassée, et pour laquelle **onmaxdur action** est défini sur **Kill**, [**KillSubmitted**] s'affiche.
- Pour une durée **maximum duration** dépassée, et pour laquelle **onmaxdur action** est défini sur **Continue**, [**Continue**] s'affiche.
- Pour une durée **minimum duration** qui n'est pas atteinte et pour laquelle un travail se termine avec succès, [**MinDurationNotReached**] s'affiche en plus du paramètre **mindur=hh:mm**.
- Pour une durée **minimum duration** qui n'est pas atteinte, et pour laquelle **onmindur action** est défini sur **Continue**, [**Continue**] s'affiche.
- Pour une durée **minimum duration** qui n'est pas atteinte, et pour laquelle **onmindur action** est défini sur **Abend**, [**Abended**] s'affiche.
- Pour une durée **minimum duration** qui n'est pas atteinte, et pour laquelle **onmindur action** est défini sur **Confirm**, [**ConfirmSubmitted**] s'affiche.
- Pour une dépendance **prompt**, le numéro d'invite s'affiche au format **#num**. Pour les invites globales, le nom d'invite vient à la suite, entre parenthèses.
- Pour les travaux en cours d'exécution, le numéro d'identification de processus (PID) s'affiche au format **#Jnnnnnn**.
- Les travaux soumis sur UNIX via les commandes IBM Workload Scheduler **at** et **batch** portent la mention [**Userjcl**].
- Lors du signalement des dépendances horaires, la commande **showjobs** indique dans la colonne **Start** :
 - Uniquement l'heure **hh:mm** si le jour où les dépendances horaires sont définies correspond au jour où la commande **showjobs** est exécutée.
 - Uniquement la date **MM/JJ** si le jour où les dépendances horaires sont définies ne correspond pas au jour où la commande **showjobs** est exécutée.
- Les travaux annulés portent la mention [**Cancelled**].
- Les travaux annulés à l'aide de l'option **pend** portent la mention [**Cancel Pend**] (Annulation en cours).

- Les travaux pour lesquels une heure d'**expiration** est définie, y compris les travaux annulés avec l'option **pend** , portent la mention **[Until]**.
- **[Recovery]** signifie que l'intervention de l'opération est requise.
- **[Confirmed]** signifie qu'une confirmation est requise car le travail a été planifié à l'aide du mot clé **confirm**.
- **[Script]** concerne les réseaux complets uniquement, c'est-à-dire que le travail comporte un script centralisé et que IBM Workload Scheduler for z/OS ne l'a pas encore chargé sur l'agent.

/ La colonne **Dépendances** contient également le nom du poste de travail
 / réel sur lequel le flot de travaux s'est exécuté. Ce détail est disponible
 / uniquement si le travail a démarré et s'est exécuté sur un poste de travail
 / de pool. Ces informations peuvent être utiles par exemple lorsque vous
 / devez déterminer votre consommation de licences et devez donc savoir sur
 / quel poste de travail du pool le travail s'est réellement exécuté. Pour plus
 / d'informations sur la licence, voir section sur la gestion des licences dans
 / IBM License Metric Tool dans *Guide d'administration*.

Format Keys

Les noms des travaux sont répertoriés, un par ligne, au format suivant :

poste_de_travail#flot_travaux hhmm mm/jj.travail

Par exemple :

CPU	Schedule	SchedTime	Job	State	Pr	Start	Elapse	RetCode	Deps
MYCPU+#SCHED_F+	0600	03/04	*****	HOLD	55	(03/04)		[03/04/06]; #33	
	(M235062+#)		JOBMDM	HOLD	30	(03/04)		#1(PRMT3);-16	JOBSLOTS-
MYCPU+#SCHED_F+	1010	03/04	*****	HOLD	55	(03/04)		[03/04/06]; #34	
	(M235062+#)		JOBMDM	HOLD	30	(03/04)		#1(PRMT3);-16	JOBSLOTS-

Format info

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du travail dans le plan.

Travail

Nom du travail. Il se peut que la mention suivante précède un nom de travail :

>> **rerun as**

Travail réexécuté à l'aide de la commande **rerun** ou par suite d'une reprise automatique.

=
=
=

>> **rerun nombre_réexecutions of total_réexecutions**

Travail faisant partie d'une séquence de réexécution et sa position dans la séquence

>> **rerun step**

Travail réexécuté à l'aide de la commande **rerun ;step**.

>> **every run**

Deuxième exécution et exécutions ultérieures d'un travail répétitif.

>>**recovery**

Exécution d'un travail de reprise.

Job File

Nom du script ou du fichier exécutable du travail. Les noms de fichiers longs risquent de faire l'objet d'un renvoi à la ligne, ce qui donne lieu à un saut de page incorrect. Pour éviter cela, ajoutez le signe | avant **more**.

Opt Option de reprise de travaux, s'il y a lieu. Les options de reprise sont **RE** pour réexécution, **CO** pour poursuite et **ST** pour arrêt.

Travail

Nom du travail de reprise, s'il y a lieu.

Invite Numéro de l'invite de reprise, s'il y a lieu.

Par exemple :

```
conman "sj;info | more
```

génère un exemple de sortie similaire à ce qui suit :

```
-----Restart-----
UC      Calendrier      Heure calend      Travail      Fichier de travail Opt  Invite
M235062+#SCHED_22 1010 03/06
                JOBMDM /usr/acct/scripts/gll
                (B236153+#)JOB_FTA echo job12
M235062+#SCHED_22 0600 03/07
                JOBMDM /usr/acct/scripts/gll
                (B236153+#)JOB_FTA echo job12
M235062+#FINAL    2359 02/13
                STARAPPSERVER /opt/IBM/TWA/TWS/../../wastools/startWas.sh
                                Colorado
                MAKEPLAN /opt/IBM/TWA/TWS/MakePlan TWSRCMAP:(RC=0) OR (RC=4)
                SWITCHPLAN /opt/IBM/TWA/TWS/SwitchPlan
M235062+#FINALPOSTREPORTS 2359 02/13
                CHECKSYNC /opt/IBM/TWA/TWS/CheckSync
                CREATEPOSTREPORTS /opt/IBM/TWA/TWS/CreatePostReports
                                Colorado
                UPDATESTATS /opt/IBM/TWA/TWS/UpdateStats
                                Colorado
M235062+#SCHED12 1010 03/06
                JOBMDM /usr/acct/scripts/gll
                (B236153+#)JOB_FTA echo job12
```

L'exemple suivant affiche le statut de dépendance de définies sur le travail JOB_WAGES, dans le flot de travaux PAYROLL. Les conditions de sortie sont affichées avec "sc" pour identifier les conditions de succès et avec "oc" pour identifier les autres conditions de sortie. Une valeur est affectée à chaque paire nom et valeur de condition afin d'identifier si la condition a été satisfaite (true), non satisfaite (false) ou pas encore évaluée (N/A). :

```
%sj RENOIR#PAYROLL.WAGES;info
```

```
----- Restart -----
```

Workstation	Job Stream	SchedTime	Job	JobFile	Opt	Job	Prompt#
RENOIR	#PAYROLL	0000 08/10					
		JOB_WAGES	dir				
				sc: STATUS_OK	true	"RC=0"	
				oc: STATUS_ERR1	false	"RC=2"	
				oc: STATUS_ERR2	false	"RC=3"	

Format Step

Le format n'est pas pris en charge sous Windows.

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du travail dans le plan.

Travail

Nom du travail. Il se peut que la mention suivante précède un nom de travail :

>> rerun as

Travail réexécuté à l'aide de la commande **rerun** ou par suite d'une reprise automatique.

>> rerun nombre_réexecutions of total_réexecutions

Travail faisant partie d'une séquence de réexécution et sa position dans la séquence

>> repeated as

Deuxième exécution et exécutions ultérieures d'un travail **répétitif**.

Etat Etat du travail ou du flot de travaux. Voir "Format standard" pour obtenir des informations relatives à l'état.

Return code

Le code retour du travail.

Job# Numéro d'identification de processus affiché au format #Jnnnnnn.

Step Liste des processus descendants qui sont associés au travail. Pour les travaux d'agent étendu, seuls les processus hôte sont répertoriés.

Format Logon

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du travail dans le plan.

Travail

Nom du travail. Il se peut que la mention suivante précède un nom de travail :

>> rerun as

Travail réexécuté à l'aide de la commande **rerun** ou par suite d'une reprise automatique.

>> rerun nombre_réexecutions of total_réexecutions

Travail faisant partie d'une séquence de réexécution et sa position dans la séquence

>> repeated as

Deuxième exécution et exécutions ultérieures d'un travail **répétitif**.

Etat Etat du travail ou du flot de travaux. Voir "Format standard" pour obtenir des informations relatives à l'état.

Return code

Le code retour du travail.

Job# Numéro d'identification de processus affiché au format #Jnnnnnn.

Logon Nom d'utilisateur sous lequel le travail s'exécute.

Sur les systèmes d'exploitation Windows, les formats suivants sont disponibles :

user name

Où *nom_utilisateur* représente le nom de l'utilisateur Windows.

domaine\nom_utilisateur

Où *domaine* correspond au domaine Windows de l'utilisateur et *nom_utilisateur* correspond au nom de l'utilisateur Windows.

nom_utilisateur@domaine_internet

Où *nom_utilisateur@domaine_internet* représente le nom d'un utilisateur système dans un format d'adresse électronique. Le nom d'utilisateur est suivi du symbole @, lui-même suivi du nom du domaine Internet auquel l'utilisateur est associé.

Remarque : Insérez le caractère d'échappement '\' avant le caractère '@' dans la valeur *nom_utilisateur@domaine_internet* de la zone de connexion. Par exemple, si vous utilisez l'utilisateur *administrator@bvt.com* dans la zone de connexion, utilisez la syntaxe suivante :

..... ; logon=administrator\@bvt.com

Format Stdlist

Un fichier de liste standard est créé automatiquement par le processus **jobmon** sous Windows ou par le processus **jobman** sous UNIX chaque fois qu'un travail est lancé par **jobmon** ou **jobman**. Vous pouvez afficher le contenu des fichiers liste standard en utilisant le programme **conman**. Les fichiers de liste standard comprennent les informations suivantes :

- Bannières de début et fin de page
- Commandes echo
- Sortie stdout du travail
- Sortie stderr du travail

Pour préciser un format de date particulier à utiliser dans les fichiers liste standard, modifiez le format de date IBM Workload Scheduler avant de créer les fichiers. Il vous suffit de modifier le format de date local.

Selon l'environnement, modifiez le format de date local comme suit :

- Sous UNIX, définissez la variable LANG dans l'environnement au démarrage du processus **netman**. Si la variable LANG n'est pas définie, l'environnement local du système d'exploitation utilise "C" par défaut.
- Sous Windows, procédez comme suit :
 1. Cliquez sur Panneau de configuration → Options régionales et linguistiques, puis modifiez l'environnement local (emplacement).
 2. Cliquez avec le bouton droit de la souris sur **Poste de travail**, sélectionnez Propriétés, cliquez sur **Avancé**, et sur Variables d'environnement, puis définissez la variable LANG dans les variables système.
 3. Arrêtez et redémarrez le système.

Les fichiers de liste standard des travaux sélectionnés s'affichent.

Format Stdlist;keys

Les noms des fichiers de liste standard des travaux sélectionnés sont répertoriés, un par ligne.

Format crit

CPU Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du travail dans le plan.

Job Nom du travail. Il se peut que la mention suivante précède un nom de travail :

>> rerun as

Travail réexécuté à l'aide de la commande **rerun** ou par suite d'une reprise automatique.

>> rerun nombre_réexecutions of total_réexecutions

Travail faisant partie d'une séquence de réexécution et sa position dans la séquence

>> repeated as

Deuxième exécution et exécutions ultérieures d'un travail **répétitif**.

Etat Etat du travail ou du flot de travaux. Voir "Format standard" pour obtenir des informations relatives à l'état.

Pr Priorité du travail ou du flot de travaux. Un signe plus (+) précédant la priorité indique que le travail a été lancé.

(Est)Start

Heure de début du travail ou du flot de travaux. Les parenthèses indiquent une estimation de l'heure de début. Si l'heure de début dépasse 24 heures dans le passé ou dans l'avenir, la date est répertoriée à la place de l'heure.

(Est)Elapse

Heure d'exécution du travail ou du flot de travaux. Les parenthèses indiquent une estimation basée sur les statistiques consignées.

CP Indique si le travail est marqué comme critique (C) et/ou promu (P).

CritStart

Heure la plus tardive à laquelle peut démarrer un travail sans gêner les échéances des successeurs essentiels à la mission.

Par exemple, le résultat de la commande générique suivante :

```
%sj @##;crit
```

est :

CPU	Schedule	SchedTime	Job	State	Pr	Start	(Est) Elapse	(Est) CP	Crit Start
MYCPU_F+#JSA	1600	03/05	*****	HOLD	10				
			J0BA1	HOLD	10		CP 1759	03/05	
			J0BA2	HOLD	10		1758	03/05	
			J0BA3	HOLD	10		1757	03/05	
			J0BA4	HOLD	10		C 1659	03/05	

Il est à noter que :

- L'indicateur **C** s'applique uniquement aux travaux définis comme critiques dans leur définition de flot de travaux. Il est défini au niveau ou à l'heure **submit**.

- L'indicateur **P** s'applique aux travaux critiques et à leurs prédécesseurs (qui sont des travaux n'étant pas définis comme critiques, mais risquant néanmoins d'avoir un impact sur la réalisation opportune du travail critique d'un successeur). Il est défini au moment de l'exécution si le travail a été promu.
- Les travaux et prédécesseurs critiques font l'objet d'une heure de début critique. Le planificateur calcule l'heure de début critique d'un travail critique en soustrayant sa durée estimée de l'échéance. Il calcule l'heure de début critique d'un prédécesseur critique en soustrayant sa durée estimée de l'heure de début critique de son successeur direct. Au sein d'un réseau critique, le planificateur calcule l'heure de début critique du premier travail en premier lieu, puis fonctionne à rebours dans la chaîne de ses prédécesseurs. Ces calculs sont réitérés autant de fois que nécessaire, tant que le travail critique n'a pas été exécuté.

Format Deps

Les travaux utilisés dans les dépendances de prédécesseur/successeur sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** standard. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;keys

Les travaux et les flots de travaux qui comportent des dépendances de prédécesseur/successeur sont répertoriés, un par ligne.

Format Deps;info

Les travaux utilisés dans les dépendances de prédécesseur/successeur sont répertoriés, suivis des travaux et des flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;info**. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;logon

Les travaux utilisés dans les dépendances de prédécesseur/successeur sont répertoriés, suivis des travaux et des flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;logon**. Les flots de travaux sont répertoriés au format **showschedules** standard.

showprompts

Affiche les informations relatives aux invites.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande `conman start`.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
{showprompts |
sp} [invite_sélectionnée]
    [;keys]
    [;offline]
```

```
{showprompts  
| sp} [invite_sélectionnée]  
[;deps[;keys | info | logon]][:offline]
```

Arguments

invite_sélectionnée

[*nom_invite* | [*poste_de_travail*#]*num_message*][:état[:...]]

nom_invite

Indique le nom d'une invite globale. Les caractères génériques sont acceptés.

poste de travail

Indique le nom du poste de travail sur lequel une invite non nommée est émise. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé.

msgnum

Indique le numéro de message d'une invite non nommée.

état

Indique l'état des invites à afficher. Les états sont les suivants :

OUI L'invite a généré l'envoi de la réponse y.

NON L'invite a généré l'envoi de la réponse n.

DEMANDEE

L'invite a été émise mais aucune réponse n'a été générée.

INACTIVE

L'invite n'a pas été émise.

keys Affiche la liste composée d'une seule colonne des objets sélectionnés par la commande.

deps Affiche les informations au format **deps**. Utilisez **keys**, **info** ou **logon** pour modifier l'affichage.

info Affiche les informations au format **info**.

logon Affiche les informations au format **logon**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Remarque : Les numéros d'invite affectés aux invites globales et locales changent lorsque le plan de production est étendu.

Résultats

La sortie de la commande est générée dans trois formats : **standard**, **keys** et **deps**. Les arguments **keys**, **info** et **logon** modifient l'affichage **deps**.

Exemples

Pour afficher le statut de toutes les invites générées sur le poste de travail où vous exécutez le programme **conman**, exécutez la commande suivante :

```
showprompts
```

Voici un exemple de sortie :

```

Message d'état ou invite
ASKED 1(PRMT3) !Voulez-vous continuer ?
INACT 3(CPUA#SCHED_12[(0600 03/12/06),(0AAAAAAAAAAAAABST)]) Etes-vous prêt à traiter job1?
INACT 5(CPUA#SCHED_12[(1010 03/12/06),(0AAAAAAAAAAAAABSU)]) Etes-vous prêt à traiter job2?
INACT 7(CPUA#SCHED_22[(0600 03/12/06),(0AAAAAAAAAAAAABTR)]) Etes-vous prêt à traiter job3?

```

Pour afficher le statut de toutes les invites mis générées au format **deps**, exécutez la commande suivante :

```
sp mis@;asked;deps
```

Pour afficher le statut de l'invite 7 sur le poste de travail CPUA, exécutez la commande suivante :

```
sp CPUA#7
```

La commande génère la sortie suivante :

```
INACT 7(CPUA#SCHED_22[(0600 03/12/06),(0AAAAAAAAAAAAABTR)]) Etes-vous prêt à traiter job3?
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche comme suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Invite**.
4. Dans la liste déroulante **Requête**, sélectionnez **Toutes les invites du plan**, qui répertorie toutes les invites, quel que soit leur statut, ou créez et sélectionnez une autre tâche.

Format standard

State Etat de l'invite.

Message ou invite

Pour les invites nommées, numéro de message, nom et texte de l'invite.

Pour les invites non nommées, numéro de message, nom du travail ou du flot de travaux et texte de l'invite.

Format Keys

Les invites sont répertoriées, une par ligne. Les invites nommées sont répertoriées avec les noms et numéros de message qui leur sont associés. Les invites non nommées sont répertoriées avec les numéros de messages qui leur sont associés et avec les noms des travaux ou flots de travaux dans lesquels elles apparaissent sous forme de dépendances.

Format Deps

Les invites utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** standard. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;keys

Les travaux et les flots de travaux qui comportent des dépendances prompt sont répertoriés à raison d'un par ligne.

Format Deps;info

Les invites utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;info**. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;logon

Les invites utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;logon**. Les flots de travaux sont répertoriés au format **showschedules** standard.

showresources

Affiche les informations relatives aux ressources.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande `conman start`.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
{showresources | sr} [[poste de  
travail#]nom_ressource]  
    [;keys]  
    [;offline]
```

```
{showresources | sr} [[poste de  
travail#]nom_ressource]  
    [;deps[;keys | info | logon]]  
    [;offline]
```

Arguments

poste de travail

Indique le nom du poste de travail sur lequel la ressource est définie. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé.

nom_ressource

Indique le nom de la ressource. Les caractères génériques sont acceptés.

keys Affiche la liste composée d'une seule colonne des objets sélectionnés par la commande.

deps Affiche les informations au format **deps**. Utilisez **keys**, **info** ou **logon** pour modifier l'affichage.

info Affiche les informations au format **info**.

logon Affiche les informations au format **logon**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**.
Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

Résultats

La sortie de la commande est générée dans trois formats : **standard**, **keys** et **deps**.
Les arguments **keys**, **info** et **logon** modifient l'affichage **deps**.

Exemples

Pour afficher les informations relatives à toutes les ressources sur le poste de travail où vous exécutez le programme **conman**, exécutez la commande suivante :
`showresources`

Voici un exemple de sortie :

```
Num UC Ressource    Total disponible Qté Util par
CPUA #JOBSLOTS 16    16    Pas de réservation pour cette ressource
```

Pour afficher les informations relatives à la ressource `jobslots` sur le poste de travail CPUA au format **deps**, exécutez la commande suivante :
`sr CPUA#JOBSLOTS;deps`

Voici un exemple de sortie :

```

                                (Est) (Est)
Workstation Job Stream SchedTime Job State Pr Start Elapse RetCode Dependencies
CPUA    #JOBSLOTS Les dépendances sont les suivantes :

FTAA    #SCHED_F+ 0600 03/04 ***** HOLD 55(03/04)    [03/04/06];#33
              (CPUA#)JOBMDM HOLD 30(03/04)              #1(PRMT3);-16 JOBSLOTS-
FTAA    #SCHED_F+ 1010 03/04 ***** HOLD 55(03/04)    [03/04/06];#34
              (CPUA#)JOBMDM HOLD 30(03/04)              #1(PRMT3);-16 JOBSLOTS-
FTAA    #SCHED_F+ 0600 03/05 ***** HOLD 55(03/05)    [03/04/06];#35
              (CPUA#)JOBMDM HOLD 30(03/05)              #1(PRMT3);-16 JOBSLOTS-
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Sur la page d'accueil, sélectionnez **Surveiller votre charge de travail**, ou dans la barre de navigation située en haut de la page, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Ressource**.
4. Dans la liste déroulante **Requête**, sélectionnez **Toutes les ressources du plan** ou une autre tâche pour surveiller les ressources.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

UC Poste de travail sur lequel la ressource est définie.

Resource

Nom de la ressource.

Total Nombre total d'unités de ressources définies.

Available

Nombre d'unités de ressources non allouées.

Qty Nombre d'unités de ressources allouées à un travail ou à un flot de travaux.

Used By

Nom du travail ou du flot de travaux.

Format Keys

Les ressources sont répertoriées à raison d'une par ligne.

Format Deps

Les ressources utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** standard. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;keys

Les travaux et les flots de travaux qui comportent des dépendances resource sont répertoriés à raison d'un par ligne.

Format Deps;info

Les ressources utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;info**. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;logon

Les ressources utilisées sous forme de dépendances sont répertoriées, suivies des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;logon**. Les flots de travaux sont répertoriés au format **showschedules** standard.

showschedules

Affiche les informations relatives aux flots de travaux.

L'information affichée est mise à jour uniquement tant que IBM Workload Scheduler (batchman) s'exécute. Le fait que batchman soit actif ou non est vérifié sur l'écran par le message Batchman LIVES ou Batchman down lorsque vous émettez la commande conman start.

Vous devez avoir un accès *list* à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.

Syntaxe

```
(showscheds | ss) [flot_travaux_sélectionné]
    [:keys]
    [:offline]
    [:showid]
```

```
{showscheds | ss} [flot_travaux_sélectionné]
    [;deps[;keys | info | logon]]
    [;offline]
    [;showid]
```

Arguments

flot_travaux_sélectionné

Voir «Sélection de flots de travaux dans les commandes», à la page 428.

keys Affiche la liste composée d'une seule colonne des objets sélectionnés par la commande.

deps Affiche des informations au format **deps**. En d'autres termes, les flots de travaux utilisés dans les dépendances de *prédécesseur/successeur* sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** de base. Les flots de travaux sont répertoriés au format **showschedules** de base. Utilisez "keys", "info" ou "logon" pour modifier l'affichage "deps".

info Affiche les informations au format **info**.

logon Affiche les informations au format **logon**.

offline

Envoie la sortie de la commande au périphérique de sortie de **conman**. Pour plus d'informations sur ce périphérique, voir «Sortie offline», à la page 412.

showid

Affiche l'identificateur de chaque flot de travaux.

Résultats

La sortie de la commande est générée dans trois formats : standard, **keys** et **deps**. Les arguments **keys**, **info** et **logon** modifient l'affichage **deps**. La liste affichée dans la sortie de la commande n'inclut pas les travaux qui ont été réexécutés lors des processus de planification précédents mais le total final les inclut.

Exemples

Pour afficher le statut du flot de travaux CLEM_DOCOM sur le poste de travail site3 et demander l'affichage de l'identificateur du flot de travaux, exécutez la commande suivante :

```
%ss @#JS_DOCOM ;showid
```

Voici un exemple de résultat de cette commande :

```
(Est) (Est) Jobs Sch
Workstation Job Stream SchedTime State Pr Start Elapse # OK Lim
site3 #JS_DOCOM 0600 11/26 SUCC 10 11/26 00:01 1 1 {0AAAAAAAAAAACRZ}
```

Pour afficher le statut de tous les flots de travaux à l'état HOLD sur le poste de travail où vous exécutez le programme **conman**, exécutez la commande suivante :

```
showschedules @+state=hold
```

Voici un exemple de sortie de cette commande :

```
(Est) (Est) Jobs Sch
Workstation Job Stream SchedTime State Pr Start Elapse # OK Lim
site3 #FILE_JS1 0600 11/26 HOLD 10 (11/26) 1 0 parms FILE_JS1`
```

Pour afficher le statut de tous les flots de travaux dont le nom commence par sched sur le poste de travail CPUA au format **deps;info**, exécutez la commande suivante :

```
ss CPUA#sched@;deps;info
```

Voici un exemple de sortie :

```
-----Restart-----
CPU  Schedule SchedTime Job  JobFile      Opt  Job    Prompt
CPUA #JS_FIRST1[(0600 03/10/12),(0AAAAAAAAAAAAABVY)] Dependencies are:
CPUA#MOD 0212 03/10
JOBMDM /usr/scripts/gl1(B236153+#)JOB_FTA1 echo Start gl1?
CPUA#MOD 0251 03/10
JOBMDM /usr/scripts/gl2(B236153+#)JOB_FTA2 echo Start gl2?
```

Pour afficher **hors ligne** le statut de tous les flots de travaux à l'état ABEND sur tous les postes de travail, exécutez la commande suivante :

```
ss @#@+state=abend;off
```

Pour afficher le statut de tous les flots de travaux sur tous les postes de travail, exécutez la commande suivante :

```
%ss @#@
```

Voici un exemple de sortie de cette commande. La sortie inclut un flot de travaux avec deux dépendances conditionnelles. Le flot de travaux JS_CALC a une dépendance conditionnelle externe à un travail prédécesseur, JOB12P, dans le flot de travaux JS_REPORT. Si le travail JOB12P satisfait la condition représentée par ON STATUS_OK, alors le travail successeur s'exécute. Le flot de travaux JS_CALC a une dépendance conditionnelle externe à un flot de travaux, JS_TAX, sur lequel sont définies deux conditions de sortie : IF ABEND | SUPPR.

Workstation	Job Stream	SchedTime	State	Pr	Start	Elapse	#	OK	Lim	
site3	#JS_DOCOM	0600 09/20	SUCC	10	09/20	00:01	1	1		
site3	#JS_SCRIPT	0600 09/20	SUCC	10	09/20	00:03	1	1		
site2	#JS_PRED1	1000 09/20	SUCC	10	09/20	00:01	1	1		
site3	#JS_SCRIPT1	0600 09/20	ABEND	10	09/20	00:01	1	0		
site3	#LFLEJOB	0600 09/20	READY	10			1	0		
site1	#RES_100	0600 09/20	SUCC	10	09/20	00:09	1	1		
site3	#FILE_JS1	0600 09/20	HOLD	10	(09/20)		1	0		parms FILE_JS1~
site3	#FILE_JOB	0600 09/20	SUCC	10	09/20	00:01	1	1		
site3	#JS_CALC	0000 09/20	HOLD	10	(09/20)		1	0		JS_REPORT (0000 09/20/15).JOB12P IF STATUS_OK
JS_TAX (0000 09/20/15).@ IF ABEND SUPPR										

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Flot de travaux**.
4. Dans la liste déroulante **Requête**, sélectionnez **Tous les flots de travaux du plan** ou une autre tâche pour surveiller les flots de travaux.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.

Format standard

UC Poste de travail sur lequel le flot de travaux s'exécute.

Schedule

Nom du flot de travaux.

SchedTime

Date et heure prévues de l'exécution du flot de travaux dans le plan.

State Etat du flot de travaux. Les états sont les suivants :

ADD Le flot de travaux a été ajouté par intervention de l'opérateur.

ABEND

Le flot de travaux s'est terminé avec un code de sortie différent de zéro.

CANCEL

Le flot de travaux est en cours d'annulation. L'annulation est reportée jusqu'à ce que toutes les dépendances, y compris l'heure at, soient résolues.

ERROR

Pour les dépendances interréseaux uniquement, une erreur s'est produite lors de la recherche du statut distant.

EXEC Le flot de travaux est en cours d'exécution.

EXTRN

Ne s'applique qu'aux dépendances interréseau. Il s'agit de l'état du flot de travaux EXTERNAL contenant des travaux qui font référence à des travaux ou des flots de travaux dans le réseau distant.

HOLD

Le flot de travaux attend la résolution des dépendances.

READY

Le flot de travaux est prêt à démarrer et toutes les dépendances sont résolues.

STUCK

L'exécution du flot de travaux a été interrompue. Aucun travail n'est lancé sans l'intervention de l'opérateur.

SUCC Le flot de travaux s'est achevé correctement.

SUPPR

Le flot de travaux est supprimé car les dépendances de condition associées à ses prédécesseurs ne sont pas satisfaites.

Pr Priorité du flot de travaux.

(Est)Start

Heure de début du travail ou du flot de travaux. Les parenthèses indiquent une estimation de l'heure de début. Si la commande est effectuée le jour où le flot de travaux est planifié pour s'exécuter, le paramètre **Start** affiche l'heure en tant que (Est)Start. Si la commande est effectuée un autre jour que le jour où le travail est planifié pour s'exécuter, le paramètre Start affiche la date en tant que (Est)Start. Par exemple, si vous avez le flot de travaux suivant dont l'heure de début a lieu le jour où le travail est planifié pour s'exécuter :

```
SCHEDULE MASTERB1#JS_B
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1800
:
MASTERB1#JOB1
END
```

Vous obtenez la sortie suivante :

```
%ss @#@
                                (Est) (Est) Jobs Sch
CPU   Schedule SchedTime State Pr Start Elapse # OK Lim
MASTERB1#JS_B   1800 08/18 HOLD 10(18:00)    1  0
```

Par exemple, si vous avez le flot de travaux suivant dont l'heure de début a lieu un autre jour que celui auquel le travail est planifié pour s'exécuter :

```
SCHEDULE MASTERB1#JS_A
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 0500
:
MASTERB1#JOB1
END
```

Vous obtenez la sortie suivante :

```
%ss @#@
                                (Est) (Est) Jobs Sch
CPU   Schedule SchedTime State Pr Start Elapse # OK Lim
MASTERB1#JS_A   0500 08/19 HOLD 10(08/19)    1  0
```

(Est)Elapse

Heure d'exécution du flot de travaux. Les parenthèses indiquent une estimation basée sur les statistiques consignées.

Jobs # Nombre de travaux du flot de travaux.

Jobs OK

Nombre de travaux dont l'exécution a abouti.

Sch Lim

Nombre maximal de travaux dans un flot de travaux. Si aucune valeur n'est indiquée, aucune limite ne s'applique.

dépendances

Liste des commentaires et des dépendances sur les flots de travaux. Une combinaison des éléments suivants peut être répertoriée :

- Pour une dépendance de prédécesseur/successeur, un nom de travail ou de flot de travaux s'affiche. Si le travail ou le flot de travaux est un prédécesseur suspendu, son nom est suivi d'un [P].
- Pour les dépendances conditionnelles, le nom du travail ou du flot de travaux prédécesseur s'affiche suivi d'une ou plusieurs conditions de sortie au format IF <nom_condition> ... où *nom_condition* correspond au statut d'exécution du flot de travaux prédécesseur, au statut du flot de travaux du travail ou à d'autres conditions basées sur la sortie ou le résultat du flot de travaux prédécesseur. Lorsque plusieurs conditions sont spécifiées, elles doivent être séparées d'une barre verticale (|). Voici l'affichage de la sortie de **showschedules** dans la colonne Dépendances d'un flot de travaux prédécesseur, JOBSTREAML1, sur lequel sont définies les conditions de statut ABEND et FAIL. L'aboutissement de JOBSTREAML1 avec le statut ABEND ou FAIL détermine le travail successeur qui s'exécute :

```
JOBSTREAML1(0000 09/15/15).JOB1 IF ABEND | FAIL
```

- Pour une dépendance **opens**, le nom de fichier s'affiche. Si le fichier réside sur un agent étendu et que son nom comporte plus de 25 caractères, seuls les 25 derniers caractères s'affichent.
- Pour une dépendance **needs**, un nom de ressource placé entre tirets (-) s'affiche. Si le nombre d'unités demandées est supérieur à un, le nombre s'affiche avant le premier tiret.
- Pour une échéance définie par l'argument **until**, l'heure précédée du signe inférieur (<) s'affiche.
- Pour une dépendance **prompt**, le numéro d'invite s'affiche au format **#num**. Pour les invites globales, le nom d'invite vient à la suite, entre parenthèses.
- Les flots de travaux annulés portent la mention **[Cancelled]**.
- Les flots de travaux annulés à l'aide de l'option **pend** portent la mention **[Cancel Pend]** (Annulation en cours).
- Pour une heure d'échéance **deadline**, l'heure précédée du signe inférieur (<) s'affiche.
- Les flots de travaux contenant le mot clé **carryforward** portent la mention **[Carry]**.
- En ce qui concerne les flots de travaux du plan de production précédent qui ont été reportés, le nom et la date d'origine s'affichent entre crochets.
- Lors du signalement des dépendances horaires, la commande **showschedules** indique dans la colonne **Start** :
 - Uniquement l'heure **hh:mm** si le jour où les dépendances horaires sont définies correspond au jour où la commande **showschedules** est exécutée.
 - Uniquement la date **mm/jj** si le jour où les dépendances horaires sont définies ne correspond pas au jour où la commande **showschedules** est exécutée.

Remarque : La date ou l'heure affichée dans la colonne **Démarrage** est convertie en fonction du fuseau horaire défini sur le poste de travail sur lequel le flot de travaux doit s'exécuter.

Format Keys

Les flots de travaux sont répertoriés, un par ligne.

Format Deps

Les flots de travaux utilisés comme dépendances sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs** standard. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;keys

Les flots de travaux qui comportent des dépendances de prédécesseur/successeur sont répertoriés un par ligne.

Format Deps;info

Les flots de travaux utilisés sous forme de dépendances sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;info**. Les flots de travaux sont répertoriés au format **showschedules** standard.

Format Deps;logon

Les flots de travaux utilisés sous forme de dépendances sont répertoriés, suivis des travaux et flots de travaux dépendants. Les travaux sont répertoriés au format **showjobs;logon**. Les flots de travaux sont répertoriés au format **showschedules** standard.

shutdown

Arrête sans condition tous les processus de production et services IBM Workload Scheduler, y compris **batchman**, **jobman**, **netman**, **mailman**, **appservman**, tous les serveurs **mailman** et tous les processus **writer**.

Même si cette commande arrête le service **appservman**, elle n'arrête pas les services WebSphere Application Server. Pour arrêter les services WebSphere Application Server, exécutez la commande **stopappserver**. Pour plus d'informations, voir «stopappserver», à la page 541.

Sur les postes de travail Windows, la commande **shutdown** n'arrête pas le service **tokensrv**.

Remarque : Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

Vous devez avoir un accès **shutdown** au poste de travail.

Syntaxe

{shutdown | shut} [;wait]

Arguments

wait Attend l'arrêt de tous les processus avant d'inviter l'utilisateur à lancer une autre commande.

Commentaires

La commande **shutdown** arrête les processus uniquement sur le poste de travail sur lequel **conman** est en cours d'exécution. Pour réexécuter **netman** uniquement, exécutez la commande **StartUp**. Pour plus d'informations sur la commande **StartUp**, voir «StartUp», à la page 722. Pour réexécuter l'intégralité de l'arborescence de processus, exécutez les commandes **conman**.

```
start
startappserver
startmon
```

Avant d'exécuter une commande **shutdown**, vous devez exécuter la commande **conman unlink @**.

Exemples

Pour arrêter la production sur le poste de travail où vous exécutez le programme **conman**, exécutez la commande suivante :

```
unlink @
shutdown
```

Pour arrêter la production sur le poste de travail où vous exécutez le programme **conman** et attendre la fin de tous les processus, exécutez la commande suivante :

```
unlink@noask  
shut ;wait
```

start

Démarre les processus de production IBM Workload Scheduler, sauf pour le moteur de surveillance d'événements et WebSphere Application Server (voir «startappserver», à la page 534 et «startmon», à la page 536 pour connaître les commandes qui démarrent ces processus).

Remarque : Assurez-vous que la commande **conman start** n'est pas émise pendant l'exécution de **JnextPlan** ou de **stageman**.

Vous devez avoir un accès *start* au poste de travail.

Syntaxe

```
start [domaine!]poste de travail  
      [;mgr]  
      [;noask]  
      [;demgr]
```

Arguments

domaine

Indique le nom du domaine dans lequel des postes de travail sont lancés. Les caractères génériques sont acceptés.

Cet argument est utile lors du lancement de plusieurs postes de travail dans un domaine. Par exemple, pour lancer tous les agents du domaine **stlouis**, utilisez la commande suivante :

```
start stlouis!@
```

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste de travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail à démarrer. Les caractères génériques sont acceptés.

Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

mgr Ce paramètre ne peut être indiqué que sur le poste de travail où **conman** est en cours d'exécution. Il démarre le poste de travail local en tant que gestionnaire de domaine. Le poste de travail est le nouveau gestionnaire de domaine et le gestionnaire de domaine en cours devient un agent tolérant aux pannes. Ce format de commande suit généralement une commande **stop**.

Remarque : Pour changer de gestionnaire de domaine, il est préférable d'utiliser une commande **switchmgr**. Pour plus d'informations, voir «switchmgr», à la page 562.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

demgr Cette option interdit toute ouverture des connexions externes durant la phase de transition entre le moment où un agent démarre en tant qu'ancien gestionnaire de domaine et le moment où la commande **switchmgr** est

exécutée, privant ainsi l'agent de la fonction de gestionnaire de domaine. Elle est exécutée automatiquement. Tant que l'ancien gestionnaire de domaine n'a pas traité l'événement **switchmgr** (dans le cas, par exemple, d'un redémarrage retardé ou suivant la réparation d'un agent endommagé), l'option *demgr* **doit** être utilisée pour lancer l'ancien gestionnaire de domaine depuis la ligne de commande locale. Pour plus d'informations sur cette option, voir *IBM Workload Scheduler - Guide d'administration*.

Commentaires

La commande **start** est utilisée au début chaque période de production pour réexécuter IBM Workload Scheduler après le traitement de préproduction. Elle initialise et lance automatiquement les agents tolérants aux pannes et les agents standard, qui sont connectés automatiquement. Les agents qui ne sont pas connectés automatiquement sont initialisés et exécutés via la commande **link**.

En supposant qu'un utilisateur possède un accès **start** aux postes de travail en cours de démarrage, les règles suivantes s'appliquent :

- Un utilisateur exécutant **conman** sur le gestionnaire de domaine maître peut démarrer n'importe quel poste de travail du réseau.
- Un utilisateur exécutant **conman** sur un autre gestionnaire que le gestionnaire de domaine maître peut démarrer n'importe quel poste de travail appartenant à ce domaine ou à des domaines subordonnés. L'utilisateur ne peut pas démarrer des postes de travail appartenant à des domaines homologues.
- Un utilisateur exécutant **conman** sur un agent peut démarrer tout poste de travail hébergé par cet agent.

Exemples

La figure 26 et le tableau 68, à la page 534 qui suivent présentent les postes de travail démarrés par les commandes **start** exécutées par les utilisateurs en différents points du réseau.

DM_n sont des gestionnaires de domaine et **A_{nn}** sont des agents.

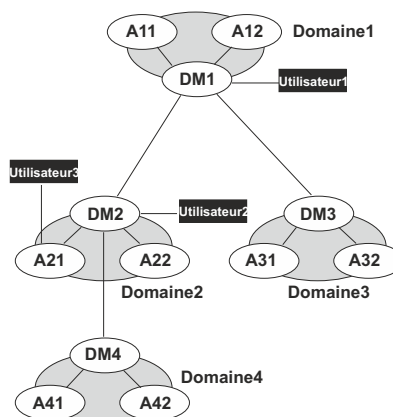


Figure 26. Exemple de réseau

Tableau 68. Postes de travail démarrés

Commande	Démarrage par Utilisateur1	Démarrage par Utilisateur2	Démarrage par Utilisateur3
start @!@	Tous les postes de travail sont démarrés	DM2 A21 A22 DM4 A41 A42	A21
start @	DM1 A11 A12	DM2 A21 A22	A21
start DOMAIN3!@	DM3 A31 A32	Non autorisé	Non autorisé
start DOMAIN4!@	DM4 A41 A42	DM4 A41 A42	Non autorisé
start DM2	DM2	DM2	Non autorisé
start A42	A42	A42	Non autorisé
start A31	A31	Non autorisé	Non autorisé

startappserver

Démarre WebSphere Application Server sur le poste de travail.

Syntaxe

startappserver [*domaine!*]*poste de travail*
[*;wait*]

Arguments

domaine

Indique le nom du domaine du poste de travail. Indique le nom du domaine du poste de travail. Parce que les postes de travail ont des noms uniques, le domaine n'est pas nécessaire lors du démarrage du serveur WebSphere Application Server sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

Poste de travail

Indique le nom du poste de travail où vous souhaitez démarrer le moteur de surveillance. Les caractères génériques sont acceptés. Si aucun domaine et poste de travail ne sont spécifiés, l'action se fait sur le poste de travail local.

wait Attendez jusqu'au démarrage du serveur WebSphere Application Server avant d'émettre une autre commande.

Commentaires

Le droit de démarrer des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

WebSphere Application Server peut également être démarré avec l'utilitaire StartUp.

startbrokerapp

Démarre l'application Dynamic Workload Broker.

Syntaxe

startbrokerapp [*domaine!*]*poste_travail*[:*wait*]

Arguments

domaine

Indique le nom du domaine du poste de travail. Dans la mesure où les postes de travail portent des noms uniques, le domaine n'est pas nécessaire lors du démarrage de Dynamic Workload Broker sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail sur lequel vous souhaitez démarrer Dynamic Workload Broker. Les caractères génériques sont acceptés. Si aucun domaine et poste de travail ne sont spécifiés, l'action se fait sur le poste de travail local.

wait

Indique qu'aucune autre commande ne doit être acceptée tant que Dynamic Workload Broker ne s'est pas arrêté.

Commentaires

Le droit de démarrer des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Vous pouvez démarrer Dynamic Workload Broker à l'aide de wastool startBrokerApplication.

starteventprocessor

Démarre le serveur de traitement des événements sur le gestionnaire de domaine maître, le maître de sauvegarde ou sur un poste de travail installé en tant que maître de sauvegarde, qui fonctionne comme un agent tolérant aux pannes normal.

Syntaxe

{starteventprocessor | startevtp} [*domaine!*]*poste de travail*

Arguments

domaine

Indique le nom du domaine du poste de travail.

poste de travail

Indique le nom du poste de travail où vous souhaitez démarrer le serveur de traitement d'événement. Les caractères génériques ne sont pas admis.

Commentaires

Vous pouvez omettre le nom du poste de travail si vous exécutez la commande localement.

Le droit de démarrer des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions > Démarrer le processeur d'événement**.

startmon

Démarré le processus monman qui active le moteur de surveillance d'événement sur le poste de travail.

Syntaxe

```
{startmon | startm} [domaine!]poste de travail  
[;noask]
```

Arguments

domaine

Indique le nom du domaine du poste de travail. Dans la mesure où les postes de travail portent des noms uniques, le domaine n'est pas nécessaire lors du démarrage du moteur de surveillance sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail où vous souhaitez démarrer le moteur de surveillance. Les caractères génériques sont acceptés.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

Le droit de démarrer des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions** > **Démarrer le moniteur d'événements**.

status

Affiche la bannière **conman** ainsi que le statut de production de IBM Workload Scheduler.

Syntaxe

```
{status | stat}
```

Résultats

Le mode du plan de production (fichier Symphony) est indiqué entre parenthèses, à la suite du mot **schedule** sur la deuxième ligne de la sortie. Le message d'information **Def** ou **Exp** peut apparaître. **Def** indique que le plan de production est en mode non étendu, et **Exp** indique qu'il est en mode étendu. Le mode du plan de production est déterminé par la valeur de l'option globale *version étendue*. Avec IBM Workload Scheduler version 8.2, les bases de données et les plans sont toujours développés, mais cette information apparaît pour une compatibilité avec les versions antérieures.

Exemples

L'exemple suivant affiche l'état du plan de production en cours.

```
%status
#####
# Eléments sous licence - Propriété d'IBM* et HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#####.
Flot de travaux (Exp) 11/26/16 (#34) sur site3.
Batchman ACTIF. Limit:19, Fence:0, Audit Level:0
```

stop

Arrête les processus de production de IBM Workload Scheduler. Pour arrêter le processus **netman**, utilisez la commande **shutdown**. Vous devez avoir un accès **stop** au poste de travail.

Syntaxe

```
stop [domaine!]poste de travail  
      [;wait]  
      [;noask]
```

Arguments

domaine

Indique le nom du domaine dans lequel des postes de travail sont arrêtés. Dans la mesure où les postes de travail portent des noms uniques, le domaine n'est pas requis lors de l'arrêt d'un poste de travail spécifique. Les caractères génériques sont acceptés.

Cet argument est utile lors de l'arrêt de plusieurs postes de travail dans un domaine. Par exemple, pour arrêter tous les agents du domaine **stlouis**, utilisez la commande suivante :

```
stop stlouis!@
```

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail à arrêter. Les caractères génériques sont acceptés.

Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

wait Indique qu'aucune autre commande ne doit être acceptée tant que tous les processus ne sont pas arrêtés.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

Si la commande **stop** ne peut pas être appliquée à un poste de travail distant (si le chemin TCP/IP n'est pas disponible, par exemple), la commande est stockée en local dans un fichier pobox et envoyée au poste de travail une fois celui-ci est connecté.

En supposant qu'un utilisateur possède un accès **stop** aux postes de travail en cours d'arrêt, les règles suivantes s'appliquent :

- Un utilisateur exécutant **conman** sur le gestionnaire de domaine maître peut arrêter n'importe quel poste de travail du réseau.
- Un utilisateur exécutant **conman** sur un autre gestionnaire que le gestionnaire de domaine maître peut arrêter n'importe quel poste de travail appartenant à ce domaine ou à des domaines subordonnés. L'utilisateur ne peut pas arrêter des postes de travail appartenant à des domaines homologues.
- Un utilisateur exécutant **conman** sur un agent peut arrêter n'importe quel poste de travail du domaine local.

Lorsque vous lancez une commande **stop @** sur un gestionnaire de domaine, une commande **conman stop** s'exécute en local sur les postes de travail distants. La commande commence son exécution sur les postes de travail situés en bas de la hiérarchie du réseau, pour terminer sur le gestionnaire de domaine. Cependant, le fichier Symphony n'est pas mis à jour tant que les postes de travail fonctionnent. C'est pourquoi, si vous lancez une commande **conman sc@!@** depuis un poste de travail, il se peut que les informations résultantes fournissent une image mise à jour des différents états des postes de travail, y compris du gestionnaire de domaine.

Exemples

La figure 27 et le tableau 69 qui suivent présentent les postes de travail arrêtés par les commandes **stop** exécutées par les utilisateurs en différents points du réseau.

DM_n sont des gestionnaires de domaine et **A_{mn}** sont des agents.

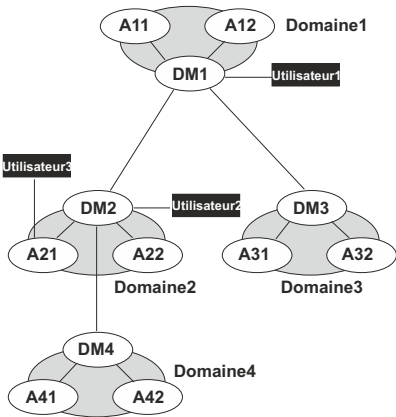


Figure 27. Exemple de réseau

Tableau 69. Postes de travail arrêtés

Commande	Arrêt par Utilisateur1	Arrêt par Utilisateur2	Arrêt par Utilisateur3
stop @!@	Tous les postes de travail sont arrêtés	DM2 A21 A22 DM4 A41 A42	DM2 A21 A22
stop @	DM1 A11 A12	DM2 A21 A22	DM2 A21 A22
stop DOMAIN3!@	DM3 A31 A32	Non autorisé	Non autorisé

Tableau 69. Postes de travail arrêtés (suite)

Commande	Arrêt par Utilisateur1	Arrêt par Utilisateur2	Arrêt par Utilisateur3
stop DOMAIN4!@	DM4 A41 A42	DM4 A41 A42	Non autorisé
stop DM2	DM2	DM2	DM2
stop A42	A42	A42	Non autorisé
stop A31	A31	Non autorisé	Non autorisé

stop ;progressive

Arrête les processus de production IBM Workload Scheduler de façon hiérarchique lorsque vous avez défini au moins un poste de travail sur BEHINDFIREWALL dans un réseau IBM Workload Scheduler. Similaire à la commande stop @!@, cette commande est plus efficace pour améliorer les performances du plan. La commande ne s'exécute pas à partir du domaine dans lequel elle a été initialement émise pour chaque domaine subordonné, elle s'exécute à chaque niveau de la hiérarchie.

Remarque : Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

Vous devez avoir un accès *stop* au poste de travail.

Syntaxe

stop ;progressive

Commentaires

Lorsque vous émettez la commande sur un gestionnaire de domaine, tous les postes de travail de ce même domaine sont arrêtés. Puis, le gestionnaire de domaine est lui-même arrêté et la commande s'exécute sur chaque domaine subordonné. La commande continue de s'exécuter de la façon hiérarchique suivante : le gestionnaire de domaine arrête les postes de travail du même domaine, s'arrête lui-même, puis la commande s'exécute ensuite sur les domaines subordonnés.

Exemples

La figure 28, à la page 541 et le tableau 70, à la page 541 montrent les postes de travail arrêtés par l'exécution de la commande **stop ;progressive** sur DM2 et DM4.

DM n sont des gestionnaires de domaine et A nn sont des agents.

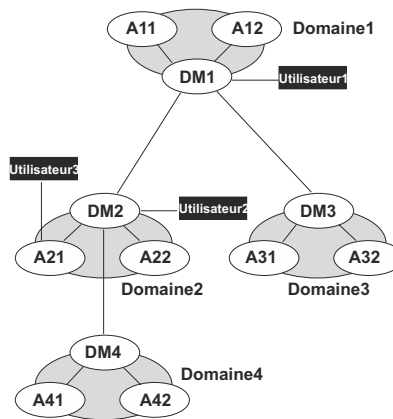


Figure 28. Exemple de réseau

Tableau 70. Postes de travail arrêtés par la commande stop ;progressive

Commande	Arrêt par DM2	Arrêt par DM4
stop ;progressive	A21 A22 DM2	A41 A42 DM4

stopappserver

Arrête WebSphere Application Server sur le poste de travail.

Syntaxe

```
{stopappserver | stopapps} [domaine!]poste de travail
[;wait]
```

Arguments

domaine

Indique le nom du domaine du poste de travail. Etant donné que les postes de travail ont des noms uniques, le domaine n'est pas nécessaire lors de l'arrêt de WebSphere Application Server sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

Poste de travail

Indique le nom du poste de travail sur lequel vous souhaitez arrêter le moteur de surveillance. Les caractères génériques sont acceptés. Si aucun domaine et poste de travail ne sont spécifiés, l'action se fait sur le poste de travail local.

wait Attendez jusqu'à l'arrêt du serveur WebSphere Application Server avant d'émettre une autre commande.

Commentaires

Le droit d'arrêter des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Sous Windows, les systèmes s'abstiennent d'utiliser les services Windows pour arrêter WebSphere Application Server. Si vous utilisez les services Windows, le processus *appserverman*, dont l'exécution se poursuit, redémarre WebSphere Application Server. Utilisez cette commande ou la commande **stopWas** à la place (sans l'option **-direct**).

Lorsque vous exécutez la commande, le processus *appserverman* vérifie d'abord si WebSphere Application Server peut extraire des données d'identification de l'utilisateur (nom d'utilisateur et mot de passe) du fichier *soap.client.props* situé dans le profil WebSphere Application Server. S'ils ne peuvent être extraits, *appserverman* les lit à partir du fichier *useropts* de l'utilisateur et exécute le script *stopServer.sh* (bat) pour les transmettre à WebSphere Application Server.

Pour pouvoir exécuter la commande, vous devez donc terminer l'une des deux procédures de personnalisation afin de fournir les données d'identification de l'utilisateur à WebSphere Application Server :

- Personnalisez les propriétés de nom d'utilisateur (*com.ibm.SOAP.loginUserId*) et de mot de passe (*com.ibm.SOAP.loginPassword*) dans le fichier *soap.client.props* situé dans :

chemin_profil_WAS/properties

(Version 9.1 et maître et agent ultérieurs)

où *chemin_profil_WAS* correspond au chemin d'accès du profil WebSphere Application Server que vous avez indiqué durant l'installation. Le chemin d'accès par défaut est *<rép_principale_TWA>/WAS/TWSPprofile*.

Vous devez également :

1. Définir la propriété *com.ibm.SOAP.securityEnabled* par *true* dans le même fichier pour activer la sécurité du client SOAP.
 2. Exécuter le script **encryptProfileProperties.sh** pour chiffrer le mot de passe. Pour plus d'informations sur cet outil du serveur d'applications, voir *IBM Workload Scheduler - Guide d'administration*.
- Personnaliser la section Attributs des connexions *conman* du fichier *localopts* en spécifiant les détails du connecteur ou du gestionnaire de domaine maître.

Vous devez également :

1. Créer (ou personnaliser, s'il existe déjà) manuellement le fichier *useropts*, en ajoutant les attributs *USERNAME* et *PASSWORD* de l'utilisateur qui exécutera *stopappserver*. Vérifiez que le nom de fichier *useropts* est entré dans la clé *USEROPTS* de la section Attributs des connexions *conman* (CLI). Voir *IBM Workload Scheduler - Guide d'administration* pour plus de détails.
2. Pour chiffrer le mot de passe dans le fichier *useropts*, il suffit d'exécuter *conman*.

stopbrokerapp

Arrête l'application Dynamic Workload Broker.

Syntaxe

{stopbrokerapp} [*domaine!*]*poste de travail*

Arguments

domaine

Indique le nom du domaine du poste de travail. Dans la mesure où les postes de travail portent des noms uniques, le domaine n'est pas nécessaire

lors de l'arrêt de Dynamic Workload Broker sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail sur lequel vous souhaitez arrêter Dynamic Workload Broker. Les caractères génériques sont acceptés. Si aucun domaine et poste de travail ne sont spécifiés, l'action se fait sur le poste de travail local.

Commentaires

Le droit d'arrêter des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Vous pouvez également arrêter Dynamic Workload Broker à l'aide de `wastool stopBrokerApplication`.

stopeventprocessor

Arrête le serveur de traitement d'événements.

Syntaxe

`{stopeventprocessor | stopevtp} [domaine!][poste_de_travail]`

Arguments

domaine

Indique le nom du domaine du poste de travail.

poste de travail

Indique le nom du gestionnaire de domaine maître, du maître de sauvegarde ou du poste de travail installé en tant que maître de sauvegarde, qui fonctionne comme un agent tolérant aux pannes normal, sur lequel vous souhaitez arrêter le serveur de traitement des événements. Les caractères génériques ne sont pas admis.

Vous pouvez omettre le nom du poste de travail si vous exécutez la commande localement.

Commentaires

Cette commande ne peut pas être émise de façon asynchrone.

Si vous émettez la commande à partir d'un poste de travail différent de celui sur lequel le processus d'événements est configuré, la commande utilise le client de ligne de commande, de manière à définir correctement les données d'identification de l'utilisateur du client de ligne de commande.

Le droit d'arrêter des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions** > **Arrêter le processeur d'événement**.

stopmon

Arrête le moteur de surveillance sur le poste de travail.

Syntaxe

```
(stopmon | stopm) [domaine!]poste de travail  
[;wait]  
[;noask]
```

Arguments

domaine

Indique le nom du domaine du poste de travail. Dans la mesure où les postes de travail portent des noms uniques, le domaine n'est pas nécessaire lors de l'arrêt du moteur de surveillance sur un poste de travail spécifique. Les caractères génériques sont acceptés.

Si l'argument *domaine* n'est pas indiqué et que l'argument *poste_de_travail* comporte des caractères génériques, le domaine par défaut est celui dans lequel **conman** est en cours d'exécution.

poste de travail

Indique le nom du poste de travail sur lequel vous souhaitez arrêter le moteur de surveillance. Les caractères génériques sont acceptés.

wait Indique qu'aucune autre commande ne doit être acceptée tant que le moteur de surveillance ne s'est pas arrêté.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

Le moteur de surveillance est redémarré automatiquement lorsque le plan de production suivant est activé (sous Windows également lorsque IBM Workload Scheduler est redémarré) à moins que vous ne désactiviez l'option locale autostart monman.

La commande est asynchrone, à moins de spécifier le mot clé **wait**.

Le droit d'arrêter des actions sur des objets cpu doit être activé dans le fichier de sécurité pour exécuter cette commande.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions > Arrêter le moniteur d'événements**.

submit docommand

Soumet une commande à lancer sous forme de travail.

Pour exécuter cette commande, dans le fichier de sécurité, vous devez disposer d'un accès *submit* pour le travail portant le nom spécifié dans sa définition de base de données et, si vous utilisez le mot clé *alias*, portant également le nom spécifié avec ce mot clé. De plus, si vous utilisez le mot clé *recoveryjob*, vous devez disposer d'un accès *submit* pour le travail spécifié avec ce mot clé.

Pour inclure des dépendances needs et prompt, vous devez avoir un accès *use* aux ressources et aux invites globales.

Si vous soumettez le travail sur un poste de travail autre que le gestionnaire de domaine maître, vous devez vous connecter en tant qu'utilisateur :

- détenant les autorisations d'accès appropriées définies dans le fichier le fichier useropts pour vous connecter au gestionnaire de domaine maître par l'intermédiaire de WebSphere Application Server
- autorisé à exécuter les commandes submit dans le fichier de sécurité stocké sur le gestionnaire de domaine maître

Syntaxe

```
{submit
docommand | sbd} [poste de travail#"cmd"
                  [;alias[=nom]]
                  [;into=[poste de travail#]
                  {ID_flot_travaux;schedid | nom_flot_travaux
                  ([hhmm[date]])}]
                  [;option_travail[:...]]
```

Arguments

Poste de travail

Indique le nom du poste de travail sur lequel le travail va être lancé. Les caractères génériques sont admis, auquel cas, le travail est lancé sur les

postes de travail éligibles. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé. Vous ne pouvez pas indiquer une classe de postes de travail ou un domaine.

Remarque : Du fait d'une limitation dans la façon dont Windows gère le signe égal (=) dans l'environnement shell, vous devez masquer le signe égal (=) comme suit `\=\` lorsque vous soumettez des commandes Windows à l'aide de **submit docommand**. Par exemple, pour définir la variable locale **var1** sur *hello*, vous devez émettre la commande suivante :

```
%sbd "set var1\="\"hello"
```

commande

Indique une commande système valide comportant jusqu'à 255 caractères. L'intégralité de la commande doit être placée entre guillemets ("). La commande est traitée sous forme de travail et toutes les règles relatives aux travaux s'appliquent.

alias=*nom*

Indique un nom unique à attribuer au travail. Si le mot clé **alias** est entré sans qu'aucun nom ne soit indiqué, un nom est créé à partir des six premiers caractères alphanumériques (en majuscules) de la commande, selon le nombre de caractères de la commande, suivis d'un nombre aléatoire de dix chiffres. Si la commande contient des caractères blancs, le nom est créé à partir des six premiers caractères alphanumériques précédant le caractère blanc. Par exemple, dans le cas de la commande **rm apfile**, le nom généré est similaire à **RM0123456789**. Si la commande comporte plus de six caractères alphanumériques telle que **"wlsinst"**, le nom créé est **WLSINS0396578515**.

Si aucun **alias** n'est indiqué la première fois que vous soumettez la commande, un nom de travail est créé à partir des 255 caractères du nom de la commande. Si vous soumettez une commande une seconde fois à partir du même poste de travail, le mot clé **alias** est obligatoire et doit être unique pour chaque commande soumise.

into=*instance_flot_travaux*

Identifie l'instance de flot de travaux dans laquelle le travail sera placé pour être lancé. Sélectionnez l'instance de flot de travaux comme suit :

```
[poste_de_travail#]nom_flot_travaux([hhmm[ date]])
```

ou

```
[poste_de_travail#]ID_flot_travaux ;schedid
```

Si l'argument **into** n'est pas indiqué, le travail est ajouté à un flot de travaux appelé **JOBS**.

option_travail

Indiquez l'un des arguments suivants :

at=*hhmm* [**timezone** | **tz** *fuseau_horaire*] [**+n days** | *mm/dd[/yy]*] | [**absolute** | **abs**]

confirmed

critical

deadline=*heure* [**timezone** | **tz** *fuseau_horaire*][**+n day[s** | *mm/jj[/aa]*]

every=*fréquence*

follows=[agent_réseau::][poste_travail#]{nom_flot_travaux(hhmm [mm/dd/yy])
[.job | @] | id_flot_travaux.travail;schedid}| travail '[IF nom_condition[|
nom_condition][| ...]]' [;nocheck];wait=heure][,...]

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

wait La durée en secondes pendant laquelle IBM Workload Scheduler attend avant d'effectuer une deuxième vérification par rapport à la dépendance si l'objet spécifié dans la dépendance n'existe pas. Si l'objet créé dans la dépendance a été créé au cours de l'intervalle **wait**, la soumission est exécutée ; sinon le travail n'est pas soumis.

nocheck

Ignorer la dépendance. Si l'objet spécifié dans la dépendance n'existe pas, la soumission est exécutée de toute façon.

Remarque : L'argument **;nocheck** n'est pas pris en charge dans les dépendances interréseaux.

interactive

Remarque : Ce mot clé ne peut être utilisé que dans les environnements Windows.

maxdur=heure[on**maxdur** action]

mindur=heure[on**mindur** action]

logon=utilisateur.

needs=[num] [poste_de_travail#]ressource[,...]

opens=[poste_travail#]"nom_fichier"[(qualificatif)][,...]

priority=[pri | hi | go]

prompt="[: | !]texte" | nom_invite[,...]

recovery=stop | continue | rerun

recoveryjob= [poste_de_travail#]nom_travail

Nom du travail de reprise différent de celui spécifié dans la définition de travail de la base de données (le cas échéant).

after [poste_de_travail#]nom_travail

abendprompt "texte"

until heure [timezone | tz fuseau_horaire][+n jour[s] | [absolute | abs]]
[;on**until** action]

La valeur par défaut de **option_travail** est l'utilisateur du poste de travail sur lequel la commande est exécutée.

Utilisation des paramètres locaux

Vous pouvez utiliser les paramètres locaux comme des valeurs associées aux mots clés suivants :

- `commande`
- `opens`
- `logon`
- `invite`
- `abendprompt`

Les paramètres locaux sont définis et gérés à l'aide de la commande d'utilitaire `parms` dans une base de données locale du poste de travail où le travail est exécuté. Les paramètres sont résolus sur le poste de travail pendant l'exécution de la commande **`submit`**.

Commentaires

Les travaux soumis en production à partir de la ligne de commande **`conman`** ne sont pas inclus dans le plan de préproduction et ne peuvent donc pas être pris en compte lors de l'identification des prédécesseurs des dépendances de prédécesseur/successeur externes.

Si aucun *poste de travail* n'est indiqué avec les options `follows`, `needs` ou `opens` ou `into`, le poste de travail sur lequel se trouve le travail est utilisé par défaut.

L'agenda classe les dépendances de prédécesseur/successeur comme *internes* lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le format `jobStreamName.workstationName.jobName`.

Lorsque vous soumettez l'objet dans un flot de travaux et que vous ajoutez une dépendance de prédécesseur/successeur qui a le même nom de flot de travaux (par exemple, vous soumettez l'objet dans le flot de travaux `schedA` et vous définissez une dépendance de prédécesseur/successeur sur `schedA.job2`), la dépendance est traitée comme une dépendance de prédécesseur/successeur *externe*. Depuis la version 8.3, contrairement aux versions précédentes, étant donné que l'agenda utilise le critère de correspondance `sameday` pour résoudre des dépendances externes, les dépendances générées de cette manière ne sont jamais ajoutées lors de la première soumission de l'objet.

Exemples

Pour soumettre une commande **`rm`** dans le flot de travaux `JOBS` avec une dépendance de prédécesseur/successeur, exécutez la commande suivante :

```
submit docommand="rm apfile";follows sked3
```

Pour soumettre une commande **`sort`** caractérisée par l'alias **`sortit`** et placer le travail dans le flot de travaux `reports` dont l'heure d'exécution **`at`** est définie sur 17:30, exécutez la commande suivante :

```
sbd "sort < file1 > file2";alias=sortit;into=reports;at=1730
```

Pour soumettre la commande **`chmod`** sur tous les postes de travail dont le nom commence par `site`, exécutez la commande suivante :

```
sbd=site@#"chmod 444 file2";alias
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

IBM Workload Automation : Dynamic Workload Console - Guide d'utilisation, section "Soumission de travaux ad hoc".

submit file

Soumet un fichier à lancer sous forme de travail.

Pour exécuter cette commande, dans le fichier de sécurité, vous devez disposer d'un accès **submit** pour le travail portant le nom spécifié dans sa définition de base de données et, si vous utilisez le mot clé **alias**, portant également le nom spécifié avec ce mot clé. De plus, si vous utilisez le mot clé **recoveryjob**, vous devez disposer d'un accès **submit** pour le travail spécifié avec ce mot clé.

Pour inclure des dépendances needs et prompt, vous devez avoir un accès **use** aux ressources et aux invites globales.

Si vous soumettez le travail sur un poste de travail autre que le gestionnaire de domaine maître, vous devez vous connecter en tant qu'utilisateur :

- Détenant les autorisations d'accès appropriées définies dans le fichier le fichier useropts pour vous connecter au gestionnaire de domaine maître par l'intermédiaire de WebSphere Application Server
- Autorisé à exécuter les commandes submit dans le fichier de sécurité stocké sur le gestionnaire de domaine maître

Syntaxe

```
{submit  
file | sbf} "nom_fichier"  
    [;alias[=nom]]  
    [;into=[poste_de_travail#{ID_flot_travaux  
;schedid | nom_flot_travaux([hhmm[date]])}]  
    [;option_travail[:...]]  
    [;noask]
```

Arguments

nom_fichier

Indique le nom du fichier (jusqu'à 255 caractères). Les caractères génériques sont acceptés. Le nom doit être placée entre guillemets (") s'il contient des caractères autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/) et traits de soulignement (_). Reportez-vous aux exemples.

alias=nom

Indique un nom unique à attribuer au travail. Si le mot clé **alias** est entré sans qu'aucun nom ne soit indiqué, un nom est créé à partir des six premiers caractères alphanumériques (en majuscules) du nom de fichier, selon le nombre de caractères du nom de fichier, suivis d'un nombre aléatoire de dix chiffres. Par exemple, dans le cas où le nom de fichier est jclttx5, le nom généré est similaire à JCLTTX0123456789.

Si vous n'indiquez pas d'**alias**, un nom de fichier est créé à partir des 255 caractères alphanumériques du nom de base du fichier, en majuscules.

Dans l'un ou l'autre des cas ci-dessus, si le nom du fichier ne commence pas par une lettre, vous êtes invité à utiliser **alias= nom**.

Si vous soumettez un fichier une seconde fois à partir du même poste de travail, le mot clé **alias** est obligatoire et doit être unique pour chaque fichier soumis.

into=instance_flot_travaux

Identifie l'instance de flot de travaux dans laquelle le travail sera placé pour être lancé. Sélectionnez l'instance de flot de travaux comme suit :

[poste_de_travail#]nom_flot_travaux([hhmm [date]])

ou

[poste_de_travail#]ID_flot_travaux ;**schedid**

Si l'argument **into** n'est pas indiqué, le travail est ajouté à un flot de travaux appelé **JOBS**.

option_travail

Indiquez l'un des arguments suivants :

at=hhmm [timezone | tz fuseau_horaire] [+n days | mm/dd[/yy]] | [absolute | abs]

confirmed

critical

deadline=heure[timezone | tz fuseau_horaire][+n days | mm/jj[/aa]]

every=fréquence

follows=[agent_réseau::][poste_travail#]{nom_flot_travaux(hhmm [mm/dd[/yy]]) [.job | @] | id_flot_travaux.travail;schedid** | travail '[IF nom_condition[| nom_condition][| ...]]' [;**nocheck**][;**wait=heure**][,...]**

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

wait La durée en secondes pendant laquelle IBM Workload Scheduler attend avant d'effectuer une deuxième vérification par rapport à la dépendance si l'objet spécifié dans la dépendance n'existe pas. Si l'objet créé dans la dépendance a été créé au cours de l'intervalle **wait**, la soumission est exécutée ; sinon le travail n'est pas soumis.

nocheck

Ignorer la dépendance. Si l'objet spécifié dans la dépendance n'existe pas, la soumission est exécutée de toute façon.

Remarque : L'argument **nocheck** n'est pas pris en charge dans les dépendances interréseaux.

interactive

Remarque : Ce mot clé ne peut être utilisé que dans les environnements Windows.

logon=*utilisateur*

maxdur=*heure*[**onmaxdur** *action*]

mindur=*heure*[**onmindur** *action*]

needs=[*num*] [*poste_de_travail*#]*ressource*[,...]

opens=[*poste_travail*#]"*nom_fichier*"[(*qualificatif*)][,...]

priority=[*pri* | **hi** | **go**]

prompt="[: | !]*texte*" | *nom_invite*[,...]

recovery=**stop** | **continue** | **rerun**

recoveryjob= [*poste_de_travail*#]*nom_travail*

Nom du travail de reprise différent de celui spécifié dans la définition de travail de la base de données (le cas échéant).

after [*poste_de_travail*#]*nom_travail*

abendprompt "*texte*"

until *heure* [**timezone** | **tz** *fuseau_horaire*][**+n jour[s]** | [**absolute** | **abs**]]
[**onuntil** *action*]

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque fichier éligible.

Utilisation des paramètres locaux

Vous pouvez utiliser les paramètres locaux comme des valeurs associées aux mots clés suivants :

- opens
- logon
- invite
- abendprompt

Les paramètres locaux sont définis et gérés à l'aide de la commande d'utilitaire parms dans une base de données locale du poste de travail où le travail est exécuté. Les paramètres sont résolus sur le poste de travail pendant l'exécution de la commande **submit**.

Commentaires

Les travaux soumis en production à partir de la ligne de commande **conman** ne sont pas inclus dans le plan de préproduction et ne peuvent donc pas être pris en compte lors de l'identification des prédécesseurs des dépendances de prédécesseur/successeur externes.

Si vous ne spécifiez pas de poste de travail avec **follows**, **needs**, **opens** ou **into**, la valeur par défaut est le poste de travail sur lequel est exécuté **conman**.

L'agenda classe les dépendances de prédécesseur/successeur comme *internes* lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le format *jobStreamName.workstationName.jobName*.

Lorsque vous soumettez l'objet dans un flot de travaux et que vous ajoutez une dépendance de prédécesseur/successeur qui a le même nom de flot de travaux (par exemple, vous soumettez l'objet dans le flot de travaux schedA et vous définissez une dépendance de prédécesseur/successeur sur schedA.job2), la dépendance est traitée comme une dépendance de prédécesseur/successeur *externe*. Depuis la version 8.3, contrairement aux versions précédentes, étant donné que l'agenda utilise le critère de correspondance sameday pour résoudre des dépendances externes, les dépendances générées de cette manière ne sont jamais ajoutées lors de la première soumission de l'objet.

Exemples

Pour soumettre un fichier dans le flot de travaux jobs en utilisant le nom de travail myjc1, exécutez la commande suivante :

```
submit file=d:\jobs\lib\daily\myjc1
```

où la séquence ;into a été omise.

Pour soumettre un fichier dans le flot de travaux missed en utilisant le nom de travail misjob4, exécutez la commande suivante :

```
sbfc /usr/lib/mis/jc14;alias=misjob4;into=misscd ;needs=2 slots
```

Le travail requiert deux unités de la ressource slots.

Pour soumettre tous les fichiers dont le nom commence par back dans le flot de travaux bkup, exécutez la commande suivante :

```
sbfc "/usr/lib/backup/back@";into=bkup
```

Pour soumettre le fichier tws_env.cmd dont le chemin d'accès contient un espace, sur un poste de travail Windows, exécutez :

- En mode interactif :

```
sbfc "\"C:\Program Files\IBM\TWS\lucaMDM\tws_env.cmd\"";alias=MYJOB
```

Sous Windows, les guillemets (") doivent être gérés par la séquence de caractères "\.

- En mode ligne de commande :

```
conman sbfc "\"\\\"C:\Program Files\IBM\TWS\lucaMDM\tws_env.cmd\\\"";alias=MYJOB
```

Sous Windows, lorsque la commande est exécutée en externe à partir de l'environnement conman, la séquence d'échappement est plus longue.

où "\" est le caractère d'échappement de l'espace dans le chemin d'accès au fichier.

submit job

Soumet un travail à lancer.

Pour exécuter cette commande, dans le fichier de sécurité, vous devez disposer d'un accès *submit* (*submitdb*) pour le travail portant le nom spécifié dans sa définition de base de données et, si vous utilisez le mot clé *alias*, portant également le nom spécifié avec ce mot clé. De plus, si vous utilisez le mot clé *recoveryjob*, vous devez disposer d'un accès *submit* pour le travail spécifié avec ce mot clé.

Notez que si vous disposez uniquement des droits de sécurité *submitdb*, vous êtes limité à soumettre les travaux définis dans la base de données. Vous ne pouvez pas soumettre de travaux ad-hoc.

Pour inclure des dépendances *needs* et *prompt*, vous devez avoir un accès *use* aux ressources et aux invites globales.

Si vous soumettez le travail sur un poste de travail autre que le gestionnaire de domaine maître, vous devez vous connecter en tant qu'utilisateur :

- Détenant les autorisations d'accès appropriées définies dans le fichier le fichier *useropts* pour vous connecter au gestionnaire de domaine maître par l'intermédiaire de WebSphere Application Server
- Autorisé à exécuter les commandes **submit** dans le fichier de sécurité stocké sur le gestionnaire de domaine maître

Si vous soumettez un travail reflet, voir Chapitre 23, «Définition et gestion des dépendances croisées», à la page 845 pour plus détails.

Syntaxe

```
{submit
job | sbj} [poste_de_travail#]nom_travail
      [:alias[=nom]]
      [:into=[poste_de_travail#{ID_flot_travaux
schedid | nom_flot_travaux([hhmm[date]])}]
      [:option_travail[:...]]
      [:vartable=nom_table]
      [:noask]
```

Arguments

Poste de travail

Indique le nom du poste de travail sur lequel le travail va être lancé. Les caractères génériques sont admis, auquel cas, le travail est lancé sur les postes de travail éligibles. Par défaut, le poste de travail sur lequel *conman* est en cours d'exécution est utilisé. Vous ne pouvez pas indiquer une classe de postes de travail ou un domaine.

nomtravail

Indique le nom du travail. Les caractères génériques sont admis, auquel cas, tous les travaux éligibles sont soumis. Si le travail figure déjà dans le plan de production et qu'il est soumis dans le même flot de travaux, vous devez utiliser le même argument **alias** pour attribuer un nom unique.

alias=*nom*

Indique un nom unique à attribuer au travail à la place de l'option *nom_travail*. Si le mot clé **alias** est entré sans spécifier de nom, un nom est créé à partir des cinq premiers caractères alphanumériques du *nom_travail*, suivis par un nombre aléatoire à dix chiffres. Le nom apparaît systématiquement en majuscules. Par exemple, si *nom_travail* est défini sur *jcrttx5*, le nom généré sera similaire à *JCRTT1234567890*.

into=*instance_flot_travaux*

Identifie l'instance de flot de travaux dans laquelle le travail sera placé pour être lancé. Sélectionnez l'instance de flot de travaux comme suit :

```
[poste_de_travail#]nom_flot_travaux([hhmm[ date]])
```

ou

[poste_de_travail#]ID_flot_travaux ;**schedid** Si l'argument **into** n'est pas indiqué, le travail est ajouté à un flot de travaux appelé **JOBS**.

option_travail

Indiquez l'un des arguments suivants :

at=hhmm [timezone | tz fuseau_horaire] [+n days | mm/dd[/yy]] | [absolute | abs]

confirmed

critical

deadline=heure[timezone | tz fuseau_horaire][+n days | mm/jj[/aa]]

every=fréquence

follows=[agent_réseau::][poste_travail#]{nom_flot_travaux(hhmm [mm/dd[/yy]]) [.job | @] | id_flot_travaux.job;**schedid**} | travail '[IF nom_condition[| nom_condition] [| ...]]' [**nocheck**][;**wait**=heure][,...]

La variable *nom_condition* indique le nom de la condition définie dans la définition de travail. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction *follows*, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

wait La durée en secondes pendant laquelle IBM Workload Scheduler attend avant d'effectuer une deuxième vérification par rapport à la dépendance si l'objet spécifié dans la dépendance n'existe pas. Si l'objet créé dans la dépendance a été créé au cours de l'intervalle **wait**, la soumission est exécutée ; sinon le travail n'est pas soumis.

nocheck

Ignorer la dépendance. Si l'objet spécifié dans la dépendance n'existe pas, la soumission est exécutée de toute façon.

Remarque : L'argument **nocheck** n'est pas pris en charge dans les dépendances interréseaux.

maxdur=heure[onmaxdur action]

mindur=heure[onmindur action]

needs=[num] [poste_de_travail#]ressource[,...]

opens=[poste_travail#]"nom_fichier"[(qualificatif)][,...]

priority=[pri | hi | go]

prompt="[: | !]texte" | nom_invite[,...]

recovery=stop | continue | rerun

recoveryjob= [poste_de_travail#]nom_travail

Nom du travail de reprise différent de celui spécifié dans la définition de travail de la base de données (le cas échéant).

after [poste_de_travail#]nom_travail

abendprompt "texte"

until heure [timezone | tz fuseau_horaire][+n jour[s] | [absolute | abs]]
[;onuntil action]

variable=nom_table

Indique le nom de la table de variables, s'il est différent du nom par défaut, dans laquelle sont définies les variables que vous souhaitez utiliser.

A faire :

- Avec cette commande, vous pouvez utiliser une substitution de variable pour les mots clés suivants :
 - opens
 - prompt
 - abendprompt
- Placez la variable entre accents circonflexes (^), puis l'ensemble de la chaîne entre guillemets. Si la variable contient une partie d'un chemin, vérifiez que les accents circonflexes ne sont pas immédiatement précédés d'une barre oblique inversée (\). En effet, la séquence \^ pourrait être interprétée comme une séquence d'échappement et serait alors résolue par le programme d'analyse syntaxique comme un accent circonflexe. Si nécessaire, placez la barre oblique inversée dans la définition de la variable entre accents circonflexes.
- Les variables spécifiées dans la définition du travail avec le format $\${nom\ de\ variable}$ ne sont pas résolues.

Si vous soumettez un travail contenant des variables définies dans une table de variables qui n'est pas la table de variables par défaut et que vous n'indiquez pas cette table dans le cycle d'exécution, dans le flot de travaux ou sur le poste de travail, les variables ne sont pas résolues. Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque travail éligible.

Commentaires

Les travaux soumis en production à partir de la ligne de commande **conman** ne sont pas inclus dans le plan de préproduction et ne peuvent donc pas être pris en compte lors de l'identification des prédécesseurs des dépendances de prédécesseur/successeur externes.

Si aucun poste de travail n'est indiqué avec les options **follows**, **needs** ou **opens** ou **into**, le poste de travail sur lequel se trouve le travail est utilisé par défaut.

at indique à quel moment peut être soumis le travail. Si le mot clé **at** est utilisé, le travail ne peut pas démarrer avant l'heure définie dans ce mot clé. Notez que si le gestionnaire de domaine maître de votre réseau s'exécute lorsque la valeur **yes** est attribuée aux options **enLegacyStartOfDayEvaluation** et **enTimeZone** afin de convertir l'heure **startOfDay** du gestionnaire de domaine maître en fuseau horaire local sur chaque poste de travail du réseau, vous devez ajouter le mot clé **absolute** pour le faire fonctionner.

L'agenda classe les dépendances de prédécesseur/successeur comme *internes* lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il

les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le format *jobStreamName.workstationName.jobName*.

Lorsque vous soumettez l'objet dans un flot de travaux et que vous ajoutez une dépendance de prédécesseur/successeur qui a le même nom de flot de travaux (par exemple, vous soumettez l'objet dans le flot de travaux schedA et vous définissez une dépendance de prédécesseur/successeur sur schedA.job2), la dépendance est traitée comme une dépendance de prédécesseur/successeur *externe*. Depuis la version 8.3, contrairement aux versions précédentes, étant donné que l'agenda utilise le critère de correspondance *sameday* pour résoudre des dépendances externes, les dépendances générées de cette manière ne sont jamais ajoutées lors de la première soumission de l'objet.

Exemples

Pour soumettre les travaux test dans le flot de travaux JOBS, exécutez la commande suivante :

```
sbj test
```

Pour soumettre un travail caractérisé par l'alias rptx4 et placer le travail dans le flot de travaux reports dont l'heure d'exécution **at** est définie sur 17:30, exécutez la commande suivante :

```
sbj rjob4;alias=rptx4;into=reports;at=1730
```

Pour soumettre le travail txjob3 sur tous les postes de travail dont le nom commence par **site**, exécutez la commande suivante :

```
sbj=site@#txjob3;alias
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

IBM Workload Automation : Dynamic Workload Console - Guide d'utilisation, section "Soumission de travaux prédéfinis".

submit sched

Soumet un flot de travaux pour le traitement.

Pour exécuter cette commande, dans le fichier de sécurité, vous devez disposer d'un accès **submit** pour le flot de travaux portant le nom spécifié dans sa définition de base de données et, si vous utilisez le mot clé **alias**, portant également le nom spécifié avec ce mot clé. Pour inclure des dépendances **needs** et **prompt**, vous devez avoir un accès **use** aux ressources et aux invites globales.

La commande **submit schedule** utilise les données d'identification définies dans le fichier **useropts** appartenant à l'utilisateur *utilisateur_TWS* qui a installé ce poste de travail.

Si vous soumettez le flot de travaux à partir d'un poste de travail différent du gestionnaire de domaine maître, vous devez être connecté en tant qu'utilisateur :

- détenant les autorisations d'accès appropriées définies dans le fichier le fichier **useropts** pour vous connecter au gestionnaire de domaine maître par l'intermédiaire de WebSphere Application Server

- autorisé à exécuter les commandes submit dans le fichier de sécurité stocké sur le gestionnaire de domaine maître

Syntaxe

```
{submit
sched | sbs} [poste de travail#]nom_flot_travaux
      [:alias[=nom]]
      [:option_flot_travaux[:...]]
      [:vartable=nom_table]
      [:noask]
```

Arguments

Poste de travail

Indique le nom du poste de travail sur lequel le flot de travaux va être lancé. Les caractères génériques sont admis, auquel cas, le flot de travaux est lancé sur les postes de travail éligibles. Par défaut, le poste de travail sur lequel **conman** est en cours d'exécution est utilisé. Vous ne pouvez pas indiquer une classe de postes de travail ou un domaine.

nom_flot_travaux

Indique le nom du flot de travaux. Les caractères génériques sont admis, auquel cas, tous les flots de travaux éligibles sont soumis. Si le flot de travaux figure déjà dans le plan de production, vous devez utiliser l'argument **alias** pour attribuer un nom unique.

alias=nom

Indique un nom unique à attribuer au flot de travaux à la place de *nom_flot_travaux*. Si une valeur lui est attribuée, elle correspond également à l'*ID_flot_travaux*. Si le mot clé **alias** est entré sans spécifier de nom, un nom est créé à partir des cinq premiers caractères alphanumériques du *nom_flot_travaux*, suivis par un nombre aléatoire à dix chiffres. Le nom apparaît systématiquement en majuscules. Par exemple, si *nom_flot_travaux* est défini sur *sttrom*, le nom généré sera similaire à *STTR01234567890*.

L'autorisation de soumettre le calendrier est vérifiée dans le fichier de sécurité à l'aide du nom d'origine et non du nom d'alias.

option_flot_travaux

Entrez l'un des éléments suivants (voir «Détail des mots clés pour la définition des flots de travaux», à la page 233 pour rechercher les options mutuellement exclusives) :

```
[at=hhmm [timezone | tz fuseau_horaire] [+n days | date] [absolute | abs]] |
[schedtime=[hhmm [date] | [+n days]]
```

où :

at indique à quel moment le flot de travaux peut être lancé. Si le mot clé **at** est utilisé, le flot de travaux ne peut pas démarrer avant l'heure définie par ce mot clé (voir la rubrique relative aux mots clés de définition du flot de travaux dans le chapitre "Définition d'objets dans la base de données" dans *"IBM Workload Scheduler - Guide d'utilisation et de référence"* pour plus d'informations sur le mot clé "at"). Notez que si le gestionnaire de domaine maître de votre réseau s'exécute lorsque la valeur **yes** est attribuée aux options **enLegacyStartOfDayEvaluation** et **enTimeZone** afin de convertir l'heure **startOfDay** du gestionnaire de domaine maître en

fuseau horaire local sur chaque poste de travail du réseau, vous devez ajouter le mot clé **absolute** pour le faire fonctionner.

schedtime représente le jour et l'heure de positionnement du flot de travaux dans le plan. Le flot de travaux est lancé si, à cet instant il ne comporte aucune dépendance et qu'aucune restriction horaire **at** n'est définie à son égard. La valeur affectée à **schedtime** ne représente pas une dépendance pour le flot de travaux. Sa valeur s'affiche alors dans les colonnes *Heure_calend* de la sortie des commandes d'affichage. Si une restriction **at** est définie, la valeur affectée à **schedtime** est remplacée par la valeur **at**. Lorsque le flot de travaux démarre effectivement, la valeur affectée à **schedtime** est remplacée par l'heure de démarrage réelle du flot de travaux.

Le format utilisé pour *date* dépend de la valeur affectée à la variable *format_date* spécifiée dans le fichier *localopts*.

Si aucun fuseau horaire supplémentaire n'est spécifié, celui défini sur le poste de travail exécutant la commande est pris en compte.

carryforward

Autorise le report d'un flot de travaux au plan de production suivant, s'il n'est pas terminé avant la fin du plan de production courant.

deadline=*heure*[**timezone** | **tz** *fuseau_horaire*][**+n days** | *date*]

Si aucun fuseau horaire supplémentaire n'est spécifié, celui défini sur le poste de travail exécutant la commande est pris en compte.

follows=[*agent_réseau*::][*poste_travail*#][*nom_flot_travaux*(*hhmm* [*mm/dd/yy*]) [*.job* | @] | *id_flot_travaux.travail*;**schedid**] | *travail* '**[IF** *nom_condition*[| *nom_condition*][| ...]' [**;nocheck**][**;wait**=*heure*][,...]

Les critères de correspondance utilisés lors de la soumission de flots de travaux en production sont différents de ceux qui sont utilisés pour résoudre les dépendances de **prédécesseur/successeur** dans le plan de préproduction. Lorsqu'un flot de travaux (JS_A, par exemple) contenant une dépendance de **prédécesseur/successeur** provenant d'un travail ou d'un flot de travaux (JS_B, par exemple) est soumis à partir du programme de ligne de commande **conman**, l'instance de prédécesseur JS_B est définie en fonction des critères suivants :

1. L'instance la plus proche de JS_B précédant JS_A.
2. Si aucune instance remplacée de JS_B n'existe, l'instance de prédécesseur est l'instance la plus proche de JS_B suivant JS_A.
3. Sinon, une erreur s'affiche et la commande échoue si le mot clé **;nocheck** n'est pas utilisé.

L'instance de flot de travaux prédécesseur est recherchée parmi les instances ajoutées au plan de production lorsque **JnextPlan** a été exécuté et les instances soumises en production à l'aide de la commande **sbs**, y compris celles qui ont été soumises avec un alias.

La variable *nom_condition* indique le nom de la condition spécifié dans la définition de travail. Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'autres **conditions de sortie**, basées sur une expression de mappage telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Dans chaque instruction **follows**, vous ne pouvez indiquer qu'un seul type de dépendance : conditions de

statut ou de sortie. Au moment de la soumission, vous pouvez ajouter des conditions de statut ou de sortie, mais pas des dépendances jointes.

wait La durée en secondes pendant laquelle IBM Workload Scheduler attend avant d'effectuer une deuxième vérification par rapport à la dépendance si l'objet spécifié dans la dépendance n'existe pas. Si l'objet créé dans la dépendance a été créé au cours de l'intervalle **wait**, la soumission est exécutée ; sinon le travail n'est pas soumis.

nocheck
Ignorer la dépendance. Si l'objet spécifié dans la dépendance n'existe pas, la soumission est exécutée de toute façon.

Remarque : L'argument **nocheck** n'est pas pris en charge dans les dépendances interréseaux.

limit=nombre_max_travaux

Limite le nombre maximal de travaux pouvant être exécutés simultanément dans un flot de travaux sur la même unité centrale.

needs=[num] [poste_de_travail#]ressource[,...]

Définit les ressources qui doivent être disponibles avant le lancement d'un travail ou d'un flot de travaux. Vous pouvez utiliser le mot clé **needs** soit dans une définition de flot de travaux, soit dans la définition des travaux contenus, mais pas dans les deux à la fois.

opens=[poste_travail#"nom_fichier"[(qualificatif)][,...]

Indique les fichiers qui doivent être disponibles avant le lancement d'un travail ou d'un flot de travaux.

priority=[pri | hi | go]

Définit la priorité d'un travail ou d'un flot de travaux. En affectant une priorité différente aux travaux ou flots de travaux, vous déterminez lequel démarre en premier, à condition que les dépendances soient résolues.

prompt="[: | !]texte" | nom_invite[,...]

Définit les invites auxquelles une réponse affirmative doit être donnée avant le lancement d'un travail ou d'un flot de travaux.

**until heure [timezone | tz fuseau_horaire][+n jour[s] | [absolute | abs]]
[;onuntil action]**

Selon la définition d'objet à laquelle appartient le mot clé **until**, spécifie la dernière heure à laquelle un flot de travaux doit être terminé ou la dernière heure à laquelle un travail peut être lancé.

Si aucun fuseau horaire supplémentaire n'est spécifié, celui défini sur le poste de travail exécutant la commande est pris en compte.

vartable=nom_table

Indique le nom de la table de variables, s'il est différent du nom par défaut, dans laquelle sont définies les variables que vous souhaitez utiliser.

A faire :

- Avec cette commande, vous pouvez utiliser une substitution de variable pour les mots clés suivants :

- opens
- prompt
- Placez la variable entre accents circonflexes (^), puis l'ensemble de la chaîne entre guillemets. Si la variable contient une partie d'un chemin, vérifiez que les accents circonflexes ne sont pas immédiatement précédés d'une barre oblique inversée (\). En effet, la séquence \^ pourrait être interprétée comme une séquence d'échappement et serait alors résolue par le programme d'analyse syntaxique comme un accent circonflexe. Si nécessaire, placez la barre oblique inversée dans la définition de la variable entre accents circonflexes.

Si vous soumettez un flot de travaux contenant des variables définies dans une table de variables qui n'est pas la table de variables par défaut et que vous n'indiquez pas cette table dans le cycle d'exécution, dans le flot de travaux ou sur le poste de travail, les variables ne sont pas résolues. Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque flot de travaux éligible.

Commentaires

Les flots de travaux soumis en production à partir de la ligne de commande **conman** ne sont pas inclus dans le plan de préproduction et ne peuvent donc pas être pris en compte lors de l'identification des prédécesseurs des dépendances de prédécesseur/successeur externes.

Si aucun poste de travail n'est indiqué avec les options *follows*, *needs* ou *opens*, le poste de travail sur lequel se trouve le flot de travaux est utilisé par défaut.

L'agenda classe les dépendances de prédécesseur/successeur comme *internes* lorsqu'elles ne sont spécifiées que par leur nom de travail dans le flot de travaux. Il les classe en temps que dépendances *externes* lorsqu'elles sont spécifiées dans le format *jobStreamName.workstationName.jobName*.

Lorsque vous soumettez un flot de travaux qui inclut un travail possédant une dépendance de prédécesseur/successeur qui porte le même nom de flot de travaux (par exemple, le flot de travaux *schedA* inclut un travail nommé *job6* qui possède une dépendance de prédécesseur/successeur sur *schedA.job2*), la dépendance est ajoutée en tant que dépendance de prédécesseur/successeur *externe*. Depuis la version 8.3, contrairement aux versions précédentes, étant donné que l'agenda utilise le critère de correspondance *sameday* pour résoudre des dépendances externes, les dépendances générées de cette manière ne sont jamais ajoutées lors de la première soumission de l'objet.

Exemples

Pour soumettre le flot de travaux *adhoc* sur le poste de travail *site1* et le définir en tant que flot de travaux **carryforward**, exécutez la commande suivante :

```
submit sched=site1#adhoc;carryforward
```

Pour soumettre le flot de travaux *fox4* caractérisé par un nombre maximal de 2 travaux, une priorité égale à 23 et une échéance fixée à minuit par l'option **until**, exécutez la commande suivante :

```
sbs fox4;limit=2;pri=23;until=0000
```

Pour soumettre le flot de travaux sched3 sur tous les postes de travail dont le nom commence par site, exécutez la commande suivante :

```
sbs=site@#sched3
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

IBM Workload Automation : Dynamic Workload Console - Guide d'utilisation, section "Soumission de flots de travaux prédéfinis".

switcheventprocessor

permuté le serveur de traitement d'événement du gestionnaire de domaine maître au maître de sauvegarde ou inversement.

Notez que vous pouvez exécuter le serveur de traitement d'événement également sur un poste de travail installé en tant que maître de sauvegarde qui s'exécute en tant que simple agent tolérant aux pannes.

Syntaxe

```
{switcheventprocessor | switchevtp} poste de travail
```

Arguments

poste de travail

Indique le nom du gestionnaire de domaine maître ou du maître de sauvegarde sur lequel vous souhaitez faire passer le serveur de traitement d'événement. Les caractères génériques ne sont pas admis.

Commentaires

Si vous émettez la commande à partir d'un poste de travail différent de celui sur lequel le processus d'événements est configuré, la commande utilise le client de ligne de commande, de manière à définir correctement les données d'identification de l'utilisateur du client de ligne de commande.

Dans le cas des maîtres de sauvegarde, le poste de travail doit avoir l'attribut full-status défini sur on.

Le droit de démarrer et d'arrêter des actions sur des objets cpu doit être activé dans le fichier de sécurité pour pouvoir exécuter cette commande.

L'état de corrélation des instances de règle de corrélation en cours est perdu lorsque le serveur est désactivé ou migré. Si la mise en cache des événements reçus est activée dans le fichier de configuration de l'écouteur EIF, les événements mis en cache sont perdus après permutation du traitement d'événement.

Important :

- Avant d'exécuter cette commande, prenez la précaution de lancer **planman deploy**. Pour ce faire, prenez soin de déployer les dernières modifications ou les derniers ajouts apportés aux règles d'événement actives avant de basculer le processeur d'événement. Il s'agit d'éviter que l'ancien processeur d'événement ne reçoive les dernière mises à jours (envoyées automatiquement en fonction de la

configuration de l'option globale **deploymentFrequency**) à la place du nouveau processeur suite à une non-concordance horaire.

- Les horloges du maître et des maîtres de sauvegarde conçus pour exécuter le processeur d'événement doivent être synchronisées en permanence afin d'éviter toutes les incohérences lors du calcul de l'intervalle de temps des règles d'événement en cours d'exécution. En effet, si le processeur d'événement bascule vers un ordinateur non synchronisé, les actions suite au dépassement du délai d'attente du processus de déclenchement doivent subir les délais imprévus. Utilisez un serveur NTP (Network Time Protocol) pour maintenir toutes les horloges synchronisées.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions** > **Devenir processeur d'événement**.

switchmgr

Remplace le gestionnaire de domaine en cours par un gestionnaire de domaine de sauvegarde.

Vous devez avoir un accès **start** et **stop** au gestionnaire de domaine de sauvegarde.

La commande **switchmgr** doit être utilisée uniquement dans le cadre d'une procédure spécifique destinée à permuter les capacités de gestion de domaine d'un gestionnaire de domaine à son gestionnaire de domaine de secours, définitivement ou temporairement. Pour plus d'informations sur ces procédures, voir *IBM Workload Scheduler - Guide d'administration*.

Syntaxe

{switchmgr | switchm} *domaine;nouv_gest*

Arguments

domaine

Indique le domaine dans lequel vous souhaitez effectuer le changement de gestionnaire.

nouv_gest

Indique le nom du nouveau gestionnaire de domaine. Il doit correspondre à un poste de travail du même domaine et être défini par avance comme agent tolérant aux pannes, les options Résolution des dépendances et Statut intégral étant activées.

Commentaires

Cette commande arrête le poste de travail indiqué et le relance ensuite en tant que gestionnaire de domaine. Tous les postes de travail faisant partie du domaine sont informés du changement et l'ancien gestionnaire de domaine est converti en agent tolérant aux pannes dans le domaine.

Lors de la prochaine exécution de JnextPlan sur l'ancien gestionnaire de domaine, le domaine agit comme si une autre commande **switchmgr** avait été exécutée et l'ancien gestionnaire de domaine reprend automatiquement ses responsabilités de gestionnaire de domaine.

Les agents tolérants aux pannes définis avec `securitylevel = on` peuvent ne pas réussir à utiliser le port SSL pour se connecter au nouveau gestionnaire de domaine maître après l'exécution de la commande **switchmgr**. Dans ce cas, pour laisser l'agent démarrer correctement, effectuez l'une des deux opérations suivantes :

- Interrompez puis rétablissez la liaison à l'agent à partir du nouveau gestionnaire de domaine maître.
- Utilisez l'option `securitylevel = force` sur l'agent.

Exemples

Pour basculer le gestionnaire de domaine sur le poste de travail orca dans le domaine masterdm, exécutez la commande suivante :

```
switchmgr masterdm;orca
```

Pour basculer le gestionnaire de domaine sur le poste de travail ruby dans le domaine bldg2, exécutez la commande suivante :

```
switchmgr bldg2;ruby
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système** > **Surveillance de la charge de travail** > **Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Plus d'actions** > **Devenir gestionnaire de domaine maître**.

Commande système

Exécute une commande système.

Syntaxe

[[: | !] *commande-système*

Arguments

commande système

Désigne une commande système valide. Le préfixe (: ou !) est requis lorsqu'un nom de commande est orthographié comme une commande **conman**.

Exemples

Pour exécuter une commande **ps** sous UNIX, exécutez la commande suivante :

```
ps -ef
```

Pour exécuter une commande **dir** sous Windows, exécutez la commande suivante :

```
dir \bin
```

tellop

Envoie un message à la console IBM Workload Scheduler.

Syntaxe

```
{tellop | to} [texte]
```

Arguments

texte Indique le texte du message. Celui-ci peut comporter jusqu'à 900 caractères.

Commentaires

Si la commande **tellop** est émise sur le gestionnaire de domaine maître, le message est envoyé à tous les postes de travail connectés. Si elle est émise sur un gestionnaire de domaine, le message est envoyé à tous les agents connectés de son domaine et des domaines subordonnés. Si elle est émise sur un poste de travail autre qu'un gestionnaire de domaine, le message est envoyé uniquement à son gestionnaire de domaine si celui-ci est connecté. Le message s'affiche uniquement si le niveau de message écran est supérieur à zéro. Voir «console», à la page 453.

Si la commande **tellop** est entrée seule, vous êtes invité à entrer le texte du message. A l'invite, tapez chaque ligne et appuyez sur la touche Retour. A la fin du message, tapez deux barres obliques (//) ou un point (.), et appuyez sur la touche Retour. Vous pouvez utiliser la séquence de retour ligne (\n) pour la mise en forme des messages. Vous pouvez quitter la commande **tellop** à tout moment sans envoyer de message en appuyant sur **Control+c**.

Exemples

Pour envoyer un message, exécutez la commande suivante :

```
tellop TWS sera arrêté à\n4:30 pendant 15 minutes.
```

Pour inviter l'utilisateur à saisir du texte avant d'envoyer un message, exécutez la commande suivante :

```
to
TELLOP>*****
TELLOP>*   TWS sera arrêté à           *
TELLOP>*   4:30 pendant 15 minutes.     *
TELLOP>*****
TELLOP>//
```

unlink

Ferme les liaisons de données entre les postes de travail.

Vous devez avoir un accès *unlink* au poste de travail cible.

Syntaxe

```
unlink [domaine!]poste de travail  
[;noask]
```

Arguments

domaine

Indique le nom du domaine dans lequel les liaisons doivent être arrêtées. Il n'est pas nécessaire d'indiquer le nom de domaine d'un poste de travail appartenant au domaine maître. Les caractères génériques sont acceptés.

Remarque : Vous devez toujours indiquer le nom de domaine lors de la suppression d'un lien d'un poste de travail n'appartenant pas au domaine maître.

Cet argument est utile lors de la suppression des liaisons de plusieurs postes de travail dans un domaine. Par exemple, pour déconnecter tous les agents du domaine *stlouis*, utilisez la commande suivante :

```
unlink stlouis!@
```

Si vous ne précisez pas *domaine* et que *poste_de_travail* contient des caractères génériques, le domaine par défaut est celui dans lequel s'exécute **conman**.

poste de travail

Indique le nom du poste de travail dont la liaison doit être supprimée. Les caractères génériques sont acceptés.

Cette commande n'est pas prise en charge sur les postes de travail de moteur distant.

noask Indique que l'utilisateur ne doit pas être invité à confirmer avant d'effectuer une opération sur chaque poste de travail éligible.

Commentaires

En supposant qu'un utilisateur possède un accès **unlink** aux postes de travail dont la liaison est en cours de suppression, les règles suivantes s'appliquent :

- Un utilisateur exécutant **conman** sur le gestionnaire de domaine maître peut supprimer la liaison de n'importe quel poste de travail du réseau.
- Un utilisateur exécutant **conman** sur un autre gestionnaire que le gestionnaire de domaine maître peut supprimer la liaison de n'importe quel poste de travail de son domaine et des domaines subordonnés. L'utilisateur ne peut pas supprimer la liaison des postes de travail dans les domaines homologues.
- Un utilisateur exécutant **conman** sur un agent peut supprimer la liaison de n'importe quel poste de travail dans son domaine local à condition que le poste de travail soit un gestionnaire de domaine ou un hôte. Un agent homologue appartenant au même domaine ne peut pas être privé de liaisons.

Pour des informations supplémentaires, voir «link», à la page 468.

Exemples

La figure 29 et le tableau 71 présentent les liaisons fermées par les commandes **unlink** exécutées par les utilisateurs en différents points du réseau.

DM_n sont des gestionnaires de domaine et **A_{mn}** sont des agents.

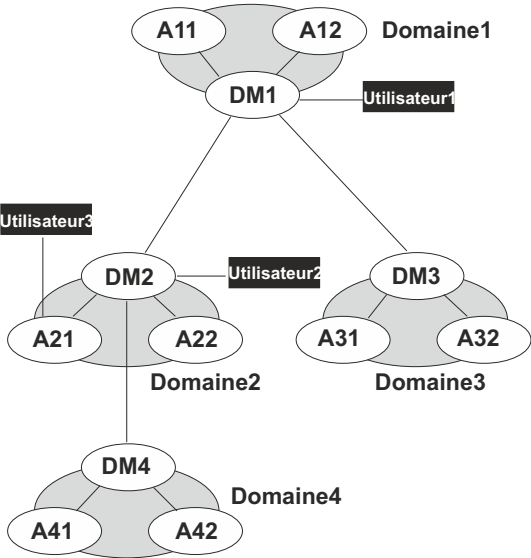


Figure 29. Postes de travail réseau non liés

Tableau 71. Postes de travail non liés

Commande	Fermeture par Utilisateur1	Fermeture par Utilisateur2	Fermeture par Utilisateur3
unlink @!@	Toutes les liaisons sont fermées	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4 DM4-A41 DM4-A42	DM2-A21
unlink @	DM1-A11 DM1-A12 DM1-DM2 DM1-DM3	DM1-DM2 DM2-A21 DM2-A22 DM2-DM4	DM2-A21
unlink DOMAIN3!@	DM3-A31 DM3-A32	Non autorisé	Non autorisé
unlink DOMAIN4!@	DM4-A41 DM4-A42	DM4-A41 DM4-A42	Non autorisé
unlink DM2	DM1-DM2	Non applicable	DM2-A21
unlink A42	DM4-A42	DM4-A42	Non autorisé
unlink A31	DM3-A31	Non autorisé	Non autorisé

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle qui suit :

1. Dans la barre de navigation en haut, cliquez sur **Statut et état de santé du système > Surveillance de la charge de travail > Surveiller la charge de travail**.
2. Sélectionnez un moteur.
3. Dans **Type d'objet**, sélectionnez **Poste de travail**.
4. Dans la liste déroulante **Requête**, sélectionnez une requête pour surveiller les postes de travail.
5. Cliquez sur **Exécuter** pour exécuter la tâche de surveillance.
6. A partir de la table contenant la liste des postes de travail, sélectionnez un poste de travail et cliquez sur **Supprimer le lien**.

version

affiche la bannière de programme **conman**, avec la version allant jusqu'au niveau du groupe de correctifs installé.

Syntaxe

{version | v}

Exemples

Pour afficher la bannière du programme **conman**, exécutez la commande suivante :

```
%version
```

La sortie est similaire à ce qui suit :

```
#####
# Eléments sous licence - Propriété d'IBM* et HCL**
# 5698-WSH
# (C) Copyright IBM Corp. 1998, 2016 All rights reserved.
# (C) Copyright HCL Technologies Ltd. 2016 All rights reserved
# * Trademark of International Business Machines
# ** Trademark of HCL Technologies Limited
#####
Installed for user "twuser".
Variable locale LANG définie sur : "fr"
Scheduled for (Exp) 05/20/16 (#8) on LB001542_MASTER.Batchman LIVES.
Limit:55,Fence:0,Audit Level:0
```

Chapitre 13. Utilisation de statistiques avancées pour prévoir la durée estimée d'un travail

Un outil statistique puissant pour la prévision des durées de travaux prévues, qui vient s'ajouter à celui qui est fourni par la commande logman, est disponible lorsque vous installez un sous-ensemble du logiciel IBM SPSS Statistics fourni avec IBM Workload Scheduler. Cet outil n'utilise pas de série temporelle spécifique pour calculer les durées de travail estimées, comme le fait logman. Au contraire, il utilise un algorithme très sophistiqué sur l'ensemble de l'historique des quatre-vingt-dix jours précédents pour prévoir les durées estimées pour les cinq jours suivants. Les prévisions sont précises à la seconde près.

Tandis que logman est conçu pour fournir des évaluations précises lorsque la charge de travail est sujette à des décalages périodiques, l'outil statistique avancé est idéal dans le cas de modèles plus complexes. Par exemple, l'outil peut être particulièrement utile pour voir au-delà des incidences acceptées des événements cycliques déjà connus, et comprendre quels sont les conflits apparemment masqués qui peuvent affecter la durée d'un travail. Il peut être pertinent de mesurer et de prévoir les durées des travaux le long d'un chemin critique qui ne respecte pas son échéance occasionnellement.

La commande logman consigne les statistiques de travaux depuis un fichier journal de plan de production. Par défaut, les statistiques sont consignées automatiquement pour tous les travaux du plan. Au contraire, l'outil statistique reposant sur SPSS traite seulement les travaux que vous avez précédemment marqués à cet effet.

Déploiement de l'outil statistique avancé

Pour utiliser l'outil statistique basé sur SPSS sur des travaux sélectionnés afin d'obtenir une prévision à cinq jours de leur durée estimée, procédez comme suit :

1. «Installation du sous-ensemble SPSS Statistics». Installez le sous-ensemble SPSS fourni avec IBM Workload Scheduler sur un agent de votre choix
2. Marquez les travaux qui doivent être traités par l'outil statistique avancé
3. Configurez et importez en tant que flot de travaux le modèle d'application de charge de travail ELAB_JOB_STAT
4. Exécutez le flot de travaux ELAB_JOB_STAT_JS après que le plan a fini de calculer les durées de travail prévues

Installation du sous-ensemble SPSS Statistics

Installez le sous-ensemble SPSS Statistics sur un agent de votre réseau IBM Workload Scheduler. Pour installer le sous-ensemble SPSS Statistics, procédez comme suit :

1. Téléchargez ou copiez le fichier .zip de l'outil de statistiques avancé de IBM Workload Scheduler sur l'agent à partir de l'image d'IBM Workload Scheduler, en fonction du système d'exploitation exécuté sur l'agent.
2. Décompressez le fichier.
3. Exécutez :
 - `./iws_spss_install.sh -rép_install chemin d'installation (UNIX)`

- `unité\iws_spss_install.bat -rép_install chemin d'installation`
(Windows)

où *chemin d'installation* est un chemin d'installation que vous indiquez sur l'agent. Vous devrez indiquer ce chemin d'installation ultérieurement dans l'un des travaux qui composent le flot de travaux ELAB_JOB_STAT_JS. Si vous exécutez `iws_spss_install` sans arguments, vous obtenez la syntaxe en retour.

L'installation est complète. Sur la plateforme AIX, une étape supplémentaire est nécessaire pour terminer l'installation.

Avertissement : Pour plateformes AIX uniquement. Cette étape de configuration supplémentaire est nécessaire pour terminer l'installation.

Si vous procédez à l'installation sur une plateforme AIX, vous devez modifier le fichier `JobManager.ini` comme suit pour terminer l'installation :

1. Localisez le fichier `JobManager.ini` sur l'instance d'agent local où s'exécutera le travail SPSS. Ce fichier se trouve dans le répertoire `TWA_home/TWS/ITA/cpa/config` sur l'agent.
2. Dans la section [ITA Env] du fichier, modifiez la valeur du mot clé `LIBPATH` en ajoutant le texte suivant affiché en gras. L'intégralité de la valeur doit être entrée sur une seule ligne continue. Cet exemple suppose que SPSS a été installé dans le dossier `/opt/IBM/SPSS` :

```
[ITA Env]
SHLIB_PATH = $SHLIB_PATH:.;
LIBPATH = $LIBPATH:/home/spss_075/TWS/JavaExt
/jre/jre/bin/j9vm:/home/spss_075/
TWS/JavaExt/jre/jre/bin/j9vm/../../lib/ppc64
:opt/IBM/SPSS:/opt/IBM/SPSS/ext/
bin/pasw.adp:/opt/IBM/SPSS/ext/bin/pasw.aedata:/opt/IBM/SPSS/ext/bin/pasw.alm:/
opt/IBM/SPSS/ext/bin/pasw.bagging:/opt/IBM/SPSS
/ext/bin/pasw.boosting:/opt/IBM/
SPSS/ext/bin/pasw.cognos:/opt/IBM/SPSS/ext/bin
/pasw.common:/opt/IBM/SPSS/ext/
bin/pasw.externalrule:/opt/IBM/SPSS/ext/bin
/pasw.glm:/opt/IBM/SPSS/ext/bin/
pasw.me:/opt/IBM/SPSS/ext/bin/pasw.netezaindb:/opt/IBM/SPSS/ext/bin/
pasw.neuralnet:/opt/IBM/SPSS/ext/bin/pasw.outerpartition:/opt/IBM/SPSS/ext/
bin/pasw.pmm1merge:/opt/IBM/SPSS/ext/bin/pasw.psm:/opt/IBM/SPSS/ext/bin/
pasw.scoring:/opt/IBM/SPSS/ext/bin/pasw.simulation:/opt/IBM/SPSS/ext/bin/
pasw.split:/opt/IBM/SPSS/ext/bin/pasw.stb:/opt/IBM
/SPSS/ext/bin/pasw.tml:/
opt/IBM/SPSS/ext/bin/pasw.transformation:/opt/IBM
/SPSS/ext/bin/pasw.tree:/opt/
IBM/SPSS/ext/bin/pasw.xmldata:/opt/IBM/SPSS/ext
/bin/spss.C5:/opt/IBM/SPSS/ext/
bin/spss.bayesiannetwork:/opt/IBM/SPSS/ext/bin
/spss.binning:/opt/IBM/SPSS/ext/
bin/spss.cplex:/opt/IBM/SPSS/ext/bin/spss.dataview:/opt/IBM/SPSS/ext/bin/
spss.inlinecsp:/opt/IBM/SPSS/ext/bin/spss.knn:/opt/IBM/SPSS/ext/bin/
spss.modelaccreditation:/opt/IBM/SPSS/ext/bin/
spss.modelevaluation:/opt/IBM/SPSS/
ext/bin/spss.optimization:/opt/IBM/SPSS/ext/bin
/spss.predictoreffectiveness:/opt/
IBM/SPSS/ext/bin/spss.predictorstat:/opt/IBM/SPSS
/ext/bin/spss.propensitymodelling:/
opt/IBM/SPSS/ext/bin/spss.selflearning:/opt/IBM
/SPSS/ext/bin/spss.svm:/opt/IBM/SPSS/
ext/bin/spss.xd:/opt/IBM/SPSS/jre/bin/classic:/opt/IBM/SPSS/jre/lib/ppc64:/opt/IBM/
SPSS/jre/lib/ppc64/j9vm:/opt/IBM/SPSS/jre/bin
LD_LIBRARY_PATH = $LD_LIBRARY_PATH:.
```

3. Sauvegardez les modifications dans le fichier.

Sélection des travaux devant être mesurés par l'outil de statistiques avancées

Pour sélectionner les travaux pour lesquels la durée estimée doit être prévue par l'outil basé sur SPSS, vous pouvez utiliser la ligne de commande Composer ou Dynamic Workload Console. Dans les deux cas, vous mettez à jour la définition du travail à l'intérieur de la définition du flot de travaux.

Dans Composer :

1. Ouvrez la définition de flot de travaux qui contient les travaux que vous souhaitez marquer.
2. Marquez les travaux en ajoutant le mot clé `statisticstype custom` à leur définition.

Dans Dynamic Workload Console :

1. Dans Gestion des définitions de charge de travail, modifiez la définition de flot de travaux qui contient les travaux que vous souhaitez marquer.
2. Dans la définition de flot de travaux, sélectionnez chaque travail que vous voulez marquer pour ouvrir sa définition.
3. Dans le panneau Propriétés du travail, sélectionnez l'onglet Prévission de durée.
4. Dans Prévission de durée, sélectionnez la case Utiliser l'analyse avancée.

Vous pouvez modifier votre sélection à tout moment concernant votre charge de travail. La durée estimée des travaux qui ne sont pas marqués pour les statistiques avancées est calculée par la commande `logman`.

Bien que vous n'ayez pas besoin d'effectuer d'étapes supplémentaires pour que les durées estimées des travaux soient calculées par `logman`, si vous souhaitez qu'elles soient calculées par l'outil de statistiques avancées, vous devez également créer et personnaliser le flot de travaux `ELAB_JOB_STAT_JS` comme décrit ci-après.

Les durées estimées des travaux marqués ne sont toutefois calculées par l'outil de statistiques avancées basé sur SPSS que lorsque vous exécutez le flot de travaux `ELAB_JOB_STAT_JS` après avoir exécuté le plan (`JnextPlan`).

Importation et configuration du flot de travaux `ELAB_JOB_STAT_JS`

Le modèle d'application de charge de travail (WAT) `ELAB_JOB_STAT` est fourni en tant que fichier zip avec le code d'installation du sous-ensemble logiciel SPSS. Vous devez extraire le modèle d'application de charge de travail du fichier zip et l'importer dans le gestionnaire de domaine maître de votre réseau en tant que flot de travaux. Le flot de travaux `ELAB_JOB_STAT_JS` est composé de trois travaux que vous devez configurer avant de pouvoir l'exécuter une fois que le plan est terminé.

`ELAB_JOB_STAT_JS` doit être lancé suite à l'exécution du plan. Sa fonction consiste à :

1. rechercher dans le plan tous les travaux marqués, obtenir leur historique de durées (couvrant jusqu'à 90 jours) et les copier dans le fichier `exportjobstat.zip` ;

2. exporter exportjobstat.zip dans l'outil statistique basé sur SPSS (qui calcule les durées estimées des travaux marqués pour les 5 jours suivants et les copie dans le fichier importjobstat.zip) ;
3. importer importjobstat.zip et rendre les durées estimées (et leurs intervalles de confiance) disponibles dans conman et dans Dynamic Workload Console.

Pour créer et personnaliser le flot de travaux ELAB_JOB_STAT_JS, procédez comme suit :

1. Pour créer le flot de travaux ELAB_JOB_STAT_JS :
 - a. Recherchez le fichier custom_stats_wat.zip dans le même support que celui d'où vous avez extrait le fichier d'installation du sous-ensemble logiciel SPSS.
 - b. Téléchargez ce fichier, décompressez-le et recherchez le fichier ELAB_JOB_STAT_WAT_Mapping.UTF8.properties.
 - c. Editez ELAB_JOB_STAT_WAT_Mapping.UTF8.properties de façon à remplacer la valeur de la clé WORKSTATION_ELAB_JOB_STAT_WS par le nom du poste de travail d'agent qui doit exécuter le flot de travaux ELAB_JOB_STAT.
 - d. Utilisez la commande suivante pour importer le fichier WAT modifié dans le gestionnaire de domaine maître:

```
wappman -import ELAB_JOB_STAT_WAT_Definitions.UTF8.xml ELAB_JOB_STAT_WAT_Mapping.UTF8.properties
```

Pour plus d'informations sur les modèles d'application de charge de travail, voir «Définition des applications de charge de travail», à la page 304.

Le flot de travaux ELAB_JOB_STAT_JS est alors créé. Il contient les travaux suivants que vous devez personnaliser :

- EXPORT_JOB_STAT
- ELAB_JOB_STAT
- IMPORT_JOB_STAT

Le nom du poste de travail qui doit exécuter le flot de travaux est celui que vous avez fourni lorsque vous avez édité le fichier de propriétés du modèle d'application de charge de travail.

2. Pour personnaliser le flot de travaux ELAB_JOB_STAT_JS :
 - Ouvrez EXPORT_JOB_STAT et IMPORT_JOB_STAT pour les éditer. Ces travaux sont des travaux de services Web RESTful. Dans le composeur, éditez la définition JSDL. Dans Dynamic Workload Console, cliquez avec le bouton droit de la souris sur les travaux dans la définition du flot de travaux, puis sélectionnez Ouvrir la définition de travail :
 - Dans Authentification, entrez le nom d'utilisateur et le mot de passe de la connexion
 - Dans Actions, entrez dans le nom de hôte et le port du gestionnaire de domaine maître dans la zone URI du service
 - Ouvrez ELAB_JOB_STAT pour l'éditer. Le type de ce travail est SPSS Job Duration Prediction. Dans le composeur, éditez la définition JSDL. Dans Dynamic Workload Console, cliquez avec le bouton droit de la souris sur les travaux dans la définition du flot de travaux, puis sélectionnez Ouvrir la définition de travail :
 - Dans SPSS Job Duration Prediction, entrez le chemin d'installation du sous-ensemble logiciel SPSS dans la zone Répertoire d'installation

- Dans SPSS Job Duration Prediction, indiquez les chemins d'accès des fichiers exportjobstat.zip et importjobstat.zip. Les répertoires doivent déjà exister avant l'exécution du flot de travaux.

Exécution du flot de travaux ELAB_JOB_STAT_JS et affichage des résultats

Soumettez le flot de travaux ELAB_JOB_STAT_JS une fois que JnextPlan s'est exécuté. En option, planifiez le flot de travaux pour s'exécuter régulièrement après JnextPlan (après le travail FINAL, s'il existe).

Les durées estimées et les intervalles de confiance associés peuvent être visualisés, comme c'est le cas avec les valeurs calculées par la commande logman, comme suit :

Ligne de commande conman

En tant que sortie de :

```
showjobs nom_travail;props
```

Dynamic Workload Console

Dans l'une des vues suivantes :

- Surveillance des travaux
- Propriétés de travail
- Chemin critique
- Analyse par hypothèses

Dépannage du flot de travaux ELAB_JOB_STAT_JS

Si le travail ELAB_JOB_STAT se termine anormalement lors de l'exécution et l'erreur suivante est enregistrée dans le journal de travail :

Echec de la connexion au serveur 'Serveur local'

procédez comme suit pour corriger ce problème :

1. Arrêtez l'agent sur lequel le travail s'exécute.
2. Editez le fichier JobManager.ini situé dans le répertoire *rep_base_TWA/TWS/ITA/cpa/config* de l'agent.

Ajoutez la chaîne suivante aux mots clés SHLIB_PATH, LIBPATH et

LD_LIBRARY_PATH situés dans la section [ITA] du fichier :

```
:rep_base_SPSS/clemrtl/cspj/bin:rep_base_SPSS:rep_base_SPSS/bin:rep_ase_SPSS/lib:
```

où *rep_base_SPSS* est le répertoire absolu où le sous-ensemble SPSS Statistics est installé (par exemple, /opt/IBM/SPSS/ModelerSolutionPublisher/16).

3. Redémarrez l'agent et soumettez à nouveau le travail ELAB_JOB_STAT ou le flot de travaux ELAB_JOB_STAT_JS.

Chapitre 14. Extension des fonctions IBM Workload Scheduler

Vous pouvez étendre les fonctions d'IBM Workload Scheduler en l'intégrant à des produits IBM ou tiers, tels que IBM Sterling Connect:Direct ou MS SQL. Cette intégration vous permet de lancer facilement des travaux IBM Workload Scheduler sur des produits externes tout en utilisant les fonctions de planification de IBM Workload Scheduler. IBM Workload Scheduler fournit également des travaux qui effectuent des opérations quotidiennes, telles que le transfert de fichiers et les services Web, ainsi que les travaux d'utilitaire qui automatisent et simplifient les opérations telles que la mise à jour centralisée de l'agent.

L'intégration se compose d'un certain nombre de types de travaux avec options avancées disponibles dans Dynamic Workload Console et de la commande **composer**.

De plus, vous pouvez créer des plug-in personnalisés pour implémenter vos propres types de travaux avec options avancées pour les applications qui ne sont pas prises en charge par IBM Workload Scheduler. Pour plus d'informations sur la création de plug-in personnalisés, reportez-vous à *Extension d'IBM Workload Automation*.

Les travaux IBM Workload Scheduler standard sont des fichiers exécutables génériques, des programmes ou des commandes. Vous pouvez définir des travaux pour exécuter des tâches spécifiques, telles que l'appel des fournisseurs OSLC, l'exécution des transferts de fichiers et l'exécution de fichiers sur des systèmes distants où aucun composant IBM Workload Scheduler n'est installé à l'aide des types de travaux avec options avancées. Vous pouvez définir ces travaux simplement, sans avoir de connaissances particulières sur les applications qui les exécutent.

Pour plus d'informations sur la définition des travaux IBM Workload Scheduler standard, voir «Définition de travail», à la page 187.

Les types de travaux avec options avancées suivants sont disponibles :

Tableau 72. Types de travail avec options avancées

Catégorie	Type de travail	Description
Natif	Windows	Travaux exécutés sur les systèmes d'exploitation Windows.
	UNIX	Travaux exécutés sur les plateformes UNIX. Travaux exécutés sur des agent tolérant aux pannes limités pour IBM i.
	Autre	Travaux exécutés sur des agents étendus. Pour plus d'informations sur les types de tâches personnalisées pour les applications acquises chez un fournisseur et prises en charge, voir Planification des applications avec IBM Workload Automation.
	z/OS	Travaux qui exécutent la commande indiquée dans l'onglet JCL sur un système JCL.
	Commande distante	Travaux exécutés sur des ordinateurs distants sur lesquels aucune installation d'agent IBM Workload Scheduler n'est installée. Remarque : Sur les systèmes z/OS, les travaux sont créés à l'aide de Dynamic Workload Console.
	IBM i	Travaux qui exécutent des commandes sur des systèmes IBM i.
	Exécutable	Travaux qui exécutent des scripts ou des commandes avec des options avancées, telles que le réacheminement des entrées et sorties standard vers un fichier.

Tableau 72. Types de travail avec options avancées (suite)

Catégorie	Type de travail	Description
ERP	Travail SAP sur les postes de travail d'agent étendu	Travaux exécutés sur un agent étendu SAP. Les types de définitions de travaux SAP R/3 suivants sont concernés : • Travail R/3 standard • Travail de chaîne de processus BW • Travail InfoPackage BW Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Travail SAP sur les postes de travail dynamiques	Travaux exécutés sur des postes de travail d'agent dynamique, sur des pools, sur des pools dynamiques et sur des agents z-centric. Les types suivants de définition de travail SAP sont disponibles : • Travail R/3 standard • Travail de chaîne de processus BW • Travail InfoPackage BW Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Méthode d'accès	Travaux qui étendent les fonctions de planification de IBM Workload Scheduler à d'autres systèmes et applications à l'aide des méthodes d'accès. Ces méthodes communiquent avec le système externe pour lancer le travail et renvoyer son statut. Les méthodes d'accès disponibles sont les suivantes : • PeopleSoft • SAP • z/OS • Unixssh • Méthodes personnalisées Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	SAP PI Channel	Travaux qui exécutent des travaux des canaux SAP Process Integration (PI) pour contrôler les canaux de communication entre l'intégrateur de processus et un système dorsal SAP R/3. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	SAP BusinessObjects Business Intelligence (BI)	Travaux qui activent l'automatisation, la surveillance et le contrôle des flots de travaux contenant des rapports SAP BusinessObjects BI (rapports Crystal et Webi). Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Oracle E-Business Suite	Travaux qui activent l'automatisation, la surveillance et le contrôle des flots de travaux contenant des travaux Oracle E-Business Suite . Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
Cloud	Workload Broker	Travaux qui gèrent le cycle de vie d'un travail Dynamic Workload Broker. Pour plus d'informations sur l'utilisation de Dynamic Workload Broker, voir IBM Workload Scheduler - Planification dynamique de la charge de travail.
	TPM (Provisioning)	Travaux qui concernent des ordinateurs physiques, des machines virtuelles et des environnements cloud privés et publics qui créent un environnement à la demande. Ce type de travail s'intègre à IBM SmartCloud Provisioning.
	Salesforce	Travaux qui activent l'automatisation, la surveillance et le contrôle des données et des activités Salesforce. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.

Tableau 72. Types de travail avec options avancées (suite)

Catégorie	Type de travail	Description
Transfert de fichier et coordination	Reflet distribué	Travaux exécutés en local et mappant d'autres travaux exécutés dans des environnements distribués IBM Workload Scheduler distants.
	Reflet z/OS	Travaux exécutés en local et mappage d'autres travaux dans un environnement IBM Workload Scheduler for z/OS distant.
	Transfert de fichier	Travaux qui exécutent des programmes pour transférer des fichiers vers et à partir d'un serveur accessible via FTP, SSH ou d'autres protocoles.
	IBM Sterling Connect:Direct	Travaux qui exécutent des programmes IBM Sterling Connect:Direct pour transférer un ou plusieurs fichiers d'un noeud principal à un noeud secondaire. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	système de fichiers distribué Hadoop	Travaux qui définissent, planifient, surveillent et gèrent les programmes de transfert de fichier entre votre station de travail et le serveur système de fichiers distribué Hadoop. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
OSLC	OSLC Automation	Travaux qui appellent n'importe quel fournisseur OSLC qui implémente la spécification d'automatisation OSLC. Les ressources d'automatisation définissent des plans d'automatisation, des requêtes d'automatisation et des résultats d'automatisation du cycle de vie de développement, de test et de déploiement du logiciel.
	OSLCTPM (Provisioning)	Travaux qui appellent n'importe quel fournisseur OSLC, tel IBM Workload Scheduler et IBM SmartCloud Orchestrator, qui implémente la spécification OSLC TPM (Provisioning). Les ressources mises à disposition définissent des plans de mise à disposition, des requêtes de mise à disposition et des résultats de mise à disposition du cycle de vie de développement, de test et de déploiement du logiciel.
Base de données et intégrations	Base de données	Travaux qui réalisent des requêtes, des instructions SQL et des travaux sur un certain nombre de bases de données, y compris des bases de données personnalisées. Vous pouvez également créer et exécuter des procédures stockées sur des bases de données DB2, Oracle, Microsoft SQL Server, Netezza, Hive et BigSql .
	IBM Cloudant	Travaux qui exécutent des actions sur la base de données IBM Cloudant, ses documents ou ses pièces jointes.
	MS SQL	Travaux qui exécutent un travail Microsoft SQL Server.
	IBM WebSphere MQ	Travaux qui activent des communications parmi les applications qui fonctionnent dans des environnements distribués différents à des heures différentes. Les communications reposent sur les modèles d'échange de message suivants : • Demande/Réponse. • Publier sur des files d'attente ou des rubriques. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Services Web	Travaux qui exécutent un service Web.
	Services Web RESTful	Travaux qui envoient des demandes via des méthodes HTTP (PUT, POST, GET, HEAD, DELETE) à des ressources Web.
	Java	Travaux qui exécutent une classe Java.
	J2EE	Travaux qui permettent à des applications Java dans un même réseau d'envoyer et de recevoir des messages à destination et en provenance d'une file d'attente JMS.
	JSR 352 Java Batch	Travaux qui permettent d'exécuter des applications Java par lots qui implémentent la spécification de programmation de norme JSR 352.
	MQTT	Travaux qui permettent d'exécuter les actions publish et subscribe sur les rubriques gérées par un courtier de messages MQTT.

Tableau 72. Types de travail avec options avancées (suite)

Catégorie	Type de travail	Description
Analyse métier	InfoSphere DataStage	Travaux qui exécutent des travaux IBM InfoSphere DataStage. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Rapports IBM Cognos	Travaux qui exécutent des rapports IBM Cognos, des rapports interactif, des requêtes et des vues de rapport. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Informatica PowerCenter	Travaux qui planifient des flots de travaux Informatica PowerCenter et effectuent le suivi de leurs résultats depuis Dynamic Workload Console et la ligne de commande IBM Workload Scheduler. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Hadoop Map Reduce	Travaux qui définissent, planifient, surveillent et gèrent l'exécution des fichiers .jar Hadoop Map Reduce. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
	Apache Oozie	Travaux qui définissent, planifient, surveillent et gèrent l'exécution des flots de travaux Apache Oozie et des travaux Hadoop suivants : • MapReduce, • Pig, • Hive, • Sqoop. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation
	IBM BigInsights	Travaux qui définissent, planifient, surveillent et gèrent des feuilles ou des applications IBM BigInsights Workbook. Pour plus d'informations, voir Planification des applications avec IBM Workload Automation.
Utilitaires d'automatisation Travaux qui facilitent des opérations IBM Workload Scheduler spécifiques.	mise à jour centralisée d'agent	Travaux permettant de planifier la mise à jour centralisée de plusieurs instances d'agent. Pour plus d'informations, voir IBM Workload Scheduler - Planification et installation.
	Table de variables	Travaux qui ajoutent ou modifient une variable dans une table de variables spécifique. Les travaux Table de variables permettent la transmission de variables d'une tâche à l'autre, que ce soit dans un même flot de travaux ou dans un flot de travaux distinct. Pour plus d'informations, voir IBM Workload Scheduler - Guide d'utilisation et de référence.
	Gestion des travaux	Travaux qui exécutent des actions sur un travail dans un flot de travaux. Pour plus d'informations, voir IBM Workload Scheduler - Guide d'utilisation et de référence.
	Soumission d'un flot de travaux	Travaux qui soumettent un flot de travaux en vue de leur traitement. Pour plus d'informations, voir IBM Workload Scheduler - Guide d'utilisation et de référence.
Les méthodes d'accès et plug-in d'application nécessaires pour exécuter les travaux répertoriés ci-dessus sont livrés avec IBM Workload Scheduler. Certains de ces éléments nécessitent l'achat d'un composant facturable distinct en plus d'IBM Workload Scheduler pour être autorisé à les utiliser. Consultez le tableau répertoriant les méthodes d'accès et les plug-in d'application IBM Workload Scheduler facturables dans le <i>Guide d'administration</i> pour obtenir la liste des composants facturables qui doivent être achetés séparément. Remarque : Pour plus d'informations sur la configuration logicielle, voir Document relatif à la configuration requise.		

Vous ne pouvez exécuter les types de travaux avec options avancées que sur les postes de travail avec fonctions dynamiques, c'est-à-dire, les agents dynamiques, les pools et les pools dynamiques. Ces types de poste de travail utilisent les fonctions dynamiques intégrées à IBM Workload Scheduler et offrent la possibilité d'associer de façon dynamique votre charge de travail soumise (ou une partie de cette charge) aux meilleures ressources disponibles lors de l'exécution. Pour plus

d'informations sur la planification dynamique, voir Chapitre 15, «Gestion des fonctions de planification dynamique dans votre environnement», à la page 667.

Procédure préalable de création de types de travaux avec options avancées

Cette rubrique explique comment définir une nouvelle définition de travail via Dynamic Workload Console.

Exécutez les étapes suivantes avant de définir et de planifier les types de travaux avec options avancées.

1. Installez un certain nombre d'agents dynamiques et ajoutez l'environnement d'exécution Java

Pour installer des agents dynamiques, exécutez le programme d'installation. Vous pouvez installer l'agents dynamiques au cours de l'installation complète de IBM Workload Scheduler ou lors d'une installation autonome de l'agent. Lors de l'installation, vous pouvez choisir d'ajouter l'environnement d'exécution Java pour exécuter les types de travaux avec options avancées ; ces types sont fournis avec le produit et les types supplémentaires sont implémentés via les plug-in personnalisés.

Suivez l'assistant d'installation pour terminer l'installation.

Consultez la section relative aux options d'installation dans *Guide de planification et d'installation* pour la description des paramètres et options d'installation.

2. Organisez les agents dynamiques en pools et pools dynamiques.

Les pools et pools dynamiques vous aident à organiser l'environnement en fonction de la disponibilité de vos postes de travail et des exigences des travaux que vous prévoyez d'exécuter.

- a. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de l'environnement de charge de travail > Création de postes de travail**.
- b. Sélectionnez un moteur.
- c. Sélectionnez le type de poste de travail à créer.
 - Pour créer un pool, indiquez les agents dynamiques que vous souhaitez ajouter au pool et le poste de travail Workload broker où est hébergé le pool.
 - Pour créer un pool dynamique, spécifiez les exigences que chaque agent dynamique doit satisfaire pour être ajouté à ce pool dynamique.

3. Octroyez l'autorisation requise pour définir les types de travaux avec options avancées.

L'administrateur IBM Workload Scheduler doit accorder des autorisations spécifiques dans le fichier de sécurité pour permettre aux opérateurs de créer des types de travaux avec options avancées.

- a. Accédez au répertoire *TWA_home/TWS* à partir duquel les commandes **dumpsec** et **makesec** doivent être exécutées.
- b. Exécutez la commande **dumpsec** pour déchiffrer le fichier de sécurité actuel dans un fichier de configuration éditables.

Pour plus d'informations, consultez la section relative à la commande **dumpsec** dans *Guide d'administration*.
- c. Ajoutez l'affichage et l'accès pour exécution au poste de travail comme suit :

- Si l'opération est effectuée sur le Connecteur IBM Workload Scheduler, l'affichage et l'accès pour utilisation est requis sur l'unité centrale correspondant au poste de travail sur lequel le travail est créé.
- Si l'opération est effectuée sur le poste de travail sur lequel le travail s'exécute, l'accès pour affichage est requis sur le poste de travail du courtier de charge de travail.

Pour plus d'informations, consultez la section relative à la configuration d'un fichier de sécurité dans *Guide d'administration*.

- d. Fermez toutes les interfaces utilisateur **conman** ouvertes à l'aide de la commande **exit**.
 - e. Arrêtez tous les connecteurs sur les systèmes exécutant des systèmes d'exploitation Windows.
 - f. Exécutez la commande **makesec** pour chiffrer le fichier de sécurité et appliquer les modifications.
Pour plus d'informations, consultez la section relative à la commande **makesec** dans *Guide d'administration*.
 - g. Si vous utilisez la sécurité locale, le fichier est immédiatement disponible sur le poste de travail où il a été mis à jour.
 - 1) Si vous utilisez un gestionnaire de domaine maître de sauvegarde, copiez-y le fichier.
 - 2) Distribuez manuellement le fichier centralisé vers tous les agents tolérants par défaut du réseau (pas les agents standard, étendus ou de courtier) et stockez-le dans le répertoire *rep_base_TWA/TWS*.
 - 3) Exécutez **JnextPlan** pour distribuer le fichier Symphony correspondant au nouveau fichier de sécurité.
4. Définissez les types de travaux avec options avancées (voir «Création de définitions de travaux avancés»).

Création de définitions de travaux avancés

A partir de la ligne de commande **composer**, vous pouvez créer à la fois des travaux standard et des types de travaux avec options avancées. La syntaxe pour la création des deux types de travail est semblable ; la seule différence se situe dans les arguments que vous utilisez pour définir le travail à exécuter.

Pour définir des types de travaux standard, utilisez les arguments **docommand** ou **scriptname** ; pour définir des types de travaux avec options avancées, utilisez l'argument **task**, comme décrit dans «Définition de travail», à la page 187. Chaque type de travail avec options avancées a des attributs spécifiques qui sont décrits en détail dans les sections suivantes.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans :

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la création et l'édition d'objets de planification, voir :

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Définition de travail - Travaux z/OS

Un travail z/OS exécute la commande que vous spécifiez dans l'onglet JCL d'un système JCL. Ce type de travail ne s'exécute que sur IBM Workload Scheduler réparti - Agent pour z/OS.

Cette section décrit les attributs obligatoires et facultatifs des travaux z/OS. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 73. Attributs obligatoires et facultatifs pour la définition d'un travail z/OS.

Attribut	Description/valeur	Obligatoire
application name	jcl	✓
byDefinition	Type de soumission de travail. Il s'agit de l'unique type de soumission pris en charge.	
jclDefinition	Opération à effectuer sur le système JCL.	✓

L'exemple suivant présente un travail qui renvoie l'état du système JCL :

```
ZOSAGENT#JCLDEF
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsd1=="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsd1"
  xmlns:jsd1jcl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsd1jcl">
  <jsd1:application name="jcl">
    <jsd1jcl:jcl>
      <jsd1jcl:JCLParameters>
        <jsd1jcl:jcl>
          <jsd1jcl:byRefOrByDef>
            <jsd1jcl:byDefinition>
              <jsd1jcl:jclDefinition>//NORMAL JOB , 'TWS JOB', CLASS=A, MSGCLASS=A,>
              // MSGLEVEL=(1,1)
            /*
            //STEP1 EXEC PGM=IEFBR14</jsd1jcl:jclDefinition>
          </jsd1jcl:byDefinition>
        </jsd1jcl:byRefOrByDef>
      </jsd1jcl:jcl>
    </jsd1jcl:JCLParameters>
    <jsd1jcl:JOBParameters>
      <jsd1jcl:jobStreamName>${tws.jobstream.name}jsd1jcl:jobStreamName>${tws.jobstream.name}>
      <jsd1jcl:inputArrival>${tws.job.ia}jsd1jcl:inputArrival>${tws.job.ia}>
    </jsd1jcl:JOBParameters>
  </jsd1jcl:jcl>
</jsd1:application>
</jsd1:jobDefinition>
  DESCRIPTION "Sample JCL Job Definition"
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux de type commande distante

Un travail de type commande distante s'exécute sur les ordinateurs distants sur lesquels l'installation de l'agent IBM Workload Scheduler n'est pas obligatoire.

Pour lire un scénario commun basé sur la réalité qui atteint les objectifs métier, y compris l'implémentation d'un travail de type commande distante, voir Scénarios cloud - Gestion de la charge de travail dans des environnements dynamiques.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Remarque : Sur les systèmes Windows, le module d'extension RemoteCommand a un délai d'attente codé en dur et établi à 5 minutes (300 secondes). Il se peut que ce délai d'attente soit atteint lorsqu'une tâche est en cours d'exécution, causant une fermeture anormale. Pour éviter cela, un nouveau fichier de propriétés RemoteCommandExecutor.properties a été ajouté au plug-in avec un attribut **timeout** permettant de définir un nombre de secondes différent pour donner plus de temps pour exécuter la tâche. Le format de l'attribut est le suivant : **timeout=sec**, où *sec* correspond à la durée en secondes. Redémarrez l'agent pour activer cette modification.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type commande distante. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 74. Attributs obligatoires et facultatifs pour la définition d'un travail de type commande distante

Attribut	Description/valeur	Obligatoire
application name	remotecommand	✓
userName	Nom d'utilisateur autorisé à établir une connexion sur l'ordinateur distant à l'aide du protocole défini. Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom_utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <pre><jsdl:password>\${password:nom_utilisateur}</jsdl:password></pre> - Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom d'utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. - Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <pre><jsdl:password>\${password:poste_travail#nom_util}</jsdl:password></pre> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <pre><jsdl:userName>\${agent:util}</jsdl:userName> <jsdl:password>\${agent:password.util}</jsdl:password></pre>	✓
password	Mot de passe de l'utilisateur autorisé. Le mot de passe est chiffré lorsque le travail est créé. Voir la description de userName pour plus de détails.	
serverName	Nom d'hôte de l'ordinateur où s'exécute l'instance commande distante.	✓
port	Numéro de port de l'ordinateur distant où la commande s'exécute.	✓

Tableau 74. Attributs obligatoires et facultatifs pour la définition d'un travail de type commande distante (suite)

Attribut	Description/valeur	Obligatoire
protocole	Valeurs possibles : AUTO Le protocole est automatiquement sélectionné à partir des protocoles existants : SSH, Windows, RSH et REXEC. Le produit tente d'utiliser d'abord le protocole SSH. Si ce protocole échoue, il utilise le protocole Windows. Lors de l'utilisation de Secure Shell (SSH), le chemin d'accès doit être au format SSH. Dans ce cas, le serveur ssh Cygwin est monté sur /home/Administrator. SSH Un protocole réseau qui fournit des fonctions d'accès aux fichiers, de transfert et de gestion de fichiers sur n'importe quel flux de données. WINDOWS Protocole de partage de fichiers de Microsoft. Le port par défaut utilisé est 445. Il doit exister au moins un partage samba sur le serveur, quelle que soit la commande à exécuter. RSH Le protocole RSH (Remote Shell Protocol) est un protocole qui permet à un utilisateur d'exécuter des commandes sur un système distant sans devoir se connecter au système. REXEC Le serveur REXEC (Remote Execution) est une application TCP/IP (Transmission Control Protocol/Internet Protocol) qui permet à un client de soumettre des commandes système à un système distant. Le protocole REXEC (Remote Execution Protocol) autorise le traitement de ces commandes ou programmes sur n'importe quel hôte du réseau. L'hôte local reçoit ensuite les résultats du traitement des commandes.	
keystoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion. Un fichier de clés est une base de données de clés. Les clés privées présentes dans un fichier de clés sont associées à une chaîne de certificats qui authentifie la clé publique correspondante sur le serveur distant. Un fichier de clés contient également les certificats provenant des entités de confiance. Concerne le protocole SSH uniquement.	
keystorePassword	Mot de passe qui protège la clé privée et qui est requis pour établir la connexion. Cet attribut est obligatoire uniquement si vous indiquez un chemin d'accès au fichier de clés. Si le chemin d'accès au fichier de clés et la combinaison fichier de clés/mot de passe ne parviennent pas à établir une connexion, une tentative est effectuée à l'aide du nom d'utilisateur et du mot de passe associés à l'utilisateur autorisé à établir une connexion sur l'ordinateur distant.	✓
command	Entrez la commande à soumettre sur l'ordinateur distant.	✓
environment	Sortie standard et fichiers d'erreur standard pour la commande distante. Ces fichiers se trouvent sur l'agent, et pas localement sur les postes de travail où s'exécute la commande distante. Vérifiez que vous disposez des droits en écriture sur les répertoires indiqués, faute de quoi aucun fichier ne sera créé. Sortie standard Chemin et nom de fichier dans lequel la sortie standard de la commande doit être sauvegardée. Indiquez un nom de chemin absolu ou un nom de chemin relatif pour accéder au répertoire de travail. Le fichier est remplacé chaque fois que la commande génère une nouvelle sortie. Erreur standard Chemin et nom de fichier dans lequel l'erreur standard de la commande doit être sauvegardée. Indiquez un nom de chemin absolu ou un nom de chemin relatif pour accéder au répertoire de travail. Le fichier est remplacé chaque fois que la commande génère une nouvelle erreur.	

L'exemple ci-dessous illustre la section JSDL "application" d'un modèle de définition de travail pour un travail de type commande distante :

```

$JOBS
NC112016#REMCMD
TASK
  <?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"

```

```

xmlns:jsdlremotecommand="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlremotecommand" name="REMOTECOMMAND">
<jsdl:application name="remotecommand">
  <jsdlremotecommand:remotecommand>
    <jsdlremotecommand:RemoteCommandParameters>
      <jsdlremotecommand:taskPanel>
        <jsdlremotecommand:command>ping -c 10 localhost </jsdlremotecommand:command>
      </jsdlremotecommand:taskPanel>
      <jsdlremotecommand:environmentPanel>
        <jsdlremotecommand:standardOutput>stdout</jsdlremotecommand:standardOutput>
        <jsdlremotecommand:standardError>stderr</jsdlremotecommand:standardError>
      </jsdlremotecommand:environmentPanel>
      <jsdlremotecommand:serverPanel>
        <jsdlremotecommand:serverInfo>
          <jsdlremotecommand:serverName>9.168.112.16</jsdlremotecommand:serverName>
          <jsdlremotecommand:port>23</jsdlremotecommand:port>
          <jsdlremotecommand:protocol>ssh</jsdlremotecommand:protocol>
        </jsdlremotecommand:serverInfo>
        <jsdlremotecommand:credentials>
          <jsdl:user>userName</jsdl:user>
          <jsdl:password>{aes}mv0GJq0HWo81buhcpFa1uL9RkGQKrYvTiAUpKTMgp90=
          </jsdl:password>
        </jsdlremotecommand:credentials>
        <jsdlremotecommand:certificates>
          <jsdlremotecommand:keystoreFilePath>/var/keyStoreFile</jsdlremotecommand:
            keystoreFilePath>
          <jsdlremotecommand:keystorePassword>pwd</jsdlremotecommand:keystorePassword>
        </jsdlremotecommand:certificates>
      </jsdlremotecommand:serverPanel>
    </jsdlremotecommand:RemoteCommandParameters>
  </jsdlremotecommand:remotecommand>
</jsdl:application>
</jsdl:jobDefinition>
RECOVERY STOP

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux IBM i

Un travail IBM i exécute la commande que vous spécifiez sur un système IBM i (anciennement connu sous le nom d'AS/400 et d'i5 OS).

La présente section décrit les attributs obligatoires et facultatifs des travaux de type IBM i. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 75. Attributs obligatoires et facultatifs pour la définition d'un travail IBM i.

Attribut	Description/valeur	Obligatoire
application name	ibmi	✓

Tableau 75. Attributs obligatoires et facultatifs pour la définition d'un travail IBM i. (suite)

Attribut	Description/valeur	Obligatoire
otherCommandType	La commande générique à exécuter sur le système IBM i.	Obligatoire pour la commande de type générique
SBMJOBType	La commande SBJJOB à exécuter sur le système IBM i.	Requis pour une commande de type SBJJOB
jobName	Nom du travail associé au travail soumis pendant son traitement par le système.	
jobDescription	Description du travail utilisé pour soumettre des travaux à des fins de traitement par lots.	
jobDescriptionLibrary	Bibliothèque qui qualifie la description du travail.	
jobQueue	Nom qualifié de la file d'attente de travaux dans laquelle le travail est placé.	
jobQueueLibrary	Bibliothèque qui qualifie la file d'attente de travaux.	
jobPriority	Priorité de planification pour le travail soumis.	
outputPriority	Priorité de sortie des fichier spoule produits par le travail soumis.	
outputQueue	Nom qualifié de la file d'attente de sortie utilisée pour les fichiers spoule.	
outputQueueLibrary	Bibliothèque qui qualifie la file d'attente de sortie.	
printDevice	Nom qualifié du périphérique d'imprimante par défaut pour le travail soumis.	
systemLibraryList	Portion système de la liste de bibliothèques initiales utilisée par le travail soumis.	
currentLibrary	Nom de la bibliothèque en cours associée au travail soumis.	
initialLibraryList	Composant de l'utilisateur initial de la liste des bibliothèques utilisée pour rechercher des noms d'objet qui ont été spécifiés sans qualificateur de bibliothèque.	

Tableau 75. Attributs obligatoires et facultatifs pour la définition d'un travail IBM i. (suite)

Attribut	Description/valeur	Obligatoire
msgReplyList	Liste de messages pour lesquels vous voulez définir une réponse automatisée. Pour chaque message, spécifiez : msgReply msgId Identificateur du message. msgCmpDta Le texte du message. msgRpy Réponse automatisée que vous souhaitez définir.	

Pour plus d'informations relatives à la définition des paramètres de la commande de **soumission du travail** (SBMJOB), voir la documentation du produit IBM i.

Voici un exemple de travail qui émet une commande SBJOB avec les paramètres connexes :

```

$JOBS
IBMI72_94#IBMI_NEWDEF_TEST
TASK _
  <?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
xmlns:jSDLibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDLibmi" name="ibmi">
  <jSDL:application name="ibmi">
    <jSDLibmi:ibmi>
      <jSDLibmi:IBMIParameters>
        <jSDLibmi:Task>
          <jSDLibmi:command>WRKSYSSTS</jSDLibmi:command>
          <jSDLibmi:commandTypeGroup>
            <jSDLibmi:SBJOBType>
              <jSDLibmi:jobName>TESTMEL</jSDLibmi:jobName>
              <jSDLibmi:jobDescription>QDFTJOB</jSDLibmi:jobDescription>
              <jSDLibmi:jobDescriptionLibrary>QGPL</jSDLibmi:jobDescriptionLibrary>
              <jSDLibmi:jobQueue/>
              <jSDLibmi:jobQueueLibrary/>
              <jSDLibmi:jobPriority>3</jSDLibmi:jobPriority>
              <jSDLibmi:outputPriority>4</jSDLibmi:outputPriority>
              <jSDLibmi:outputQueue>*DEV</jSDLibmi:outputQueue>
              <jSDLibmi:outputQueueLibrary/>
              <jSDLibmi:printDevice>PRT01</jSDLibmi:printDevice>
              <jSDLibmi:systemLibraryList/>
              <jSDLibmi:currentLibrary>*CRTDFT</jSDLibmi:currentLibrary>
              <jSDLibmi:initialLibraryList>QGPL QTEMP QDEVELOP
              QBLDSYS</jSDLibmi:initialLibraryList>
            </jSDLibmi:SBJOBType>
          </jSDLibmi:commandTypeGroup>
        </jSDLibmi:Task>
      </jSDLibmi:IBMIParameters>
    </jSDLibmi:ibmi>
  </jSDL:application>
</jSDL:jobDefinition>
RECOVERY STOP

```

L'exemple suivant présente un travail qui exécute une commande sur un système IBM i et définit une réponse automatisée pour un message : Pour plus d'informations relatives à la définition d'une réponse automatisée pour un message, voir «Planification des travaux sur les systèmes IBM i», à la page 863.

```

$JOBS
AGTIBMI_MEL#IBMI_MSG_REPLY
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
    xmlns:jSdLibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSdLibmi" name="ibmi">
    <jSDL:application name="ibmi">
      <jSdLibmi:ibmi>
        <jSdLibmi:IBMIParameters>
          <jSdLibmi:Task>
            <jSdLibmi:command>CALL PGM(MICKLIB/SENDMSG)</jSdLibmi:command>
            <jSdLibmi:commandTypeGroup>
              <jSdLibmi:otherCommandType/>
            </jSdLibmi:commandTypeGroup>
            <jSdLibmi:msgReplyList>
              <jSdLibmi:msgReply>
                <jSdLibmi:msgId>CPA2401</jSdLibmi:msgId>
                <jSdLibmi:msgCmpDta>NECESSARY LIFE MONTHLY CUTOFF TODAY,CONFIRM(Y/N)?</jSdLibmi:msgCmpDta>
                <jSdLibmi:msgRpy>y</jSdLibmi:msgRpy>
              </jSdLibmi:msgReply>
            </jSdLibmi:msgReplyList>
          </jSdLibmi:Task>
        </jSdLibmi:IBMIParameters>
      </jSdLibmi:ibmi>
    </jSDL:application>
  </jSDL:jobDefinition>
  RECOVERY STOP

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux exécutables

Un travail exécutable exécute des scripts ou des commandes avec des options avancées, telles que le réacheminement des entrées et sorties standard vers un fichier.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs des travaux exécutables. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 76. Attributs obligatoires et facultatifs pour la définition d'un travail exécutable.

Attribut	Description/valeur	Obligatoire
nom d'application	exécutable	✓
interactive	Indique si le travail requiert une intervention de l'utilisateur. Cette option s'applique uniquement aux travaux qui s'exécutent sur les systèmes d'exploitation Windows.	✓

Tableau 76. Attributs obligatoires et facultatifs pour la définition d'un travail exécutable. (suite)

Attribut	Description/valeur	Obligatoire
value	Indiquez le nom et la valeur d'un ou plusieurs arguments.	
script	Entrez un script à exécuter par le travail. Le script est créé et exécuté lors de l'exécution du travail. Vous pouvez indiquer les arguments dans cette balise ou les entrer dans la balise value et les appeler dans le script.	✓
suffix	Indiquez l'extension de nom de fichier du script à exécuter par le travail. Cette option s'applique uniquement aux travaux qui s'exécutent sur les systèmes d'exploitation Windows. N'incluez pas "." au début du nom d'extension.	

L'exemple suivant présente un travail qui exécute une commande ping sur deux sites Web. L'adresse des sites Web est définie dans la balise **value** et est appelée dans la balise **script**. Ce travail possède une relation d'affinité avec le travail `affine_test`, ce qui signifie qu'il s'exécute sur le même poste de travail qu'`affine_test` :

```
$JOBS
AGENT#EXECUTABLE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
  <jsd1:application name="executable">
    <jsdle:executable interactive="false" workingDirectory="c:\">
      <jsdle:arguments>
        <jsdle:value>www.mysite.com</jsdle:value>
        <jsdle:value>www.yoursite.com</jsdle:value>
      </jsdle:arguments>
      <jsdle:script>ping %1 ping %2</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Defined using composer."
TWSAFFINITY "affine_test"
RECOVERY STOP
```

L'exemple suivant présente un travail qui exécute un script vbs sur les systèmes d'exploitation Windows. L'extension du nom de fichier est définie dans l'attribut **suffix** de la balise **script** :

```
WIN_WKS1#VBS_NAT1
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
  <jsd1:application name="executable">
    <jsdle:executable interactive="true" workingDirectory="c:\tws">
      <jsdle:script suffix="vbs">Wscript.Echo "ciao"</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
RECOVERY STOP
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux de la méthode d'accès

Les travaux de méthode d'accès étendent les fonctions de planification de IBM Workload Scheduler à d'autres systèmes et applications utilisant des méthodes d'accès. Ces méthodes communiquent avec le système externe pour lancer le travail et renvoyer son statut.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs des travaux de la méthode d'accès. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 77. Attributs obligatoires et facultatifs pour la définition d'un travail de la méthode d'accès.

Attribut	Description/valeur	Obligatoire
application name	xajob	✓
accessMethod	Nom de la méthode d'accès utilisée pour communiquer avec le système externe afin de lancer le travail et de renvoyer l'état de ce travail.	✓
target	Nom d'un fichier d'options.	
taskString	Commande devant être interprétée par la méthode sélectionnée. La longueur de ligne maximale est de 8 Ko.	✓

Tableau 77. Attributs obligatoires et facultatifs pour la définition d'un travail de la méthode d'accès. (suite)

Attribut	Description/valeur	Obligatoire
droits d'accès	Le nom et le mot de passe de l'utilisateur qui exécute ce travail. Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom_utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <pre><jsdl:password>\${password:nom_utilisateur}</jsdl:password></pre> - Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. - Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <pre><jsdl:password>\${password:poste_travail#nom_util}</jsdl:password></pre> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). Si vous avez défini le nom d'utilisateur avec la variable <i>user</i> et un mot de passe, les instructions correspondantes de données d'identification se présenteront ainsi : <pre><jsdl:userName>\${agent:utilisateur}</jsdl:userName> <jsdl:password>\${agent:password.utilisateur}</jsdl:password></pre> - Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645.	

L'exemple suivant illustre un travail qui crée un fichier dans le dossier /methods à l'aide d'un travail de méthode d'accès par défaut :

```
$JOBS
AGENT#XA_JOB
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:jsdlxa=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlxa" name="xajob">
  <jsdl:application name="xajob">
    <jsdlxa:xajob accessMethod="unixloc1" target="optionFile">
      <jsdlxa:taskString>touch file</jsdlxa:taskString>
      <jsdlxa:credentials>
        <jsdlxa:userName>TestUser</jsdlxa:userName>
        <jsdlxa:password>{aes}IEr/DES8wRzQEij1ySQBfUR587QBxM0iwfQ1EWJaDds=</jsdlxa:password>
      </jsdlxa:credentials>
    </jsdlxa:xajob>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Defined using composer."
RECOVERY STOP
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Procédure préalable de création de travaux TPM (Provisioning)

Comment créer une nouvelle définition de travail TPM (Provisioning) à l'aide du Dynamic Workload Console.

Pour créer une définition de travail TPM (Provisioning), vous devez d'abord compléter la procédure préalable ci-dessous .

1. Installez IBM SmartCloud Provisioning. Pour déterminer la version prise en charge par le produit, générez le rapport d'Intégration de données à partir du site Web d'IBM Software Product Compatibility Reports et sélectionnez l'onglet **Supported Software**.
2. Obtenez le certificat du serveur SmartCloud HTTP et enregistrez-le dans un répertoire pour une utilisation ultérieure. L'administrateur TPM (Provisioning) peut fournir le certificat, ou vous pouvez extraire ce dernier en effectuant la procédure suivante dans votre navigateur. L'exemple suivant utilise Mozilla Firefox :
 - a. Connectez-vous au serveur TPM (Provisioning) en fournissant les données d'identification TPM (Provisioning).
 - b. Pour télécharger le certificat, cliquez sur :
Outils>Options>Avancé>Chiffrement>Afficher les certificats
 - c. Sélectionnez **IBM>IBM WebSphere Cloudburst Appliance**, puis cliquez sur **Exporter**. Un fichier, appelé **IBMWebSphereCloudBurstAppliance.crt** (Certificate X509 - PEM) est créé.
3. Accédez au répertoire où un environnement d'exécution Java est installé, par exemple : `C:\Program Files\IBM\TWS\javaExt\JRE\JRE`
4. Créez un magasin en exécutant la commande suivante : `keytool -genkeypair -alias certificatekey -keyalg RSA -validity 7 -keystore keystore.jks`, où, *keystore.jks* est le chemin d'accès au fichier de clés.
5. Ajoutez un certificat IBM SmartCloud au magasin en exécutant la commande suivante : `keytool -import -file repertoire_certificat\IBMWebSphereCloudBurstAppliance.crt -alias scp -keystore repertoire_magasin\keystore.jks`,
6. Ouvrez le fichier `REP_BASE_TWA\TWS\ITA\cpa\config\JobManager.ini`, et localisez la section `JavaJobLauncher`, ligne `JVMOptions`.
7. Ajoutez les instructions suivantes à la ligne : `"-Djavax.net.ssl.trustStore= DIRECTORY_TRUSTSTORE/keystore.jks -Djavax.net.ssl.trustStorePassword=MOT_DE_PASSE_TRUSTSTORE"` Par exemple :
`JVMOptions = -Djavax.net.ssl.trustStore=C:/myUtils/keystore.jks -Djavax.net.ssl.trustStorePassword=passwd`
8. Pour terminer la procédure, arrêtez et redémarrez l'agent.

Travaux IBM SmartCloud Provisioning

Un travail TPM (Provisioning) interagit avec IBM SmartCloud Provisioning pour gérer des ressources dans un environnement de Cloud Computing ; vous pouvez gérer des ressources, créer et gérer des configuration dans le cloud, et déployer des images virtuelles à l'aide de modèles et de scripts.

Pour consulter un scénario basé sur la réalité qui montre la façon dont IBM Workload Scheduler et IBM SmartCloud Provisioning peuvent vous aider à atteindre vos objectifs métier, voir Gestion des charges de travail dans des environnements dynamiques.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Pour plus d'informations sur TPM (Provisioning), voir le centre de documentation d'IBM SmartCloud Provisioning

La présente section décrit les attributs obligatoires et facultatifs des travaux de type TPM (Provisioning). Chaque définition de travail utilise le format et les arguments suivants :

Tableau 78. Attributs obligatoires et facultatifs pour la définition d'un travail de type TPM (Provisioning)

Attribut	Description et valeur	Obligatoire
application name	application des accès	✓
actionType	Valeurs valides : deploy Déployez une image virtuelle dans le groupe de clouds et créez une nouvelle instance de système virtuel qui contient le nombre d'instances d'images virtuelles que vous spécifiez. manage Démarrez, arrêtez ou supprimez des ressources virtuelles (par exemple, des machines virtuelles ou des images virtuelles d'un environnement complexe).	✓
cloudGroupId	Identificateur unique du groupe Cloud à partir duquel vous choisissez l'instance virtuelle à déployer.	✓
instanceId	Identificateur unique de l'instance virtuelle existante sur laquelle vous agissez. Une instance virtuelle peut être une ou plusieurs machines virtuelles. Si vous voulez agir uniquement sur une seule machine virtuelle qui appartient à l'instance actuelle, indiquez son ID dans l'attribut virtualMachineId .	✓
virtualMachineId	Identificateur unique de la machine virtuelle existante sur laquelle vous agissez.	
VirtualImageId	Identificateur unique de l'image virtuelle existante qui est utilisée comme modèle pour une nouvelle image virtuelle que vous déployez.	✓
instanceNameDeploy	Nom unique de l'instance virtuelle.	✓
numberOfVirtualMachines	Nombre de machines virtuelles que vous déployez.	✓
description	Chaîne de texte qui décrit l'instance de système virtuel.	
tags	Matrice de balises définies par l'utilisateur. Ces données utilisateur indiquées ici peuvent être extraites à l'intérieur de la machine virtuelle. Elles peuvent être utilisées pour configurer une machine virtuelle au premier amorçage ou pour exécuter un script d'amorçage enregistré dans la machine virtuelle.	
Size	Taille de l'instance. Il s'agit de la taille de chaque machine virtuelle unique appartenant à l'instance. Valeurs valides : • xsmall : très petite • small : petite • medium : moyenne • large : grande • xlarge : très grande	✓
winPassword	Mot de passe administrateur permettant d'accéder aux système Windows déployés.	
unixSSHPublicKey	Applicable uniquement à UNIX. La clé SSH doit être fournie par l'administrateur TPM (Provisioning).	

Tableau 78. Attributs obligatoires et facultatifs pour la définition d'un travail de type TPM (Provisioning) (suite)

Attribut	Description et valeur	Obligatoire
userName mot de passe	Données d'identification associées au serveur TPM (Provisioning). Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom_utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <code><jsd1:password>\${password:nom_utilisateur}</jsd1:password></code> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <code><jsd1:password>\${password:poste_travail#nom_util}</jsd1:password></code> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <code><jsd1:userName>\${agent:util}</jsd1:userName></code> <code><jsd1:password>\${agent:password.util}</jsd1:password></code> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645.	✓
hostname	Nom d'hôte du serveur TPM (Provisioning).	✓
port	Numéro de port du serveur d'TPM (Provisioning). La valeur par défaut est 443.	✓
manageType	Valeurs valides : actionStart : démarre l'instance du système virtuel. actionStop : arrête l'instance du système virtuel. actionDelete : supprime l'instance du système virtuel.	✓

L'exemple suivant présente une définition de travail à utiliser pour déployer une machine virtuelle dans l'environnement TPM (Provisioning) :

```
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsd1="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsd1" xmlns:
  jsd1provisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsd1provisioning"
  name="PROVISIONING">
  <jsd1:application name="provisioning">
    <jsd1provisioning:provisioning>
      <jsd1provisioning:ProvisioningParameters>
        <jsd1provisioning:actions>
          <jsd1provisioning:actionType>
            <jsd1provisioning:deploy>
              <jsd1provisioning:groupCloud>
                <jsd1provisioning:cloudGroupId>8</jsd1provisioning:cloudGroupId>
              </jsd1provisioning:groupCloud>
              <jsd1provisioning:groupVirtualImage>
                <jsd1provisioning:virtualImageId>52</jsd1provisioning:virtualImageId>
              </jsd1provisioning:groupVirtualImage>
              <jsd1provisioning:instanceNameDeploy>TESTPROP</jsd1provisioning:instanceNameDeploy>
              <jsd1provisioning:numberOfVirtualMachines>1</jsd1provisioning:numberOfVirtualMachines>
              <jsd1provisioning:description/>
            </jsd1provisioning:tags/>
            <jsd1provisioning:size>xsmall</jsd1provisioning:size>
            <jsd1provisioning:winPassword/>
            <jsd1provisioning:unixSSHPublicKey/>
          </jsd1provisioning:actionType>
        </jsd1provisioning:actions>
      </jsd1provisioning:ProvisioningParameters>
    </jsd1provisioning:provisioning>
  </jsd1:application>
</jsd1:jobDefinition>
```



```

        </jsdlprovisioning:deploy>
      </jsdlprovisioning:actionType>
    </jsdlprovisioning:actions>
  <jsdlprovisioning:connectionInfo>
    <jsdlprovisioning:credentials>
      <jsdl:userName>cbadmin</jsdl:userName>
      <jsdl:password>{aes}2WfJH/3a0xyX2f+QXew+1YnrN2tM4z338QMY1YVgp0A=</jsdl:password>
    </jsdlprovisioning:credentials>
    <jsdlprovisioning:server>
      <jsdlprovisioning:hostname>9.168.58.192</jsdlprovisioning:hostname>
      <jsdlprovisioning:port>443</jsdlprovisioning:port>
    </jsdlprovisioning:server>
  </jsdlprovisioning:connectionInfo>
</jsdlprovisioning:ProvisioningParameters>
</jsdlprovisioning:provisioning>
</jsdl:application>
</jsdl:jobDefinition>

```

L'exemple suivant présente une définition de travail à utiliser pour arrêter une machine virtuelle :

```

<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
  jsdlprovisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlprovisioning"
  name="PROVISIONING">
  <jsdl:application name="provisioning">
    <jsdlprovisioning:provisioning>
      <jsdlprovisioning:ProvisioningParameters>
        <jsdlprovisioning:actions>
          <jsdlprovisioning:actionType>
            <jsdlprovisioning:manage>
              <jsdlprovisioning:manageType>actionStop</jsdlprovisioning:manageType>
              <jsdlprovisioning:instanceId>102</jsdlprovisioning:instanceId>
              <jsdlprovisioning:virtualMachineId/>
            </jsdlprovisioning:manage>
          </jsdlprovisioning:actionType>
        </jsdlprovisioning:actions>
        <jsdlprovisioning:connectionInfo>
          <jsdlprovisioning:credentials>
            <jsdl:userName>cbadmin</jsdl:userName>
            <jsdl:password>{aes}2WfJH/3a0xyX2f+QXew+1YnrN2tM4z338QMY1YVgp0A=</jsdl:password>
          </jsdlprovisioning:credentials>
          <jsdlprovisioning:server>
            <jsdlprovisioning:hostname>9.168.58.192</jsdlprovisioning:hostname>
            <jsdlprovisioning:port>443</jsdlprovisioning:port>
          </jsdlprovisioning:server>
        </jsdlprovisioning:connectionInfo>
      </jsdlprovisioning:ProvisioningParameters>
    </jsdlprovisioning:provisioning>
  </jsdl:application>
</jsdl:jobDefinition>

```

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux IBM SmartCloud Provisioning en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire à l'aide de la commande **kill**. Cependant, cette action est pertinente uniquement sur le travail IBM Workload Scheduler, et elle n'affecte pas le ou les processus s'exécutant sur le poste de travail IBM SmartCloud Provisioning. Lorsque vous arrêtez le travail IBM Workload Scheduler, IBM Workload Scheduler affecte le statut **Error** ou **ABEND** au travail IBM Workload Scheduler, quel que soit le statut du travail IBM SmartCloud Provisioning.

Surveillance du travail

Si l'IBM Workload Scheduler s'arrête lorsque vous soumettez le travail TPM (Provisioning) ou pendant que le travail est en cours d'exécution, IBM Workload Scheduler lance la surveillance du travail là où il s'était arrêté dès que l'agent redémarre.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux reflet

Un travail reflet est un travail défini sur le poste de travail en local et qui est utilisé pour mapper un autre travail s'exécutant sur un poste de travail distant, appelé travail distant. Vous pouvez utiliser des travaux reflet pour intégrer la charge de travail qui s'exécute sur des moteurs différents, lesquels peuvent être des moteurs IBM Workload Scheduler for z/OS ou IBM Workload Scheduler.

Les travaux reflet sont définis à l'aide de la syntaxe XML. Les attributs de clé permettant d'identifier l'instance de travail distant et les critères de correspondance associés dépendent du type de moteur distant sur lequel l'instance de travail distant est définie. Les zones en gras sont celles utilisées pour identifier l'instance de travail distant.

Etant donné que les moteurs z/OS prennent uniquement en charge les critères de correspondance précédents les plus proches, le modèle XML permettant de définir un travail reflet z/OS est le suivant :

```
$JOBS
WORKSTATION#ZSHADOW_CLOS_PRES
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jSDL:jobDefinition
    xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
    xmlns:zshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/zshadow">
    <jSDL:application name="zShadowJob">
      <zshadow:ZShadowJob>
        <zshadow:JobStream>JobStream</zshadow:JobStream>
        <zshadow:JobNumber>JobNumber</zshadow:JobNumber>
        <zshadow:matching>
          <zshadow:previous>
        </zshadow:matching>
      </zshadow:ZShadowJob>
    </jSDL:application>
  </jSDL:jobDefinition>
  DESCRIPTION "Sample Job Definition"
  RECOVERY STOP
```

Remarque : Assurez-vous d'entrer des paramètres valides dans les zones **JobStream** et **JobNumber**.

Les travaux reflet distribués prennent en charge les quatre critères de correspondance disponibles pour les dépendances externes.

Vous trouverez ci-dessous les modèles XML que vous pouvez utiliser pour définir les travaux reflet distribués :

Critères de correspondance : précédents les plus proches

Modèle XML :

```
$JOBS
WORKSTATION#DSHADOW_CLOS_PRES
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jsd1:jobDefinition
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
    xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsd1:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:previous/>
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
  </j>
DESCRIPTION "Sample Job Definition"
RECOVERY STOP
```

Critères de correspondance : dans un intervalle absolu

Modèle XML :

```
$JOBS
WORKSTATION#DSHADOW_ABSOLUTE
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jsd1:jobDefinition
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
    xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsd1:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:absolute from="0600 -4" to="1100 +3"/>
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
  </j>
DESCRIPTION "Sample Job Definition"
RECOVERY STOP
```

Critères de correspondance : dans un intervalle relatif

```
$JOBS
WORKSTATION#DSHADOW_RELATIVE
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jsd1:jobDefinition
    xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
    xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
    <jsd1:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
    <dshadow:JobStream>JobStream</dshadow:JobStream>
    <dshadow:Workstation>Workstation</dshadow:Workstation>
    <dshadow:Job>Job</dshadow:Job>
    <dshadow:matching>
    <dshadow:relative from="-400" to="+500" />
    </dshadow:matching>
    </dshadow:DistributedShadowJob>
    </jsdl:application>
    </jsdl:jobDefinition>
  </j>
DESCRIPTION "Sample Job Definition"
RECOVERY STOP
```

Critères de correspondance : même date planifiée

Modèle XML :

```
$JOBS
WORKSTATION#DSHADOW_SAMEDAY
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition
  xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
  xmlns:dshadow="http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow">
  <jSDL:application name="distributedShadowJob">
  <dshadow:DistributedShadowJob>
  <dshadow:JobStream>JobStream</dshadow:JobStream>
  <dshadow:Workstation>Workstation</dshadow:Workstation>
  <dshadow:Job>Job</dshadow:Job>
  <dshadow:matching>
  <dshadow:sameDay/>
  </dshadow:matching>
  </dshadow:DistributedShadowJob>
  </jSDL:application>
  </jSDL:jobDefinition>
DESCRIPTION "Sample Job Definition"
RECOVERY STOP
```

Pour plus d'informations sur les critères de correspondance, voir «Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux», à la page 71.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux de transfert de fichiers

Un travail de transfert de fichier exécute des programmes pour transférer des fichiers vers et à partir d'un serveur accessible via FTP, SSH ou d'autres protocoles.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et les attributs facultatifs des travaux de transfert de fichier. Seuls les transferts de fichier unique sont autorisés. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 79. Attributs obligatoires et facultatifs pour la définition d'un travail de transfert de fichier

Attribut	Description et valeur	Obligatoire
application name	filetransfer	✓

Tableau 79. Attributs obligatoires et facultatifs pour la définition d'un travail de transfert de fichier (suite)

Attribut	Description et valeur	Obligatoire
Type de transfert de fichier (importation ou téléchargement)	Indique si le transfert de fichier est un transfert ou un téléchargement. Enserrez tous les attributs de transfert de fichier de balises <code>jsdlfiletransfer:uploadInfo</code> ou <code>jsdlfiletransfer:downloadInfo</code> comme indiqué dans l'exemple.	✓
Permissions (notation octale)	En cas de téléchargement, indiquez des droits d'accès aux fichiers pour l'utilisateur sur le système local. Des droits d'accès aux fichiers sont exprimés sous forme de notation octale.	
Supprimer les fichiers source après transfert	En cas de chargement, indiquez si les fichiers source doivent être supprimés après le transfert.	
serveur	L'adresse du serveur de transfert de fichier (et du numéro de port, s'il est différent du port standard, lorsque vous choisissez FTP comme protocole).	✓
localfile et remotefile	- Si vous effectuez une importation, <code>localfile</code> est le nom et le chemin qualifié complet du fichier à importer, tandis que <code>remotefile</code> est le nom et le chemin qualifié complet du fichier à créer sur la cible distante. Vous pouvez utiliser des astérisques (*) ou des points d'interrogation (?) comme caractères génériques dans <code>localfile</code>. - Si vous effectuez un téléchargement, <code>localfile</code> est le nom et le chemin qualifié complet du fichier à créer sur la cible locale, tandis que <code>remotefile</code> est le nom et le chemin qualifié complet du fichier à télécharger. Vous pouvez utiliser des astérisques (*) ou des points d'interrogation (?) comme caractères génériques dans <code>remotefile</code>.	✓
localCredentials et remoteCredentials	Noms et mots de passe des utilisateurs autorisés sur les systèmes locaux et distants. Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom d'utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <code><jsdl:password>\${password:nom_utilisateur}</jsdl:password></code> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <code><jsdl:password>\${password:poste_travail#nom_util}</jsdl:password></code> - Saisissez un utilisateur et un mot de passe définis avec la commande <code>param</code> localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <code><jsdl:userName>\${agent:util}</jsdl:userName></code> <code><jsdl:password>\${agent:password.util}</jsdl:password></code> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645. Conseil : Le mot de passe de l'utilisateur distant n'est pas requis si un chemin d'accès et un mot de passe de fichier de clés sont indiqués lorsque le protocole SSH est utilisé.	✓
keystoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion. Un fichier de clés est une base de données de clés. Les clés privées présentes dans un fichier de clés sont associées à une chaîne de certificats qui authentifie la clé publique correspondante sur le serveur distant. Un fichier de clés contient également les certificats provenant des entités de confiance. Concerne le protocole SSH uniquement.	

Tableau 79. Attributs obligatoires et facultatifs pour la définition d'un travail de transfert de fichier (suite)

Attribut	Description et valeur	Obligatoire
keystorePassword	Mot de passe qui protège la clé privée et qui est requis pour établir la connexion. Cet attribut est obligatoire uniquement si vous indiquez un chemin d'accès au fichier de clés. Si le chemin d'accès au fichier de clés et la combinaison fichier de clés/mot de passe ne parviennent pas à établir une connexion, une tentative est effectuée à l'aide du nom d'utilisateur et du mot de passe associés à l'utilisateur autorisé à établir une connexion sur l'ordinateur distant.	✓
protocole	peut être : WINDOWS Protocole de partage de fichiers de Microsoft. Si vous ne précisez pas de protocole, le système suppose WINDOWS. Spécifiez le répertoire partagé dans le mot clé <code>remoteFile</code>, sans indiquer de chemin où est imbriqué le répertoire partagé. Indiquez l'adresse du poste de travail hébergeant le répertoire partagé dans le mot clé <code>server</code>. Le répertoire de travail par défaut est : <code>C:\Program Files\IBM\TWA\TWS\ITA</code>. SSH Un protocole réseau qui fournit des fonctions d'accès au fichier, de transfert de fichier et de gestion de fichier sur un flot de données fiable. Si vous ne précisez pas de protocole, SSH est supposé par défaut. Pour accéder aux postes de travail Windows distants, installez Cygwin ou MKS sur chaque poste de travail Windows cible. FTP Un protocole réseau standard utilisé pour échanger des fichiers sur un réseau basé sur TCP/IP, tel qu'Internet. FTPS Extension du protocole FTP qui ajoute la prise en charge du protocole cryptographique TLS. Plus précisément, le transfert de fichier s'effectue implicitement à l'aide du protocole TLS pour les sessions FTP, fournissant une clé de sécurité privée pour la connexion de données. La version 1 du protocole TLS est prise en charge. La configuration de réutilisation de session SSL n'est pas prise en charge. FTPES Extension du protocole FTP qui ajoute la prise en charge du protocole cryptographique TLS. Plus précisément, le transfert de fichier s'effectue explicitement à l'aide du protocole TLS pour les sessions FTP fournissant une clé de sécurité privée pour la connexion de données. La version 1 du protocole TLS est prise en charge. La configuration de réutilisation de session SSL n'est pas prise en charge. AUTO Le protocole est sélectionné automatiquement entre les protocoles Windows et SSH. Le produit essaie d'abord d'utiliser le protocole WINDOWS. En cas d'échec, il utilise le protocole SSH. Lors de l'utilisation de Secure Shell (SSH), le chemin d'accès doit être au format SSH. Dans ce cas, le serveur ssh Cygwin est monté sur <code>/home/Administrator</code>.	
transferMode	Peut être <code>binary</code> (valeur par défaut) ou <code>ascii</code> .	
remoteCodepage	Page de codes utilisée sur le poste de travail distant. Si vous souhaitez utiliser une page de codes personnalisée, définissez le paramètre <code>remoteCodepage</code> comme suit : <pre><jsdlfiletransfer:remoteCodepage>USER:CUSTOM_CP </jsdlfiletransfer:remoteCodepage></pre> Où <code>CUSTOM_CP</code> est la page de codes définie par l'utilisateur. Par exemple, pour utiliser la page de codes personnalisée <code>tcpip.ftpd.ftpx1bin.frence3</code>, définissez le paramètre <code>remoteCodepage</code> comme suit : <pre><jsdlfiletransfer:remoteCodepage>USER:tcpip.ftpd.ftpx1bin.frence3 </jsdlfiletransfer:remoteCodepage></pre>	Obligatoire si vous spécifiez <code>localCodepage</code>
localCodepage	Page de codes utilisée sur le poste de travail local.	Obligatoire si vous spécifiez <code>remoteCodepage</code>

Tableau 79. Attributs obligatoires et facultatifs pour la définition d'un travail de transfert de fichier (suite)

Attribut	Description et valeur	Obligatoire
Délai d'attente	Indique le nombre de secondes à utiliser pour l'opération de transfert de fichier. La valeur par défaut est 60 secondes.	
portsRange	Lorsque le mode actif est activé, la section portsRange restreint les numéros de port envoyés par la commande FTP PORT. Cette option accepte des règles de pare-feu très strictes. La section portsRange définit la plage de ports à utiliser côté client des connexions de données TCP. Si vous ne spécifiez pas la section portsRange, le système d'exploitation détermine les numéros de port à utiliser. Le paramètre portsRange requiert les paramètres suivants : min (port_min) La valeur minimale du port à utiliser côté client des connexions de données TCP. Les valeurs admises sont comprises entre 0 et 65535. Par exemple, si vous définissez cette valeur à 1035, IBM Workload Scheduler restreint les numéros de port au port 1035 et supérieur max (port_max) La valeur maximale du port à utiliser côté client des connexions de données TCP. Les valeurs admises sont comprises entre 0 et 65535. Par exemple, si vous définissez cette valeur à 1038, IBM Workload Scheduler restreint les numéros de port au port 1038 et inférieur Par exemple, pour limiter la plage de ports à utiliser côté client entre le port 1035 et le port 1040, indiquez : <pre><jsdlfiletransfer:portsRange> <jsdlfiletransfer:min>1035</jsdlfiletransfer:min> <jsdlfiletransfer:max>1040</jsdlfiletransfer:max> </jsdlfiletransfer:portsRange></pre>	
passiveMode	Indique si le client est passif ou actif dans l'établissement des connexions pour les transferts de données. Si vous définissez cette option sur NO, le serveur établit la connexion de données avec le client (mode actif). Si vous définissez cette option sur YES, le client établit la connexion de données avec le serveur (mode passif). La valeur par défaut est NO.	

Le fichier xml suivant présente la section "application" JSDL d'un exemple de définition de travail pour un type de travail de transfert de fichier :

```
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>FTP_SERVER</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>LOCAL_FILE</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>REMOTE_FILE</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:user:username>USERNAME</jsdl:user:username>
<jsdl:password>PASSWORD</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>PROTOCOL</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ASCII_BINARY</jsdlfiletransfer:transferMode>
<jsdlfiletransfer:codepageConversion>
<jsdlfiletransfer:remoteCodepage>RM_CP</jsdlfiletransfer:remoteCodepage>
<jsdlfiletransfer:localCodepage>LC_CP</jsdlfiletransfer:localCodepage>
</jsdlfiletransfer:codepageConversion>
<jsdlfiletransfer:timeout>CONNECTION_TIMEOUT</jsdlfiletransfer:timeout>
<jsdlfiletransfer:portsRange>
<jsdlfiletransfer:min>MIN_PORT</jsdlfiletransfer:min>
<jsdlfiletransfer:max>MAX_PORT</jsdlfiletransfer:max>
</jsdlfiletransfer:portsRange>
```

```

<jsd1filetransfer:passiveMode>YES_NO</jsdlfiletransfer:passiveMode>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>

```

L'exemple suivant montre une tâche généralisée qui télécharge un fichier à partir d'un poste de travail distant dont l'adresse est 10.0.0.8 :

```

$JOBS
AGENT#FILE_TRANSFER
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlfiletransfer"
  name="FILETRANSFER">
  <jsd1:application name="filetransfer">
    <jsd1filetransfer:filetransfer>
      <jsd1filetransfer:downloadInfo>
        <jsd1filetransfer:server>10.0.0.8</jsdlfiletransfer:server>
        <jsd1filetransfer:localfile>c:\MyTextFile.txt</jsdlfiletransfer:localfile>
        <jsd1filetransfer:remoteFile>./MyRemoteFile.txt</jsdlfiletransfer:remoteFile>
        <jsd1filetransfer:localCredentials>
          <jsd1:userName>${agent:locluser}</jsdl:userName>
          <jsd1:password>${agent:password.${agent:locluser}}</jsdl:password>1
        </jsdlfiletransfer:localCredentials>
        <jsd1filetransfer:remoteCredentials>
          <jsd1:userName>remuser</jsdl:userName>
          <jsd1:password>${password:remuser}</jsdl:password>2
        </jsdlfiletransfer:remoteCredentials>
        <jsd1filetransfer:protocol>FTP</jsdlfiletransfer:protocol>
        <jsd1filetransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
        <jsd1filetransfer:codepageConversion>
          <jsd1filetransfer:remoteCodepage>IBM-280</jsdlfiletransfer:remoteCodepage>
          <jsd1filetransfer:localCodepage>ISO8859-1</jsdlfiletransfer:localCodepage>
        </jsdlfiletransfer:codepageConversion>
      </jsdlfiletransfer:downloadInfo>
    </jsdlfiletransfer:filetransfer>
  </jsdl:application>
</jsdl:jobDefinition>

```

Remarque : dans les sections de droit d'accès,

1. le nom d'utilisateur local a été défini sur l'agent qui exécute le travail avec une variable locluser par le biais de la commande param. La valeur définie pour locluser sera récupérée au moment de l'exécution auprès du fichier de variables situé sur l'agent. De même, le mot de passe pour la valeur représentée par locluser a été défini sur l'agent à l'aide de la commande param et sera résolu au moment de l'exécution à partir du fichier de variables local.
2. Le nom d'utilisateur distant a été défini avec la commande composer user et est stocké dans la base de données avec son mot de passe sous le nom d'utilisateur remuser. Le mot de passe pour remuser sera extrait de la base de données au moment de l'exécution.

L'exemple suivant présente une définition de travail à utiliser pour télécharger un fichier texte à l'aide du protocole FTPES avec mode actif, délai d'attente de 10 secondes et plage de ports à utiliser comprise entre 1035 et 1038 :

```

<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
  jsdlfiletransfer" name="FTPES_DOWNLOAD_TEXT">
  <jsd1:application name="filetransfer">
    <jsd1filetransfer:filetransfer>
      <jsd1filetransfer:downloadInfo>
        <jsd1filetransfer:server>myServerFtp</jsdlfiletransfer:server>
        <jsd1filetransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
        <jsd1filetransfer:remoteFile>tmp/MyRemoteFile.txt</jsdlfiletransfer:remoteFile>
        <jsd1filetransfer:remoteCredentials>
          <jsd1:userName>myUser</jsdl:userName>
          <jsd1:password>myPassword</jsdl:password>

```



```

</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>FTPES</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
<jsdlfiletransfer:timeout>10</jsdlfiletransfer:timeout>
<jsdlfiletransfer:portsRange>
<jsdlfiletransfer:min>1035</jsdlfiletransfer:min>
<jsdlfiletransfer:max>1038</jsdlfiletransfer:max>
</jsdlfiletransfer:portsRange>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>

```

L'exemple suivant présente une définition de travail à utiliser pour transférer un fichier texte à l'aide du protocole SSH :

```

<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlfiletransfer" name="SSH_UPLOAD">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:uploadInfo>
<jsdlfiletransfer:server>myServer</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>/tmp/MyRemoteFile.txt</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>myUser</jsdl:userName>
<jsdl:password>myPassword</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>SSH</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
</jsdlfiletransfer:uploadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>

```

L'exemple suivant présente une définition de travail à utiliser pour transférer un fichier texte à l'aide du protocole WINDOWS :

```

<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlfiletransfer="http://www.ibm.com/xmlns/prod/scheduling/1.0/
jsdlfiletransfer" name="WINDOWS_DOWNLOAD">
<jsdl:application name="filetransfer">
<jsdlfiletransfer:filetransfer>
<jsdlfiletransfer:downloadInfo>
<jsdlfiletransfer:server>myServer</jsdlfiletransfer:server>
<jsdlfiletransfer:localfile>d:\MyLocalFile.txt</jsdlfiletransfer:localfile>
<jsdlfiletransfer:remotefile>mySharedFolder\MyRemoteFile.txt
</jsdlfiletransfer:remotefile>
<jsdlfiletransfer:remoteCredentials>
<jsdl:userName>myUser</jsdl:userName>
<jsdl:password>myPassword</jsdl:password>
</jsdlfiletransfer:remoteCredentials>
<jsdlfiletransfer:protocol>WINDOWS</jsdlfiletransfer:protocol>
<jsdlfiletransfer:transferMode>ascii</jsdlfiletransfer:transferMode>
</jsdlfiletransfer:downloadInfo>
</jsdlfiletransfer:filetransfer>
</jsdl:application>
</jsdl:jobDefinition>

```

Le type de travail de transfert de fichier fournit les codes retour suivants. Les codes retour sont disponibles uniquement pour les travaux terminés.

RC=0 Transfert de fichier terminé correctement.

RC=-1 Transfert de fichier non effectué. Le travail a échoué avec le code d'erreur suivant :
AWKFTE007E

Explication : une erreur s'est produite pendant l'opération de transfert de fichier.

Raisons possibles : fichier distant introuvable ou droits d'accès refusés.

RC=-2 Transfert de fichier non effectué. Le travail a échoué avec le code d'erreur suivant :
AWKFTE020E

Explication : uniquement pour les protocoles SSH ou WINDOWS. Une erreur a été renvoyée pendant la tentative de conversion de la page de codes.

Raisons possibles : pour les protocoles SSH ou WINDOWS, la page de codes est automatiquement détectée et convertie. Dans ce cas, il existe une erreur dans la page de codes du fichier à transférer qui ne respecte pas la page de codes du système local.

RC=-3 Transfert de fichier non effectué. Le travail a échoué avec le code d'erreur suivant :
WKFTE015E

Explication : une erreur s'est produite pendant l'opération de transfert de fichier.

Raisons possibles : fichier local introuvable.

RC=-4 Transfert de fichier effectué avec la page de codes par défaut. Le travail a échoué avec le code d'erreur suivant :
AWKFTE023E

Explication : la conversion de la page de codes spécifiée n'a pas été effectuée. Le transfert de fichier a été effectué avec la page de codes par défaut.

Raisons possible : la page de codes spécifiée est indisponible.

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux de transfert de fichiers IBM Workload Scheduler en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Après soumission, si le travail en cours d'exécution contient plusieurs fichiers à transférer et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire à l'aide de la commande **kill**.

Redémarrage du travail

Si l'agent IBM Workload Scheduler s'arrête lorsque vous soumettez le travail de transfert de fichier ou pendant que le travail est en cours d'exécution, le travail redémarre automatiquement au redémarrage de l'agent.

Propriétés de travail

Vous pouvez voir les propriétés du travail en exécutant `conman sj <nom_travail_transfert_fichier>;props`, où `<nom_travail_transfert_fichier>` est le nom du travail de transfert de fichier.

Pour un travail de transfert de fichier qui, par exemple, transfère quatre fichiers, dans la section Extra Information de la commande de sortie, vous voyez les propriétés suivantes :

```
Informations supplémentaires
File.0.LocalFile = ./MyFile3.log
File.0.LocalUser = agtParis
File.0.Protocol = FTP
File.0.RemoteFile = /export/home/FileTransferRep/MyFile3.log
File.0.RemoteUser = ftpuser
File.0.Size = 983
File.0.TotalTransferTime = 0
File.1.LocalFile = ./MyFile.log
File.1.LocalUser = agtParis
File.1.Protocol = FTP
File.1.RemoteFile = /export/home/FileTransferRep/MyFile.log
File.1.RemoteUser = ftpuser
File.1.Size = 983
File.1.TotalTransferTime = 0
File.2.LocalFile = ./MyFile2.log
File.2.LocalUser = agtParis
File.2.Protocol = FTP
File.2.RemoteFile = /export/home/FileTransferRep/MyFile2.log
File.2.RemoteUser = ftpuser
File.2.Size = 983
File.2.TotalTransferTime = 0
File.3.LocalFile = ./MyFile1.log
File.3.LocalUser = agtParis
File.3.Protocol = FTP
File.3.RemoteFile = /export/home/FileTransferRep/MyFile1.log
File.3.RemoteUser = ftpuser
File.3.Size = 983
File.3.TotalTransferTime = 0
Number of transferred files = 4
```

où pour chaque fichier transféré, vous avez :

File.compteur.LocalFile

Pour le numéro de fichier #, le nom du fichier local que vous souhaitez transférer.

File.compteur.LocalUser

Pour le numéro de fichier #, le nom d'utilisateur permettant d'accéder au poste de travail en local.

File.compteur.Protocol

Pour le numéro de fichier #, le protocole à utiliser pour le transfert de fichier.

File.compteur.RemoteFile

Pour le numéro de fichier #, le nom du fichier distant que vous souhaitez transférer.

File.compteur.RemoteUser

Pour le numéro de fichier #, le nom d'utilisateur permettant d'accéder au poste de travail distant.

File.compteur.Size

Pour le numéro de fichier #, la taille du fichier transféré.

File.compteur.TotalTransferTime

Pour le numéro de fichier #, la durée de transfert totale, en secondes.

Nombre de fichiers transférés

Le nombre de fichiers transférés avec le travail. Vous n'avez qu'une seule valeur de cette propriété pour chaque travail.

Vous pouvez exporter les propriétés du travail de transfert de fichier que vous pouvez voir dans la section Extra Information vers un travail successif dans la même instance de flot de travaux.

Pour plus d'informations sur la liste des propriétés de travail que vous pouvez exporter, voir tableau 102, à la page 657.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Procédure préalable de création de travaux OSLC Automation et OSLC Provisioning

Comment créer une nouvelle définition de travail OSLC Automation et OSLC Provisioning à l'aide de Dynamic Workload Console.

Pour créer une définition de travail OSLC Automation ou OSLC Provisioning, vous devez d'abord terminer les étapes nécessaires mentionnées ci-après.

Remarque : Avant de réaliser la procédure suivante, assurez-vous d'avoir installé Jazz for Service Management Registry Services à partir d'Installation Manager.

1. Obtenez le certificat du serveur Registry Services et sauvegardez-le dans un répertoire que vous utiliserez ultérieurement. L'administrateur Registry Services peut fournir le certificat ou, avec le navigateur Firefox par exemple, vous pouvez extraire le certificat à l'aide de la procédure suivante :
 - a. Connectez-vous à Registry Services (par exemple, <https://nomhôte:16311/oslc/pr>)
 - b. Téléchargez le certificat en cliquant dans la barre d'outils du navigateur sur : **Outils>Options>Avancé>Chiffrement>Afficher les certificats**

- c. Sélectionnez **IBM>nom_hôte_Registry_Services:port** et cliquez sur **Exporter**. Un fichier est créé avec le nom indiqué, par exemple `monserveur:16311`.
2. Accédez au répertoire dans lequel un environnement d'exécution Java est installé, par exemple : `C:\Program Files\IBM\TWA_<utilisateur_TWS>\TWS\JavaExt\jre\jre\bin`
3. Créez un nouveau fichier de clés certifiées en lançant la commande suivante :
`keytool -genkeypair -alias certificatekey -keyalg RSA -validity 7 -keystore trustore_directory\keystore.jks`

Remarque : Vérifiez que `trustore_directory` n'est pas créé dans le chemin `javaExt\JRE`.

4. Ajoutez le certificat IBM Registry Services au magasin en exécutant la commande suivante : `keytool -import -file repertoire_certificat\ nom_certificat -alias oslc -keystore repertoire_magasin\keystore.jks`
5. Ouvrez le fichier `REP_BASE_TWA\TWS\ITA\cpa\config\JobManager.ini`, et localisez la section `JavaJobLauncher`, ligne `JVMOptions`.
6. Ajoutez les instructions suivantes à la ligne : `"-Djavax.net.ssl.trustStore=REPETOIRE_MAGASIN/keystore.jks -Djavax.net.ssl.trustStorePassword=MOT_DE_PASSE_MAGASIN"`. Par exemple :
`JVMOptions = -Djavax.net.ssl.trustStore=C:/myUtils/keystore.jks -Djavax.net.ssl.trustStorePassword=passw0rd`
7. Arrêtez et redémarrez l'agent.
8. Créez les fichiers `OSLCAutomation.properties` et `OSLCProvisioning.properties`, respectivement pour les travaux OSLC Automation et OSLC Provisioning, puis recherchez-les dans `<TWA_Home>/TWS/JavaExt/cfg/`.
Indiquez les catalogues de fournisseur de services (ou les services de registre) que vous utiliserez plus tard pour créer le travail au format suivant :
`ServiceProviderCatalogName=RegistryServicesURI`
9. Sur le gestionnaire de domaine maître, extrayez le certificat de sécurité depuis le fichier de clés et ajoutez-le au magasin Jazz for Service Management. Les chemins d'accès au fichier et les noms sont les suivants :

Fichier de clés du gestionnaire de domaine maître
`<TWA_Home>/WAS/TWSPProfile/etc/TWSServerKeyFile.jks`

Magasin de clés de confiance Jazz for Service Management
`<JazzSM_Home>/profile/config/cells/JazzSMNode01Cell/nodes/JazzSMNode01Cell/trust.p12`

10. Sur le poste de travail sur lequel Jazz for Service Management est installé, extrayez le certificat de sécurité depuis le fichier de clés et ajoutez-le au magasin du gestionnaire de domaine maître. Les chemins d'accès au fichier et les noms sont les suivants :

Magasin de clés Jazz for Service Management
`<JazzSM_Home>/profile/config/cells/JazzSMNode01Cell/nodes/JazzSMNode01Cell/key.p12`

Magasin de clés de confiance du gestionnaire de domaine maître
`<TWA_Home>/WAS/TWSPProfile/etc/TWSServerTrustFile.jks`

11. Fermez et redémarrez WebSphere Application Server sur le gestionnaire de domaine maître et sur Jazz for Service Management.

Définition de travail - OSLC Automation

Un travail OSLC Automation appelle n'importe quel fournisseur OSLC qui implémente la spécification d'automatisation OSLC. Les ressources d'automatisation définissent des plans d'automatisation, des requêtes d'automatisation et des résultats d'automatisation du cycle de vie de développement, de test et de déploiement du logiciel. Pour plus d'informations, voir <http://open-services.net/wiki/automation/OSLC-Automation-Specification-Version-2.0/>.

Cette section décrit les attributs obligatoires et facultatifs des travaux d'automatisation OSLC. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 80. Attributs obligatoires et facultatifs pour la définition d'un travail d'automatisation OSLC.

Attribut	Description et valeur	Obligatoire
URI de Registry Services	Adresse de Registry Services (par exemple, https://myhost.mydomain:16311/oslc/pr).	
Nom d'utilisateur de Registry Services	Utilisateur se connectant à Registry Services.	
Mot de passe de Registry Services	Mot de passe associé à l'utilisateur se connectant à Registry Services.	
URI du fournisseur de services	Adresse du fournisseur de services.	✓
Nom d'utilisateur du fournisseur de services	Utilisateur se connectant au fournisseur de services.	
Mot de passe d'accès au fournisseur de services	Mot de passe associé à l'utilisateur se connectant au fournisseur de services.	
Demande	Représentation RDF de la demande d'automatisation.	✓

L'exemple suivant affiche un travail qui planifie un flot de travaux IBM Workload Scheduler :

```
$JOBS
WKS#AUTOMATION
TASK
  <?xml version="1.0" encoding="UTF-8"?>
  <jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
    xmlns:jSDLoslcAutomation="
http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDLoslcAutomation" name="OSLCAUTOMATION">
    <jSDL:application name="oslcAutomation">
      <jSDLoslcAutomation:oslcAutomation>
        <jSDLoslcAutomation:OSLCAutomationParameters>
          <jSDLoslcAutomation:AutomationRequest>
            <jSDLoslcAutomation:automationRequestGroup>
              <jSDLoslcAutomation:automationRequestBody>

<!-- ajoutez la représentation rdf de la ressource -->

            </jSDLoslcAutomation:automationRequestBody>
          </jSDLoslcAutomation:automationRequestGroup>
        </jSDLoslcAutomation:AutomationRequest>
      <jSDLoslcAutomation:ConnectionInfo>
        <jSDLoslcAutomation:ServiceProviderCatalogGroup>
          <jSDLoslcAutomation:catalogURI>
https://registryserviceshost.domain:16311/oslc/pr
          </jSDLoslcAutomation:catalogURI>
        </jSDLoslcAutomation:ServiceProviderCatalogGroup>
      </jSDLoslcAutomation:ConnectionInfo>
    </jSDL:application>
  </jSDL:jobDefinition>
```

```

        </jsdloslautomation:catalogURI>
        <jsdloslautomation:username>registryUser</jsdloslautomation:
            username>
        <jsdloslautomation:password>registryPassword</jsdloslautomation:
            password>
    </jsdloslautomation:ServiceProviderCatalogGroup>
    <jsdloslautomation:ServiceProviderGroup>
        <jsdloslautomation:serviceProviderURI>
https://serviceprovideraddress.domain:16310/oslc/providers/1360665198982</jsdloslautomation:
            serviceProviderURI>
        <jsdloslautomation:usernameSP>
            serviceProviderUser
        </jsdloslautomation:usernameSP>
        <jsdloslautomation:passwordSP>
            serviceProviderPassword
        </jsdloslautomation:passwordSP>
    </jsdloslautomation:ServiceProviderGroup>
    </jsdloslautomation:ConnectionInfo>
    </jsdloslautomation:OSLCAutomationParameters>
</jsdloslautomation:oslautomation>
</jsdl:application>
</jsdl:jobDefinition>

```

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux d'automatisation IBM Workload Scheduler OSLC en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console ou de la ligne de commande **comman**.

Après la soumission, une fois que le travail est en cours d'exécution et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire en utilisant la commande **kill** à partir de la ligne de commande ou de Dynamic Workload Console. Cependant, cette action est pertinente uniquement sur le travail IBM Workload Scheduler, et elle n'affecte aucun processus s'exécutant sur le poste de travail OSLC Automation. Lorsque vous arrêtez le travail IBM Workload Scheduler, IBM Workload Scheduler affecte le statut **Error** ou **ABEND** avec le code retour 0 au travail IBM Workload Scheduler, quel que soit le statut du travail OSLC Automation.

Surveillance du travail

Si l'agent IBM Workload Scheduler s'arrête lorsque vous soumettez le travail d'automatisation OSLC ou pendant que le travail est en cours d'exécution, IBM Workload Scheduler reprend la surveillance du travail là où il s'est arrêté.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Définition de travail - Application des accès OSLC

Un travail d'application des accès OSLC appelle un fournisseur OSLC (IBM Workload Scheduler et IBM SmartCloud Orchestrator, par exemple) qui implémente la spécification OSLC Provisioning. Les ressources d'application des accès définissent des plans d'application des accès, des requêtes d'application des accès et des résultats d'application des accès du cycle de vie de développement, de test et de déploiement du logiciel.

La présente section décrit les attributs obligatoires et facultatifs des travaux d'application des accès OSLC. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 81. Attributs obligatoires et facultatifs pour la définition d'un travail d'application des accès OSLC.

Attribut	Description/valeur	Obligatoire
URI de Registry Services	Adresse de Registry Services (par exemple, https://myhost.mydomain:16311/odlc/pr).	✓
Nom d'utilisateur de Registry Services	Utilisateur se connectant à Registry Services.	✓
Mot de passe de Registry Services	Mot de passe associé à l'utilisateur se connectant à Registry Services.	✓
URI du fournisseur de services	Adresse du fournisseur de services.	✓
Nom d'utilisateur du fournisseur de services	Utilisateur se connectant au fournisseur de services.	✓
Mot de passe d'accès au fournisseur de services	Mot de passe associé à l'utilisateur se connectant au fournisseur de services.	✓
Instance	Représentation RDF de l'instance à déployer.	✓

L'exemple suivant présente un travail qui planifie l'application des accès d'un modèle de système :

```
WKS#PROVSAMPLETASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdloslcp provisioning="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdloslcp provisioning"
  name="OSLCPROVISIONING">
  <jsdl:application name="oslcprovisioning">
    <jsdloslcp provisioning:oslcprovisioning>
      <jsdloslcp provisioning:OSLCProvisioningParameters>
        <jsdloslcp provisioning:actionPanel>
          <jsdloslcp provisioning:instanceFromTemplate>
            <jsdloslcp provisioning:instance>
              <!-- Définition RDF de l'instance>
              &lt;?xml version=&quot;1.0&quot; encoding=&quot;UTF-8&quot;?>
              &lt;rdf:RDF
                xmlns:rdf=&quot;http://www.w3.org/1999/02/22-rdf-syntax-ns#&quot;
                xmlns:oslc=&quot;http://open-services.net/ns/core#&quot;
                xmlns:sco=&quot;http://jazz.net/ns/ism/provisioning/sco#&quot;
                xmlns:oslc_auto=&quot;http://open-services.net/ns/auto#&quot;
                xmlns:dcterms=&quot;http://purl.org/dc/terms/&quot;
                xmlns:rdfs=&quot;http://www.w3.org/2000/01/rdf-schema#&quot; >
                &lt;rdf:Description rdf:about=&quot;
                  http://myServiceProvider.domain:31115/CLIModelWeb/OSLC/BatchApplicationInstance/BatchApplication/
                    0f01af24-72e0-3c4b-b95c-18f908c76898&quot;?>
                  &lt;oslc_auto:parameterDefinition rdf:nodeID=&quot;A1&quot;?>
```



```

        <olc_auto:parameterDefinition rdf:nodeID="A2"/>
        <rf:type rdf:resource="http://jazz.net/ns/ism/provisioning/sco#Entity"/>
        <olc_auto:parameterDefinition rdf:nodeID="A4"/>
        <dcterms:identifier>0f01af24-72e0-3c4b-b95c-18f908c76898</dcterms:identifier>
        <olc_auto:parameterDefinition rdf:nodeID="A5"/>
        <dcterms:title>InstanceName</dcterms:title>
        <olc_auto:parameterDefinition rdf:nodeID="A0"/>
    </rdf:Description>
    <rdf:Description rdf:nodeID="A5">
        <olc:name>XML</olc:name>
        <olc:value><?xml version="1.0" encoding="UTF-8"?>
            <model:TWSBatchApplicationInstance xmlns:model="
http://www.ibm.com/xmlns/prod/scheduling/1.0/Model">
                <model:Name>InstanceName</model:Name>
                <!--Here starts the definition of jobs, job streams, etc contained in this instance>
                ...
                <!--Here starts the definition >
            </model:TWSBatchApplicationInstance></model:Name>
            <olc:defaultValue></olc:defaultValue>
            <rf:type rdf:resource="http://open-services.net/ns/core#Property"/>
        </rdf:Description>
        <!--Mapping of the instance>
        <rdf:Description rdf:nodeID="A0">
            <olc:name>JOB_SAMPLE</olc:name>
            <olc:value>JOB_TARGET</olc:value>
            <olc:defaultValue>JOB_TARGET</olc:defaultValue>
            <rf:type rdf:resource="http://open-services.net/ns/core#Property"/>
        </rdf:Description>
        <!--Here continues the definition of jobs, job streams, etc contained in this instance>
        ...
        <!--Here ends the mapping>
        <rdf:Description rdf:nodeID="A7">
            <sco:node rdf:resource="
http://thinklinux.31115/CLIModelWeb/OSLC/BatchApplicationInstance/BatchApplication/
0f01af24-72e0-3c4b-b95c-18f908c76898"/>
            <rf:type rdf:resource="http://jazz.net/ns/ism/provisioning/sco#Template"/>
            </rdf:Description>
            <rdf:Description rdf:nodeID="A1">
                <olc:name>ICON</olc:name>
                <olc:value>../js/images/default.png</olc:value>
                <olc:defaultValue>../js/images/default.png</olc:defaultValue>
                <rf:type rdf:resource="http://open-services.net/ns/core#Property"/>
            </rdf:Description>
            <rdf:RDF></jsdloslcp provisioning:instance>
            </jsdloslcp provisioning:instanceFromTemplate>
            </jsdloslcp provisioning:actionPanel>
            <jsdloslcp provisioning:connectionInfo>
            <jsdloslcp provisioning:serviceProviderCatalog>
            <jsdloslcp provisioning:catalogURI>
                https://myregistry.domain:16311/olc/pr</jsdloslcp provisioning:catalogURI>

            <jsdloslcp provisioning:username>registryServicesUser</jsdloslcp provisioning:username>

            <jsdloslcp provisioning:password>registryServicesPassword</jsdloslcp provisioning:password>
            </jsdloslcp provisioning:serviceProviderCatalog>
            <jsdloslcp provisioning:serviceProvider>
            <jsdloslcp provisioning:serviceProviderURI>
                https://myregistry.domain:16311/olc/providers/1380617052297
            </jsdloslcp provisioning:serviceProviderURI>

            <jsdloslcp provisioning:usernameSP>myServiceProviderUser</jsdloslcp provisioning:usernameSP>
            <jsdloslcp provisioning:passwordSP>myServiceProviderPassword</jsdloslcp provisioning:passwordSP>
            </jsdloslcp provisioning:serviceProvider>
            </jsdloslcp provisioning:connectionInfo>
            </jsdloslcp provisioning:OSLCProvisioningParameters>
            </jsdloslcp provisioning:oslcprovisioning>
        </jsdl:application>
    </jsdl:jobDefinition>

```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux de base de données

Un travail de base de données se connecte par interface à un certain nombre de bases de données, y compris des bases de données personnalisées, pour exécuter des requêtes, des instructions SQL et des travaux. Vous pouvez également créer et exécuter des procédures stockées sur les bases données DB2, Oracle, Microsoft SQL Server et Netezza.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Le tableau suivant répertorie les attributs obligatoires et facultatifs pour les travaux DB2, Oracle et Netezza :

Tableau 82. Attributs obligatoires et facultatifs pour la définition d'un travail de base de données.

Attribut	Description et valeur	Obligatoire
nom d'application	base de données	✓
dbms	Type de base de données dans lequel vous souhaitez exécuter le travail. Les valeurs prises en charge sont : db2 Pour les bases de données DB2 mssql Pour les bases de données Microsoft SQL Server oracle Pour les bases de données Oracle IBM Netezza Pour la base de données Netezza BigSQL Pour la base de données InfoSphere BigInsights BigSQL Hive Pour la base de données Apache Hive	✓
serveur	Nom d'hôte du serveur sur lequel la base de données est installée.	✓
port	Numéro de port pour le travail de base de données.	✓
database	Nom de la base de données.	Obligatoire uniquement pour les bases de données DB2.
driverClass	Nom de classe du pilote JDBC	Obligatoire si vous spécifiez une base de données personnalisée.
connectionUrl	Chaîne qui est utilisée pour la connexion à la base de données, contenant l'URL de base de données, le nom d'utilisateur et le mot de passe	Obligatoire si vous spécifiez une base de données personnalisée.
driverPath	Chemin d'accès aux fichiers JAR du client de base de données. Cette valeur remplace la valeur indiquée dans le fichier de configuration DatabaseJobExecutor.properties, s'il existe. Si vous sélectionnez la base de données Microsoft SQL Server, la version 4 des pilotes JDBC est requise.	
dbStatement	Nom du travail de base de données à exécuter.	Obligatoire si vous spécifiez une instruction SQL

Tableau 82. Attributs obligatoires et facultatifs pour la définition d'un travail de base de données. (suite)

Attribut	Description et valeur	Obligatoire						
storedProcedure name	Nom de la procédure stockée dans des bases de données DB2, Oracle ou MSSQL. La procédure ne peut pas être stockée sur DB2 si la base de données contient déjà une ou plusieurs procédures stockées avec le même nom et le même schéma. Par exemple, si la base de données comporte plusieurs procédures stockées appelées TEST.STORE_PROC1, avec différents paramètres comme dans ce qui suit : TEST.STORE_PROC1(VARCHAR,?) TEST.STORE_PROC1(VARCHAR,VARCHAR,?) TEST.STORE_PROC1(VARCHAR,?,?) TEST.STORE_PROC1(VARCHAR,VARCHAR,?,?) le travail de base de données ne peut pas être créé et le message suivant est renvoyé : AWKDBE033E Le nom de procédure stockée fourni correspond à plus d'une définition de procédure stockée dans la base de données. Afin de lever toute ambiguïté, indiquez aussi le schéma. Remarque : Cet attribut n'est pas pris en charge pour les bases de données BigSQL et Hive.	Obligatoire si vous spécifiez une procédure stockée.						
ParameterTableValue key	Nom et valeurs de la procédure exprimés conformément à la syntaxe suivante : Type de variable de la procédure stockée Les valeurs prises en charge sont : • IN • OUT • INOUT Nom de la variable Nom de la variable tel que défini dans la procédure stockée. Type de variable Type de la variable. Les types SQL pris en charge sont les suivants : • DATE • DECIMAL • INTEGER • VARCHAR 0...n Position de chaque variable telle que définie dans la procédure stockée. Par exemple : <table><tr><td>Nom</td><td>Valeur</td></tr><tr><td>IN VARIN DATE 0</td><td>2012-01-01</td></tr><tr><td>OUT VAROUT DATE 1</td><td>?</td></tr></table>	Nom	Valeur	IN VARIN DATE 0	2012-01-01	OUT VAROUT DATE 1	?	Obligatoire si vous spécifiez une procédure stockée.
Nom	Valeur							
IN VARIN DATE 0	2012-01-01							
OUT VAROUT DATE 1	?							
ParameterTableValue content	Valeur de variable. Pour les variables de sortie, la valeur doit être : ?. Pour entrer une variable de date, utilisez le format suivant : aaaa-mm-jj. Si aucune valeur n'est indiquée pour un paramètre, la valeur est considérée comme NULL dans la base de données.	Obligatoire si vous spécifiez une procédure stockée.						
Output file	Chemin d'accès complet à l'emplacement où vous voulez enregistrer la sortie de votre requête SQL.							

Tableau 82. Attributs obligatoires et facultatifs pour la définition d'un travail de base de données. (suite)

Attribut	Description et valeur	Obligatoire
credentials	Nom d'utilisateur et mot de passe d'accès à la base de données (les utilisateurs de domaine sont pris en charge sous la forme DOMAINE\UTILISATEUR). Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom d'utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <pre><jsdl:password>\${password:nom_utilisateur}</jsdl:password></pre> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <pre><jsdl:password>\${password:poste_travail#nom_util}</jsdl:password></pre> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <pre><jsdl:userName>\${agent:util}</jsdl:userName> <jsdl:password>\${agent:password.util}</jsdl:password></pre> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645. - Lorsque la sécurité intégrée est utilisée pour le serveur Microsoft SQL, la bibliothèque sqljdbc_auth.dll (qui fait partie du package Microsoft JDBC Driver 4.0 for SQL Server) doit être placée dans le même répertoire que le pilote JDBC, et ce chemin doit être ajouté à la variable d'environnement système PATH.	

Le schéma des bases de données DB2, Oracle et Netezza est très similaire ; l'exemple suivant montre donc un travail qui exécute une requête sur une base de données DB2 :

```

$JOBS
AGENT#DATABASE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdl:database="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl:database" name="database">
  <jsdl:application name="database">
    <jsdl:database:database>
      <jsdl:database:sqlActionInfo>
        <jsdl:database:dbms>db2</jsdl:database:dbms>
        <jsdl:database:server>localhost</jsdl:database:server>
        <jsdl:database:port>50000</jsdl:database:port>
        <jsdl:database:database>TWS32</jsdl:database:database>
        <jsdl:database:statements>
          <jsdl:database:dbStatement>SELECT * FROM DWB.ARE_ABSTRACT_
RESOURCES</jsdl:database:dbStatement>
        </jsdl:database:statements>
        <jsdl:database:credentials>
          <jsdl:userName>${agent:dbvars..dbtwouser}</jsdl:userName>
          <jsdl:password>${agent:password.${agent:dbvars..dbtwouser}}</jsdl:password>
        </jsdl:database:credentials>
        </jsdl:database:sqlActionInfo>
      </jsdl:database:database>
    </jsdl:application>
  </jsdl:jobDefinition>

```

```

</jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Defined using composer."
RECOVERY STOP

```

Remarque : (1) le nom d'utilisateur a été défini sur l'agent qui exécute le travail avec une variable dbtwouser par le biais de la commande param. La valeur définie pour dbtwouser sera récupérée au moment de l'exécution auprès du fichier de variables dbvars situé sur l'agent. De même, le mot de passe pour la valeur représentée par dbtwouser a été défini sur l'agent à l'aide de la commande param et sera résolu au cours de l'exécution à partir du même fichier de variables.

Définition de travail - Travaux MSSQL

Cette section décrit les attributs obligatoires et facultatifs des travaux MSSQL. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 83. Attributs obligatoires et facultatifs pour la définition d'un travail MSSQL.

Attribut	Description/valeur	Obligatoire
nom d'application	base de données	✓
dbms	Type de base de données dans lequel vous souhaitez exécuter le travail. Etant donné que ce travail est spécifique pour la base de données Microsoft SQL Server, la seule valeur prise en charge est mssql.	✓
driverPath	Chemin d'accès aux fichiers JAR du client de base de données. Cette valeur remplace la valeur indiquée dans le fichier de configuration DatabaseJobExecutor.properties, s'il existe. La version 4 des pilotes JDBC est requise.	
serveur	Nom d'hôte du serveur sur lequel la base de données est installée.	✓
port	Numéro de port pour le travail de base de données.	✓
database	Nom de la base de données.	✓
dbStatement	Instruction SQL. Pour séparer les instructions, utilisez une ligne vide.	✓

Tableau 83. Attributs obligatoires et facultatifs pour la définition d'un travail MSSQL. (suite)

Attribut	Description/valeur	Obligatoire
credentials	Nom d'utilisateur et mot de passe d'accès à la base de données (les utilisateurs de domaine sont pris en charge sous la forme DOMAINE\UTILISATEUR). Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom_utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur <i>nom_utilisateur</i> (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <code><jsdl:password>\${password:nom_utilisateur}</jsdl:password></code> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <code><jsdl:userName>\${agent:util}</jsdl:userName></code> <code><jsdl:password>\${agent:password.util}</jsdl:password></code> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645. - Lorsque la sécurité intégrée est utilisée pour le serveur Microsoft SQL, la bibliothèque sqljdbc_auth.dll (qui fait partie du package Microsoft JDBC Driver 4.0 for SQL Server) doit être placée dans le même répertoire que le pilote JDBC, et ce chemin doit être ajouté à la variable d'environnement système PATH.	

L'exemple suivant présente un travail qui exécute un travail sur une base de données MSSQL :

```

$JOBS
AGENT#MSSQLJOB
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdl:database="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl:database" name="database">
  <jsdl:application name="mssqljob">
    <jsdl:database:database>
      <jsdl:database:sqlActionInfo>
        <jsdl:database:dbms>mssql</jsdl:database:dbms>
        <jsdl:database:server>localhost</jsdl:database:server>
        <jsdl:database:port>111</jsdl:database:port>
        <jsdl:database:database>MYDATABASE</jsdl:database:database>
        <jsdl:database:statements>
          <jsdl:database:dbStatement type="job">sada</jsdl:database:dbStatement>
        </jsdl:database:statements>
        <jsdl:database:credentials>
          <jsdl:userName>mssqluser</jsdl:userName>
          <jsdl:password>${agent:password.mssqluser}</jsdl:password>1
        </jsdl:database:credentials>
      </jsdl:database:sqlActionInfo>
    </jsdl:database:database>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Defined using composer."
RECOVERY STOP

```

Remarque : (1) Le mot de passe de l'utilisateur mssqluser a été défini à l'aide de la commande d'utilitaire param dans le fichier de variables sur l'agent qui doit exécuter le travail. Il sera résolu au cours de l'exécution avec la valeur définie.

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux de base de données IBM Workload Scheduler en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console ou d'Application Lab.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état EXEC dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire à l'aide de la commande **kill**. Le travail de base de données est supprimé et une annulation automatique est exécutée. Lorsque l'opération d'annulation se termine, un nouveau travail de base de données IBM Workload Scheduler est lancé automatiquement.

Propriétés de travail

Vous pouvez afficher les propriétés du travail en exécutant la commande `conman sj <nom_travail>;props`, où `<nom_travail>` correspond au nom du travail de base de données.

Vous pouvez exporter certaines des propriétés de travail figurant dans la section Informations supplémentaires de la commande de sortie, vers un travail successeur dans le même flot de travaux ou un flot de travaux différent. Pour plus d'informations relatives à la liste des propriétés de travail que vous pouvez exporter, voir tableau 113, à la page 661.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande `conman sj <nom_travail>;stdlist`, où `<nom_travail>` correspond au nom du travail de la base de données.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Configuration requise pour exécuter des travaux de branche

Pour exécuter un travail de branche générique, vérifiez que la configuration système suivante est respectée.

Sur les systèmes d'exploitation Windows :

- Etant donné que les systèmes d'exploitation Windows ne peuvent pas, en natif, interpréter les scripts de shell UNIX, vous devez installer un interpréteur de

shell pour utiliser le script de shell *rép_principal_TWA\TWS\bin\branch.sh* sur le gestionnaire de domaine maître Windows.

- Le répertoire *C:\cygwin\bin* doit pointer vers le sous-répertoire *bin* de votre répertoire d'installation Cygwin. Si vous avez installé Cygwin dans un répertoire différent du répertoire par défaut, utilisez le chemin correspondant.

Sur les systèmes d'exploitation UNIX, seuls les interpréteurs de commandes Bash ou Korn sont pris en charge.

Si votre poste de travail UNIX utilise un interpréteur de commandes Bourne, ouvrez le fichier **branch.sh** situé dans *rép_principal_TWA/TWS/bin* sur le gestionnaire de domaine maître et modifiez *#!/bin/sh* de manière à pointer sur un interpréteur de commandes Bash (*#!/bin/bash*) ou Korn (*#!/bin/ksh*).

travaux de services Web

Un travail de services Web exécute un service Web de votre choix. Les travaux de services Web ne fonctionnent qu'avec les paramètres d'entrée et de sortie qui utilisent des types de données simples.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs pour les travaux de services Web. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 84. Attributs obligatoires et facultatifs pour la définition d'un travail de services Web

Attribut	Description/valeur	Obligatoire
application name	ws	✓
opération	Le nom de la commande de services Web que vous appelez.	✓
wsdlURL	Nom de l'URL de service Web.	✓
arguments	Les paramètres requis par la commande de services Web que vous appelez (le nombre de valeurs dépend de la commande).	

Tableau 84. Attributs obligatoires et facultatifs pour la définition d'un travail de services Web (suite)

Attribut	Description/valeur	Obligatoire
credentials	Le nom et le mot de passe de l'utilisateur qui exécute ce travail. Comme alternative au codage en dur des valeurs réelles, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom d'utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <code><jsdl:password>\${password:nom_utilisateur}</jsdl:password></code> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <code><jsdl:password>\${password:poste_travail#nom_util}</jsdl:password></code> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <code><jsdl:userName>\${agent:util}</jsdl:userName></code> <code><jsdl:password>\${agent:password.util}</jsdl:password></code> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645. Si vous utilisez une connexion HTTPS, assurez-vous que les certificats de sécurité sont configurés pour le gestionnaire de travaux sur le poste de travail sur lequel le travail doit s'exécuter.	

Le résultat de la commande est enregistré dans le journal de travail.

L'exemple suivant présente une tâche qui exécute la commande de services Web getSum. La définition de tâche fournit dans la section arguments les deux valeurs qui doivent être ajoutées.

```

$JOBS
AGENT#WEB_SERVICE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlws="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlws" description="Calls a web
service to perform a sum of two numbers" name="SumNumber">
<jsdl:annotation>annotation</jsdl:annotation>
<jsdl:variables>
<jsdl:stringVariable description="URL of Web Service"
name="wsdlURL">http://np515516.cyber.com:9080/
Sum/services/Sum/wsdl/Sum.wsdl</jsdl:stringVariable>
<jsdl:stringVariable description="Operation to Invoke"
name="Operation">getSum</jsdl:stringVariable>
</jsdl:variables>
<jsdl:application name="ws">
<jsdlws:ws>
<jsdlws:wsToInvoke operation="{Operation}" wsdlURL="{wsdlURL}">
<jsdlws:arguments>

```

```

<jsdls:value>1</jsdls:value>
<jsdls:value>2</jsdls:value>
</jsdls:arguments>
<jsdls:credentials>
<jsd1:userName>administrator</j1:userName>
<jsd1:password>mdp</j1:password>
</jsdls:credentials>
</jsdls:wsToInvoke>
</jsdls:ws>
</j1:application>
<j1:resources>
<j1:candidateHosts>
<j1:hostName>${host}</j1:hostName>
</j1:candidateHosts>
</j1:resources>
</j1:jobDefinition>

```

L'exemple suivant s'applique si vous utilisez une connexion HTTPS avec l'agent exécutant la tâche de services Web. Il montre comment configurer la clé JVMOptions dans le fichier jobManager.ini de l'agent afin de pointer vers les certificats de sécurité.

```

JVMOptions=-Djavax.net.ssl.keyStore=/images/ITAuser/TWA/TWS/JavaExt/cfg/agentKeystore.jks
-Djavax.net.ssl.keyStorePassword=tdwb8nxt
-Djavax.net.ssl.trustStore=/images/ITAuser/TWA/TWS/JavaExt/cfg/agentKeystore.jks
-Djavax.net.ssl.trustStorePassword=tdwb8nxt
-Djavax.net.ssl.trustStoreType=JKS

```

Par défaut, le délai d'attente des services Web est de 90 secondes. Si le service Web ne s'est pas bien exécuté dans cette période, le travail IBM Workload Scheduler associé se termine par une erreur. Pour éviter l'échec des travaux lorsque l'exécution des services Web prend plus de 90 secondes, vous pouvez personnaliser le délai d'attente.

Le délai d'attente est spécifié dans le fichier WSJobExecutor.properties situé dans le répertoire TWA_HOME/TWS/JavaExt/cfg. Ce délai est structuré comme suit :

```

# Web Service timeout
# Specify the timeout in seconds
# Default Value is 90 seconds
TIMEOUT=90

```

Par exemple, pour un service Web dont l'exécution prend 5 minutes, personnalisez le fichier de configuration en indiquant : TIMEOUT=300.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

travaux de type services Web RESTful

Un travail de type services Web RESTful envoie des demandes à une ressource Web par le biais des méthodes HTTP suivantes :

DELETE

Supprime une ressource

GET Extrait l'en-tête et le corps d'une ressource**HEAD**

Extrait uniquement l'en-tête d'une ressource sans son corps

POST Crée une ressource**PUT** Modifie l'état d'une ressource ou la met à jour

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs des travaux de type services Web RESTful. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 85. Attributs obligatoires et facultatifs pour la définition d'un travail de type services Web RESTful

Attribut			Description/valeur	Obligatoire
Si le service Web nécessite une authentification, vous pouvez utiliser des informations d'identification ou des certificats à cet effet.	credentials	userName	Le nom d'utilisateur requis pour accéder à l'URI du service Web.	
		mot de passe	Le mot de passe à l'aide duquel userName accède au service Web.	
	CertificateGroup	keyStoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion. Seuls les fichiers de clés avec l'extension jks sont pris en charge.	
		mot de passe	Mot de passe qui protège la clé privée et qui est requis pour établir la connexion.	
		HostnameVerifyCheckbox	Vérifie que la syntaxe du nom du serveur Web, tel qu'il figure dans le fichier de clés, correspond exactement à l'URI du service Web. S'ils ne correspondent pas, aucune autorisation d'accès au serveur n'est accordée. Ce contrôle n'est pas appliqué si ce mot clé n'est pas utilisé.	
RESTAction	URI		L'ID ressource du service Web auquel vous envoyez la demande.	✓
	method		La méthode HTTP qui définit la demande. Il peut s'agir de : GET, DELETE, HEAD, PUT ou POST.	✓
	outputFileName		Le nom complet du fichier dans lequel la réponse de la ressource Web doit être renvoyée.	
	QueryParametersValue	QueryParametersValue	Le nom du paramètre de requête et sa valeur. Par exemple : <code><jsdlrestful:QueryParametersValue key="colour">blue</jsdlrestful:QueryParametersValue></code> où colour est le nom du paramètre et blue est sa valeur.	

Tableau 85. Attributs obligatoires et facultatifs pour la définition d'un travail de type services Web RESTful (suite)

Attribut				Description/valeur	Obligatoire
Body	contentType			Le type de support du corps d'entité envoyé à la ressource destinataire ; par exemple : json, xml, html. Dans le cas de la méthode HEAD, entrez le type de support qui aurait été envoyé si la demande avait été GET.	
	BodyGroup	FileBody	InputFileName	Le nom du fichier contenant le corps de la demande.	
	Entrez soit le nom du fichier qui contient le corps de la demande, soit directement le texte du corps. Avertissement : Les requêtes DELETE refusent un corps dans la demande. N'entrez pas d'entrée de corps ou la demande échouera.	TextBody	InputTextBody	Le corps de la demande que vous envoyez à l'aide de la méthode HTTP.	
Advanced	HeadersValues	HeadersValue		Le nom de l'en-tête et sa valeur sous la forme : <jsdl:restful:HeadersValue key="Accept-Charset">utf-8</jsdl:restful:HeadersValue>	
	Accept			Le type de support dans lequel la réponse de la ressource Web doit être renvoyée. Généralement, le type spécifié par la ressource à laquelle vous envoyez la demande.	
	JSONPropertiesGroup	JsonObjectResultQuery		Une requête qui extrait la valeur d'une zone spécifique dans une réponse de type JSON. Cette valeur se trouve dans l'élément Résultat JSON dans les propriétés du travail renvoyées une fois le travail terminé et peut être transmise aux autres travaux. Par exemple : query.results.quote.AskRealtime	
	Retry options	NumberOfRetries		Nombre maximal de nouvelles tentatives, en cas de panne de connexion. La valeur par défaut est 0.	
		RetryIntervalSeconds		Temps d'attente entre les nouvelles tentatives en cas de panne de connexion. La valeur par défaut est 30 secondes.	

L'exemple suivant montre la définition d'un travail qui envoie une demande GET au service Web de LB AIRLINES afin d'obtenir la liste des passagers d'un certain vol :

```
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlrestful="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlrestful" name="RESTFUL"
<jsdl:variables>
  <jsdl:stringVariable name="tws.jobstream.name">JOBS</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.jobstream.id">JOBS</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.name">getflightpasslst</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.workstation">FNC050799</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.iawstz">201406290000</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.promoted">N0</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.resourcesForPromoted">10</jsdl:stringVariable>
  <jsdl:stringVariable name="tws.job.num">612741339</jsdl:stringVariable>
</jsdl:variables>
<jsdl:application name="restful">
  <jsdlrestful:restful>
    <jsdlrestful:RestfulParameters>
      <jsdlrestful:Authentication>
        <jsdlrestful:credentials>
          <jsdl:userName>aladmin</jsdl:userName>
          <jsdl:password>{aes}t77HX7Zv9/NwG6DmFbePPdk6d
+JESD0AWRN00XudYeM=</jsdl:password>
        </jsdlrestful:credentials>
      </jsdlrestful:Authentication>
    </jsdlrestful:RestfulParameters>
  </jsdlrestful:restful>
</jsdl:application>
</jsdl:jobDefinition>
```

```

        </jsdlrestful:Authentication>
        <jsdlrestful:RESTAction>
            <jsdlrestful:URI>https://query.1blines.com</jsdlrestful:URI>
            <jsdlrestful:method>GET</jsdlrestful:method>
<jsdlrestful:outputFileName>passnames.xls</jsdlrestful:outputFileName>
            <jsdlrestful:QueryParametersValues>
                <jsdlrestful:QueryParametersValue
key="flight">1b915</jsdlrestful:QueryParametersValue>
                <jsdlrestful:QueryParametersValue
key="day">thursday</jsdlrestful:QueryParametersValue>
                <jsdlrestful:QueryParametersValue
key="time">8:45</jsdlrestful:QueryParametersValue>
            </jsdlrestful:QueryParametersValues>
        </jsdlrestful:RESTAction>
        <jsdlrestful:Body>
<jsdlrestful:contentType>application/json</jsdlrestful:contentType>
            <jsdlrestful:BodyGroup>
                <jsdlrestful:FileBody>
                    <jsdlrestful:InputFileName/>
                </jsdlrestful:FileBody>
            </jsdlrestful:BodyGroup>
        </jsdlrestful:Body>
        <jsdlrestful:Advanced>
            <jsdlrestful:HeadersValues>
                <jsdlrestful:HeadersValue
key="Accept-Charset">utf-8</jsdlrestful:HeadersValue>
                <jsdlrestful:HeadersValue
key="Accept-Language">en-CA</jsdlrestful:HeadersValue>
            </jsdlrestful:HeadersValues>
            <jsdlrestful:Accept>json</jsdlrestful:Accept>
            <jsdlrestful:JSONPropertiesGroup>
<jsdlrestful:JsonObjectResultQuery>query.results.travellerlist</jsdlrestful:JsonObjectResultQuery>
            </jsdlrestful:JSONPropertiesGroup>
        </jsdlrestful:Advanced>
    </jsdlrestful:RestfulParameters>
</jsdlrestful:restful>
</jsdl:application>
<jsdl:resources>
    <jsdl:orderedCandidatedWorkstations>
        <jsdl:workstation>71E6B2B225EF498E916BF57AD817A228</jsdl:workstation>
    </jsdl:orderedCandidatedWorkstations>
</jsdl:resources>
</jsdl:jobDefinition>

```

Planification, arrêt et reconnexion au travail dans IBM Workload Scheduler

Vous planifiez des travaux de type services Web RESTful IBM Workload Scheduler en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Vous pouvez soumettre des travaux dans un environnement z/OS à l'aide de Dynamic Workload Console ou de l'application ISPF.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire à l'aide de la commande **kill**. Lorsque vous arrêtez le travail, IBM Workload Scheduler collecte le journal de travail lorsque l'agent redémarre et affecte le statut **Error** ou **ABEND** au travail.

Si l'agent IBM Workload Scheduler devient non disponible lorsque vous soumettez le travail ou pendant que le travail est en cours d'exécution, dès que l'agent redevient disponible, IBM Workload Scheduler lance la surveillance du travail là où il s'était arrêté.

Propriétés de travail

Vous pouvez voir les propriétés du travail en exécutant conman sj <nom_travail>;props, où <nom_travail> est le nom du travail de type services Web RESTful.

Les propriétés sont répertoriées dans la section Extra Information de la commande de sortie. Les propriétés sont tous les HTTP_HEADER renvoyés par le serveur au client à l'exception de Status Code et de JSON Result. Les noms des propriétés d'en-tête dépendent de la réponse du service RESTful qui varie d'un serveur à l'autre. Certains noms d'en-tête HTTP sont standard, mais la plupart des en-têtes sont facultatifs ou personnalisés.

Contenu du journal de travail

Vous pouvez voir le contenu du journal de travail en exécutant conman sj <nom_travail>;stdlist, où <nom_travail> est le nom du travail de type services Web RESTful.

Vous pouvez visualiser le contenu du journal de travail à partir de Dynamic Workload Console et d'Application Lab. Consultez la section relative à l'analyse du journal de travail dans *Planification des applications avec IBM Workload Automation*.

Scénario métier

NEIGHBOURHOOD BROKERS exécute, à l'heure juste, un flot de travaux qui extrait les cours de bourse actuels de trois sociétés cotées à la bourse de New-York et les affiche sur son site Web. Le flot de travaux s'exécute toutes les heures et il est composé des travaux suivants :

- Un certain nombre de travaux de type services Web RESTful qui extraient à l'aide d'une demande GET la valeur de l'action d'une société. La valeur de l'action est extraite et enregistrée avec les propriétés du travail dans le journal de travail.
- Un travail qui extrait la valeur à partir des propriétés du travail de chaque journal de travail et l'enregistre dans un fichier temporaire.
- Un travail final qui prend ces valeur dans le fichier temporaire et les publie (POST) sur le site Web de NEIGHBOURHOOD BROKERS.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

travaux Java

Un travail Java exécute une classe Java de votre choix.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs des travaux Java. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 86. Attributs obligatoires et facultatifs pour la définition d'un travail Java.

Attribut	Description/valeur	Obligatoire
application name	java	✓
jarPath	Répertoire dans lequel les fichiers jar sont stockés. Sont inclus tous les fichiers jar stockés dans le répertoire et tous les sous-répertoires spécifiés.	
className	Nom de la classe que le travail doit exécuter.	✓
parameter key	Les paramètres à utiliser lors de l'exécution de la classe Java.	

Pour plus d'informations sur le développement d'un travail Java, voir *IBM Workload Automation - Guide du développeur : Extension d'IBM Workload Automation*.

L'exemple suivante présente un travail qui exécute une classe de nom `com.ibm.test.Test` et un paramètre `failExecution` :

```
$JOBS
AGENT#JAVA
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
xmlns:jSDLjava="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDLjava" name="java">
  <jSDL:application name="java">
    <jSDLjava:java>
      <jSDLjava:javaParms>
        <jSDLjava:jarPath>C:\JavaExecutors</jSDLjava:jarPath>
        <jSDLjava:className>com.ibm.test.Test</jSDLjava:className>
        <jSDLjava:parameters>
          <jSDLjava:parameter key="input">failExecution</jSDLjava:parameter>
        </jSDLjava:parameters>
      </jSDLjava:javaParms>
    </jSDLjava:java>
  </jSDL:application>
</jSDL:jobDefinition> DESCRIPTION "Defined using composer."
RECOVERY STOP
```

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux J2EE

Un travail J2EE permet aux applications Java d'un même réseau d'envoyer et de recevoir des messages vers et à partir d'une destination JMS.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

Cette section décrit les attributs obligatoires et facultatifs des travaux J2EE. Chaque définition de travail utilise le format et les arguments suivants :

Tableau 87. Attributs obligatoires et facultatifs pour la définition d'un travail J2EE.

Attribut	Description/valeur	Obligatoire
application name	j2ee	✓
jms operation	L'opération à effectuer. Les valeurs prises en charge sont : • send. Il s'agit de la valeur par défaut. • receive. Si vous indiquez receive, vous pouvez facultativement définir une valeur pour l'attribut timeout.	
timeout	Le délai, exprimé en secondes, dans lequel la tâche doit s'effectuer. Si vous n'indiquez pas de délai ou que vous le définissez sur 0, la tâche se poursuit indéfiniment.	
connectionURL	L'URL de WebSphere Application Server.	
connFactory	Un objet administré qu'un client utilise pour créer une connexion au fournisseur JMS. Pour spécifier la fabrique de connexions, vous pouvez utiliser une expression de variable contenant une ou plusieurs références de variable, par exemple \${var}, ou un caractère, ou une chaîne.	✓
destination	Un objet administré qui encapsule l'identité d'une destination de message, qui est l'emplacement où les messages sont fournis et consommés. Pour spécifier la destination, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne.	✓
message	Le message à envoyer.	✓ Remarque : Cet attribut est obligatoire uniquement pour la tâche d'envoi.

Tableau 87. Attributs obligatoires et facultatifs pour la définition d'un travail J2EE. (suite)

Attribut	Description/valeur	Obligatoire
Credentials	Spécifiez le nom d'utilisateur et le mot de passe à utiliser lors de l'exécution de l'application J2EE. Utilisez cette zone si la sécurité globale est activée sur WebSphere Application Server. L'utilisateur doit être défini sur WebSphere Application Server. Pour spécifier ces données d'identification, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable telles que \${var}, que vous pouvez associer, le cas échéant, à tout caractère ou à une chaîne simple. De plus, vous pouvez paramétrer de l'une des manières suivantes : - Entrez un <i>nom d'utilisateur</i> spécifié dans la base de données avec la définition de l'utilisateur <i>nom d'utilisateur</i> (applicable à tous les systèmes d'exploitation sur ce type de travail) et saisissez l'instruction suivante : <pre><jsd1:password>\${password:nom_utilisateur}</jsd1:password></pre> Le mot de passe est récupéré dans la définition de l'utilisateur <i>nom_utilisateur</i> dans la base de données et résolu au cours de l'exécution. Pour plus d'informations, voir «Utilisation des définitions d'utilisateur sur des types de travaux avec des options avancées», à la page 204. Vous pouvez également spécifier l'utilisateur d'un poste de travail différent et utiliser la syntaxe suivante pour le mot de passe : <pre><jsd1:password>\${password:poste_travail#nom_util}</jsd1:password></pre> - Saisissez un utilisateur et un mot de passe définis avec la commande param localement sur l'agent dynamique qui va exécuter le travail (si le travail consiste à être soumis à un pool ou à un pool dynamique, la définition doit être présente sur tous les agents du pool). A condition que vous ayez défini le nom d'utilisateur avec la variable <i>utilisateur</i> et un mot de passe, les instructions de données d'identification se présenteront ainsi : <pre><jsd1:userName>\${agent:util}</jsd1:userName> <jsd1:password>\${agent:password.util}</jsd1:password></pre> Les variables de l'utilisateur et du mot de passe sont résolues sur l'agent au moment de l'exécution. Pour plus d'informations, voir «Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques», à la page 645.	

L'exemple suivant présente une tâche send qui envoie un message dans la file d'attente MyQueue :

```

$JOBS
AGENT#JOB_NAME_JMS_SEND
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsd1="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsd1"
  xmlns:jsdlj="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlj" name="JMS_JOB_SEND">
<jsd1:application name="j2ee">
<jsd1j:j2ee>
<jsd1j:jms operation="send">
<jsd1j:connectionURL>corbaloc:iiop:washost.mydomain.com:2809</jsd1j:connectionURL>
<jsd1j:connFactory>jms/MyCF</jsd1j:connFactory>
<jsd1j:destination>jms/MyQueue</jsd1j:destination>
<jsd1j:message>Submission of jms job: SEND MESSAGE</jsd1j:message>
</jsd1j:jms>
<jsd1j:credentials>
<jsd1j:userName>jtwoeuser</jsd1j:userName>
<jsd1j:password>${password:jtwoeuser}</jsd1j:password>
</jsd1j:credentials>
</jsd1j:j2ee>
</jsd1:application>
</jsd1:jobDefinition>

```

Remarque : (1) l'utilisateur jtwoeuser a été défini dans la base de données IBM Workload Scheduler à l'aide de la commande de définition d'utilisateur *username*. Le mot de passe associé, spécifié par la chaîne \${password:jtwoeuser} dans la tâche, sera extrait de la base de données au cours de l'exécution.

L'exemple suivant présente une tâche qui lit des messages à partir de la file d'attente MyQueue :

```
$JOBS
AGENT#JOB_NAME_JMS_RECEIVE
TASK
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdlj="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlj" name="JMS_JOB_RECEIVE">
<jsd1:application name="j2ee">
<jsd1j:j2ee>
<jsd1j:jms operation="receive" timeout="180">
<jsd1j:connFactory>jms/MyCF</jsdlj:connFactory>
<jsd1j:destination>jms/MyQueue</jsdlj:destination>
</jsdlj:jms>
<jsd1j:credentials>
<jsd1j:userName>userName</jsdlj:userName>
<jsd1j:password>password</jsdlj:password>
</jsdlj:credentials>
</jsdlj:j2ee>
</jsdl:application>
</jsdl:jobDefinition>
```

Planification et arrêt du travail dans IBM Workload Scheduler

Vous planifiez des travaux IBM Workload Scheduler J2EE en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire en utilisant la commande **kill** à partir de Dynamic Workload Console. Cette action n'est pas pertinente sur l'opération **send**, mais elle est prise en charge sur l'opération **receive**, ce qui permet à l'application Java d'arrêter d'attendre des messages.

Redémarrage du travail

Si l'agent IBM Workload Scheduler s'arrête lorsque vous soumettez le travail J2EE ou pendant que le travail est en cours d'exécution, le travail redémarre automatiquement au redémarrage de l'agent.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

JSR 352 Java Batch

Utilisez le type de travail JSR 352 Java Batch pour planifier l'exécution de programmes batch Java qui implémentent la spécification de programmation standard JSR 352.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type JSR 352 Java Batch. Chaque définition de travail utilise le format et les attributs suivants :

Tableau 88. Attributs obligatoires et facultatifs pour la définition d'un travail de type JSR 352 Java Batch

Attribut	Description et valeur	Obligatoire
hostname	Nom d'hôte du serveur d'applications WebSphere Liberty sur lequel vous voulez exécuter votre application par lots Java.	✓
port	Le numéro du port sur lequel le serveur d'applications est en mode écoute.	✓
protocol	Protocole de connexion au serveur d'applications. Les valeurs admises sont http et https .	✓
user name	Nom d'utilisateur à indiquer pour accéder au serveur d'applications. Cet attribut est facultatif, selon les paramètres de votre serveur.	
mot de passe	Mot de passe à utiliser pour accéder au serveur d'applications. Cet attribut est facultatif, selon les paramètres de votre serveur.	
keystoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion.	Si vous utilisez une connexion HTTPS, assurez-vous que les certificats de sécurité sont configurés pour le gestionnaire de travaux sur le poste de travail sur lequel le travail doit s'exécuter.
keystore file password	Mot de passe qui protège la clé privée et requis pour établir la connexion.	Obligatoire uniquement si vous indiquez un chemin d'accès au fichier de clés.
number of retries	Nombre de fois où le programme est relancé lors de la connexion au serveur d'applications. La valeur par défaut est 0.	
retry interval (seconds)	Nombre de secondes pendant lesquelles le programme attend avant de tenter de nouveau une connexion au serveur d'applications. La valeur par défaut est 30 secondes.	
application name	Nom de l'application JSR 352 Java Batch à démarrer.	✓
module name	Nom du fichier WAR dans lequel est enregistrée l'application JSR 352 Java Batch.	
EJB Component name	Nom du composant EJB dans le module EJB de l'application par lots. S'il est indiqué, le travail est soumis sous le contexte de composant EJB.	Obligatoire uniquement lorsque le module est un module EJB. S'il s'agit d'un module WAR, cet attribut n'est pas obligatoire.
job XML name	Nom du fichier XML du langage de spécification de travail qui sera utilisé pour soumettre ce travail.	

Tableau 88. Attributs obligatoires et facultatifs pour la définition d'un travail de type JSR 352 Java Batch (suite)

Attribut	Description et valeur	Obligatoire
job XML file	Chemin d'accès complet et nom du fichier XML de travail utilisé pour soumettre le travail JSR 352 Java Batch. Si vous indiquez ce paramètre, les instructions du XML de travail spécifiées dans la zone Job XML sont ignorées lors de l'exécution.	
job XML	Entrez les instructions XML de travail à utiliser lors de l'exécution pour soumettre le travail JSR 352 Java Batch. Les instructions XML de travail sont validées lorsque vous sauvegardez le travail IBM Workload Scheduler JSR 352 Java Batch. Les instructions XML de travail sont ignorées si vous indiquez un chemin d'accès à un fichier XML de travail au paramètre Job XML file .	
job parameters	Un ou plusieurs paramètres, et leurs valeurs, éventuellement nécessaires à votre application JSR 352 Java Batch.	

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux JSR 352 Java Batch en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état **EXEC** dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire en utilisant la commande **kill** à partir de Dynamic Workload Console. Cette action arrête également l'exécution du programme sur le serveur d'applications.

Redémarrage d'un travail

Si l'agent IBM Workload Scheduler s'arrête lorsque vous soumettez le travail JSR 352 Java Batch ou pendant que le travail est en cours d'exécution, le travail redémarre automatiquement au redémarrage de l'agent.

Propriétés de travail

Vous pouvez afficher les propriétés du travail en exécutant la commande **conman sj <nom_travail>;props**, où **<nom_travail>** est le nom du travail JSR 352 Java Batch.

L'exemple suivant présente les propriétés que vous pouvez trouver dans la section Informations supplémentaires de la commande de sortie :

```
Cache-Control= no-cache="set-cookie, set-cookie2"
Content-Length= 658
Content-Type= application/json; charset=UTF-8
Date = Thu, 04 Jun 2015 14:36:12 GMT
ExitStatus = COMPLETED
Expires= Thu, 01 Dec 2019 16:00:00 GMT
Job Batch Status= COMPLETED
Job Instance Id= 223
Job Name= sleepy-batchlet
Set-Cookie= LtpaToken2=-2xktDMTwg6MhU6Xm3MbmKgzLc1xFSKwZhw0BM1XpTa+Qvs
Status Code= 200 OK
Step.1.BatchStatus= COMPLETED
Step.1.Name= step1
X-Powered-By= Servlet/3.1
Job Execution Id= 220
```

Job Execution = <https://localhost:9443/ibm/api/batch/jobinstances/223/jobexecutions/220>
Job Logs= <https://localhost:9443/ibm/api/batch/jobinstances/223/joblogs>
Self= <https://localhost:9443/ibm/api/batch/jobinstances/223>

Vous pouvez exporter certaines des propriétés de travail JSR 352 Java Batch que contient la section Informations supplémentaires vers un travail successeur de la même instance de flot de travaux. Pour plus d'informations relatives à la liste des propriétés de travail que vous pouvez exporter, voir tableau 106, à la page 658.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande `conman sj <nom_travail>;stdlist`, où `<nom_travail>` est le nom du travail JSR 352 Java Batch.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

MQTT

Utilisez le type de travail MQTT pour planifier l'exécution des actions **publish** et **subscribe** sur les rubriques gérées par un courtier de messages MQTT.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type MQTT. Chaque définition de travail utilise le format et les attributs suivants :

Tableau 89. Attributs obligatoires et facultatifs pour la définition d'un travail MQTT

Attribut	Description et valeur	Obligatoire
server	Nom d'hôte du poste de travail sur lequel est installé le serveur MQTT.	✓
port number	Numéro de port TCP/IP du poste de travail sur lequel le serveur MQTT est en mode écoute.	✓
use SSL	Version des protocoles cryptographiques SSL que vous voulez utiliser. La valeur par défaut est Aucun.	
user name	Nom d'utilisateur à indiquer pour accéder au serveur MQTT.	Requis, selon votre serveur MQTT.
mot de passe	Mot de passe à indiquer pour accéder au serveur MQTT.	Obligatoire si vous avez indiqué un nom d'utilisateur.
keystoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion.	Requis, selon votre serveur MQTT.

Tableau 89. Attributs obligatoires et facultatifs pour la définition d'un travail MQTT (suite)

Attribut	Description et valeur	Obligatoire
keystore file password	Mot de passe qui protège la clé privée et qui est requis pour établir la connexion.	Obligatoire uniquement si vous indiquez un chemin d'accès au fichier de clés.
client id	Identificateur unique du client MQTT utilisé pour communiquer avec votre serveur MQTT. Le serveur utilise l'identificateur de client pour stocker des données relatives au client.	Requis, selon votre serveur MQTT.
timeout	Durée en secondes pendant laquelle le travail attend la connexion au serveur MQTT. La valeur par défaut est 30 secondes.	
topic	Rubrique à laquelle s'abonner sur le serveur MQTT. Des caractères spéciaux et des caractères génériques pris en charge vous permettent de vous abonner à plusieurs rubriques en même temps. Pour connaître les caractères pris en charge et les instructions spécifiques liés à l'abonnement à des rubriques, voir la documentation du serveur MQTT.	✓
QoS	Qualité de service. Définit à quel point le serveur ou le client tente de garantir qu'un message est reçu. Vous avez le choix entre les options suivantes : 0 Le serveur ou le client distribue le message une seule fois, sans confirmation. 1 Le serveur ou le client distribue le message au moins une fois, avec confirmation requise. 2 Le serveur ou le client distribue le message une fois seulement avec établissement de liaison en quatre temps. La valeur par défaut est 0.	✓
subscribe	Sélectionnez cette option si vous voulez vous abonner à une rubrique spécifiée sur le serveur MQTT.	
publish	Sélectionnez cette option si vous voulez publier un message dans une rubrique sur le serveur MQTT.	
MQTT message	Message que vous voulez publier dans une rubrique sur le serveur MQTT.	Obligatoire si vous sélectionnez l'option publish.

Planification et arrêt d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux MQTT en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Après la soumission, lorsque le travail est en cours d'exécution et qu'il est signalé comme étant à l'état EXEC dans IBM Workload Scheduler, vous pouvez l'arrêter si nécessaire à l'aide de la commande **kill**. Cette action arrête également l'exécution du programme sur le serveur MQTT.

Redémarrage d'un travail

Si l'agent IBM Workload Scheduler s'arrête lorsque vous soumettez le travail MQTT ou pendant que le travail est en cours d'exécution, et s'il s'agit d'une action **subscribe**, le travail redémarre automatiquement dès que l'agent redémarre et tente de nouveau de recevoir un message.

Propriétés de travail

Vous pouvez afficher les propriétés du travail en exécutant la commande `conman sj <nom_travail>;props`, où `<nom_travail>` est le nom du travail MQTT.

La seule propriété que contient la section Informations supplémentaires de la commande de sortie est le texte du message que reçoit l'action **subscribe**.

Vous pouvez également exporter le texte du message que reçoit l'action **subscribe** vers un travail successeur de la même instance de flot de travaux. Pour plus d'informations, voir tableau 107, à la page 659.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande `conman sj <nom_travail>;stdlist`, où `<nom_travail>` est le nom du travail MQTT.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux de Table de variables

Un travail Table de variables ajoute ou modifie une variable dans une commande **variable table** indiquée.

Utilisez le type de travail Table de variables pour ajouter ou modifier une variable dans une **table de variables** spécifiée. Le type de travail Table de variables permet la transmission de variables d'une tâche à l'autre, que ce soit dans un même flot de travaux ou dans un flot de travaux différent.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type Table de variables. Les attributs obligatoires peuvent être spécifiés au moment de la définition de travail ou dans le fichier de propriétés du plug-in des travaux. Chaque définition de travail utilise le format et les attributs suivants :

Tableau 90. Attributs obligatoires et facultatifs pour la définition d'un travail de type Table de variables

Attribut	Description et valeur	Obligatoire
nom d'hôte	Nom d'hôte du poste de travail où le serveur maître IBM Workload Scheduler est installé, et sur lequel vous souhaitez ajouter ou modifier une variable.	✓

Tableau 90. Attributs obligatoires et facultatifs pour la définition d'un travail de type Table de variables (suite)

Attribut	Description et valeur	Obligatoire
port	Numéro de port TCP/IP du poste de travail sur lequel le serveur maître IBM Workload Scheduler est installé.	✓
protocol	Protocole pour la connexion de l'agent dynamique qui exécute la tâche Table de variables et le serveur maître IBM Workload Scheduler. Les valeurs prises en charge sont http et https.	✓
user name	Utilisateur à indiquer pour accéder au serveur maître IBM Workload Scheduler.	
mot de passe	Mot de passe à utiliser pour accéder au serveur maître IBM Workload Scheduler.	Obligatoire si vous avez indiqué un nom d'utilisateur.
keystoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion.	
keystore file password	Mot de passe qui protège la clé privée et requis pour établir la connexion.	
number of retries	Nombre de fois où le programme est relancé lors de la connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 0.	
RetryIntervalSeconds	Nombre de secondes pendant lesquelles le programme attend avant de tenter de nouveau une connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 30 secondes.	
table de variables	Nom de la table de variables sur le serveur maître IBM Workload Scheduler où vous souhaitez ajouter ou modifier une variable.	✓
varListValues	Liste des variables et des valeurs connexes que vous souhaitez ajouter ou modifier, dans la table de variables sélectionnée. Indiquez au moins une variable.	✓

L'exemple suivant montre la définition d'un travail Table de variables :

```
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
xmlns:jSDLvariabletable="http://www.ibm.com/xmlns/prod/scheduling
/1.0/jSDLvariabletable"
name="VARIABLETABLE">
<jSDL:application name="variabletable">
<jSDLvariabletable:variabletable>
<jSDLvariabletable:VariableTableParameters>
<jSDLvariabletable:Connection>
<jSDLvariabletable:connectionInfo>
<jSDLvariabletable:hostname>nx000140.xxxxlab.yy.zzz.com
</jSDLvariabletable:hostname>
<jSDLvariabletable:port>35116</jSDLvariabletable:port>
<jSDLvariabletable:protocol>https</jSDLvariabletable:protocol>
<jSDLvariabletable:credentials>
<jSDL:user name>twS_user</jSDL:user name>
<jSDL:password>{aes}/2GNMAY8Z2pSx6JXHqcbKwd2xxxxxkyKXD/WNgtHhNw=
</jSDL:password>
</jSDLvariabletable:credentials>
<jSDLvariabletable:CertificateGroup>
<jSDLvariabletable:keyStoreFilePath/>
<jSDLvariabletable:keyStorePassword/>
</jSDLvariabletable:CertificateGroup>
</jSDLvariabletable:connectionInfo>
<jSDLvariabletable:retryOptions>
<jSDLvariabletable:NumberOfRetries>0</jSDLvariabletable:NumberOfRetries>
<jSDLvariabletable:RetryIntervalSeconds>30</jSDLvariabletable:
RetryIntervalSeconds>
</jSDLvariabletable:retryOptions>
</jSDLvariabletable:Connection>
<jSDLvariabletable:Action>
<jSDLvariabletable:actionInfo>
<jSDLvariabletable:varTable>TABLE1</jSDLvariabletable:varTable>
<jSDLvariabletable:varListValues>
<jSDLvariabletable:varListValue key="DB2Name">HCLDB
</jSDLvariabletable:varListValue>
```



```

|         </jsdlvariabletable:varListValues>
|     </jsdlvariabletable:actionInfo>
| </jsdlvariabletable:Action>
| </jsdlvariabletable:VariableTableParameters>
| </jsdlvariabletable:variabletable>
| </jsdl:application>
| </jsdl:jobDefinition>

```

Planification d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux Table de variables en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Arrêt et redémarrage d'un travail

L'arrêt et le redémarrage d'un travail Table de variables ne sont pas pris en charge.

Fichier Table de variablesJobExecutor.properties

Le fichier de propriétés est automatiquement généré soit lorsque vous exécutez un "Test de connexion" à partir de Dynamic Workload Console dans les panneaux de définition de travail, soit lorsque vous soumettez le travail pour une première exécution. Une fois le fichier créé, vous pouvez le personnaliser. Cela s'avère pratique lorsqu'il vous faut planifier plusieurs travaux du même type. Vous pouvez indiquer les valeurs dans le fichier de propriétés et éviter de devoir fournir des informations, telles que les données d'identification, pour chaque travail. Vous pouvez substituer ces valeurs dans le fichier de propriétés en définissant différentes valeurs lors de la définition de travail.

Le fichier TWS_INST_DIR\TWS\JavaExt\cfg\VariableTableJobExecutor.properties contient les propriétés suivantes :

```

| #Variable Table properties
| hostname=
| port=
| protocol=http
| user=
| password=
| keyStoreFilePath=
| keyStorePassword=
| HostnameVerifyCheckbox=false
| NumberOfRetries=0
| RetryIntervalSeconds=30
| varTable=
| #add here the variables in the format VARLISTPROPERTY.<variable_name>=<variable_value>
| #For example VARLISTPROPERTY.queueName=default

```

Propriétés de travail

Vous pouvez afficher les propriétés du travail en exécutant la commande **conman sj <nom_travail>;props**, où **<nom_travail>** est le nom du travail Table de variables.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande **conman sj <nom_travail>;stdlist**, où **<nom_travail>** est le nom du travail Table de variables.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux Gestion des travaux

Un travail Gestion des travaux exécute des actions sur un travail dans un flot de travaux.

Utilisez le type de travail Gestion des travaux pour exécuter des actions sur un travail dans un flot de travaux.

Les actions que vous pouvez effectuer sur un travail sont les suivantes :

- Réexécuter
- Réexécuter le travail et tous ses travaux successeurs
- Réexécuter le travail et ses travaux successeurs dans le même flot de travaux
- Libérer
- Libérer les dépendances
- Annuler
- Annuler en instance
- Mettre en attente
- Supprimer (kill)
- Confirmer ABEND
- Confirmer SUCC

Pour plus d'informations sur les actions que vous pouvez exécuter sur un travail par le biais de Dynamic Workload Console, voir

à la section Contrôle du traitement des travaux et des flots de travaux du document *Dynamic Workload Console - Guide d'utilisation*.

Pour plus d'informations sur les actions que vous pouvez exécuter sur un travail par le biais de la ligne de commande **conman**, voir

la section Gestion des objets dans le plan - **conman** du document *IBM Workload Scheduler - Guide d'utilisation et de référence*.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type Gestion des travaux. Les attributs obligatoires doivent être spécifiés lors de la

définition des travaux. Aucun fichier de propriétés n'est disponible pour ce plug-in. Chaque définition de travail utilise le format et les attributs suivants :

Tableau 91. Attributs obligatoires et facultatifs pour la définition d'un travail de type Gestion des travaux

Attribut	Description et valeur	Obligatoire
URL	Variable spécifiant l'adresse Web du serveur maître IBM Workload Scheduler. Vous pouvez remplacer cette variable par l'adresse Web du serveur maître et toute copie de sauvegarde maître IBM Workload Scheduler (au cas où le maître n'est pas disponible). Utilisez une virgule ou un point-virgule pour séparer les différentes adresses que vous spécifiez. Si vous ne substituez pas cette variable lors de la définition du travail, la variable est résolue automatiquement lors de la phase d'exécution. Dans ce cas, si la connexion au serveur maître n'est pas disponible, le plug-in tente de se connecter automatiquement à l'un des maîtres de sauvegarde.	✓
userName	Utilisateur à indiquer pour accéder au serveur maître IBM Workload Scheduler.	
mot de passe	Mot de passe à utiliser pour accéder au serveur maître IBM Workload Scheduler.	Obligatoire si vous avez indiqué un nom d'utilisateur.
keyStoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut doit être égal à : <code>\${agent-config:keystore-file}</code> Vous pouvez également spécifier cet attribut au format suivant : <code>[<type-magasin-clés>::]<chemin-magasin-clés></code>	
keyStorePassword	Mot de passe qui protège la clé privée et requis pour établir la connexion. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut doit être égal à : <code>\${agent-config:keystore-file-password}</code>	
HostnameVerifyCheckbox	Utilisez cet attribut pour exiger que la syntaxe du nom de serveur maître IBM Workload Scheduler, telle qu'indiquée dans le fichier de clés, corresponde exactement à l'URL. S'ils ne correspondent pas, aucune autorisation d'accès au serveur n'est accordée. Si cet attribut n'est pas indiqué, le contrôle n'est pas imposé. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut ne peut pas être spécifié.	
NumberOfRetries	Nombre de fois où le programme est relancé lors de la connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 0.	
RetryIntervalSeconds	Nombre de secondes pendant lesquelles le programme attend avant de tenter de nouveau une connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 30 secondes.	
nom_travail	Nom du travail sur lequel vous souhaitez exécuter l'action.	✓
workstation	Variable spécifiant le nom du poste de travail sur lequel le travail est exécuté. Vous pouvez remplacer cette variable par un nom de poste de travail. Si vous ne substituez pas cette variable lors de la définition du travail, la variable est résolue automatiquement lors de la phase d'exécution.	✓
jobstreamid	Variable spécifiant le nom du flot de travaux contenant le travail. Vous pouvez remplacer cette variable par un nom de flot de travaux. Si vous ne substituez pas cette variable lors de la définition du travail, la variable est résolue automatiquement lors de la phase d'exécution.	✓
method	Action que vous souhaitez exécuter sur le travail. Les valeurs valides sont les suivantes : • rerun • rerunsuccessors • reruninternalsuccessors • release • releasedependencies • cancel • cancelpending • hold • kill • confirm_abend • confirm_succ	✓
sameworkstation	Spécifiez ce paramètre pour l'action de réexécution uniquement. Utilisez-le si vous souhaitez réexécuter le travail sur le même poste de travail que celui sur lequel il a été exécuté précédemment. Ce paramètre s'applique uniquement aux postes de travail en pool et en pool dynamique.	
condition	Nom de la condition pour confirmer le statut SUCC ou ABEND pour les conditions de sortie indiquées. Toutes conditions non spécifiées sont définies sur non satisfaite.	

L'exemple suivant présente la définition d'un travail Gestion des travaux qui exécute l'action **rerun** :

```
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.abc.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdljobmanagement="http://www.abc.com/xmlns/prod/
scheduling/1.0/jsdljobmanagement"
name="JOBMANAGEMENT">
<jsd1:application name="jobmanagement">
<jsd1jobmanagement:jobmanagement>
  <jsd1jobmanagement:JobManagementParameters>
    <jsd1jobmanagement:Connection>
      <jsd1jobmanagement:connectionInfo>
        <jsd1jobmanagement:Url>${agent-config:master-address}</jsdljobmanagement:Url>
        <jsd1jobmanagement:credentials>
          <jsd1:userNametwsuser1</jsdl:userNamet>
          <jsd1:password>{aes}ywIpc7ISIQSq9xb7xrzqxoYJn04rNj/d1IfLa20r7Rg=
            </jsdl:password>
        </jsdljobmanagement:credentials>
        <jsd1jobmanagement:CertificateGroup>
          <jsd1jobmanagement:keyStoreFilePath></jsdljobmanagement:keyStoreFilePath>
          <jsd1jobmanagement:keyStorePassword></jsdljobmanagement:keyStorePassword>
          <jsd1jobmanagement:HostnameVerifyCheckbox/>
        </jsdljobmanagement:CertificateGroup>
      </jsdljobmanagement:connectionInfo>
      <jsd1jobmanagement:retryOptions>
        <jsd1jobmanagement:NumberOfRetries>0</jsdljobmanagement:NumberOfRetries>
        <jsd1jobmanagement:RetryIntervalSeconds>30</jsdljobmanagement:RetryIntervalSeconds>
      </jsdljobmanagement:retryOptions>
    </jsdljobmanagement:Connection>
    <jsd1jobmanagement:Action>
      <jsd1jobmanagement:informations>
        <jsd1jobmanagement:jobname>JOBDIR</jsdljobmanagement:jobname>
        <jsd1jobmanagement:workstation>LAPTOP-E0DIBP1_2 (type: Agent, version: 9.4.0.01)
          </jsdljobmanagement:workstation>
        <jsd1jobmanagement:jobstreamid>${tws.jobstream.id}</jsdljobmanagement:jobstreamid>
      </jsdljobmanagement:informations>
      <jsd1jobmanagement:actions>
        <jsd1jobmanagement:method>rerun</jsdljobmanagement:method>
      </jsdljobmanagement:actions>
      <jsd1jobmanagement:options>
        <jsd1jobmanagement:sameworkstation/>
        <jsd1jobmanagement:condition></jsdljobmanagement:condition>
      </jsdljobmanagement:options>
    </jsdljobmanagement:Action>
  </jsdljobmanagement:JobManagementParameters>
</jsdljobmanagement:jobmanagement>
</jsdl:application>
</jsdl:jobDefinition>
```

L'exemple suivant présente la définition d'un travail Gestion des travaux qui exécute l'action **rerunsuccessors** :

```
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.abc.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdljobmanagement="http://www.abc.com/xmlns/prod/
scheduling/1.0/jsdljobmanagement"
name="JOBMANAGEMENT">
<jsd1:application name="jobmanagement">
<jsd1jobmanagement:jobmanagement>
  <jsd1jobmanagement:JobManagementParameters>
    <jsd1jobmanagement:Connection>
      <jsd1jobmanagement:connectionInfo>
        <jsd1jobmanagement:Url>https://localhost:31116</jsdljobmanagement:Url>
        <jsd1jobmanagement:credentials>
          <jsd1:userNametwsuser1</jsdl:userNamet>
          <jsd1:password>{aes}ywIpc7ISIQSq9xb7xrzqxoYJn04rNj/d1IfLa20r7Rg=
```

```

=>         </jsdl:password>
=>     </jsdljobmanagement:credentials>
=>     <jsdljobmanagement:CertificateGroup>
=>         <jsdljobmanagement:keyStoreFilePath></jsdljobmanagement:keyStoreFilePath>
=>         <jsdljobmanagement:keyStorePassword></jsdljobmanagement:keyStorePassword>
=>         <jsdljobmanagement:HostnameVerifyCheckbox/>
=>     </jsdljobmanagement:CertificateGroup>
=> </jsdljobmanagement:connectionInfo>
=> <jsdljobmanagement:retryOptions>
=>     <jsdljobmanagement:NumberOfRetries>0</jsdljobmanagement:NumberOfRetries>
=>     <jsdljobmanagement:RetryIntervalSeconds>30</jsdljobmanagement:RetryIntervalSeconds>
=> </jsdljobmanagement:retryOptions>
=> </jsdljobmanagement:Connection>
=>     <jsdljobmanagement:Action>
=> <jsdljobmanagement:informations>
=>     <jsdljobmanagement:jobname>JOBDIR</jsdljobmanagement:jobname>
=>     <jsdljobmanagement:workstation>LAPTOP-E0DIBP1_2 (type: Agent, version: 9.4.0.01)
=>     </jsdljobmanagement:workstation>
=>     <jsdljobmanagement:jobstreamid>${tw.s.jobstream.id}</jsdljobmanagement:jobstreamid>
=> </jsdljobmanagement:informations>
=> <jsdljobmanagement:actions>
=>     <jsdljobmanagement:method>rerunsuccessors</jsdljobmanagement:method>
=> </jsdljobmanagement:actions>
=> <jsdljobmanagement:options>
=>     <jsdljobmanagement:condition></jsdljobmanagement:condition>
=> </jsdljobmanagement:options>
=>     </jsdljobmanagement:Action>
=> </jsdljobmanagement:JobManagementParameters>
=> </jsdljobmanagement:jobmanagement>
=> </jsdl:application>
=> </jsdl:jobDefinition>

```

L'exemple suivant présente la définition d'un travail Gestion des travaux qui exécute l'action **confirm_succ** :

```

=> <?xml version="1.0" encoding="UTF-8"?>
=> <jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
=> xmlns:jsdljobmanagement="http://www.ibm.com/xmlns/prod/
=> scheduling/1.0/jsdljobmanagement"
=> name="JOBMANAGEMENT">
=> <jsdl:application name="jobmanagement">
=> <jsdljobmanagement:jobmanagement>
=>     <jsdljobmanagement:JobManagementParameters>
=>         <jsdljobmanagement:Connection>
=> <jsdljobmanagement:connectionInfo>
=>     <jsdljobmanagement:Url>https://localhost:31116</jsdljobmanagement:Url>
=>     <jsdljobmanagement:credentials>
=>         <jsdl:userNam>twuser1</jsdl:userNam>
=>         <jsdl:password>{aes}ywIpc7ISIQS9xb7xrzqxoYJn04rNj/d1IfLa20r7Rg=
=>         </jsdl:password>
=>     </jsdljobmanagement:credentials>
=>     <jsdljobmanagement:CertificateGroup>
=>         <jsdljobmanagement:keyStoreFilePath></jsdljobmanagement:keyStoreFilePath>
=>         <jsdljobmanagement:keyStorePassword></jsdljobmanagement:keyStorePassword>
=>     </jsdljobmanagement:CertificateGroup>
=> </jsdljobmanagement:connectionInfo>
=> <jsdljobmanagement:retryOptions>
=>     <jsdljobmanagement:NumberOfRetries>0</jsdljobmanagement:NumberOfRetries>
=>     <jsdljobmanagement:RetryIntervalSeconds>30</jsdljobmanagement:RetryIntervalSeconds>
=> </jsdljobmanagement:retryOptions>
=>     </jsdljobmanagement:Connection>
=>     <jsdljobmanagement:Action>
=> <jsdljobmanagement:informations>
=>     <jsdljobmanagement:jobname>JOBDIR</jsdljobmanagement:jobname>
=>     <jsdljobmanagement:workstation>LAPTOP-E0DIBP1_2 (type: Agent, version: 9.4.0.01)
=>     </jsdljobmanagement:workstation>
=>     <jsdljobmanagement:jobstreamid>${tw.s.jobstream.id}</jsdljobmanagement:jobstreamid>

```

```

=> </jsdljobmanagement:informations>
=> <jsdljobmanagement:actions>
=> <jsdljobmanagement:method>confirm_succ</jsdljobmanagement:method>
=> </jsdljobmanagement:actions>
=> <jsdljobmanagement:options>
=> <jsdljobmanagement:condition>CONF_SUCC_CONDITION</jsdljobmanagement:condition>
=> </jsdljobmanagement:options>
=> </jsdljobmanagement:Action>
=> </jsdljobmanagement:JobManagementParameters>
=> </jsdljobmanagement:jobmanagement>
=> </jsdl:application>
=> </jsdl:jobDefinition>

```

Planification d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux Gestion des travaux en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Arrêt et redémarrage d'un travail

L'arrêt et le redémarrage d'un travail Gestion des travaux ne sont pas pris en charge.

Propriétés de travail

Vous pouvez afficher les propriétés du travail en exécutant la commande **conman sj <nom_travail>;props**, où **<nom_travail>** est le nom du travail Gestion des travaux.

Vous pouvez exporter certaines des propriétés de travail Gestion des travaux figurant dans la section Informations supplémentaires de la commande de sortie, vers un travail successeur dans le même flot de travaux ou un flot de travaux différent. Pour plus d'informations relatives à la liste des propriétés de travail que vous pouvez exporter, voir tableau 111, à la page 661.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande **conman sj <nom_travail>;stdlist**, où **<nom_travail>** est le nom du travail Gestion des travaux.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Travaux Soumission d'un flot de travaux

Un travail Soumission d'un flot de travaux soumet un flot de travaux en vue de son traitement.

Le travail Soumission d'un flot de travaux est l'un des utilitaires d'automatisation qui facilitent des opérations IBM Workload Scheduler spécifiques. Utilisez le type de travail Soumission d'un flot de travaux pour automatiser la soumission d'un flot de travaux à des fins de traitement.

La description des propriétés du travail et les valeurs admises sont fournies dans l'aide contextuelle de Dynamic Workload Console en cliquant sur l'icône point d'interrogation (?) dans le coin supérieur droit de la sous-fenêtre des propriétés.

La présente section décrit les attributs obligatoires et facultatifs des travaux de type Soumission d'un flot de travaux. Les attributs obligatoires doivent être spécifiés lors de la définition des travaux. Aucun fichier de propriétés n'est disponible pour ce plug-in. Chaque définition de travail utilise le format et les attributs suivants :

Tableau 92. Attributs obligatoires et facultatifs pour la définition d'un travail de type Soumission d'un flot de travaux

Attribut	Description et valeur	Obligatoire
URL	Variable spécifiant l'adresse Web du serveur maître IBM Workload Scheduler. Vous pouvez remplacer cette variable par l'adresse Web du serveur maître et toute copie de sauvegarde maître IBM Workload Scheduler (au cas où le maître n'est pas disponible). Utilisez une virgule ou un point-virgule pour séparer les différentes adresses que vous spécifiez. Si vous ne substituez pas cette variable lors de la définition du travail, la variable est résolue automatiquement lors de la phase d'exécution. Dans ce cas, si la connexion au serveur maître n'est pas disponible, le plug-in tente de se connecter automatiquement à l'un des maîtres de sauvegarde.	✓
userName	Utilisateur à indiquer pour accéder au serveur maître IBM Workload Scheduler.	
mot de passe	Mot de passe à utiliser pour accéder au serveur maître IBM Workload Scheduler.	Obligatoire si vous avez indiqué un nom d'utilisateur.
keyStoreFilePath	Chemin d'accès complet du fichier de clés contenant la clé privée utilisée pour établir la connexion. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut doit être égal à : <code>\${agent-config:keystore-file}</code> Vous pouvez également spécifier cet attribut au format suivant : <code>[<type-magasin-clés>:]<chemin-magasin-clés></code>	
keyStorePassword	Mot de passe qui protège la clé privée et requis pour établir la connexion. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut doit être égal à : <code>\${agent-config:keystore-file-password}</code>	
Case HostnameVerify	Utilisez cet attribut pour exiger que la syntaxe du nom de serveur maître IBM Workload Scheduler, telle qu'indiquée dans le fichier de clés, corresponde exactement à l'URL. S'ils ne correspondent pas, aucune autorisation d'accès au serveur n'est accordée. Si cet attribut n'est pas indiqué, le contrôle n'est pas imposé. Remarque : Pour les utilisateurs de Workload Automation on Cloud, cet attribut ne peut pas être spécifié.	
NumberOfRetries	Nombre de fois où le programme est relancé lors de la connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 0.	
RetryIntervalSeconds	Nombre de secondes pendant lesquelles le programme attend avant de tenter de nouveau une connexion au serveur maître IBM Workload Scheduler. La valeur par défaut est 30 secondes.	
specifyjobstream	Permet de définir le flot de travaux que vous souhaitez soumettre à des fins de traitement.	
workstation	Nom du poste de travail sur lequel le flot de travaux a été défini.	✓
jobstreamname	Nom du flot de travaux.	✓
alias	Nom unique à attribuer au flot de travaux à la place de jobstreamname.	
resubcurrjobstream	Permet de soumettre à nouveau le flot de travaux en cours. Cet attribut est une alternative à specifyjobstream.	
startafter	Permet de définir un décalage en heures pour l'heure de début.	

= *Tableau 92. Attributs obligatoires et facultatifs pour la définition d'un travail de type Soumission d'un flot de*
 = *travaux (suite)*

Attribut	Description et valeur	Obligatoire
delayforhours	Le décalage est calculé à partir de l'heure de soumission du travail Soumission d'un flot de travaux. Les valeurs possibles sont comprises entre 00:00 et 23:59. La valeur par défaut est 00:00.	
startat	Permet de définir l'heure avant laquelle le flot de travaux ne doit pas démarrer. Cet attribut est une alternative à startafter.	
time	Heure avant laquelle le flot de travaux ne doit pas démarrer. Les valeurs possibles sont comprises entre 00:00 et 23:59.	
delayfordays	Vous pouvez spécifier un décalage en jours pour l'heure de début. Le décalage est calculé à partir du jour de soumission du travail Soumission d'un flot de travaux. La valeur par défaut est 0.	
variabletablename	Nom de la table de variables devant être utilisée par le flot de travaux.	
variablelistValues	Liste des variables dans la table de variable et leurs valeurs.	

L'exemple suivant montre la définition d'un travail Soumission d'un flot de travaux :

```
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
  xmlns:jsdljobstreamsubmission="http://www.ibm.com/xmlns/prod/
scheduling/1.0/jsdljobstreamsubmission"
  name="JOBSTREAMSUBMISSION">
  <jsd1:application name="jobstreamsubmission">
  <jsd1jobstreamsubmission:jobstreamsubmission>
  <jsd1jobstreamsubmission:JobstreamSubmissionParameters>
  <jsd1jobstreamsubmission:Connection>
  <jsd1jobstreamsubmission:connectionInfo>
  <jsd1jobstreamsubmission:Url>${agent-config:master-address}</jsdljobstreamsubmission:Url>
  <jsd1jobstreamsubmission:credentials>
  <jsd1:userName>twuser</jsdl:userName>
  <jsd1:password>{aes}FKFBcQTNuxyPMYe+hqRGdXC2Ya7chYe8rk2Ia80JDNY=
  </jsdl:password>
  </jsdljobstreamsubmission:credentials>
  <jsd1jobstreamsubmission:CertificateGroup>
  <jsd1jobstreamsubmission:keyStoreFilePath>
  </jsdljobstreamsubmission:keyStoreFilePath>
  <jsd1jobstreamsubmission:keyStorePassword>
  </jsdljobstreamsubmission:keyStorePassword>
  <jsd1jobstreamsubmission:HostnameVerifyCheckbox/>
  </jsdljobstreamsubmission:CertificateGroup>
  </jsdljobstreamsubmission:connectionInfo>
  <jsd1jobstreamsubmission:retryOptions>
  <jsd1jobstreamsubmission:NumberOfRetries>0</jsdljobstreamsubmission:NumberOfRetries>
  <jsd1jobstreamsubmission:RetryIntervalSeconds>30
  </jsdljobstreamsubmission:RetryIntervalSeconds>
  </jsdljobstreamsubmission:retryOptions>
  </jsdljobstreamsubmission:Connection>
  <jsd1jobstreamsubmission:Action>
  <jsd1jobstreamsubmission:method>
  <jsd1jobstreamsubmission:specifyjobstream>
  <jsd1jobstreamsubmission:workstation>LAPTOP-E0DIBP1_3
  </jsdljobstreamsubmission:workstation>
  <jsd1jobstreamsubmission:jobstreamname>JS1</jsdljobstreamsubmission:jobstreamname>
  <jsd1jobstreamsubmission:alias>JSalias</jsdljobstreamsubmission:alias>
  </jsdljobstreamsubmission:specifyjobstream>
  </jsdljobstreamsubmission:method>
  <jsd1jobstreamsubmission:earlieststart>
  <jsd1jobstreamsubmission:timeoptions>
  <jsd1jobstreamsubmission:startafter>
  <jsd1jobstreamsubmission:delayforhours>00:10</jsdljobstreamsubmission:delayforhours>
  </jsdljobstreamsubmission:startafter>
  </jsdljobstreamsubmission:timeoptions>
  </jsdljobstreamsubmission:earlieststart>
  <jsd1jobstreamsubmission:managevariabletable>
  <jsd1jobstreamsubmission:variablename></jsdljobstreamsubmission:variablename>
  </jsdljobstreamsubmission:managevariabletable>
```



```

=> </jsdljobstreamsubmission:Action>
=> </jsdljobstreamsubmission:JobstreamSubmissionParameters>
=> </jsdljobstreamsubmission:jobstreamsubmission>
=> </jsdl:application>
=> </jsdl:jobDefinition>

```

Planification d'un travail dans IBM Workload Scheduler

Vous planifiez des travaux Soumission d'un flot de travaux en les définissant dans des flots de travaux. Ajoutez le travail à un flot de travaux avec tous les arguments de planification nécessaires et soumettez-le.

Vous pouvez soumettre des travaux à l'aide de Dynamic Workload Console, d'Application Lab ou de la ligne de commande **conman**.

Arrêt et redémarrage d'un travail

L'arrêt et le redémarrage d'un travail Soumission d'un flot de travaux ne sont pas pris en charge.

Propriétés de travail

Pour savoir comment afficher les propriétés de travail depuis les différentes interfaces prises en charge, voir section sur l'analyse du journal de travail dans *Planification des applications avec IBM Workload Automation*. Par exemple, vous pouvez afficher les propriétés du travail en exécutant la commande **conman sj <nom_travail>;props**, où **<nom_travail>** est le nom du travail Soumission d'un flot de travaux.

Vous pouvez exporter certaines des propriétés de travail Soumission d'un flot de travaux figurant dans la section Informations supplémentaires de la commande de sortie, vers un travail successeur dans le même flot de travaux ou un flot de travaux différent. Pour plus d'informations relatives à la liste des propriétés de travail que vous pouvez exporter, voir tableau 112, à la page 661.

Contenu du journal de travail

Pour afficher le contenu du journal de travail, exécutez la commande **conman sj <nom_travail>;stdlist**, où **<nom_travail>** est le nom du travail Soumission d'un flot de travaux.

Voir aussi

A partir de Dynamic Workload Console, vous pouvez effectuer la même tâche que celle décrite dans

Dynamic Workload Console - Guide d'utilisation, section "Création de définitions de travail".

Pour plus d'informations sur la manière de créer et d'éditer des objets de planification, voir

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Codes retour

La liste suivante répertorie les codes retour pour les types de travaux avec options avancées

Travaux de base de données :

CR = 0 -> Le travail s'est terminé avec succès

CR = -1 -> L'instruction SQL a été exécutée avec un code de sortie différent de 1

CR = -2 -> Erreur de travail MSSQL

CR = -3 -> L'instruction SQL n'a pu être exécutée en raison d'une erreur dans l'instruction

Travaux de transfert de fichiers :

RC = 0 -> Le transfert de fichier a réussi.

RC = -1 -> Le transfert de fichier n'est pas effectué. Le travail échoue avec le code d'erreur suivant : AWKFTE007E

Explication : une erreur s'est produite lors du transfert de fichier.

Causes possibles : fichier distant introuvable ou droits refusés.

RC = -2 -> Le transfert de fichier n'est pas effectué. Le travail échoue avec le code d'erreur suivant : AWKFTE020E

Explication : uniquement pour les protocoles SSH ou WINDOWS. Une erreur a été renvoyée pendant la tentative de conversion de la page de codes.

Causes possibles : pour les protocoles SSH ou Windows, la page de codes est automatiquement détectée et convertie. Dans ce cas, une erreur se trouve dans la page de code du fichier à transférer et la page de codes du système local n'est pas respectée.

RC = -3 -> Le transfert de fichier n'est pas effectué. Le travail échoue avec le code d'erreur suivant : AWKFTE015E

Explication : une erreur s'est produite lors du transfert de fichier.

Causes possibles : fichier local introuvable.

RC = -4 -> Le transfert de fichier est effectué avec la page de codes par défaut. Le travail échoue avec le code d'erreur suivant : AWKFTE023E

Explication : la conversion de la page de codes spécifiée n'a pas été effectuée. Le transfert de fichier a été effectué avec les pages de code par défaut.

Causes possibles : la page de codes spécifiée est indisponible.

Travaux IBM i :

Code retour = code retour utilisateur lors de la récupération

Code retour = 0 -> Le travail s'est terminé avec succès

Code retour > -1 -> travail terminé avec succès

Travaux Java :

CR = 0 -> Le travail s'est terminé avec succès

CR = -1 -> En raison d'une exception rencontrée, l'application Java lancée par le travail a échoué

Travaux de services Web :

CR = 0 -> Le travail s'est terminé avec succès

CR = -1 -> Le nom d'hôte du serveur contenu dans l'URL du service Web est inconnu

CR = -2 -> Appel d'erreur du service Web

Lorsque le code retour utilisateur est récupéré, IBM i l'agent de surveillance lui assigne une priorité.

Installation automatique de plug-in en exécutant des types de travaux avec options avancées sur des agents dynamiques

Installation automatique de plug-in en exécutant des types de travaux avec options avancées sur des agents dynamiques

Dans les versions antérieures de IBM Workload Scheduler, lorsque vous définissez un type de travail avec options avancées sur un poste de travail d'agent dynamique, si le plug-in qui doit exécuter le travail n'est pas installé sur cet agent, le travail échoue et vous devez installer le plug-in manuellement, puis réexécuter le travail. Si la version du plug-in est antérieure à la version installée dans le gestionnaire de domaine maître, vous installez manuellement la version plus récente du plug-in.

Depuis IBM Workload Scheduler V9.3, si vous définissez un type de travail avec options avancées sur un poste de travail d'agent dynamique, lorsque le travail s'exécute, les scénarios suivants se présentent :

Le plug-in n'est pas installé sur cet agent :

L'agent dynamique se connecte automatiquement au poste de travail du gestionnaire de domaine maître, il télécharge, installe et configure le plug-in requis sur le poste de travail d'agent dynamique. Après l'installation, le type de travail avec options avancées s'exécute correctement.

Le plug-in est installé sur cet agent, mais la version du plug-in est plus ancienne :

L'agent dynamique se connecte automatiquement au poste de travail du gestionnaire de domaine maître, il télécharge et installe la version plus récente du plug-in sur le poste de travail d'agent dynamique. Le poste de travail d'agent dynamique n'est pas relancé, même si le plug-in est déjà démarré.

Définition des variables et mots de passe pour la résolution locale sur des agents dynamiques

Pour les types de travaux avec des options avancées, vous pouvez autoriser la définition et la résolution locales des variables et des mots de passe sur des agents dynamiques (y compris les pools et les pools dynamiques).

Cela est particulièrement utile dans le cas des mots de passe car il n'est pas nécessaire de les indiquer dans la définition du travail. L'avantage réside dans le fait que, si le mot de passe doit être changé, vous ne modifiez pas la définition du travail mais changez le mot de passe à l'aide de la commande param en local sur les agents (ou sur les agents du pool) qui exécutent ou peuvent exécuter le travail. Si le travail doit être soumis sur un pool ou un pool dynamique, vous pouvez copier le fichier avec les définitions de variable sur tous les agents membre de ce pool, de sorte que les variables sont résolues en local quel que soit l'emplacement d'exécution du travail.

Cette fonction n'est pas limitée aux postes de travail Windows. Vous pouvez l'utiliser également sous UNIX, à condition de l'appliquer à des types de travaux avec options avancées.

Pour définir une variable ou un mot de passe en local sur un agent dynamique, utilisez la commande d'utilitaire param. Cette commande à le pouvoir de créer,

supprimer et répertorier des variables locales sur des agents dynamiques. Pour en savoir plus sur son utilisation, reportez-vous aux détails de la commande.

Indication de variables et de mots de passe locaux dans les définitions de travail

Après avoir défini une variable et sa valeur avec la commande `param` pour l'ajouter dans une définition de travail de sorte qu'elle est résolue localement sur l'agent lors de l'exécution, utilisez la syntaxe suivante :

```
${agent:variable_name}
```

Après avoir défini un mot de passe avec la commande `param` pour l'ajouter dans une définition de travail de sorte qu'il est résolu localement sur l'agent lors de l'exécution, utilisez la syntaxe suivante :

```
${agent:password.user_name}
```

Vous pouvez imbriquer des variables dans les mots de passe. Si vous avez utilisé une variable pour définir un utilisateur, entrez le mot de passe comme suit dans la définition du travail :

```
${agent:password.${agent:variable_name}}
```

où *variable_name* a été défini précédemment comme ayant la valeur *user_name* avec la commande `param`.

Exemple

Un administrateur IBM Workload Scheduler doit définir une tâche de transfert de fichier qui télécharge un fichier depuis un serveur distant vers l'un des agents dynamiques constituant un pool d'agents. L'administrateur souhaite paramétrer dans la définition du travail :

- Le nom avec lequel le fichier distant va être enregistré localement
- L'utilisateur distant et son mot de passe

L'administrateur procède de la manière suivante sur l'un des agents :

1. Il définit une variable nommée `localfile`. La variable est égale à `./npp.5.1.1.Installer.DW.exe` et elle est créée dans un fichier nommé `FTPvars` (aucune section). La commande exécute :

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..localfile ./npp.5.1.1.Installer.DW.exe
```

2. Il définit une variable nommée `remoteUser`. Elle est égale à `FTPuser` et elle est créée dans le fichier `FTPvars` (aucune section). La commande exécute :

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars..remoteUser FTPuser
```

3. Il définit le mot de passe pour `FTPuser`. Le mot de passe est égal à `tdwb8nxt` et il est créé dans la section `password` du fichier `FTPvars`. La commande exécute :

```
E:\IBM\TWA\TWS\CLI\bin>param -c FTPvars.password.FTPuser tdwb8nxt
```

4. Il ouvre le fichier dans un éditeur de texte

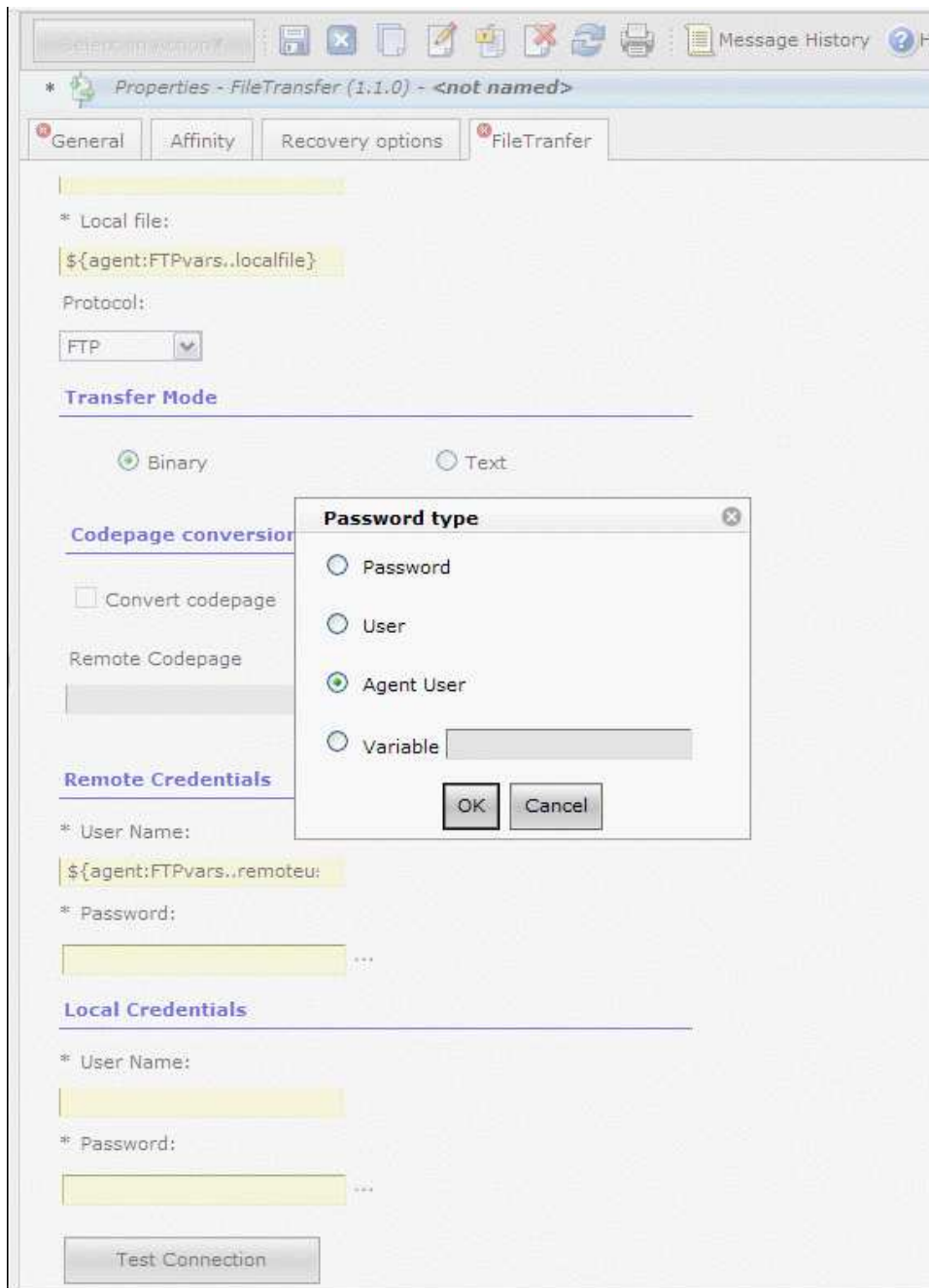
```
E:\IBM\TWA\TWS\ITA\cpa\config\jm_variables_files\FTPvars
```

et vérifie son contenu :

```
localfile = ./npp.5.1.1.Installer.DW.exe  
remoteuser = FTPuser
```

```
[password]  
FTPuser = {aes}XMMYMY2zBHvDEDBo5DdZVmw0Jao60pX1K6x2HhRcovA=
```

5. Il copie le fichier FTPvars dans le chemin d'accès *agent_installation_path*\TWA\TWS\ITA\cpa\config\jm_variables_files> de chaque autre agent défini dans le pool.
6. Il commence la définition d'un nouveau travail de transfert de fichier dans la fenêtre Concepteur de charge de travail de Dynamic Workload Console. Dans la fenêtre FileTransfer :
 - a. Il entre \${agent:FTPvars..localfile} dans la zone Fichier local.
 - b. Il entre \${agent:FTPvars..remoteuser} dans la zone Données d'identification distante →Nom d'utilisateur.
 - c. Cliquez sur le bouton ... en regard de la zone Données d'identification distante →Mot de passe. La fenêtre Type de mot de passe apparaît et l'administrateur sélectionne Utilisateur d'agent.



- d. Après que l'administrateur clique sur le bouton OK pour confirmation dans la fenêtre en incrustation, la zone Données d'identification distante →Mot de passe est renseignée avec la valeur
`${agent:password.${agent:FTPvars..remoteuser}}`.

7. Il définit toutes les autres zones pour terminer la définition du travail.

Lors de l'exécution du travail, les entités et les mots de passe saisis comme variables sont résolus avec les valeurs définies dans le fichier FTPvars.

Définition de variables dans les travaux Dynamic Workload Broker

Cette section explique comment ajouter des variables aux travaux que vous envisagez d'exécuter avec Dynamic Workload Broker.

Vous pouvez inclure des variables dans votre définition de travail. La résolution des variables a lieu au moment de la soumission.

Les variables prises en charge sont les suivantes :

Tableau 93. Variables IBM Workload Scheduler prises en charge dans les définitions JSDL.

Variables pouvant être insérées dans la définition de travail Dynamic Workload Broker	Description
tws.host.workstation	Nom du poste de travail hôte.
tws.job.date	Date du travail soumis.
tws.job.fqname	Nom complet du travail (UNISON_JOB).
tws.job.ia	Heure d'arrivée des données du travail.
tws.job.interactive	Le travail est interactif. Une valeur peut être true (vraie) ou false (fausse). S'applique uniquement aux travaux compatibles avec les versions antérieures.
tws.job.logon	Droits d'accès de l'utilisateur exécutant le travail (LOGIN). S'applique uniquement aux travaux compatibles avec les versions antérieures.
tws.job.name	Nom du travail soumis.
tws.job.num	UNISON_JOBNUM.
tws.job.priority	Priorité du travail soumis.
tws.job.promoted	Le travail est promu. Les valeurs peuvent être YES ou No. Pour plus d'informations sur la promotion des travaux dynamiques, voir «Promotion des travaux planifiés sur des pools dynamiques», à la page 674.
tws.job.recnum	Numéro d'enregistrement du travail.
tws.job.resourcesForPromoted	Quantité des ressources logiques requises affectées sur un pool dynamique à un travail promu. La valeur est soit 1 si le travail est promu, soit 10 si le travail n'est pas promu. Pour plus d'informations sur la promotion de travaux dynamiques, voir «Promotion des travaux planifiés sur des pools dynamiques», à la page 674.
tws.job.taskstring	Chaîne de tâche du travail soumis. S'applique uniquement aux travaux compatibles avec les versions antérieures.
tws.job.workstation	Nom du poste de travail sur lequel le travail est défini.
tws.jobstream.id	ID du flot de travaux incluant le travail (UNISON_SCHED_ID).
tws.jobstream.name	Nom du flot de travaux incluant le travail (UNISON_SCHED).
tws.jobstream.workstation	Nom du poste de travail sur lequel le flot de travaux contenant le travail est défini.
tws.master.workstation	Nom du gestionnaire de domaine maître (UNISON_MASTER).

Tableau 93. Variables IBM Workload Scheduler prises en charge dans les définitions JSDL. (suite)

Variables pouvant être insérées dans la définition de travail Dynamic Workload Broker	Description
tw.s.plan.date	Date de début du plan de production (UNISON_SCHED_DATE).
tw.s.plan.date.epoch	Date de début du plan de production, dans le format époque (UNISON_SCHED_EPOCH).
tw.s.plan.runnumber	Numéro d'exécution du plan de production (UNISON_RUN).

Transmission de variables entre travaux

Dans de nombreux scénarios, la sortie ou une propriété du premier travail d'un flot de travaux peut désigner l'entrée correspondant à l'exécution des travaux successifs dans le même flot de travaux ou dans un flot de travaux différent.

Dans le scénario qui suit, *JobA* et *JobB* appartiennent à la même instance de flot de travaux et *JobA* est un prédécesseur de *JobB*. *JobA* transmet certaines valeurs de variable à *JobB* au moment de l'exécution.

Vous pouvez transmettre les variables suivantes de *JobA* à *JobB* :

- *JobA* exporte certaines propriétés et *JobB* y fait référence en tant que variables dans sa définition dans un format prédéfini. Lors de l'exécution, les variables de *JobB* sont automatiquement résolues. Les propriétés de travail que vous pouvez exporter sont liées au type de travail que vous définissez. Voir «Transmission des propriétés de travail d'un travail à l'autre dans la même instance de flot de travaux», à la page 651.
- *JobA* exporte sa valeur de sortie standard et *JobB* y fait référence en tant que variable. Lors de l'exécution, la variable de *JobB* est automatiquement résolue. Voir «Transmission de la sortie standard de travail d'un travail à l'autre dans la même instance de flot de travaux», à la page 661.
- *Seulement pour les travaux exécutables*. *JobA* exporte sa valeur de sortie standard et *JobB* y fait référence en tant que valeur d'entrée standard. Voir «Transmission de la sortie standard de travail d'un travail à l'autre en tant qu'entrée standard dans la même instance de flot de travaux», à la page 662.
- *Seulement pour les travaux natifs et exécutables*. *JobA* définit la valeur de certaines variable à l'aide de l'utilitaire **jobprop** sur les systèmes d'exploitation UNIX et de l'utilitaire **jobprop.exe** sur les systèmes d'exploitation Windows installés sur les agents dynamiques. *JobB* fait référence à la valeur de ces variable dans sa définition. Lors de l'exécution, les variables de *JobB* sont automatiquement résolues. Voir «Transmission d'un jeu de variables d'un travail à un autre dans la même instance de flot de travaux à l'aide de l'utilitaire **jobprop**», à la page 663.

Dans un autre scénario, *JobA* exporte les variables dans une **table de variables**. La **table de variables** rend les variables exportées disponibles dans *JobB*, où *JobB* est un travail successeur, dans un même flot de travaux ou dans un flot de travaux différent. Voir «Transmission de variables d'un travail à un autre, dans un même flot de travaux ou dans un flot de travaux distinct, via les tables de variables», à la page 665.

Remarque : Le flot de travaux USERJOBS créé par les processus IBM Workload Scheduler, ne prend pas en charge la transmission de variables entre les travaux de ce flot.

Transmission des propriétés de travail d'un travail à l'autre dans la même instance de flot de travaux

Les propriétés de travail que vous pouvez exporter d'un travail dynamique au suivant dans la même instance de flot de travaux sont liées au type de travail que vous définissez. Pour ajouter une propriété de travail dans une autre définition de travail successeur de façon à ce qu'elle soit résolue localement sur l'agent lors de l'exécution, utilisez la syntaxe suivante :

```
${job:<NOM_TRAVAIL>.<nom_propriété>}
```

où <NOM_TRAVAIL> est la valeur de nom ou d'alias du travail à partir duquel vous exportez les valeurs de propriété, et où <nom_propriété> est la propriété en référence. La valeur <nom_propriété> est insensible à la casse.

Seuls certains types de travaux peuvent transmettre les valeurs de propriétés au travail successeur. C'est le cas des types de travaux suivants :

Travaux IBM InfoSphere DataStage

Le tableau 94, à la page 654 présente la liste des propriétés qui sont transmissibles d'un travail IBM InfoSphere DataStage à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables. Pour plus d'informations sur les travaux IBM InfoSphere DataStage, voir *Planification des applications avec IBM Workload Automation*.

Travaux reflet

Le tableau 95, à la page 655 présente la liste des propriétés qui sont transmissibles d'un travail reflet à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux OSLC

Le tableau 96, à la page 655 présente la liste des propriétés qui sont transmissibles d'un travail OSLC à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux IBM WebSphere MQ

Le tableau 97, à la page 655 présente la liste des propriétés qui sont transmissibles d'un travail IBM WebSphere MQ à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux IBM Sterling Connect:Direct

Le tableau 98, à la page 655 présente la liste des propriétés qui sont transmissibles d'un travail IBM Sterling Connect:Direct à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux Salesforce

Le tableau 99, à la page 656 présente la liste des propriétés qui sont transmissibles d'un travail Salesforce à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux SAP BusinessObjects BI

Le tableau 100, à la page 656 présente la liste des propriétés qui sont transmissibles d'un travail SAP BusinessObjects BI à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux Oracle E-Business Suite

Le tableau 101, à la page 656 présente la liste des propriétés qui sont transmissibles d'un travail Oracle E-Business Suite à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de transfert de fichiers

Le tableau 102, à la page 657 présente la liste des propriétés qui sont transmissibles d'un travail de transfert de fichier à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Etant donné que vous pouvez utiliser des caractères génériques pour indiquer une condition de nom partiel, vous pouvez transférer plusieurs fichiers dans le cadre du même travail, et vous pouvez avoir un ensemble complet de propriétés pour chaque fichier transféré.

Travaux Hadoop Map Reduce

Le tableau 103, à la page 657 présente la liste des propriétés qui sont transmissibles d'un travail Hadoop Map Reduce à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux système de fichiers distribué Hadoop

Le tableau 104, à la page 658 présente la liste des propriétés qui sont transmissibles d'un travail système de fichiers distribué Hadoop à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux IBM BigInsights

Le tableau 105, à la page 658 présente la liste des propriétés qui sont transmissibles d'un travail d'application IBM BigInsights à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux JSR 352 Java Batch

Le tableau 106, à la page 658 présente la liste des propriétés qui sont transmissibles d'un travail d'application IBM BigInsights à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux MQTT

Le tableau 107, à la page 659 présente la liste des propriétés transmissibles d'un travail MQTT à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux Apache Oozie

Le tableau 108, à la page 659 présente la liste des propriétés transmissibles d'un travail Apache Oozie à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de Cloudant

Le tableau 109, à la page 660 présente la liste des propriétés qui sont

transmissibles d'un travail Cloudant à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de OpenWhisk

Le tableau 110, à la page 660 présente la liste des propriétés qui sont transmissibles d'un travail OpenWhisk à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de Gestion des travaux

Le tableau 111, à la page 661 présente la liste des propriétés qui sont transmissibles d'un travail Gestion des travaux à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de Soumission d'un flot de travaux

Le tableau 112, à la page 661 présente la liste des propriétés qui sont transmissibles d'un travail Soumission d'un flot de travaux à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Travaux de base de données

Le tableau 113, à la page 661 présente la liste des propriétés qui sont transmissibles d'un travail de base de données à un autre ; il indique également le mappage entre les propriétés Informations supplémentaires du travail et les propriétés utilisables.

Exemple

L'exemple ci-dessous décrit comment la spécification de variables dans différents formats permet aux variables d'avoir plusieurs valeurs, étant donné qu'elles sont résolues à différents moments. Il explique également comment les variables peuvent être transmises d'un travail à un autre dans une instance de flot de travaux. Le flot de travaux WIN92MAS_REW#VP_JS_141800058 contient les travaux JOBA et JOBB. Le travail exécutable JOBB fait référence aux propriétés suivantes du travail reflet JOBA :

- ScheduledTime
- dJobName
- dJobStreamName
- dJobStreamWorkstation

Définitions de la base de données :

```
SCHEDULE WIN92MAS_REW#VP_JS_141800058
:
WIN92MAS_REW#JOBA
TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:dshadow=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow" xmlns:
jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
    <jsdl:application name="distributedShadowJob">
        <dshadow:DistributedShadowJob>
            <dshadow:JobStream>VPJS_141800058</dshadow:JobStream>
            <dshadow:Workstation>ncI25133</dshadow:Workstation>
            <dshadow:Job>VP_JOBMON_141800058</dshadow:Job>
            <dshadow:matching>
                <dshadow:previous/>
            </dshadow:matching>
        </dshadow:DistributedShadowJob>
```

```

    </jsdl:application>
</jsdl:jobDefinition>
  DESCRIPTION "Sample Job Definition for DISTRIBUTED environment"
  RECOVERY STOP
NC125133#JOB
  TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:XMLSchema=
"http://www.w3.org/2001/XMLSchema" xmlns:jsdl="http://www.ibm.com/xmlns/
prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle"
XMLSchema:text="resolveVariableTable" name="executable">
  <jsdl:application name="executable">
    <jsdle:executable>
      <jsdle:script>
echo ScheduledTime:${job:JOBA.ScheduledTime}
echo JobName:${job:JOBA.dJobName}
echo JobStreamName:${job:JOBA.dJobStreamName}
echo JobStreamWorkstation:${job:JOBA.dJobStreamWorkstation}
</jsdle:script>
      </jsdle:executable>
    </jsdl:application>
  </jsdl:jobDefinition>
  DESCRIPTION "Added by composer."
  RECOVERY STOP
  FOLLOWS JOBA

END

```

IBM Workload Scheduler utilise les propriétés des travaux IBM InfoSphere DataStage en l'état. Les valeurs des propriétés Informations supplémentaires des travaux IBM InfoSphere DataStage sont liées à l'environnement local du poste de travail sur lequel IBM InfoSphere DataStage est installé. Le tableau 94 présente les valeurs des propriétés Informations supplémentaires des travaux IBM InfoSphere DataStage sur un poste de travail dont l'environnement local est en_US.

Tableau 94. Propriétés des travaux IBM InfoSphere DataStage

Propriétés des travaux IBM InfoSphere DataStage pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux IBM InfoSphere DataStage
\${job:<NOM_TRAVAIL>.Interim Status}	Interim Status
\${job:<NOM_TRAVAIL>.Invocation ID}	Invocation ID
\${job:<NOM_TRAVAIL>.Invocation List}	Invocation List
\${job:<NOM_TRAVAIL>.Job Control}	Job Control
\${job:<NOM_TRAVAIL>.Job Controller}	Job Controller
\${job:<NOM_TRAVAIL>.Job Process ID}	Job Process ID
\${job:<NOM_TRAVAIL>.Job Restartable}	Job Restartable
\${job:<NOM_TRAVAIL>.Job Start Time}	Job Start Time
\${job:<NOM_TRAVAIL>.Job Status}	Job Status
\${job:<NOM_TRAVAIL>.Job Wave Number}	Job Wave Number
\${job:<NOM_TRAVAIL>.Last Run Time}	Last Run Time
\${job:<NOM_TRAVAIL>.User Status}	User Status

Tableau 95. Propriétés des travaux reflet

Propriétés des travaux reflet pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux reflet
\${job:<NOM_TRAVAIL>.ScheduledTime}	Heure planifiée du travail distant
\${job:<NOM_TRAVAIL>.dJobName}	Travail distant
\${job:<NOM_TRAVAIL>.dJobStreamName}	Flot de travaux distant
\${job:<NOM_TRAVAIL>.dJobStreamWorkstation}	Poste de travail du flot de travaux distant

Tableau 96. Propriétés des travaux OSLC

Propriétés des travaux OSLC pouvant être transférées à un travail	Propriétés Informations supplémentaires des travaux OSLC
\${job:<NOM_TRAVAIL>.RESULT_URI}	RESULT_URI

Tableau 97. Propriétés des travaux IBM WebSphere MQ

Propriétés des travaux IBM WebSphere MQ pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux IBM WebSphere MQ
\${job:<NOM_TRAVAIL>.CHANNEL_PROP}	Canal
\${job:<NOM_TRAVAIL>.SERVER_PROP}	Server
\${job:<NOM_TRAVAIL>.MANAGER_PROP}	Manager
\${job:<NOM_TRAVAIL>.PORT_PROP}	Port
\${job:<NOM_TRAVAIL>.MSG_ID_PROP}	ID message
\${job:<NOM_TRAVAIL>.CORRELATION_ID_PROP}	ID de corrélation
\${job:<NOM_TRAVAIL>.MESSAGE_SENT_PROP}	Message envoyé
\${job:<NOM_TRAVAIL>.MESSAGE_RECEIVED_PROP}	Message reçu
\${job:<NOM_TRAVAIL>.OPERATION_PROP}	Opération

Tableau 98. Propriétés des travaux IBM Sterling Connect:Direct

Propriétés des travaux IBM Sterling Connect:Direct pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux IBM Sterling Connect:Direct
\${job:<NOM_TRAVAIL>.PrimaryNodeAddress}	Adresse du noeud principal
\${job:<NOM_TRAVAIL>.PrimaryNodeUserID}	Utilisateur du noeud principal
\${job:<NOM_TRAVAIL>.SecondaryNodeName}	Nom du noeud secondaire
\${job:<NOM_TRAVAIL>.SecondaryNodeUserID}	Utilisateur du noeud secondaire
\${job:<NOM_TRAVAIL>.ProcessName}	Nom du processus
\${job:<NOM_TRAVAIL>.DestinationDisposition}	Disposition de cible
\${job:<NOM_TRAVAIL>.ProcessFileName}	Nom du fichier de processus
\${job:<NOM_TRAVAIL>.ProcessFileLocation}	Emplacement du fichier de processus
\${job:<NOM_TRAVAIL>.CompressType}	Type de compression

Tableau 98. Propriétés des travaux IBM Sterling Connect:Direct (suite)

Propriétés des travaux IBM Sterling Connect:Direct pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux IBM Sterling Connect:Direct
\${job:<NOM_TRAVAIL>.CheckPointRestart}	Redémarrage du point de vérification
\${job:<NOM_TRAVAIL>.ActionSelected}	Action sélectionnée
\${job:<NOM_TRAVAIL>.SourceFilePath}	Chemin d'accès au fichier source
\${job:<NOM_TRAVAIL>.DestinationFilePath}	Chemin d'accès au fichier cible
\${job:<NOM_TRAVAIL>.ProcessNumber}	Numéro de processus

Tableau 99. Propriétés des travaux Salesforce

Propriétés des travaux Salesforce pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux Salesforce
\${job:<NOM_TRAVAIL>.ApexJobID}	ID de travail APEX
\${job:<NOM_TRAVAIL>.TotalJobItems}	Nombre total d'élément du travail
\${job:<NOM_TRAVAIL>.NumberOfErrors}	Nombre d'erreurs
\${job:<NOM_TRAVAIL>.Status}	Statut du traitement par lots
\${job:<NOM_TRAVAIL>.ExtendedStatus}	Statut étendu

Tableau 100. Propriétés des travaux SAP BusinessObjects BI

Propriétés des travaux SAP BusinessObjects BI pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux SAP BusinessObjects BI
\${job:<NOM_TRAVAIL>.AuthType}	Type d'autorisation
\${job:<NOM_TRAVAIL>.BOInstanceID}	Ressource SAP BusinessObjects ID d'instance
\${job:<NOM_TRAVAIL>.BOInstanceStatus}	Ressource SAP BusinessObjects statut de l'instance
\${job:<NOM_TRAVAIL>.BOObject}	Ressource SAP BusinessObjects
\${job:<NOM_TRAVAIL>.Server}	Adresse de serveur
\${job:<NOM_TRAVAIL>.UserName}	Nom d'utilisateur

Tableau 101. Propriétés des travaux Oracle E-Business Suite

Propriétés des travaux Oracle E-Business Suite pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux Oracle E-Business Suite
\${job:<NOM_TRAVAIL>.DevelopmentPhase}	Phase de développement La phase de la demande en tant que chaîne constante qui peut être utilisée à des fins de comparaison des logiques de programme.
\${job:<NOM_TRAVAIL>.DevelopmentStatus}	Statut du développement Le statut de la demande en tant que chaîne constante qui peut être utilisée à des fins de comparaison des logiques de programme.

Tableau 101. Propriétés des travaux Oracle E-Business Suite (suite)

Propriétés des travaux Oracle E-Business Suite pouvant être transmises dans une autre définition de travail	Propriétés Informations supplémentaires des travaux Oracle E-Business Suite
<code>\${job:<NOM_TRAVAIL>.JobId}</code>	ID du travail L'ID de la demande soumise et gérée par Oracle E-Business Suite.
<code>\${job:<NOM_TRAVAIL>.Message}</code>	Message Le message d'achèvement associé à la demande terminée.
<code>\${job:<NOM_TRAVAIL>.Phase}</code>	Phase La phase de demande conviviale.
<code>\${job:<NOM_TRAVAIL>.Status}</code>	Statut Le statut de demande conviviale.

Tableau 102. Propriétés des travaux de transfert de fichier

Pour le numéro de fichier <i>compteur</i> , les propriétés qui peuvent être transmises à une autre définition de travail (un ensemble de propriétés pour chaque fichier transféré).	Pour le numéro de fichier <i>compteur</i> , les propriétés Extra Information (Informations supplémentaires) du travail (un ensemble de propriétés pour chaque fichier transféré).
<code>\${job:<NOM_TRAVAIL>.File.compteur.LocalFile}</code>	<code>File.compteur.LocalFile</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.LocalUser}</code>	<code>File.compteur.LocalUser</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.Protocol}</code>	<code>File.compteur.Protocol</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.RemoteFile}</code>	<code>File.NOM_TRAVAIL.RemoteFile</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.RemoteUser}</code>	<code>File.compteur.RemoteUser</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.Size}</code>	<code>File.compteur.Size</code>
<code>\${job:<NOM_TRAVAIL>.File.compteur.TotalTransferTime}</code>	<code>File.compteur.TotalTransferTime</code>
<code>\${job:<NOM_TRAVAIL>.NumberOfTransferredFiles}</code>	Nombre de fichiers transférés Une seule valeur pour chaque travail

Tableau 103. Propriétés des travaux Hadoop Map Reduce

Propriétés des travaux Hadoop Map Reduce pouvant être transférées à un travail	Libellé
<code>\${job::<NOM_TRAVAIL>.HadoopDir}</code>	Répertoire d'installation Hadoop
<code>\${job::<NOM_TRAVAIL>.JarFile}</code>	Fichier JAR
<code>\${job::<NOM_TRAVAIL>.MainClassName}</code>	Classe principale
<code>\${job::<NOM_TRAVAIL>.Arguments}</code>	Arguments
<code>\${job::<NOM_TRAVAIL>.Job<INDEX>JobId}</code>	ID travail
<code>\${job::<NOM_TRAVAIL>.Job<INDEX>JobStatus}</code>	Job Status
<code>\${job::<NOM_TRAVAIL>.Application<INDEX>AppId}</code>	Application ID
<code>\${job::<NOM_TRAVAIL>.Application<INDEX>.AppTrackingUrl}</code>	Application Tracking URL

Tableau 104. Propriétés des travaux système de fichiers distribué Hadoop

Propriétés des travaux système de fichiers distribué Hadoop pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.RemoteFile}</code>	Fichier distant
<code>\${job:<NOM_TRAVAIL>.LocalFile}</code>	Fichier local
<code>\${job:<NOM_TRAVAIL>.Permissions}</code>	Permissions
<code>\${job:<NOM_TRAVAIL>.NewRemoteFile}</code>	Nouveau fichier distant
<code>\${job:<NOM_TRAVAIL>.FileDeleted}</code>	Fichier supprimé
<code>\${job:<NOM_TRAVAIL>.AccessTime}</code>	Heure d'accès
<code>\${job:<NOM_TRAVAIL>.BlockSize}</code>	Taille de bloc
<code>\${job:<NOM_TRAVAIL>.Group}</code>	Groupe
<code>\${job:<NOM_TRAVAIL>.Length}</code>	Longueur
<code>\${job:<NOM_TRAVAIL>.ModificationTime}</code>	Heure de modification
<code>\${job:<NOM_TRAVAIL>.Owner}</code>	Propriétaire
<code>\${job:<NOM_TRAVAIL>.PathSuffix}</code>	Suffixe du chemin–
<code>\${job:<NOM_TRAVAIL>.Replication}</code>	Réplication
<code>\${job:<NOM_TRAVAIL>.FileType}</code>	Type de fichier

Remarque : Les propriétés `${job:<NOM_TRAVAIL>.RemoteFile}` et `${job:<NOM_TRAVAIL>.LocalFile}` s'appliquent à toutes les actions dans le travail système de fichiers distribué Hadoop. Les propriétés restantes s'appliquent uniquement à l'action **Attendre un fichier**.

Tableau 105. Propriétés des travaux IBM BigInsights, section d'application

Propriétés des travaux IBM BigInsights pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.Status}</code>	Statut
<code>\${job:<NOM_TRAVAIL>.ElapsedTime}</code>	Temps écoulé
<code>\${job:<NOM_TRAVAIL>.StartTime}</code>	Heure de début
<code>\${job:<NOM_TRAVAIL>.EndTime}</code>	Heure de fin
<code>\${job:<NOM_TRAVAIL>.Progress}</code>	Progression
<code>\${job:<NOM_TRAVAIL>.User}</code>	User
<code>\${job:<NOM_TRAVAIL>.Workflow}</code>	Flot de travaux

Tableau 106. Propriétés des travaux JSR 352 Java Batch, section d'application

Propriétés des travaux JSR 352 Java Batch pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.ExitStatus}</code>	Job Exit Status
<code>\${job:<NOM_TRAVAIL>.JobInstanceId}</code>	Job Instance Id
<code>\${job:<NOM_TRAVAIL>.executionId}</code>	Job Execution Id
<code>\${job:<NOM_TRAVAIL>.JobName}</code>	Job Name

Tableau 106. Propriétés des travaux JSR 352 Java Batch, section d'application (suite)

Propriétés des travaux JSR 352 Java Batch pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.JobBatchStatus}</code>	Job Batch Status
<code>\${job:<NOM_TRAVAIL>.self}</code>	Self
<code>\${job:<NOM_TRAVAIL>.joblogs}</code>	Job Logs
<code>\${job:<NOM_TRAVAIL>.Jobexecution}</code>	Job Execution
Les variables suivantes sont exportées pour chaque étape du travail :	
<code>\${job:<NOM_TRAVAIL>.<NOM_ETAPE>.Name}</code>	Step Name
<code>\${job:<NOM_TRAVAIL>.<NOM_ETAPE>.BatchStatus}</code>	Step Batch Status

Tableau 107. Propriétés des travaux MQTT, section d'application

Propriétés des travaux MQTT pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.Message}</code>	Message

Tableau 108. Propriétés des travaux Apache Oozie, section d'application

Propriétés des travaux Apache Oozie pouvant être transférées à un travail	Libellé
<code>{job:<NOM_TRAVAIL>.id}</code>	id
<code>{job:<NOM_TRAVAIL>.appName}</code>	appName
<code>{job:<NOM_TRAVAIL>.appPath}</code>	appPath
<code>{job:<NOM_TRAVAIL>.status}</code>	statut
<code>{job:<NOM_TRAVAIL>.externalId}</code>	externalId
<code>{job:<NOM_TRAVAIL>.user}</code>	user
<code>{job:<NOM_TRAVAIL>.conf}</code>	conf
<code>{job:<NOM_TRAVAIL>.createdTime}</code>	createdTime
<code>{job:<NOM_TRAVAIL>.startTime}</code>	startTime
<code>{job:<NOM_TRAVAIL>.endTime}</code>	endTime
<code>{job:<NOM_TRAVAIL>.run}</code>	run
<code>\${job:<NOM_TRAVAIL>.action.<INDEX>.name}</code>	action.<INDEX>.name
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.id}</code>	action.<NOM_ACTION>.id
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.type}</code>	action.<NOM_ACTION>.type
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.status}</code>	action.<NOM_ACTION>.status
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.transition}</code>	action.<NOM_ACTION>.transition
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.startTime}</code>	action.<NOM_ACTION>.startTime

Tableau 108. Propriétés des travaux Apache Oozie, section d'application (suite)

Propriétés des travaux Apache Oozie pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.endTime}</code>	<code>action.<NOM_ACTION>.endTime</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.externalId}</code>	<code>action.<NOM_ACTION>.externalId</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.externalStatus}</code>	<code>action.<NOM_ACTION>.externalStatus</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.conf}</code>	<code>action.<NOM_ACTION>.conf</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.retries}</code>	<code>action.<NOM_ACTION>.retries</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.consoleUrl}</code>	<code>action.<NOM_ACTION>.consoleUrl</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.trackerUri}</code>	<code>action.<NOM_ACTION>.trackerUri</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.errorCode}</code>	<code>action.<NOM_ACTION>.errorCode</code>
<code>\${job:<NOM_TRAVAIL>.action.<NOM_ACTION>.errorMessage}</code>	<code>action.<NOM_ACTION>.errorMessage</code>

Tableau 109. Propriétés des travaux Cloudant, section d'application

Propriétés des travaux Cloudant pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.size}</code>	Taille de la base de données
<code>\${job:<NOM_TRAVAIL>.document}</code>	ID document
<code>\${job:<NOM_TRAVAIL>.rev}</code>	Révision du document
<code>\${job:<NOM_TRAVAIL>.source}</code>	Base de données source
<code>\${job:<NOM_TRAVAIL>.target}</code>	Base de données cible
<code>\${job:<NOM_TRAVAIL>.replication_id}</code>	ID réplication
<code>\${job:<NOM_TRAVAIL>.attachName}</code>	Nom de la pièce jointe

Tableau 110. Propriétés des travaux OpenWhisk, section d'application

Propriétés des travaux OpenWhisk pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.actionName}</code>	Nom de l'action
<code>\${job:<NOM_TRAVAIL>.triggerName}</code>	Nom du déclencheur
<code>\${job:<NOM_TRAVAIL>.namespace}</code>	Espace-noms
<code>\${job:<NOM_TRAVAIL>.operationType}</code>	Type d'opération
<code>\${job:<NOM_TRAVAIL>.parameters}</code>	Paramètres
<code>\${job:<NOM_TRAVAIL>.success}</code>	Réussite
<code>\${job:<NOM_TRAVAIL>.status}</code>	Statut

Tableau 110. Propriétés des travaux OpenWhisk, section d'application (suite)

Propriétés des travaux OpenWhisk pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.activationId}</code>	ID activation

Tableau 111. Propriétés des travaux Gestion des travaux, section d'application

Propriétés des travaux Gestion des travaux pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.jobName}</code>	jobname
<code>\${job:<NOM_TRAVAIL>.jobstreamId}</code>	jobstreamid
<code>\${job:<NOM_TRAVAIL>.workstation}</code>	workstation
<code>\${job:<NOM_TRAVAIL>.user}</code>	userName
<code>\${job:<NOM_TRAVAIL>.url}</code>	URL

Tableau 112. Propriétés des travaux Soumission d'un flot de travaux, section d'application

Propriétés des travaux Soumission d'un flot de travaux pouvant être transférées à un travail	Libellé
<code>\${job:<NOM_TRAVAIL>.Url}</code>	URL
<code>\${job:<NOM_TRAVAIL>.workstation}</code>	Poste de travail
<code>\${job:<NOM_TRAVAIL>.jobstreamname}</code>	Nom du flot de travaux
<code>\${job:<NOM_TRAVAIL>.datetime}</code>	Heure/date de démarrage minimale
<code>\${job:<NOM_TRAVAIL>.variabletablename}</code>	Nom de la table de variables
<code>\${job:<NOM_TRAVAIL>.variablelistValues}</code>	Valeurs de table de variables {"key1":value1,"key2":value2,...}

Tableau 113. Propriétés des travaux de base de données, section d'application

Propriétés des travaux de base de données pouvant être transférées à un autre travail	Libellé
<code>\${job:<NOM_TRAVAIL>.NumberOfRows}</code>	Nombre de lignes

Transmission de la sortie standard de travail d'un travail à l'autre dans la même instance de flot de travaux

Vous pouvez exporter la sortie standard de travail d'un travail dynamique à son successeur dans la même instance de flot de travaux. La variable de sortie standard du travail est utilisée dans la zone script de la définition du travail

Pour ajouter une sortie standard du travail dans une autre définition de travail, de sorte qu'elle soit résolue localement sur l'agent lors de l'exécution, utilisez la syntaxe suivante :

```
${job:<NOM_TRAVAIL>.stdlist}
```

où <NOM_TRAVAIL> est le nom ou l'alias du travail à partir duquel vous exportez la sortie standard du travail.

Exemple

Dans cet exemple, le flot de travaux WIN92MAS_REW#VP_JS_141800058 contient JOBA_ALIAS, qui est l'alias de JOBA, et les travaux JOBB. Le travail exécutable JOBB fait référence à la sortie standard JOBA_ALIAS.

Définitions de la base de données :

SCHEDULE WIN92MAS_REW#VP_JS_141800058

:

WIN92MAS_REW#JOBA as JOBA_ALIAS

TASK

```
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:dshadow=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/dshadow" xmlns:
jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL">
  <jSDL:application name="distributedShadowJob">
    <dshadow:DistributedShadowJob>
      <dshadow:JobStream>VPJS_141800058</dshadow:JobStream>
      <dshadow:Workstation>nc125133</dshadow:Workstation>
      <dshadow:Job>VP_JOBMON_141800058</dshadow:Job>
      <dshadow:matching>
        <dshadow:previous/>
      </dshadow:matching>
    </dshadow:DistributedShadowJob>
  </jSDL:application>
</jSDL:jobDefinition>
DESCRIPTION "Sample Job Definition for DISTRIBUTED environment"
RECOVERY STOP
```

NC125133#JOBB

TASK

```
<?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:XMLSchema="http://www.w3.org/2001/XMLSchema" xmlns:
jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL" xmlns:
jSDLE="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDLE"
XMLSchema:text="resolveVariableTable" name="executable">
  <jSDL:application name="executable">
    <jSDLE:executable>
      <jSDLE:script>echo &quot;stdlist: ${job:JOBA_ALIAS.stdlist}&quot;;
    </jSDLE:script>
  </jSDLE:executable>
</jSDL:application>
</jSDL:jobDefinition>
DESCRIPTION "Added by composer."
RECOVERY STOP
FOLLOWS JOBA_ALIAS
FIN
```

Transmission de la sortie standard de travail d'un travail à l'autre en tant qu'entrée standard dans la même instance de flot de travaux

Vous pouvez exporter la sortie standard de travail d'un travail dynamique à un travail exécutable successeur en tant qu'entrée standard dans la même instance de flot de travaux. La variable de sortie standard du travail est utilisée dans la zone input de la définition du travail exécutable

Pour ajouter une sortie standard du travail dans une autre définition de travail exécutable, de sorte qu'elle soit résolue localement sur l'agent lors de l'exécution, utilisez la syntaxe suivante :

```
${job:<NOM_TRAVAIL>.stduri}
```

où <NOM_TRAVAIL> est la valeur de nom ou d'alias du travail à partir duquel vous exportez la sortie standard du travail.

Remarque : Le transfert de la variable *stduri* n'est pas pris en charge pour les travaux reflet. Puisque les travaux reflets ne produisent pas de journal de travail, si vous transmettez la variable *stdout* du travail en tant qu'entrée à un autre travail du même flot de travaux, la sortie du travail est vide. Un travail reflet peut uniquement imprimer un message d'état.

Exemple

Dans cet exemple, le flot de travaux NC112019#JS_PROP contient JOBALIAS_A, qui est l'alias de NC112019#JOBA, et les travaux NC112019#JOB B. Le travail exécutable NC112019#JOB B fait référence à la sortie standard JOBALIAS_A comme l'entrée standard.

Définitions de la base de données :

```
SCHEDULE NC112019#JS_PROP
:
NC112019#JOBA AS JOBALIAS_A
TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/
1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" name="executable">
    <jsdl:application name="executable">
        <jsdle:executable interactive="false" path="ls"/>
    </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Ajouté par Composer pour le flot de travaux : NC112019#JS_PROP."
RECOVERY STOP

NC112019#JOB B
TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:XMLSchema="http://www.w3.org/2001/XMLSchema"
xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" XMLSchema:text="resolveVariableTable"
name="executable">
    <jsdl:application name="executable">
        <jsdle:executable input="${job:JOBALIAS_A.stduri}" interactive="false" path="cat"/>
    </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Ajouté par Composer pour le flot de travaux : WIN92MAS#JS_PROP."
RECOVERY STOP
FOLLOWS JOBALIAS_A
FIN
```

Transmission d'un jeu de variables d'un travail à un autre dans la même instance de flot de travaux à l'aide de l'utilitaire jobprop

Vous pouvez utiliser l'utilitaire **jobprop** installé sur agents dynamiques pour définir la variable et sa valeur dans un travail et transmettre la variable au travail successif dans la même instance de flot de travaux.

Pour définir la variable et sa valeur dans le premier travail, utilisez la syntaxe suivante :

```
jobprop <NOM_VAR> <valeur>
```

où <NOM_VAR> désigne la variable que vous pouvez exporter dans un autre travail et <valeur> désigne la valeur affectée à <NOM_VAR>. Pour plus d'informations sur l'utilitaire **jobprop**, voir «jobprop», à la page 735.

Pour définir les variables dans un autre travail, utilisez la syntaxe suivante :

```
${job:<NOM_TRAVAIL>.<NOM-VAR>}
```

où <NOM_TRAVAIL> est la valeur de nom ou d'alias du travail à partir duquel vous exportez la valeur de variable <NOM-VAR>.

Exemple

Dans cet exemple, le flot de travaux WIN92MAS#JS_PROP contient les travaux exécutables NC125133#JOBA et NC125133#JOBB. Le travail NC125133#JOBB fait référence aux valeurs de variables NC125133#JOBA suivantes définies à l'aide de l'utilitaire **jobprop** :

- Variable VAR1 définie sur la valeur value1.
- Variable VAR2 définie sur la valeur value2.
- Variable VAR3 définie sur la valeur value3.
- Variable VAR4 définie sur la valeur value4.

Définitions de la base de données :

```
SCHEDULE WIN92MAS#JS_PROP
```

```
:
```

```
NC125133#JOBA
```

```
TASK
```

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/
prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>#!/bin/sh
      . /home/ITAuser/TWA/TWS/tws_env.sh
      jobprop VAR1 value1
      jobprop VAR2 value2
      jobprop VAR3 value3
      jobprop VAR4 value4
    </jsdle:script>
  </jsdle:executable>
</jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Sample Job Definition"
RCCONDSUCC "RC">=0"
RECOVERY STOP
```

```
NC125133#JOBB
```

```
TASK
```

```
<?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl=
"http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/
scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>
```

```

echo VAR1={{job:joba.VAR1}}
echo VAR2={{job:joba.VAR2}}
echo VAR3={{job:joba.VAR3}}
echo VAR4={{job:joba.VAR4}}
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Sample Job Definition"
RCCONDSUCC "RC>=0"
RECOVERY STOP
FOLLOWS JOBA
FIN

```

Transmission de variables d'un travail à un autre, dans un même flot de travaux ou dans un flot de travaux distinct, via les tables de variables

Vous pouvez utiliser des **tables de variables** pour définir des variables exportées à partir d'un travail et les transmettre à n'importe quel travail successeur, dans un même flot de travaux ou dans un flot de travaux différent.

Pour exporter des variables à partir d'un travail dans une table de variables, un type de travail IBM Workload Scheduler adapté est disponible : VariableTable. Le travail VariableTable doit être ajouté au flot de travaux en tant que successeur du travail qui exporte les variables. Le travail VariableTable définit les variables exportées dans une **table de variables** et les met à disposition de n'importe quel autre travail successeur, dans un même flot de travaux ou dans un flot de travaux différent.

Vous pouvez aisément définir un travail VariableTable à l'aide deDynamic Workload Console ou de la ligne de commande **composer**. Pour plus d'informations sur la définition d'un travail VariableTable, voir la section «Travaux de Table de variables», à la page 633

Exécution d'un script à l'issue d'un travail

Dans la plupart des scénarii, lorsqu'un travail est terminé, vous pouvez exécuter une ou plusieurs actions en utilisant les informations relatives à l'exécution du travail. A cet effet, vous pouvez écrire un fichier script et l'enregistrer dans un répertoire du système de fichier de l'agent. Ce script est exécuté à chaque fois qu'un travail est terminé, avec succès ou pas. Le script s'exécute avec les mêmes données d'identification que l'utilisateur d'agent qui exécute le travail.

Remarque : L'utilisateur d'agent doit être autorisé à accéder au fichier script et à son répertoire.

Pour indiquer à IBM Workload Scheduler le chemin d'accès au fichier script, vous devez modifier le fichier JobManager.ini comme suit :

1. Recherchez le fichier JobManager.ini sur l'instance d'agent local où s'exécutera le script. Ce fichier se trouve dans le répertoire TWA_home/TWS/ITA/cpa/config sur l'agent.
2. Dans la section [NativeJobLauncher] du fichier, définissez la valeur de la propriété **PostJobExecScriptPathName**, avec le chemin qualifié complet du fichier script que vous souhaitez exécuter à la fin d'un travail.
3. Sauvegardez les modifications dans le fichier.

Si vous ne spécifiez aucun chemin de fichier ou que le fichier script n'existe pas, aucune action n'est entreprise à l'issue du travail. Pour plus de détails sur la personnalisation de la propriété **PostJobExecScriptPathName**, voir le *Guide d'administration*.

Les variables de travail suivantes peuvent être utilisées dans le script :

- JOB_ID
- JOB_ALIAS
- JOB_SPOOL_DIR
- JOB_STATUS
- JOB_RETURN_CODE
- JOB_DURATION
- JOB_START_TIME
- JOB_END_TIME

Le script est exécuté pour l'un des états finaux de travail suivants :

- SUCCEEDED_ EXECUTION
- UNKNOWN
- CANCELLED
- FAILED_ EXECUTION

Dans le fichier `JobManager_message.log`, un message vous indique si le travail s'est arrêté correctement ou si une erreur a empêché le travail de démarrer. Pour analyser le résultat de l'exécution du script, vous pouvez consulter le fichier `out.log` dans le sous-répertoire `post_script` du travail `SpoolDir`.

Chapitre 15. Gestion des fonctions de planification dynamique dans votre environnement

Cette section explique comment vous pouvez gérer des fonctions de planification dynamique dans votre environnement afin de planifier des travaux IBM Workload Scheduler existants et des types de travaux avec options avancées ; ceux-ci sont fournis avec le produit et les types supplémentaires sont implémentés à l'aide de plug-in personnalisés.

Les fonctions dynamiques vous aident à conserver des stratégies métier et à vérifier les contrats de service en :

- reconnaissant automatiquement des ressources d'environnement de planification ;
- faisant correspondre des exigences de travaux avec des ressources disponibles ;
- contrôlant et optimisant l'utilisation de ressources ;
- suivant automatiquement des modifications de ressources ;
- demandant des ressources supplémentaires, le cas échéant.

Vous pouvez activer des fonctions dynamiques dans votre environnement en définissant un ensemble de types de poste de travail :

Agent dynamique

Poste de travail qui gère une grande variété de types de travaux, comme par exemple les travaux d'une base de données ou d'un protocole FTP spécifiques, en plus des types de travaux existants. Ce poste de travail est automatiquement défini et enregistré dans la base de données IBM Workload Scheduler lorsque vous installez le agent dynamique. Vous pouvez regrouper les agents dynamiques dans des pools et des pools dynamiques.

Dans un réseau simple, les agents dynamiques se connectent directement au gestionnaire de domaine maître ou via un gestionnaire de domaine dynamique. Dans des topologies de réseau plus complexes, où le gestionnaire de domaine maître ou le gestionnaire de domaine dynamique ne peut pas communiquer directement avec l'agent dynamique, vous pouvez configurer vos agents dynamiques pour utiliser une passerelle locale ou distante.

Pool Un poste de travail regroupant un ensemble d'agents dynamiques doté de caractéristiques matérielles ou logicielles similaires auxquelles sont soumis les travaux. IBM Workload Scheduler équilibre les travaux entre les agents dynamiques du pool et réaffecte automatiquement les travaux aux agents dynamiques disponibles lorsqu'un agent dynamique n'est plus disponible. Pour créer un pool d'agents dynamiques dans votre environnement IBM Workload Scheduler, définissez un poste de travail de type **pool** hébergé par le poste de travail Workload Broker, puis sélectionnez les agents dynamiques que vous souhaitez ajouter au pool. Vous pouvez définir le pool à l'aide de Dynamic Workload Console ou de la commande **composer**.

Pool dynamique

Poste de travail qui regroupe un ensemble de agents dynamiques, défini de manière dynamique en fonction des exigences de ressource que vous indiquez et hébergé par le poste de travail Workload Broker. Par exemple, si vous avez besoin d'un poste de travail sollicitant peu l'unité centrale et

que Windows est installé pour exécuter votre travail, vous spécifiez les exigences suivantes à l'aide de Dynamic Workload Console ou de la commande **composer**. Lorsque vous sauvegardez l'ensemble d'exigences, un nouveau poste de travail est automatiquement créé dans la base de données IBM Workload Scheduler. Ce poste de travail est hébergé par le poste de travail Workload Broker. Ce poste de travail mappe tous les agents dynamiques de votre environnement qui répondent aux exigences spécifiées. Le pool généré est mis à jour dynamiquement dès qu'un nouvel agent dynamique approprié devient disponible. Les travaux s'exécutent sur le premier poste de travail du pool dynamique qui répond à toutes les exigences. Les travaux planifiés sur ce poste de travail héritent automatiquement des exigences définies pour le poste de travail.

Pour plus d'informations sur la création de pools et de pools dynamiques à l'aide de l'IBM Dynamic Workload Console, voir

la section consacrée à la création d'un pool d'agents dans *IBM Dynamic Workload Console - Guide d'utilisation*. Pour plus d'informations sur la création de pools et de pools dynamiques à l'aide de la commande **composer**, voir *Guide d'utilisation et de référence, SC11-2009*.

Les pools agents dynamiques et les pools dynamiques optimisent la fonction dynamique intégrée à IBM Workload Scheduler et donnent la possibilité d'associer de façon dynamique votre charge de travail soumise (ou une partie) aux meilleures ressources disponibles lors de l'exécution. Vous pouvez activer des fonctions de planification dynamique sur les postes de travail lors de l'installation. Pour plus d'informations sur l'installation de agents dynamiques, voir la section relative à l'installation d'un nouvel agent dans le *Guide de planification et d'installation, SC11-2008*.

Vous pouvez agents dynamiques, les pools et les pools dynamiques pour planifier les types de travaux avec options avancées. Les types de travaux avec options avancées incluent ceux fournis avec le produit et les types supplémentaires implémentés via les plug-in personnalisés. Les deux types de travail s'exécutent uniquement sur agents dynamiques, les pools et les pools dynamiques. Pour plus d'informations sur la planification des types de travaux avec options avancées, voir «Création de définitions de travaux avancés», à la page 581. Pour plus d'informations sur la création de plug-in personnalisés, reportez-vous à *Extension d'IBM Workload Automation*.

Vous pouvez également utiliser agents dynamiques, des pool et des pools et des pools dynamiques, pour exécuter les travaux que vous avez créés pour les types de poste de travail IBM Workload Scheduler existants. Pour exécuter ces travaux sur les types de poste de travail dynamique, il suffit de modifier la spécification du poste de travail sur lequel vous souhaitez procéder à l'exécution. Pour plus d'informations sur la planification de travaux IBM Workload Scheduler existants, voir «Ajout de fonctions dynamiques aux travaux IBM Workload Scheduler existants», à la page 674.

Si vous souhaitez optimiser la fonction dynamique lors de la planification des types de travaux avec options avancées, vous les planifiez sur des pools et des pools dynamiques, ce qui affecte dynamiquement le travail aux meilleures ressources disponibles. Si n'avez besoin de définir que des types de travaux avec options avancées, sans utiliser la fonction de planification dynamique, vous planifiez ces travaux sur un agent dynamique spécifique, sur lequel le travail s'exécute de façon statique.

Scénario métier sur fonction dynamique

Cette section illustre un exemple de scénario métier qui expose les avantages de types de travaux avec options avancées et des fonctions dynamiques.

Une compagnie d'assurances exécute un certain nombre de travaux dans la nuit afin d'enregistrer les données traitées au cours de la journée dans la base de données de sauvegarde. Elle doit également recueillir toutes les données concernant les transactions effectuées pendant la journée sur tous les postes de travail des filiales. Elle utilise des bases de données DB2. A l'aide de types de travaux avec options avancées fournis dans le concepteur de charge de travail, elle crée un travail pour effectuer une sauvegarde de base de données et un autre travail pour extraire les données des transactions quotidiennes. Pour effectuer ces opérations, elle utilise la nouvelle base de données type de travail avec options avancées.

Après le regroupement des données de tous les postes de travail de l'entreprise, elle copie les données de résultat sur un poste de travail unique et les traite afin de générer un rapport. Elle choisit de manière dynamique le meilleur poste de travail disponible en définissant les exigences nécessaires pour exécuter le travail : un poste de travail disposant d'un espace disque important, une UC puissante et le programme requis pour générer le rapport.

Si l'administrateur ne souhaite pas modifier le flot de travaux qu'il a utilisé avant IBM Workload Scheduler version 8.6 pour exécuter un travail Java, par exemple, il peut modifier le nom du poste de travail sur lequel le travail doit s'exécuter en insérant le nom d'un pool ou d'un pool dynamique de agents dynamiques où l'exécutable Java est installé. IBM Workload Scheduler traduit la syntaxe du travail de sorte qu'il peut être exécuté par le programme Java et affecte le travail aux meilleures ressources disponibles dans le pool.

Le rapport met en évidence le nombre de nouveaux contrat signés et le nombre de clients en retard de paiement. Le message qui est envoyé au chef comptable répertorie le nombre de nouveaux contrats et le nombre de clients en retard.

Une entreprise peut atteindre ses objectifs de la manière suivante :

- Utiliser les nouveaux postes de travail avec des fonctions dynamiques pour exécuter les travaux que l'administrateur a créés pour les postes de travail IBM Workload Scheduler existants. Pour exécuter ces travaux sur les nouveaux postes de travail, l'administrateur change uniquement le poste de travail sur lequel il souhaite exécuter le travail. Le principal avantage réside dans le fait qu'il peut utiliser les flux de travaux qu'il a précédemment créés sans action supplémentaire.
- Définir plusieurs types de travaux avec options avancées sans avoir de connaissances particulières sur les applications exécutant les travaux.

Ces types de travaux avec options avancées s'exécutent sur les postes de travail suivants :

agents dynamiques

Poste de travail pouvant exécuter des travaux existants et des types de travaux avec options avancées.

Pools Groupes pour lesquels vous pouvez ajouter des agents dynamiques en fonction de vos besoins. Les travaux sont attribués de manière dynamique au meilleur agent disponible.

Pools dynamiques

Groupes de agents dynamiques pour lesquels vous indiquez vos exigences et laissez IBM Workload Scheduler sélectionner les agents dynamiques correspondant à vos besoins. Les travaux sont affectés de manière dynamique au meilleur agent dynamique disponible.

Scénario : création d'une définition de travail et soumission à un pool dynamique

Dans ce scénario, vous devez définir les exigences pour l'exécution du travail lors de la création du pool dynamique, par exemple, vous pouvez inclure dans le pool dynamique tous les postes de travail doté du système d'exploitation Windows, de DB2 et d'une UC utilisée à 50 % au maximum. Le pool dynamique est ensuite rempli avec les postes de travail qui correspondent à vos exigences et il est alors prêt pour le travail qui sera soumis.

Pour créer un pool dynamique et lui soumettre un travail, procédez comme suit :

1. Connectez-vous à Dynamic Workload Console.
2. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de l'environnement de charge de travail > Création de postes de travail**.
3. Sélectionnez un moteur et cliquez sur **OK**.
4. La page **Propriétés du poste de travail** s'affiche.
5. Sélectionnez **Pool dynamique** dans le menu **Type de poste de travail**.
6. Complétez les zones requises.
7. Cliquez sur **Editer les exigences**. La page **Exigences** s'affiche.
8. Spécifiez les exigences suivants :
 - Sélectionnez le système d'exploitation dans la page **Système d'exploitation**.
 - Sélectionnez l'utilisation de l'UC dans la page **Utilisation de l'UC**.
 - Sélectionnez la ressource logique requise dans la page **Ressources logiques**. Pour inclure les postes de travail dotés de DB2, cliquez sur **Ajouter** et spécifiez votre exigence dans la page **Exigences**.
 - Le cas échéant, sélectionnez la politique d'optimisation requise.
9. Cliquez sur **OK** pour sauvegarder vos exigences.
10. Cliquez sur **Sauvegarder** pour enregistrer le pool dynamique.
11. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de la charge de travail > Gestion des définitions de charge de travail**.
12. Spécifiez un nom de moteur, distribué ou z/OS. Le concepteur de charge de travail s'ouvre. Les types et les caractéristiques des travaux varient selon que vous sélectionnez un moteur distribué ou z/OS.
13. Sélectionnez **Nouveau > Définition du travail > Base de données et intégrations > Base de données**.
14. Remplissez les zones obligatoires et spécifiez le pool dynamique que vous avez créé précédemment dans la zone **Poste de travail**.
15. Entrez vos instructions SQL dans l'onglet **SQL**.
16. Cliquez sur **Sauvegarder** pour sauvegarder le travail.
17. A partir de la barre d'outils de navigation, cliquez sur **Administration > Soumission de la charge de travail > Soumission de travaux prédéfinis**.

Scénario : création d'une définition de travail et soumission à un pool

Dans ce scénario, vous avez besoin d'exécuter un script de mise à jour d'inventaire. Par conséquent, vous créez un pool pour regrouper les postes de travail dans lequel le programme requis est installé et vous créez un travail pour exécuter le script. Vous sélectionnez le système cible pour le travail du pool invadmin.

Pour créer un pool et lui soumettre un travail, procédez comme suit :

1. Connectez-vous à Dynamic Workload Console.
2. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de l'environnement de charge de travail > Création de postes de travail**.
3. Sélectionnez un moteur et cliquez sur **OK**.
4. La page **Propriétés du poste de travail** s'affiche.
5. Sélectionnez **Pool** dans le menu **Type de poste de travail**.
6. Nommez le pool invadmin et complétez les zones obligatoires.
7. Sélectionnez les postes de travail que vous souhaitez ajouter au pool. Sélectionnez les postes de travail sur lesquels le programme requis est installé.
8. Cliquez sur **Sauvegarder** pour enregistrer le pool.
9. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de la charge de travail > Gestion des définitions de charge de travail**.
10. Spécifiez un nom de moteur, distribué ou z/OS. Le concepteur de charge de travail s'ouvre. Les types et les caractéristiques des travaux varient selon que vous sélectionnez un moteur distribué ou z/OS.
11. Sélectionnez **Nouveau > Définition du travail > Natif > Exécutable**.
12. Spécifiez un nom pour le travail.
13. Dans la zone **Poste de travail**, spécifiez le pool invadmin que vous avez défini précédemment.
14. Accédez à l'onglet **Tâche** et sélectionnez **Commande**.
15. Entrez le nom et le chemin d'accès du fichier exécutable que vous souhaitez exécuter, situé sur chaque poste de travail du pool.
16. Cliquez sur **Sauvegarder** pour sauvegarder le travail.
17. A partir de la barre d'outils de navigation, cliquez sur **Administration > Soumission de la charge de travail > Soumission de travaux prédéfinis**.

Définition de relations d'affinité

Les relations d'affinité déclenchent l'exécution de travaux sur le même poste de travail. Le poste de travail sur lequel le premier travail s'exécute est choisi de façon dynamique par Dynamic Workload Broker, le travail ou les travaux d'affinité s'exécutant sur le même poste de travail. Cette section s'applique aux types de travaux avec options avancées et aux travaux Workload Broker.

Dans Dynamic Workload Broker, vous pouvez définir des relations d'affinité entre deux travaux ou plus lorsque vous souhaitez les exécuter sur le même poste de travail. Lors de la soumission du travail à partir de l'environnement IBM Workload Scheduler, vous pouvez définir l'affinité qui sera résolue par Dynamic Workload Broker en ajoutant une définition d'affinité à la section **Chaîne de la tâche** du travail IBM Workload Scheduler à l'aide du nom de travail IBM Workload Scheduler comme suit :

jobName [-var *varName*=*varValue*,...,-twsaffinity *jobname*=*twsJobName*

où

twsJobName

Correspond au nom de l'instance du travail IBM Workload Scheduler avec lequel vous souhaitez établir une relation d'affinité.

Remarque : Les travaux doivent appartenir au même flot de travaux

Définition des dépendances de fichier dans la planification dynamique

Vous pouvez gérer les dépendances de fichier avec des agents dynamiques, des pools et des pools dynamiques

Introduction aux dépendances de fichier

Vous pouvez utiliser les dépendances de fichier dans la planification dynamique pour contrôler le traitement des travaux et des flots de travaux en fonction de l'existence d'un ou de plusieurs fichiers ou répertoires. Lorsque vous spécifiez une dépendance de fichier, les processus d'IBM Workload Scheduler vérifient si le fichier ou le répertoire indiqué existe avant de lancer le traitement des travaux ou des flots de travaux.

Vous pouvez sélectionner une ou plusieurs des conditions suivantes, qui sont associées au fichier et qui doivent être satisfaites avant que les travaux ou le traitement de flot de travaux ne commencent :

- Le fichier existe.
- Le fichier existe et il s'agit d'un répertoire.
- Le fichier existe et il s'agit d'un fichier standard.
- Le fichier existe et il est accessible en lecture.
- Le fichier existe et sa taille n'est pas nulle.
- Le fichier existe et il est accessible en écriture.

Comportement des dépendances de fichier

Les dépendances de fichier ont des comportements différents avec les agents dynamiques, les pools et les pools dynamiques

Agents dynamiques

IBM Workload Scheduler gère la résolution de dépendance de fichier pour les agents dynamiques de la même manière que pour le poste de travail qui est défini dans la planification classique en tant qu'agent tolérant aux pannes, gestionnaire de domaine maître et sa sauvegarde, gestionnaire de domaine et sa sauvegarde, et ainsi de suite.

Si vous définissez le travail JOB_A sur l'agent dynamique DYN_A, qui dépend du fichier FILE_A, avant que le travail JOB_A ne s'exécute, les processus IBM Workload Scheduler effectuent une vérification d'existence de fichier sur le poste de travail DYN_A. Lorsque le fichier FILE_A est trouvé, la dépendance est résolue et le travail JOB_A s'exécute sur l'agent dynamique DYN_A.

Pools et pools dynamiques

Un pool contient plusieurs postes de travail d'agent dynamique ayant des caractéristiques matérielles et logicielles similaires. Un pool dynamique contient plusieurs postes de travail d'agent dynamique qui sont définis de façon dynamique en fonction des exigences de ressources que vous indiquez. Si vous définissez une dépendance de fichier pour des travaux ou des flots de travaux définis dans un pool ou un pool dynamique, les processus IBM Workload Scheduler effectuent une vérification d'existence de fichier sur chaque poste de travail d'agent dynamique du pool ou du pool dynamique. La dépendance de fichier est résolue lorsque le premier fichier est trouvé sur un poste de travail d'agent dynamique. Le travail ne s'exécute pas nécessairement sur le poste de travail agent dynamique où se trouve le fichier, mais il s'exécute sur l'un des postes de travail actifs dans le pool ou le pool dynamique lorsque la dépendance de fichier est résolue.

Dans votre environnement, vous avez P00L_A qui contient les agents dynamiques DYN_A1, DYN_A2 et DYN_A3. Si vous définissez le travail JOB_A dans le pool P00L_A, qui dépend du fichier FILE_A, avant que le travail JOB_A ne s'exécute, les processus IBM Workload Scheduler effectuent une vérification d'existence de fichier sur les postes de travail DYN_A1, DYN_A2 et DYN_A3. Si le fichier FILE_A se trouve sur le poste de travail DYN_A2, la dépendance de fichier est résolue. Le travail JOB_A s'exécute automatiquement sur DYN_A3 qui est le poste de travail d'agent dynamique actif dans le pool P00L_A.

Comment définir des dépendances de fichier

Vous pouvez définir des dépendances de fichier pour des travaux et des flots de travaux sur des agents dynamiques, des pools ou des postes de travail de pool dynamique, en définissant le mot clé opens dans la définition de planification du travail ou du flot de travaux. Pour plus d'informations sur la syntaxe du mot clé opens, voir «opens», à la page 274.

L'exemple suivant montre que les processus IBM Workload Scheduler vérifie que le fichier /opt/SF1-DIR/myfileSF1.txt existe et qu'il est accessible en lecture sur le poste de travail d'agent dynamique DYN-AGT-SF1 avant que le travail SF1-JOB-HOSTNAME-0001 ne s'exécute sur le poste de travail DYN-AGT-SF1 :

```
SCHEDULE MDMWKSNY1#NY1-JS1
VARIABLE VARTABLENY
OPENS DYN-AGT-SF1#"/opt/SF1-DIR/myfileSF1.txt" (-r %p)
:
DYN-AGT-SF1#SF1-JOB-HOSTNAME-0001
TASK
    <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/
scheduling/1.0/jsdl"
xmlns:jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle" na
me="executable">
    <jsdl:application name="executable">
        <jsdle:executable interactive="false">
            <jsdle:script suffix="">>hostname >>/opt/SF1-DIR/myfileSF1.txt
        </jsdle:script>
    </jsdle:executable>
</jsdl:application>
</jsdl:jobDefinition>
RECOVERY STOP
FIN
```

Voir aussi

Vous pouvez également définir les dépendances de fichier pour des travaux et des flots de travaux à l'aide de Dynamic Workload Console.

Pour plus d'informations sur la façon de définir des dépendances de fichier à l'aide de Dynamic Workload Console, voir :

Dynamic Workload Console - Guide d'utilisation, section "Conception de votre charge de travail".

Promotion des travaux planifiés sur des pools dynamiques

Cette section explique comment promouvoir un travail critique planifié sur un pool dynamique. Un travail promu peut s'exécuter sur de nombreux agents dynamiques dans le pool dynamique, à l'inverse d'un travail non promu. Cela permet de s'assurer qu'un travail important s'exécute avant d'autres travaux qui sont moins importants.

Pour garantir qu'un travail critique obtient les ressources nécessaires et qu'il est traité dans les délais, spécifiez les variables suivantes :

tw.job.promoted

Cette variable indique si le travail est promu. Les valeurs prises en charge sont **YES** et **NO**. La valeur de cette variable s'applique à tous les travaux soumis dans l'environnement spécifié.

tw.job.resourcesForPromoted

Cette variable est définie dans la définition du pool dynamique et indique la quantité de ressources logiques requises affectée sur un pool dynamique pour promouvoir un travail. La valeur peut être **1** si le travail est promu ou **10** si le travail n'est pas promu. La quantité est indiquée avec cette notation : `${tw.job.resourcesForPromoted}`.

Lorsqu'un travail est planifié sur un pool dynamique, la valeur de la variable **tw.job.promoted** dans le travail détermine le comportement du pool dynamique :

- Si la valeur de la variable **tw.job.promoted** est **NO**, la valeur de la variable **tw.job.resourcesForPromoted** sur le pool dynamique est 10, ce qui signifie que peu de ressources correspondent au besoin.
- Si la valeur de la variable **tw.job.promoted** est **YES**, la valeur de la variable **tw.job.resourcesForPromoted** sur le pool dynamique est 1, ce qui signifie que plusieurs ressources correspondent au besoin car le pool dynamique inclut les postes de travail avec une quantité de ressource supérieure ou égale à 1 et pas uniquement les postes de travail avec une valeur supérieure ou égale à 10.

Par exemple, vous pouvez écrire un script qui vérifie la valeur affectée à la variable **tw.job.promoted** dans le travail et effectue différentes actions selon que le travail est promu ou non.

Ajout de fonctions dynamiques aux travaux IBM Workload Scheduler existants

Cette section décrit la façon de modifier un travail existant pour utiliser les fonctionnalités dynamiques fournies avec les agents dynamiques, les pools et les pools dynamiques.

Vous pouvez modifier vos travaux IBM Workload Scheduler existants pour utiliser les fonctionnalités dynamiques fournies avec les agents dynamiques, les pools et les pools dynamiques. Pour modifier un travail existant, procédez comme suit :

1. Installez le nombre de agents dynamiques requis.
2. Le cas échéant, affectez des agents dynamiques aux pools ou créez des pools dynamiques en fonction de vos exigences.
3. Analysez vos travaux IBM Workload Scheduler existants et décidez lesquels permettraient d'obtenir les meilleurs résultats lors de l'utilisation de la fonctionnalité dynamique.
4. Connectez-vous à Dynamic Workload Console.
5. A partir de la barre d'outils de navigation, cliquez sur **Administration > Conception de la charge de travail > Gestion des définitions de charge de travail**
6. Spécifiez un nom de moteur, distribué ou z/OS. Le concepteur de charge de travail s'ouvre. Les types et les caractéristiques des travaux varient selon que vous sélectionnez un moteur distribué ou z/OS.
7. Dans le panneau Liste de travail, sélectionnez **Rechercher > Définition du travail**. Le panneau de recherche s'affiche.
8. Entrez vos critères de recherche et cliquez sur **Rechercher**.
9. Sélectionnez un ou plusieurs travaux parmi les résultats de recherche et cliquez sur **Editer**. Les travaux sélectionnés s'affichent dans la partie droite du panneau pour modification.
10. Dans l'onglet **Général**, cliquez sur le bouton d'exploration de la zone **Poste de travail**. Le panneau de recherche s'affiche.
11. Entrez vos critères de recherche et cliquez sur **Rechercher**.
12. Sélectionnez le agent dynamique, pool ou pool dynamique approprié, puis cliquez sur **OK**. Le travail est à présent affecté au poste de travail spécifié et s'exécutera dessus lorsqu'il sera planifié.

Limitations de la planification dynamique

Fonctions et propriétés partiellement ou pas du tout prises en charge dans la planification dynamique

La planification dynamique prend en charge la plupart des fonctions de planification statique de IBM Workload Scheduler. Le tableau 114 liste certaines fonctions ou propriétés qui sont partiellement ou pas du tout prises en charge.

Tableau 114. Fonctions partiellement ou pas du tout prises en charge pour la planification dynamique

Fonction	agent dynamique
Automatisation d'une charge de travail gérée par des événements. Remarque : Pour plus de détails sur les types d'événements, voir <i>IBM Workload Scheduler - Guide d'utilisation et de référence : Annexes - Définitions des événements et actions d'automatisation d'une charge de travail gérée par des événements</i>	Événements TivoliWorkloadSchedulerObjectMonitor pris en charge.
	Événements FileMonitor pris en charge, sauf pour les systèmes IBM i.
	Événements TivoliWorkloadSchedulerApplicationMonitor non pris en charge.
Commandes d'utilitaire (datecalc, jobinfo, etc...).	Non pris en charge.

Limitations spécifiques aux travaux appartenant au flot de travaux USERJOBS lors de la planification dynamique

Fonctions et propriétés partiellement ou pas du tout prises en charge pour le flot de travaux *USERJOBS* lors de la planification dynamique.

Les travaux du flot de travaux *USERJOBS* prennent en charge la plupart des fonctions ou propriétés de la planification dynamique. Le tableau 115 liste certaines fonctions ou propriétés qui sont partiellement ou pas du tout prises en charge.

Tableau 115. Fonctions partiellement ou pas du tout prises en charge pour les travaux du flot de travaux *USERJOBS*

Fonction/Propriété	USERJOBS
Les types de travaux avec des options avancées peuvent perdre certaines informations lorsqu'ils sont déplacés vers le flot de travaux <i>USERJOBS</i> .	Il est possible que le type de travail soit manquant ou différent de celui d'origine.
Affichage des types de travaux avec des propriétés d'options avancées à l'aide de la ligne de commande conman .	Si vous exécutez la commande sj <nom_travail>; props , où <i><nom_travail></i> désigne les types de travaux avec le nom des options avancées, vous rencontrez les problèmes suivants : • La valeur de la zone Tâche de la section Informations générales est tronquée. Remarque : Ce problème concerne uniquement l'affichage du travail mais pas le travail lui-même. • La section Informations supplémentaires affiche le nom interne des propriétés à la place du nom externe traduit.
Affichage des types de travaux avec des options avancées à l'aide de Dynamic Workload Console.	Si vous utilisez les vues graphiques de Dynamic Workload Console, les travaux reflètent ne s'affichent pas avec les points standard.
Transmission de variables entre des travaux appartenant à la même instance du flot de travaux. Pour plus d'informations sur cette fonction, voir «Transmission de variables entre travaux», à la page 650.	Cette fonction n'est pas prise en charge. La transmission de variables de travail n'est pas résolue lorsque les travaux sont déplacés du flot de travaux initial vers le flot de travaux <i>USERJOBS</i> .

Chapitre 16. Utilisation des commandes d'utilitaire

Le présent chapitre décrit les commandes d'utilitaire IBM Workload Scheduler. Ces commandes, à l'exception de celles répertoriées ci-dessous, sont installées dans le répertoire *TWS_home/bin*. Les commandes d'utilitaire sont exécutées à partir de l'invite de commande du système d'exploitation.

StartUp est installée dans le répertoire *racine_TWS* et **version** dans le répertoire *racine_TWS/version*.

Description des commandes

Le tableau 116 contient la liste des commandes d'utilitaire avec, pour chaque commande, sa description et les systèmes d'exploitation qu'elle prend en charge.

Tableau 116. Liste des commandes d'utilitaire

Commande	Description	Système d'exploitation
at	Soumet un travail à exécuter à un moment donné.	UNIX
batch	Soumet un travail à exécuter dès que possible.	UNIX
cpuinfo	Renvoie les informations indiquées dans une définition de poste de travail.	UNIX, Windows
datecalc	Convertit la date et l'heure au format requis.	UNIX, Windows
supprimer	Supprime les fichiers script et les fichiers de liste standard par nom.	UNIX, Windows
evtdef	Importe/exporte des définitions d'événements personnalisées.	UNIX, Windows
evtsize	Définit la taille maximale des fichiers de messages d'événements.	UNIX, Windows
filemonitor	Recherche des modifications dans les fichiers (fichiers qui ont été créés ou modifiés)	UNIX, Windows
exportserverdata	Télécharge la liste des instances de courtier de charge de travail à partir de la base de données IBM Workload Scheduler et modifie un numéro de port ou un nom d'hôte.	UNIX, Windows
importserverdata	Télécharge la liste des instances de Dynamic Workload Broker dans la base de données IBM Workload Scheduler après l'édition du fichier temporaire pour changer un numéro de port ou un nom d'hôte.	UNIX, Windows
jobinfo	Renvoie des informations sur un travail.	UNIX, Windows
jobstdl	Renvoie le nom des chemins d'accès des fichiers de liste standard.	UNIX, Windows
listproc	Répertorie les processus. Cette commande n'est pas prise en charge.	Windows

Tableau 116. Liste des commandes d'utilitaire (suite)

Commande	Description	Système d'exploitation
killproc	Supprime les processus. Cette commande n'est pas prise en charge.	Windows
maestro	Renvoie le répertoire principal IBM Workload Scheduler.	UNIX, Windows
makecal	Permet de créer des agendas personnalisés.	UNIX, Windows
metronome.pl	Est remplacé par twc_inst_pull_info .	UNIX, Windows
morestdl	Affiche le contenu des listes standard.	UNIX, Windows
movehistorydata	Déplace les données présentes dans la base de données IBM Workload Scheduler vers les tables d'archivage.	UNIX, Windows
param	Crée, affiche et supprime des variables et des mots de passe utilisateur sur des agents dynamiques.	UNIX, Windows
parms	Affiche, modifie et ajoute des paramètres.	UNIX, Windows
release	Libère les unités d'une ressource.	UNIX, Windows
rmstdlist	Supprime les fichiers de liste standard en fonction de leur âge.	UNIX, Windows
sendevent	Envoie des événements génériques au serveur de processeur d'événement actif.	UNIX, Windows
showexec	Affiche des informations sur l'exécution des travaux.	UNIX
shutdown	Arrête le processus netman et, facultativement, WebSphere Application Server.	UNIX, Windows
ShutDownLwa	Arrête l'agent localement.	UNIX, Windows Remarque : Sur les systèmes UNIX, peut être exécuté par <i>l'util_TWS</i> ou par l'utilisateur racine uniquement.
StartUp	Démarre le processus netman et WebSphere Application Server éventuellement.	UNIX, Windows

Tableau 116. Liste des commandes d'utilitaire (suite)

Commande	Description	Système d'exploitation
StartUpLwa	Démarre l'agent localement.	UNIX, Windows Remarque : Sur les systèmes UNIX, peut être exécuté par l'util_TWS ou par l'utilisateur racine uniquement.
twc_inst_pull_info	Collecte des données sur l'instance IBM Workload Scheduler locale, le poste de travail WebSphere Application Server et DB2 à des fins de diagnostic. Cette commande est décrite dans <i>IBM Workload Scheduler - Guide d'identification et de résolution des problèmes</i> .	UNIX, Windows
version	Affiche les informations de version.	UNIX

at et batch

Soumettez les commandes et travaux ad hoc devant être lancés par IBM Workload Scheduler.

Ces commandes s'exécutent uniquement sous UNIX.

Pour plus d'informations sur la disponibilité pour les utilisateurs, reportez-vous aux commandes **at.allow** et **at.deny** ci-dessous.

Syntaxe

at -V | -U

at {-sflot_travaux | -q file_attente} spéc_heure

batch -V | -U

batch [-s flot_travaux]

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-s flot_travaux

Indique l'ID *ID_flot_travaux* de l'instance du flot de travaux dans laquelle le travail est soumis. Si aucune instance de flot de travaux avec cet *ID_flot_travaux* n'existe, un flot de travaux est créé avec *flot_travaux* comme alias et *ID_flot_travaux*. Le nom du flot doit commencer par une lettre et peut contenir des caractères alphanumériques et des tirets. Il peut comporter jusqu'à 16 caractères.

Si les arguments **-s** et **-q** sont omis, un nom de flot de travaux est sélectionné sur la base de la valeur de la variable d'environnement *ATSCRIPT*. Si *ATSCRIPT* contient le mot **maestro**, l'alias du flot de travaux correspond aux huit premiers caractères du nom de groupe de l'utilisateur. Si *ATSCRIPT* n'est pas définie ou comporte une valeur autre que **maestro**, l'alias du flot de travaux sera **at** (pour les travaux soumis avec **at**) ou **batch** (pour les travaux soumis avec **batch**).

Pour plus d'informations sur les flots de travaux, voir «Autres remarques», à la page 682.

Les mots clés suivants s'appliquent uniquement aux travaux **at** :

-q*file_attente*

Indique que le travail doit être soumis dans un flot de travaux portant le nom *file_attente*, lequel ne peut comporter qu'une seule lettre (comprise entre a et z). Pour plus d'informations sur les flots de travaux, voir «Autres remarques», à la page 682.

spéc-heure

Indique l'heure à laquelle le travail va être lancé. La syntaxe est identique à celle de la commande UNIX **at**.

Commentaires

Après avoir entré **at** ou **batch**, lancez les commandes qui constituent le travail. Pour terminer chaque ligne d'entrée, appuyez sur la touche de retour. Le programme se termine par une marque de fin de fichier (généralement **Ctrl+d**) ou par une ligne avec un point (.). Vous avez également la possibilité d'utiliser le signe inférieur (<) pour lire des commandes à partir d'un fichier. Voir «Exemples».

Les informations relatives aux travaux **at** et **batch** sont transmises au gestionnaire de domaine maître où les travaux sont ajoutés dans des flots de travaux du plan de production, Symphony. Les travaux sont lancés en fonction des dépendances incluses dans les flots de travaux.

Le shell UNIX utilisé pour exécuter les travaux soumis via les commandes **at** et **batch** est déterminé par la variable *SHELL_TYPE* du script de configuration *jobmanrc*. N'utilisez pas le shell C. Pour plus d'informations, voir «Personnalisation du traitement des travaux sur un poste de travail UNIX - *jobmanrc*», à la page 57.

Une fois soumis, les travaux sont lancés de la même manière que d'autres travaux planifiés. Chaque travail est exécuté dans l'environnement de l'utilisateur qui le soumet. Pour vous assurer que l'environnement est complet, des commandes **set** sont insérées dans le script pour que celui-ci corresponde aux valeurs des variables de l'environnement de l'utilisateur.

Exemples

Pour soumettre un travail dans le flot de travaux *ID_flot_travaux sched8* qui sera lancé dès que possible, exécutez la commande suivante :

```
batch -s sched8
commande <Retour>
...
<Control d>
```

Pour soumettre un travail qui sera lancé deux heures après la saisie de la commande, exécutez la commande suivante :

```
at now + 2 hours
commande <Retour>
...
<Control d>
```

Si la variable *ATSCRIPT* est indéfinie (null), le travail est soumis dans un flot de travaux dont le nom est identique à celui du groupe de l'utilisateur. Sinon, il est soumis dans un flot de travaux intitulé *at*.

Pour soumettre un travail dans une instance de flot de travaux avec *ID_flot_travaux* sked-mis en vue d'un lancement à 17:30, exécutez la commande suivante :

```
at -s sked-mis 17h30
commande <Retour>
...
<Control d>
```

La commande suivante est identique à celle de l'exemple précédent, excepté que le programme lit les commandes du travail à partir d'un fichier :

```
at -s sked-mis 17h30 < ./mon_travail
```

Le fait que les commandes soient lues à partir d'un fichier ne modifie pas la façon dont elles sont traitées. Cela signifie que les commandes sont copiées du fichier *./mon_travail* dans un fichier script.

Remplacement des commandes UNIX

Les commandes UNIX **at** et **batch** standard peuvent être remplacées par des commandes IBM Workload Scheduler. Dans l'exemple suivant, nous expliquons comment remplacer les commandes UNIX **at** et **batch** :

```
$ mv /usr/bin/at /usr/bin/uat
$ mv /usr/bin/batch /usr/bin/ubatch
$ ln -s racine_TWS/bin/at /usr/bin/at
$ ln -s racine_TWS/bin/batch /usr/bin/batch
```

Fichiers **at.allow** et **at.deny**

Les commandes **at** et **batch** utilisent les fichiers */usr/lib/cron/at.allow* et */usr/lib/cron/at.deny* pour limiter l'accès. Si le fichier *at.allow* existe, seuls les utilisateurs répertoriés dans le fichier sont autorisés à utiliser **at** et **batch**. Si le fichier n'existe pas, *at.deny* est examiné pour vérifier si les droits d'accès de l'utilisateur sont explicitement refusés. S'il n'existe aucun fichier, seul l'utilisateur **root** est autorisé à utiliser les commandes.

Fichiers script

Les commandes entrées avec **at** ou **batch** sont stockées dans des fichiers script. Ces derniers sont créés par IBM Workload Scheduler à l'aide de la convention d'appellation suivante :

racine_TWS/atjobs/epoch.sss

où :

epoch Indique le nombre de secondes écoulées depuis 00:00, 1/1/70.

sss Représente les trois premiers caractères du nom du flot de travaux.

Remarque : IBM Workload Scheduler supprime les fichiers script des travaux non reportés. Vous devez toutefois surveiller l'espace disponible dans le répertoire *atjobs* et supprimer les fichiers caducs si nécessaire.

Noms de travail

IBM Workload Scheduler attribue un nom unique à chaque travail **at** et **batch** lors de la soumission. Les noms sont constitués de l'ID processus de l'utilisateur, précédé du nom d'utilisateur, tronqué pour ne plus comporter que huit caractères. Le nom résultant est mis en majuscules.

Autres remarques

- Il est conseillé de créer à l'avance, via le programme `composer`, les flots de travaux dans lesquels les travaux **at** et **batch** sont soumis. Les flots de travaux peuvent contenir des dépendances qui déterminent l'heure de lancement des travaux. Les flots de travaux doivent contenir au moins le mot clé **carryforward**. Ainsi, les travaux qui n'aboutissent pas ou qui ne sont pas lancés pendant le plan de production courant sont reportés au plan de production suivant.
- Indiquez l'expression **on everyday** pour que les flots de travaux soient sélectionnés chaque jour.
- Utilisez le mot clé **limit** pour limiter le nombre des travaux soumis qui peuvent être exécutés simultanément.
- Utilisez le mot clé **priority** pour définir la priorité des travaux soumis par rapport aux autres travaux.

Si la valeur de temps est antérieure à l'heure courante, elle est prise en compte pour le jour suivant. Si la valeur de temps est postérieure à l'heure courante, elle est prise en compte pour le jour courant.

cpuinfo

Renvoie les informations indiquées dans une définition de poste de travail.

Syntaxe

cpuinfo -V | -U

cpuinfo *poste_de_travail* [*type_info*] [...]

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.

Poste de travail

Nom du poste de travail.

type_info

Type d'information à afficher. Indiquez un ou plusieurs des éléments suivants :

os_type

Renvoie la valeur de la zone **os** : **UNIX**, **WNT**, **ZOS**, **OTHER** et **IBM i**. La valeur **ZOS** s'applique uniquement aux postes de travail de moteur distant utilisés pour communiquer avec un contrôleur IBM Workload Scheduler for z/OS.

node

Renvoie la valeur de la zone **node**. Pour un serveur courtier de charge de travail, il s'agit du nom d'hôte ou de l'adresse TCP/IP du poste de travail sur lequel vous avez installé IBM Workload Scheduler Bridge. Pour un poste de travail distant, il s'agit du nom

d'hôte du poste de travail sur lequel le moteur distant est installé. Dans les autres cas, indiquez le nom d'hôte ou l'adresse TCP/IP du poste de travail.

port Renvoie la valeur de la zone **tcpaddr**. Si vous définissez un poste de travail Workload Broker, spécifiez la valeur de la propriété **TWS.Agent.Port** du fichier **TWSAgentConfig.properties**. Pour les postes de travail de moteur distant, la valeur de cette zone est le numéro de port HTTP utilisé par le moteur distant. Si le protocole HTTPS est utilisé, la valeur de cette zone est 3111.

sslport Renvoie la valeur de la zone **secureaddr**. Il s'agit du port utilisé pour écouter les connexions SSL entrantes. Pour les postes de travail de moteur distant, la valeur de cette zone est le numéro de port HTTPS utilisé par le moteur distant. Si le protocole HTTP est utilisé, la valeur de cette zone est 3113.

protocole Renvoie la valeur de la zone **protocol** : HTTP ou HTTPS. Lorsque le type de poste de travail est un moteur distant, cette valeur indique le protocole utilisé pour communiquer entre le serveur courtier et le moteur distant.

sec_level Renvoie la valeur de la zone **securitylevel** : NONE, ENABLED, ON ou FORCE.

autolink Renvoie la valeur de la zone **autolink** : ON ou OFF.

fullstatus Renvoie la valeur de la zone **fullstatus** : ON ou OFF.

resolvedep Renvoie ON ou OFF. N'est plus utilisé dans la version 8.6.

behindfirewall Renvoie la valeur de la zone **behindfirewall** : ON ou OFF.

host Renvoie la valeur de la zone **host**. Il s'agit du nom du poste de travail hébergeant l'agent.

domain Renvoie la valeur de la zone **domain**.

ID Renvoie l'identificateur d'agent utilisé par le poste de travail lorsqu'il est connecté au serveur courtier. Pour les postes de travail de type : AGENT, REM-ENG, POOL, D-POOL.

method Pour les agents étendus et réseau uniquement. Renvoie la valeur de la zone **access**.

server Renvoie la valeur de la zone **server**.

type Renvoie la valeur de la zone **type**. Indique le type de poste de travail : MASTER, MANAGER, FTA, S-AGENT, REM-ENG, AGENT, POOL, D-POOL et X-AGENT.

time_zone Renvoie la valeur de la zone **timezone**. Indique le fuseau horaire

du poste de travail. Pour un agent étendu, la zone est vide. Pour un poste de travail de moteur distant, il s'agit du fuseau horaire du moteur distant.

version

Renvoie la version de IBM Workload Scheduler en cours d'exécution sur le poste de travail. Pour un agent étendu, la zone est vide.

info

Renvoie la version du système d'exploitation et le modèle du poste de travail. Pour les agents étendus, la zone est vide. Pour les postes de travail de moteur distant, cette zone affiche le moteur distant.

Commentaires

Les valeurs sont renvoyées, à raison d'une par ligne, dans l'ordre de saisie des arguments sur la ligne de commande. Si aucun argument n'est indiqué, toutes les informations applicables sont renvoyées avec des libellés, à raison d'une par ligne.

Exemples

Les exemples ci-dessous sont basés sur la définition de poste de travail suivante :

Nom du poste de travail	Type	Domaine	Mis à jour le	Verrouillé par
RE-ZOS	REM-ENG	-	09/06/2010	-

```
CPUNAME RE-ZOS
OS ZOS
NODE 9.168.119.189 TCPADDR 635
FOR MAESTRO HOST NC123162_DWB
TYPE REM-ENG
PROTOCOL HTTP
FIN
```

Pour imprimer le **type** et le **protocole** du poste de travail RE-ZOS, exécutez la commande suivante :

```
>cpuinfo RE-ZOS type protocol
REM-ENG
HTTP
```

Pour imprimer toutes les informations du poste de travail RE-ZOS, exécutez la commande suivante :

```
>cpuinfo RE-ZOS
OS_TYPE: ZOS
NODE: 9.168.119.189
PORT: 635
SSLPORT: 31113
ENGINEADDR: 0
PROTOCOLE: HTTP
AUTOLINK: OFF
FULLSTATUS: OFF
RESOLVEDEP: OFF
BEHINDFIREWALL: OFF
HOST: NC123162_DWB
DOMAIN: MASTERDM
ID: D795263CBCD2365CA7B5C5BC0C3DD363
SERVER:
```

TYPE: REM-ENG
TIME_ZONE: Europe/Rome
VERSION: 8.6
INFO: Remote Engine

datecalc

Exécutez l'utilitaire datecalc sur un poste de travail de gestionnaire de domaine maître ou d'agent tolérant aux pannes (non pris en charge sur les agents dynamiques) afin de résoudre les expressions de date et les dates de retour au format de votre choix.

Syntaxe

datecalc -V | -U

datecalc *date-base*
[*décalage*]
[**pic** *format*]
[**freedays** *nom_agenda* [-**sa**] [-**su**]]

datecalc -t *heure*
[*date-base*]
[*décalage*]
[**pic** *format*]

datecalc *aaaammjjhhtt*
[*décalage*]
[**pic** *format*]

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.

base-date

Indiquez l'un des arguments suivants :

jour | *date* | **today** | **tomorrow** | **scheddate**

où :

jour Indique un jour de la semaine. Les valeurs admises sont les suivantes : **su**, **mo**, **tu**, **we**, **th**, **fr** ou **sa**.

date Indique une date au format *élément/élément[/élément]*, où *élément* représente : *j[j]*, *m[m]* et *aa[aa]*. Tout autre forma de date n'est pas valide.

Si deux chiffres sont utilisés pour l'année (*aa*), un nombre supérieur à 70 correspond à une date du 20e siècle et un nombre inférieur à 70 correspond à une date du 21e siècle.

Le paramètre se réfère à la date en cours, et non à la commande *date* d'UNIX. L'exemple ci-après illustre une option utilisant la sortie *date* d'UNIX comme entrée pour le paramètre *date* de IBM Workload Scheduler.

```
hdate=`date +"%m/%d/%y"`
echo $hdate
datecalc $hdate pic mm/dd/yyyy
```

Les valeurs admises pour le mois (*m[m]*) sont **jan, feb, mar, apr, may, jun, jul, aug, sep, oct, nov** ou **dec**.

Les barres obliques (/) peuvent être remplacées par des tirets (-), des points (.), des virgules n,) ou des espaces. Par exemple, vous pouvez entrer une des dates suivantes pour le 28 mars 2005 :

```
03/28/05
3-28-2005
28.mar.05
05,28,3
mar 28 2005
28 3 05
```

Si des nombres sont utilisés, il est possible d'entrer une date ambiguë, par exemple, 2,7,04. Dans ce cas, **datecalc** utilise le format de date défini dans le catalogue de messages IBM Workload Scheduler pour interpréter la date. Si celle-ci ne respecte pas le format, **datecalc** génère un message d'erreur.

today Indique la date système en cours.

tomorrow

Indique la date système en cours plus un jour ou, dans le cas des calculs d'heure, plus 24 heures.

scheddate

Indique la date du plan de production. Celle-ci peut être différente de la date système. Si vous l'utilisez dans les travaux d'un flot de travaux qui n'est pas reporté, le programme renvoie la date à laquelle le travail doit être exécuté. Elle peut être différente de la date de production du flot de travaux si le travail possède une dépendance at.

Si vous l'utilisez dans les travaux d'un flot de travaux qui est reporté, le programme renvoie la date à laquelle le travail aurait dû être exécuté. Elle peut être différente de la date de production du flot de travaux reporté si le travail possède une dépendance at. Si la dépendance at respecte la syntaxe suivante : **at=hlmm + n days**, les *n days* ne seront pas ajoutés à la variable **TIVOLI_JOB_DATE**. Par conséquent, la commande **datecalc** ne reportera pas les jours en question.

Par exemple, considérez un plan pour la journée du 15/01/2015 avec un début de journée défini à 0700, et le planning suivant

```
SCHEDULE NET92A#JS0200CF
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 0200
:
NET92A#DATECALC
END
```

Si le travail s'exécute à 0200, **datecalc** renvoie l'heure 0200 de la journée du 16/01/2015. Si le planning est reporté et que le travail s'exécute à 1000, le résultat enregistré pour **datecalc** est l'heure 1000 de la journée du 15/01/2015.

-t *heure* [*date-base*]

Indiquez *heure* dans l'un des formats suivants :

now | **noon** | **midnight** | [*h*][*h*][:*mm*] [**am** | **pm**] [**zulu**]

où :

now Indique la date et l'heure système en cours.

noon Indique midi (ou 1200).

midnight

Indique minuit (ou 0000).

h[*h*][:*mm*]

Indique l'heure au format 12 heures (si des heures **am** ou **pm** sont utilisées) ou au format 24 heures. Le délimiteur (:) facultatif peut être remplacé par un point (.), une virgule (,), une apostrophe ('), la lettre **h** ou un espace. Par exemple, vous pouvez utiliser les formats suivants pour représenter l'heure 20:00 :

8:00pm

20:00

0800pm

2000

8pm

20

8,00pm

20.00

8\'00pm

20 00

zulu Indique que l'heure entrée correspond à l'heure de Greenwich (heure universelle). **datecalc** la convertit en heure locale.

aaaammjjhhtt

Indique l'année, le mois, le jour, l'heure et les minutes, exprimés en douze chiffres exactement. Par exemple, tapez **200505070915** pour représenter le 7 mai 2005, 9 heures 15.

décalage

Indique un décalage par rapport à la *date-base* au format suivant :

{[+ | > | - | < *nombre* | **nearest**] | **next**} **day[s]** | **weekday[s]** | **workday[s]** | **week[s]** | **month[s]** | **year[s]** | **hour[s]** | **minute[s]** | *jour* | *agenda*

où :

+ | > Indique un décalage par rapport à une date ou une heure ultérieures. Utilisez + (signe plus) sous Windows ; utilisez > (supérieur à) sous UNIX. Veillez à ajouter une barre oblique inversée (\) avant le crochet (>).

- | < Indique un décalage par rapport à une date ou une heure antérieures. Utilisez - (signe moins) sous Windows ; utilisez < (inférieur à) sous UNIX. Veillez à ajouter une barre oblique inversée (\) avant le crochet (>).

nombre Nombre d'unités du type indiqué.

nearest

Indique un décalage par rapport à l'occurrence la plus proche du type d'unité (antérieure ou ultérieure).

next Indique l'occurrence suivante du type d'unité.

day[s] Indique tous les jours.

weekday[s]

Indique tous les jours sauf le samedi et le dimanche.

workday[s]

Identique à **weekday[s]**, mais exclut également les dates de l'agenda **holidays**.

week[s]

Indique sept jours.

month[s]

Indique des mois de l'agenda.

year[s]

Indique des années de l'agenda.

hour[s]

Indique des heures d'horloge.

minute[s]

Indique des minutes d'horloge.

jour Indique un jour de la semaine. Les valeurs admises sont les suivantes : **su**, **mo**, **tu**, **we**, **th**, **fr** ou **sa**.

agenda Indique les entrées d'un agenda portant ce nom.

pic format

Indique le format de renvoi de la date et de l'heure. Les caractères du *format* ont les significations suivantes :

m Mois.

d Jour du mois.

y Année.

j Jour du mois du calendrier julien.

h Heure.

t Minute.

^|/ Un espace. Utilisez / (barre oblique) sous Windows ; utilisez ^ (caret) sous UNIX (ajoutez une barre oblique inversée (\) avant le caret (^) si vous vous trouvez dans le shell Bourne).

Vous pouvez également inclure des signes de ponctuation. Il s'agit des délimiteurs utilisés dans *date* et *heure*.

Si aucun format n'est défini, **datecalc** renvoie la date et l'heure au format défini par les variables d'environnement NLS (Native Language Support). Si les variables NLS ne sont pas définies, la langue nationale C est utilisée par défaut.

freedays

Indique le nom d'un agenda de jours chômés *nom_agenda* devant remplacer les **holidays** dans l'évaluation des *jours ouvrés*.

Dans le cas présent, les *jours ouvrés* sont considérés comme étant *tous les jours de la semaine* sauf *samedi*, *dimanche* et toutes les dates répertoriées dans *Nom_agenda*.

Par défaut, *saturday* et *sunday* ne sont pas considérés comme des *jours ouvrés*, sauf si vous indiquez explicitement le contraire en ajoutant **-sa** et **-su** après *nom_agenda*.

Vous pouvez également affecter le nom **holidays** à l'agenda des jours chômés.

Exemples

Pour renvoyer la date suivante (à compter du jour courant) de l'agenda de fin de mois monthend, exécutez la commande suivante :

```
>datecalc today next monthend
```

Dans les exemples suivants, la date système en cours est le vendredi 16 avril 2006.

```
>datecalc today +2 days pic mm/jj/aaaa
04/16/2006

>datecalc today next tu pic aaaa\^mm\^jj
2006 04 16

>LANG=american;export LANG
>datecalc -t 14:30 tomorrow
Sat, Apr 17, 2006 02:30:00 PM

>LANG=french;datecalc -t 14:30 tomorrow
Samedi 17 avril 2006 14:30:00
```

Dans l'exemple suivant, l'heure système en cours est 10:24.

```
>datecalc -t now \> 4 hours pic hh:tt
14:24
```

datamigrate

La commande d'utilitaire **datamigrate** du gestionnaire de domaine maître vous permet d'importer dans la base de données les données enregistrées dans les fichiers à plat créés à l'aide de la ligne de commande **composer extract** dans une autre instance de IBM Workload Scheduler, version 8.3 ou ultérieure.

Syntaxe

Utilisez la syntaxe suivante pour importer des données :

```
datamigrate -u | -v
```

```
datamigrate -<type_objet> <fichier_type_objet> [-tmppath <répertoire_temporaire>]
```

Arguments

-u Affiche des informations sur la syntaxe de la commande et quitte l'application.

-v Affiche la version de la commande et quitte l'application.

-<type_objet> <fichier_type_objet>

-topology nom_fichier_topologie

Importe toutes les définitions de domaine, de poste de travail et de classe de poste de travail enregistrées dans le fichier

nom_fichier_topologie, créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-prompts *nom_fichier_invites*

Importe toutes les définitions d'invite enregistrées dans le fichier *nom_fichier_invites* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-calendars *nom_fichier_calendriers*

Importe toutes les définitions de calendrier enregistrées dans le fichier *nom_fichier_calendriers* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-parms *nom_fichier_params*

Importe toutes les définitions de paramètre enregistrées dans le fichier *nom_fichier_params* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-resources *nom_fichier_ressources*

Importe toutes les définitions de ressource enregistrées dans le fichier *nom_fichier_ressources* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-rcgroups *nom_fichier_groupece*

Importe toutes les définitions de groupe de cycle d'exécution enregistrées dans le fichier *nom_fichier_groupece* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-users *nom_fichier_utilisateurs*

Importe toutes les définitions d'utilisateurs enregistrées dans le fichier *nom_fichier_utilisateurs* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

Remarque : Veillez à remplacer toutes les entrées "*****" du fichier d'utilisateurs par les véritables mots de passe avant de lancer la commande d'utilitaire **datamigrate**.

-jobs *nom_fichier_travail*

Importe toutes les définitions de travail enregistrées dans le fichier *nom_fichier_travail* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-scheds *nom_fichier_planning*

Importe toutes les définitions de flot de travaux (calendrier) enregistrées dans le fichier *nom_fichier_planning* créé à l'aide de la commande **composer** dans l'instance de IBM Workload Scheduler.

-tmppath *chemin_temp*

chemin_temp désigne le chemin d'accès temporaire où la commande **datamigrate** peut insérer ses fichiers de travail temporaires. Le chemin d'accès par défaut est <rép_installation_TWS>\tmp sur les systèmes d'exploitation Windows, et <rép_installation_TWS>/tmp sur les systèmes UNIX.

Commentaires

Si vous préférez l'utilisation de la commande **datamigrate** à celle de la commande **composer** avec les options add ou replace, veillez à suivre l'ordre d'importation des données qui convient.

Exemples

Pour importer les données exportées de IBM Workload Scheduler version 8.6 dans la base de données IBM Workload Scheduler version 9.2, lancez les commandes suivantes dans l'ordre indiqué :

```
datamigrate -topology topology86.txt
```

```
datamigrate -prompts prompt86.txt
```

```
datamigrate -calendars cal86.txt
```

```
datamigrate -parms parms86.txt
```

```
datamigrate -resources res86.txt
```

```
datamigrate -rcgroups rcs86.txt
```

```
datamigrate -users users86.txt
```

```
datamigrate -jobs jobs86.txt
```

```
datamigrate -scheds js86.txt
```

delete

Supprime des fichiers. Bien que cette commande permette de supprimer des fichiers de la liste standard, il est conseillé d'utiliser plutôt la commande `rmstdlist`. Les utilisateurs **maestro** et **root** (sous UNIX) et **Administrateur** (sous Windows) peuvent supprimer tous les fichiers. Les autres utilisateurs ne peuvent supprimer que les fichiers associés à leurs propres travaux.

Syntaxe

delete -V | -U

delete *nom_fichier*

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

nom_fichier

Indique le nom du fichier ou du groupe de fichiers à supprimer. Ce nom doit être placé entre guillemets (") s'il contient des caractères autres que les suivants : caractères alphanumériques, tirets (-), barres obliques (/), barres obliques inversées (\) et traits de soulignement (_). Les caractères génériques sont acceptés.

Remarque : Utilisez cette commande avec précaution. L'utilisation incorrecte de caractères génériques peut entraîner la suppression accidentelle des fichiers.

Exemples

Pour supprimer tous les fichiers de liste standard du 4 novembre 2004, exécutez la commande suivante :

```
delete d:\win32app\maestro\stdlist\2004.4.11\@
```

Le script suivant, inclus dans un travail planifié sous UNIX, supprime le fichier de liste standard de ce travail s'il n'y a pas d'erreur :

```
...
#Suppression de la stdlist de ce travail :
if grep -i error $UNISON_STDLIST
then
exit 1
else
`maestro`/bin/delete $UNISON_STDLIST
fi
...
```

Le script de configuration standard, `jobmanrc`, affecte à la variable **UNISON_STDLIST** le nom du fichier de liste standard du travail. Pour plus d'informations sur `jobmanrc`, voir «Personnalisation du traitement des travaux sur un poste de travail UNIX - `jobmanrc`», à la page 57.

evtdef

Importe/exporte un fichier de définition XML pour le fournisseur d'événements génériques, dans lequel vous pouvez ajouter ou modifier les types d'événements personnalisés. Vous pouvez ensuite, via la commande **sendevent**, envoyer ces événements au serveur de traitement d'événements. Pour plus d'informations, voir aussi «Définition d'événements personnalisés», à la page 161.

Syntaxe

evtdef -U | -V

evtdef [*paramètres de connexion*] **dumpdef** *chemin d'accès au fichier*

evtdef [*paramètres de connexion*] **loaddef** *chemin d'accès au fichier*

Arguments

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

paramètres de connexion

Si vous utilisez **evtdef** à partir du gestionnaire de domaine maître, les paramètres de connexion ont été configurés lors de l'installation et n'ont pas besoin d'être fournis, à moins que vous ne vouliez pas utiliser les valeurs par défaut.

Si vous utilisez **evtdef** à partir du client de ligne de commande sur un autre poste de travail, les paramètres de connexion peuvent être fournis par l'une des méthodes suivantes :

- Ils sont stockés dans le fichier `localopts`
- Ils sont stockés dans le fichier `useropts`
- Ils sont fournis à la commande dans un fichier de paramètres
- Ils sont fournis à la commande au sein d'une chaîne de commande

Pour une présentation de ces options, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

Pour des détails complets sur les paramètres de configuration, voir la rubrique relative à la configuration de l'accès du client de ligne de commande dans *IBM Workload Scheduler - Guide d'administration*.

dumpdef *chemin d'accès au fichier*

Télécharge le fichier XML du fournisseur d'événements génériques. Le fichier est téléchargé avec le nom et le chemin d'accès spécifiés par la variable *chemin_d'accès_au_fichier*. Vous pouvez éditer le fichier pour y ajouter des types d'événements personnalisés.

Le nom du fournisseur d'événements génériques fourni avec le produit est GenericEventPlugIn. Vous pouvez modifier ce nom en intervenant sur la balise name du mot clé eventPlugIn.

Important : Vous devez utiliser ce nom comme valeur :

- du mot clé source de la commande «sendevent», à la page 719
- du mot clé eventProvider dans la définition des règles d'événement déclenchées par ces événements personnalisés.

loaddef *chemin d'accès au fichier*

Charge le fichier XML modifié du fournisseur d'événements génériques à partir du nom de fichier et du chemin d'accès spécifiés dans la variable *chemin_d'accès_au_fichier*.

Commentaires

Les schémas linguistiques suivants sont appliqués pour valider les définitions de règles et, selon l'éditeur XML dont vous êtes équipé, pour fournir une aide à la syntaxe::

- eventDefinitions.xsd
- common.xsd

Les fichiers sont situés dans le sous-répertoire schemas du répertoire d'installation de IBM Workload Scheduler.

Lorsque vous téléchargez le fichier modèle du fournisseur d'événements génériques, ce dernier se présente comme suit :

```
<?xml version="1.0" encoding="UTF-8"?>
<eventDefinitions
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/plugins/events"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
plugins/events/eventDefinitions.xsd" >
<eventPlugIn>
<complexName displayName="Événement personnalisé" name="GenericEventPlugIn" />
<scopes>
<scope name="Generic">
<scopedef text="{Param1} on {Workstation}" />
</scope>
</scopes>
<!-- Generic Event -->
<event baseAliasName="genericEvt" scope="Generic">
<complexName displayName="Événement générique" name="Event1" />
<displayDescription>L'événement est envoyé lorsqu'une correspondance avec
l'expression spécifiée est trouvée.</displayDescription>
<property type="string" required="true" wildcardAllowed="true"
multipleFilters="true" minlength="1">
<complexName displayName="Paramètre 1" name="Param1" />
<displayDescription>Valeur de Paramètre 1</displayDescription>
</property>
<property type="string" required="true" wildcardAllowed="false">
```

```

multipleFilters="false" minlength="1">
<complexName displayName="Poste de travail" name="Workstation" />
<displayDescription>Poste de travail pour lequel l'événement est
généré.</displayDescription>
</property>
</event>
</eventPlugin>
</eventDefinitions>

```

Vous pouvez ensuite éditer ce fichier pour ajouter les types de propriété dont vous avez besoin afin de définir un événement spécifique. Les types de propriété suivants sont disponibles :

Tableau 117. Propriétés supplémentaires pouvant être utilisées pour la définition d'événements personnalisés.

Type de propriété	A ajouter dans le fichier d'événements XML comme illustré
boolean	<pre> <property type="boolean" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Boolean field" name="Boolean"/> <displayDescription>Add a boolean field</displayDescription> </property> </pre>
date	<pre> <property type="date" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Date field" name="Date"/> <displayDescription>Add a date field</displayDescription> </property> </pre>
datetime	<pre> <property type="datetime" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Date Time field" name="Datetime"/> <displayDescription>Add a date time field</displayDescription> </property> </pre>
datetimeutc	<pre> <property type="datetimeutc" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Date Time UTC field" name="Datetimeutc"/> <displayDescription>Add a date time UTC field</displayDescription> </property> </pre>
duration	<pre> <property type="duration" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Duration field" name="Duration"/> <displayDescription>Add a duration field</displayDescription> </property> </pre>
fileSize	<pre> <property type="fileSize" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="File size field" name="filesize"/> <displayDescription>Add a file size field</displayDescription> </property> </pre>
nonnegativeinteger	<pre> <property type="nonnegativeinteger" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Non negativeinteger field" name="nonnegativeinteger"/> <displayDescription>Add a non negativeinteger field</displayDescription> </property> </pre>
numeric	<pre> <property type="numeric" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <property type="numeric" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <displayDescription>Add a numeric field</displayDescription> </property> </pre>
percentage	<pre> <property type="percentage" required="false" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="Percentage field" name="percentage"/> <displayDescription>Add a percentage field</displayDescription> </property> </pre>

Tableau 117. Propriétés supplémentaires pouvant être utilisées pour la définition d'événements personnalisés. (suite)

Type de propriété	A ajouter dans le fichier d'événements XML comme illustré
chaîne	<pre> <property type="string" required="true" wildcardAllowed="true" multipleFilters="true" minlength="1"> <complexName displayName="String with wildcards" name="StringWithWildcards"/> <displayDescription>Add a string with wildcards</displayDescription> </property> ou <property type="string" required="true" wildcardAllowed="false" multipleFilters="true" minlength="1"> <complexName displayName="String without wildcards" name="StringWithoutWildcards"/> <displayDescription>Add a string without wildcards</displayDescription> </property> </pre>

Vous pouvez modifier les valeurs de tous les attributs de propriété, sauf type, pour répondre à vos besoins.

Les propriétés ainsi définies sont converties dans les zones d'entrée après le téléchargement de la définition d'événement et l'ouverture de cette dernière dans Dynamic Workload Console.

Vous pouvez également définir plusieurs événements en répétant les sections `<eventPlugin>...</eventPlugin>`. Par exemple :

```

<?xml version="1.0" encoding="UTF-8"?>
<eventDefinitions
xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/plugins/events"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/
plugins/events/eventDefinitions.xsd" >
  <eventPlugin>
    <complexName displayName="Événement personnalisé" name="GenericEventPlugIn" />
    <scopes>
      <scope name="Art042StockQuantity">
        <scopedef text="{Art042pieces}" />
      </scope>
    </scopes>
    <!-- Generic Event -->
    <event baseAliasName="genericEvt" scope="Generic">
      <complexName displayName="Stock of article 042 reaches minimum level" name="art042qty"/>
      <displayDescription>The event is sent when the number of art.42 items on stock reaches
      minimum level.</displayDescription>
      <property type="numeric" required="true" wildcardAllowed="false"
      multipleFilters="true" minlength="1">
        <complexName displayName="Art042 items on stock" name="art042items"/>
        <displayDescription>The number of art042 items left</displayDescription>
      </property>
    </event>
    <!-- Generic Event 2-->
    <event baseAliasName="Hard drive saturation" scope="Generic">
      <complexName displayName="Hard drive saturation" name="HDSatEvent"/>
      <displayDescription>displayDescription>The event is sent when the percentage field
      reaches the warning level.</displayDescription>
      <property type="percentage" required="true" wildcardAllowed="false"
      multipleFilters="true" minlength="1">
        <complexName displayName="Percentage Full" name="PercentFull"/>
        <displayDescription>The percentage of total disk space used</displayDescription>
      </property>
      <property type="string" required="true" wildcardAllowed="false"
      multipleFilters="false" minlength="1">
        <complexName displayName="Poste de travail" name="Workstation" />
        <displayDescription>The workstation where the hard drive is installed</displayDescription>

```

```

</property>
</event>
</eventPlugin>
</eventDefinitions>

```

Exemples

Dans cet exemple, vous pouvez :

1. Télécharger le fichier XML du fournisseur d'événements génériques sous la forme `c:\custom\myevents.xml`
`evtdef dumpdef c:\custom\myevents.xml`
2. Editer le fichier pour y ajouter des type d'événements personnalisés.
3. Une fois l'opération terminée, téléchargez le fichier XML du fournisseur d'événements génériques à partir du fichier `c:\custom\myevents.xml`.
`evtdef loaddef c:\custom\myevents.xml`

evtsize

Définit la taille des fichiers message de IBM Workload Scheduler. Cette commande est utilisée par l'administrateur IBM Workload Scheduler pour augmenter la taille d'un fichier de messages après la réception du message "End of file on events file" (fin du fichier des événements atteinte) ou pour contrôler la taille de la file d'attente de messages contenue dans le fichier de messages. Vous devez être un utilisateur **maestro** ou **root** sous UNIX, ou **Administrateur** sous Windows pour exécuter **evtsize**. Arrêtez le moteur IBM Workload Scheduler avant d'exécuter cette commande.

Syntaxe

evtsize **-V** | **-U**

evtsize *nom_fichier* *taille*

evtsize **-compact** *nom_fichier* [*taille*]

evtsize **-info** *nom_fichier*

evtsize **-show** *nom_fichier*

evtsize **-info** | **-show** **pobox**

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-compact *nom_fichier* [*taille*]

Réduit la taille du fichier de message indiqué à la taille occupée par les messages présents lors de l'exécution de la commande. Facultativement, vous pouvez utiliser ce mot clé pour indiquer également une nouvelle taille de fichier.

-info *nom_de_fichier*

Affiche le pourcentage d'utilisation de la file d'attente de messages contenue dans le fichier de message.

-show *nom_fichier*

Affiche la taille de la file d'attente de messages contenue dans le fichier de messages.

nom_fichier

Nom du fichier des événements. Indiquez l'un des arguments suivants :

Courier.msg
Intercom.msg
Mailbox.msg
Planbox.msg
pobox/poste_de_travail.msg
mirrorbox.msg
mirrorbox<n>.msg

taille Taille maximale, en octets, du fichier des événements. Elle ne doit pas être inférieure à 1048576 octets (1 Mo).

Lors de la création du fichier par IBM Workload Scheduler, la taille maximale est définie sur 60 Mo.

Remarque : La taille du fichier de messages est supérieure ou égale à la taille réelle de la file d'attente de messages qu'il contient et elle augmente progressivement jusqu'à ce que la file d'attente de messages soit vide ; au moment où cela se produit, le fichier de messages est vidé.

-info | **-show pobox**

Affiche le nom du fichier de message, dans le répertoire pobox, avec la taille de file d'attente la plus grande calculée en pourcentage de la taille de fichier totale. Le nom du fichier et le pourcentage utilisé sont tous deux renvoyés. **-info** et **-show** renvoient les mêmes résultats.

Exemples

Pour définir la taille maximale du fichier Intercom.msg à 20 Mo, exécutez la commande suivante :

```
evtsize Intercom.msg 20000000
```

Pour définir la taille maximale du fichier pobox du poste de travail chicago à 15 Mo, exécutez la commande suivante :

```
evtsize pobox\chicago.msg 15000000
```

La commande suivante :

```
evtsize -show Intercom.msg
```

renvoie le résultat suivant :

```
Tivoli Workload Scheduler (UNIX)/EVTSIZE 8.3 (1.2.2.4) Eléments sous licence -  
Propriété d'IBM(R)  
5698-WSH
```

```
(C) Copyright IBM Corp 1998, 2006 All rights reserved.
```

```
US Government User Restricted Rights
```

```
Use, duplication or disclosure restricted by
```

```
GSA ADP Schedule Contract with IBM Corp.
```

```
IBM est une marque d'International Business Machines
```

```
Corporation aux Etats-Unis et/ou dans certains autres pays.
```

```
AWSDEK703I Taille de la file d'attente courant 240,  
10000000 octets maximum (48 en lecture, 288 en écriture)
```

où :

240 représente la taille de la file d'attente actuelle du fichier Intercom.msg
10000000
 représente la taille maximale du fichier Intercom.msg
48 en lecture
 Indique la position du pointeur pour lire les enregistrements
288 en écriture
 Indique la position du pointeur pour écrire les enregistrements

Si la commande suivante :

```
evtsize -info Mailbox.msg
```

renvoie :

25

cela signifie que 25 pour cent du fichier a été utilisé.

Filemonitor

Faites appel à l'utilitaire **filemonitor** pour rechercher les modifications dans les fichiers (qui ont été créés ou modifiés). Il peut vous être utile, par exemple, lorsque vous souhaitez vous assurer qu'un fichier existe avant d'exécuter un travail de traitement sur ce dernier. En définissant un travail qui exécute l'utilitaire **filemonitor**, vous pouvez implémenter une dépendance de fichier, à savoir, une relation entre un fichier et une opération dans laquelle l'activité sur le fichier détermine le démarrage de l'opération.

Vous pouvez utiliser l'utilitaire **filemonitor** en tant que commande autonome ou définir les mots clés **filemonitor** en tant que paramètres supplémentaires pour la condition de démarrage d'un flot de travaux, dans le concepteur de charge de travail ou à partir de la ligne de commande **composer**. Pour plus d'informations sur la condition de démarrage, voir Chapitre 7, «Automatisation conditionnelle de la charge de travail», à la page 135.

Exécutez l'utilitaire **filemonitor** pour surveiller si les fichiers spécifiés ont été modifiés au cours d'un intervalle de temps donné. Tous les composants du produit doivent au minimum être au niveau de version 9.4, groupe de correctifs 1.

Syntaxe

filemonitor -V | -U

```
filemonitor -path chemin_à_surveiller  

-event {fileCreated | fileModified [-modificationCompletedTime secondes]}  

[-repositoryName nom_référentiel  

[-repositoryPath chemin_référentiel  

[-recursive] [-outputFile nom_fichier_sortie  

[-scanInterval intervalle_analyse  

[-maxEventsThreshold nbre_max_événements  

[-minFileSize taille_min_fichier  

[-timeout secondes  

[-preserveEventsOnDelete]
```

```
filemonitor -reset  

[-repositoryName nom_référentiel  

[-repositoryPath chemin_référentiel - generateEventsOnFirstScan]
```


Arguments

Remarque : Si vous définissez le même argument plusieurs fois, la dernière valeur spécifiée est appliquée et aucun message d'erreur n'est signalé.

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-path *chemin_à_surveiller*

Chemin d'accès aux fichiers à traiter. Vous pouvez insérer des caractères blancs ou spéciaux entre guillemets. Les caractères génériques sont pris en charge. Pour inclure des fichiers supplémentaires dans le processus de surveillance, stockez-les dans un répertoire donné et spécifiez ce dernier dans l'option **-path**. Les chemins comportant des espaces doivent figurer entre guillemets. Les chemins UNC (convention de dénomination universelle) sont également pris en charge avec les types de syntaxe suivants :

- `\\nom_serveur\nom_partage\nom_répertoire\...`
- `\\?\UNC\nom_serveur\nom_partage\nom_répertoire`
- `\\?\nom_chemin`

où le point d'interrogation (?) indique des chemins à longueur étendue.

-event {**fileCreated** | **fileModified** [**-modificationCompletedTime** *secondes*]}

Type d'événement à surveiller. Les types pris en charge sont **fileCreated** et **fileModified**. Cet argument est obligatoire lorsque vous spécifiez **-path**. Si vous spécifiez l'argument **fileModified**, vous pouvez en option spécifier **-modificationCompletedTime** qui est l'intervalle (en secondes) depuis la dernière modification, à l'issue duquel le fichier peut être considéré comme étant dans sa version finale. Si vous ne spécifiez pas **-modificationCompletedTime**, un événement est généré dès que le fichier est modifié.

-repositoryName *nom_référentiel*

Indiquez éventuellement une base de données dans laquelle consigner le statut des fichiers extraits.. Par défaut, le nom `filemonitor.db` est utilisé si vous n'en spécifiez aucun.

-repositoryPath *chemin_référentiel*

Chemin vers la base de données **filemonitor**. Par défaut, le chemin `TWA_Home/TWS/stdlist/JM/filemonitor` est utilisé si vous n'en indiquez aucun. Les chemins comportant des espaces doivent figurer entre guillemets. Les caractères génériques ne sont pas pris en charge.

-generateEventsOnFirstScan

Tous les fichiers extraits lors de la première analyse effectuée par **filemonitor** sont considérés comme créés ou modifiés et peuvent générer des événements. Cet argument est disponible uniquement si vous indiquez l'argument **repositoryPath**.

-recursive

Les sous-dossiers sont inclus lors de la surveillance des fichiers. Paramètre facultatif.

-outputFile *nom_fichier_sortie*

Fichier de sortie dans lequel stocker les événements extraits. Vérifiez que le répertoire dans lequel le fichier de sortie doit être créé existe déjà. La sortie

de la commande est également imprimée vers la sortie standard et stockée dans les propriétés du travail si vous lancez la commande **filemonitor** à partir d'un travail. Les chemins comportant des espaces doivent figurer entre guillemets. Les caractères génériques ne sont pas pris en charge. Paramètre facultatif.

-scanInterval *intervalle_analyse*

Délai, en secondes, entre deux vérifications consécutives des fichiers en cours de création ou de modification. La valeur par défaut est 60 secondes. L'intervalle pris en charge va de 1 à 3600 secondes. Paramètre facultatif.

-maxEventsThreshold *nbre_max_événements*

Nombre maximal d'événements à renvoyer. La valeur par défaut est 1. Si vous spécifiez **all**, tous les événements seront renvoyés. Paramètre facultatif.

-minFileSize *taille_min_fichier*

Taille minimale, en octets, des fichiers à inclure dans l'analyse. La valeur par défaut est 0.

-timeout *secondes*

Temps maximal, en secondes, pendant lequel **filemonitor** attend la survenue de l'événement. Si vous ne spécifiez pas ce paramètre, **filemonitor** attend indéfiniment. Paramètre facultatif.

-preserveEventsOnDelete

Renvoie les événements sur le fichier indiqué, également si ce dernier a été supprimé entre-temps. Si vous ne spécifiez pas cet argument, lorsqu'un fichier est supprimé, tous les événements éventuels précédant cette suppression sont supprimés.

-reset Réinitialise les informations collectées. Avec cet argument, vous pouvez le cas échéant spécifier les arguments **-repositoryPath** et **-repositoryName**.

Configuration des propriétés de trace pour filemonitor

Pour configurer les propriétés de trace pour **filemonitor**, éditez la section [FileMonitor.Logging] du fichier <TWAHome>/TWS/ITA/cpa/config/FileMonitor.ini et redémarrez l'utilitaire **filemonitor**.

La section qui contient les propriétés de trace est appelée :

[FileMonitor.Logging.cclog]

FileMonitor.trhd.fileName

Nom du fichier de trace.

FileMonitor.trhd.maxFileBytes

Taille maximale que peut atteindre le fichier de trace. La valeur par défaut est de 1024000 octets.

FileMonitor.trhd.maxFiles

Nombre maximum de fichiers de trace stockés. La valeur par défaut est 3.

FileMonitor.trfl.level

Détermine le type des messages trace consignés. Modifiez cette valeur pour tracer plus ou moins d'événements, comme il convient, ou à la demande du service de support logiciel. Les valeurs valides sont les suivantes :

DEBUG_MAX

Traçage maximum. Chaque message de trace du code est écrit dans les journaux de trace.

INFO Les messages *informatifs*, *d'avertissement*, *d'erreur* et *critiques* sont écrits dans la trace. Valeur par défaut.

AVERTISSEMENT

Les messages *d'avertissement*, *d'erreur* et *critiques* sont écrits dans la trace.

ERROR

Tous les messages de trace *error* et *critical* sont écrits dans la trace.

CRITICAL

Seuls les messages qui entraînent l'arrêt de l'agent sont écrits sur la trace.

La trace de sortie (par défaut, FileMonitor_trace.log) est fournie au format XML et se trouve dans <TWA_Home>/TWS/stdlist/JM.

Codes retour

- 0 L'opération a bien été exécutée.
- 4 L'exécution de **Filemonitor** s'est arrêtée car le délai d'attente a expiré. Aucun résultat n'a été renvoyé.
- 1 Une erreur s'est produite. Pour plus de détails, recherchez le journal de trace (par défaut, FileMonitor_trace.log).

Commentaires

Les paramètres définis avec cette commande prévalent sur les valeurs par défaut définies en interne par IBM Workload Scheduler.

Si un ou plusieurs fichiers ont été créés ou modifiés pendant l'intervalle entre des appels ultérieurs, les modifications sont détectées. Cependant, les fichiers déjà détectés lors d'une exécution précédente ne sont pas de nouveau listés dans les appels suivants. Les caractères génériques sont admis dans les noms de fichier et de répertoire.

Exemples

Dans l'exemple suivant, la commande **filemonitor** recherche toutes les 2 minutes les fichiers créés dans le chemin C:\temp\logs et dont la taille minimale est supérieure à 1024 octets. La recherche s'effectue dans tous les sous-dossiers et les résultats sont stockés dans C:\backup\logs\reports.txt :

```
filemonitor -path "C:\temp\logs" -event fileCreated -recursive  
-outputFile "C:\backup\logs\reports.txt"  
-scanInterval 120 -maxEventsThreshold all -minFileSize 1024
```

jobinfo

Utilisée dans un script de travail pour renvoyer des informations relatives au travail. Cette commande n'est pas prise en charge sur les agents dynamiques, les pools, les pools dynamiques et les types de travaux avec options avancées.

Syntaxe

jobinfo -V | -U

jobinfo option-travail [...]

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.

option-travail

Option de travail. Indiquez un ou plusieurs des éléments suivants :

confirm_job

Renvoie **YES** si le travail doit être confirmé.

is_command

Renvoie **YES** si le travail a été planifié ou soumis à l'aide de la structure **docommand**.

nom_travail

Renvoie le nom du travail sans le nom du poste de travail et le nom du flot de travaux.

job_pri

Renvoie le niveau de priorité du travail.

programmatic_job

Renvoie **YES** si le travail a été soumis à l'aide de la commande **at** ou **batch**. Sous UNIX uniquement.

re_job Renvoie **YES** si le travail est de nouveau exécuté suite à l'exécution d'une commande **conman rerun** ou de l'option de reprise **rerun**.

re_type

Renvoie l'option de reprise du travail (**stop**, **continue** ou **rerun**).

rstt_flag

Renvoie **YES** si le travail est exécuté en tant que travail de reprise.

rstt_retcode

Si le travail en cours est un travail de reprise, renvoie le code retour du travail parent.

schedule

Renvoie le nom du flot de travaux dans lequel le travail s'exécute.

schedule_ia

Renvoie l'heure et la date où le lancement du flot de travaux est planifié.

schedule_id

Renvoie l'ID *ID_flot_travaux* du flot de travaux dans lequel le travail est soumis.

time_started

Renvoie l'heure à laquelle l'exécution du travail a commencé.

Commentaires

Les valeurs d'option de travail sont renvoyées, à raison d'une par ligne, dans l'ordre dans lequel elles ont été demandées.

Exemples

1. Le fichier script `/jcl/backup` est référencé à deux reprises et les noms de travaux **partback** et **fullback** lui sont attribués. Si le travail est exécuté sous la

forme **partback**, il effectue une sauvegarde partielle. S'il est exécuté sous la forme **fullback**, il effectue une sauvegarde complète. Cette distinction est déterminée dans le script à l'aide de commandes similaires aux commandes suivantes :

```
#Choix entre partiel (1) ou intégral (2) :
if [ "`\maestro\bin\jobinfo job_name`" = "PARTBACK" ]
then
bkup=1
else
bkup=2
fi
...
```

2. Pour afficher le code retour du travail parent, exécutez la commande suivante si le travail courant est un travail de reprise :

```
$ jobinfo rstrt_retcode
```

Le premier travail (travail parent) a été défini dans le script `recovery.sh` alors que le second travail (travail de reprise) est activé uniquement si le premier se termine de manière anormale.

Si vous associez un code retour à la commande **jobinfo rstrt_retcode**, vous pouvez diriger le travail de reprise pour réaliser différentes actions selon le code retour du travail parent. Dans l'exemple suivant, nous avons illustré un travail de reprise :

```
$JOBS
MASTER#DBSELOAD DOCOMMAND "/usr/local/tws/maestro/scripts/populate.sh"
STREAMLOGON "^TWSUSER^"
DESCRIPTION "alimentation manuelle de la base de données"
RECOVERY RERUN AFTER MASTER#RECOVERY
RCONDSUCC "(RC = 0) OR ((RC > 4) AND (RC < 11))"
```

Remarque : Le travail est défini par l'action de reprise RERUN qui permet au travail de reprise d'intervenir avant que le programme ne tente d'exécuter une nouvelle fois le travail parent.

Dans l'exemple suivant, le travail de reprise proprement dit est défini :

```
$ JOBS
MASTER#RECOVERY DOCOMMAND "^TWSHOME^/scripts/recovery.sh"
STREAMLOGON "^TWSUSER^"
DESCRIPTION "reprise manuelle de la base de données"
RECOVERY STOP
```

jobstdl

Renvoie les noms des listes standard. Cette commande doit être exécutée par l'utilisateur pour lequel IBM Workload Scheduler a été installé. Si vous utilisez la commande sans paramètres, vous devez être connecté en tant qu'utilisateur IBM Workload Scheduler.

Syntaxe

```
jobstdl -V | -U
```

```
jobstdl
```

```
[-day num]
[{-first | -last | -num n | -all}]
[-twslog]
[{-name ["nom_flot_travaux [(hhmm date),(ID_flot_travaux)]."]nom_travail"
 | num_travail | -schedid ID_flot_travaux.nom_travail}]
```

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- day *num***
Renvoie les noms de liste standard qui datent du nombre de jours indiqué (1 pour hier, 2 pour avant-hier, etc.). Valeur par défaut : zéro (aujourd'hui).
- first** Renvoie le nom de la première liste standard éligible.
- last** Renvoie le nom de la dernière liste standard éligible.
- num *n***
Renvoie le nom de la liste standard correspondant à l'exécution indiquée d'un travail.
- all** Renvoie le nom de toutes les listes standard éligibles.
- tw slog**
Renvoie le chemin du fichier *stdlist* du jour courant.
- name ["*nom_flot_travaux*[(*hhmm date*), (*id_flot_travaux*)].]*nom_travail*" | *num_travail***
Indique l'instance du flot de travaux et le nom du travail pour lesquels des noms de fichier de liste standard sont renvoyés.

num_travail
Indique le numéro du travail pour lequel des noms de liste standard sont renvoyés.
- schedid *ID_flot_travaux.nom_travail***
Indique l'ID de flot de travaux et le nom du travail pour lequel des noms de fichier de liste standard sont retournés.

Commentaires

Les noms de fichier sont renvoyés dans un format approprié pour être entrés dans d'autres commandes. Si le programme renvoie plusieurs noms, il les sépare par un espace.

Lorsque vous utilisez la syntaxe complète de l'argument **-name**, les crochets dans l'expression [(*hhmm date*), (*ID_flot_travaux*)] font partie de la commande et ne sont pas des indicateurs de syntaxe. De plus, toute la chaîne d'identification de travail doit être placée entre guillemets si la partie désignant l'instance de flot de travaux contient des blancs. Par exemple, parce que *schedtime*, représenté par *hhmm date*, contient un blanc, vous devez placer toute l'identification de travail entre guillemets.

Vous pouvez également exécuter des versions abrégées de l'argument **-name** à l'aide d'une syntaxe plus simple. Si vous souhaitez obtenir moins de sorties spécifiques de la commande, vous pouvez indiquer uniquement *schedtime* (la *date* n'est pas requise si c'est pour le même jour) ou l'*ID_flot_travaux* conjointement avec le *nom_travail*. Tant qu'il n'y a pas d'espaces dans les arguments, vous pouvez omettre les guillemets. Vous pouvez également omettre les crochets droits si vous n'indiquez pas à la fois les *schedtime* et *ID_flot_travaux*.

Les exemples suivants présentent la syntaxe que vous devez utiliser avec le même argument **-name** pour les différents types d'informations que vous attendez en retour, allant du plus spécifique au plus général. Dans l'exemple, *job_stream1* est

le nom du flot de travaux, 0600 04/05/06 est le temps planifié, 0AAAAAAAAAAAAAB5 est l'ID de flot de travaux, et job1 est le nom du travail. Le numéro de travail de job1 est 310. Vous pouvez exécuter **jobstdl** pour job1 comme suit :

```
jobstdl -name "job_stream1[(0600 04/05/10),(0AAAAAAAAAAAAAB5)].job1"
```

Renvoie le nom de fichier de liste standard de job1 pour l'instance spécifique de job_stream1 avec les *schedtime* et *ID_flot_travaux* spécifiés.

```
jobstdl -name job_stream1(0AAAAAAAAAAAAAB5).job1
```

Renvoie le nom de fichier de liste standard de job1 pour l'instance de job_stream1 avec l'ID 0AAAAAAAAAAAAAB5.

```
jobstdl -name "job_stream1(0600 04/05/10).job1"
```

Renvoie les noms de fichier de liste standard de job1 pour toutes les instances possibles de job_stream1 planifiées pour s'exécuter à 0600 le 04/05/10.

```
jobstdl -name job_stream1(0600).job1
```

Renvoie les noms de fichier de liste standard de job1 pour toutes les instances possibles de job_stream1 planifiées pour s'exécuter à 0600 le jour courant.

```
jobstdl -name 310
```

Renvoie les noms de fichier de liste standard de job1 pour toutes les instances pour lesquelles il avait le numéro de travail 310.

Exemples

Pour renvoyer les noms de chemin de tous les fichiers de liste standard du jour courant, exécutez la commande suivante :

```
jobstdl
```

Pour renvoyer le nom de chemin de la liste standard pour la première exécution du travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR du jour courant, exécutez la commande suivante :

```
jobstdl -first -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour retourner le chemin d'accès de la liste standard pour la première exécution du travail 0AAAAAAAAAAAAAEE.DIR dans la journée, exécutez la commande suivante :

```
jobstdl -first -schedid 0AAAAAAAAAAAAAEE.DIR
```

Pour renvoyer le nom de chemin de la liste standard pour la deuxième exécution du travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR du jour courant, exécutez la commande suivante :

```
jobstdl -num 2 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour renvoyer les noms de chemin des fichiers de la liste standard pour toutes les exécutions du travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR remontant à trois jours, exécutez la commande suivante :

```
jobstdl -day 3 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour renvoyer le nom de chemin de la liste standard pour la dernière exécution du travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR remontant à quatre jours, exécutez la commande suivante :

```
jobstdl -day 4 -last -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour renvoyer le nom de chemin de la liste standard du travail **455**, exécutez la commande suivante :

```
jobstdl 455
```

Pour imprimer le contenu du fichier de liste standard du travail **455**, exécutez la commande suivante :

```
cd `maestro`/bin  
lp -p 6 `jobstdl 455`
```

maestro

Renvoie le nom de chemin du répertoire principal IBM Workload Scheduler, désigné par *racine_TWS*.

Syntaxe

maestro [-V | -U]

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.

Exemples

Pour afficher le répertoire principal IBM Workload Scheduler, exécutez la commande suivante :

```
$ maestro  
/usr/lib/maestro
```

Pour modifier le répertoire principal IBM Workload Scheduler, exécutez la commande suivante :

```
$ cd `maestro`
```

makecal

Crée un agenda personnalisé. Sous UNIX, vous devez utiliser le shell Korn pour exécuter la commande.

Syntaxe

makecal [-V | -U]

makecal

```
[-c name]  
-d n  
| -e  
| {-f 1 | 2 | 3 -s date}  
| -l  
| -m  
| -p n  
| {-r n -s date}  
| -w n  
[-i n]  
[-x | -z]  
[-freedays nom_agenda [-sa] [-su]]
```


Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.
- c *nom* Indique un nom pour l'agenda. Les mots clés IBM Workload Scheduler (tels que *Freedays* ou *Schedule*) ne peuvent pas être utilisés comme noms d'agenda. Ce nom peut comporter jusqu'à huit caractères alphanumériques et doit commencer par une lettre. N'utilisez pas les noms des jours de la semaine pour nommer un agenda. Le nom par défaut est le suivant : *Chhmm*, où *hhmm* est l'heure courante exprimée en heures et en minutes.
- d *n* Indique le *énième* jour de chaque mois.
- e Indique le dernier jour de chaque mois.
- f 1 | 2 | 3
Crée un agenda de fin de mois fiscal contenant le dernier jour du mois fiscal. Indiquez un des formats suivants :
 - 1 Format de semaine 4-4-5
 - 2 Format de semaine 4-5-4
 - 3 Format de semaine 5-4-4Cet argument requiert l'argument -s.
- i *n* Indique que *n* dates doivent être placées dans l'agenda.
- l Indique le dernier jour ouvré de chaque mois. Cet argument ne fonctionne correctement que si le plan de production (fichier Symphony) et l'agenda **holidays** existent déjà.

Remarque : L'utilisation de cet argument a pour résultat d'inclure également dans le nouvel agenda le dernier jour ouvré du mois précédant la date de création de l'agenda.
- m Indique le premier et le quinzième jour de chaque mois.
- p *n* Indique le jour ouvré précédant le *énième* jour de chaque mois. Cet argument ne fonctionne correctement que si le plan de production (fichier Symphony) et l'agenda **holidays** existent déjà.
- r *n* Indique chaque *énième* jour. Cet argument requiert l'argument -s.
- s *date* Indique la date de début pour les arguments -f et -r. Cette date doit être valide et non ambiguë et placée entre apostrophes ; par exemple, utilisez **JAN 10 2005** et non **1/10/05**. Pour plus d'informations sur les formats de date, voir *base-date* pour **datecalc** à la page «base-date», à la page 685.
- w *n* Indique le jour ouvré suivant le *énième* jour du mois. Cet argument ne fonctionne correctement que si le plan de production (fichier Symphony) et l'agenda **holidays** existent déjà.
- x Envoie la sortie de l'agenda à **stdout** au lieu de l'ajouter dans la base de données.
- z Ajoute l'agenda dans la base de données et compile le plan de production (fichier Symphony).

Remarque : Cet argument soumet à nouveau des travaux et des flots de travaux à partir du plan de production du jour. Il est peut-être nécessaire d'annuler des flots de travaux et des travaux.

-freedays

Indique le nom d'un agenda de jours chômés *nom_agenda* devant remplacer les **holidays** dans l'évaluation des *jours ouvrés*.

Dans le cas présent, les *jours ouvrés* sont considérés comme étant *tous les jours de la semaine* sauf *samedi*, *dimanche* et toutes les dates répertoriées dans *Nom_agenda*.

Par défaut, le *samedi* et le *dimanche* ne sont pas considérés comme des *jours ouvrés*, sauf si vous indiquez explicitement le contraire en ajoutant **-sa** et/ou **-su** après *nom_agenda*.

Vous pouvez également affecter le nom **holidays** à l'agenda des jours chômés.

Ce mot clé affecte le traitement de **makecal** avec les options **-l**, **-p** et **-w**.

Exemples

Pour créer un agenda de deux ans où le dernier jour de chaque mois est sélectionné, exécutez la commande suivante :

```
makecal -e -i 24
```

Pour créer un agenda de 30 jours débutant le 30 mai 2005 et où un jour sur trois est sélectionné, exécutez la commande suivante :

```
makecal -r 3 -s "30 MAY 2005" -i 30
```

metronome

La commande metronome est remplacée par **twins_inst_pull_info**. Pour obtenir des informations sur cette commande, consultez le document *IBM Workload Scheduler - Guide d'identification et résolution des problèmes*.

morestdl

Affiche le contenu des listes standard. Cette commande doit être exécutée par l'utilisateur pour lequel IBM Workload Scheduler a été installé. Si vous utilisez la commande sans paramètres, vous devez être connecté en tant qu'utilisateur IBM Workload Scheduler. Cette commande est prise en charge pour les agents tolérants aux pannes et les agents standard.

Syntaxe

morestdl -V | -U

morestdl

[-day num]

[-first | -last | -num n | -all]

[-twolog]

[{-name ["nom_flot_travaux [(hhmm date),(ID_flot_travaux)].]nom_travail"
| num_travail | -schedid ID_flot_travaux.nom_travail}]

Arguments

-V Affiche la version de la commande et quitte l'application.

- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- day *num***
Affiche les listes standard qui datent du nombre de jours indiqué (1 pour hier, 2 pour avant-hier, etc.). Valeur par défaut : zéro (aujourd'hui).
- first** Affiche la première liste standard éligible.
- last** Affiche la dernière liste standard éligible.
- num *n***
Affiche la liste standard correspondant à l'exécution indiquée d'un travail.
- all** Affiche toutes les listes standard éligibles.
- twslog**
Affiche le contenu du fichier *stdlist* du jour courant.
- name ["*nom_flot_travaux* [(*hhmm date*),(*ID_flot_travaux*)].*nom_travail*" | *num_travail***
Indique l'instance du flot de travaux et le nom du travail pour lesquels le fichier de la liste standard est affiché.
- num_travail***
Indique le numéro du travail pour lequel la liste standard est affichée.
- schedid *ID_flot_travaux.nom_travail***
Indique l'ID de flot de travaux et le nom du travail pour lequel des noms de fichier de liste standard sont retournés.

Commentaires

Les crochets droits dans l'expression [(*hhmm date*), (*ID_flot_travaux*)] font partie de la commande et ne sont pas des indicateurs de syntaxe. Vous pouvez donc définir l'un ou l'autre des éléments suivants pour l'argument **-name** :

```
morestdl -name ["nom_flot_travaux[(hhmm date),(ID_flot_travaux)].nom_travail"
morestdl -name num_travail
```

Toute la chaîne d'identification de travail doit être placée entre guillemets si la partie désignant l'instance de flot de travaux contient des blancs. Par exemple, parce que *schedtime*, représenté par *hhmm date*, contient un blanc, vous devez placer toute l'identification de travail entre guillemets.

Si vous voulez simplement identifier un nom de travail, vous n'avez pas besoin de le placer entre guillemets.

Voici un exemple de la syntaxe à utiliser lors de l'identification d'un travail avec et sans son flot de travaux. Dans l'exemple, *job_stream1* est le nom du flot de travaux, *0600 04/05/06* est le temps planifié, *0AAAAAAAAAAAAAB5* est l'ID de flot de travaux, et *job1* est le nom du travail. Vous pouvez exécuter la commande **morestdl** sur *job1* en utilisant l'un des deux formats ci-dessous :

```
morestdl -name "job_stream1[(0600 04/05/06),(0AAAAAAAAAAAAAB5)].job1"
morestdl -name job1
```

Exemples

Pour afficher le fichier de liste standard pour la première exécution du travail *MY_CPU#ELI*[(1824 03/09/06), (0AAAAAAAAAAAAAEE)].DIR du jour courant, exécutez la commande suivante :

```
morestdl -first -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour afficher le fichier de liste standard pour la première exécution du travail 0AAAAAAAAAAAAAEE.DIR du jour courant, exécutez la commande suivante :

```
morestdl -first -schedid 0AAAAAAAAAAAAAEE.DIR
```

Pour afficher le fichier de liste standard pour la deuxième exécution du travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR du jour courant, exécutez la commande suivante :

```
morestdl -num 2 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour afficher les fichiers de liste standard pour toutes les exécutions de travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR remontant à trois jours, exécutez la commande suivante :

```
morestdl -day 3 -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour afficher le fichier de liste standard pour la dernière exécution de travail MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR remontant à quatre jours, exécutez la commande suivante :

```
morestdl -day 4 -last -name "MY_CPU#ELI[(1824 03/09/06),(0AAAAAAAAAAAAAEE)].DIR"
```

Pour imprimer le fichier de liste standard du travail 455, exécutez la commande suivante :

```
morestdl 455 | lp -p 6
```

parms

Exécutez l'utilitaire parms sur un poste de travail de gestionnaire de domaine maître ou d'agent tolérant aux pannes (non pris en charge sur les agents dynamiques) afin de gérer les paramètres définis en local sur les postes de travail. Les paramètres gérés par **parms** peuvent uniquement être utilisés dans une définition de travail ou de flot de travaux avec les mots clés **scriptname** ou **opens** ou dans un fichier de script de travail.

Ces paramètres sont résolus lors de la soumission sur le poste de travail sur lequel le travail ou le flot de travaux est soumis. S'il n'existe aucune correspondance entre le nom *nom_paramètre* défini et le nom des paramètres définis dans la base de données locale sur le poste de travail, une valeur *null* est renvoyée.

Autorisation

Vous devez avoir un accès *display* à la base de données des paramètres définis au niveau local. De plus, vous devez disposer du droit d'accès suivant :

build sur un fichier objet

Si vous utilisez l'option **-b** pour créer ou reconstruire la base de données des paramètres locaux.

delete Si vous utilisez l'option **-d** pour supprimer les définitions de paramètres locaux.

modify sur un fichier objet

Si vous utilisez l'option **-replace** pour ajouter ou modifier des définitions de paramètres.

Syntaxe

parms {[-V | -u] | -build}

parms {-replace | -extract} *nom_fichier*

parms [-d]*nom_paramètre*

parms -c *valeur_nom_paramètre*

Arguments

-V Affiche la version de la commande et quitte l'application.

-u Affiche des informations sur la syntaxe de la commande et quitte l'application.

-build Crée la base de données des paramètres sur le poste de travail si elle n'existe pas. Reconstitue la base de données de paramètres si elle existe déjà en supprimant les enregistrements inutilisés et en évitant la fragmentation provenant de nombreuses additions et suppressions.

-extract

Extrait toutes les définitions de paramètres de la base de données locale et les stocke dans le fichier avec le nom *nom_fichier*.

Utilisez cette option pour exporter les définitions de paramètres locaux afin de les importer en tant que définitions de paramètres globaux dans la base de données des objets de planification à l'aide des commandes «add», à la page 338 ou «replace», à la page 381.

-replace

Ajoute dans la base de données locale de nouvelles définitions de paramètres stockés dans un fichier nommé *nom_fichier* ou remplace ceux déjà existants.

Utilisez cette option pour importer en tant que définition de paramètres locaux, les définitions de paramètres globaux contenues dans le fichier nommé *nom_fichier* et extraites de la base de données des objets de planification à l'aide de la commande «extract», à la page 352.

-d Supprime les paramètres avec le nom *parametername* de la base de données locale sur le poste de travail.

nom_paramètre

Indique le nom du paramètre dont la valeur est affichée. Indique le nom du paramètre dont la valeur est affichée. Lorsqu'il est utilisé avec l'argument **-d** il représente le nom du paramètre à supprimer.

-c *nom valeur*

Indique le nom et la valeur d'un paramètre. Ce nom peut comporter jusqu'à 16 caractères alphanumériques, incluant les tirets (-) et les traits de soulignement (_), et doit commencer par une lettre. La valeur peut comporter jusqu'à 72 caractères. La valeur doit être délimitée par des guillemets si elle contient des caractères spéciaux. Si le paramètre n'existe pas, il est ajouté à la base de données. S'il existe déjà, sa valeur est modifiée.

Commentaires

Lorsque vous exécutez la commande **parms** sur la ligne de commande sans aucun argument, vous êtes invité à entrer des noms et valeurs de paramètres.

L'utilisation de **parms** dans les définitions de travail et les fichiers de script de travail nécessite que le paramètre existe déjà au niveau local dans la base de données des paramètres sur le poste de travail.

Voici un exemple d'un paramètre local, MYFILE, dans une clause de dépendance de fichier :

```
schedule test_js
on everyday
opens "/usr/home/tws_99/'/usr/home/tws_99/bin/parms MYFILE'"
:
test_job
end
```

L'exemple suivant décrit la façon dont la variable *var*, entourée par des accents circonflexes (^), est remplacée lorsque le travail est en cours. Si le travail est soumis comme travail ad hoc, le paramètre *var* est étendu, à savoir remplacé par la valeur attribuée à *var* dans la base de données locale lors de la soumission, et non lors du lancement du travail.

Exemple de définition de travail UNIX :

```
DATA#UX_P_TEST DOCOMMAND "ls ^var^"
STREAMLOGON "mae82"
DESCRIPTION "Test parms in job definition on UNIX."
RECOVERY STOP
```

Exemple de définition de travail Windows :

```
BORG#WIN_P_TEST DOCOMMAND "dir ^var^"
STREAMLOGON "mae82"
DESCRIPTION "Test parms in job definition on Windows."
RECOVERY STOP
```

Lorsque le paramètre est utilisé dans un fichier de script de travail, il n'est pas étendu avant le lancement du script. Il n'est pas étendu lorsque le flot de travaux contenant le travail est traité par **JnextPlan**. Vous trouverez ci-dessous des exemples sur l'utilisation du paramètre *var* dans les fichiers de script de travail.

Exemple de script UNIX :

```
#!/bin/sh
TWS_HOME="/opt./tws/mae82/maestro"
export TWS_HOME
MDIR='$TWS_HOME/bin/parms var'
export MDIR
ls -l $MDIR
```

Exemple de script Windows :

```
set TWS_HOME=d:\win32app\TWS\mae82\maestro
echo %TWS_HOME%
FOR /F "Tokens=*" %a in (%TWS_HOME%\bin\parms var) do set MDIR=%a
echo %MDIR%
dir %MDIR%
```

Exemples

Pour renvoyer la valeur monparm, exécutez la commande suivante :

```
parms monparm
```

Pour modifier la valeur monparm, exécutez la commande suivante :

```
parms -c monparm "item 123"
```

Pour créer un nouveau paramètre appelé sonparm, exécutez la commande suivante :

```
parms -c sonparm  
"item 789"
```

Pour modifier la valeur monparm et ajouter le paramètre sonparm, exécutez la commande suivante :

```
parms  
Name of parameter ? myparm < Return>  
Value of parameter? "item 456" < Return>  
Name of parameter ? herparm < Return>  
Value of parameter? "item 123" < Return>  
Name of parameter ? < Return>
```

Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

release

Exécutez l'utilitaire release sur un poste de travail de gestionnaire de domaine maître ou d'agent tolérant aux pannes (non pris en charge sur les agents dynamiques) afin de libérer les travaux et les flots de travaux des dépendances **needs** sur une ressource. Cette commande doit être émise uniquement à partir d'un fichier de script de travail. Cette commande n'est efficace que si les travaux et les flots de travaux cibles sont en cours d'exécution lorsque la commande est émise. Si vous réexécutez le travail ou le flot de travaux, les ressources requises sont réallouées. Tous les composants du produit doivent au minimum être au niveau de version 9.4.

Syntaxe

release -V | -U

release

```
[-s | -schedule]  
[poste_de_travail#]  
nom_ressource  
[comptage]
```

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-s | -schedule

Libère la dépendance **needs** à partir de la ressource spécifiée uniquement au niveau du flot de travaux.

Si **-s** n'est pas utilisé, la dépendance **needs** de la ressource spécifiée est libérée au niveau du travail ou au niveau du flot de travaux **needs** est introuvable au niveau du travail.

poste_de_travail#

Indique le nom du poste de travail ou de la classe de postes de travail sur laquelle la ressource est définie. Par défaut, il s'agit du poste de travail local.

nom_ressource

Indique le nom de la ressource impliquée dans la dépendance **needs**.

nombre Indique le nombre d'unités de la ressource à libérer. Si aucun nombre n'est spécifié, toutes les ressources sont libérées.

Commentaires

Les unités d'une ressource sont acquises par un travail ou un flot de travaux lorsque celui-ci est lancé, et libérées automatiquement à la fin du travail ou du flot de travaux. La commande **release** peut être utilisée dans un script de travail pour libérer des ressources avant la fin du travail ou du flot de travaux ou pour libérer manuellement des travaux et des flots de travaux à partir des dépendances **needs** dans des situations d'urgence.

Exemples

Dans le flot de travaux suivant, deux unités de la ressource **dbase** sont requises par le flot de travaux **sked5** :

```
schedule ux1#sked5 on tu
nécessite 2 dbase :
job1
jobrel follows job1
job2 follows jobrel
end
```

Pour libérer la ressource **dbase** avant le début du travail **job2**, le fichier script de **jobrel** contient la commande suivante :

Sur les systèmes d'exploitation UNIX :

```
`maestro`/bin/release -s dbase
```

Sur les systèmes d'exploitation Windows :

```
<rép_base_TWS>\bin\release -s dbase
```

Remarque : L'argument **-s** peut être omis, dans la mesure où aucune ressource n'a été bloquée au niveau du travail.

L'exemple suivant explique comment libérer partiellement des ressources au niveau du flot de travaux.

Dans le flot de travaux suivant, quatre unités de la ressource **dbase** sont requises par le flot de travaux **sked5** :

```
schedule ux1#sked5 on tu
needs 4 dbase :
job1
jobrel follows job1
job2 follows jobrel
end
```


Pour libérer la ressource dbase avant le début du travail job2, le fichier script de jobre1 contient la commande suivante :

Sur les systèmes d'exploitation UNIX :

```
`maestro`/bin/release -s dbase 3
```

Sur les systèmes d'exploitation Windows :

```
<rép_base_TWS>\bin\release -s dbase 3
```

Dans ce cas, pendant l'exécution du travail job1, le nombre de ressources requises par le flot de travaux ux1#sked5 est 4. Lorsque le travail jobre1 démarre et que la commande **release** est lancée, le nombre de ressources utilisées passe à 1, car la commande **release** a libéré 3 ressources.

Voici la sortie de la commande **conman sr @##** lancée alors que job1 est en cours d'exécution :

```
%sr @##  
Num UC Ressource    Total disponible Qté Util par  
UX1#DBASE           10    6           4    UX1#SKED5[(1032 11/03/16),(0AAAAAAAAAAAAABM)]
```

Voici la sortie de la commande **conman sr @##** une fois que le deuxième travail (jobre1) du flot de travaux est terminé et avant que le dernier travail (job2) du flot de travaux soit terminé :

```
%sr @##  
Num UC Ressource    Total disponible Qté Util par  
UX1#DBASE           10    9           1    UX1#SKED5[(1032 11/03/16),(0AAAAAAAAAAAAABM)]
```

L'exemple suivant explique comment libérer complètement des ressources au niveau du flot de travaux.

Dans le flot de travaux suivant, quatre unités de la ressource dbase sont requises par le flot de travaux sked5 :

```
schedule ux1#sked5 on tu  
needs 4 ux1#dbase :  
job1  
jobre1 follows job1  
job2 follows jobre1  
end
```

Pour libérer la ressource dbase avant le début du travail job2, le fichier script de jobre1 contient la commande suivante :

Sur les systèmes d'exploitation UNIX :

```
`maestro`/bin/release -s dbase 4
```

Sur les systèmes d'exploitation Windows :

```
<rép_base_TWS>\bin\release -s dbase 4
```

Dans ce cas, pendant l'exécution du travail job1, le nombre de ressources requises par le flot de travaux ux1#sked5 est 4. Lorsque le travail jobre1 commence à lancer la commande **release**, le nombre de ressources utilisées passe à 0, car la commande **release** a libéré les 4 ressources. Vous pouvez obtenir le même résultat en spécifiant un nombre de ressources plus élevé que le nombre de ressources réellement requises par le flot de travaux, ou en ne spécifiant aucun nombre : dans les deux cas, la commande libère toutes les ressources requises par le flot de travaux.

Voici la sortie de la commande **conman sr @##** lancée alors que job1 est en cours d'exécution :

```
|
| %sr @#@
| Num UC Ressource      Total disponible  Qté Util par
| UX1#DBASE             10      6           4    UX1#SKED5[(1040 11/03/16),(0AAAAAAAAAAAAABM)]
```

| Voici la sortie de la commande **conman sr @#@** une fois que le deuxième travail (jobrel) du flot de travaux est terminé et avant que le dernier travail (job2) du flot de travaux soit terminé :

```
|
| %sr @#@
| Num UC Ressource      Total disponible  Qté Util par
| UX1#DBASE             10      10          Pas de réservation pour cette ressource
```

| L'exemple suivant explique comment libérer partiellement des ressources au niveau du travail.

| Dans le flot de travaux suivant, quatre unités de la ressource dbase sont requises par le travail jobrel :

```
|
| schedule ux1#sked5 on tu
| :
| job1
| jobrel follows job1
| needs 4 fta1#dbase
| job2 follows jobrel
| end
```

| Pour libérer la ressource dbase avant le début du travail job2, le fichier script de jobrel contient la commande suivante :

| **Sur les systèmes d'exploitation UNIX :**

```
| `maestro~/bin/release dbase 1
```

| **Sur les systèmes d'exploitation Windows :**

```
| <rép_base_TWS>\bin\release dbase 1
```

| Dans ce cas, pendant l'exécution du travail job1, le nombre de ressources requises est 0. Dès que le travail jobrel démarre et avant que la commande **release** qu'il contient soit lancée, le nombre de ressources utilisées passe à 4. Lorsque la commande **release** du travail jobrel est lancée, le nombre de ressources utilisées passe à 3, car la commande **release** a libéré 1 ressource.

| L'exemple suivant explique comment libérer partiellement des ressources au niveau du flot de travaux.

| Dans le flot de travaux suivant, 34 unités de la ressource dbase sont requises par le travail jobrel :

```
|
| schedule ux1#sked5 on tu
| needs 34 dbase
| :
| job1
| jobrel follows job1
| job2 follows jobrel
| end
```

| Pour libérer la ressource dbase avant le début du travail job2, le fichier script de jobrel contient la commande suivante :

| **Sur les systèmes d'exploitation UNIX :**

```
| `maestro~/bin/release dbase
```

| **Sur les systèmes d'exploitation Windows :**

```
| <rép_base_TWS>\bin\release dbase
```

Dans ce cas, pendant l'exécution du travail job1, le nombre de ressources requises par le flot de travaux ux1#sked5 est 34. Lorsque le travail jobre1 démarre, le nombre de ressources utilisées passe à deux. Cela se produit car le produit divise les ressources en blocs constitués de 32 unités. La dépendance de 34 ressources est évaluée par Workload Scheduler en tant que dépendance double : la première dépendance comportant 32 unités, et la deuxième deux unités. Lorsque la commande **release** du travail jobre1 est lancée, le nombre de ressources utilisées passe à 2 car la commande **release** (pour laquelle aucune quantité n'a été définie) a libéré la première dépendance contenant 32 unités.

Voici la sortie de la commande **conman sr @##** lancée alors que job1 est en cours d'exécution :

```
%sr @##
Num UC Ressource    Total disponible Qté Util par
UX1#DBASE           34    0           34  UX1#SKED5[(1101 11/03/16),(0AAAAAAAAAAAAABR)]
```

Voici la sortie de la commande **conman sr @##** une fois que le deuxième travail (jobre1) du flot de travaux est terminé et avant que le dernier travail (job2) du flot de travaux soit terminé :

```
%sr @##
Num UC Ressource    Total disponible Qté Util par
UX1#DBASE           34   32           2  UX1#SKED5[(1101 11/03/16),(0AAAAAAAAAAAAABR)]
```

L'exemple suivant illustre le fonctionnement interne du produit et la raison pour laquelle aucune ressource n'est libérée dans ce cas.

Dans le flot de travaux suivant, quatre unités de la ressource dbase sont requises par le travail job1 :

```
schedule ux1#sked5 on tu
:
job1
needs 4 dbase
jobre1
job2 follows jobre1
end
```

Pour libérer la ressource dbase avant le début du travail job2, le fichier script de jobre1 contient la commande suivante :

Sur les systèmes d'exploitation UNIX :

```
`maestro`/bin/release dbase 2
```

Sur les systèmes d'exploitation Windows :

```
<rép_base_TWS>\bin\release dbase 2
```

Dans ce cas, le travail job1 nécessite 4 ressources. Lorsque le travail jobre1 démarre, la commande release qu'il contient n'a aucun effet car aucune dépendance de ressource n'est présente pour le travail jobre1. Cela se produit car la commande **release** ne libère des ressources que pour l'instance de travail qui exécute la commande. Si d'autres travaux ou flots de travaux, ou d'autres instances du même travail qui lance la commande **release**, utilisent les unités d'une ressource spécifique, ces unités ne sont pas libérées, même lorsque la ressource utilisée correspond au nom de ressource dans la commande.

rmstdlist

Supprime ou affiche des listes standard en fonction de l'ancienneté du fichier. Cette fonctionnalité doit être utilisée par l'administrateur IBM Workload Scheduler pour gérer l'environnement de planification.

Syntaxe

rmstdlist -V | -U

rmstdlist [-p] [jours]

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- p** Affiche les noms des répertoires des listes standard éligibles. Aucun répertoire ou fichier n'est supprimé. Si **-p** n'est pas indiqué, les listes standard éligibles sont supprimées.
- jours* Age minimum, en jours, des répertoires de listes standard à afficher ou à supprimer. La valeur par défaut est 10 jours.

Remarque : Du fait que la liste des répertoires et des fichiers affichés ou supprimés à l'aide de **rmstdlist** est générée en fonction de leur date de dernier accès, les dates indiquées dans la liste des répertoires peuvent être différentes des dates affichées dans la liste des fichiers.

Syntaxe

À titre de règle, vous devez régulièrement supprimer les fichiers de la liste standard, tous les 10 à 20 jours environ. Des retards plus importants peuvent être plus difficiles à gérer et si le nombre de fichiers devient trop important vous pouvez avoir à effacer certains d'entre eux manuellement avant de pouvoir utiliser à nouveau **rmstdlist**.

Ce problème peut se produire sur les systèmes AIX particulièrement à cause d'une limitation actuellement non résolue avec la commande **rm -rf**. Si **rmstdlist** échoue à cause de cette limitation, la commande n'affiche pas d'erreur autre que le code de sortie 126. Si vous souhaitez que l'erreur **rm-rf** soit affichée, vous pouvez modifier le script **rmstdlist** de la façon suivante :

1. Localisez le script dans le répertoire *TWS_home/bin*
2. Trouvez la ligne :

```
rm -rf `cat /tmp/rm$$` 2> /dev/null
```
3. Supprimez la redirection */dev/null* de façon à ce que la ligne devienne :

```
rm -rf `cat /tmp/rm$$`
```

Exemples

Pour afficher le nom des répertoires de fichiers de liste standard qui ont plus de 14 jours, exécutez la commande suivante :

```
rmstdlist -p 14
```

Pour supprimer tous les fichiers de liste standard (et leurs répertoires) qui ont plus de sept jours, exécutez la commande suivante :

```
rmstdlist 7
```

sendevent

La commande envoie un les événements personnalisés définis par la commande `evtdef` vers le serveur de processeur d'événements actuellement actif dans le plan de production. Lors de la réception des événements par le processeur d'événements, ils déclenchent les règles d'événement dans lesquelles ils ont été spécifiés.

Les utilisateurs peuvent ignorer le serveur de destination par défaut (défini par les options globales) en spécifiant l'hôte et le port d'un nouveau serveur.

Syntaxe

sendevent -V | ? | -help | -u | -usage

```
sendevent [-hostname nom_hôte]  
          [{-port | -sslport} port]  
          type_événement  
          source  
          [[attribute=valeur]...]
```

Arguments

-V Affiche la version de la commande et quitte l'application.

? | -help | -u | -usage
Affiche des informations sur la syntaxe de la commande et quitte l'application.

-hostname *nom_hôte*
Indique le nom d'hôte d'un serveur de processeur d'événement autre que celui actuellement actif.

Ce paramètre est obligatoire si la commande est lancée à partir d'un client de ligne de commande.

-port | -sslport} *port*
Indique le numéro de port d'un serveur de processeur d'événements autre que celui actuellement actif. **-sslport** définit le port utilisé pour détecter les connections SSL entrantes.

Ce paramètre est obligatoire si la commande est lancée à partir d'un client de ligne de commande.

type_événement
L'un des types d'événements personnalisés défini par la commande `evtdef` dans le fournisseur d'événements génériques et spécifié en tant qu'événement déclencheur dans une définition de règle d'événement.

source Nom du fournisseur d'événements que vous avez personnalisé avec `evtdef`. Il s'agit également du nom que vous devez indiquer comme argument du mot clé `eventProvider` dans la définition des règles d'événement déclenchées par ces événements personnalisés.

Le nom par défaut est `GenericEventPlugIn`.

attribut=valeur
L'un ou plusieurs des attributs qualifiant le type d'événement personnalisé qui sont spécifiés en tant qu'attributs d'événement déclencheur pour la règle d'événement.

Commentaires

Cette commande peut également être exécutée sur les systèmes hébergeant uniquement le client de ligne de commande IBM Workload Scheduler.

Exemples

Dans cet exemple, une application envoie le type d'événement personnalisé BusProcCompleted vers un serveur de traitement d'événements alternatif nommé master3. L'événement est que le traitement du fichier calcweek est terminé.

```
sendevent -hostname master3 -port 4294 BusProcCompleted  
GenericEventPlugIn TransacName=calcweek Workstation=ab5supp
```

Le nom de fichier et du poste de travail associé sont les deux attributs d'événement BusProcCompleted qui ont été spécifiés en tant qu'attributs d'événement déclencheur dans une règle d'événement associée.

showexec

Affiche le statut des travaux qui sont en cours d'exécution. Cette commande s'applique uniquement à UNIX. Elle est destinée aux agents standard. Sur les gestionnaires de domaine et les agents tolérants aux pannes, utilisez plutôt la commande **conman showjobs**.

Syntaxe

```
showexec [-V | -U | INFO]
```

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.
- INFO Affiche le nom du fichier de travail au lieu de l'utilisateur, de la date et de l'heure.

Résultats

La sortie de la commande est disponible dans deux formats : **standard** et **INFO**.

Exemples

Pour afficher les travaux en cours d'exécution au format **standard**, exécutez la commande suivante :

```
showexec
```

Pour afficher les travaux en cours d'exécution au format **INFO**, exécutez la commande suivante :

```
showexec INFO
```

Format standard

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux dans lequel le travail est exécuté.

Travail

Nom du travail.

Job# Numéro du travail.

User Nom d'utilisateur du travail.

Start Date

Date de début d'exécution du travail.

Start Time

Heure de début d'exécution du travail.

(Est) Elapse

Durée d'exécution estimée du travail (en minutes).

Format info

UC Poste de travail sur lequel le travail s'exécute.

Schedule

Nom du flot de travaux dans lequel le travail est exécuté.

Travail

Nom du travail.

Job# Numéro du travail.

JCL Nom de fichier du travail.

shutdown

Arrête les processus IBM Workload Scheduler et, éventuellement, WebSphere Application Server. Seuls les postes de travail Windows sont concernés. Vous devez avoir un accès *shutdown* au poste de travail.

Syntaxe

shutdown [-V | -U] [-appsrv]

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-appsrv

Arrête également WebSphere Application Server.

Commentaires

Vérifiez que le *TWS_user* que vous utilisez appartient au groupe Administrators défini sur le poste de travail Windows.

Exemples

Pour afficher le nom et la version de la commande, exécutez la commande suivante :

```
shutdown -V
```

Pour arrêter les processus IBM Workload Scheduler et WebSphere Application Server, exécutez la commande suivante :

```
shutdown -appsrv
```

ShutDownLwa - Arrêt de l'agent

Arrête l'agent. Sur les systèmes Windows, aucun accès spécifique au poste de travail n'est obligatoire. Sur les systèmes UNIX, peut être exécuté par l'utilisateur *TWS* ou par l'utilisateur racine uniquement. Exécutez cette commande en local sur l'agent que vous souhaitez arrêter.

Syntaxe

ShutDownLwa

Arguments

Aucun argument n'est nécessaire.

Exemples

Pour arrêter l'agent, exécutez la commande suivante :

```
ShutDownLwa
```

StartUp

Démarre le processus **netman** de gestion du réseau IBM Workload Scheduler.

Vous devez avoir un accès *start* au poste de travail.

Syntaxe

StartUp [-V | -U]

Arguments

-V Affiche la version de la commande et quitte l'application.

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

Commentaires

Sous Windows, le service **netman** est lancé automatiquement au redémarrage de l'ordinateur. **StartUp** peut être utilisé pour redémarrer le service si celui-ci est arrêté pour une raison quelconque.

Sous UNIX, la commande **StartUp** peut être exécutée automatiquement en l'appelant à partir du fichier `/etc/inittab`, de façon à ce que l'infrastructure WebSphere Application Server et **netman** soient démarrés à chaque fois qu'un ordinateur est redémarré. **StartUp** peut être utilisé pour redémarrer le processus **netman** si celui-ci est arrêté pour une quelconque raison.

Le reste de l'arborescence des processus peut être redémarré avec les commandes

```
conman start  
conman startmon
```

.

Pour plus d'informations, voir «start», à la page 532.

Exemples

Pour afficher le nom et la version de la commande, exécutez la commande suivante :

```
StartUp -V
```

Pour lancer le processus **netman**, exécutez la commande suivante :

```
StartUp
```

StartUpLwa - Démarrage de l'agent

Démarre l'agent. Sur les systèmes Windows, aucun accès spécifique au poste de travail n'est obligatoire. Sous UNIX, peut être exécuté par *TWS_user* ou par l'utilisateur racine uniquement. Exécutez cette commande en local sur l'agent que vous souhaitez arrêter.

Syntaxe

StartUpLwa

Arguments

Aucun argument n'est nécessaire.

Exemples

Pour démarrer l'agent, exécutez la commande suivante :

```
StartUpLwa
```

twc_inst_pull_info

Il s'agit d'un script qui génère des informations sur votre environnement IBM Workload Scheduler et votre poste de travail local et peut prendre des clichés des données DB2 et WebSphere Application Server sur le gestionnaire de domaine maître, en les enregistrant en tant que module daté.

Il peut également générer un rapport contenant non seulement les résultats de la capture d'écran mais également de nombreux paramètres de configuration et d'environnement. Cet outil s'avère très utile pour décrire un problème au service de support logiciel IBM. Pour de meilleurs résultats, il doit être exécuté dès que le problème est identifié.

Commentaires

Pour plus d'informations sur cette commande, voir *IBM Workload Scheduler - Guide d'identification et résolution des problèmes*.

version

Affiche des informations sur la version de IBM Workload Scheduler actuellement installée sur le système. Cette commande s'applique uniquement à UNIX. Les informations sont extraites d'un fichier de version.

Syntaxe

version -V | -u | -h

version [-a] [-f *fichier*] [*fichier* [...]]

Arguments

- V** Affiche la version de la commande et quitte l'application.
- u** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- h** Affiche des informations d'aide sur la commande et quitte l'application.
- a** Affiche des informations relatives aux fichiers produit. Par défaut, seules les informations relatives aux fichiers indiqués s'affichent.
- f *fichier***
Indique le chemin d'accès et le nom du fichier de version s'il est différent de la valeur par défaut. Il s'agit par défaut d'un fichier intitulé **version.info** dans le répertoire de travail en cours.
- fichier*** Indique les noms des fichiers produit, séparés par des espaces, pour lesquels des informations de version sont affichées. Par défaut, aucune information relative aux fichiers ne s'affiche ou, si **-a** est utilisé, toutes les informations relatives aux fichiers s'affichent.

Résultats

L'en-tête de la sortie contient le nom, la version, le système d'exploitation, le niveau de modification et la date d'installation du produit. Le reste de l'écran répertorie des informations relatives au(x) fichier(s) indiqué(s). Les fichiers sont répertoriés au format suivant :

Fichier

Nom du fichier.

Révision

Numéro de révision du fichier.

Modification

Niveau de modification du fichier, s'il y a lieu.

Taille (octets)

Taille du fichier, en octets.

Total de contrôle

Total de contrôle du fichier. Le total de contrôle est calculé via la commande UNIX **sum**. Sous AIX, la commande **sum** est associée à l'argument **-o**.

Commentaires

Les informations relatives au fichier IBM Workload Scheduler se trouvent dans le fichier **version.info**. Ce dernier est placé dans le répertoire *racine_TWS*version lors de l'installation. Le fichier **version.info** est dans un format spécifique et n'est pas modifié.

Vous pouvez transférer le fichier **version.info** vers un autre répertoire. Toutefois, vous devez ensuite inclure l'argument **-f** pour rechercher le fichier.

Exemples

Pour afficher les informations sur la version de IBM Workload Scheduler installée, exécutez la commande suivante :

```
./version
```

Voici un exemple de sortie de cette commande :

```
IBM Workload Scheduler/VERSION 9.21 (C) Copyright IBM Corp 1998, 2013
```

```
IBM Workload Scheduler 9.2 UNIX
```

Pour afficher les informations relatives à tous les fichiers, exécutez la commande suivante :

```
version/version -a -f version/version.info
```

Pour afficher les informations relatives au fichier customize, exécutez la commande suivante :

```
cd version  
./version customize
```

Pour afficher les informations relatives au fichier customize (si version.info se trouve dans le répertoire /apps/maestro), exécutez la commande suivante :

```
cd version  
./version -f /apps/maestro/version.info customize
```

Commandes non prises en charge

Les commandes non prises en charge suivantes proposent des fonctions sous Windows, qui sont similaires aux commandes UNIX **ps** et **kill**. Ils peuvent être utilisés si des utilitaires Windows similaires ne sont pas disponibles.

Syntaxe

listproc

killproc *pid*

Commentaires

listproc

Affiche la liste des processus du système sous forme d'une table.

killproc

Supprime le processus dont l'ID est *pid*.

Remarque : Exécutée par l'administrateur, la commande **killproc** permet de supprimer les processus système.

Chapitre 17. Utilisation des commandes d'utilitaire dans l'environnement dynamique

Le présent chapitre décrit les commandes d'utilitaire IBM Workload Scheduler pour l'environnement dynamique. Alors que certaines commandes s'exécutent sur le gestionnaire de domaine maître ou sur un gestionnaire de domaine dynamique, d'autres s'exécutent sur les agents. Elles sont installées dans *TWA_home*/TDWB/bin, sous réserve des exceptions répertoriées ci-dessous, et s'exécutent avec les systèmes d'exploitation UNIX et Windows. Les commandes d'utilitaire sont exécutées à partir de l'invite de commande du système d'exploitation, sauf l'utilitaire **jobprop** qui s'utilise uniquement dans une définition de travail (voir la section «Transmission d'un jeu de variables d'un travail à un autre dans la même instance de flot de travaux à l'aide de l'utilitaire **jobprop**», à la page 663).

Le tableau 118 contient la liste des commandes d'utilitaire avec, pour chacune d'entre elles, la description et le type de poste de travail où vous pouvez l'exécuter.

Tableau 118. Liste des commandes d'utilitaire pour les postes de travail dynamiques

Commande	Description	Type de poste de travail
exportserverdata	Télécharge la liste des instances de courtier de charge de travail à partir de la base de données IBM Workload Scheduler et modifie un numéro de port ou un nom d'hôte.	gestionnaire de domaine maître ou gestionnaire de domaine dynamique
importserverdata	Télécharge la liste des instances de Dynamic Workload Broker dans la base de données IBM Workload Scheduler après l'édition du fichier temporaire pour changer un numéro de port ou un nom d'hôte.	gestionnaire de domaine maître ou gestionnaire de domaine dynamique
jobprop	Définit les valeurs de variables dans un travail à transmettre au travail successif dans la même instance de flot de travaux. Pour plus d'informations sur l'utilisation de cet utilitaire dans une définition de travail, voir «Transmission d'un jeu de variables d'un travail à un autre dans la même instance de flot de travaux à l'aide de l'utilitaire jobprop », à la page 663. Cet utilitaire est installé dans le répertoire <REP_INSTALLATION_TWS>/TWS/bin et s'exécute sur les systèmes d'exploitation UNIX et Windows.	agents
movehistorydata	Déplace les données présentes dans la base de données IBM Workload Scheduler vers les tables d'archivage	gestionnaire de domaine maître ou gestionnaire de domaine dynamique
param	Crée, affiche et supprime des variables et des mots de passe utilisateur sur des agents dynamiques.	agents Cette commande est installée dans <i>rep_principal_TWA</i> /TWS/CLI/bin.

Tableau 118. Liste des commandes d'utilitaire pour les postes de travail dynamiques (suite)

Commande	Description	Type de poste de travail
resource	Crée, modifie, associe, interroge ou définit des ressources en ligne ou hors ligne.	gestionnaire de domaine maître, gestionnaire de domaine dynamique ou agents
sendevent	Envoie des événements génériques au serveur de processeur d'événement actif.	gestionnaires de domaine dynamiques et agents. Cette commande est installée dans <i>rép_principal_TWA/TWS/CLI/bin</i> .
twstrace	Lors de l'exécution, modifie les paramètres pour la fonction de trace sur les agents	agents Cette commande est installée dans <i>rép_principal_TWA/TWS/CLI/bin</i> .

Remarque : Pour supprimer les fichiers des journaux de travaux pour les postes de travail des agents dynamiques, définissez la valeur de la propriété `MaxAge` dans `JobManager.ini`. Pour plus de détails, voir *IBM Workload Scheduler : Guide d'administration - Configuration de l'agent - Configuration des propriétés communes aux lanceurs [Lanceurs]*.

Fichier de configuration de ligne de commande

Le fichier `CLIConfig.properties` contient les informations de configuration utilisées lors de la saisie de commandes. Par défaut, les arguments requis lors de la saisie des commandes sont extraits de ce fichier, à moins qu'ils ne soient explicitement spécifiés dans la syntaxe des commandes.

Le fichier `CLIConfig.properties` est créé au moment de l'installation et situé sur le gestionnaire de domaine maître dans le chemin d'accès suivant :

`TWA_home/TDWB/config`

Le fichier `CLIConfig.properties` contient l'ensemble de paramètres suivants :

Propriétés par défaut Dynamic workload Broker

ITDWBServerHost

Spécifie l'adresse IP de Dynamic Workload Broker.

ITDWBServerPort

Définit le numéro de port du Dynamic Workload Broker. La valeur par défaut est **9550**.

ITDWBServerSecurePort

Définit le numéro de port du Dynamic Workload Broker lorsque la sécurité est activée. La valeur par défaut est **9551**.

use_secure_connection

Indique si une connexion sécurisée doit être utilisée. La valeur par défaut est **false**.

Nom de fichier et chemin d'accès du fichier de clés et de clés certifiées

Fichier de clés

Indique le nom et le chemin d'accès du fichier de clés contenant les clés privées. Un fichier de clés contient des clés publiques et des

clés privées. Les clés publiques sont stockées comme des certificats de signataire tandis que les clés privées sont stockées dans des certificats personnels. La valeur par défaut est /Certs/TDWBClientKeyFile.jks.

Fichier de clés certifiées

Indique le nom et le chemin d'accès du fichier de clés certifiées contenant les clés publiques. Un fichier de clés certifiées est un fichier de la base de données de clés qui contient des clés publiques. La clé publique est stockée comme un certificat de signataire. Les clés sont utilisées à des fins diverses, y compris pour l'authentification et l'intégrité des données. La valeur par défaut est /Certs/TDWBClientTrustFile.jks.

Mots de passe des fichiers de clés et des fichiers de clés certifiées

keyStorepwd

Définit le mot de passe du fichier de clés.

trustStorepwd

Définit le mot de passe du fichier de clés certifiées.

Types de fichier pour les fichiers de clés et les fichiers de clés certifiées

keyStoreType

Définit le type de fichier pour le fichier de clés certifiées. La valeur par défaut est JKS.

trustStoreType

Définit le type de fichier pour le fichier de clés certifiées. La valeur par défaut est JKS.

ID utilisateur et mot de passe pour Dynamic Workload Broker

tdwb_user

Désigne le nom d'un utilisateur autorisé à effectuer des opérations sur Dynamic Workload Broker lorsque la sécurité est activée. La valeur par défaut est **ibmschedcli**. Ce mot de passe doit être défini auparavant sur IBM WebSphere. Pour plus d'informations sur la sécurité, voir *IBM Workload Scheduler - Guide d'administration*, SC11-6396.

tdwb_pwd

Indique le mot de passe d'un utilisateur autorisé à effectuer des opérations sur Dynamic Workload Broker lorsque la sécurité est activée. Ce mot de passe doit être défini auparavant sur IBM WebSphere. Pour plus d'informations sur la sécurité, voir *IBM Workload Scheduler - Guide d'administration*.

Niveau de détail du journal de ligne de commande et des informations de trace

logger.Level

Indique le niveau de détail des fichiers de trace et des fichiers journaux de ligne de commande. Les fichiers de trace et les fichiers journaux de ligne de commande sont créés aux emplacements suivants :

fichier journal

TWA_home/TDWB/logs/Msg_cli.log.log

Fichier de trace

TWA_home/TDWB/logs/Trace_cli.log

La valeur par défaut est INFO.

logger.consoleLevel

Indique le niveau de détail pour les informations de journal et de trace à diriger vers une sortie standard. La valeur par défaut est FINE. Les valeurs prises en charge pour les paramètres **consoleLevel** et **loggerLevel** sont les suivantes :

TOUS Indique que tous les messages sont consignés.

SEVERE

Indique que les messages d'erreur grave sont consignés.

AVERTISSEMENT

Indique que les messages d'avertissement sont consignés.

INFO Indique que les messages d'information sont consignés.

CONFIG

Indique que les messages de configuration statique sont consignés.

FINE Indique que les informations de la fonction de trace sont consignées.

FINER

Indique que les informations détaillées de la fonction de trace sont consignées.

FINEST

Indique que les informations très détaillées de la fonction de trace sont consignées.

OFF Indique que la fonction de consignation est désactivée.

logger.limit

Indique la taille maximale, en octets, d'un fichier journal. La valeur par défaut est 400000. Lorsque la taille maximale est atteinte, un nouveau fichier est créé jusqu'à ce que le nombre maximal de fichiers soit atteint. Lorsque tous les fichiers atteignent la taille maximale et que le nombre maximal de fichier est dépassé, le fichier le plus ancien est réécrit.

logger.count

Indique le nombre maximal de fichiers journaux. La valeur par défaut est 6. Lorsque la taille maximale est atteinte, un nouveau fichier est créé jusqu'à ce que le nombre maximal de fichiers soit atteint. Lorsque tous les fichiers atteignent la taille maximale et que le nombre maximal de fichier est dépassé, le fichier le plus ancien est réécrit. Lorsqu'un fichier est créé, le suffixe 0 est ajouté après l'extension de fichier. Le fichier qui comporte le suffixe 0 est toujours le fichier actuel. Tous les fichiers plus anciens sont renumérotés.

java.util.logging.FileHandler.pattern

Indique que les informations de trace pour la machine virtuelle Java sont consignées dans le fichier Trace_cli.log. La valeur par défaut est INFO.

java.util.logging.FileHandler.limit

Indique la taille maximale, en octets, d'un fichier de trace. La valeur par défaut est 400000. Lorsque la taille maximale est atteinte, un nouveau fichier est créé jusqu'à ce que le nombre

maximal de fichiers soit atteint. Lorsque tous les fichiers atteignent la taille maximale et que le nombre maximal de fichier est dépassé, le fichier le plus ancien est réécrit.

java.util.logging.FileHandler.count

Indique le nombre maximal de fichiers de trace. La valeur par défaut est 6. Lorsque le nombre maximal est atteint, un nouveau fichier est créé jusqu'à ce que le nombre maximal de fichiers soit atteint. Lorsque tous les fichiers atteignent la taille maximale et que le nombre maximal de fichier est dépassé, le fichier le plus ancien est réécrit. Lorsqu'un fichier est créé, le suffixe 0 est ajouté après l'extension de fichier. Le fichier qui comporte le suffixe 0 est toujours le fichier actuel. Tous les fichiers plus anciens sont renumérotés.

java.util.logging.FileHandler.formatter

Indique le composant de formatage à utiliser pour le fichier Trace_cli.log. La valeur par défaut est com.ibm.logging.icl.jsr47.CBEFormatter.

Configuration DAO commune

Cette section définit les paramètres du système SGBDR pour les commandes **exportserverdata**, **importserverdata** et **movehistorydata**. Ces commandes utilisent un système de gestion de base de données relationnelle installé sur Dynamic Workload Broker. Les valeurs de ces paramètres sont renseignées au moment de l'installation et ne doivent pas être modifiées, à l'exception de com.ibm.tdwb.dao.rdbms.useSSLConnections comme indiqué ci-après.

com.ibm.tdwb.dao.rdbms.rdbmsName

Indique le nom du système SGBDR.

com.ibm.tdwb.dao.rdbms.useDataSource

Indique la source de données à utiliser.

com.ibm.tdwb.dao.rdbms.jdbcPath

Indique le chemin d'accès pour le pilote JDBC.

com.ibm.tdwb.dao.rdbms.jdbcDriver

Indique le pilote JDBC.

com.ibm.tdwb.dao.rdbms.userName

Indique le nom de l'utilisateur du système SGBDR.

com.ibm.tdwb.dao.rdbms.password

Indique le mot de passe de l'utilisateur du système SGBDR.

com.ibm.tdwb.dao.rdbms.useSSLConnections

Indique que l'accès à la base de données DB2 IBM Workload Scheduler par certaines des commandes CLI s'effectue sur SSL. Paramétré sur FALSE par défaut. Vous devez le définir sur TRUE si la base de données est DB2 et que vous utilisez la sécurité FIPS pour exécuter les commandes suivantes :

- **exportserverdata**
- **importserverdata**
- **movehistorydata**

exportserverdata

Utiliser la commande **exportserverdata** pour télécharger (exporter) la liste des instance Dynamic Workload Broker à partir de la base de données IBM Workload Scheduler et modifier un numéro de port ou un nom d'hôte.

Syntaxe

exportserverdata ?

exportserverdata -dbUsr *db_user_name* **-dbPwd** *db_user_password* **-exportFile** *filename*

Description

Cette commande extrait une liste des identificateurs URI (Uniform Resource Identifier) de tous les serveurs instance Dynamic Workload Broker à partir de la base de données IBM Workload Scheduler et les copie dans un fichier temporaire de sorte que, si le nom d'hôte ou le numéro de port de n'importe quelle instance répertoriée est modifié, l'administrateur peut enregistrer ces informations dans le fichier et les replacer dans la base de données avec la commande **importserverdata**. Par défaut, la liste des identificateurs URI est sauvegardée dans le fichier `server.properties`, situé dans le répertoire en cours.

Cette action est nécessaire car la liste des serveurs instance Dynamic Workload Broker doit être tenue à jour et disponible à tout moment, puisque les agents Resource Advisor se connectent périodiquement à l'instance active pour envoyer leurs données sur les ressources détectées sur chaque ordinateur. Ils peuvent passer automatiquement d'une instance de cette liste à l'autre et trouver celle qui est active pour copier ces données dans son référentiel de ressources. Etant donné que le gestionnaire de domaine maître et que chaque maître de sauvegarde sont installés avec un instance Dynamic Workload Broker, le instance Dynamic Workload Broker actif s'exécute dans le gestionnaire de domaine maître pendant qu'une instance de veille réside sur chaque maître de sauvegarde.

L'URI pointant vers chaque instance Dynamic Workload Broker est le suivant :
`https://nom_hôte:numéro_port/JobManagerRESTWeb/JobScheduler`

Vous ne pouvez modifier que le nom d'hôte et le numéro de port.

Important : La liste est ordonnée. Vous pouvez modifier l'ordre des instances tel qu'elles apparaissent dans cette liste et les agents suivront cet ordre. Si vous disposez de plusieurs maîtres de sauvegarde et que vous décidez de suivre un ordre de basculement spécifique en cas d'échec d'un maître, vous pouvez demander aux agents de permuter sur l'instance correcte à l'aide de cette liste ordonnée, ce qui accélère le temps de transition.

Si votre base de données IBM Workload Scheduler est DB2 et que vous utilisez la sécurité FIPS, vous devez définir l'option **com.ibm.tdwb.dao.rdbms.useSSLConnections** sur **TRUE** dans le fichier `CLIConfig.properties` pour exécuter cette commande avec succès.

Options

? Affiche des informations d'aide.

-dbUsr *db_user_name*

Nom d'utilisateur nécessaire pour accéder au serveur de base de données IBM Workload Scheduler.

-dbPwd *db_user_password*

Mot de passe utilisateur nécessaire pour accéder au serveur de base de données IBM Workload Scheduler.

-exportFile *filename*

Nom du fichier temporaire dans lequel les identificateurs URI extraits à partir de la base de données sont copiés pour édition. Ce fichier texte est créé lorsque vous exécutez la commande et vous pouvez l'ouvrir avec n'importe quel éditeur pour modifier le nom d'hôte ou le numéro de port. Si vous n'indiquez pas de chemin, le fichier est créé dans le répertoire où figure la commande :

<TWA_home>/TDWB/bin

Dans la pratique, si vous spécifiez un chemin d'accès différent, assurez-vous que le chemin existe avant d'exécuter cette commande.

Exemple

Pour télécharger la liste actualisée de tous les serveurs instance Dynamic Workload Broker (serveurs actifs et serveurs de sauvegarde) et les copier dans un fichier intitulé `c:\myservers\uris160709`, exécutez :

```
exportserverdata -dbUsr twsadm -dbPwd fprefect -exportFile c:\myservers\uris160709
```

La commande renvoie le fichier `uris160709` qui se présente comme suit :

```
https://accrec015:42127/JobManagerRESTWeb/JobScheduler
https://prodop099:52529/JobManagerRESTWeb/JobScheduler
https://prodop111:31116/JobManagerRESTWeb/JobScheduler
```

`prodop099` est le serveur instance Dynamic Workload Broker actif car il est hébergé par le gestionnaire de domaine maître actuellement actif, tandis que `accrec015` et `prodop111` sont en veille car ils sont hébergés par des maîtres de sauvegarde.

Vous pouvez éditer ce fichier pour appliquer vos modifications avant d'utiliser la commande `importserverdata` pour envoyer à nouveau les identificateurs URI à la base de données.

Voir aussi

«`importserverdata`»

importserverdata

Utiliser la commande **importserverdata** pour importer la liste des serveurs instance Dynamic Workload Broker dans la base de données IBM Workload Scheduler après l'édition du fichier temporaire pour modifier un numéro de port ou un nom d'hôte.

Syntaxe

importserverdata ?

importserverdata -dbUsr *db_user_name* -dbPwd *db_user_password* -importFile *filename*

Description

Cette commande remplace la liste des serveurs instance Dynamic Workload Broker dans la base de données IBM Workload Scheduler à partir du fichier temporaire dans lequel ils ont été téléchargés à l'aide de la commande `exportserverdata`.

Utilisez les commandes `exportserverdata` et `importserverdata` si vous devez enregistrer des modifications de nom d'hôte ou de numéro de port dans les identificateurs URI des instances. Il est nécessaire de disposer de la liste actualisée des serveurs instance Dynamic Workload Broker à tout moment, car les agents Resource Advisor se connectent périodiquement à l'instance active pour envoyer leurs données sur les ressources identifiées sur chaque ordinateur. Ils peuvent passer automatiquement d'une instance de cette liste à l'autre et trouver celle qui est active pour copier ces données dans son référentiel de ressources. Étant donné que le gestionnaire de domaine maître et que chaque maître de sauvegarde sont installés avec une instance Dynamic Workload Broker, l'instance Dynamic Workload Broker actif s'exécute dans le gestionnaire de domaine maître pendant qu'une instance de veille réside sur chaque maître de sauvegarde.

Important : La liste est ordonnée. Vous pouvez modifier l'ordre des instances tel qu'elles apparaissent dans cette liste et les agents suivront cet ordre. Si vous disposez de plusieurs maîtres de sauvegarde et que vous décidez de suivre un ordre de basculement spécifique en cas d'échec d'un maître, vous pouvez demander aux agents de permuter sur l'instance correcte à l'aide de cette liste ordonnée, ce qui accélère le temps de transition.

Si votre base de données IBM Workload Scheduler est DB2 et que vous utilisez la sécurité FIPS, vous devez définir l'option `com.ibm.tdwb.dao.rdbms.useSSLConnections` sur `TRUE` dans le fichier `CLIconfig.properties` pour exécuter cette commande avec succès.

Options

? Affiche des informations d'aide.

-dbUsr *db_user_name*

Nom d'utilisateur nécessaire pour accéder au serveur de base de données IBM Workload Scheduler.

-dbPwd *db_user_password*

Mot de passe utilisateur nécessaire pour accéder au serveur de base de données IBM Workload Scheduler.

-importFile *filename*

Nom du fichier temporaire que vous spécifiez avec le mot clé `-exportFile` dans la commande `exportserverdata`.

Exemple

Pour importer la liste modifiée des identificateurs URI instance Dynamic Workload Broker à partir du fichier `c:\myservers\uris160709` vers la base de données IBM Workload Scheduler :

```
importserverdata -dbUsr twsadm -dbPwd fperfect -importFile c:\myservers\uris160709
```

Voir aussi

«`exportserverdata`», à la page 732

jobprop

Utilisez la commande **jobprop** sur une définition de travail afin de définir des variables localement pour un travail sur des agents dynamiques.

Vous pouvez utiliser cette commande sur un travail natif ou exécutable afin de définir la valeur de variable pouvant être transférée dans un travail successif du même flot de travaux. Les valeurs sont définies lors de l'exécution.

Syntaxe

jobprop *variable valeur*

Arguments

Variable

Nom de variable.

valeur Valeur de *variable*.

Commentaires

Les noms de variable sont sensibles à la casse. Si les noms de variable ou les valeurs contiennent des espaces, ils doivent être placés entre guillemets simples.

Exemples

Sur les systèmes d'exploitation UNIX, l'utilitaire **jobprop** définit les variables suivantes dans le travail exécutable NC125133#JOBA :

- Variable *VAR1* définie sur la valeur *value1*.
- Variable *VAR2* définie sur la valeur *value2*.
- Variable *VAR3* définie sur la valeur *value3*.
- Variable *VAR4* définie sur la valeur *value4*.

NC125133#JOBA

```
TASK
  <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/
scheduling/1.0/jsdl" xmlns:
jsdle="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdle">
  <jsdl:application name="executable">
    <jsdle:executable interactive="false">
      <jsdle:script>#!/bin/sh
. /home/ITAuser/TWA/TWS/tws_env.sh
jobprop VAR1 value1
jobprop VAR2 value2
jobprop VAR3 value3
jobprop VAR4 value4
</jsdle:script>
    </jsdle:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Sample Job Definition"
RCONDSUCC "RC>=0"
RECOVERY STOP
```

Sur les systèmes d'exploitation Windows, l'utilitaire **jobprop** définit les variables suivantes dans le travail exécutable WIN1#JOB1 :

- Variable *var1* définie sur la valeur *value1*.
- Variable *var2* définie sur la valeur *value2*.

- Variable *var3* définie sur la valeur value3.
- Variable *var4* définie sur la valeur value4.

WIN1#JOB1

```
TASK
  <?xml version="1.0" encoding="UTF-8"?>
<jsdl:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/
scheduling/1.0/jsdl" xmlns:
jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl">
  <jsdl:application name="executable">
    <jsdl:executable interactive="false">
      <jsdl:script>
call C:\Progra~1\IBM\TWA\TWS\twc_env.cmd
jobprop var1 value1
jobprop var2 value2
jobprop var3 value3
jobprop var4 value4
      </jsdl:script>
    </jsdl:executable>
  </jsdl:application>
</jsdl:jobDefinition>
DESCRIPTION "Sample Job Definition"
RCCONDSUCC "RC">=0"
RECOVERY STOP
```

Remarque : Avant d'exécuter l'utilitaire **jobprop** dans la définition de travail, veuillez à exécuter la commande **twc_env.cmd** avec la syntaxe correcte `call <REP_INST_TWS>\TWS\twc_env.cmd` où `<REP_INST_TWS>` désigne le répertoire d'installation de IBM Workload Scheduler.

movehistorydata

Utilisez la commande **movehistorydata** lorsque l'accès à la base de données devient trop lent. Cette commande déplace les données du référentiel de travaux vers les tables d'archive

La lenteur de l'accès à la base de données peut être due à un grand nombre d'enregistrements dans celle-ci, par exemple lorsque des soumissions de travaux groupées sont réalisées.

Lorsque vous exécutez cette commande, les travaux sont déplacés dans les tables suivantes de la base de données :

JOA_JOB_ARCHIVES

Contient les instances de travail archivées

JRA_JOB_RESOURCE_ARCHIVES

Contient des informations de ressource liées aux travaux

MEA_METRIC_ARCHIVES

Contient des mesures collectées pour les travaux

Pour plus d'informations sur les tables d'historique, voir *IBM Workload Scheduler - Guide d'administration*, SC11-6396.

Remarque : Selon le nombre de travaux et d'accès à la base de données, une opération de nettoyage peut provoquer des pics d'utilisation de la mémoire ou de l'UC.

Si votre base de données IBM Workload Scheduler est DB2 et que vous utilisez la sécurité FIPS, vous devez définir l'option

com.ibm.tdwb.dao.rdbms.useSSLConnections sur TRUE dans le fichier **CLIconfig.properties** pour exécuter cette commande avec succès.

Syntaxe

movehistorydata ?

```
movehistorydata -dbUsr nom_utilisateur_bdd -dbPwd mdp_utilisateur_bdd  
[-successfulJobsMaxAge heures_max_conserv_travaux_terminés  
[-notSuccessfulJobsMaxAge heures_max_conserv_travaux_échoués ]
```

Description

Cette commande effectue un nettoyage dans la base de données du référentiel de travaux. En fonction des valeurs que vous indiquez, les informations sur les travaux soumis sont déplacées vers la base de données d'archive, puis les informations de cette base de données sont supprimées.

Utilisez cette commande pour substituer temporairement les paramètres définis dans le fichier **JobDispatcherConfig.properties** lorsque des événements inattendus nécessitent un nettoyage de base de données immédiat. Les paramètres contenus dans le fichier **JobDispatcherConfig.properties** restent inchangés. Pour plus d'informations sur le fichier **JobDispatcherConfig.properties**, voir *IBM Workload Scheduler - Guide d'administration*.

Options

? Affiche des informations d'aide.

-dbUsr *db_user_name*

Indique le nom d'utilisateur d'un utilisateur autorisé à effectuer des opérations sur la ligne de commande.

-dbPwd *db_user_password*

Indique le mot de passe d'un utilisateur autorisé à effectuer des opérations sur la ligne de commande.

-successfulJobsMaxAge *successfulJobsMaxAge*

Indique le nombre d'heures de conservation des travaux terminés avec succès ou annulés dans la base de données du référentiel de travaux avant archivage. La valeur est de 240 heures, soit 10 jours.

-notSuccessfulJobsMaxAge *notSuccessfulJobsMaxAge*

Indique le nombre d'heures de conservation des travaux en échec ou dont le statut est inconnu dans la base de données du référentiel de travaux avant archivage. La valeur est de 720 heures, soit 30 jours.

Valeurs de retour

La commande **movehistorydata** renvoie l'une des valeurs suivantes :

0 Indique que la commande **movehistorydata** s'est terminée avec succès.

< > 0 Indique que la commande **movehistorydata** a échoué.

Exemples

1. Pour déplacer dans la base de données d'archive tous les travaux réussis dans les 40 dernières heures, entrez la commande suivante :

```
movehistorydata -dbUsr halmst -dbPwd dgordon -successfulJobsMaxAge 40
```

2. Pour déplacer dans la base de données d'archive tous les travaux, quel que soit leur statut, et retirer de la base de données d'archive tous les travaux de plus de 700 heures, entrez la commande suivante :

```
movehistorydata -dbUsr halmst -dbPwd dgordon -successfulJobsMaxAge 0  
-notSuccessfulJobsMaxAge 0
```

param

Utilisez la commande **param** pour définir et gérer des mots de passe et des variables utilisateur en local sur des agents dynamiques et sur des agents IBM Workload Scheduler for z/OS.

Vous pouvez utiliser cette commande sur les types de travaux avec options avancées. Les valeurs sont résolues lors de la soumission sur l'agent sur lequel le travail est soumis.

Remarque : Sous Windows 2012, la commande n'est pas prise en charge sur Windows PowerShell.

Autorisation

Pour créer, supprimer ou afficher des variables et des mots de passe, vous devez disposer des droits administrateur ou superutilisateur sur le poste de travail qui exécute l'agent ou des droits *TWS_user* sur l'agent.

Syntaxe

```
param -u | -V |  
      {-c | -ec} [fichier.section.|fichier..|section.] variable [valeur] |  
      [fichier.section.|fichier..|section.] variable |  
      {-d | -fd} [fichier.section.|fichier..|section.] variable
```

Arguments

-u Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

-c | -ec

Crée la variable ou le mot de passe *variable* et définit sa valeur *valeur*. La variable ou le mot de passe est placé dans un espace de nom *fichier* que vous pouvez organiser en une ou plusieurs sections nommées *section*.

Si vous ne pouvez pas fournir de nom de fichier *fichier*, la variable ou le mot de passe est placé dans le fichier par défaut *jm_variables* dans le chemin d'accès *agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files* (*/TWA/TWS/ITA/cpa/config/jm_variables_files*) sur l'agent dynamique.

Si vous ne pouvez pas fournir de nom de section *section*, la variable ou le mot de passe est placé dans le corps du fichier.

Important : Si vous définissez un mot de passe, vous devez spécifier une section nommée *password* pour *variable*. Cela indique que *variable* est un mot de passe.

Si vous créez une variable, *variable* est le nom de variable et *valeur* représente sa valeur. Si vous créez un mot de passe, *variable* représente le

nom d'utilisateur et *valeur* est son mot de passe. Si vous n'entrez pas de *valeur* dans les arguments, la commande demande de manière interactive la saisie d'une valeur.

Argument **-c** crée la variable en clair. Argument **-ec** crée la variable dans un format chiffré. Les mots de passe sont chiffrés par défaut également si vous utilisez l'argument **-c**.

-d | -fd

Supprime (**-d**) ou force la suppression (**-fd**) d'un fichier, d'une section ou d'une variable (mot de passe). Vous pouvez utiliser les caractères génériques suivants :

***** Remplace un ou plusieurs caractères alphanumériques.

? Remplace un caractère alphanumérique.

Avec **-d**, la commande demande une confirmation avant la suppression. Avec **-fd**, la commande effectue la suppression sans demander de confirmation.

Lorsque vous supprimez toutes les variables d'une section, cette dernière est supprimée du fichier. Lorsque vous supprimez toutes les sections et toutes les variables d'un fichier, celui-ci est supprimé.

fichier Nom du fichier utilisé comme espace de nom pour *variable*. Si vous n'indiquez pas de *fichier*, la commande utilise le fichier par défaut `jm_variables` dans le chemin d'accès `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files` (`/TWA/TWS/ITA/cpa/config/jm_variables_files`).

Tous les espaces de nom de variables sont dirigés dans le chemin d'accès `agent_installation_path\TWA\TWS\ITA\cpa\config\jm_variables_files` (`/TWA/TWS/ITA/cpa/config/jm_variables_files`).

section Nom de la section à l'intérieur de *fichier* où *variable* est défini. Lorsque *variable* est utilisé pour un mot de passe, est placé dans une section nommée `password`. Aucun nom de section n'est requis pour stocker des variables.

valeur Valeur de *variable*.

Variable

Il peut s'agir d'un nom de variable ou d'une identification utilisateur. S'il est utilisé pour l'identification, il doit être placé dans une section nommée `password` dans le fichier d'espaces de nom.

Commentaires

Pour afficher une variable ou un mot de passe, un fichier d'espaces de noms ou une section, utilisez la commande comme suit :

param [*fichier.section.* | *fichier..* | *section.*] *variable*

Où vous pouvez utiliser les caractères génériques ***** et **?** décrits pour la commande de suppression.

Les fichiers d'espaces de nom, y compris le fichier par défaut `jm_variables`, n'ont pas d'extension.

Les noms de variable sont sensibles à la casse.

Sur les systèmes IBM i, si vous utilisez les interpréteurs de commandes QP2TERM et QSH, les mots de passe sont visibles pendant le processus de création avec param et apparaissent clairement dans les journaux de l'interpréteur de commandes. Pour garantir l'obscurcissement des mots de passe, vous devez utiliser les interpréteurs de commandes AIXTERM ou XTERM.

Exemples

La commande :

```
param -c compassets.hardware.platform1 unix
```

définit la variable platform1 avec la valeur unix dans la section hardware du nouveau fichier ou du fichier existant nommé compassets. La valeur n'est pas chiffrée.

La commande :

```
param -c compassets..platform1 unix
```

définit la variable platform1 avec la valeur unix dans le nouveau fichier ou le fichier existant nommé compassets. La valeur n'est pas chiffrée.

La commande :

```
param -ec hardware.platform1 unix
```

définit la variable platform1 avec la valeur unix dans la section hardware du fichier par défaut *agent_installation_path*\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables. La valeur est chiffrée.

La commande :

```
param -c compassets.password.jladams san07rew
```

définit la variable jladams avec la valeur san07rew dans la section password du nouveau fichier ou du fichier existant nommé compassets. Puisque jladams est défini dans la section password, il est interprété comme un nom d'utilisateur. La valeur san07rew est chiffrée par défaut puisqu'elle est interprétée comme un mot de passe.

La commande :

```
param *.*.platform1
```

répertorie la variable platform1 dans tous ses emplacements définis. C'est-à-dire :

```
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.hardware.platform1=unix
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets..platform1=unix
...\TWA\TWS\ITA\cpa\config\jm_variables_files\jm_variables.hardware.platform1=***
```

La commande :

```
param password.*adam*
```

répertorie toutes les variables y compris la chaîneadam contenue dans la section password de tous les fichiers. Dans ce cas :

```
...\TWA\TWS\ITA\cpa\config\jm_variables_files\compassets.password.jladams=*****
```

La commande :

```
param -d compassets.password.jladams
```

supprime la variable jladams.

La commande :

```
param -d compassets.password.*
```

supprime toutes les variables trouvées dans la section password et supprime donc la section du fichier compassets.

La commande :

```
param -d compassets.*.*
```

supprime tous les contenus (variables et sections contenant des variables) trouvés dans le fichier compassets et supprime donc le fichier.

resource

La commande **resource** permet de créer, modifier, associer ou définir des ressources en ligne ou hors ligne.

En configurant correctement le fichier **CLIConfig.properties** sur l'agent, vous pouvez également exécuter cette commande à partir de n'importe quel agent IBM Workload Scheduler connecté. Pour plus d'informations, voir «Utilisation de la commande resource à partir d'un agent», à la page 749.

Syntaxe

resource ?

```
resource [-usr nom_utilisateur -pwd mot_de_passe ]
{
[-create{ -logical nom -type type[-quantity quantité ][-offline ] |
-group nom[-offline ]}]
|
[-delete{-logical nom |
-group nom }]
|
[-update{-computer nom{[ -setOnline | -setOffline]} |
-logical nom
[-setName nom]
[-setType type]
[-setQuantity quantité]
[-setOnline | -setOffline]
[-addComputer nom |
-addComputerByID ID |
-removeComputer nom |
-removeComputerByID ID]
|
-group nom
[-setName nom]
[-setOnline | -setOffline]
[-addComputer nom |
-addComputerByID ID |
-removeComputer nom |
-removeComputerByID ID |
-addLogical nom |
-removeLogical nom]]}
```

```
|
[-query{-computer nom [-v] |
-logical nom [-v] |
-group nom [-v]}
[-configFile fichier_configuration]
}
```

Description

Utilisez cette commande pour gérer des ordinateurs, des ressources logiques et des groupes de ressources. En particulier, il est possible de :

- Créer, mettre à jour, répertorier et supprimer des ressources logiques ou des groupes
- Créer des ressources logiques, les associer à des ordinateurs, définir des groupes de ressources logiques ou d'ordinateurs, et les définir en ligne ou hors ligne
- Récupérer et mettre à jour des propriétés de ressource à l'aide des options de requête et de mise à jour
- Etablir la liste d'ordinateurs associés à une ressource logique en effectuant une requête détaillée sur celle-ci
- Modifier l'association entre des ordinateurs et des ressources logiques
- Définir des ressources en ligne ou hors ligne et récupérer des propriétés d'ordinateur

Options

? Affiche des informations d'aide.

-usr *nom_utilisateur*

Indique le nom d'un utilisateur autorisé à effectuer des opérations sur la ligne de commande. Cette option est requise lorsque la sécurité est activée et que le nom d'utilisateur n'est pas défini dans le fichier de configuration `CLIconfig.properties` (avec le mot clé `tdwb_user`).

-pwd *mot_passe*

Indique le mot de passe d'un utilisateur autorisé à effectuer des opérations sur la ligne de commande. Cette option est requise lorsque la sécurité est activée et que le mot de passe n'est pas défini dans le fichier de configuration `CLIconfig.properties` (avec le mot clé `tdwb_pwd`).

-create -logical *nom* **-type** *type*

Crée la ressource logique avec le nom et le type spécifiés. Il est également possible de définir une quantité spécifique ou de définir la ressource hors connexion à l'aide de paramètres facultatifs de la façon suivante :

-create -logical *nom* **-type** *type* **-quantity** *quantité* **-offline**

-create -group *nom*

Crée le groupe de ressources du nom spécifié. Il est également possible de le définir hors connexion à l'aide du paramètre facultatif `-offline` de la façon suivante :

-create -group *nom* **-offline**

-delete -logical *nom*

Supprime la ressource logique du nom spécifié.

-delete -group *nom*

Supprime le groupe de ressources du nom spécifié.

-update -computer *nom*

Met à jour le système informatique du nom spécifié. Vous pouvez mettre l'ordinateur en ligne ou hors ligne comme suit :

-update -computer *nom* -setOnline

Définit l'ordinateur spécifié en ligne.

-update -computer *nom* -setOffline

Définit l'ordinateur spécifié hors ligne.

-update -logical *nom*

Met à jour la ressource logique spécifiée. Vous pouvez mettre à jour les propriétés et le statut d'une ressource comme suit :

-update -logical *nom* -setName *nom*

Met à jour le nom de la ressource logique spécifiée.

-update -logical *nom* -setType *type*

Met à jour le type de la ressource logique spécifiée.

-update -logical *nom* -setQuantity *quantité*

Met à jour la quantité de la ressource logique spécifiée.

-update -logical *nom* -setOnline

Définit en ligne la ressource logique spécifiée.

-update -logical *nom* -setOffline

Définit hors ligne la ressource logique spécifiée.

Vous pouvez modifier l'association entre une ressource logique et un ordinateur comme suit :

-update -logical *nom* -addComputer *nom*

Associe la ressource logique spécifiée à l'ordinateur dont le nom est spécifié.

-update -logical *nom* -addComputerByID *ID*

Associe la ressource logique spécifiée à l'ordinateur dont l'ID est spécifié.

-update -logical *nom* -removeComputer *nom*

Supprime l'association entre la ressource logique spécifiée et l'ordinateur dont le nom est spécifié.

-update -logical *nom* -removeComputerByID *ID*

Supprime l'association entre la ressource logique spécifiée et l'ordinateur dont l'ID est spécifié.

-update -group *nom*

Met à jour le groupe de ressources spécifié. Vous pouvez mettre à jour les propriétés et le statut d'un groupe de ressources comme suit :

-update -group *nom* -setName *nom*

Met à jour le nom du groupe de ressources spécifié.

-update -group *nom* -setOnline

Définit en ligne le groupe de ressources spécifié.

-update -group *nom* -setOffline

Définit hors ligne le groupe de ressources spécifié.

Vous pouvez ajouter ou supprimer des ressources logiques ou des ordinateurs dans un groupe de ressources comme suit :

-update -group *nom* -addLogical *nom*

Ajoute la ressource logique avec le nom spécifié au groupe de ressources.

-update -group *nom* -removeLogical *nom*

Supprime la ressource logique avec le nom spécifié du groupe de ressources.

-update -group *nom* -addComputer *nom*

Ajoute l'ordinateur avec le nom spécifié au groupe de ressources.

-update -group *nom* -addComputerByID *ID*

Ajoute l'ordinateur avec l'ID spécifié au groupe de ressources.

-update -group *nom* -removeComputer *nom*

Supprime l'ordinateur avec le nom spécifié du groupe de ressources.

-update -group *nom* -removeComputerByID *ID*

Supprime l'ordinateur avec l'ID spécifié du groupe de ressources.

-query -computer *nom*

Récupère les propriétés suivantes de l'ordinateur spécifié :

- Nom
- ID ordinateur
- Nom du système d'exploitation
- Type du système d'exploitation
- Version du système d'exploitation
- Statut
- Statut de disponibilité

Récupère les propriétés supplémentaires suivantes si vous ajoutez l'option -v :

- Mémoire physique
- Mémoire virtuelle
- Utilisation de l'UC
- Mémoire physique disponible
- Mémoire virtuelle disponible
- Espace de permutation disponible
- Mémoire physique attribuée
- Mémoire virtuelle attribuée
- Espace de permutation distribué
- Nombre de processeurs
- Nombre de processeurs attribué
- Type de processeur
- Vitesse de processeur
- Fabricant
- Modèle
- Numéro de série
- Interfaces réseau
- Systèmes de fichiers

Vous pouvez utiliser un astérisque (*) en guise de caractère générique comme suit :

En tant que paramètre unique

Vous devez l'indiquer entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin>resource -query -computer "*"
```

Cette commande renvoie la liste de tous les ordinateurs existants.

Pour compléter un nom d'ordinateur

Vous devez le nom entier entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin> resource -query -computer "lab123"
```

Cette commande renvoie la liste de tous les ordinateurs existants dont le nom commence par lab123.

-query -logical *nom*

Récupère le nom et le type de la ressource logique spécifiée. Récupère les propriétés supplémentaires suivantes si vous ajoutez l'option -v :

- Statut
- Quantité
- Allocation en cours
- Liste d'ordinateurs

Vous pouvez utiliser un astérisque (*) en guise de caractère générique comme suit :

En tant que paramètre unique

Vous devez l'indiquer entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin>resource -query -logical "*"
```

Cette commande renvoie la liste de toutes les ressources logiques existantes.

Pour compléter un nom de ressource

Vous devez le nom entier entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin> resource -query -logical "myRes"
```

Cette commande renvoie la liste de toutes les ressources logiques existantes dont le nom commence par maRes.

-query -group *nom*

Récupère le nom et le statut du groupe de ressources spécifié. Récupère la liste d'ordinateurs et de ressources logiques contenue dans le groupe de ressources si vous utilisez l'option -v.

Vous pouvez utiliser un astérisque (*) en guise de caractère générique comme suit :

En tant que paramètre unique

Vous devez l'indiquer entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin>resource -query -group "*"
```

Cette commande renvoie la liste de tous les groupes de ressources existants.

Pour compléter un nom de groupe de ressources

Vous devez le nom entier entre guillemets, par exemple :

```
C:\IBM\TWA\TDWB\bin> resource -query -group "myResGrou"
```

Cette commande renvoie la liste de tous les groupes de ressources existants dont le nom commence par myResGrou.

-configFile *fichier_configuration*

Indique le nom et le chemin d'accès d'un fichier de configuration personnalisé. Ce mot clé est facultatif. Si vous ne l'indiquez pas, le fichier de configuration par défaut est utilisé. Pour plus d'informations sur le fichier de configuration, consultez la section relative à **CLIConfig.properties**.

Autorisation

Le nom d'utilisateur et le mot de passe de la commande sont définis dans le fichier CLIConfig.properties. Pour substituer les paramètres définis dans ce fichier, vous pouvez entrer le nom d'utilisateur et le mot de passe dans la commande. Pour plus d'informations sur le fichier de configuration CLIConfig.properties, consultez la section qui lui est consacrée.

Valeurs de retour

La commande **resource** renvoie une des valeurs suivantes :

- 0 Indique que la commande s'est exécutée correctement.
- < > 0 Indique que la commande a échoué.

Exemples

- Pour créer une ressource logique nommée myApplication (de type Applications), entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -create -logical myApplication  
-type Applications
```

Le résultat suivant apparaît :

```
AWKCLI153I  
Ressource logique "myApplication" créée.
```

- Pour mettre à jour la quantité de la ressource logique myApplication, entrez la commande suivante :

```
resource.bat -update -logical myApplication -setQuantity 5  
-usr john -pwd BXVFDCGS
```

Le résultat suivant apparaît :

```
AWKCLI165I Ressource logique "myApplication" mise à jour.
```

- Pour ajouter la relation entre une ressource logique et un ordinateur, entrez la commande suivante :

```
resource.bat -update -logical myApplication -addComputer myComputer  
-usr john -pwd BXVFDCGS
```

Le résultat suivant apparaît :

```
AWKCLI165I Ressource logique "myApplication" mise à jour.
```

- Pour récupérer les détails d'une ressource logique myApplication, entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -query -logical myApplication -v
```

Le résultat suivant apparaît :

```
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer  
une requête sur les ressources.
```

```
AWKCLI172I "1" ressources logiques ont été trouvées pour votre requête.  
Les détails sont les suivants :
```

```
Resource Name:myApplication
```



```
Resource Type:Applications
Resource Status:Online
Resource Quantity:5
Resource Current Allocation:0
Computers List:
  Computer Name:myComputer
  Computer ID:D656470E8D76409F9F4FDEB9D764FF59
  Computer Status:Online
  Computer Availability Status:Unavailable
```

- Pour définir la ressource logique myApplication hors ligne, entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -update -logical myApplication
-setOffline
```

Le résultat suivant apparaît :

```
AWKCLI165I Ressource logique "myApplication" mise à jour.
```

- Pour définir l'ordinateur myApplication hors ligne, entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -update -computer myComputer
-setOffline
```

Le résultat suivant apparaît :

```
AWKCLI165I
Ordinateur "myComputer" mis à jour.
```

- Pour récupérer les propriétés de base de l'ordinateur myComputer, entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -query -computer myComputer
```

Le résultat suivant apparaît :

```
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.
AWKCLI174I "1" ordinateurs ont été trouvés pour votre requête.
Les détails sont les suivants :
```

```
Computer Name:myComputer
Computer ID:D656470E8D76409F9F4FDEB9D764FF59
Computer OS Name: Microsoft Windows XP Professional English (United States) version
Computer OS Type:Windows XP
Computer OS Version:5
Computer Status:Offline
Computer Availability Status:Unavailable
```

- Pour récupérer les propriétés détaillées de l'ordinateur myComputer, entrez la commande suivante :

```
resource.bat -usr john -pwd BXVFDCGS -query -computer myComputer -v
```

Le résultat suivant apparaît :

```
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.
AWKCLI174I "1" ordinateurs ont été trouvés pour votre requête.
Les détails sont les suivants :
```

```
Computer Name:myComputer
Computer ID:D656470E8D76409F9F4FDEB9D764FF59
Computer OS Name:Microsoft Windows XP Professional English (United States) version
Computer OS Type:Windows XP
Computer OS Version:5
Computer Status:Offline
Computer Availability Status:Unavailable
```

Computer details:

Physic memory = 2095536.0
Virtual memory = 3513788.0
Cpu utilization = 16.0
Free physic memory = 947972.0
Free virtual memory = 2333484.0
Free swap space = 52.0
Allocated physic memory = 0.0
Allocated virtual memory = 0.0
Allocated swap space = 0.0
Processors number = 1.0
Allocated processors number = 0.0
Processor type = x86
Processor speed = 1995.00
Manufacturer = IBM
Model = 2668F8G
Serial number = L3WZYNC

- Pour récupérer les propriétés détaillées de la ressource logique geneva, y compris la liste des ordinateurs associés, entrez la commande suivante :
resource.bat -usr john -pwd BXVFDCGS -query -logical geneva -v

Le résultat suivant apparaît :

Définition des variables d'environnement de l'interface CLI....
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.
AWKCLI172I "1" ressources logiques ont été trouvées pour votre requête.
Les détails sont les suivants :

Resource Name:geneva
Resource Type:prod_wks
Resource Status:Online
Resource Quantity:1
Resource Current Allocation:0
Computers List:
Computer Name:bd_ff139_1
Computer ID:666AADE61CBA11E0ACBECD0E6F3527DE
Computer Status:Online
Computer Availability Status:Available
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.

- Pour créer un groupe de ressources nommé myGroup, entrez la commande suivante :
resource.bat -usr john -pwd BXVFDCGS -create -group myGroup

Le résultat suivant apparaît :

AWKCLI153I
Groupe de ressources "myGroup" créé.

- Pour récupérer les propriétés de base du groupe de ressources myGroup, entrez la commande suivante :
resource.bat -query -group myGroup

Le résultat suivant apparaît :

Définition des variables d'environnement de l'interface CLI....
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.
AWKCLI173I "1" groupes ont été trouvés pour votre requête.
Les détails sont les suivants :

```
Group Name:myGroup
Group Status:Online
```

- Pour ajouter l'ordinateur myComputer à un groupe de ressources nommé myGroup, entrez la commande suivante :

```
resource.bat -update -group myGroup -addComputer myComputer
```

Le résultat suivant apparaît :

```
Définition des variables d'environnement de l'interface CLI....
AWKCLI165I Groupe de ressources "myGroup" mis à jour.
```

- Pour récupérer les détails d'un groupe de ressources nommé myGroup, entrez la commande suivante :

```
resource.bat -query -group myGroup -v
```

Le résultat suivant apparaît :

```
Définition des variables d'environnement de l'interface CLI....
AWKCLI171I Appel du référentiel de ressources en cours pour effectuer
une requête sur les ressources.
AWKCLI173I "1" groupes ont été trouvés pour votre requête.
Les détails sont les suivants :
```

```
Group Name:myGroup
Group Status:Online
Computers List:
  Computer Name:myComputer
    Computer ID:D656470E8D76409F9F4FDEB9D764FF59
    Computer Status:Online
  Computer Availability Status:Unavailable
```

```
Resources List:
```

Utilisation de la commande resource à partir d'un agent

Vous pouvez créer et gérer des ressources et des groupes de ressources à partir d'agents IBM Workload Scheduler autres que le gestionnaire de domaine maître.

Activation de la commande resource

Pour activer cette fonction, vous devez :

1. Ajouter l'environnement d'exécution pour les travaux Java lors de l'installation de l'agent. Pour plus d'informations sur la procédure d'installation de l'agent, voir *Planification et installation*.
2. Configurez le fichier **CLIconfig.properties**. Voir la section sur le fichier **CLIconfig.properties**.
3. Exécutez la commande **resource**. Voir «Exécution de la commande resource», à la page 750.

A cet effet, une instance supplémentaire du fichier **CLIconfig.properties** est installée sur chaque agent. Si vous envisagez d'exécuter la commande **resource** à partir d'un agent, vous devez configurer le fichier **CLIconfig.properties** localement.

Configuration du fichier CLConfig.properties local

Lorsque vous installez l'agent, une copie locale de **CLConfig.properties** est installée automatiquement et configurée partiellement sur votre agent dans le chemin d'accès suivant :

rép_base_TWA/TWS/TDWB_CLI/config

Pour exécuter la commande **resource.bat** ou **resource.sh** à partir de l'agent, personnalisez les mots clés suivants du fichier **CLConfig.properties** local :

ITDWBServerHost

Indiquez l'adresse IP ou le nom d'hôte du gestionnaire de domaine maître.

ITDWBServerPort

Indiquez le numéro de port HTTP de WebSphere Application Server.

ITDWBServerSecurePort

Indiquez le numéro de port HTTPS de WebSphere Application Server.

tdwb_user

Indiquez le nom d'utilisateur d'un utilisateur autorisé à effectuer des opérations sur IBM Workload Scheduler lorsque la sécurité est activée. Cet utilisateur doit être défini auparavant sur IBM WebSphere. Pour plus d'informations sur la sécurité, voir *IBM Workload Scheduler - Guide d'administration*, SC11-6396.

tdwb_pwd

Indiquez le mot de passe d'un utilisateur autorisé à effectuer des opérations sur IBM Workload Scheduler lorsque la sécurité est activée. Ce mot de passe doit être défini auparavant sur IBM WebSphere. Pour plus d'informations sur la sécurité, voir *IBM Workload Scheduler - Guide d'administration*.

Exécution de la commande resource

En fonction du système d'exploitation utilisé, entrez ce qui suit pour exécuter la commande :

Sous Windows

`resource.bat`

Sous UNIX

`resource.sh`

Permutation des gestionnaires

Vous pouvez définir dans le fichier **CLConfig.properties** les serveurs de courtier de sauvegarde que l'interface CLI de ressource doit contacter si le serveur de courtier en cours ne répond pas. Pour configurer l'interface CLI de ressource pour contacter les serveurs de sauvegarde en cas d'indisponibilité du serveur actuel, vous devez spécifier dans le fichier **CLConfig.properties** les propriétés de connexion de chacun des serveurs de courtier de sauvegarde. Les propriétés à spécifier sont les mêmes que celles du serveur de courtier exécuté sur le gestionnaire de domaine maître principal.

Définissez les propriétés de connexion suivantes :

```
ITDWBServerHost
ITDWBServerPort
ITDWBServerSecurePort
use_secure_connection
tdwb_user
tdwb_pwd
```

Pour les serveurs de sauvegarde, le même nombre ordinal doit être ajouté à chaque nom de propriété associé au même serveur de sauvegarde.

Dans l'exemple suivant, le fichier `CLIconfig.properties` spécifie le serveur de courtier exécuté sur le gestionnaire de domaine maître principal et deux serveurs de courtier de sauvegarde :

```
# Propriétés du serveur de courtier exécuté sur le gestionnaire de domaine maître principal
ITDWBServerHost = BrokerServer.mycompany.com
ITDWBServerPort = 51117
ITDWBServerSecurePort = 51118
use_secure_connection = true
tdwb_user = tdwbUser
tdwb_pwd = xxxx

# Propriétés du premier serveur de courtier de sauvegarde (_1)
ITDWBServerHost_1 = FirstBackupBrokerServer.mycompany.com
ITDWBServerPort_1 = 41117
ITDWBServerSecurePort_1 = 41118
use_secure_connection_1 = false
tdwb_user_1 = backup1TdwbUser
tdwb_pwd_1 = yyyy

# Propriétés du second serveur de courtier de sauvegarde
ITDWBServerHost_2 = SecondBackupBrokerServer.mycompany.com
ITDWBServerPort_2 = 61117
ITDWBServerSecurePort_2 = 61118
use_secure_connection_2 = false
tdwb_user_2 = backup2TdwbUser
tdwb_pwd_2 = zzzz
```

Vous pouvez définir jusqu'à 10 serveurs de courtier.

Pour que l'interface CLI de ressource ne contacte pas des serveurs indisponibles, le nom du serveur de courtier connecté est enregistré dans la propriété `ITDWBLastGoodServerHost` du fichier `CLIconfig.properties`.

sendevent

La commande envoie à partir d'un agent dynamique ou d'un gestionnaire de domaine les événements personnalisés définis par la commande `evtdef` vers le serveur de processeur d'événements actuellement actif dans le plan de production. Lors de la réception des événements par le processeur d'événements, ils déclenchent les règles d'événement dans lesquelles ils ont été spécifiés.

Remarque : Sous Windows 2012, la commande n'est pas prise en charge sur Windows PowerShell.

Les utilisateurs peuvent ignorer le serveur de destination par défaut (défini par les options globales) en spécifiant le nom d'hôte et le port d'un nouveau serveur.

Syntaxe

sendevent -V | ? | -help | -u | -usage

sendevent [-hostname *nom_hôte*]
[-port *port*]
type_événement
source
[[*attribute=valeur*]...]

Arguments

-V Affiche la version de la commande et quitte l'application.

? | -help | -u | -usage

Affiche des informations sur la syntaxe de la commande et quitte l'application.

-hostname *nom_hôte*

Indique le nom d'hôte d'un serveur de processeur d'événement autre que celui actuellement actif.

-port *port*

Indique le numéro de port d'un serveur de processeur d'événements autre que celui actuellement actif.

type_événement

L'un des types d'événements personnalisés défini par la commande evtdef dans le fournisseur d'événements génériques et spécifié en tant qu'événement déclencheur dans une définition de règle d'événement.

source Nom du fournisseur d'événements que vous avez personnalisé avec evtdef. Il s'agit également du nom que vous devez indiquer comme argument du mot clé eventProvider dans la définition des règles d'événement déclenchées par ces événements personnalisés.

Le nom par défaut est GenericEventPlugIn.

attribut=valeur

L'un ou plusieurs des attributs qualifiant le type d'événement personnalisé qui sont spécifiés en tant qu'attributs d'événement déclencheur pour la règle d'événement.

Commentaires

La commande s'applique à l'environnement dynamique uniquement. Pour envoyer des événements à partir d'agents non dynamiques, voir «sendevent», à la page 719.

Exemples

Dans cet exemple, une application s'exécutant sur un agent dynamique envoie le type d'événement personnalisé BusProcCompleted vers le processeur d'événement par défaut. L'événement est que le traitement du fichier calcweek est terminé.

```
sendevent BusProcCompleted GenericEventPlugIn TransacName=calcweek  
Workstation=acagn002
```

Le nom de fichier et du poste de travail associé sont les deux attributs d'événement BusProcCompleted qui ont été spécifiés en tant qu'attributs d'événement déclencheur dans une règle d'événement associée.

twstrace

Modifie le niveau de trace sur l'agent dynamique et, en même temps, définit le même niveau de trace sur la passerelle du gestionnaire de travaux, sans qu'il soit nécessaire d'arrêter ou de redémarrer l'agent.

Remarque : Sous Windows 2012, la commande n'est pas prise en charge sur Windows PowerShell.

Autorisation

Vous devez vous connecter avec les données d'identification de l'utilisateur qui a installé l'agent dynamique. Vous pouvez également utiliser toute autorisation supérieure à celle de l'utilisateur qui a installé l'agent dynamique.

Syntaxe

```
twstrace -u | -V | -enable | -disable [-level valeur] [-maxFilesnombre_fichiers] [-maxFileBytes nombre_octets] [-getLogs [-zipFile nom_fichier_zip] [-hostnom_hôte] [-protocol {http|https}] [-port numéro_port] [-infile nom_fichier_ini]]
```

Arguments

enable

Active la trace à son niveau maximum. Le niveau maximum est **3000**. Par défaut, les traces sont désactivées.

disable

Désactive la fonction de trace.

level *valeur*

Niveau de détail des traces :

- 1000** Les messages d'erreur, d'avertissement et d'information font l'objet d'une trace.
- 2000** Les messages d'erreur et d'avertissement font l'objet d'une trace.
- 3000** Les messages d'erreur font l'objet d'une trace.

maxFiles *nombre_fichiers*

Nombre maximal de fichiers de trace que vous souhaitez créer.

maxFileBytes *nombre_octets*

Taille maximale en octets que peut atteindre le fichier de trace. La valeur par défaut est **1024000** octets.

getLogs

Permet de collecter les fichiers de trace, les fichiers messages et les fichiers de configuration dans un fichier compressé.

zipfile *nom_fichier_zip*

Nom du fichier compressé qui contient toutes les informations, à savoir des journaux, des traces et des fichiers de configuration (ita.ini et jobManager.ini) pour l'agent. La valeur par défaut est logs.zip.

host *nom_hôte*

Nom d'hôte ou adresse IP de l'agent pour lequel vous souhaitez collecter les traces. La valeur par défaut est **localhost**.

protocol *http|https*

Protocole de l'agent pour lequel vous collectez la trace. Par défaut, il s'agit du protocole indiqué dans le fichier **.ini** de l'agent.

port *numéro_port*

Port de l'agent. Par défaut, il s'agit du numéro de port de l'agent sur lequel vous exécutez la ligne de commande.

inifile *nom_fichier_ini*

Nom du fichier **.ini** qui contient la configuration SSL de l'agent pour lequel vous souhaitez collecter les traces. Par défaut, il s'agit du fichier **.ini** de l'agent local. Si vous collectez la trace d'un agent distant pour lequel vous avez personnalisé les certificats de sécurité, vous devez importer le certificat sur l'agent local et indiquer le nom du fichier **.ini** qui contient la configuration. Pour cela, procédez comme suit :

1. Extrayez le certificat du fichier de clés de l'agent distant.
2. Importez le certificat sur le fichier de clés d'un agent local. Vous pouvez créer un fichier de clés ad hoc dont le nom doit être **TWSClientKeyStore.kdb**.
3. Créez un fichier **.ini** dans lequel vous précisez :
 - 0 dans la propriété **tcp_port** comme suit :
tcp_port=0
 - Port de l'agent distant dans la propriété **ssl_port** comme suit :
ssl_port=<ssl_port>
 - Chemin d'accès au fichier de clés que vous avez créé à l'étape 2 dans la propriété **key_repository_path** comme suit :
key_repository_path=<local_agent_keystore_path>

r Affiche la syntaxe de la commande.

V Affiche la version du produit.

Exemples

Pour définir le niveau de trace afin d'enregistrer les messages d'erreur et d'avertissement, exécutez la commande suivante :

```
twstrace -enable -level 2000
```

Pour extraire les informations sur le niveau de trace, exécutez la commande suivante :

```
twstrace -level -maxFiles -maxFileBytes
```

AWSITA1761 Les propriétés de trace sont les suivantes : level="1000", maxFiles="3", file size="1024000"

Chapitre 18. Extraction des états et des statistiques

Le présent chapitre détaille les commandes de génération d'états qui permettent de préparer un récapitulatif ou les détails du plan de production précédent ou suivant. Ces commandes sont exécutées à partir de l'invite de commande du système d'exploitation sur le gestionnaire de domaine maître. Il comprend les sections suivantes :

- «Configuration de l'utilisation des commandes de génération d'états»
- «Description des commandes», à la page 756
- «Exemples de sorties d'état», à la page 764
- «Programmes d'extraction d'états», à la page 774
- «Exécution de rapports Dynamic Workload Console et de rapports de traitement par lots», à la page 785
- «Exécution des rapports de traitement par lots à partir de l'interface de ligne de commande», à la page 792

Configuration de l'utilisation des commandes de génération d'états

Pour configurer l'environnement pour utiliser les commandes de génération d'états, définissez les variables *PATH* et *TWS_TISDIR* en exécutant l'un des scripts suivants :

- `./racine_TWS/tws_env.sh` pour les interpréteurs de commandes Bourne et Korn sous UNIX
- `./racine_TWS/tws_env.csh` pour les interpréteurs de commandes C sous UNIX
- `racine_TWS\tws_env.cmd` dans Windows

Les commandes de génération d'états doivent être exécutées à partir du répertoire *TWS_home*.

Les variables d'environnement suivantes contrôlent le résultat des commandes de génération d'états :

MAESTROL

Indique la destination du résultat d'une commande. La valeur par défaut est **stdout**. Vous pouvez lui attribuer l'une des valeurs suivantes :

nom_fichier

Le résultat est écrit dans un fichier.

> *nom_fichier*

Sous UNIX uniquement. Le résultat est redirigé vers un fichier dont le contenu est remplacé. Si le fichier n'existe pas, il est créé.

>> *nom_fichier*

Sous UNIX uniquement. Le résultat est redirigé vers un fichier et ajouté à la fin de ce dernier. Si le fichier n'existe pas, il est créé.

| *commande*

Sous UNIX uniquement. La sortie est dirigée vers une commande système ou un processus. La commande système est toujours exécutée.

|| commande

Sous UNIX uniquement. La sortie est dirigée vers une commande système ou un processus. La commande système n'est pas exécutée en l'absence de résultat.

MAESTRO_OUTPUT_STYLE

Indique le style du résultat pour les noms d'objet longs. Avec la valeur **LONG**, des zones longues sont utilisées pour les noms d'objet.

Si la valeur affectée à cette variable est différente de **LONG**, les noms longs sont tronqués à huit caractères suivis du signe plus. Par exemple : **A1234567+**.

Pour obtenir le format de rapport d'état correct, vous devez utiliser une taille de police fixe.

Modification du format de date

Dans IBM Workload Scheduler, le format de date affecte toutes les commandes qui acceptent une date comme option d'entrée (à l'exception de la commande **datecalc**) ainsi que les en-têtes de tous les états. Le format de date par défaut est *mm/jj/aa*. Pour sélectionner un format différent, éditez l'option locale *format de date* dans le fichier *localopts*. Les valeurs possibles sont les suivantes :

Tableau 119. Formats de date

Valeur du format de date	Résultat de format de date correspondant
0	jj/mm/aa
1	jj/mm/aa
2	jj/mm/aa
3	Les variables NLS.

Voir *IBM Workload Scheduler - Guide d'administration* pour plus d'informations sur la modification des variables locales du fichier *localopts*.

Description des commandes

Les commandes de génération d'états de IBM Workload Scheduler sont répertoriées dans le tableau 120 :

Tableau 120. Liste des commandes de génération d'états

Commande	Description
rep1	Etat 01 - Liste des caractéristiques des travaux
rep2	Etat 02 - Liste des invites
rep3	Etat 03 - Liste des agendas des utilisateurs
rep4a	Etat 04A - Liste des paramètres des utilisateurs
rep4b	Etat 04B - Liste des ressources TWS
rep7	Etat 07 - Liste des historiques des travaux
rep8	Etat 08 - Histogramme des travaux
rep11	Etat 11 - Calendrier de production planifiée

Tableau 120. Liste des commandes de génération d'états (suite)

Commande	Description
reptr	Etat 09A - Récapitulatif de la production planifiée Etat 09B - Détail de la production planifiée Etat 09D - Détail de la production planifiée (format étendu) Etat 10A - Récapitulatif de la production réelle Etat 10B - Détails de la production réelle
xref	Etat 12 - Etat de références croisées

rep1 à rep4b

Ces commandes permettent d'imprimer les états suivants :

Etat 01

Liste des caractéristiques des travaux

Etat 02

Liste des invites

Etat 03

Liste des agendas

Etat 04A

Liste des paramètres

Etat 04B

Liste des ressources

Syntaxe

rep[*x*] [-V|-U]

Exécutez la commande à partir du répertoire *TWS_home*.

Pour rep3, exécutez la commande à partir d'un répertoire sur lequel vous avez l'accès en écriture *write*.

Lors de l'impression des rapports pour les types de travaux avec options avancées, la zone du fichier JCL renvoie le nom de l'application.

Arguments

- x* Numéro de l'état. Les numéros possibles sont les suivants : **1, 2, 3, 4a** ou **4b**.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- V** Affiche la version de la commande et quitte l'application.

Commentaires

La Liste des caractéristiques des travaux (rapport 01) ne peut pas contenir de travaux ayant été soumis via un nom d'alias.

Le temps écoulé affiché pour un travail replet est le temps écoulé du travail distant lié.

Exemples

Imprimez l'Etat 03 "Liste des agendas des utilisateurs" :

```
rep3
```

Affichez les informations d'utilisation pour la commande **rep2** :

```
rep2 -U
```

Sous UNIX, imprimez deux exemplaires de l'Etat 04A "Liste des paramètres des utilisateurs" sur l'imprimante lp2 :

```
MAESTROLP="| lp -dlp2 -n2"
```

```
export MAESTROLP
```

```
rep4a
```

Voici un exemple de rapport pour le travail WAGES2_1 :

```
Travail : WAGES2_1      #FTP      Description :
Fichier JCL : filetransfer
Connexion :              Créateur : tws86
Travail de reprise :
Type de reprise : STOP
Invite de reprise :
Autodoc composer : Oui
Nombre total d'exécutions : 0 - 0 réussies, 0 abandonnées
```

		Ecoulé(mins)	CPU(secs)	
Total	0	0		
Normal	0			
Dernière exécution	0		0 (De 0 à 0)	
Maximum	0	0 (De 0)		
Minimum	0	0 (De 0)>		

rep7

Cette commande permet d'imprimer l'Etat 07 - Liste des historiques des travaux.

Syntaxe

```
rep7 -V|-U
```

```
rep7
```

```
[-c poste_travail]
[-s nom_flot_données]
[-j travail]
[-f date ]
[-t date]
[-l]
```

Exécutez la commande à partir du répertoire *TWS_home*.

Arguments

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

-c *poste_travail*

Indique le nom du poste de travail sur lequel les travaux s'exécutent. Par défaut, tous les postes de travail sont pris en compte.

-s *nom_flot_travaux*

Indique le nom du flot de travaux dans lequel les travaux s'exécutent. Par défaut, tous les flots de travaux sont pris en compte.

-j *travail*

Indique le nom du travail. Par défaut, tous les travaux sont pris en compte.

-f *date*

Indique que l'historique des travaux est imprimé à partir de cette date. Entrez la date au format *aaaammjj*. Par défaut, la date disponible la plus ancienne est prise en compte.

-t *date*

Indique que l'historique des travaux est imprimé jusqu'à cette date. Entrez la date au format *aaaammjj*. Par défaut, la date la plus récente est prise en compte.

-l

Limite les informations de ligne de résumé aux travaux appartenant à la plage de dates définie par les options **-f** ou **-t**. L'utilisation de cette option entraîne l'inversion du résultat : la ligne de résumé est imprimée après les lignes d'exécution individuelles. Cette option est valable uniquement si vous précisez aussi au moins l'une des options **-f** ou **-t**.

Commentaires

Le temps écoulé affiché pour un travail reflet est le temps écoulé du travail distant lié.

Chaque fois que vous exécutez **rep7**, la sortie de la commande contient les informations enregistrées jusqu'à la dernière exécution de **JnextPlan**. Les informations concernant l'exécution du plan de production courant seront contenues dans la sortie **rep7** lors de la prochaine exécution de **JnextPlan**. Pour cette raison, si vous exécutez **rep7** après la première génération du plan de production ou après une commande **ResetPlan**, la sortie de la commande ne contient aucune information statistique.

Exemples

Imprimez l'historique de tous les travaux pour le poste de travail ux3 :

```
rep7 -c ux3
```

Imprimez l'historique de tous les travaux contenus dans le flot de travaux sked25 :

```
rep7 -s sked25
```

Imprimez l'historique de tous les travaux contenus dans le flot de travaux mysked sur le poste de travail x15 entre le 21 janvier 2005 et le 25 janvier 2005 :

```
rep7 -c x15 -s mysked -f 20050121 -t 20050125
```

rep8

Cette commande permet d'imprimer l'état 08 - Histogramme des travaux.

Syntaxe

rep8 **-V|-U**

rep8

[-f date -b heure -t date -e heure]

[-i fichier]

[-p]

rep8
[-b *heure* -e *heure*]
[-i *fichier*]
[-p]

Exécutez la commande à partir du répertoire *TWS_home*.

Arguments

- U Affiche des informations sur la syntaxe de la commande et quitte l'application.
 - V Affiche la version de la commande et quitte l'application.
 - f *date* Indique que l'historique des travaux est imprimé à partir de cette date. Entrez la date au format *aaaammjj*. La valeur par défaut est la date du jour.
 - b *heure*
Indique que l'historique des travaux est imprimé à partir de cette heure. Entrez l'heure au format *hhmm*. La valeur par défaut correspond au paramètre *startOfDay* de IBM Workload Scheduler.
 - t *date* Indique que l'historique des travaux est imprimé jusqu'à cette date. Entrez la date au format *aaaammjj*. Par défaut, la date la plus récente est prise en compte.
 - e *heure*
Indique que l'historique des travaux est imprimé jusqu'à cette heure. Entrez l'heure au format *hhmm*. La valeur par défaut correspond à l'heure de début de la journée d'IBM Workload Scheduler.
 - i *fichier*
Indique le nom du fichier journal duquel est extrait l'historique des travaux. Notez que les fichiers journaux sont stockés dans le répertoire *schedlog*. Par défaut, il s'agit du fichier *Symphony* courant.
- Remarque :** Assurez-vous que la plage horaire indiquée par les arguments [-f *date* -b *heure* -t *date* -e *heure*] est comprise dans la plage horaire et la date définies dans le nom du fichier journal -i *fichier*.
- p Indique qu'un saut de page est inséré après chaque date d'exécution.

Commentaires

Chaque fois que vous exécutez **rep8**, la sortie de la commande contient les informations enregistrées jusqu'à la dernière exécution de **JnextPlan**. Les informations concernant l'exécution du plan de production courant seront contenues dans la sortie **rep8** lors de la prochaine exécution de **JnextPlan**. Pour cette raison, si vous exécutez **rep8** après la première génération du plan de production ou après une commande **ResetPlan**, la sortie de la commande ne contient aucune information statistique.

Exemples

Imprimez un histogramme des travaux qui inclut toutes les informations du plan courant (fichier *Symphony*) :

rep8

Imprimez un histogramme des travaux qui commence à 6 heures du matin le 25 janvier 2005 et qui se termine à 5 heures 59 du matin le 26 janvier 2005.

```
rep8 -f 20050125 -b 0600 -t 20050126 -e 0559 -i schedlog/M199801260601
```

Imprimez un histogramme des travaux, à partir du plan courant (fichier Symphony), qui commence à 6 heures du matin et se termine à 22 heures :

```
rep8 -b 0600 -e 2200
```

rep11

Cette commande permet d'imprimer l'état 11 "Calendrier de production planifiée".

Syntaxe

rep11 -V|-U

rep11

```
[-m mm[aa] [...]]  
[-c stat_travail [...]]  
[-s nom_flot_données]  
[-o sortie]
```

Exécutez la commande à partir du répertoire *TWS_home*.

Arguments

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

-m mm[aa]

Indique les mois à prendre en compte. Entrez le numéro du mois au format *mm*. Par défaut, le mois courant est pris en compte.

Vous pouvez également indiquer une année en utilisant le format *aa*. Par défaut, l'année courante est prise en compte ou l'année suivante si vous indiquez un mois qui précède le mois courant.

-c poste_travail

Indique les postes de travail à prendre en compte dans les états. Par défaut, tous les postes de travail sont pris en compte.

-s nom_flot_travaux

Indique le nom du flot de travaux dans lequel les travaux s'exécutent. Par défaut, tous les flots de travaux sont pris en compte.

-o sortie

Indique le fichier de sortie. Il s'agit, par défaut, du fichier défini par la variable *MAESTROL*. Si la variable *MAESTROL* n'est pas définie, le fichier par défaut est **stdout**.

Exemples

Générez un état pour les mois de juin, juillet et août 2004 pour les postes de travail principal, sitel et agent1 :

```
rep11 -m 0604 0704 0804 -c main sitel agent1
```

Générez un état pour les mois de juin, juillet et août de l'année en cours pour tous les postes de travail et dirigez la sortie vers le fichier r11out :

```
rep11 -m 06 07 08 -o r11out
```

Générez un état pour le mois et l'année en cours pour le poste de travail site2 :
repl1 -c site2

reptr

Cette commande permet d'imprimer les états suivants :

Etat 09A

Récapitulatif de la production planifiée

Etat 09B

Détail de la production planifiée

Etat 10A

Récapitulatif de la production réelle

Etat 10B

Détail de la production réelle

Les états 09A et 09B se rapportent à un traitement de production futur alors que les états 10A et 10B affichent les résultats et le statut du traitement de chaque travail de la production déjà traitée.

Syntaxe

reptr [-V|-U]

reptr -pre

[-{summary | detail}]

[fichier_sym]

reptr -post

[-{summary | detail}]

[fichier_journal]

Exécutez la commande à partir d'un répertoire auquel vous avez accès write.

Arguments

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

-pre Permet de demander l'impression des états de préproduction (09A et 09B).

-post Permet de demander l'impression des états de postproduction (10A et 10B).

-summary

Permet de demander l'impression des états récapitulatifs (09A et 10A). Si les arguments **-summary** et **-detail** sont omis, les deux types d'état sont imprimés.

-detail Permet de demander l'impression des états détaillés (09B et 10B). Si les arguments **-summary** et **-detail** sont omis, les deux types d'état sont imprimés.

fichier_sym

Indique le nom du fichier de plan de production à partir duquel les états seront imprimés. Le fichier par défaut est Symnew dans le répertoire courant. Si le fichier ne se trouve pas dans le répertoire de travail en cours, vous devez ajouter le chemin absolu au nom du fichier.

fichier_journal

Spécifie le nom complet du fichier journal à partir duquel les rapports seront imprimés. Notez que les fichiers journaux du plan de production sont stockés dans le répertoire schedlog. Le fichier par défaut est celui du plan courant (fichier Symphony).

Si la commande est exécutée sans options, les deux états **pre** (09A et 09B) sont imprimés et les informations sont extraites du fichier Symphony.

Exemples

Imprimez l'état détaillé de préproduction à partir du fichier Symnew :

```
reptr -pre -detail
```

Imprimez l'état récapitulatif de préproduction à partir du fichier mysym :

```
reptr -pre -summary mysym
```

Imprimez l'état récapitulatif de postproduction à partir du fichier journal M199903170935 :

```
reptr -post -summary schedlog/M199903170935
```

Imprimez les états de préproduction à partir du fichier Symphony.

```
reptr
```

Lorsque les arguments sont spécifiés, les états de préproduction s'appuient sur les informations contenues dans le fichier Symnew alors que les états de postproduction s'appuient sur les informations contenues dans le fichier Symphony.

xref

Cette commande permet d'imprimer l'Etat 12 "Etat de références croisées".

Syntaxe

xref [-V|-U]

xref

[-cpu *poste_de_travail*]

[-depends|-files|-jobs|-prompts|-resource|-schedules|-when[...]]

Exécutez la commande à partir du répertoire *TWS_home*.

Arguments

-U Affiche des informations sur la syntaxe de la commande et quitte l'application.

-V Affiche la version de la commande et quitte l'application.

-cpu *poste_de_travail*

Permet d'imprimer l'état du poste de travail désigné. Le caractère générique @ étant autorisé, les informations de tous les postes de travail qualifiés sont prises en compte. Par défaut, tous les postes de travail sont pris en compte.

-depends

Permet d'imprimer un état détaillant les flots de travaux et les travaux avec leurs successeurs.

- files** Permet d'imprimer un état détaillant les flots de travaux et les travaux dépendants de chaque fichier.
- jobs** Permet d'imprimer un état détaillant les flots de travaux dans lesquels chaque travail est exécuté.
- prompts** Permet d'imprimer un état détaillant les flots de travaux et les travaux dépendants de chaque invite.
- resource** Permet d'imprimer un état détaillant les flots de travaux et les travaux dépendants de chaque ressource.
- schedules** Permet d'imprimer un état détaillant les flots de travaux et les travaux remplaçants de chaque flot de travaux.
- when** Permet d'imprimer un état détaillant les dates Include et Exclude des flots de travaux.

Si la commande est exécutée sans options, tous les postes de travail et toutes les options sont sélectionnés.

Exemples

Imprimez un état pour tous les postes de travail, qui détaille toutes les informations de référence croisée :

```
xref
```

Imprimez un état pour tous les postes de travail. Incluez les informations de référence croisée relatives à toutes les dépendances remplaçantes :

```
xref -cpu @ -depends -schedules
```

Exemples de sorties d'état

Etat 01 - Liste des caractéristiques des travaux :

```
TWS for UNIX (AIX)/REPORT1 8.3 (1.7)      ibm      Page    1
Etat 01      Liste détails travaux      03/06/06
```

```
Travail      : FTAWIN8+      #SCHEDDDD
Description :
Fichier JCL   : dir
Connexion    : maestro_adm
Créateur     : root
Travail de reprise :
Type de reprise : STOP
Invite de reprise :
Autodoc composer : Yes
Nombre total d'exécutions : 0 - 0 réussies, 0 abandonnées
```

```
      Ecoulé(mins) CPU(secs)
```

```
Total 00:00:00 0
Normale 00:00:00
Dernière exécution 00:00:00 0 (Sur à 00:00)
Maximum 00:00:00 0 (Sur )
Minimum 00:00:00 0 (Sur )
```

```
Travail      : MASTER8+      #JnextPlan
Description   : AJOUTE PAR LE PROGRAMME composer POUR LE CALENDRIER MASTER821#FINAL.
```

```

Fichier JCL      : /test/maestro_adm/tws/JnextPlan
Connexion       : maestro_adm
Créateur        : maestro_adm
Travail de reprise :
Type de reprise : STOP
Invite de reprise :
Autodoc composer : Yes
Nombre total d'exécutions : 11 - 11 réussies, 0 abandonnées

```

```

                        Ecoulé(mins) CPU(secs)
Total                00:00:14      44
Normale              00:00:01
Dernière exécution   00:00:01      4 (Le 03/05/06 à 23:16)
Maximum              00:00:02      4 (Le 03/04/06)
Minimum              00:00:01      4 (Le 03/04/06)

```

```

Travail          : MASTER8+   #JOB1
Description       : AJOUTE PAR LE PROGRAMME composer
Fichier JCL      : pwd
Connexion        : ^ACCLLOGIN^
Créateur         : root
Travail de reprise :
Type de reprise : STOP
Invite de reprise :
Autodoc composer : Yes
Nombre total d'exécutions : 1 - 1 réussies, 0 abandonnées

```

```

                        Ecoulé(mins) CPU(secs)
Total                00:00:01      0
Normale              00:00:01
Dernière exécution   00:00:01      0 (Le 03/05/06 à 22:22)
Maximum              00:00:01      0 (Le 03/05/06)
Minimum              00:00:01      0 (Le 03/05/06)

```

* * * * Fin de l'état * * * *

Dans la sortie, les valeurs définies dans la section «Travail», à la page 940 sont indiquées comme suit :

Autodoc composer

Indique si l'instruction du travail a été décrite dans la définition du flot de travaux à l'aide de l'interface de ligne de commande.

CPU (secs)

Représente la durée réelle, exprimée en secondes, pendant laquelle le travail a utilisé l'UC pour s'exécuter.

Total Représente le temps UC total enregistré pour le 'Nombre total d'exécutions'.

Normal

Représente la valeur moyenne de temps UC enregistrée au cours du 'Nombre total d'exécutions'.

Dernière exécution

Représente le temps UC enregistré au cours de la dernière exécution du travail.

Maximum

Représente la valeur maximale parmi les valeurs collectées pour le temps UC au cours du 'Nombre total d'exécutions' (calculé uniquement pour les travaux s'étant achevés correctement).

Minimum

Représente la valeur minimale parmi les valeurs collectées pour le temps UC au cours du 'Nombre total d'exécutions' (calculé uniquement pour les travaux s'étant achevés correctement).

Créateur

Représente le nom de l'utilisateur qui a créé la définition du travail.

Description

Correspond à la description textuelle du travail définie dans la zone **description** de l'instruction de définition du travail.

Ecoulé

Représente le temps écoulé, exprimé en minutes, qui inclut non seulement le temps pendant lequel le travail a utilisé l'UC, mais aussi le temps pendant lequel il a dû attendre que d'autres processus la libèrent.

Total Représente le temps total écoulé enregistré pour le 'Nombre total d'exécutions'.

Normal

Représente la valeur moyenne de temps écoulé enregistrée au cours du 'Nombre total d'exécutions'.

Dernière exécution

Représente le temps écoulé enregistré au cours de la dernière exécution du travail.

Maximum

Représente la valeur maximale parmi les valeurs collectées pour le temps écoulé au cours du 'Nombre total d'exécutions' (calculé uniquement pour les travaux s'étant achevés correctement).

Minimum

Représente la valeur minimale parmi les valeurs collectées pour le temps écoulé au cours du 'Nombre total d'exécutions' (calculé uniquement pour les travaux s'étant achevés correctement).

Remarque : Le temps écoulé affiché pour un travail reflet est le temps écoulé du travail distant lié.

Fichier JCL

Représente le nom du fichier défini dans la zone **scriptname** qui contient le script à exécuter, ou la commande spécifiée dans la zone **docommand** indiquant la commande à appeler lors de l'exécution du travail.

Travail

Identifiant du travail, [*poste_de_travail#*]*nom_travail*.

Connexion

Correspond au nom d'utilisateur spécifié dans la zone **streamlogon** et sous lequel le travail est exécuté.

Travail de reprise

Travail, spécifié par *after* [*poste_de_travail#*]*nom_travail*, exécuté en cas d'abandon du travail parent.

Invite de reprise

Représente le texte de l'invite, spécifié dans la zone **abendprompt**, qui s'affiche en cas d'abandon de ce travail.

Représente l'option de reprise définie dans la définition du travail. Elle peut avoir pour valeur **stop**, **continue** ou **rerun**.

TWS for UNIX (AIX)/REPORT2 8.3 (1.7) ibm Page 1
Etat 02 Liste des invites 03/06/06

```
PROMPT1 Répondez OUI dès que vous êtes prêt à exécuter les processus acc103 et acc104.
PROMPT2 Tous les utilisateurs se sont-ils déconnectés ?
CALLNO 555-0911
CALLOPER Appelez ^PERSON2CALL^ au ^CALLNO^ pour vérifier que toutes les cartes de présences ont été traitées.
PERSON2CALL Louis Dubois
```

* * * * Fin de l'état * * * *

TWS for UNIX (AIX)/REPORT3 8.3 (1.7) ibm Page 1
Etat 03 Liste des agendas des utilisateurs 03/06/06

Jan 2006							Fév 2006							Mars 2006						
Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam
						.			.	.	.	.				.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	
.	.	.	.	.	.	.	.	28	.	.	.	.		.	.	.	.	.	.	
.	31	.	.	.	.	.	.	.	.	.	.	.		.	31	.	.	.	.	

Avril 2006							Mai 2006					Juin 2006								
Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam
					.	.		.	.	.	.	.		.	.	.	.	.	.	.
.	.	.		.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	.
.	.	.		.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	.
.	.	.	.	.	.	30	.	.	31	.	.	.		.	.	.	.	.	.	.

Juil 2006							Août 2006							Sept 2006							
Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Lun	Mar	Mer	Jeu	Ven	Sam	
					.	.		.	.	.	.	.	.						.	.	.
.	.	.		.	.	.	.	.	.	.	.	.	.		.	.	.	.	.	.	.
.	.	.		.	.	.	.	.	.		.	.	.		.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.		31		.	.		.	.	.	.	.	.	.
31								.	.				.	.				30			

[illegible]

03/03/06	01:46	MASTER8+#JS1	1	4	SU
03/03/06	19:08	MASTER8+#JS2	1	4	SU
03/03/06	19:33	MASTER8+#JS3	1	4	SU
03/03/06	19:37	MASTER8+#JS4	1	4	SU
03/03/06	23:08	MASTER8+#JS5	2	4	SU
03/03/06	05:59	MASTER8+#JS_A	1	4	SU
03/05/06	05:59	MASTER8+#JS_G	1	4	SU
03/06/06	05:59	MASTER8+#JS_H	1	4	SU
03/06/06	21:57	MASTER8+#TIMEJ	2	4	SU
03/06/06	23:16	MASTER8+#SLEEPJ	1	4	SU

Travail :MASTER8+#JOB1 Exécutions : Abandonnées 0 Réussies 1 Temps écoulé : Normal 1 Min 1 Max 1

03/06/06	22:22	MASTER8+#JOBS	1	0	SU

* * * * Fin de l'état * * * *

L'Etat 7 lit les informations relatives à l'exécution du travail enregistrées dans la base de données et les affiche. Les différents états possibles sont les suivants :

AB pour les travaux qui ont échoué

SU pour les travaux qui se sont correctement terminés

DN pour les travaux soumis dont l'état est inconnu car ni un message de succès, ni un message d'échec n'ont encore été reçus.

Etat 08 - Histogramme des travaux :

TWS for UNIX (AIX)/REPORT8 8.3 (1.7)	ibm	Page 1
Etat 08	Histogramme des travaux 03/05/06 14:05 - 03/06/06 14:04	03/06/05

Intervalle par col. : 15 minutes

1	1	1	1	2	2	2	0	0	0	0	0	0	0	1	1	1
4	5	7	8	0	1	3	0	2	3	5	6	8	9	1	2	4
0	3	0	3	0	3	0	3	0	3	0	3	0	3	0	3	0
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	4

Nom du travail Etat
03/06/06

CF05066+.JnextPlan SU .*.

* * * * Fin de l'état * * * *

La sortie de l'Etat 8 indique les périodes au cours desquelles les travaux s'exécutent. Les chiffres fournis dans la partie supérieure de l'histogramme des travaux correspondent à des heures, écrites verticalement de haut en bas ; par exemple, la première colonne se lit 1405, ce qui signifie 14 heures 05. Les périodes pendant lesquelles le travail s'exécute sont marquées par des points et des astérisques lorsque la position du marqueur est alignée avec une heure écrite de haut en bas.

Etat 9B - Détail de la production planifiée :

TWS for UNIX (AIX) REPORTER 8.3 (1.7)	ibm	Page 1
Etat 09B Symnew	Détail de la production planifiée pour le 03/06/06	03/06/06

Temps UC			
Nom du travail	estimé	Prio	Heure début Jusqu'à Fréquence Limite Dépendances

```

Schedule NETAG #EXTERNAL      0
      E0000000      0
      Total      00:00
      Total      00:00

Schedule MYFTA #IWDSKE      10
      JOBIWD      10      23:00(03/06/06) 01:00
      Total      00:00

Schedule MYMST #TESTSKE 00:29 10
      TESTCRO+ 00:01 10
      NEWTEST 00:29 10 08:30(03/06/06)      TESTCROME
      Total      00:29

Calendrier MYMST #FINAL 00:00 10 05:59(03/07/06)
      JnextPlan 00:01 10
      Total      00:01
      Total      00:34

```

* * * * Fin de l'état * * * *

La sortie de l'Etat 9B montre ce qui doit être exécuté à la date sélectionnée dans l'environnement IBM Workload Scheduler. Les informations affichées proviennent des définitions stockées dans la base de données IBM Workload Scheduler. La sortie indique les flots de travaux dont l'exécution est prévue le 6 mars 2006, avec leur description, la liste des travaux qu'ils contiennent, les dépendances horaires, la fréquence de répétition, le nombre maximal de travaux (s'il est défini) et la dépendance par rapport à d'autres travaux ou flots de travaux. Par exemple, le flot de travaux nommé iwdske dont l'exécution est prévue sur MYFTA possède une dépendance follows sur le travail NETAG#EXTERNAL.E0000000 dont l'exécution est prévue sur l'agent réseau nommé NETAG.

La zone **Heure de début** dans la sortie des états générés par la commande **reptr** affiche :

Une restriction de temps définie dans la définition de flot de travaux à l'aide du mot clé at.

Si la date est comprise entre des parenthèses (), par exemple :

```

Heure de début
06:00(03/20/06)

```

L'heure d'exécution planifiée du flot de travaux dans la définition de flot de travaux à l'aide du mot clé schedtime.

Si la date est comprise entre des accolades {}, par exemple :

```

Heure de début
06:00{03/20/06}

```

L'heure de début réelle d'exécution du flot de travaux.

Si la date n'est ni entre parenthèses, ni entre accolades, par exemple :

```

Heure de début
06:00 03/20/06

```

Etat 10B - Détail de la production réelle :

```

TWS for UNIX (AIX) REPORTER 8.3 (1.7)      ibm      Page      1
Report 10B  Symphony      Actual Production Detail For 03/06/06      03/07/06

      Estimated      Actual      CPU      Job
      Job Name  Run Time Priority Start Time  Run Time Seconds Number  Status

Schedule NETAG #EXTERNAL      0      EXTRN
      E0000000      0      ERROR

```


	Total	00:00		00:00	0		
Schedule MYMST	#MONTHSKE	00:02	10	06:01(03/06/06)	00:03		SUCC
	GETLOGS	00:02	10	06:01(03/06/06)	00:03	#J11612	SUCC
	Total	00:02			00:03	0	
Schedule MYFTA	#IWSKE		10				HOLD
	JOBIWD		10				HOLD
	Total	00:00			00:00	0	
Schedule MYMST	#TESTSKE	00:29	10	06:01(03/06/06)	00:02		STUCK
	TESTCRO+	00:01	10	06:01(03/06/06)	00:02	#J11613	ABEND
	NEWTST	00:29	10				HOLD
	Total	00:30			00:02	0	
Schedule MYMST	#FINAL	00:01	10	05:59(03/07/06)			HOLD
	JnextPlan	00:01	10				HOLD
	Total	00:01			00:00	0	
	Total	01:38			00:09	0	

* * * * Fin de l'état * * * *

La sortie de l'Etat 10B indique les états des activités de planification en cours d'exécution sur le réseau IBM Workload Scheduler. Les informations affichées proviennent d'une copie du fichier Symphony qui est actuellement utilisé et mis à jour dans l'ensemble de l'environnement de planification. Ainsi, chaque fois que cette commande de génération d'état est exécutée au cours du traitement, les informations affichées reflètent le statut réel de l'activité planifiée.

Si vous comparez cette sortie à la sortie de l'Etat 9B, vous constatez que le flot de travaux MONTHSKE s'est exécuté pendant la journée de production en cours, le 6 mars, mais que son exécution n'est pas prévue le lendemain, 7 mars. Le flot de travaux EXTERNAL a échoué sur l'agent de réseau NETAG et, par conséquent, le flot de travaux IWSKE qui possède une dépendance de prédécesseur/successeur du flot de travaux EXTERNAL, reste à l'état HOLD.

Le flot de travaux TESTSKE se trouve à l'état STUCK, ce qui signifie que l'intervention de l'opérateur est requise du fait que, pendant le temps d'exécution du flot de travaux, le travail TESTCROME, après avoir démarré avec l'ID de travail J11613, a échoué à l'état ABEND et a entraîné le passage à l'état HOLD du travail dépendant NEWTEST.

Etat 11 - Calendrier de production planifiée :

TWS for UNIX (AIX)/REPORT11 8.3 (1.7) Page 1
Etat 11 Calendrier de production planifiée pour FEV 2006 03/08/065

Poste de travail : FTAWIN8+

Nomb. Temps UC	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Cal travaux estimé	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu
SCHED1	1	1			*																							

Un * entre le nom du calendrier et Nomb. travaux indique que le calendrier possède des travaux en cours d'exécution sur d'autres postes de travail.

Temps UC estimé par jour en secondes

Lun	Mar	Mer	Jeu	Ven	Sam	Dim
1	2	3	4	5	6	
0	0	0	0	0	0	0

7	8	9	10	11	12	13
0	0	0	0	1	0	0
14	15	16	17	18	19	20
0	0	0	0	0	0	0
21	22	23	24	25	26	27
0	0	0	0	0	0	0
28						
0						

TWS for UNIX (AIX)/REPORT11 8.3 (1.7) Page 2
Etat 11 Calendrier de production planifiée pour FEV 2006 03/08/06

Poste de travail : MASTER8+

	Nomb.	Temps	UC	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Cal	travaux	estimé	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	Ma	Me	Je	Ve	Sa	Di	Lu	
FINAL	1	4	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	

Un * entre le nom du calendrier et Nomb. travaux indique que le calendrier possède des travaux en cours d'exécution sur d'autres postes de travail.

Temps UC estimé par jour en secondes
Lun Mar Mer Jeu Ven Sam Dim

			1	2	3	4	5	6
			4	4	4	4	4	4
7	8	9	10	11	12	13		
4	4	4	4	4	4	4		
14	15	16	17	18	19	20		
4	4	4	4	4	4	4		
21	22	23	24	25	26	27		
4	4	4	4	4	4	4		
28								
4								

* * * * Fin de l'état * * * *

La sortie de l'Etat 11 indique le moment où l'exécution des flots de travaux est planifiée au cours du mois sélectionné. La première ligne contient le nombre de travaux que le flot de travaux contient, le temps UC estimé utilisé par le flot de travaux pour s'exécuter et le moment où l'exécution du flot de travaux est prévue. Le tableau indique pour chaque jour du mois sélectionné le temps UC estimé utilisé pour l'exécution de ce flot de travaux.

Etat 12 - Etat de références croisées :

La sortie de l'Etat 12 indique des informations différentes en fonction de l'indicateur utilisé lors de l'émission de la commande xref. Dans cette section, vous trouverez quelques exemples de sortie. Pour chacun de ces exemples, l'indicateur correspondant utilisé avec la commande xref est mis en évidence.

xref -when

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 1
Etat 12 Etat de références croisées pour les options ON, EXCEPT(*) et FREEDAYS(f). 03/08/06

Poste de travail : FTAHP

WHEN	Utilisé par les calendriers suivants :
REQUEST	TRFINAL

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 2
Etat 12 Etat de références croisées pour les options ON, EXCEPT(*) et FREEDAYS(f). 03/08/06

Poste de travail : FTAWIN8+

WHEN	Utilisé par les calendriers suivants :
MONTHEND	SCHED1
REQUEST	SCHED1 , SCHEDDAA

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 3
Etat 12 Etat de références croisées pour les options ON, EXCEPT(*) et FREEDAYS(f). 03/08/06

Poste de travail : MASTER8+

WHEN	Utilisé par les calendriers suivants :
EVERYDAY	FINAL
REQUEST	TMP

* * * * Fin de l'état * * * *

xref -jobs

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 4
Etat 12 Etat de références croisées pour les noms de travail. 03/08/06

Poste de travail : FTAWIN8+

Nom du travail	Existe dans les calendriers
SCHEDDDD	SCHED1

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 5
Etat 12 Etat de références croisées pour les noms de travail. 03/08/06

Poste de travail : MASTER8+

Nom du travail	Existe dans les calendriers
JnextPlan	FINAL
JOB1	TMP

* * * * Fin de l'état * * * *

xref -resource

TWS for UNIX (AIX)/CROSSREF 8.3 (1.7) ibm Page 8
Etat 12 Etat de références croisées pour les utilisateurs de ressources. 03/08/06

Poste de travail : FTAWIN8+

Ressource	Utilisée par :
QUKTAPES(N/F)	SCHED1

Poste de travail : MASTER8+

Ressource Utilisée par :
 TAPES(N/F) TMP

* * * * Fin de l'état * * * *

xref -prompts

Poste de travail : FTAWIN8+

Invite Utilisée par :

Texte défini SCHED1
 par l'utilisateur

Poste de travail : MASTER8+

Invite Utilisée par :
 BADEXIT FTAWIN8+#SCHED1
 GOODEXIT FTAWIN8+#SCHED1 , TMP

Texte défini TMP
 par l'utilisateur

* * * * Fin de l'état * * * *

xref -files

Poste de travail : MASTER8+

Nom du fichier Utilisé par :
 /root/MY_FILE.sh FTAWIN8+#SCHED1 , TMP

* * * * Fin de l'état * * * *

Programmes d'extraction d'états

Des programmes d'extraction de données sont utilisés pour générer plusieurs des états IBM Workload Scheduler. Les programmes sont répertoriés dans le tableau 121 :

Tableau 121. Programmes d'extraction d'états

Programme d'extraction d'états	Description
jbxtract	Permet de générer l'état 01 - Liste des caractéristiques des travaux et l'état 07 - Liste des historiques des travaux

Tableau 121. Programmes d'extraction d'états (suite)

Programme d'extraction d'états	Description
prxtract	Permet de générer l'état 02 - Liste des invites
caxtract	Permet de générer l'état 03 - Liste des agendas
paxtract	Permet de générer l'état 04A - Liste des paramètres
rextract	Permet de générer l'état 04B - Liste des ressources
r11xtr	Permet de générer l'état 11 - Calendrier de production planifiée
xrxtrct	Permet de générer l'état 12 - Etat de références croisées

La sortie des programmes d'extraction est contrôlée par la variable **MAESTRO_OUTPUT_STYLE** qui définit la façon dont les noms d'objet longs sont traités. Pour plus d'informations sur la variable **MAESTRO_OUTPUT_STYLE**, voir «Description des commandes», à la page 756.

jbxtract

Extrait de la base de données les informations relatives aux travaux.

Syntaxe

```
jbxtract [-V | -U]
          [-j travail]
          [-c poste_travail]
          [-o sortie]
```

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- j travail**
Indique le travail pour lequel l'extraction est effectuée. Par défaut, tous les travaux sont pris en compte.
- c poste_travail**
Indique le poste de travail des travaux pour lesquels l'extraction est effectuée. Par défaut, tous les postes de travail sont pris en compte.
- o sortie**
Indique le fichier de sortie. La valeur par défaut est **stdout**.

Résultats

La variable **MAESTRO_OUTPUT_STYLE** spécifie le style de sortie pour les noms d'objet longs. Avec la valeur **LONG**, des zones longues sont utilisées pour les noms d'objet. Si la valeur affectée à cette variable est différente de **LONG**, les noms longs sont tronqués à huit caractères suivis du signe plus. Par exemple : A1234567+.

Chaque enregistrement de travail contient des zones de longueur variable délimitées par des tabulations. Ces zones sont décrites dans le tableau 122, à la page 776.

Tableau 122. Zones de sortie Jbextract

Zone	Description	Longueur maximale (octets)
1	Nom du poste de travail	16
2	Nom du travail	16
3	Nom du fichier du script de travail	4096
4	Description du travail	65
5	Nom du travail de reprise	16
6	Option de reprise (0=stop, 1=rerun, 2=continue)	5
7	Texte de l'invite de reprise	64
8	Option d'auto-documentation (0=désactivée, 1=activée)	5
9	Nom de connexion de l'utilisateur pour le travail	36
10	ID utilisateur du créateur du travail	36
11	Nombre d'exécutions réussies	5
12	Nombre d'exécutions abandonnées	5
13	Durée totale de toutes les exécutions de travaux	8
14	Temps nécessaire au poste de travail pour toutes les exécutions des travaux	8
15	Temps moyen écoulé	8
16	Date de la dernière exécution (aammjj)	8
17	Heure de la dernière exécution (hhmm)	8
18	Secondes de la dernière exécution	8
19	Dernier temps écoulé	8
20	Durée (en secondes) de sollicitation du poste de travail	8
21	Temps écoulé maximal	8
22	Date d'exécution maximale (aammjj)	8
23	Sollicitation minimale du poste de travail en secondes	8
24	Temps minimal écoulé	8
25	Date d'exécution minimale (aammjj)	8

Remarque : Le temps écoulé affiché pour un travail reflet est le temps écoulé du travail distant lié.

Exemples

Pour extraire les informations sur le travail `mon_travail` du poste de travail principal et diriger les résultats dans le fichier `jinfo`, exécutez la commande suivante :

```
jbxtract -j mon_travail -c principal -o jinfo
```

prxtract

Extrait de la base de données les informations relatives aux invites.

Syntaxe

prxtract [-V | -U] [-o *sortie*]

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- o *sortie***
Indique le fichier de sortie. La valeur par défaut est **stdout**.

Résultats

Chaque enregistrement d'invite contient des zones de longueur variable délimitées par des tabulations. Ces zones sont décrites dans le tableau 123.

Tableau 123. Zones de sortie *Prxtract*

Zone	Description	Longueur maximale (octets)
1	Nom de l'invite	8
2	Valeur de l'invite	200

Exemples

Pour extraire les informations sur toutes les définitions d'invite et diriger les résultats vers le fichier `prinfo`, exécutez la commande suivante :

```
prxtract -o prinfo
```

caxtract

Extrait de la base de données les informations relatives aux agendas.

Syntaxe

caxtract [-V | -U] [-o *sortie*]

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- o *sortie***
Indique le fichier de sortie. La valeur par défaut est **stdout**.

Résultats

Chaque enregistrement d'agenda contient des zones de longueur variable, délimitées par des tabulations. Ces zones sont décrites dans le tableau 124, à la page 778.

Tableau 124. Zones de sortie Caxtract

Zone	Description	Longueur maximale (octets)
1	Nom de l'agenda	8
2	Description de l'agenda	64

Exemples

Pour extraire les informations sur toutes les définitions d'agenda et diriger les résultats vers le fichier cainfo, exécutez la commande suivante :

```
caxtract -o cainfo
```

paxtract

Permet d'extraire des informations relatives aux paramètres globaux (variables globales) de la base de données.

Syntaxe

```
paxtract [-V | -U] [-o sortie] [-a]
```

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- o *sortie***
Indique le fichier de sortie. La valeur par défaut est **stdout**.
- a** Permet d'afficher toutes les variables définies dans toutes les tables de variables. S'il n'est pas indiqué, seules les variables définies dans la table de variables par défaut s'affichent.

Résultats

Chaque enregistrement de variable contient des zones de longueur variable délimitées par des tabulations. Ces zones sont décrites dans le tableau 125.

Tableau 125. Zones de sortie Paxtract

Zone	Description	Longueur maximale (octets)
1	nom de table	80
2	nom de la variable	16
3	valeur de variable	72

A faire : Si vous ne spécifiez pas l'option **-a** (tout) dans la commande, seules les zones 2 et 3 s'affichent et les variables répertoriées sont celles uniquement contenues dans la table de variables par défaut.

Exemples

Pour extraire les informations relatives à toutes les définitions de variable et diriger les résultats vers le fichier `allvarinfo`, exécutez la commande suivante :

```
pextract -a -o allvarinfo
```

rextract

Extrait de la base de données les informations relatives aux ressources.

Syntaxe

```
rextract [-V | -U] [-o sortie]
```

Arguments

- V** Affiche la version de la commande et quitte l'application.
- U** Affiche des informations sur la syntaxe de la commande et quitte l'application.
- o sortie**
Indique le fichier de sortie. La valeur par défaut est **stdout**.

Résultats

Chaque enregistrement de ressource contient des zones de longueur variable délimitées par des tabulations. Ces zones sont décrites dans le tableau 126.

Tableau 126. Zones de sortie Rextract

Zone	Description	Longueur maximale (octets)
1	Nom du poste de travail	8/16
2	nom de la ressource	8
3	Unités de ressource totales	4
4	Description de la ressource	72

Exemples

Pour extraire les informations sur toutes les définitions de ressource et diriger les résultats vers le fichier `reinfo`, exécutez la commande suivante :

```
rextract -o reinfo
```

r11xtr

Extrait de la base de données les informations relatives aux flots de travaux.

Syntaxe

```
r11xtr [-V | -U]  
      [-m mm[aa]]  
      [-c poste_travail]  
      [-o sortie]  
      [-s nom_flot_données]
```

Arguments

- V Affiche la version du programme et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.
- m *mm[aa]*
Indique le mois (*mm*) et éventuellement l'année (*aa*) des flots de travaux.
La valeur par défaut est le mois et l'année en cours.
- c *poste_travail*
Indique le poste de travail à prendre en compte dans les états. Par défaut, tous les postes de travail sont pris en compte.
- s *nom_flot_travaux*
Indique le nom du flot de travaux dans lequel les travaux s'exécutent. Par défaut, tous les flots de travaux sont pris en compte.
- o *sortie*
Indique le fichier de sortie. La valeur par défaut est **stdout**.

Résultats

La variable **MAESTRO_OUTPUT_STYLE** spécifie le style de sortie pour les noms d'objet longs. Avec la valeur **LONG**, des zones longues sont utilisées pour les noms d'objet. Si la valeur affectée à cette variable est différente de **LONG**, les noms longs sont tronqués à huit caractères suivis du signe plus. Par exemple : A1234567+.

Chaque enregistrement de flot de travaux contient des zones de longueur variable délimitées par des tabulations. Ces zones sont décrites dans le tableau 127.

Tableau 127. Zones de sortie R11xtr

Zone	Description	Longueur maximale (octets)
1	Nom du poste de travail	16
2	nom du flot de travaux	16
3	Date du flot de travaux (aammjj)	6
4	Durée estimée (en secondes) de sollicitation du poste de travail	6
5	Option plusieurs postes de travail (* signifie que certains travaux sont exécutés sur d'autres postes de travail)	1
6	Nombre de travaux	4
7	Jour de la semaine (Di, Lu, Ma, Me, Je, Ve, Sa)	2

Exemples

Pour extraire les informations sur les flots de travaux remontant à juin 2004 sur le poste de travail principal, exécutez la commande suivante :

```
r11xtr -m 0604 -c principal
```

Pour extraire les informations sur tous les flots de travaux remontant au mois de juin sur tous les postes de travail et diriger le résultat vers le fichier r11out, exécutez la commande suivante :

```
rllxtr -m 06 -o rllout
```

xrxtrct

Extrait de la base de données les informations relatives aux références croisées.

Syntaxe

```
xrxtrct [-V | -U]
```

Arguments

- V Affiche la version de la commande et quitte l'application.
- U Affiche des informations sur la syntaxe de la commande et quitte l'application.

Résultats

La variable *MAESTRO_OUTPUT_STYLE* spécifie le style de sortie pour les noms d'objet longs. Avec la valeur **LONG**, des zones longues sont utilisées pour les noms d'objet. Si la valeur affectée à cette variable est différente de **LONG**, les noms longs sont tronqués à huit caractères suivis du signe plus. Par exemple : A1234567+.

Le résultat de la commande est écrit dans huit fichiers, **xdep_job**, **xdep_sched**, **xfile**, **xjob**, **xprompt**, **xresources**, **xsched** et **xwhen**. Ces fichiers sont enregistrés dans le répertoire de travail en cours. Pour exécuter la commande, vous devez disposer du droit en écriture et des droits d'exécution sur ce répertoire.

Exemples

Pour extraire les informations relatives à toutes les références croisées, exécutez la commande suivante :

```
xrxtrct
```

Fichier xdep_job

Le fichier **xdep_job** contient deux types d'enregistrement. Le premier enregistrement contient les informations sur les travaux et les flots de travaux qui dépendent d'un travail. Chaque enregistrement d'un travail et d'un flot de travaux dépendant contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 128.

Tableau 128. Zones de sortie Xdep_job

Zone	Description	Longueur (octets)
1	03	2
2	Nom du poste de travail	16
3	Nom du travail	40
4	nom du flot de travaux	16
5	Non utilisée	240
6	Nom du poste de travail du flot de travaux dépendant	16
7	Nom du flot de travaux dépendant	16
8	Nom du poste de travail du travail dépendant	16

Tableau 128. Zones de sortie Xdep_job (suite)

Zone	Description	Longueur (octets)
9	Nom du travail dépendant	40
10	Non utilisée	6
11	Non utilisée	6
12	Non utilisée	8
13	Fin d'enregistrement (null)	1

Le deuxième type d'enregistrement contient des informations sur les travaux et les flots de travaux soumis aux interdépendances de réseaux interconnectés. Chaque enregistrement de travail et de flot de travaux contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 129.

Tableau 129. Zones de sortie Xdep_job (suite)

Zone	Description	Longueur (octets)
1	08	2
2	Nom du poste de travail	16
3	Nom du travail	120
4	Non utilisée	128
5	Nom du poste de travail du flot de travaux dépendant	16
6	Nom du flot de travaux dépendant	16
7	Nom du poste de travail du travail dépendant	16
8	Nom du travail dépendant	40
9	Non utilisée	6
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xdep_sched

Le fichier **xdep_sched** contient des informations sur les travaux et les flots de travaux qui dépendent d'un flot de travaux. Chaque enregistrement de travail ou de flot de travaux contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 130.

Tableau 130. Zones de sortie Xdep_sched

Zone	Description	Longueur (octets)
1	02	2
2	Nom du poste de travail	16
3	nom du flot de travaux	16
4	Non utilisée	248
5	Nom du poste de travail du flot de travaux dépendant	16
6	Nom du flot de travaux dépendant	16
7	Nom du poste de travail du travail dépendant	16
8	Nom du travail dépendant	40
9	Non utilisée	6

Tableau 130. Zones de sortie Xdep_sched (suite)

Zone	Description	Longueur (octets)
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xfile

Le fichier **xfile** contient des informations sur les travaux et les flots de travaux qui dépendent d'un fichier. Chaque enregistrement contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 131.

Tableau 131. Zones de sortie Xfile

Zone	Description	Longueur (octets)
1	07	2
2	Nom du poste de travail	16
3	nom de fichier	256
4	Nom du poste de travail du flot de travaux dépendant	16
5	Nom du flot de travaux dépendant	16
6	Nom du poste de travail du travail dépendant	16
7	Nom du travail dépendant	40
8	Non utilisée	6
9	Non utilisée	6
10	Non utilisée	8
11	Fin d'enregistrement (null)	1

Fichier xjob

Le fichier **xjob** contient des informations sur les flots de travaux dans lesquels figure chaque travail. Chaque enregistrement de travail contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 132.

Tableau 132. Zones de sortie Xjob

Zone	Description	Longueur (octets)
1	04	2
2	Nom du poste de travail	16
3	Nom du travail	40
4	Non utilisée	248
5	Nom du poste de travail des flots de travaux	16
6	nom du flot de travaux	16
7	Non utilisée	8
8	Non utilisée	8
9	Non utilisée	6
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xprompt

Le fichier **xprompt** contient des informations sur les travaux et les flots de travaux qui dépendent d'une invite. Chaque enregistrement d'invite contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 133.

Tableau 133. Zones de sortie Xprompts

Zone	Description	Longueur (octets)
1	05	2
2	Nom du poste de travail	16
3	Nom ou texte de l'invite	20
4	Non utilisée	236
5	Nom du poste de travail du flot de travaux dépendant	16
6	Nom du flot de travaux dépendant	16
7	Nom du poste de travail du travail dépendant	16
8	Nom du travail dépendant	40
9	Non utilisée	6
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xresource

Le fichier **xresource** contient des informations sur les travaux et les flots de travaux qui dépendent d'une ressource. Chaque enregistrement de ressource contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 134.

Tableau 134. Zones de sortie Xresource

Zone	Description	Longueur (octets)
1	06	2
2	Nom du poste de travail	16
3	nom de la ressource	8
4	Non utilisée	248
5	Nom du poste de travail du flot de travaux dépendant	16
6	Nom du flot de travaux dépendant	16
7	Nom du poste de travail du travail dépendant	16
8	Nom du travail dépendant	40
9	Unités allouées	6
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xsched

Le fichier **xsched** contient des informations sur les flots de travaux. Chaque enregistrement de flot de travaux contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 135, à la page 785.

Tableau 135. Zones de sortie Xsched

Zone	Description	Longueur (octets)
1	00	2
2	Nom du poste de travail	16
3	nom du flot de travaux	16
4	Non utilisée	248
5	Nom du poste de travail (identique à 2 ci-dessus)	16
6	Nom du flot de travaux (identique à 3 ci-dessus)	16
7	Non utilisée	8
8	Non utilisée	8
9	Non utilisée	6
10	Non utilisée	6
11	Non utilisée	8
12	Fin d'enregistrement (null)	1

Fichier xwhen

Le fichier **xwhen** contient des informations sur le moment auquel les flots de travaux seront exécutés. Chaque enregistrement de flot de travaux contient des zones de longueur fixe sans délimiteur. Ces zones sont décrites dans le tableau 136.

Tableau 136. Zones de sortie Xwhen

Zone	Description	Longueur (octets)
1	01	2
2	Nom du poste de travail	16
3	Nom ou date ON/EXCEPT	8
4	Option except (*=EXCEPT)	1
5	Non utilisée	128
6	Nom du poste de travail	16
7	nom du flot de travaux	16
8	Non utilisée	8
9	Non utilisée	8
10	Non utilisée	6
11	Numéro de décalage	6
12	Unité de décalage	8
13	Fin d'enregistrement (null)	1

Exécution de rapports Dynamic Workload Console et de rapports de traitement par lots

Vous pouvez exécuter les rapports suivants à partir de Dynamic Workload Console :

Rapport d'historique d'exécution du travail

Rapport collectant les données historiques relatives à l'exécution des travaux pendant une période donnée. Il s'avère utile d'identifier les travaux qui se sont terminés avec une erreur ou les travaux en retard, ainsi que les

travaux critiques et promus et la dernière heure à laquelle le travail peut démarrer sans que le travail critique ne dépasse son échéance. Il contient également les travaux dont les délais n'ont pas été respectés ou qui ont duré plus longtemps que prévu, ainsi que les indicateurs de réexécution.

Rapport de statistiques d'exécution du travail

Rapport collectant les statistiques d'exécution du travail. Ce rapport permet de détecter les réussites, les taux d'erreur ; la durée minimale, maximale et moyenne ; les retards et les longues durées.

Rapport récapitulatif de la charge de travail du poste de travail

Rapport indiquant la charge de travail sur les postes de travail spécifiés. La charge de travail correspond au nombre de travaux qui se sont exécutés dessus. Ce rapport permet d'ajuster la planification de la capacité (modélisation de la charge de travail et réglage des postes de travail).

Rapport du temps d'exécution de la charge de travail du poste de travail

Rapport indiquant les heures et la durée d'exécution des travaux sur les postes de travail spécifiés. Ce rapport permet d'ajuster la planification de la capacité (modélisation de la charge de travail et réglage des postes de travail).

Rapport des détails de la production planifiée

Rapport basé sur les informations stockées dans un plan d'essai ou un plan prévisionnel. Les informations contenues dans ces plans proviennent de la base de données IBM Workload Scheduler. Un rapport détaillé de production planifiée peuvent être exécutés sur des moteurs distribués (gestionnaire de domaine maître et gestionnaire de domaine de sauvegarde). Un rapport de production réelle extrait d'un agent tolérant aux pannes peut contenir des informations différentes par rapport à un plan extrait d'un gestionnaire de domaine maître. Par exemple, pour un même nombre de travaux et de flots de travaux, le statut peut être différent car un travail exécuté correctement dans le domaine maître peut être suspendu ou prêt dans l'agent. Seul le taux de mise à jour du statut est identique sur l'agent en mode statut intégral qui s'exécute dans le domaine maître.

Rapport des détails de la production réelle

Rapport basé sur les informations stockées dans le plan en cours ou un plan archivé. Les informations contenues dans ces plans sont extraites des fichiers Symphony. Les rapports détaillés de production réelle peuvent être exécutés sur des moteurs distribués (gestionnaire de domaine maître, gestionnaire de domaine de sauvegarde, gestionnaire de domaine avec connecteur et agent tolérant aux pannes avec connecteur).

Rapport SQL personnalisé

Ce rapport vous permet de créer des rapports en exécutant vos propres requêtes SQL. Le rapport résultant contient un tableau comportant le nom de colonne spécifié dans la portion SELECT de l'instruction SQL. Les données sur lesquelles porte le rapport sont stockées dans une base de données relationnelle DB2 et résident du côté distribué. IBM Workload Scheduler for z/OS se connecte à la base de données au moyen de l'interface Java Database Connectivity (JDBC). Un pilote JDBC de type 4 est utilisé pour la connexion au système distant DB2 for LUW version 8.2, ou suivante.

Pour plus d'informations sur la définition et l'exécution de rapports depuis Dynamic Workload Console, voir Dynamic Workload Console - Guide d'utilisation (section sur la génération de rapports).

Certains de ces rapports sont également disponibles en tant que *rapports de traitement par lots* et peuvent être exécutés à partir d'une ligne de commande. Pour plus d'informations sur l'exécution des rapports de traitement par lots, voir «Exécution des rapports de traitement par lots à partir de l'interface de ligne de commande», à la page 792.

Selon l'interface à partir de laquelle vous exécutez le rapport ou le système d'exploitation du moteur, les formats de sortie suivants sont disponibles :

Tableau 137. Formats de sortie de rapport pris en charge

Nom du rapport	Formats de sortie pris en charge par Dynamic Workload Console	Formats de sortie pris en charge par les rapports de traitement par lots
Rapport d'historique d'exécution du travail	HTML, CSV Format de tableau uniquement	HTML, CSV, PDF Format de tableau uniquement
Rapport de statistiques d'exécution du travail	HTML, CSV Formats de tableau et de diagramme	HTML, CSV, PDF Formats de tableau et de diagramme
Rapport récapitulatif de la charge de travail du poste de travail	HTML, CSV Formats de tableau et de diagramme	HTML, CSV, PDF Formats de tableau et de diagramme
Rapport du temps d'exécution de la charge de travail du poste de travail	HTML, CSV Formats de tableau et de diagramme	HTML, CSV, PDF Formats de tableau et de diagramme
Rapport des détails de la production planifiée	XML, CSV Format de tableau uniquement	Indisponible
Rapport des détails de la production réelle	XML, CSV Format de tableau uniquement	Indisponible
Rapport SQL personnalisé	HTML, CSV Format de tableau uniquement	HTML, CSV, PDF Format de tableau uniquement
Rapport d'audit général (Pour plus d'informations, voir les informations relatives au suivi des modifications de base de données à l'aide des rapports d'audit dans <i>Guide d'administration</i>)	Indisponible	HTML, CSV, PDF Format de tableau uniquement
Rapport d'audit général (Pour plus d'informations, voir les informations relatives au suivi des modifications de base de données à l'aide des rapports d'audit dans <i>Guide d'administration</i>)	Indisponible	HTML, CSV, PDF Format de tableau uniquement

Pour pouvoir exécuter ces rapports, vous devez être titulaire des autorisations appropriées dans le fichier de sécurité des objets de rapport (accordées par défaut

à *utilisateur_tws* sur les nouvelles installations). Voir *Guide d'administration* pour obtenir des informations sur le fichier de sécurité.

Voir également le *Guide d'administration* pour apprendre à configurer Dynamic Workload Console afin de visualiser les rapports.

Rapports historiques

Le tableau ci-après récapitule les rapports historiques par :

- Fonctionnalité
- Critères de sélection
- Options du contenu de sortie

Tableau 138. Récapitulatif des rapports historiques

Nom du rapport	Description	Critères de sélection	Options du contenu de sortie
Historique d'exécution du travail	Correspond au Rapport 07. Collecte les données historiques relatives à l'exécution des travaux pendant un intervalle de temps donné. Permet d'obtenir les éléments suivants : • Travaux s'étant achevés avec des erreurs • Travaux en retard • Échéances dépassées • Longue durée • Indicateurs de réexécution • Autres informations historiques.	• Nom de travail, nom de flot de travaux, nom de poste de travail (flot de travaux). Chaque zone peut être spécifiée à l'aide d'un caractère générique. • Statut (Succès, Erreur, Inconnu) • Indicateurs de retard • Intervalle d'exécution du travail • Inclure/exclure les réexecutions	Vous pouvez sélectionner l'une des options suivantes : • Heure de début réelle • Durée estimée • Durée réelle • Numéro de travail • Démarré en retard (retard) • Arrêté en retard (retard) • Statut • Dernier démarrage critique • Critique • Promu • Longue durée • Nom de la définition du travail • Consommation CPU (indisponible sur les postes de travail Windows) • Connexion utilisateur • Type de réexécution • Numéro d'itération • Code retour La sortie est contenue dans la vue tabulaire.

Tableau 138. Récapitulatif des rapports historiques (suite)

Nom du rapport	Description	Critères de sélection	Options du contenu de sortie
Statistiques d'exécution du travail	Correspond au Rapport 01 Collecte les statistiques relatives à l'exécution des travaux. Permet d'obtenir les éléments suivants : • Taux de succès/erreur • Temps UC écoulé maximal et minimal • Durée moyenne • Statistiques d'exécution tardive et de longue durée Remarque : Ce rapport ne contient pas les travaux ayant été soumis via un nom d'alias.	• Nom de travail, nom de flot de travaux et nom de connexion utilisateur. Chaque zone peut être spécifiée à l'aide d'un caractère générique. • Pourcentage de travaux ayant le statut Succès, Erreur, Démarré en retard, Arrêté en retard et Longue durée • Nombre total d'exécutions et de réexecutions	Vous pouvez sélectionner l'une des options suivantes : • Détails sur le travail : – Connexion utilisateur – Créateur du travail – Description – Script – Informations de reprise • Statistiques du travail : – Total des exécutions (Exécutions terminées, Erreurs) – Total d'exceptions d'exécution (Démarré en retard, Terminé en retard, Longue durée) – Durées et temps UC minimal, maximal et moyen (pour les exécutions ayant abouti uniquement) – Consommation CPU (indisponible sur les postes de travail Windows) • Format du rapport : – Vue Graphiques – Vue Table – Inclure la table des matières par travail ou par poste de travail

Tableau 138. Récapitulatif des rapports historiques (suite)

Nom du rapport	Description	Critères de sélection	Options du contenu de sortie
Récapitulatif de la charge de travail du poste de travail	Fournit des données sur la charge de travail suivant le nombre de exécutés sur chaque poste de travail. Permet d'ajuster les prévisions de charge (modélisation de la charge de travail et optimisation des postes de travail).	- Noms des postes de travail. Chaque zone peut être spécifiée à l'aide d'un caractère générique. - Plages de dates ou jours spécifiques pour le filtrage de la charge de travail. - Intervalles de temps relatifs (permet de réutiliser la même tâche de rapport afin d'exécuter celui-ci quotidiennement et extraire le rapport correspondant à la production de la veille)	Vous pouvez sélectionner l'une des options suivantes : - Cohérence des informations sur les postes de travail organisées d'après les critères suivants : - Heure - Jour - Jour de production - Options de regroupement d'informations : - Fournit des informations récapitulatives cumulatives et agrégées, relatives à tous les postes de travail ou à un sous-ensemble défini de celles-ci - Format du rapport : - Vue Graphiques - Vue Table - Inclure la table des matières par date ou par poste de travail

Tableau 138. Récapitulatif des rapports historiques (suite)

Nom du rapport	Description	Critères de sélection	Options du contenu de sortie
Temps d'exécution de la charge de travail du poste de travail	Correspond au Rapport 08 . Fournit des données sur les exécutions de travaux (heure et durée) sur le poste de travail. Permet d'ajuster les prévisions de charge (modélisation de la charge de travail et optimisation des postes de travail).	- Noms de travail et de poste de travail. Chaque zone peut être spécifiée à l'aide d'un caractère générique. - Période d'exécution de la charge de travail - Intervalle de temps quotidien	Vous pouvez sélectionner l'une des options suivantes : - Informations regroupées par : - Poste de travail - Date d'exécution - Peut être trié par réexécution - Jour de production - Information du travail - Durée réelle - Statut - Réexécution - Nom de la définition du travail - Format du rapport : - Vue Graphiques - Vue Table
SQL personnalisé	Un assistant vous aide à définir vos requêtes SQL personnalisées (uniquement dans les vues de la base de données pour lesquelles vous disposez d'une autorisation d'accès).	Les critères indiqués dans la requête SQL personnalisée.	Le rapport résultant contient un tableau comportant le nom de colonne spécifié dans la portion SELECT de l'instruction SQL.

Rapports de production

Le tableau ci-après récapitule les rapports de production par :

- Fonctionnalité
- Critères de sélection
- Options du contenu de sortie

Tableau 139. Récapitulatif des rapports de production

Nom du rapport	Description	Critères de sélection	Options du contenu de sortie
Détail de la production réelle	Correspond au Rapport 10B . Fournit des données sur les plans en cours et archivés.	- Nom du travail - Nom du poste de travail (travail) - Nom du flot de travaux - Nom du poste de travail (flot de travaux)	Vous pouvez sélectionner l'une des options suivantes : - Format du rapport : - A plat - CSV - Microsoft Project - Inclut : - Prédécesseur de premier niveau - Journal du travail
Détail de la production planifiée	Correspond au Rapport 9B . Fournit des données sur les plans d'essai et les plans prévisionnels.	- Nom du travail - Nom du poste de travail (travail) - Nom du flot de travaux - Nom du poste de travail (flot de travaux)	Vous pouvez sélectionner l'une des options suivantes : - Format du rapport : - A plat - CSV - Microsoft Project - Inclut : - Prédécesseur de premier niveau - Journal du travail

Exécution des rapports de traitement par lots à partir de l'interface de ligne de commande

Cette section décrit comment exécuter les rapports répertoriés dans «Rapports historiques», à la page 788 à partir de la ligne de commande.

A l'aide d'une interface de ligne de commande, vous pouvez planifier ces rapports afin qu'ils s'exécutent en temps voulu.

Modèle de scénario métier

Pour éviter le ralentissement inattendu du traitement de la charge de travail, l'analyste d'une grande entreprise a besoin de rapports hebdomadaires collectant les informations d'historique sur la charge de travail traitée afin de déterminer et d'analyser les heures pleines éventuelles.

Pour répondre à cette demande, le TWSWEBUIDeveloper crée des rapports récapitulatifs de poste de travail de charge de travail (*Workload Workstation Summary Reports (WWS)*) et des rapports d'exécution de poste de travail de charge de travail (*Workload Workstation Runtimes Reports (WWR)*).

Pour accomplir cette tâche, il exécute les étapes suivantes :

1. Il personnalise les fichiers de propriétés associés aux rapports Workload Workstation Summary et Workload Workstation Runtimes, en spécifiant le format et le contenu de la sortie de rapport.

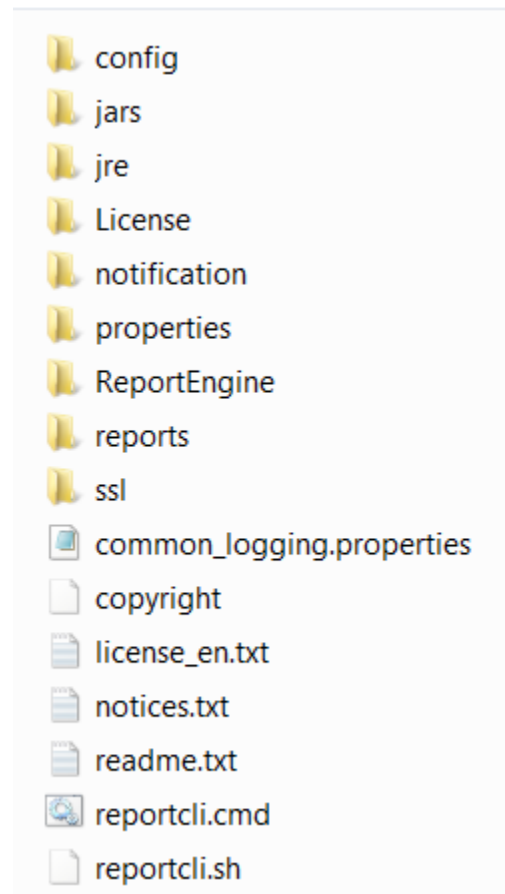
2. Il planifie les travaux pour obtenir des rapports WWS et WWR :
 - Le premier travail génère un rapport WWS à sauvegarder localement.
 - Le second travail génère un rapport WWR pendant la nuit sur les périodes horaires d'heures pleines de charge de travail. La sortie de rapport est envoyée à l'analyste par mail. Les informations collectées permettent d'optimiser l'équilibre de la charge de travail sur les systèmes.
3. Il ajoute les deux travaux à un flot de travaux planifié pour s'exécuter chaque semaine et ajoute le plan.

Configuration de la génération de rapport de traitement par lots de ligne de commande

Avant d'exécuter des rapports de traitement par lots, vous devez exécuter quelques étapes de configuration :

1. Le logiciel nécessaire pour exécuter des rapports de traitement par lots est contenu dans un module nommé `TWSBatchReportCli`, contenu dans l'image d'installation IBM Workload Scheduler, dans le répertoire `TWSBatchReportCli`. Si vous prévoyez d'exécuter des rapports de traitement par lots à partir d'un travail planifié, procédez à l'extraction du fichier de module sur l'un des systèmes d'exploitation répertoriés dans Document relatif à la configuration requise, à l'adresse <http://www-01.ibm.com/support/docview.wss?rs=672&uid=swg27048859>.

Après avoir décompressé le module, vous disposerez de la structure de fichiers suivante :



L'utilitaire natif d'archivage sur bande UNIX ne prenant pas en charge les longs noms de fichier, veillez à ce que la version GNU la plus récente d'archivage sur bande (gtar) soit installée si vous souhaitez extraire avec succès les fichiers sur AIX, Solaris ou les systèmes HP-UX.

Remarque :

- a. Vérifiez que vous exécutez les commandes suivantes dans le répertoire où vous avez extrait les fichiers :

Sur UNIX

```
chmod -R +x *
chown -R nom_utilisateur *
```

Sur Windows

Vérifiez qu'IBM Workload Scheduler est installé.

```
setown -u nom_utilisateur *
```

où *nom_utilisateur* est l'utilisateur IBM Workload Scheduler qui exécutera les rapports.

- b. Si vous prévoyez de planifier des travaux qui exécutent des rapports de traitement par lots, le système sur lequel vous extrayez le module doit être accessible en tant que système de fichiers réseau à partir d'un agent tolérant aux pannes défini dans l'environnement de planification local.
2. Si vous utilisez une base de données Oracle, téléchargez les pilotes JDBC nécessaires pour votre version du serveur Oracle.
3. Copiez les pilotes JDBC dans les répertoires *report_cli_installation_dir\jars* et *report_cli_installation_dir\ReportEngine\plugins\org.eclipse.birt.report.data.oda.jdbc_4.2.1.v20120820\drivers*. Le rapport de la ligne de commande découvre automatiquement les deux fichiers JAR.
4. Configurez le fichier modèle *.\config\common.properties* en indiquant les informations suivantes :
 - a. Si vous utilisez une base de données Oracle, connectez-vous comme suit à la base de données où sont stockées les données d'historique :
 - 1) Recherchez l'emplacement des pilotes JDBC. Cette information figure dans la propriété **com.ibm.tws.webui.oracleJdbcURL** du fichier *<TWA_home>\WAS\TWSprofile\properties\TWSConfig.properties*.
Pour plus d'informations sur ce fichier, voir la section relative à la configuration d'une base de données Oracle dans *IBM Workload Scheduler - Guide d'administration*.
 - 2) Indiquez l'emplacement des pilotes JDBC Oracle dans la propriété **PARAM_DataSourceUrl** du fichier *common.properties*.

Aucune personnalisation n'est requise si vous utilisez DB2.
 - b. Définir le format de date et d'heure, y compris le fuseau horaire. Le fichier *.\config\timezone.txt* contient une liste des fuseaux horaires pris en charge par IBM Workload Scheduler et des informations indiquant comment les définir. Les noms des fuseaux horaires sont sensibles à la casse.
 - c. Rendez le résultat de rapport disponible sur l'URL spécifiée dans la zone **ContextRootUrl**. Voici un exemple de paramètres de configuration :


```
#####
# Informations de serveur HTTP
#####

#Spécifiez la racine de contexte où le rapport sera disponible
#Pour tirer parti de cette fonction, il faut spécifier dans le répertoire de sortie de rapport
```


#le répertoire auquel fait référence votre serveur HTTP avec cette racine de contexte

```
ContextRootUrl=http://myserver/reportoutput
```

Dans ce cas, vérifiez que le *rép_rapport_sortie* spécifié lors de l'exécution de la commande des rapports de traitement par lots pointe vers le répertoire spécifié dans la zone **ContextRootUrl**.

- d. Envoyez la sortie de rapport à l'aide d'un courriel. Voici un exemple de paramètres de configuration :

```
#####
# Configuration du serveur de messagerie
#####
PARAM_SendReportByEmail=true

#SMTP server
mail.smtp.host=myhost.mydomain.com
#IMAP provider
mail.imap.socketFactory.fallback=false
mail.imap.port=993
mail.imap.socketFactory.port=993
#POP3 provider
mail.pop3.socketFactory.fallback=false
mail.pop3.port=995
mail.pop3.socketFactory.port=995

#####
# Propriétés de courriel
#####
PARAM_EmailFrom=user1@your_company.com
PARAM_EmailTo=user2@your_company.com,user3@your_company.com
PARAM_EmailCC=user4@your_company.com
PARAM_EmailBCC=user5@your_company.com
PARAM_EmailSubject=Test send report by email
PARAM_EmailBody=This is the report attached
```

Une explication de toutes les zones personnalisables est contenue dans le modèle de fichier.

Remarque : Si vous prévoyez d'exécuter les rapports Workstation Workload Runtime, vérifiez que le système de fichier dans lequel la base de données est installée possède assez d'espace libre. Si un manque d'espace disque se produit, une exception SQL du type suivant est déclenchée :

```
DB2 SQL error: SQLCODE: -968, SQLSTATE: 57011
```

Exécution des rapports de traitement par lots

Le répertoire `\reports\templates` contient un fichier modèle pour chaque type de rapport.

Avant d'exécuter l'un de ces rapports, vérifiez que vous personnalisez le fichier modèle correspondant.

Dans ce fichier appelé *nom_rapport.properties*, vous pouvez spécifier :

- Les informations à afficher dans l'en-tête du rapport.
- Comment filtrer les informations pour afficher le résultat attendu.
- Le format et le contenu de la sortie de rapport.

Pour plus d'informations sur les paramètres spécifiques, voir l'explication fournie dans le modèle de fichier à côté de chaque zone.

Si vous utilisez un jeu de caractères codé sur deux octets pour indiquer les paramètres dans le modèle de fichier de propriétés, veillez à enregistrer le fichier en codage UTF-8.

Après avoir configuré l'environnement tel qu'il est décrit dans «Configuration de la génération de rapport de traitement par lots de ligne de commande», à la page 793, et avoir configuré le modèle de fichier rapport, utilisez la syntaxe suivante pour exécuter le rapport :

```
reportcli -p nom_rapport.property
      [-o rép_rapport_sortie]
      [-r nom_sortie_rapport]
      [-k key=valeur ]
      [-k key=valeur ]
      .....
```

Où :

- p** *nom_rapport.property*
Indique le nom du chemin vers le fichier modèle de rapport.
- o** *rép_rapport_sortie*
Indique le répertoire de sortie du résultat de rapport.
- r** *nom_sortie_rapport*
Indique le nom de la sortie de rapport.
- k** *key=valeur*
Indique la valeur d'un paramètre. Cette valeur remplace la valeur correspondante, si elle est définie, dans le fichier `common.properties` ou dans le fichier *nom_rapport.properties*.

Exemples

1. Dans cet exemple, `reportcli.cmd` est exécuté avec le paramètre par défaut et génère le rapport `jrhl` :

```
reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\jrhl.properties
-r jrhl
```
2. Dans cet exemple, `reportcli.cmd` est exécuté avec le paramètre `-k` pour remplacer les valeurs définies pour **PARAM_DateFormat** dans le fichier `.\config\common.properties` et génère le rapport `jrhl` :

```
reportcli.cmd -p D:\ReportCLI\TWSReportCli\reports\templates\jrhl.properties
-r jrhl2 -k PARAM_DateFormat=short
```
3. Dans cet exemple, `reportcli.cmd` est exécuté à l'aide du paramètre `-k` pour remplacer le format spécifié pour la sortie de rapport dans le fichier `PROPERTIES` et génère le rapport `jrhl` :

```
./reportcli.sh -p /TWSReportCli/REPLI/reports/templates/wwr.properties
-r wwr3 -k REPORT_OUTPUT_FORMAT=html -k OutputView=charts
```
4. Procédez comme suit si vous souhaitez exécuter un rapport SQL personnalisé et rendre disponible le résultat du rapport à l'URL suivante `http://myserver/reportoutput/report1.html` :
 - a. Configurez le paramètre `ContextRootUrl` dans le fichier `common.properties` comme suit :

```
#####
# Informations de serveur HTTP
#####

#Spécifiez la racine de contexte où le rapport sera disponible
```

#Pour tirer parti de cette fonction, il faut spécifier dans le répertoire de sortie de rapport
#le répertoire auquel fait référence votre serveur HTTP avec cette racine de contexte

ContextRootUrl=http://myserver/reportoutput

- b. Lorsque vous utilisez une commande de rapports de traitement par lots, indiquez comme *rép_rapport_sortie* un répertoire qui pointe vers le répertoire indiqué dans ContextRootUrl. Par exemple, si vous avez mappé localement http://myserver/ comme pilote R:, vous pouvez exécuter la commande suivante :

```
reportclibat  
-p REPORT_CLI_DIR\reports\TWS\historical\templates\sql.properties  
-r report1  
-o R:\reportoutput
```

- c. Confirmant que l'exécution du rapport a réussi, le message suivant s'affiche :

```
AWSBRC0106I  
Rapport disponible sur : http://myserver/reportoutput/report1.html
```

Cette URL indique l'emplacement où la sortie de rapport est disponible.

Remarque : Si le rapport est exécuté via un travail IBM Workload Scheduler, le résultat de la commande s'affiche dans la sortie du rapport.

Journaux et traces des rapports de traitement par lots

Le fichier ./common_logging.properties contient les paramètres que vous pouvez utiliser pour configurer la fonction de trace et la journalisation.

Ce fichier contient les paramètres suivants :

```
logFileName=reportcli.log  
traceFileName=trace.log  
trace=off  
birt_trace=off
```

Où :

logFileName

Spécifie le nom du fichier contenant des informations génériques, un avertissement sur les problèmes potentiels et des informations sur les erreurs. Ce fichier est stocké dans ./log.

traceFileName

Spécifie le nom du fichier qui contient les traces. Si vous définissez trace=on, le fichier de trace est stocké dans ./log.

trace Spécifie d'activer ou non les traces. Activez les traces en définissant trace=on si vous souhaitez approfondir une enquête sur une erreur

birt_trace

Spécifie d'activer ou non les traces pour diagnostiquer les erreurs dans le moteur BIRT. Si vous définissez birt_trace=on, un fichier contenant la trace et nommé ReportEngine_aaaa_mm_dd_hh_mm_ss.log est stocké dans le dossier /ReportEngine/logs

Chapitre 19. Gestion des fuseaux horaires

IBM Workload Scheduler prend en charge plusieurs fuseaux horaires. Si vous activez les fuseaux horaires, vous pouvez gérer votre charge de travail dans un environnement à fuseaux horaires multiples.

Les notations à 3 caractères et à longueur variable sont prises en charge, mais il est conseillé d'utiliser les noms longs s'ils existent. Si vous planifiez dans les fuseaux horaires MST (Mountain Standard Time) ou EST (Eastern Standard Time), vous devez utiliser les noms longs, par exemple, "Amérique/New_York" pour EST. En effet, ces fuseaux horaires ne respectent pas les règles de l'heure d'été et le produit planifie des travaux ou des flots de travaux de façon incorrecte durant les heures d'été avec une heure de décalage pour les dépendances de temps.

La notation à 3 caractères est prise en charge pour la compatibilité avec les versions antérieures de IBM Workload Scheduler.

Le format de notation à longueur variable est de type région/ville, par exemple, Europe/Paris, qui est l'équivalent d'ECT (European Central Time).

Le chapitre est composé des sections suivantes :

- «Activation de la gestion des fuseaux horaires»
- «Comment IBM Workload Scheduler gère les fuseaux horaires», à la page 800
- «Passage à l'heure d'été», à la page 802
- «Désactivation de l'heure d'été», à la page 802
- «Règles générales», à la page 803

Activation de la gestion des fuseaux horaires

Vous pouvez activer ou désactiver la gestion des fuseaux horaires en modifiant le paramètre attribué à l'option globale *enTimeZone* sur le gestionnaire de domaine maître à l'aide de la ligne de commande **optman**. Ce paramètre prend effet après l'exécution suivante de **JnextPlan**. Vous trouverez ci-dessous les paramètres disponibles :

- Non** Désactivation de la gestion des fuseaux horaires. En d'autres termes, les valeurs attribuées à tous les mots clés **timezone** dans les définitions sont ignorées. Toutes les restrictions de temps définies par **at**, **until** et **deadline** sont gérées individuellement par chaque agent tolérant aux pannes, y compris les gestionnaires de domaine, en ignorant par conséquent le fuseau horaire de l'agent qui planifie le travail ou flot de travaux. Il en résulte que, lorsque plusieurs fuseaux horaires sont impliqués :
- Pour les travaux, des informations incorrectes sont indiquées sur ces dépendances horaires lorsqu'elles sont considérées du point de vue d'un agent autre que le propriétaire de travail. Toutefois, ceci n'a aucune incidence sur le traitement du travail.
 - Pour les flots de travaux, cela implique que chaque agent traite les dépendances horaires en fonction de son propre fuseau horaire, et donc à des heures différentes, ce qui entraîne l'exécution à des heures différentes des travaux appartenant à un même flot, mais définis sur des agents différents.

Yes Activation de la gestion des fuseaux horaires. En d'autres termes, les valeurs attribuées aux paramètres **timezone** sont utilisées pour calculer l'heure où les travaux et les flots de travaux s'exécuteront sur les postes de travail cibles.

Par défaut, la valeur **yes** est attribuée à l'option *enTimeZone*.

Pour plus d'informations sur l'utilisation de la ligne de commande **optman** pour gérer les options globales sur le gestionnaire de domaine maître, voir *IBM Workload Scheduler - Guide d'administration*.

Comment IBM Workload Scheduler gère les fuseaux horaires

Lorsque le fuseau horaire est activé, vous pouvez utiliser les paramètres de fuseau horaire sur les définitions du poste de travail, du travail et du flot de travaux.

En exécutant les activités de gestion de plan, IBM Workload Scheduler convertit la valeur définie pour les fuseaux horaires en définitions de l'objet. Les conversions sont appliquées dans cet ordre :

1. Lorsque les instances de flot de travaux sont ajoutées au plan de préproduction, le fuseau horaire figurant dans les définitions de flot de travaux est converti en fuseau horaire GMT et les dépendances de prédécesseur/successeur externes sont résolues.
2. Lorsque le plan de préproduction est créé ou étendu, les instances de flot de travaux sont affectées aux postes de travail sur lesquels l'instance est programmée pour s'exécuter et le fuseau horaire est converti de GMT vers le fuseau horaire figurant dans la définition du poste de travail cible.

C'est la raison pour laquelle si vous utilisez les commandes **conman showsched** ou **conman showjobs** pour afficher les informations sur les travaux et flots de travaux planifiés, vous verrez les valeurs de fuseau horaire exprimées avec le fuseau horaire du poste de travail sur lequel le travail ou le flot de travaux doit s'exécuter. Selon la configuration de l'option globale *enLegacyStartOfDayEvaluation*, vous pouvez choisir la manière dont le produit va gérer les fuseaux horaires lors du traitement, et plus précisément :

Si la valeur **no** est attribuée à *enLegacyStartOfDayEvaluation*

La valeur affectée à l'option *startOfDay* sur le gestionnaire de domaine maître n'est pas convertie au fuseau horaire local défini sur chaque poste de travail du réseau. Par conséquent, si l'option *startOfDay* est définie sur 0600 sur le gestionnaire de domaine maître, il est 0600 sur le fuseau horaire local défini sur chaque poste de travail du réseau. Ceci signifie également que le jour de traitement commence à la même heure, mais pas au même moment, sur tous les postes de travail.

La figure 30, à la page 801 montre comment le début de la journée défini sur 0600 sur le gestionnaire de domaine maître, est appliqué aux différents fuseaux horaires sur les deux agents tolérants aux pannes. La même conversion de temps est appliquée aux trois instances du flot de travaux **JS1** planifié pour s'exécuter sur les trois machines et contenant une dépendance de temps **at** sur le fuseau horaire 0745 US/Central. Le cadre temporel qui identifie le nouveau jour de traitement est en grisé dans la figure 30, à la page 801.

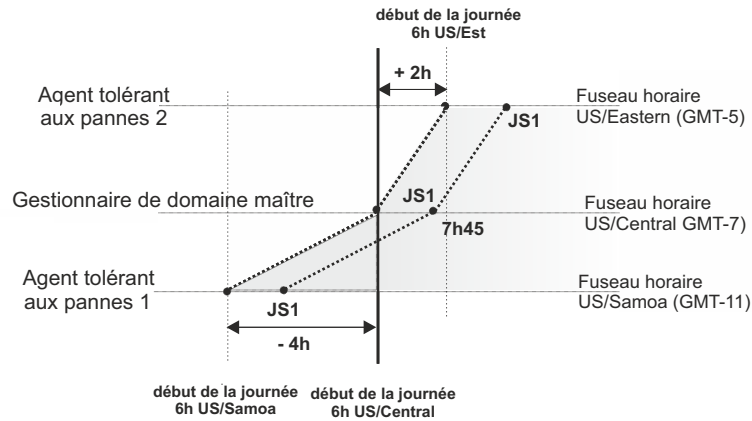


Figure 30. Exemple où la conversion de début de journée n'est pas appliquée

Si la valeur *yes* est attribuée à *enLegacyStartOfDayEvaluation*

La valeur affectée à l'option *startOfDay* sur le gestionnaire de domaine maître est convertie au fuseau horaire local défini sur chaque poste de travail du réseau. Ceci signifie que si l'option *startOfDay* est définie sur 0600 sur le gestionnaire de domaine maître, elle est convertie sur chaque poste de travail vers le temps correspondant du fuseau horaire local défini sur ce poste de travail. Ceci signifie également que le jour de planification commence au même moment, mais pas nécessairement à la même heure sur chaque poste de travail du réseau.

La figure 31 montre comment le début de la journée défini sur 0600 sur le gestionnaire de domaine maître, est appliqué aux différents fuseaux horaires sur les deux agents tolérants aux pannes. Cette figure montre également comment le délai des trois instances du flot de travaux **JS1** planifié pour s'exécuter sur les trois machines et contenant une dépendance de temps **at** sur le fuseau horaire 0745 US/Central n'est pas modifiée du fait de la conversion *startOfDay*. Le cadre temporel qui identifie le nouveau jour de traitement est en grisé dans la figure 31.

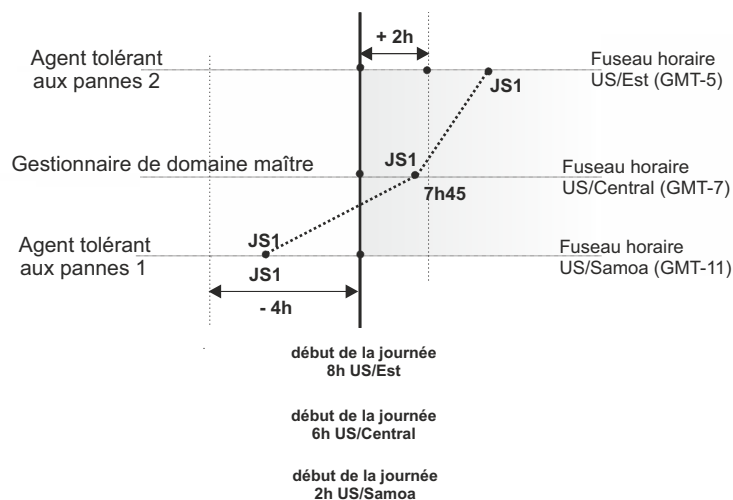


Figure 31. Exemple où la conversion de début de journée est appliquée

Remarque : A partir de la version 8.3 il n'y a pas de liaison entre le temps défini pour l'option *startOfDay* et le moment où **JnextPlan** est exécuté. **JnextPlan** peut

être exécuté à tout moment et l'option *startOfDay* indique uniquement le moment où le nouveau jour de traitement démarre.

Par défaut, la valeur **no** est attribuée à l'option globale *enLegacyStartOfDayEvaluation*.

Pour plus d'informations sur l'utilisation de la ligne de commande **optman** pour gérer les options globales sur le gestionnaire de domaine maître, voir *IBM Workload Scheduler - Guide d'administration*.

Passage à l'heure d'été

IBM Workload Scheduler gère le passage à l'heure d'été lors de la génération du plan de production. La date et l'heure d'exécution assignées aux travaux et flots de travaux du plan sont donc déjà converties vers la date et l'heure correspondant à l'heure d'été.

L'exemple suivant explique comment IBM Workload Scheduler applique la conversion de temps lorsque **JnextPlan** est exécuté pour générer ou étendre le plan de production pendant le passage à l'heure d'été.

Si l'heure d'été est activée à 3:00, tous les flots de travaux planifiés pour commencer entre 2:00 et 2:59 sont définis pour démarrer une heure plus tard. Par exemple, un travail défini pour s'exécuter à 2:30 du matin s'exécutera à 03:30.

A la date du passage à l'heure d'été, si le début de la journée coïncide avec l'heure manquante (par exemple de 00:00 à 00:59 heure de Sao Paulo ou de 02:00 à 02:59 heure de Chicago) tant la veille que le lendemain du passage à l'heure d'été, le plan est prolongé de 24 heures, ce qui reporte le démarrage du plan de production d'une heure par rapport à l'heure prévue. Pour éviter ce problème, modifiez manuellement la première extension de plan après le passage à l'heure d'été afin d'obtenir une prolongation de 23 heures au lieu de 24.

Désactivation de l'heure d'été

Pour la désactivation de l'heure d'été, l'horloge est reculée d'une heure par rapport à l'heure d'été. Pour maintenir la cohérence avec les critères de planification de production, IBM Workload Scheduler vérifie que les instances de flot de travaux planifiées pour s'exécuter pendant l'heure précédant le changement d'heure sont exécutées uniquement une fois. Parce que la conversion de temps est appliquée lors de la génération ou de l'extension du plan de production, la date et l'heure d'exécution affectées aux travaux et flots de travaux du plan sont déjà converties en date et heure correspondant à la désactivation de l'heure d'été.

Si un flot de travaux ou un travail s'exécute sur un fuseau horaire pour lequel l'heure d'été (DST) est désactivée, c'est-à-dire lorsque l'horloge est reculée d'une heure, et que vous définissez une dépendance à un autre fuseau horaire pour ces flux de travaux ou ces travaux, il est possible que cette dépendance horaire se produise lors du second intervalle de temps répété. Dans ce cas, la dépendance de temps sera résolue lors du premier intervalle de temps.

IBM Workload Scheduler reconnaît que la dépendance de temps se produit sur le second intervalle de temps répété et le résout en conséquence.

Règles générales

Lorsque le fuseau horaire est activé dans l'environnement IBM Workload Scheduler, quelle que soit la valeur définie pour l'option *enLegacyStartOfDayEvaluation*, certaines règles générales sont appliquées. Ces règles sont décrites ci-après, par sujet :

Identifier les paramètres de fuseau horaire par défaut pour les travaux et flots de travaux :

Vous pouvez définir un fuseau horaire dans une définition de flot de travaux pour l'ensemble du flot de travaux et pour les travaux contenus dans le flot de travaux. Ces fuseaux horaires peuvent être différents les uns des autres. Pour gérer tous les paramètres de fuseau horaire possibles la conversion de fuseau horaire est effectuée en respectant les critères suivants :

- Si aucun fuseau horaire n'est défini pour un travail contenu dans un flot de travaux, ce travail hérite du fuseau horaire défini sur le poste de travail où il est censé s'exécuter.
- Si aucun fuseau horaire n'est défini pour un flot de travaux, le fuseau horaire utilisé correspond à celui déjà défini sur le poste de travail où le flot de travaux est censé s'exécuter.
- Si aucun des fuseaux horaires mentionnés n'est défini, le fuseau horaire utilisé est celui défini sur le gestionnaire de domaine maître.

Choisir le fuseau horaire correct pour les postes de travail :

Pour éviter les incohérences, avant d'activer la fonction de gestion des fuseaux horaires sur le réseau IBM Workload Scheduler, vérifiez que, si un fuseau horaire est spécifié dans la définition de poste de travail, il est identique au fuseau horaire défini sur le système où est installé ce poste de travail.

Réglage de fuseau horaire par défaut pour le gestionnaire de domaine maître :

Si un fuseau horaire n'est pas défini dans la définition du gestionnaire de domaine maître, il hérite du fuseau horaire défini sur le système où est installé le gestionnaire de domaine maître. Pour voir quel fuseau horaire est défini sur le gestionnaire de domaine maître vous pouvez exécuter la commande suivante :

```
conman showcpu;info
```

Utilisation du fuseau horaire sur les agents étendus :

Les agents étendus héritent du fuseau horaire du gestionnaire de domaine maître.

Affichage du paramètre de fuseau horaire en production pour une dépendance de temps AT :

Si vous utilisez les commandes **conman**, **sj** ou **ss** pour afficher un travail ou un flot de travaux ayant une dépendance de temps **at** avec un fuseau horaire défini, l'heure spécifiée pour la dépendance **at** s'affiche avec le fuseau horaire défini sur le poste de travail où le travail ou le flot de travaux doit s'exécuter.

Appliquer un décalage à un fuseau horaire lors de la planification d'un flot de travaux :

Si vous soumettez en production un flot de travaux spécifiant une dépendance **at** avec un décalage de +n jour, IBM Workload Scheduler ajoute d'abord le décalage à la date puis convertit le fuseau horaire défini

dans la dépendance **at** . Ceci est important particulièrement en faisant référence à l'heure lors du passage à l'heure d'été.

A titre de meilleure pratique, si vous activez la gestion de fuseau horaire, définissez un fuseau horaire sur chaque poste de travail de votre réseau IBM Workload Scheduler.

Chapitre 20. Définitions des méthodes d'accès des agents

Les méthodes d'accès permettent d'étendre les fonctions de planification de travaux de IBM Workload Scheduler aux autres systèmes et applications. Elles s'exécutent sur les systèmes suivants :

Agents étendus

Postes de travail logiques associés à une méthode d'accès hébergée par un poste de travail IBM Workload Scheduler physique (pas pour un agent étendu). Plusieurs postes de travail d'agent étendu peuvent être hébergés par le même poste de travail IBM Workload Scheduler et utiliser la même méthode d'accès. L'agent étendu s'exécute sur des agents tolérants aux pannes définis via une définition de poste de travail IBM Workload Scheduler standard, qui lui donne un nom et identifie la méthode d'accès. La méthode d'accès est un programme exécuté par le poste de travail hôte chaque fois que IBM Workload Scheduler soumet un travail à un système externe.

Des travaux sont définis pour un agent étendu de la même manière que pour les autres postes de travail IBM Workload Scheduler, sauf que les attributs des travaux sont dictés par l'application ou le système externe.

Des informations sur l'exécution des travaux sont envoyées à IBM Workload Scheduler à partir d'un agent étendu à l'aide du fichier `stdlist` du travail. Un fichier d'options de méthode peut spécifier des connexions de remplacement pour lancer des travaux et vérifier les dépendances de fichier `opens`. Pour plus d'informations, voir *Guide d'utilisation et de référence*.

Un poste de travail physique peut héberger 255 agents étendus au maximum.

agents dynamiques et agents IBM Workload Scheduler for z/OS

Ils communiquent avec les systèmes externes pour démarrer le travail et renvoyer son statut. Pour exécuter des méthodes d'accès sur les applications externes à l'aide d'agents dynamiques, vous devez définir un travail de type **méthode d'accès**.

Des méthodes d'accès sont disponibles sur les systèmes et applications suivants.

- SAP R/3
- z/OS
- Méthodes personnalisées
- `unixssh`
- `unixrsh`
- UNIX en local (agents tolérants aux pannes uniquement)

Pour plus d'informations sur les méthodes d'accès UNIX incluses avec IBM Workload Scheduler, voir la section connexe du *Guide d'administration*.

Si vous utilisez agents dynamiques, pour plus d'informations sur la définition des postes de travail IBM Workload Scheduler, voir la section qui explique comment définir des postes de travail dans la base de données, *Guide d'utilisation et de référence*. Pour plus d'informations sur l'écriture de méthodes d'accès, voir la section relative à l'interface de méthode d'accès dans *Guide d'utilisation et de référence*.

Pour plus d'informations sur les méthodes d'accès, voir *Planification des applications avec IBM Workload Automation*.

Interface de méthode d'accès

L'interface entre IBM Workload Scheduler et une méthode d'accès est composée des informations transmises à la méthode par la ligne de commande et des messages renvoyés à IBM Workload Scheduler dans **stdout**.

Syntaxe de la ligne de commande des méthodes

L'hôte de IBM Workload Scheduler exécute une méthode d'accès à l'aide de la syntaxe de ligne de commande suivante :

nom_méthode **-t** *tâche* *options* **--** *chaîne_tâche*

Où :

nom_méthode

Désigne le nom de fichier de la méthode d'accès. Toutes les méthodes d'accès doivent figurer dans le répertoire : *racine_TWS/methods*

-t *tâche*

Désigne la tâche à effectuer, où *tâche* correspond à l'une des tâches suivantes :

LJ Lance un travail.

MJ Gère un travail lancé précédemment. Cette option permet de se resynchroniser si une tâche **LJ** précédente s'est arrêtée de manière inattendue.

CF Agents étendus uniquement. Vérifie la disponibilité d'un fichier. Cette option permet de vérifier les dépendances opens de fichier.

GS Agents étendus uniquement. Obtient le statut d'un travail. Cette option permet de vérifier les dépendances follows de fichier.

options Désigne les options associées à la tâche. Pour plus d'informations, voir «Options de tâche».

chaîne_tâche

Une chaîne comportant jusqu'à 255 caractères associée à la tâche. Voir «Options de tâche».

Options de tâche

Les options de tâche figurent dans le tableau 140. Un X signifie que l'option est admise pour la tâche.

Tableau 140. Options de tâche des commandes des méthodes

Tâche	-c	-n	-p	-r	-s	-d	-l	-o	-j	-q	-w	-S	Chaîne de la tâche
LJ	X	X	X	X	X	X	X	X	X			X	<i>chaîne_lj</i>
MJ	X	X	X	X	X	X	X	X	X				<i>chaîne_mj</i>
CF	X	X	X							X			<i>chaîne_cf</i>
GS	X	X	X	X		X					X		<i>chaîne_gs</i>

- c *agent_étendu,hôte,maître*
Indique les noms de l'agent, de l'hôte et du gestionnaire de domaine maître séparés par des virgules.
- n *nom_noeud*
Indique le nom de noeud de l'ordinateur associé à l'agent, le cas échéant. Ce nom est défini dans la zone **Noeud** de la définition de poste de travail de l'agent étendu.
- p *numéro_port*
Indique le numéro de port TCP/IP associé à l'agent, le cas échéant. Cette information est définie dans la zone d'**adresse TCP** de la définition de poste de travail d'agent.
- r *exécution_courante,exécution_spécifique*
Désigne le numéro d'exécution en cours de IBM Workload Scheduler et le numéro d'exécution spécifique associé au travail, séparés par une virgule. Les numéros d'exécution en cours et spécifiques peuvent être distincts si le travail a été reporté depuis une exécution antécédente.
- s *flot_travaux*
Désigne le nom du flot de travaux du travail.
- d *date_planification,période*
Désigne la date du flot de travaux (*aammjj*) ainsi que la période équivalente, séparées par une virgule.
- l *utilisateur*
Indique le nom d'utilisateur du travail. Ce nom est défini dans la zone **Connexion** de la définition du travail.
- o *liste_std*
Désigne le chemin d'accès complet du fichier de liste standard du travail. Toute sortie émanant de ce travail doit figurer dans ce fichier.
- j *nom_travail,ID*
Indique le nom du travail et l'identificateur unique affecté par IBM Workload Scheduler, séparés par une virgule. Ce nom est défini dans la zone **Nom du travail** de la définition du travail.
- q *qualifiant*
Désigne le qualifiant qui doit être utilisé dans une commande de test émise par une méthode pour un fichier.
- w *délai*
Indique le temps, exprimé en secondes, pendant lequel IBM Workload Scheduler attend une réponse sur un travail externe avant d'envoyer un signal SIGTERM à la méthode d'accès. La valeur par défaut est 300.
- S *nouveau_nom*
Indique que le travail est réexécuté à l'aide de ce nom, à la place du nom de travail d'origine. Dans le script d'un travail, vous pouvez utiliser la commande `jobinfo` pour renvoyer le nom du travail et exécuter le script différemment à chaque itération.
- *chaîne_lj*
Utilisé avec la tâche **LJ**. La chaîne provenant de la zone **Fichier Script** ou la zone **Commande** de la définition du travail.
- *chaîne_mj*
Utilisé avec la tâche **MJ**. Il s'agit des informations fournies à IBM Workload Scheduler par la méthode dans un message indiquant un changement

d'état du travail **%CJ** (pour plus d'informations sur les messages de ce type, voir «Messages de réponse des méthodes») suite à une tâche **LJ**. Généralement, elles identifient le travail qui a été lancé. Par exemple, une méthode UNIX peut fournir le processus d'identification (PID) du travail qu'elle a lancé et qui est ensuite envoyé par IBM Workload Scheduler en tant que partie d'une tâche **MJ**.

-- *chaîne_cf*

Utilisé avec la tâche **CF**. Pour une dépendance opens de fichier, correspond à la chaîne figurant dans la zone **Opens Files** de la définition du flot de travaux.

-- *chaîne_gs*

Utilisé avec la tâche **GS**. Désigne le travail dont le statut est vérifié. Le format est le suivant :

followsjob[,jobid]

où :

followsjob

Correspond à la chaîne figurant dans la liste **Follows Sched/Job** de la définition de flot de travaux.

id_travail

Correspond à un identificateur de travail facultatif reçu par IBM Workload Scheduler dans une réponse **%CJ** à une tâche **GS** précédente.

Messages de réponse des méthodes

Les méthodes renvoient des informations à IBM Workload Scheduler sous la forme de messages écrits dans **stdout**. Toutes les lignes commençant par le symbole de pourcentage (%) et finissant par une nouvelle ligne sont considérées comme un message. Les messages présentent le format suivant :

%CJ *statut* [*chaîne_mj* | *id_travail*]

%JS [*temps_UC*]

%RC *cr*

%UT [*message_erreur*]

Où :

CJ Change l'état du travail.

état L'état dans lequel le travail passe suite au changement. Tous les états des travaux IBM Workload Scheduler sont valides, à l'exception de **HOLD** et **READY**. Pour la tâche **GS**, les états suivants sont également valides :

ERROR

Une erreur s'est produite.

EXTRN

Le statut est inconnu.

chaîne_mj

Chaîne pouvant comporter jusqu'à 255 caractères, que IBM Workload Scheduler inclut dans toute tâche **MJ** associée au travail.

id_travail

Chaîne pouvant comporter jusqu'à 64 caractères, que IBM Workload Scheduler inclut dans toute tâche **GS** associée au travail.

JS [*temps_UC*]

Indique qu'un travail a abouti et fournit la durée de son exécution en secondes.

RC *rc*

rc est un nombre qui est interprété par IBM Workload Scheduler comme étant le code retour du travail de l'agent étendu. Le code retour est pris en compte seulement si la condition du code retour a été indiquée dans la définition du travail de l'agent étendu. Dans le cas contraire, il est ignoré et un message apparaît pour indiquer que le travail de l'agent étendu a abouti. **%JS** [*temps_poste_de_travail*]. De la même manière, si cette méthode n'envoie pas le message **%RC**, un message apparaît pour indiquer que le travail de l'agent étendu a abouti **%JS** [*temps_UC*].

UT [*message_erreur*]

Indique que la tâche requise n'est pas supportée par la méthode. Affiche une chaîne pouvant comporter jusqu'à 255 caractères, que IBM Workload Scheduler inclura dans son message d'erreur.

Fichier d'options de méthode

Pour les agents étendus, les agents et l'agent IBM Workload Scheduler for z/OS, vous pouvez, à l'aide d'un fichier d'options de méthode, fournir des informations de connexion et indiquer d'autres options.

Un fichier d'options est un fichier texte situé dans le répertoire `methods` de l'installation de IBM Workload Scheduler et contenant un ensemble d'options destinées à personnaliser le comportement de la méthode d'accès. Vous devez indiquer une option par ligne en respectant le format suivant (n'insérez pas d'espace) :

option=valeur

Toutes les méthodes d'accès utilisent deux types de fichier d'options :

Agents étendus

Fichier d'options globales

Fichier de configuration commun créé par défaut pour chaque méthode d'accès installée, dont les paramètres s'appliquent à tous les postes de travail d'agent étendu définis pour cette méthode. Lors de sa création, le fichier d'options globales contient uniquement l'option **LJuser**, qui correspond à l'ID utilisateur du système d'exploitation utilisé pour lancer la méthode d'accès. Vous pouvez personnaliser ce fichier en ajoutant les options correspondant à la méthode d'accès.

Fichier d'options locales

Fichier de configuration spécifique à chaque poste de travail d'agent étendu au sein de l'installation particulière d'une méthode d'accès. Ce fichier est nommé

NOM_AGENT_ETENDU_méthode_accès.opts, où :

NOM_AGENT_ETENDU

Correspond au nom du poste de travail d'agent étendu. La valeur de *NOM_AGENT_ETENDU* doit être indiquée en majuscules et en caractères alphanumériques. Le jeu de caractères codé sur deux octets, le jeu de caractères à simple octet et le texte bidirectionnel ne sont pas pris en charge.

méthode_accès

Correspond au nom de la méthode d'accès.

Si vous ne créez pas de fichier d'options locales, le système utilise par défaut le fichier d'options globales. Chaque poste de travail d'agent étendu (sauf pour z/OS) doit avoir un fichier d'options locales avec ses propres options de configuration.

Par exemple, si l'installation de la méthode d'accès inclut deux postes de travail d'agent étendu, CPU1 et CPU2, les fichiers d'options locales sont respectivement nommés *CPU1_méthode_accès.opts* et *CPU2_méthode_accès.opts*.

IBM Workload Scheduler lit le fichier d'options s'il existe avant d'exécuter une méthode d'accès. Pour les agents étendus, si le fichier d'options est modifié après le démarrage de IBM Workload Scheduler, les changements sont pris en compte uniquement lors de l'arrêt et du redémarrage du programme.

agents IBM Workload Scheduler for z/OS et agents

Fichier d'options globales

Fichier de configuration commun créé par défaut pour chaque méthode d'accès installée, dont les paramètres s'appliquent à tous les postes de travail d'agent définis pour cette méthode. Lors de sa création, le fichier d'options globales contient uniquement l'option **LJuser**, qui correspond à l'ID utilisateur du système d'exploitation utilisé pour exécuter la méthode d'accès. Vous pouvez personnaliser ce fichier en ajoutant les options correspondant à la méthode d'accès.

Le fichier d'options globales est nommé *méthode_accès.opts*, où *méthode_accès* est le nom de la méthode en cours de création.

Fichier d'options locales

Fichier de configuration spécifique à chaque méthode d'accès. Ce fichier est nommé *fichier_options_méthode_accès.opts*.

Dans un environnement distribué :

- Si vous définissez un travail pour exécuter la méthode d'accès via Dynamic Workload Console, il s'agit du fichier d'options que vous avez spécifié dans l'onglet de tâche d'agent étendu **Nouveau > Définition du travail > ERP > Méthode d'accès** Tâche XA.
- Si vous définissez la méthode d'accès via **composer**, il s'agit du fichier d'options que vous avez spécifié dans l'attribut **target** de la définition de travail.

Si vous ne créez pas de fichier d'options locales, le système utilise par défaut le fichier d'options globales.

Dans un environnement z/OS :

- Si vous définissez un travail pour exécuter la méthode d'accès via Dynamic Workload Console, il s'agit du fichier d'options que vous avez spécifié dans l'onglet de tâche d'agent étendu **Nouveau > ERP > Méthode d'accès** Tâche XA.
- Si vous définissez la méthode d'accès via l'instruction **JOBREC**, il s'agit du nom du poste de travail où s'exécute la méthode d'accès.

Si vous ne créez pas de fichier d'options locales, le système utilise par défaut le fichier d'options globales.

Si vous n'indiquez pas d'option dans le fichier *méthode_accès_fichier_options.opts*, le produit utilise la valeur spécifiée pour cette option dans le fichier d'options globales. Si vous ne les indiquez ni dans le fichier *méthode_accès_fichier_options.opts*, ni dans le fichier d'options globales, le produit émet un message d'erreur.

Le fichier d'options doit présenter le même nom de chemin que ses méthodes d'accès, avec l'extension de fichier *.opts*. Par exemple, le nom du chemin d'accès Windows au fichier d'options de la méthode *netmeth* est *racine_TWS\methods\netmeth.opts*

IBM Workload Scheduler lit le fichier d'options s'il existe avant d'exécuter une méthode d'accès.

Les options reconnues par IBM Workload Scheduler sont les suivantes :

LJuser=*nom_utilisateur*

Désigne la connexion à utiliser pour les tâches **LJ** et **MJ**. La valeur par défaut correspond à la connexion figurant dans la définition du travail. Pour plus d'informations, voir «Tâche de lancement d'un travail (LJ)», à la page 812 et «Tâche de gestion du travail (MJ)», à la page 813.

CFuser=*nom_utilisateur*

Agents étendus uniquement. Désigne la connexion à utiliser pour la tâche **CF**. La valeur par défaut pour UNIX est **root** et pour Windows, le nom d'utilisateur du compte dans lequel le produit a été installé. Voir *awsrccheckfiletask.dita*.

GSuser=*nom_utilisateur*

Désigne la connexion à utiliser pour les tâches **GS**. La valeur par défaut pour UNIX est **root** et pour Windows, le nom d'utilisateur du compte avec lequel IBM Workload Scheduler a été installé. Voir Tâche d'obtention d'un statut (GS) - agents étendus uniquement.

GStimeout=*secondes*

Indique le temps, exprimé en secondes, pendant lequel IBM Workload Scheduler attend une réponse avant d'arrêter la méthode d'accès. La valeur par défaut est **300** secondes.

nodename=*nom_noeud*

Indique le nom d'hôte ou l'adresse IP si la méthode en cours de définition l'exige. Pour la méthode d'accès **unixssh**, il s'agit du nom d'hôte ou de l'adresse IP utilisé(e) pour la connexion au moteur distant.

PortNumber=*numéro_port*

Indique le numéro de port si la méthode en cours de définition l'exige. Pour la méthode d'accès **unixssh**, il s'agit du port utilisé pour la connexion au moteur distant.

Pour les agents IBM Workload Scheduler for z/OS et les agents, vous pouvez indiquer le nom de noeud et le numéro de port également dans le fichier `JobManager.ini`.

Si vous ne les indiquez pas dans le fichier `méthode_accès_fichier_options.opts`, le produit utilise la valeur spécifiée dans le fichier d'options globales. Si vous ne les indiquez ni dans le fichier `méthode_accès_fichier_options.opts`, ni dans le fichier d'options globales, le produit utilise la valeur spécifiée dans la section `fichier_options` du fichier `JobManager.ini`.

Remarque : Si un hôte d'agent étendu est un ordinateur Windows, ces utilisateurs doivent être définis comme des objets utilisateur IBM Workload Scheduler.

Exécution des méthodes

Les sous-sections suivantes décrivent les échanges entre IBM Workload Scheduler et une méthode d'accès.

Tâche de lancement d'un travail (LJ)

La tâche **LJ** charge la méthode de l'agent étendu de lancer un travail sur une application ou un système externe. Avant d'exécuter la méthode, IBM Workload Scheduler crée un environnement d'exécution. Le paramètre **LJuser**, lu à partir du fichier d'options de la méthode, indique avec quel compte utilisateur la méthode doit être exécutée. Si le paramètre n'est pas présent ou que le fichier d'options n'existe pas, le compte utilisateur qui sera utilisé correspond à celui défini dans la zone **Connexion** de la définition du travail. En outre, les variables d'environnement suivantes sont définies :

HOME

Répertoire racine de l'utilisateur de la connexion.

LOGNAME

Nom d'utilisateur de la connexion.

PATH Pour UNIX, il s'agit du répertoire `/bin:/usr/bin`. Pour Windows, il s'agit du répertoire `%SYSTEM%\SYSTEM32`.

TWS_PROMOTED_JOB

Prend la valeur YES si le travail (un travail critique ou l'un de ses prédécesseurs) est promu.

TZ Fuseau horaire

Si la méthode ne peut pas être exécutée, l'état du travail devient FAIL.

Lorsqu'une méthode est en cours d'exécution, elle écrit des messages à son **stdout** pour indiquer le statut du travail sur le système externe. Les messages sont résumés dans le tableau 141, à la page 813.

Tableau 141. Messages de la tâche de lancement d'un travail (LJ)

Tâche	Réponse de la méthode	Action de IBM Workload Scheduler
LJ et MJ	%CJ <i>statut</i> [<i>chaîne_mj</i>]	Affecte l'état <i>état</i> au travail. Inclut <i>chaîne_mj</i> dans toutes les tâches MJ subséquentes.
	%JS [<i>temps_UC</i>]	Affecte l'état SUCC au travail.
	Exit code=non-zero	Affecte l'état ABEND au travail.
	%UT [<i>message_erreur</i>] et Exit code=2	Affecte l'état ABEND au travail et affiche <i>message_erreur</i> .

Une séquence typique est constituée d'un ou de plusieurs messages %CJ indiquant un changement dans l'état du travail, puis un message %JS avant que la méthode n'existe indiquant que le travail a abouti. Si le travail n'aboutit pas, la méthode doit exister sans écrire le message %JS. Une méthode qui ne supporte pas la tâche LJ écrit un message %UT vers **stdout** et renvoie le code de sortie 2.

Tâche de gestion du travail (MJ)

La tâche MJ assure la synchronisation avec un travail précédemment lancé dans le cas où IBM Workload Scheduler détermine que la tâche LJ s'est achevée de façon inattendue. IBM Workload Scheduler configure l'environnement de la même façon que pour la tâche LJ et lui transmet *chaîne_mj*. Pour plus d'informations, voir «Tâche de lancement d'un travail (LJ)», à la page 812.

Si la méthode localise le travail indiqué, elle renvoie les mêmes messages qu'une tâche LJ. Si la méthode ne parvient pas à localiser le travail, elle quitte l'application avec un code de sortie différent de zéro. IBM Workload Scheduler affecte alors l'état ABEND au travail.

Suppression d'un travail

Lors de l'exécution d'une tâche LJ ou MJ, la méthode doit capter un signal SIGTERM (signal 15). Le signal est envoyé lorsqu'un opérateur lance une commande **kill** à partir du gestionnaire de console de IBM Workload Scheduler. A la réception du signal, la méthode doit essayer de supprimer (**kill**) le travail et sortir sans qu'un message %JS ne soit généré.

Tâche de vérification d'un fichier (CF) - agents étendus uniquement

La tâche CF charge la méthode de l'agent étendu de vérifier la disponibilité d'un fichier sur le système externe. Avant d'exécuter la méthode, IBM Workload Scheduler crée un environnement d'exécution. Le paramètre **CFuser**, lu à partir du fichier d'options de la méthode, indique avec quel compte utilisateur la méthode doit être exécutée. Si le paramètre n'est pas présent ou si le fichier des options n'existe pas, sous UNIX, l'utilisateur **root** est utilisé, et sous Windows, le nom d'utilisateur du compte dans lequel IBM Workload Scheduler a été installé est utilisé. Si la méthode ne peut pas être exécutée, la dépendance opens du fichier est marquée comme échouée. En fait, le programme définit le statut du fichier sur **NO** et ne peut pas exécuter les travaux ou flots de travaux dépendants.

Pendant la phase d'exécution, la méthode exécute une commande test ou son équivalent sur le fichier utilisant le qualifiant qui lui a été transmis via l'option de ligne de commande **-q**. Si la valeur du fichier test est true, la méthode existe avec

un code de sortie 0. Si la valeur du fichier test est false, la méthode existe avec un code de sortie différent de 0. Pour plus d'informations, voir tableau 142.

Tableau 142. Messages de la tâche de vérification d'un fichier (CF)

Tâche	Réponse de la méthode	Action de IBM Workload Scheduler
CF	Exit code=0	Définit le statut du fichier à YES .
	Exit code=nonzero	Définit le statut du fichier à NO .
	%UT [message_erreur] et Exit code=2	Définit le statut du fichier à NO .

Une méthode qui ne supporte pas la tâche **CF** écrit un message %UT vers **stdout** et sort en renvoyant le code de sortie 2.

Tâche d'obtention d'un statut (GS) - agents étendus uniquement

La tâche **GS** charge la méthode de l'agent étendu de vérifier le statut d'un travail. Cette opération est nécessaire lorsque l'exécution d'un autre travail dépend du succès de l'exécution d'un travail externe. Avant l'exécution de la méthode, le paramètre **GSuser** est lu à partir du fichier d'options de la méthode pour déterminer le compte utilisateur avec lequel la méthode doit être exécutée. Si le paramètre n'est pas présent ou si le fichier des options n'existe pas, sous UNIX, l'utilisateur **root** est utilisé, et sous Windows, le nom d'utilisateur du compte dans lequel IBM Workload Scheduler a été installé est utilisé. Si la méthode ne peut pas être lancée, le travail ou le flot de travaux dépendant ne peut pas être exécuté. Si l'*id_travail* d'une précédente tâche **GS** est disponible, il est transmis à la méthode.

La méthode vérifie le statut du travail indiqué et le renvoie dans un message %CJ écrit vers **stdout**. Elle existe alors avec le code de sortie 0. Conformément à la fréquence définie par l'option locale *bm check status*, la méthode est réexécutée avec une tâche **GS** jusqu'à ce que l'un des statuts de travail suivants soit renvoyé :

abend Le travail s'est arrêté de manière anormale.

succ Le travail a abouti.

cancel Le travail a été annulé.

done Le travail est terminé, sans que l'on sache s'il a abouti ou échoué.

fail Le travail n'a pas pu être exécuté.

erreur Une erreur s'est produit dans la méthode lors de la vérification du statut du travail.

extrn La vérification du travail a échoué ou le statut du travail n'a pas pu être déterminé.

Notez que **GStimeout** dans le fichier d'options de la méthode indique le temps pendant lequel IBM Workload Scheduler attend une réponse avant de supprimer la méthode. Pour plus d'informations, voir «Fichier d'options de méthode», à la page 809.

Les réponses de la méthode sont résumées dans le tableau 143, à la page 815 :

Tableau 143. Messages de la tâche d'obtention d'un état (GS)

Tâche	Réponse de la méthode	Action de IBM Workload Scheduler
GS	%CJ <i>statut</i> [<i>jobid</i>]	Définit l'état du travail à <i>statut</i> et inclut <i>id_travail</i> dans toute tâche GS suivante.
	%UT [<i>message_erreur</i>] et Exit code=2	L'état du travail est inchangé.

Une méthode qui ne prend pas en charge la tâche **GS** écrit un message %UT à **stdout** et sort avec le code de sortie 2.

Commande Cpuinfo pour agents étendus uniquement

La commande **cpuinfo** peut être utilisée dans une méthode d'accès pour renvoyer des informations à partir d'une définition de poste de travail. Pour plus d'informations sur la commande complète, voir «Commande Cpuinfo pour agents étendus uniquement».

Identification des incidents

Les sections suivantes décrivent comment identifier et déboguer des problèmes survenus au niveau d'agents étendus et de méthodes d'accès.

Messages d'erreur de la liste standard d'un travail

Tous les messages en sortie émanant d'une méthode d'accès, exceptés ceux commençant par un symbole de pourcentage (%), sont écrits dans le fichier de la liste standard (**stdlist**) du travail. Pour les tâches **GS** et **CF** qui ne sont pas associées aux travaux IBM Workload Scheduler, les messages sont écrits dans le fichier de liste standard IBM Workload Scheduler. Pour obtenir des informations relatives à tout problème, reportez-vous à ces fichiers.

Méthode non exécutable

Si une méthode d'accès ne peut pas être exécutée, les événements suivants se produisent :

- Pour des tâches **LJ** et **MJ**, le travail se voit affecter l'état FAIL.
- Pour la tâche **CF**, la dépendance de fichier n'est pas résolue et le travail dépendant reste à l'état HOLD.
- Pour la tâche **GS**, la dépendance de travail n'est pas résolue et le travail dépendant reste à l'état HOLD.

Pour plus d'informations, consultez la liste des fichiers de liste standard (**stdlist**) concernant le travail et IBM Workload Scheduler.

Messages du gestionnaire de console pour agents étendus uniquement

Ce message d'erreur est affiché si vous émettez une commande **start**, **stop**, **link**, ou **unlink** pour un agent étendu :

AWSBH058E La commande émise pour le poste de travail : *nom_poste_travail*, ne peut pas être exécutée parce que le poste de travail est un agent étendu, sur lequel la commande n'est pas prise en charge.

Messages des programmes Composer et Compiler pour agents étendus uniquement

Le programme génère les messages d'erreur suivants dès que le programme **composer** rencontre une syntaxe incorrecte dans une définition de poste de travail :

AWSDEM045E La définition du poste de travail contient une erreur. Le mot clé ACCESS n'a pas été suivi par une méthode valide. Les méthodes valides correspondent au nom d'un fichier dans le répertoire des méthodes *TWS_home/* (le fichier n'a pas besoin d'être présent lorsque la méthode d'accès est définie).

AWSDEM046E La définition du poste de travail contient une erreur. Le mot clé ACCESS a été indiqué plusieurs fois.

AWSDEM047E La définition du poste de travail contient une erreur. Le mot clé ACCESS n'a pas été suivi par une méthode valide. Les méthodes valides correspondent au nom d'un fichier dans le répertoire des méthodes *TWS_home/* (le fichier n'a pas besoin d'être présent lorsque la méthode d'accès est définie).

Si un agent étendu est défini avec une méthode d'accès mais sans hôte, le message suivant s'affiche :

AWSBIA140E Pour un agent étendu, vous devez spécifier l'hôte et la méthode d'accès.

Messages de Jobman pour agents étendus uniquement

Pour les agents étendus, les messages d'information, d'avertissement ou d'erreur sont consignés dans le fichier **stdlist** de **jobman**.

Un travail dont le lancement a abouti génère le message suivant :

AWSBDW019I Launched job *nom_travail*, #Jnum_exécution pour l'utilisateur *ID_utilisateur*.

Un travail dont le lancement n'a pas abouti génère le message suivant :

AWSBDW057E Le travail *job_name* n'a pas été lancé pour le motif suivant :
message_erreur

L'échec d'une tâche de vérification de fichier génère le message suivant :

AWSBDW062E Jobman n'a pas réussi à appeler le fichier de méthode suivant *nom_méthode* pour l'agent étendu. L'erreur du système d'exploitation est :
erreur_système

L'échec d'une tâche de gestion de fichier génère le message suivant :

AWSBDW066E Planman a demandé à jobman d'exécuter une tâche qui n'est pas prise en charge sur l'agent ciblé. Le fichier d'options de méthode suivant a été utilisé :
fichier_options_méthode. L'identificateur de travail et le PID du moniteur sont les suivants : *travail*, #Jpid_moniteur

Lorsqu'une méthode envoie un message non reconnu à **jobman**, le message suivant est généré :

AWSBDW064E Un travail contrôlé par jobman a renvoyé le message non reconnu suivant : *message_incorrect*. L'identificateur de travail, le PID du moniteur et le fichier de méthode sont les suivants :
nom_travail, #Jpid_moniteur
utilisant le *fichier de méthode*.

Chapitre 21. Gestion des dépendances interréseaux

Les *dépendances interréseau* de IBM Workload Scheduler permettent aux travaux et aux flots de travaux d'un réseau local d'utiliser les travaux et flots de travaux d'un réseau distant sous la forme de dépendances de *prédécesseur/successeur*. Le présent chapitre explique comment personnaliser votre environnement afin de pouvoir définir les dépendances interréseau et la gestion des dépendances interréseaux.

Il comprend les sections suivantes :

- «Présentation sur les dépendances interréseaux»
- «Configuration d'un agent réseau», à la page 819
- «Définition d'une dépendance interréseau», à la page 821
- «Gestion des dépendances interréseau dans le plan», à la page 822
- «Dépendances interréseaux dans un environnement mixte», à la page 825

Remarque : Selon vos besoins et exigences, vous pouvez choisir entre des dépendances interréseau et des dépendances croisées pour établir une dépendance entre un travail s'exécutant sur le moteur local et un travail s'exécutant sur un moteur IBM Workload Scheduler distant. Voir «Définition de dépendances», à la page 23 pour une description des différences entre ces deux types de dépendances.

Présentation sur les dépendances interréseaux

Avant d'indiquer une dépendance interréseau, vous devez créer une définition de poste de travail pour l'*agent réseau*. Un agent réseau est un poste de travail IBM Workload Scheduler qui gère les dépendances de prédécesseur/successeur entre son réseau local et un réseau IBM Workload Scheduler distant.

Le réseau IBM Workload Scheduler local peut contenir plusieurs agents réseau, chacun représentant un réseau IBM Workload Scheduler distant spécifique où les travaux et les flots de travaux concernant les dépendances interréseaux définies en local sont spécifiées. Les dépendances interréseau sont affectées aux travaux et aux flots de travaux de la même façon que les dépendances de prédécesseur/successeur locales, si ce n'est que le nom de l'agent réseau est inclus afin d'identifier le travail ou le flot de travaux suivi.

Un flot de travaux spécial appelé **EXTERNAL** est automatiquement créé par IBM Workload Scheduler pour chaque agent réseau dans le réseau local. Il contient des travaux en guise de marques de réservation pour représenter chaque dépendance interréseau.

Un travail EXTERNAL est créé pour chaque dépendance interréseau appartenant aux flots de travaux dont le démarrage est planifié à des jours différents et à des dates de planification différentes. Ainsi, un travail EXTERNAL diffère d'un autre par :

- Le nom du fichier script qui identifie le travail ou le flot de travaux distant dont le travail ou le flot de travaux local dépend.
- La date prévue pour le démarrage du flot de travaux local contenant la dépendance interréseau. Si la dépendance est définie dans un travail du flot de travaux, la date prévue pour le démarrage du flot de travaux est prise en compte.

La vérification de la dépendance interréseau ne démarre pas tant que le flot de travaux ne correspond pas à sa dépendance horaire ou qu'il n'est pas libéré.

Prenons le cas de deux travaux appartenant à différents flots de travaux et désignant la même dépendance interréseau. Si l'un de leurs flots de travaux est libéré et si le travail démarre, la dépendance interréseau est vérifiée et libérée, le cas échéant. Dans ce cas, lorsque le deuxième travail commence à vérifier sa dépendance interréseau, il trouve que la dépendance est déjà résolue, mais pas obligatoirement le jour prévu. Si vous souhaitez éviter cette situation, vous devez réexécuter le travail représentant la dépendance interréseau une fois qu'elle a été résolue pour la première fois.

IBM Workload Scheduler vérifie le statut des travaux et flots de travaux concernés dans le réseau distant, puis mappe leur statut dans les travaux représentant les dépendances interréseaux dans le flot de travaux EXTERNAL. Le statut de ces travaux et flots de travaux est vérifié sur un intervalle de temps fixe jusqu'à ce que le travail ou le flot de travaux distant soit à l'état SUCC, CANCL ou ERROR.

Affichage d'une dépendance interréseau

La présente section décrit un exemple de scénario sur les dépendances interréseau et la liaison du travail représentant la dépendance interréseau au flot de travaux où elle est définie. Supposons que :

- Vous avez défini un flot de travaux nommé ELISCHE qui s'exécute sur le poste de travail TWS206 contenant un travail nommé ELI avec une dépendance interréseau à partir d'un flot de travaux TWS207#FINAL.MAKEPLAN qui s'exécute sur un réseau IBM Workload Scheduler différent.
- XA_MAST est l'agent réseau défini dans le réseau local pour gérer les dépendances interréseau à partir des travaux et des flots de travaux définis dans le réseau distant.

Utilisez la commande **conman sj** pour afficher l'ensemble des dépendances interréseaux :

```

                                (Est) (Est)
CPU  Schedule SchedTime Job State Pr Start Elapse RetCode Deps
TWS206#ELISCHE 0600 03/31 **** HOLD 10 (03/31)
      ELI  HOLD 10                                XA-MAST::"TWS207#MYJS.JOB1"
```

où (03/31) représente la restriction de temps **at** définie dans TWS206#ELISCHE. En commençant à partir de (03/31), le programme vérifie le statut de TWS207#MYJS.JOB1 dans le réseau distant pour savoir si la dépendance interréseau XA-MAST::"TWS207#MYJS.JOB1" est satisfaite.

Si vous exécutez la commande :

```
%sj XA-MAST#EXTERNAL;info
```

Le programme affiche la liste des travaux représentant les dépendances interréseaux définies dans les flots ou les flots de travaux s'exécutant dans le réseau local à partir des travaux et des flots de travaux définis dans le réseau distant accessible via l'agent réseau XA-MAST :

```

CPU  Schedule SchedTime Job  JobFile      Opt Job
Prompt
XA-MAST #EXTERNAL

      E8802332 TWS207#MYJS.JOB1
```


Vous pouvez afficher les détails sur le travail ou le flot de travaux dépendant de TWS207#MYJS.JOB1 dans la dépendance interrèseau représentée par le travail E8802332 dans le flot de travaux EXTERNAL en exécutant la commande suivante :

```
%sj @#EXTERNAL.E8802332;deps
```

La sortie montre le lien entre le travail dépendant et la dépendance interrèseau :

```

                                (Est) (Est)
CPU  Schedule SchedTime Job State Pr Start Elapse RetCode Deps
XA-MAST#EXTERNAL.E8802332 Les dépendances sont les suivantes :

TWS206#ELISCHE 0600 03/31 **** HOLD 10 (03/31)
                                ELI  HOLD 10                                XA-MAST::"TWS207#MYJS.JOB1"
```

La vérification de la dépendance interrèseau ne démarre pas tant que le flot de travaux TWS206#ELISCHE ne correspond pas à sa dépendance horaire (03/31) ou qu'il n'est pas libéré.

Si un autre travail est défini dans un autre flot de travaux dans le réseau local avec une dépendance sur TWS207#MYJS.JOB1 et si le démarrage du flot de travaux local est prévu le même jour, 31/03/06, la dépendance de cet autre travail sur TWS207#MYJS.JOB1 sera répertoriée dans le travail E8802332 dans le flot de travaux XA-MAST#EXTERNAL.

Configuration d'un agent réseau

Les postes de travail d'agent réseau sont définis comme des agents étendus et nécessitent un poste de travail physique hôte et une méthode d'accès. La méthode d'accès des agents réseau est nommée **netmth**.

Le processus **batchman** sur le gestionnaire de domaine maître demande **netmth** sur l'agent réseau à des intervalles de temps fixes pour obtenir le statut du travail ou du flot de travaux remplacé distant. Vous pouvez personnaliser l'intervalle de temps entre deux vérifications consécutives en définissant l'option globale *bm check status* dans le fichier *localopts* sur le gestionnaire de domaine maître. IBM Workload Scheduler continue à effectuer les vérifications jusqu'à ce que le travail ou le flot de travaux distant soit à l'état SUCC, CANCL ou ERROR.

Vous devez créer un fichier d'options nommé **netmth.opts** sur le poste de travail où l'agent de réseau s'exécute. L'utilisateur sous lequel la méthode d'accès s'exécute est défini dans ce fichier, ainsi que le délai d'attente d'une réponse de la méthode d'accès avant sa fermeture. Ce fichier d'options doit disposer du même chemin que la méthode d'accès :

```
racine_TWS/methods/netmth.opts
```

Le contenu du fichier *netmth.opts* possède la structure suivante :

```
GSuser=nom_connexion
GStimeout=secondes
```

où :

nom_connexion

Est le nom de connexion utilisé pour exécuter la méthode. Si l'hôte de l'agent réseau est un ordinateur Windows, cet utilisateur doit être défini dans IBM Workload Scheduler.

secondes

Est le nombre de secondes pendant lequel IBM Workload Scheduler attend une réponse avant d'arrêter la méthode d'accès. Le paramètre par défaut est 300 secondes. La prochaine fois que **batchman** doit vérifier le statut du travail ou du flot de travaux remplacé distant, la méthode d'accès démarre automatiquement.

Les modifications apportées à ce fichier ne prennent effet qu'une fois que vous avez arrêté puis redémarré IBM Workload Scheduler.

Exemple de définition d'un agent réseau

L'exemple suivant montre comment définir un poste de travail d'agent réseau pour un réseau distant, Réseau A, qui permet au réseau local, Réseau B, d'utiliser les travaux et flots de travaux du réseau distant sous la forme de dépendances interréseaux.

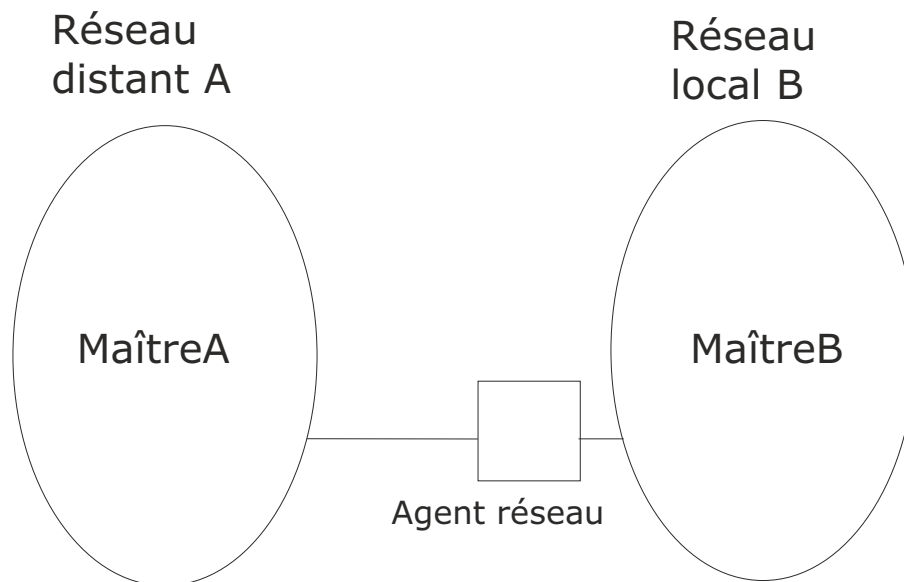


Figure 32. Réseaux local et distant

Supposons que :

- MasterA est le gestionnaire de domaine maître du réseau distant, Réseau A, et que :
 - **MasterA_tws** est l'utilisateur *utilisateur_TWS* défini sur MasterA.
 - Le numéro de port TCP de MasterA est 12345.
 - Le noeud sur lequel MasterA est défini est MasterA.rome.tivoli.com.
- MasterB est le gestionnaire de domaine maître du réseau local, Réseau B, et que :
 - **MasterB_tws** est l'utilisateur *utilisateur_TWS* défini sur MasterB.
 - Le noeud sur lequel MasterB est défini est MasterB.rome.tivoli.com.

Un poste de travail d'agent réseau nommé Agent réseau, défini sur MasterB pour gérer les dépendances interréseaux sur les travaux ou flots de travaux définis sur Réseau A peut être le suivant :

```
CPUNAME NETAGT
DESCRIPTION "AGENT RESEAU"
OS OTHER
NODE MASTERA.ROME.TIVOLI.COM
```

```
TCPADDR 12345
FOR maestro
HOST MASTERB
ACCESS netmth
FIN
```

Important : Saisissez le nom de l'accès réseau netmth en minuscule sur les systèmes d'exploitation sensibles à la casse.

Ce fichier d'options, **netmth.opts** défini sur MasterB peut être :

```
GSuser=maîtreB_tws
GStimeout=600
```

Remarque : L'agent réseau peut être défini sur le gestionnaire de domaine maître ou sur un agent tolérant aux pannes.

Définition d'une dépendance interréseau

La syntaxe utilisée pour définir une dépendance interréseau dans une définition de flot de travaux se présente comme suit :

follows nom_agent_réseau::poste_de_travail_distant#nom_flot_travaux(heure [date]).nom_travail

où (*heure [date]*) sont spécifiques au fuseau horaire utilisé sur le poste de travail du réseau distant auquel l'agent réseau est connecté ; dans notre exemple, le fuseau horaire de MasterA. Si (*heure [date]*) n'est pas spécifié dans cette syntaxe ou si plus d'un flot de travaux contient les mêmes valeurs (*heure[date]*), le premier flot de travaux trouvé est pris en compte.

Supposons que :

- calendrierA est un flot de travaux défini dans la base de données MasterA.
- travailA est un travail défini dans la base de données MasterA.
- calendrierB est un flot de travaux défini dans la base de données MasterB.
- travailB est un travail défini dans la base de données MasterB.

Vous pouvez définir des dépendances interréseaux à l'aide des instructions **follows** suivantes :

Pour définir une dépendance interréseau de calendrierB à partir de l'instance du flot de travaux calendrierA(1100)

Utilisez l'instruction suivante :

```
schedule calendrierB
on everyday
follows AgtRéseau::MasterA#calendrierA(1100)
:
end
```

Pour définir une dépendance interréseau de travailB à partir de travailA contenu dans l'instance de flot de travaux calendrierA(1100)

Utilisez l'instruction suivante :

```
schedule calendrierB
on everyday
:
travailB
follows AgtRéseau::MasterA#planA(1100).jobA
end
```

Vous pouvez également définir des dépendances interréseaux d'un travail sur un flot de travaux ou vice versa.

Gestion des dépendances interréseau dans le plan

Les dépendances interréseau sont gérées dans le plan à partir de la ligne de commande **conman** par gestion du flot de travaux EXTERNAL. Dans le flot de travaux EXTERNAL, les dépendances sont répertoriées en tant que travaux, indépendamment du fait qu'elles soient définies pour des travaux ou des flots de travaux. Il existe un flot de travaux EXTERNE pour chaque agent réseau du plan.

Dans le flot de travaux EXTERNAL, les noms de travaux uniques représentant les dépendances interréseau sont générés comme suit :

Ennnmmss

où :

nnn est un nombre aléatoire.

mm représente les minutes en cours.

ss représente les secondes en cours.

Le nom réel du travail ou du flot de travaux est enregistré dans la spécification des fichiers script de l'enregistrement de travail.

Remarque : Les travaux et les flots de travaux distants sont définis et exécutés sur leur réseau local de façon standard. Qu'ils soient utilisés comme dépendances interréseau n'a aucun effet sur leur comportement local.

Etats des travaux définis dans le flot de travaux EXTERNAL

Le statut des travaux définis dans le flot de travaux EXTERNAL est déterminé par la méthode d'accès et répertorié dans la zone Statut de libération du flot de travaux EXTERNAL. Le statut indiqué se rapporte à la dernière heure où le réseau distant a été vérifié. Les travaux peuvent donner l'impression de sauter des états lorsque les états changent entre deux vérifications.

Tous les états des travaux ou flots de travaux, sauf FENCE, sont répertoriés. Outre ces états, il existe trois états spécifiques aux travaux EXTERNAL :

CANCL

Le travail ou flot de travaux correspondant dans le réseau distant a été annulé.

ERROR

Une erreur est survenue lors de la vérification du statut distant.

EXTRN

Il s'agit de l'état initial. Si le travail est introuvable dans le réseau distant, il reste à l'état EXTRN.

Remarque : Vous pouvez annuler dans le réseau local les instances de travaux ou de flots de travaux dépendants de la même instance d'un travail ou d'un flot de travaux défini dans le réseau distant. Dans ce cas, veuillez aussi à annuler manuellement le travail représentant cette dépendance interréseau dans le flot de travaux EXTERNAL pour éviter les reports continus du flot de travaux EXTERNAL. La même remarque s'applique lorsque le flot de travaux local dépendant du travail ou du flot de travaux défini dans le réseau distant n'est pas reporté vers le nouveau plan.

Gestion des travaux définis dans le flot de travaux EXTERNAL

Vous pouvez effectuer les actions ci-dessous sur des travaux dans un flot de travaux EXTERNAL :

Annuler

Annule le travail EXTERNAL en libérant la dépendance interréseau de tous les travaux ou flots de travaux locaux. La vérification du statut de la dépendance prend fin.

Rerun Demande à **conman** de redémarrer la vérification du statut du travail EXTERNE. Après toute exécution de la commande **rerun**, le travail passe immédiatement à l'état EXTRN.

La réexécution est intéressante pour les travaux EXTERNAL ayant l'état ERROR. Par exemple, si un travail EXTERNAL ne peut pas être lancé parce que la méthode d'accès réseau n'accorde pas l'autorisation d'exécution, le travail acquiert l'état ERROR et son statut cesse d'être vérifié. Lorsque vous aurez corrigé les autorisations, la méthode pourra démarrer, mais **conman** ne démarrera la vérification du statut du travail EXTERNE que lorsque vous aurez effectué une réexécution (**rerun**).

Confirm SUCC / ABEND

Définit le statut du travail EXTERNAL sur SUCC ou ABEND en libérant la dépendance de tous les travaux et flots de travaux locaux dépendants. La vérification du statut de la dépendance prend fin.

Remarque : Aucune de ces commandes n'a d'effet sur le travail ou sur le flot de travaux distant dans le réseau distant. Elles permettent simplement de manipuler les dépendances du réseau local.

Exemples de scénarios de gestion des dépendances interréseaux

La présente section fournit des exemples de scénarios décrivant la gestion des dépendances interréseau dans la production à l'aide des commandes de la ligne de commande **conman**.

Supposons que vous avez déjà défini ce qui suit :

- Un poste de travail local appelé `local1`
- Un flot de travaux défini pour le poste de travail local `local1` appelé `plan1`
- Un travail défini dans `local1#plan1` appelé `travail1`
- Un agent réseau appelé `agtréseau` défini dans le réseau local pour gérer la dépendance interréseau à partir des travaux et des flots de travaux définis dans le réseau distant.
- Un poste de travail dans le réseau distant appelé `distant1`
- Un flot de travaux défini pour le poste de travail distant `distant1` appelé `pland`
- Un travail situé dans `distant1#pland` appelé `travaild`

Vous pouvez effectuer plusieurs opérations à partir de la ligne de commande **conman** dans le réseau local. La liste suivante contient plusieurs exemples :

Ajout d'une dépendance interréseau d'un travail distant à un travail local.

Par exemple, pour ajouter le travail distant `travaild` en tant que dépendance interréseau pour `travail1`, exécutez la commande suivante :

```
adj local1#plan1.travail1;follows=agtréseau::distant1#pland.travaild
```

Ajout d'une dépendance interréseau d'un flot de travaux distant à un flot de travaux local.

Par exemple, pour ajouter le flot de travaux distant `pland` en tant que dépendance interréseau pour le flot de travaux `plan1`, exécutez la commande suivante :

```
ads local1#plan1;follows=agtréseau::distant1#pland
```

Annulation des dépendances interréseau gérées par un agent réseau.

Par exemple, pour annuler tous les travaux EXTERNAL d'un agent réseau `agtréseau`, exécutez l'une des deux commandes suivantes :

```
cj netagt#EXTERNAL.@
```

```
cj agtréseau::@
```

Confirmation de la terminaison correcte de la dépendance interréseau.

Par exemple, pour confirmer que le travail distant `distant1#pland.travaild` s'est correctement terminé et libérer ainsi la dépendance interréseau correspondante, exécutez la commande suivante :

```
confirm agtréseau::distant1#pland.travaild;succ
```

Suppression d'une dépendance interréseau d'un travail vers un autre travail.

Par exemple, pour supprimer la dépendance interréseau du travail distant `distant1#pland.travaild` vers le travail local `local1#plan1.travail1`, exécutez la commande suivante :

```
ddj local1#plan1.travail1;follows=agtréseau::distant1#pland.travaild
```

Suppression d'une dépendance interréseau d'un travail vers un flot de travaux.

Par exemple, pour supprimer la dépendance interréseau du travail distant `distant1#pland.travaild` vers le flot de travaux local `local1#plan1`, exécutez la commande suivante :

```
dds local1#plan1;follows=agtréseau::distant1#pland.travaild
```

Libération d'un travail local à partir d'une dépendance interréseau d'un travail distant.

Par exemple, pour libérer un travail à partir d'une dépendance interréseau d'un travail distant, exécutez la commande suivante :

```
rj
```

```
local1#plan1.travail1;follows=agtréseau::distant1#pland.travaild
```

Libération d'un flot de travaux local à partir d'une dépendance interréseau d'un travail distant.

Par exemple, pour libérer un flot de travaux à partir d'une dépendance interréseau d'un travail distant, exécutez la commande suivante :

```
rs
```

```
local1#plan1;follows=agtréseau::distant1#pland.travaild
```

Réexécution d'un travail dans le flot de travaux EXTERNAL pour redémarrer la vérification d'une dépendance.

Par exemple, pour réexécuter un travail appartenant au flot de travaux EXTERNAL afin de redémarrer la vérification de la dépendance interréseau à partir du travail distant `distant1#pland.travaild`, exécutez l'une des deux commandes suivantes :

```
rr agtréseau#EXTERNAL.travaild
```

```
rr agtréseau::distant1#pland.travaild
```

Affichage des dépendances interréseaux à partir des travaux ou flots de travaux définis dans un réseau distant.

Par exemple, pour afficher toutes les dépendances interréseau définies d'un agent réseau avec leurs noms d'origine et leurs noms de travaux générés, exécutez la commande suivante :

```
sj netagt#EXTERNAL.@";info
```

Soumission d'un travail avec une dépendance interréseau à partir d'un flot de travaux défini dans un réseau distant

Par exemple, pour soumettre une commande `rm` dans le flot de travaux `JOBS` avec une dépendance interréseau sur un flot de travaux distant, exécutez la commande suivante :

```
sbd "rm afile";follows=agtréseau::distant1#pland
```

Dépendances interréseaux dans un environnement mixte

Le tableau 144 montre la configuration prise en charge pour les dépendances interréseaux définies dans un environnement mixte de version 8.3. Les termes clés du tableau sont les suivants :

Net_A Agent de réseau défini sur le réseau local.

Wks_B

Poste de travail sur le réseau distant auquel l'agent de réseau `Net_A` est connecté. `Wks_B` représente le poste de travail qui identifie et contrôle l'état du travail ou du flot de travaux distant spécifié dans la dépendance interréseau.

Sym_A

Fichier Symphony traité sur le réseau local.

Sym_B

Fichier Symphony traité sur le réseau distant.

niveau antérieur

Version 8.1, 8.2 ou 8.2.1

Tableau 144. Dépendances interréseaux dans un environnement mixte

	Niveau ant. Net_A Niveau ant. Sym_A	Net_A 8.3 Niveau ant. Sym_A	Niveau ant. Net_A Sym_A 8.3	Net_A 8.3 Sym_A 8.3
Niveau ant. Wks_B Niveau ant. Sym_B	Ce n'est pas un environnement mixte de la version 8.3.	Net_A envoie les informations à Wks_B comme si les deux versions étaient identiques à Wks_B.	Net_A envoie les informations à Wks_B au format 8.1, 8.2 ou 8.2.1. L'utilisation du mot clé <i>schedtime</i> dans la définition du travail n'est pas prise en charge.	Net_A envoie les informations à Wks_B comme si les deux versions étaient identiques à Wks_B. Si le mot clé <i>schedtime</i> est défini dans la définition du travail, il est automatiquement supprimé par Net_A.
Wks_B 8.3 Niveau ant. Sym_B	Wks_B fonctionne comme si sa version était identique à celle de Net_A.	Net_A envoie les informations à Wks_B. Si le mot clé <i>schedtime</i> est défini dans la définition du travail, il est automatiquement supprimé par Wks_B.	Net_A envoie les informations à Wks_B. Si le mot clé <i>schedtime</i> est défini dans la définition du travail, il est automatiquement supprimé par Wks_B.	Net_A envoie les informations à Wks_B. Si le mot clé <i>schedtime</i> est défini dans la définition du travail, il est automatiquement supprimé par Wks_B.

Tableau 144. Dépendances interréseaux dans un environnement mixte (suite)

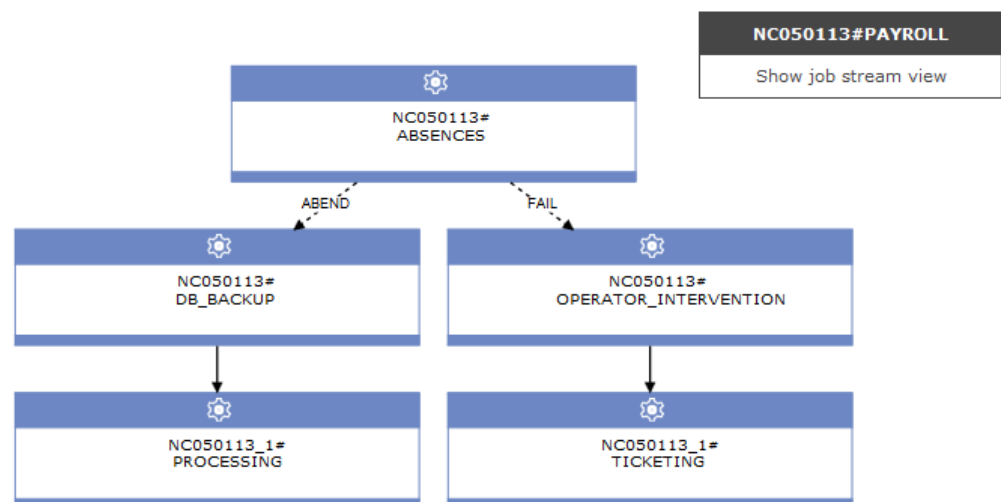
	Niveau ant. Net_A Niveau ant. Sym_A	Net_A 8.3 Niveau ant. Sym_A	Niveau ant. Net_A Sym_A 8.3	Net_A 8.3 Sym_A 8.3
Niveau ant. Wks_B Sym_B 8.3	Non pris en charge.	Non pris en charge.	Non pris en charge.	Non pris en charge.
Wks_B 8.3 Sym_B 8.3	Non pris en charge.	Non pris en charge.	Net_A envoie les informations à Wks_B. Si le mot clé <i>schedtime</i> est défini, il est analysé par Wks_B.	C'est un environnement de la version 8.3.

Chapitre 22. Application d'une logique de branchement conditionnelle

Avec IBM Workload Scheduler, vous pouvez définir des travaux à exécuter chaque fois et aussi souvent que nécessaire. Certains travaux doivent parfois attendre que l'exécution d'autres travaux soit terminée avant de démarrer à leur tour. Offrez plus de flexibilité à vos flux de travaux en choisissant le travail à exécuter en fonction du statut ou de la sortie d'un précédent travail. Chaque fois que des conditions indiquent si un segment de votre flux de travaux doit ou non s'exécuter, il existe une dépendance conditionnelle.

Lorsque vous spécifiez des dépendances, vous pouvez définir des flux de travaux dotés de branches alternatives en fonction de conditions, destinées spécifiquement à obtenir les mêmes résultats que lors de l'utilisation des instructions IF/THEN/ELSE. Vous pouvez utiliser des codes retour, le statut du travail, les variables de sortie et le contenu du journal de travail comme éléments de *logique conditionnelle* afin de déterminer le début d'un travail successeur. En plus d'apporter de la flexibilité à vos flux de travaux, la Vue graphique propose une représentation graphique des relations entre les travaux et les flots de travaux, y compris des dépendances et des conditions. Cette vue d'ensemble de votre flux de travaux est facile à lire et elle vous permet également d'éditer votre flux de travaux.

L'exemple suivant présente le flot de travaux PAYROLL qui démarre avec le travail ABSENCES, lequel est un travail prédécesseur, et qui est ensuite suivi de deux branches possibles de travaux pouvant être exécutées. La branche qui s'exécute dépend du résultat du travail initial, le travail prédécesseur ABSENCES. Les résultats possibles du travail ABSENCES sont définis dans *conditions de sortie*. Dans le cas où les conditions de sortie ne sont pas satisfaites, les travaux du flux qui ne s'exécutent pas passent à l'état SUPPRESSED, alors qu'avec des dépendances classiques, les travaux passent à l'état Hold jusqu'à ce que le prédécesseur soit à l'état SUCC. Les prédécesseurs peuvent être des travaux ou des flots de travaux.



Les conditions qui peuvent être des **conditions de statut**, basées sur le statut du travail, ou d'**autres conditions de sortie**, basées sur une expression de mappage

telle qu'un code retour, des variables de sortie ou une sortie dans un journal de travail. Lorsque le prédécesseur est un flot de travaux, la dépendance conditionnelle ne peut être qu'une condition de statut.

Conditions de statut

Ce sont des conditions basées sur le statut d'un travail, par exemple, si le travail a démarré, ou s'il s'est terminé avec l'état FAIL, ABEND, SUCC ou SUPPR. Pour les flots de travaux, les statuts valides sont SUCC, SUPPR et ABEND.

Autres conditions de sortie

D'autres types de conditions, y compris des conditions de sortie de réussite, peuvent être indiquées à l'aide d'une expression de mappage, qui peut être :

- Un code retour (agents tolérants aux pannes et dynamiques)
- Des variables de sortie (agents dynamiques)
- Du contenu de journal de travail (agents dynamique)

Une relation de dépendance de condition est définie à l'aide d'une *condition*. Vous définissez les conditions de sortie dans la définition de travail. Vous pouvez définir un nombre illimité de dépendances conditionnelles. Lorsque vous choisissez d'ajouter une dépendance conditionnelle à ce travail, vous sélectionnez les conditions de statut et de sortie à prendre en compte lors du traitement du travail. Voici un exemple d'autres conditions de sortie configurées dans la définition de travail.

Properties - Executable (9.3.0.01) - NC050024_1_1#ABSENCES

General Affinity Recovery Options Credentials Environment Task

* Name: ABSENCES * Workstation: NC050024_1_1

Description:

☐ Variable resolution at runtime

Successful Output Conditions (force the job status to successful):

Condition Name	Condition Value

Other Output Conditions (do not change the job status):

Condition Name	Condition Value
DB_FAIL	RC=1
TEMP_FULL	RC=3
UNKNOWN_ERR	RC>3
WAS_FAIL	RC=2

Vous pouvez configurer à la fois des conditions de sortie de réussite qui, lorsqu'elles sont satisfaites, signifient que le travail s'est correctement exécuté, et d'autres conditions de sortie qui, lorsqu'elles sont satisfaites, déterminent le travail successeur à exécuter. Les conditions de sortie sont évaluées dans "OR".

Lorsque ce travail est ajouté dans un flot de travaux en tant que travail successeur et qu'une dépendance conditionnelle est ajoutée au travail qui le précède (travail prédécesseur), une sélection des conditions est effectuée. Le panneau de propriétés pour la dépendance interne ou externe est mis à jour de façon dynamique afin d'inclure les conditions spécifiées initialement dans la définition de travail. En plus des conditions issues de la définition de travail, vous pouvez sélectionner des conditions basées sur le statut du travail. Si les conditions sélectionnées sont satisfaites lors du traitement du travail, le travail successeur correspondant

s'exécute.

Properties - Internal Job Dependency - **ABSENCES**

General

* Name:
ABSENCES

* Workstation:
NC050024

☒ Conditional Dependency

NOTE: The successor job will be suppressed if the conditional dependency is not satisfied. [More information](#)

Conditional Dependency Resolution Criteria

☐ Job started

☒ Successor job runs if the predecessor job or job stream completes with any of these statuses

☐ ABEND ☐ FAIL ☒ SUCC ☐ SUPPRESS

☐ Successor job runs if any of these conditions are satisfied

☒ UNKNOWN_ERR ? [Add a condition](#)

☐ DB_FAIL ?

☐ TEMP_FULL ?

☐ WAS_FAIL ?

Vous pouvez également joindre ou agréger des dépendances conditionnelles associées à différents prédécesseurs. Une jointure contient plusieurs dépendances, mais vous décidez lesquelles de ces dépendances doivent être satisfaites pour que la jointure soit considérée comme étant satisfaite. Vous pouvez définir un nombre illimité de dépendances conditionnelles, standard, ou les deux, dans une jointure.

Les dépendances conditionnelles sont prise en charge seulement pour les dépendances pour lesquelles le prédécesseur est un travail ou un flot de travaux au sein du même réseau et où tous les composants sont au moins au niveau de version 9.3 groupe de correctifs 2. Elles ne sont ni prises en charge sur les dépendances interréseaux, ni sur les agents tolérants aux pannes limités pour IBM i.

Configuration de dépendances conditionnelles

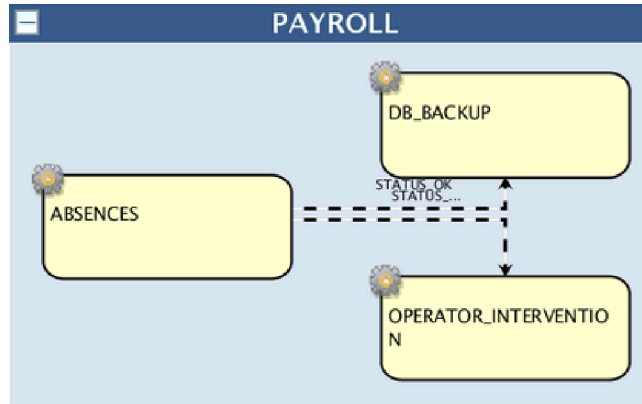
Vous pouvez configurer des dépendances conditionnelles afin de définir des flux de travaux avec des branches alternatives basées sur des conditions.

Les dépendances conditionnelles vous permettent de contrôler le moment du démarrage d'un travail successeur en fonction de la combinaison d'une ou plusieurs expressions de mappage (par exemple, des codes retour) ou des statuts d'un travail prédécesseur.

Assurez-vous que tous les composants de l'environnement IBM Workload Scheduler sont à la version 9.3, groupe de correctifs 1, ou version ultérieure.

Dans l'exemple suivant, le travail DB_BACKUP s'exécute si le travail ABSENCES satisfait la condition associée à STATUS_OK, et le travail

OPERATOR_INTERVENTION s'exécute si le travail ABSENCES satisfait la condition définie pour STATUS_ERR1.



Pour configurer ce type de traitement de travail, effectuez les étapes suivantes :

1. Création de la définition de travail et définition des conditions de sortie.
2. Définition de la dépendance conditionnelle.

Pour créer le flot de travaux PAYROLL, effectuez les étapes suivantes :

1. Ajoutez les travaux ABSENCES, DB_BACKUP et OPERATOR_INTERVENTION au flot de travaux nommé PAYROLL.
2. Ajoutez une dépendance, ici une dépendance interne, au travail DB_BACKUP. Dans les propriétés de la dépendance de travail interne, définissez la dépendance comme une dépendance conditionnelle en cochant la case **Dépendance conditionnelle**.
3. Dans la section Critères de résolution de la dépendance conditionnelle, sélectionnez **Successor job runs if any of these conditions are satisfied**, puis **STATUS_OK**.
4. Ajoutez une dépendance au travail OPERATOR_INTERVENTION. Dans les propriétés de la dépendance de travail interne, définissez la dépendance comme une dépendance conditionnelle en cochant la case **Dépendance conditionnelle**.
5. Dans la section Critères de résolution de la dépendance conditionnelle, sélectionnez **Successor job runs if any of these conditions are satisfied**, puis **STATUS_ERR1**.
6. Sauvegardez le flot de travaux.

Vous pouvez définir le même scénario en utilisant la ligne de commande Composer comme suit :

Définition de travail

```

WK1#ABSENCES
SCRIPTNAME "monscript.sh"
STREAMLOGON root
DESCRIPTION "Exemple de définition de travail"
TASKTYPE UNIX
SUCCOUTPUTCOND STATUS_OK "RC=0"
OUTPUTCOND STATUS_ERR1 "RC =2"
RECOVERY CONTINUE
END

```

Flot de travaux

```

SCHEDULE WK1#PAYROLL
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1800

```

```

CARRYFORWARD
:
WK1#DB_BACKUP
  FOLLOWS WK1#ABSENCES IF STATUS_OK
WK1#OPERATOR_INTERVENTION
  FOLLOWS WK1#ABSENCES IF STATUS_ERR1
END

```

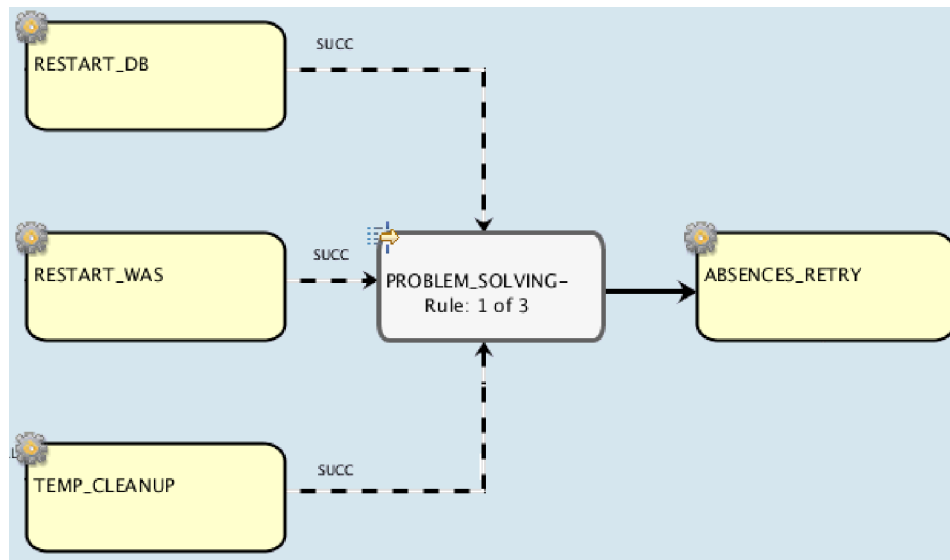
Pour plus d'informations sur la définition de dépendances conditionnelles à l'aide de la ligne de commande Composer, voir «Définition de travail», à la page 187, «Définition de flot de travaux», à la page 226 et «follows», à la page 248.

Jointure ou combinaison de dépendances

Vous pouvez décider de combiner un ensemble de dépendances dans une dépendance jointe.

Vous pouvez ajouter plusieurs dépendances relatives à différents prédécesseurs à une dépendance de jointure, puis indiquer combien de ces dépendances doivent être satisfaites pour considérer la jointure comme étant satisfaite. Lorsque la jointure est satisfaite, le travail successeur s'exécute.

Dans l'exemple qui suit, une dépendance de jointure, **PROBLEM_SOLVING**, a été insérée dans le flot de travaux. Elle contient trois dépendances conditionnelles relatives à des travaux prédécesseurs différents. Une seule dépendance conditionnelle doit être satisfaite pour considérer la jointure **PROBLEM_SOLVING** comme étant satisfaite. Le travail **ABSENCES_RETRY** suit la jointure **PROBLEM_SOLVING** et s'exécute lorsqu'au moins un des trois travaux prédécesseurs s'achève avec succès, c'est-à-dire en satisfaisant la règle établie sur la jointure. Si la jointure n'est pas satisfaite le travail **ABSENCES_RETRY** est supprimé.



Pour configurer la jointure de cet exemple, procédez comme suit :

1. Ajoutez les travaux **RESTART_DB**, **RESTART_WAS**, **TEMP_CLEANUP** et **ABSENCES_RETRY** au flot de travaux.
2. Cliquez sur le travail **ABSENCES_RETRY** avec le bouton droit de la souris, puis sélectionnez **Ajouter des dépendances > Join Dependencies** (dépendances de jointure).

3. Dans les propriétés de la dépendance de jointure, attribuez le nom **PROBLEM_SOLVING**.
4. La règle à appliquer à la jointure **PROBLEM_SOLVING** est qu'au moins l'une des dépendances doit être satisfaite. Conservez la sélection par défaut **At least 1**.
5. Cliquez avec le bouton droit de la souris sur la dépendance de jointure **PROBLEM_SOLVING** dans la vue Détails, puis sélectionnez **Ajouter des dépendances > Travail dans le même flot de travaux**.
6. Cliquez sur **Rechercher** pour afficher tous les travaux du flot de travaux.
7. Sélectionnez les travaux **RESTART_DB**, **RESTART_WAS** et **TEMP_CLEANUP**, puis cliquez sur **Ajouter**.
8. Editez les propriétés de chaque dépendance de travail interne pour en faire une dépendance conditionnelle.
 - a. Cochez la case **Dépendance conditionnelle**.
 - b. Dans **Critères de résolution de la dépendance conditionnelle**, sélectionnez **Successor job runs if the predecessor job or job stream completes with any of these statuses**.
 - c. Cochez la case **SUCC**.
 - d. Sauvegardez les modifications.

Les trois dépendances conditionnelles sur le travail **ABSENCES_RETRY** ont été regroupées dans la dépendance de jointure **PROBLEM_SOLVING** où au moins une des conditions (travail prédécesseur achevés avec l'état **SUCC**) doit être satisfaite pour que le travail **ABSENCES_RETRY** s'exécute.

Vous pouvez définir le même scénario en utilisant la ligne de commande Composer comme suit :

```
SCHEDULE WK1#PROCESSINFO
ON RUNCYCLE RULE1 "FREQ=DAILY;"
AT 1800
:
WK1#ABSENCES_RETRY
JOIN PROBLEM_SOLVING 1 OF
[DESCRIPTION "...."]
  FOLLOWS WK2#RESTART_DB IF SUCC
  FOLLOWS W32#RESTART_WAS IF SUCC
  FOLLOWS W32#TEMP_CLEANUP IF SUCC
ENDJOIN
END
```

Pour plus d'informations sur la définition d'une jointure à partir de la ligne de commande Composer, voir «Jointure», à la page 256.

Planification et soumission de dépendances conditionnelles

Vous planifiez des travaux IBM Workload Scheduler en les définissant dans des flots de travaux.

Vous pouvez utiliser Dynamic Workload Console ou la ligne de commande **conman** pour planifier et soumettre des travaux.

Après avoir défini un travail IBM Workload Scheduler, ajoutez-le à un flot de travaux avec tous les arguments de planification nécessaires, puis soumettez-le pour exécution.

Planification d'un travail ou flot de travaux à l'aide de Dynamic Workload Console

Pour planifier un travail ou un flot de travaux en fonction de critères spécifiques, voir la section relative à la conception de votre charge de travail dans *Dynamic Workload Console - Guide d'utilisation*.

Planification d'un travail ou flot de travaux à l'aide de la ligne de commande

Pour planifier un travail ou un flot de travaux en fonction de critères spécifiques, voir la section relative au contrôle du traitement des travaux et des flots de travaux dans *Guide d'utilisation et de référence*.

Soumission d'un travail ou flot de travaux à l'aide de Dynamic Workload Console

Pour soumettre un travail ou un flot de travaux en vue de son exécution selon un calendrier défini, voir la section relative à la soumission de la charge de travail à la demande en production dans *Dynamic Workload Console - Guide d'utilisation*.

Soumission d'un travail ou flot de travaux à partir de la ligne de commande conman

Pour soumettre travail en vue de son traitement, voir la section relative à la commande **submit sched** dans *Guide d'utilisation et de référence*. Pour soumettre un travail pour lancement, voir la section relative à la commande **submit job** dans *Guide d'utilisation et de référence*.

Evaluation et traitement d'un flux de dépendances conditionnelles

Une fois que vous avez soumis vos travaux ou flots de travaux, IBM Workload Scheduler les ajoute au plan de production et les évalue.

Les dépendances conditionnelles accroissent la flexibilité de votre charge de travail en sélectionnant le travail à exécuter en fonction de l'état du travail ou du résultat d'un précédent travail.

Les dépendances conditionnelles sont évaluées une fois toutes les dépendances standard du travail ou du flot de travaux sont satisfaites.

Si vous réexécutez un travail ou un flot de travaux, l'évaluation du flux de dépendances conditionnelles est effacée et toutes les dépendances sont de nouveau évaluées.

Si un travail prédécesseur ou un flot de travaux est annulé, et que le prédécesseur est à l'état final, la condition de sortie ou la condition d'état est évaluée et peut ou peut ne pas être évaluable. Si le prédécesseur n'est pas à l'état final et n'a évalué aucune condition, le successeur reste à l'état HOLD et le travail ou flot de travaux reste à l'état STUCK.

Dépendances de prédécesseur/successeur

Une dépendance de prédécesseur/successeur est satisfaite lorsque le travail sur lequel la dépendance est définie aboutit.

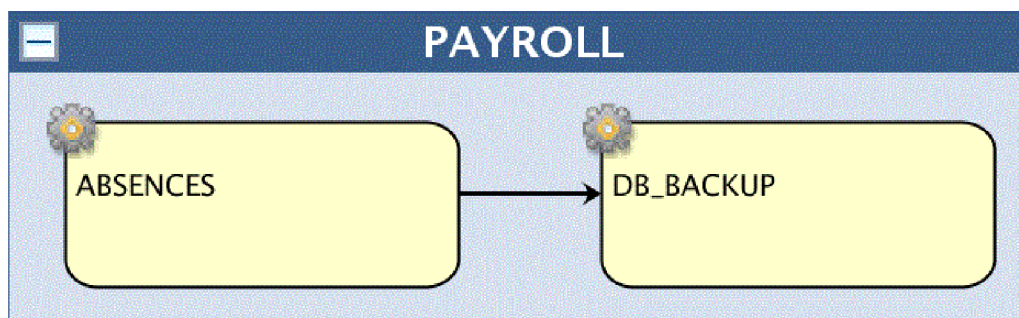


Figure 33. Dépendance de prédécesseur/successeur sur le travail ABSENCES

Dans cet exemple, le travail DB_BACKUP ne peut commencer qu'une fois le travail ABSENCES correctement achevé. Si le travail ABSENCES n'aboutit pas, le travail DB_BACKUP reste à l'état HOLD.

Dépendances conditionnelles à l'état du travail

Vous pouvez utiliser des dépendances conditionnelles pour apporter plus de flexibilité à votre charge de travail et automatiser certaines tâches récurrentes. Par exemple, décider qu'un travail spécifique doit s'exécuter lorsqu'un prédécesseur aboutit et un autre travail lorsque le prédécesseur n'aboutit pas.

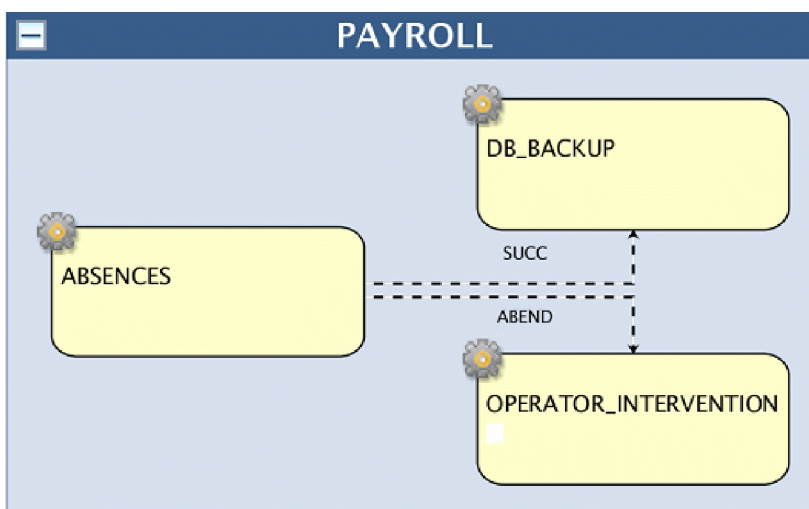


Figure 34. Deux dépendances conditionnelles différentes aux états SUCC et ABEND sur le travail ABSENCES

Dans cet exemple, si le travail ABSENCES aboutit, le travail DB_BACKUP s'exécute et le travail OPERATOR_INTERVENTION est supprimé. Par contre, si le travail ABSENCES se termine avec l'état ABEND, le travail OPERATOR_INTERVENTION s'exécute et le travail DB_BACKUP est supprimé. Si vous avez une dépendance de prédécesseur/successeur sur le travail DB_BACKUP, par exemple, le travail est supprimé, la dépendance de prédécesseur/successeur est libérée et le travail successeur peut s'exécuter. Si vous voulez propager l'état SUPPR du travail DB_BACKUP vers, par exemple, un travail successeur, définissez une dépendance conditionnelle à l'état SUPPR sur le travail DB_BACKUP. Cette dépendance entraîne le passage à l'état SUPPR du travail successeur du travail DB_BACKUP lorsque le travail DB_BACKUP se termine avec l'état SUPPR.

Lorsque toutes les dépendances conditionnelles définies dans le flot de travaux sont satisfaites, le travail ou le flot de travaux passe à l'état READY. Lorsqu'une dépendance conditionnelle n'est pas satisfaite, le travail ou le flot de travaux successeur associé passe à l'état SUPPR.

Dépendances conditionnelles à des conditions de sortie de travaux

Vous pouvez également conditionner le comportement de votre charge de travail en fonction du code retour du travail et automatiser un ensemble de réponses basé sur les problèmes qu'a rencontré le travail prédécesseur. L'échec du travail ABSENCES peut avoir de multiples raisons mais certaines peuvent aisément être anticipées et résolues. Le travail peut échouer parce que la base de données s'est arrêtée, que le WebSphere Application Server s'est arrêté, etc., ou qu'un problème inattendu qui nécessite l'intervention d'un opérateur s'est produit.

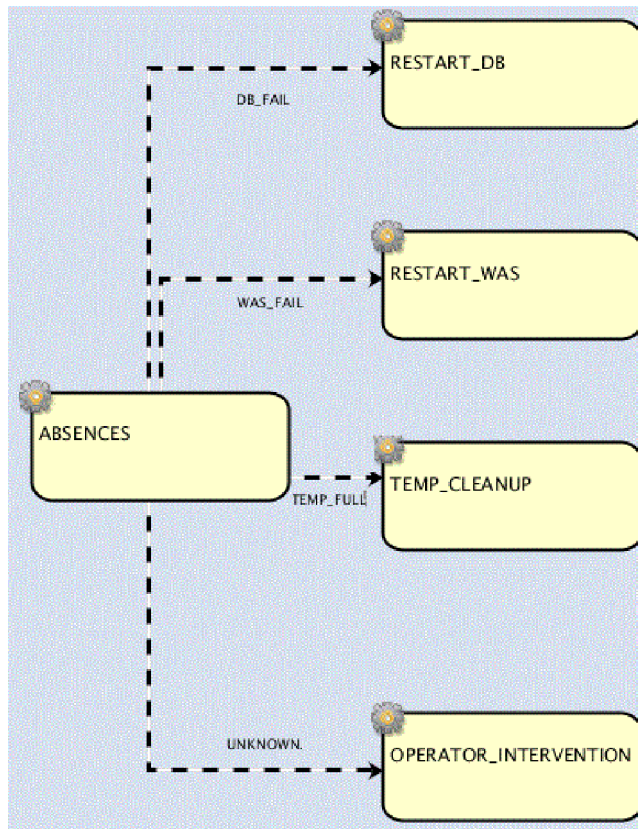


Figure 35. Dépendances conditionnelles sur les conditions de sortie du travail ABSENCES

Lors de la définition du travail ABSENCES, vous associez un code retour spécifique aux problèmes qui peuvent se produire de sorte qu'un travail spécifique démarre pour tenter de résoudre le problème. Par exemple, un code retour 1 indique que la base de données n'est pas accessible et entraîne l'exécution du travail DB_RESTART, ce qui démarre la base de données ; un code retour 2 indique que le serveur WebSphere Application Server n'est pas accessible et entraîne l'exécution du travail WAS_RESTART, ce qui démarre le serveur WebSphere Application Server, etc. Tout code retour supérieur à 3 signale une erreur inattendue et démarre le travail OPERATOR_INTERVENTION qui prévient l'opérateur.

Si le travail ABSENCES échoue avec l'un des codes retour définis dans les conditions de sortie, le travail correspondant démarre et les travaux restants sont supprimés.

Lorsqu'aucune des conditions de sortie n'est satisfaite, le travail ou flot de travaux reste à l'état HOLD.

Jointure de dépendances conditionnelles

Vous pouvez également combiner un ensemble de dépendances en une dépendance de jointure et indiquer combien d'entre elles doivent être satisfaites pour que la dépendance de jointure soit satisfaite.

Prenons, par exemple, la portion suivante du flot de travaux PAYROLL :

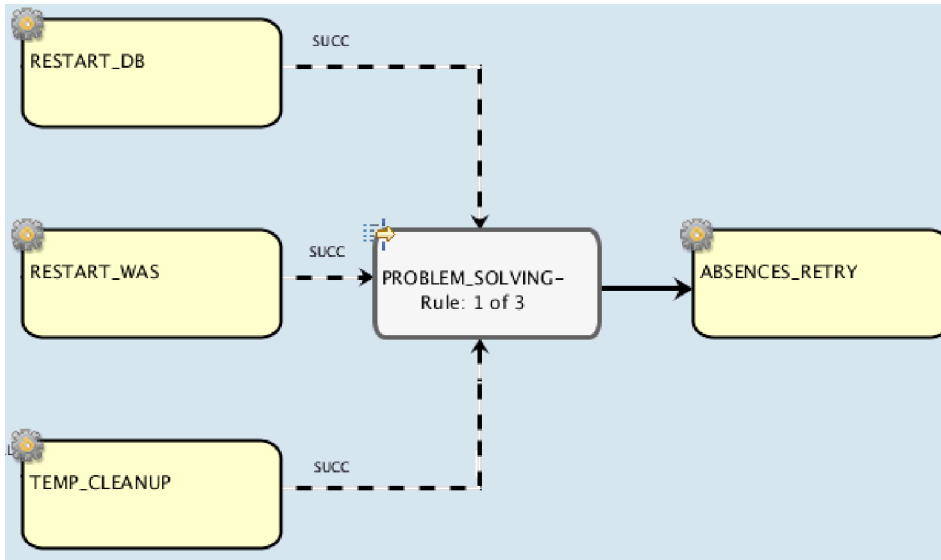


Figure 36. Dépendance de jointure contenant trois dépendances à l'état SUCC

Ici, la dépendance de jointure PROBLEM_SOLVING contient trois dépendances à l'état SUCC sur trois travaux différents. Cela signifie que lorsqu'au moins un des travaux RESTART_DB, RESTART_WAS ou TEMP_CLEANUP aboutit, la dépendance de jointure est satisfaite et le travail ABSENCES_RETRY peut démarrer.

Si aucun des travaux prédécesseurs n'aboutit, la dépendance de jointure PROBLEM_SOLVING n'est pas satisfaite et le travail ABSENCES_RETRY est supprimé.

Si le nombre de dépendances conditionnelles défini dans la dépendance jointure est satisfait, le travail ou flot de travaux passe à l'état READY. Si le nombre de dépendances conditionnelles défini dans la dépendance de jointure n'est pas satisfait, le travail ou flot de travaux passe à l'état SUPPR.

Evaluation de dépendances conditionnelles dans des flots de travaux

L'évaluation des dépendances conditionnelles dans des flots de travaux dépend de plusieurs facteurs, comme décrit dans les exemples suivants.

Lorsqu'un travail d'un flot de travaux est à l'état SUPPR, il est évalué comme étant à l'état CANCELLED. Si tous les travaux du flot de travaux sont à l'état SUPPR, le flot de travaux passe à l'état SUCC. Ce même comportement provoque le passage à l'état SUCC d'un flot de travaux ne contenant que des travaux à l'état CANCELLED.

Lorsque vous passez un flot de travaux à l'état SUPPR, tous les travaux qu'il contient qui n'ont pas atteint un état final passent à l'état SUPPR. Ce processus s'applique, par exemple, aux travaux qui sont à l'état READY ou HOLD.

CONDITION D'ETAT NON SATISFAITE ET PARAMETRE RECOVERY STOP PROVOQUANT L'ACHEVEMENT DU FLOT DE TRAVAUX AVEC L'ETAT ABEND

Le travail PERFORMANCE a une dépendance conditionnelle à l'achèvement du travail SALES avec l'état SUCC. Or, le travail SALES se termine avec l'état ABEND. La dépendance conditionnelle au travail SALES est évaluée comme n'étant pas satisfaite de sorte que le travail PERFORMANCE est supprimé. Le flot de travaux se termine avec l'état ABEND car le travail SALES qui s'est terminé de manière anormale est défini sur **recovery stop** et l'état SUPPR du travail PERFORMANCE est considéré comme un état CANCELLED.

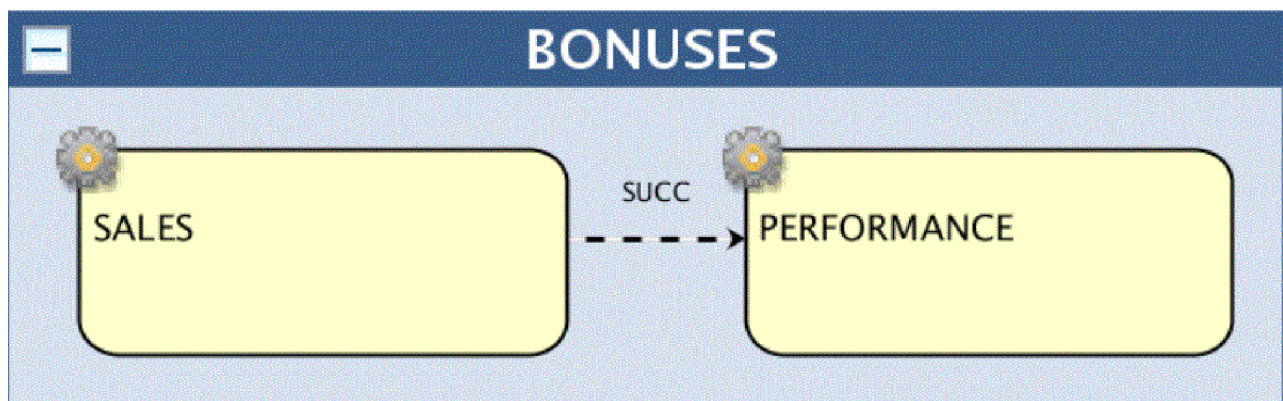


Figure 37. Dépendance conditionnelle d'état sur un travail avec le paramètre **recovery stop**

Pour plus d'informations sur le paramètre **recovery stop**, voir la section relative à la définition d'actions de reprise de travaux dans le document *Guide d'utilisation et de référence*.

DEPENDANCE D'ETAT NON SATISFAITE PROVOQUANT L'ACHEVEMENT DU FLOT DE TRAVAUX AVEC L'ETAT SUCC

Le travail OPERATOR_CALL a une dépendance conditionnelle à l'achèvement du travail DB2_BACKUP avec l'état ABEND. Or, le travail DB2_BACKUP se termine avec l'état SUCC. La dépendance conditionnelle au travail DB2_BACKUP est évaluée comme n'étant pas satisfaite de sorte que le travail OPERATOR_CALL est supprimé. L'état du flot de travaux est évalué comme étant SUCC car l'état SUPPR du travail OPERATOR_CALL est considéré comme un état CANCELLED.

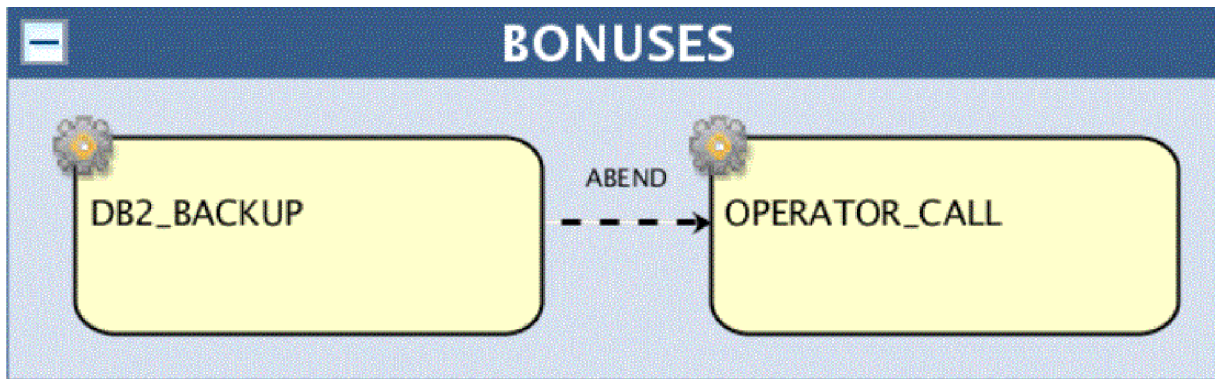


Figure 38. Dépendance conditionnelle d'état ABEND

DEPENDANCE D'ETAT NON SATISFAITE PROVOQUANT L'ACHEVEMENT DU FLOT DE TRAVAUX AVEC L'ETAT STUCK

Le travail PERFORMANCE a une dépendance conditionnelle à l'achèvement du travail SALES avec la condition de sortie STATUS_OK. Or, le travail SALES se termine avec l'état ABEND et aucune condition de sortie n'est satisfaite. En conséquence, la dépendance conditionnelle au travail SALES n'est pas évaluée et le flot de travaux s'achève avec l'état STUCK.

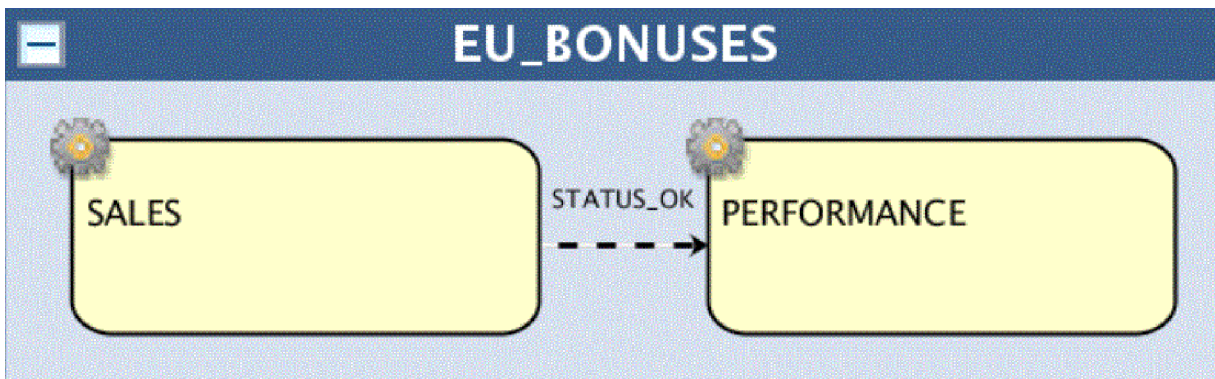


Figure 39. Condition de sortie STATUS_OK

CONDITION DE SORTIE SATISFAITE ET PARAMETRE RECOVERY STOP PROVOQUANT L'ACHEVEMENT DU FLOT DE TRAVAUX AVEC L'ETAT ABEND

Le travail OPERATOR_CALL a une dépendance conditionnelle à l'achèvement du travail DB2_BACKUP avec la condition de sortie ERROR. Le travail DB2_BACKUP se termine avec l'état ABEND et la condition de sortie ERROR est satisfaite. En conséquence, la dépendance conditionnelle au travail DB2_BACKUP est évaluée comme étant satisfaite. Le travail OPERATOR_CALL s'achève avec l'état SUCC. Toutefois, le flot de travaux s'achève avec l'état ABEND car il contient au moins un travail à l'état ABEND défini sur **recovery stop**. Pour que le flot de travaux s'achève avec l'état SUCC, définissez le travail DB2_BACKUP sur **recovery continue**.

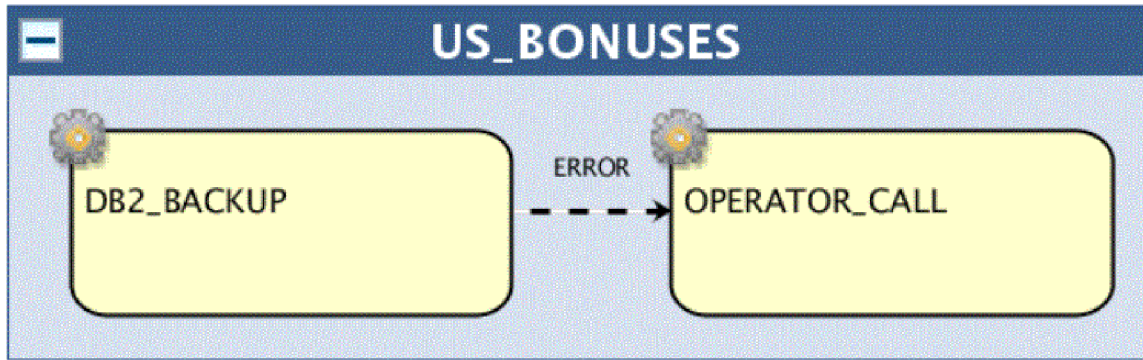


Figure 40. Condition de sortie ERROR

CONDITION DE SORTIE NON SATISFAITE ET PARAMETRE RECOVERY STOP PROVOQUANT L'ACHEVEMENT DU FLOT DE TRAVAUX AVEC L'ETAT ABEND

Le travail PERFORMANCE a une dépendance conditionnelle à l'achèvement du travail SALES avec la condition de sortie STATUS_OK. Or, le travail SALES se termine avec l'état ABEND et la condition de sortie STATUS_OK n'est pas satisfaite. En conséquence, la dépendance conditionnelle au travail SALES est évaluée comme n'étant pas satisfaite et le travail PERFORMANCE est supprimé. Le flot de travaux se termine avec l'état ABEND car le travail SALES qui s'est terminé de manière anormale est défini sur **recovery stop**, et l'état SUPPR du travail PERFORMANCE est considéré comme étant à l'état CANCELLED.

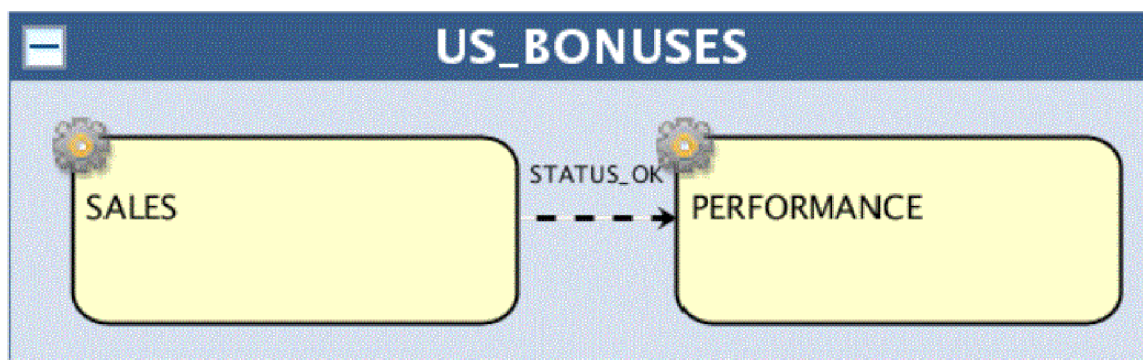


Figure 41. Condition de sortie STATUS_OK

Surveillance de dépendances conditionnelles

Surveillez les travaux IBM Workload Scheduler à l'aide de Dynamic Workload Console ou de la ligne de commande **conman**.

Dynamic Workload Console affiche les informations d'exemple suivantes :

'PAYROLL' Properties

Monitor Workload (Owner: wasadmin; En... > 'PAYROLL' Properties

Refresh

Close View

Expand All

Collapse All

"PAYROLL" Properties

"PAYROLL" Properties	Value
▼ Properties	
▼ General	
▼ Information	
Name	PAYROLL
Workstation Name	NC050024
Original Name	
Internal Identifier	0AAAAAAAAAAAAAPL7
Scheduled Time	11/19/15 10:33 AM EST
Information	
Limit	
Contains Monitored Job	No
Priority	10
Carry Forward	No
Monitored	No
▼ Status	
Status	Ready
Internal Status	READY

Vous pouvez surveiller les conditions de sortie à l'aide de la ligne de commande **conman showjobs** et **showschedules**.

Dans cet exemple, vous pouvez voir le statut du flot de travaux PAYROLL après sa soumission et avant son exécution :

```
%ss @#PAYROLL
(Est) (Est) Jobs Sch
Workstation Job Stream SchedTime State Pr Start Elapse # OK Lim
NC050024      #PAYROLL          0958 11/19  HOLD  10              7  0
```

Cet exemple fournit des détails sur les travaux du flot de travaux PAYROLL :

```
%sj @#PAYROLL
(Est) (Est)
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies
NC050024      #PAYROLL          0958 11/19 ***** READY 10
(NC050024_1_1#)OPERATOR_INTERVENTION          HOLD 10          ABSENCES IF UNKNOWN_ERR
(NC050024_1_1#)DB_BACKUP                      HOLD 10          JOIN SUCCESS 1 OF
          ABSENCES IF SUCC
          ABSENCES_RETRY IF SUCC
(NC050024_1_1#)ABSENCES                      READY 10
(NC050024_1_1#)RESTART_DB                    HOLD 10          ABSENCES IF DB_FAIL
(NC050024_1_1#)RESTART_WAS                   HOLD 10          ABSENCES IF WAS_FAIL
(NC050024_1_1#)TEMP_CLEANUP                  HOLD 10          ABSENCES IF TEMP_FULL
(NC050024_1_1#)ABSENCES_RETRY                HOLD 10          JOIN PROBLEM_SOLVING 1 OF
          RESTART_DB
          RESTART_WAS
          TEMP_CLEANUP
```

Si vous exécutez la commande **showjob;info** sur l'un des travaux, vous pouvez obtenir certains détails concernant ce travail :

```
%sj NC050024#PAYROLL.ABSENCES;info
----- Restart -----
Workstation      Job Stream      SchedTime  Job      JobFile      Opt  Job      Prompt
NC050024        #PAYROLL        0958 11/19
(NC050024_1_1#)ABSENCES
.....
                                oc: DB_FAIL n/a "RC=1"
                                oc: TEMP_FULL n/a "RC=3"
                                oc: UNKNOWN_ERR n/a "RC>3"
                                oc: WAS_FAIL n/a "RC=2"
```

Pour obtenir plus de détails, exécutez la commande **showjob;props**. L'exemple de sortie suivant est un sous-ensemble des informations que vous obtenez en exécutant cette commande :

```
%sj NC050024#PAYROLL.ABSENCES;props
General Information
  Job = ABSENCES
  Workstation = NC050024_1_1
.....
Other Output Conditions
  DB_FAIL = n/a "RC=1"
  TEMP_FULL = n/a "RC=3"
  UNKNOWN_ERR = n/a "RC>3"
  WAS_FAIL = n/a "RC=2"
```

Surveillance de jointure de dépendances conditionnelles

L'Dynamic Workload Console affiche les informations d'exemple suivantes :

Dependency Status	Internal Status	Job	Job Stream	Workstation (Job Stream)	Scheduled Time	Conditional Dependency	Join	Unsatisfied Join Conditions
☒ Satisfied	SUCC	ABSENCES_RETRY	PAYROLL	NC050024	11/25/15 5:03 AM EST	SUCC	SUCCESS(Rule:1 of 2)	0
☒ Join Satisfied	ABEND	ABSENCES	PAYROLL	NC050024	11/25/15 5:03 AM EST	SUCC	SUCCESS(Rule:1 of 2)	0

Remarque : Dans Dynamic Workload Console, vous pouvez extraire des informations sur les dépendances de la jointure, si elles sont satisfaites ou non, tandis que cela n'est pas possible depuis la ligne de commande.

Les jointures de dépendances sont représentées en utilisant une syntaxe de type Composer. Seules les dépendances de jointure non résolues et indéterminées sont affichées. Si la dépendance de jointure est satisfaite, aucune dépendance ne s'affiche.

Voici un exemple de la façon dont les dépendances conditionnelles de la jointure sont affichées à l'aide de **conman**. Vous pouvez voir que la dépendance de jointure **PROBLEM_SOLVING** a été satisfaite et qu'elle n'est plus affichée. La dépendance de jointure **SUCCESS** sera ensuite évaluée :

```
%sj @#PAYROLL
(Est) (Est)
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies
NC050024      #PAYROLL      1007 11/19 *****
(NC050024_1_1#)OPERATOR_INTERVENTION SUPPR 10 10:08 (00:01)
(NC050024_1_1#)DB_BACKUP HOLD 10 (00:01) ABSENCES IF UNKNOWN_ERR
JOIN SUCCESS 1 OF
ABSENCES IF SUCC
```

(NC050024_1_1#)ABSENCES	ABEND 10 10:08 00:01	ABSENCES_RETRY IF SUCC
(NC050024_1_1#)RESTART_DB	SUCC 10 10:08 00:01	1 #J355305499
(NC050024_1_1#)RESTART_WAS	SUPPR 10	0 #J355305500
(NC050024_1_1#)TEMP_CLEANUP	SUPPR 10	ABSENCES IF WAS_FAIL
(NC050024_1_1#)ABSENCES_RETRY	SUCC 10 10:09 00:01	ABSENCES IF TEMP_FULL
		0 #J355305501

Une fois que le flot de travaux est terminé, le travail ABSENCES s'est terminé avec le statut ABEND, ce qui a entraîné le démarrage du travail RESTART_DB et la résolution du problème. Les travaux REATART_WAS et TEMP_CLEANUP ont été supprimés, car ils ne sont plus nécessaires. La dépendance de jointure PROBLEM_SOLVING est maintenant satisfaite. Le travail ABSENCES_RETRY se termine également avec succès, la dépendance de jointure SUCCESS est donc satisfaite, ce qui entraîne à son tour le démarrage du travail DB_BACKUP. Vous pouvez extraire les informations suivantes à partir de la commande **showjobs** :

```
%sj @#PAYROLL
(Est) (Est)
Workstation Job Stream SchedTime Job State Pr Start Elapse ReturnCode Dependencies
NC050024 #PAYROLL 1007 11/19 ***** ABEND 10 10:08 00:02
(NC050024_1_1#)OPERATOR_INTERVENTION SUPPR 10 ABSENCES IF UNKNOWN_ERR
(NC050024_1_1#)DB_BACKUP SUCC 10 10:09 00:01 0 #J355305502
(NC050024_1_1#)ABSENCES ABEND 10 10:08 00:01 1 #J355305499
(NC050024_1_1#)RESTART_DB SUCC 10 10:08 00:01 0 #J355305500
(NC050024_1_1#)RESTART_WAS SUPPR 10 ABSENCES IF WAS_FAIL
(NC050024_1_1#)TEMP_CLEANUP SUPPR 10 ABSENCES IF TEMP_FULL
(NC050024_1_1#)ABSENCES_RETRY SUCC 10 10:09 00:01 0 #J355305501
```

Remarque : Il existe quelques différences quant à la manière d'évaluer et de résoudre des dépendances lorsque la réplication de plan est activée. Si elle est activée (paramètre par défaut), les dépendances sont résolues dès que les conditions nécessaires à leur satisfaction sont réunies. Si la réplication de plan n'est pas activée, un ordre spécifique d'évaluation est respecté, dans lequel les flots de travaux présentant des dépendances sur des flots de travaux ou des travaux externes sont résolues en premier, les dépendances entre les travaux à l'intérieur du flot de travaux étant résolues en second lieu.

De la même manière, si la réplication de plan est activée, toutes les dépendances d'un travail ou d'un flot de travaux à l'état SUPPRESS sont évaluées. Si la réplication de plan n'est pas activée, l'évaluation des dépendances s'arrête dès qu'un travail ou un flot de travaux passe à l'état SUPPRESS lorsqu'une dépendance conditionnelle n'a pas été satisfaite. Dans les deux cas, le résultat final est le même. Toutefois, lors de la surveillance de l'avancée des travaux et des flots de travaux dans ces deux situations, vous pouvez constater des résultats différents.

Gestion du plan de dépendances conditionnelles

Les dépendances conditionnelles sont gérées dans le plan, soit à partir de la ligne de commande **conman**, soit depuis l'interface utilisateur Web de Dynamic Workload Console.

Vous pouvez exécuter les actions disponibles suivantes sur des travaux où une dépendance conditionnelle est définie entre le travail prédécesseur et le travail successeur :

Ajouter une dépendance

- Ajouter une dépendance à un travail : «adddep job», à la page 439
- Ajouter une dépendance à un flot de travaux : «adddep sched», à la page 442

Supprimer une dépendance

- Supprimer une dépendance d'un travail : «deldep job», à la page 455

- Supprimer une dépendance d'un flot de travaux : «deldep sched», à la page 457

Libérer une dépendance

- Libérer un travail de ses dépendances : «release job», à la page 476
- Libérer un flot de travaux de ses dépendances : «release sched», à la page 478

Confirmer l'achèvement

Confirmer l'achèvement du travail dans différents statuts : «confirm», à la page 450

Chapitre 23. Définition et gestion des dépendances croisées

Les *dépendances croisées* IBM Workload Scheduler vous aident à intégrer et automatiser le traitement des travaux lorsque :

- La charge de travail est répartie sur différents environnements de planification, car certaines des activités s'exécutent sur différents sites ou impliquent différentes unités d'organisation, ou requièrent des compétences différentes pour s'exécuter.
- Même si la plus grande partie de la charge de travail par lots est gérée localement, aucun de ces environnements n'est complètement isolé des autres car ils interagissent fréquemment pour échanger ou synchroniser des données et activités.

Plus spécifiquement, la fonction de dépendance croisée est essentielle lorsque vous avez besoin de synchroniser facilement des tâches entre différents environnements de planification afin de pouvoir :

- Définir dans un environnement de planification des dépendances sur des tâches par lots gérées par un autre environnement de planification.
- Surveiller l'état des travaux prédécesseurs distants comme s'ils s'exécutaient dans votre environnement local.

De plus, vous pouvez contrôler le statut de ces dépendances en naviguant à partir d'une interface utilisateur unique sur les différents environnements de planification.

Ce chapitre décrit comment définir et utiliser les dépendances croisées.

Il contient les sections suivantes :

- «Présentation des dépendances croisées»
- «Flux de processus dans l'environnement de planification distribué», à la page 847
- «Définition d'une dépendance croisée», à la page 850
- «Modification de l'état du travail reflet après que la liaison est établie», à la page 852
- «Liaison d'un travail reflet z/OS», à la page 854
- «Modification de l'état du travail reflet après que la liaison est établie», à la page 857

Remarque : Selon vos besoins et exigences, vous pouvez choisir entre des dépendances interréseau et des dépendances croisées pour établir une dépendance entre un travail s'exécutant sur le moteur local et un travail s'exécutant sur un moteur IBM Workload Scheduler distant. Voir «Définition de dépendances», à la page 23 pour une description des différences entre ces deux types de dépendances.

Présentation des dépendances croisées

Une dépendance croisée est, d'un point de vue logique, une dépendance qu'un travail local possède sur une instance de travail qui est planifiée pour s'exécuter sur le moteur distant.

Utilisez les dépendances croisées pour intégrer la charge de travail s'exécutant sur différents moteurs, qui peuvent être les moteurs IBM Workload Scheduler for z/OS (contrôleur) ou les moteurs IBM Workload Scheduler (gestionnaire de domaine maître et gestionnaire de domaine maître de sauvegarde).

Les objets et termes suivants sont utilisés pour décrire et implémenter des dépendances croisées :

Poste de travail de moteur distant

Nouveau type de poste de travail qui représente localement un moteur IBM Workload Scheduler distant, distribué ou z/OS. Ce type de poste de travail utilise une connexion basée sur le protocole HTTP ou HTTPS pour permettre à l'environnement local de communiquer avec l'environnement distant.

Travail distant

Travail planifié pour s'exécuter sur un moteur IBM Workload Scheduler distant.

Travail reflet

Travail défini localement, sur un poste de travail de moteur distant, qui est utilisé pour mapper un travail distant. La définition de travail reflet contient toutes les informations nécessaires pour associer correctement l'instance de travail distante dans le plan de moteur distant.

Liaison

Processus associant un travail reflet avec une instance de travail distant dans le plan du moteur IBM Workload Scheduler distant.

D'un point de vue logique dans l'environnement local :

- Le poste de travail du moteur distant permet de mapper le moteur IBM Workload Scheduler distant.
- Le travail reflet défini sur ce poste de travail de moteur distant, est utilisé pour mapper une instance de travail distant planifiée pour s'exécuter sur ce moteur IBM Workload Scheduler.

Définissez une dépendance croisée lorsque vous souhaitez qu'un *travail local* (exécution sur votre moteur local) soit dépendant d'un *travail distant* (exécution sur un moteur distant).

Pour ce faire, procédez comme suit :

1. Créez un travail *reflet* qui s'exécute sur votre moteur local.
2. Définissez une dépendance normale qui rend votre travail *local* dépendant du travail *reflet*.

Lorsque vous créez le reflet, tenez compte des éléments suivants :

- Il doit être défini sur un poste de travail de type moteur distant qui pointe vers le moteur distant (qui est le moteur sur lequel le travail distant est planifié pour s'exécuter).
- Vous devez vous assurer qu'il pointe vers le travail distant avec lequel vous créez la dépendance croisée.

La figure 42, à la page 847 présente le flux logique implémentant des dépendances croisées :

1. Lors du processus de liaison, le travail reflet est associé à l'instance du travail distant.

2. Une fois que la liaison est établie, l'état du travail reflet est mis à jour en fonction de la transition d'état du travail distant.
3. Lorsque l'état du travail reflet devient SUCC, la dépendance normale du travail local est libérée, ainsi que la dépendance croisée de ce travail local sur le travail distant.

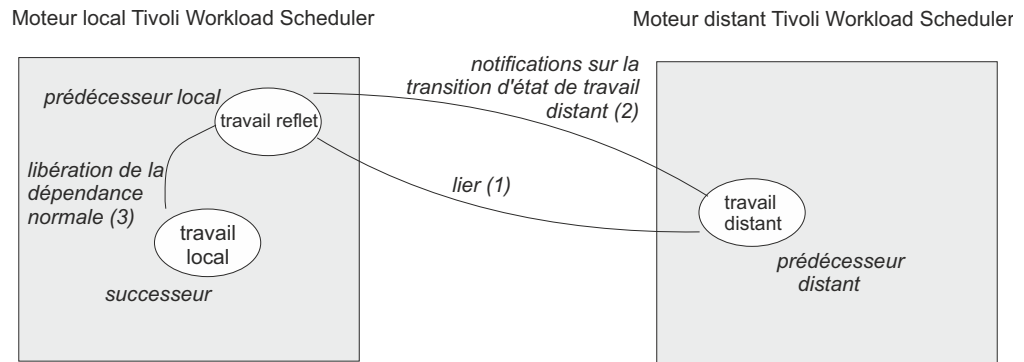


Figure 42. Logique de dépendance croisée

Flux de processus dans l'environnement de planification distribué

Selon que le moteur local émet ou reçoit une demande de liaison, le flux de processus et les composants impliqués changent. Dans les deux cas, le poste de travail courtier de l'environnement local doit être prêt et fonctionner pour permettre la gestion des demandes de liaison.

Flux de processus lorsque le moteur local envoie une demande de liaison sur un moteur distant

Lorsque vous définissez un travail reflet, vous spécifiez les informations nécessaires pour établir une liaison avec un travail dans le plan du moteur distant.

Lorsque l'heure planifiée du travail reflet est atteinte, si le travail reflet ne possède pas de dépendances, il est sélectionné par la commande **batchman** locale pour soumission et son statut est défini par INTRO.

La demande de liaison est envoyée au moteur distant. L'état du travail reflet est défini par WAIT.

Dès que le processus de liaison est achevé, le moteur distant renvoie au moteur local une notification avec le résultat de liaison.

Le tableau 145, à la page 848 présente l'état du travail reflet selon que :

- L'instance à lier existe ou non dans le plan du moteur distant.
- L'état du travail distant est lié.

Tableau 145. Transition d'état de travail reflet

Etat du travail reflet dans le plan de production :	Sur le moteur distant :
BOUND	z/OS L'instance de flot de travaux distant pour la liaison a été trouvée dans le plan à long terme ou dans le plan courant. Réparti L'instance de flot de travaux distant pour la liaison a été trouvée dans le plan de préproduction.
ERROR	z/OS L'une des situations suivantes s'est produite : • L'instance du flot de travaux distant pour la liaison n'existe ni dans le plan à long terme, ni dans le plan en cours. • L'instance du flot de travaux distant pour la liaison a été trouvée dans le plan à long terme mais, lorsqu'elle est incluse dans le plan en cours, elle ne contient pas l'instance du travail demandé. Réparti L'une des situations suivantes s'est produite : • L'instance du flot de travaux distant à lier n'existe pas dans le plan de préproduction. • L'instance du flot de travaux distant pour la liaison a été trouvée dans le plan de préproduction, mais lorsqu'elle est incluse dans le plan de production, elle ne contient pas l'instance du travail demandé. • L'utilisateur de liaison distant n'est pas autorisé à accéder à l'instance de travail demandée dans le plan de production.
EXEC	L'état du travail distant est EXEC.
SUCC	L'état du travail distant est SUCC.
FAIL	L'état du travail distant est FAIL.
ABEND	L'état du travail distant est ABEND.
SUCC	L'état du travail distant est CANCELED.

Remarque : L'état du travail reflet est FAIL lorsque sa soumission a également échoué.

Pour plus de détails sur la transition d'état du travail reflet, voir «Modification de l'état du travail reflet après que la liaison est établie», à la page 852 et «Modification de l'état du travail reflet après que la liaison est établie», à la page 857.

Flux de processus lorsque le moteur distant reçoit une demande de liaison du moteur local

Lorsque le moteur distant reçoit une demande de liaison du moteur local, les informations contenues dans la demande sont utilisées pour exécuter la liaison dans le plan de préproduction distant.

La demande de liaison contient également une liste ordonnée des URL que le moteur distant utilise pour envoyer des notifications au moteur local. Si le moteur local est distribué, la liste est constituée des URL spécifiées dans la propriété JDURL du fichier nommé *rép_base_TDWB/config/JobDispatcherConfig.properties*.

Remarque : Par défaut, IBM Workload Scheduler utilise TWSUser pour exécuter la liaison dans le plan de production. Si vous souhaitez limiter et contrôler les travaux qui peuvent être liés, vous pouvez indiquer un autre utilisateur à l'aide de l'option globale **bindUser**. L'utilisateur spécifié n'a pas besoin d'être défini comme utilisateur sur le système d'exploitation, ni même de posséder un mot de passe, mais il doit exister en tant qu'entrée dans le fichier de sécurité avec les autorisations suivantes :

- Autorisation DISPLAY sur les objets *job* et *schedule* qui peuvent être liés
- Autorisation LIST sur les objets *job* qui peuvent être liés. Cette autorisation est requise uniquement si l'option globale enListSecChk est définie sur yes.

Si les autorisations requises ne sont pas spécifiées, une notification comportant une erreur est renvoyée au moteur qui a demandé la liaison. Le moteur distant renvoie au moteur local :

Une notification avec l'état BOUND

Si le plan de préproduction contient au moins une instance du flot de travaux spécifié dans la demande de liaison et que la définition de ce flot de travaux contient le travail à lier.

Une notification avec l'état de l'instance du travail lié

Si l'instance du travail à lier est trouvée dans le plan de production et dès que son état change.

Une notification avec une erreur

Si l'instance du travail à lier est introuvable ou si l'utilisateur de liaison n'a pas d'autorisation.

Le processus distant **batchman** écrit une entrée dans la file d'attente PlanBox.msg dès que l'état d'un travail distant est modifié.

Toutes les 30 secondes, le système analyse la file d'attente PlanBox.msg et y recherche de nouvelles entrées documentant un changement d'état des travaux distants liés. Dès qu'un changement d'état est trouvé, une notification contenant l'état du travail distant lié est renvoyée au moteur qui a demandé la liaison.

Remarque : Pour modifier le délai d'interrogation, indiquez une valeur en secondes pour **com.ibm.tws.planner.monitor.checkPlanboxInterval** dans le fichier *RACINE_WAS/profiles/pronom_fichier/properties/TWSConfig.properties* puis redémarrez WebSphere Application Server.

Définition d'une dépendance croisée

Effectuez ces étapes pour définir une dépendance croisée entre un travail qui s'exécute dans votre environnement et un autre travail s'exécutant sur un moteur IBM Workload Scheduler différent :

1. Création d'un poste de travail de moteur distant

Vous pouvez créer un poste de travail de moteur distant pour un moteur distant spécifique lorsque vous avez besoin de définir des dépendances à des instances de travail qui s'exécutent sur ce moteur distant. Sur un poste de travail de moteur distant, vous ne pouvez exécuter que des travaux reflet.

Pour une utilisation optimale, Si le moteur distant IBM Workload Scheduler est distribué, vous pouvez définir un pool dynamique contenant une liste ordonnée des postes de travail de moteur distant pointant vers le maître distant et vers ses maîtres de sauvegarde afin de garantir que les fonctions de reprise et de modification de gestionnaires sont appliquées. Pour de plus amples informations sur le pool de postes de travail, voir «Poste de travail», à la page 11.

Remarque : Il est recommandé que :

- Tous les environnements distribués impliqués aient activé leur fonction de fuseau horaire. Pour de plus amples informations, voir «Activation de la gestion des fuseaux horaires», à la page 799.
- Vous spécifiez une propriété TIMEZONE sur les postes de travail de moteur distant où le fuseau horaire est défini sur le système d'exploitation des gestionnaires de domaine maîtres ou les gestionnaires de domaines maîtres de sauvegarde vers lesquels ils pointent.

Pour de plus amples informations sur les paramètres spécifiques à utiliser lors de la définition d'un poste de travail de moteur distant, voir «Définition de poste de travail», à la page 166.

2. Définition d'un travail reflet s'exécutant sur le poste de travail de moteur distant

Créez un travail reflet pointant vers une instance de travail spécifique définie sur un moteur distant lorsque vous souhaitez suivre l'état de ce travail distant dans votre environnement local et définir des dépendances croisées sur ce travail distant.

Sur les environnements IBM Workload Scheduler distribués, vous pouvez utiliser des alias pour les noms de travaux et de flots de travaux. Si vous définissez un travail reflet distribué, assurez-vous que :

- Le nom du flot de travaux distant spécifié contient le nom du flot de travaux tel qu'il est défini dans la base de données.
- Le nom de travail distant spécifié contient l'alias du travail distant à lier, s'il est défini.

Si vous ne suivez pas ces consignes, la liaison échoue et l'état du travail reflet devient ERROR.

Dans la définition du travail reflet, définissez `COMPLETE_IF_BIND_FAILS` dans la zone `rccondsucc` pour spécifier si l'état du travail reflet doit être forcé à être SUCC ou ERROR en cas d'échec de la liaison dans le plan du moteur distant.

Pour de plus amples informations sur les paramètres spécifiques à utiliser lors de la définition d'un travail reflet, voir «Définition de travail», à la page 187.

Selon que le moteur distant est z/OS ou distribué, vous pouvez utiliser différents critères de correspondance :

Si le moteur distant est distribué

Vous pouvez choisir l'un des critères de correspondance suivants :

Tableau 146. Critères de correspondance pour les travaux reflet distribués

Sur le Dynamic Workload Console	Mot clé correspondant utilisé dans composer
Valeur précédente la plus proche	précédent
Dans un intervalle relatif	relative from= <i>temps_avant_heure_planifiée</i> to= <i>temps_après_heure_planifiée</i>
Dans un intervalle absolu	absolute from= <i>début_intervalle</i> to= <i>fin_intervalle</i>
Même date de planification	sameDay

Pour de plus amples informations sur ces critères de correspondance, voir «Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux», à la page 71.

Si le moteur distant est z/OS

La valeur précédente la plus proche est le seul critère de correspondance pris en charge par IBM Workload Scheduler for z/OS.

L'heure planifiée de l'instance de flot de travaux contenant le travail reflet est utilisée pour le critère de correspondance dans le plan du moteur distant.

La transition d'état du travail reflet est mappée vers la transition d'état de l'instance de travail distant.

3. Ajoutez une dépendance sur le travail reflet

Vous ajoutez une dépendance croisée pour un travail local sur le travail distant en définissant une dépendance du travail local sur le travail reflet qui :

- Pointe vers l'instance de travail distant.
- Est définie sur un poste de travail local qui pointe vers le moteur distant sur lequel le travail distant est défini.

La dépendance croisée sur l'instance du travail distant est libérée lorsque la dépendance locale sur le travail reflet est libérée.

Surveillance d'une résolution de dépendance croisée dans le plan de production

Les travaux reflet sont ajoutés au plan comme suit :

- Lors de l'exécution si l'une des situations suivantes se produit :
 - Une définition de travail reflet est soumise à l'aide de la commande **sbj**.
 - Un flot de travaux contenant une définition de travail reflet est soumis à l'aide de la commande **sbs**.

Remarque : Lorsque vous soumettez un travail reflet, spécifiez un flot de travaux de destination avec une heure planifiée connue pour contrôler de façon plus efficace l'instance de travail distante qui sera liée.

- Lorsque le plan de préproduction est étendu ou créé et que son cadre temporel inclut l'heure planifiée du travail reflet.

Lorsqu'une instance de travail reflet est ajoutée au plan, vous pouvez commencer à surveiller son état.

Modification de l'état du travail reflet après que la liaison est établie

La figure 43 résume comment l'état d'un travail reflet est modifié après que la liaison est établie.

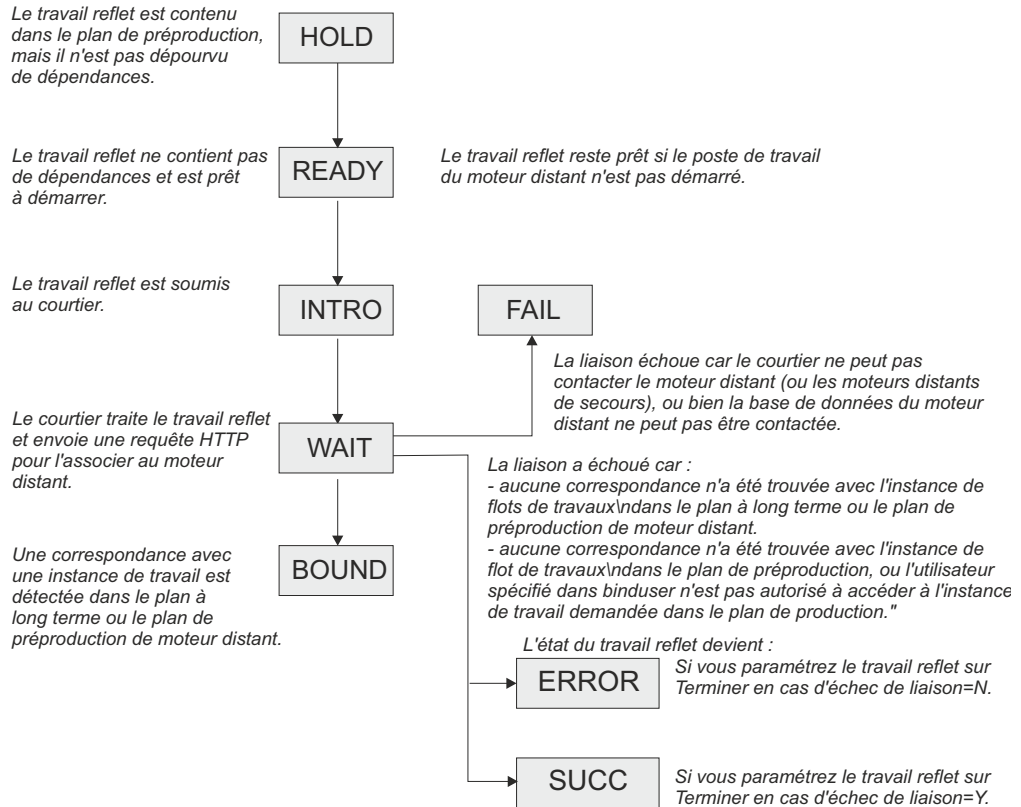


Figure 43. Transition de l'état d'un travail reflet jusqu'à ce que la liaison soit établie

Comme pour tout autre travail, l'état initial du travail reflet est HOLD et devient READY lorsque le travail est libéré des dépendances et prêt à démarrer.

Le planificateur envoie ensuite une demande HTTP sur le moteur distant contenant à la fois les informations permettant d'identifier le travail reflet dans le plan de production local et les informations permettant d'identifier de manière unique l'instance du travail distant à lier dans le plan du moteur distant, y compris les critères de correspondance. Le planificateur doit également être notifié de l'état de l'instance de travail distant liée.

Le planificateur tente de contacter le moteur distant, à intervalles réguliers, jusqu'à l'expiration d'un délai spécifique. Si, d'ici là le moteur distant n'a pas pu être atteint, l'état du travail reflet est défini sur FAIL. Pour modifier le délai et l'intervalle, spécifiez une valeur en secondes pour MaxWaitingTime et StatusCheckInterval dans le fichier `REP_BASE_TDWB/config/ResourceAdvisorConfig.properties`, puis redémarrez le courtier.

Si le plan de préproduction n'existe pas sur le moteur distant lorsque la demande de liaison est reçue, l'état du travail reflet distribué reste WAIT jusqu'à ce que la génération du plan de préproduction soit terminée et que la demande de liaison soit traitée. Cela peut se produire, par exemple, lorsque le plan de préproduction est recréé entièrement sur le moteur distant.

Pour plus d'informations sur la raison de l'échec (FAIL) de l'état du travail reflet, voir «Affichage des raisons de l'état d'échec du travail reflet (FAIL)», à la page 859.

Lorsque le moteur distant reçoit la demande HTTP, il tente d'identifier l'instance de flot de travaux à utiliser pour la liaison dans son plan ; il s'agit du plan de préproduction si le moteur distant est distribué, ou du plan à long terme si le moteur distant est z/OS. La définition du flot de travaux doit contenir la définition du travail distant à lier.

Pour plus d'informations sur l'établissement de la correspondance dans un plan de moteur distant distribué, voir «Liaison d'un travail reflet distribué».

Pour plus d'informations sur l'établissement de la correspondance dans un plan de moteur distant z/OS, voir «Liaison d'un travail reflet z/OS», à la page 854.

Liaison d'un travail reflet distribué

Si le moteur distant est un gestionnaire de domaine maître ou un gestionnaire de domaine maître se sauvegarde IBM Workload Scheduler, la recherche de l'instance de travail distant à lier est effectuée dans le plan de préproduction. Les instances de travail distant distribué, appartenant aux flots de travaux JOBS ou USERJOBS, ne sont pas impliquées dans le processus de liaison. Toutefois, les travaux distants déplacés dans USERJOBS après la liaison continuent d'envoyer des notifications de changement de statut.

L'intervalle correspondant, excepté pour le critère de correspondance précédent le plus proche qui ne nécessite pas de calcul d'intervalle, est calculé sur le moteur distant à l'aide des paramètres spécifiés dans la définition du travail reflet distribué.

Par exemple, lorsque le critère de correspondance **sameDay** est spécifié, le jour auquel il est fait référence est le jour spécifié sur le moteur distant en termes de [*startOfDay*, *startOfDay*+23:59].

Lors de l'utilisation de critères de correspondance basés sur intervalle, la demande HTTP envoyée sur le moteur distant contient les informations suivantes pour permettre au moteur distant de calculer l'intervalle de correspondance :

Pour les critères de correspondance d'intervalle absolus :

Les valeurs *hhmm*, *HHMM* et, facultativement, *d* et *D*, spécifiés dans la clause :

```
<dshadow:matching>  
<dshadow:absolute from="hhmm [+/-d day[s]]" to="HHMM [+/-D day[s]]"/>  
</dshadow:matching>
```

Les valeurs de limite sont *hhmm* -6 days et *HHMM* +6 days.

Le fuseau horaire utilisé pour les critères de correspondance est celui du travail reflet.

Pour les critères de correspondance relatifs :

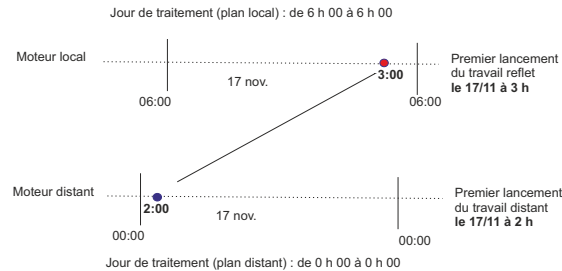
L'heure planifiée du travail reflet et les valeurs [*hh*]*hmm* et [*HH*]*HMM* spécifiées dans la clause :

```
<dshadow:matching>  
<dshadow:relative from="+/- [hh]hmm" to="+/- [HH]HMM" />  
</dshadow:matching>
```

Les valeurs limites sont +/-167:59 heures.

Par exemple, pour créer un travail reflet qui correspond à une instance de travail distant dont le début le plus tôt est le 17 novembre à 2:00 AM, vous pouvez spécifier l'un des critères de correspondance suivants :

- **Même date planifiée**
- **Dans un intervalle absolu** en spécifiant comme décalage : **1 jour avant la première date de début.**



L'instance de travail distant pour laquelle une correspondance doit être trouvée est identifiée sur le moteur distant en fonction des règles mentionnées pour les dépendances de prédécesseur/successeur externes. Pour plus de détails sur les critères de résolution des dépendances de prédécesseur/successeur externes, voir «Gestion des dépendances externes de prédécesseur/successeur des travaux et des flots de travaux», à la page 71.

Pour plus d'informations sur la définition des travaux reflet, voir «Définition de travail», à la page 187.

Liaison d'un travail reflet z/OS

Si le moteur distant est un contrôleur IBM Workload Scheduler for z/OS, la recherche de l'instance distante à lier est effectuée comme suit :

- L'instance est d'abord recherchée dans le plan à long terme dans la partie de l'intervalle de liaison qui suit l'heure de fin du plan en cours et précède l'heure planifiée du travail reflet.
- Si aucune instance n'est trouvée, l'instance est recherchée dans le plan en cours, dans la partie de l'intervalle de liaison qui précède la fin du plan en cours.

Remarque : Si le contrôleur distant reçoit une demande de liaison avec l'URI de notification du client qui n'est pas définie parmi les destinations HTTP, la demande de liaison est supprimée et le message EQQHT62W est consigné dans le MLOG.

Les sections suivantes décrivent les scénarios qui peuvent se produire lors de la liaison d'un travail reflet z/OS ayant les éléments suivants :

- Heure planifiée : 18:00
- Informations de travail distant :
 - ID d'application : **JS2**
 - Numéro d'opération : **OP2**

Dans les figures :

- La boîte blanche indique l'intervalle de temps couvert par le plan à long terme.
- La boîte gris clair indique l'intervalle de temps couvert par le plan en cours.
- La boîte gris foncé indique l'intervalle dans le plan de moteur distant durant lequel l'instance de travail à lier doit être recherchée.
- L'occurrence **JS2** mise en évidence en gras est l'instance sélectionnée pour la liaison.

Scénario 1 : l'intervalle du plan en cours contient l'heure planifiée du travail reflet et des occurrences JS2 existent.

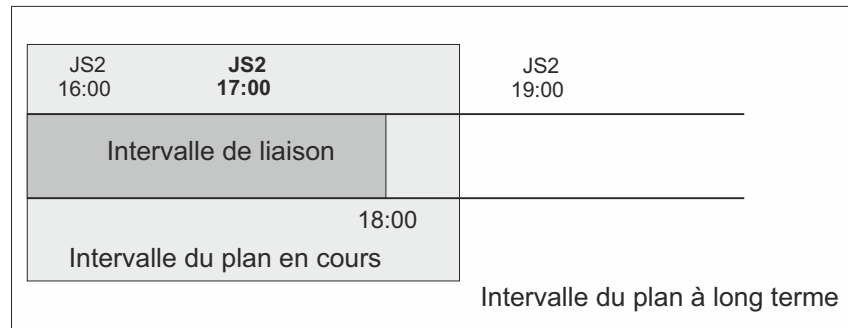


Figure 44. Instance à lier si l'heure planifiée du travail reflet est incluse dans l'intervalle du plan en cours

La figure 44 indique, mise en évidence en gras, l'instance **JS2** qui précède de la façon la plus proche l'heure planifiée du travail reflet. Cette instance est sélectionnée pour la liaison car l'heure planifiée est contenue dans le plan en cours. Le travail reflet et l'instance de travail distant sont associés. Si, par la suite, une nouvelle instance de **JS2** qui précède de la façon la plus proche l'heure planifiée du travail reflet est soumise ad hoc dans le plan du moteur distant, la correspondance avec l'instance **JS2** sélectionnée pour la liaison n'est *pas* modifiée.

A ce stade, l'une des situations suivantes peut se produire :

L'instance JS2 sélectionnée contient OP2.

La liaison avec **OP2** appartenant à **JS2** est établie et une notification contenant :

- Les informations de travail distant identifiant l'instance **OP2** dans le plan de moteur distant
- L'état en cours de cette instance **OP2**

est renvoyée, l'instance du travail distant est mise à jour avec les informations de travail distant et son état est mis à jour en conséquence.

L'instance JS2 sélectionnée ne contient plus OP2 parce qu'il a été supprimé et qu'un plan quotidien l'a supprimé du plan en cours ou parce qu'il n'a jamais été contenu dans JS2.

La liaison échoue. Une notification informant que la liaison a échoué est renvoyée et l'état du travail reflet est mis à jour selon ce que vous avez défini dans la zone **Terminer en cas d'échec de liaison**.

L'instance JS2 sélectionnée contient l'OP2 qui a été supprimée mais pas encore retirée du plan en cours.

La liaison est établie et une notification informant de l'état d'exécution réussi est renvoyée. L'instance du travail reflet est marquée comme SUCC. Ses successeurs peuvent démarrer.

Scénario 2 : l'intervalle du plan en cours contient l'heure planifiée du travail reflet, l'instance JS2 qui précède l'heure planifiée du travail reflet de la façon la plus proche existe dans le plan à long terme mais a été annulée du plan en cours.

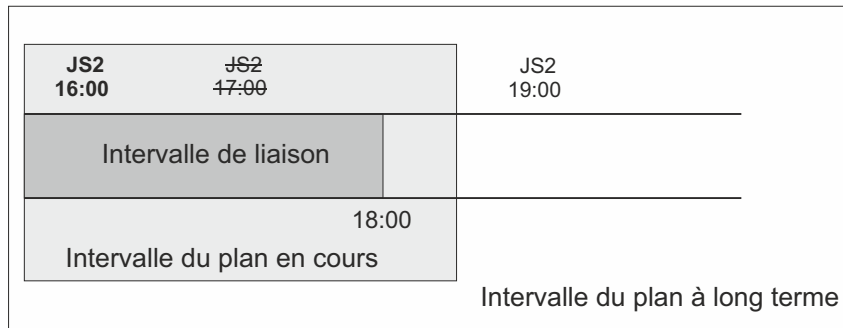


Figure 45. Instance à lier si l'instance qui précède de la façon la plus proche l'heure planifiée du travail reflet existe dans le plan à long terme mais a été annulée du plan en cours.

La figure 45 indique, mise en évidence en gras, l'instance **JS2** qui est sélectionnée pour la liaison, parce que l'occurrence qui correspondait le mieux a été supprimée.

La liaison avec **OP2** appartenant à **JS2** est établie et une notification contenant :

- Les informations de travail distant identifiant l'instance **OP2** dans le plan de moteur distant
- L'état en cours de cette instance **OP2**

est renvoyée, l'instance du travail distant est mise à jour avec les informations de travail distant et son état est mis à jour en conséquence.

Scénario 3 : l'intervalle du plan en cours contient l'heure planifiée du travail reflet mais aucune occurrence JS2 n'existe.

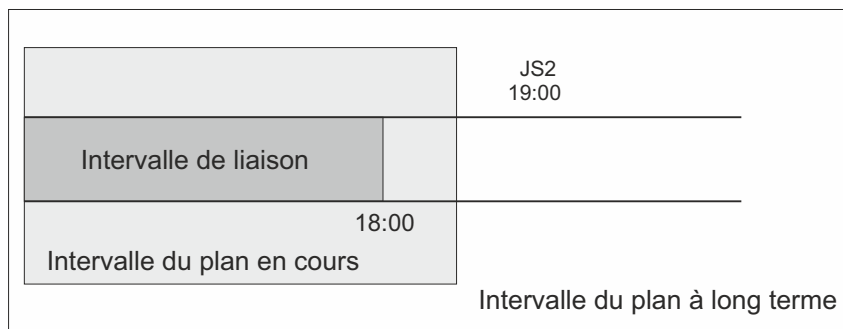


Figure 46. L'heure planifiée du travail reflet est incluse dans le plan en cours mais aucune instance à lier n'existe

La figure 46 indique que l'instance **JS2** qui précède l'heure planifiée de travail reflet de la façon la plus proche n'existe pas.

La liaison échoue. Une notification informant que la liaison a échoué est renvoyée et l'état du travail reflet est mis à jour selon ce que vous avez défini dans la zone **Terminer en cas d'échec de liaison**.

Scénario 4 : L'intervalle du plan à long terme contient l'heure planifiée du travail en cours et le plan en cours ne comprend pas encore l'instance JS2 immédiatement précédente.

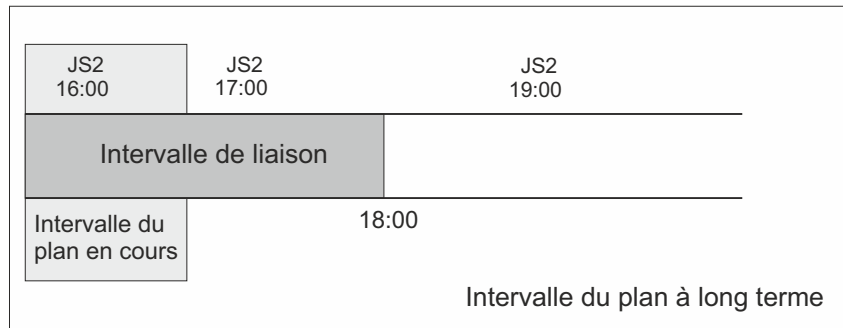


Figure 47. L'instance à lier existe mais n'est pas encore incluse dans le plan en cours

La figure 47 indique l'instance **JS2** qui peut être associé avec le travail reflet, même lorsque le travail **JOB2** n'est pas encore présent dans le plan en cours.

Une notification informant que la liaison est établie est renvoyée et l'état du travail reflet est défini par **BOUND**.

Scénario 5 : l'intervalle du plan à long terme ne contient toujours pas l'heure planifiée du travail reflet.

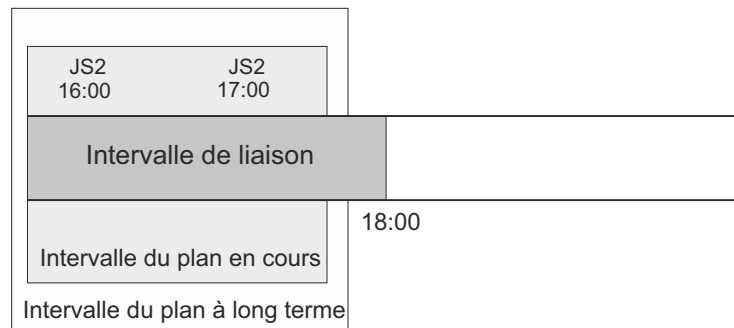


Figure 48. L'intervalle du plan à long terme ne contient toujours pas l'heure planifiée du travail reflet.

La figure 48 indique qu'aucune instance **JS2** ne peut être associée au travail reflet car, jusqu'au moment où le plan à long terme inclut l'heure planifiée du travail reflet, des instances **JS2** précédentes plus proches peuvent encore être ajoutées.

Dans ce cas, la demande de liaison est mise en attente jusqu'à ce que le plan à long terme soit étendu pour inclure l'heure planifiée du travail reflet. Avant cela, l'état du travail reflet reste **WAIT**.

Modification de l'état du travail reflet après que la liaison est établie

Lorsqu'une liaison est établie, le moteur distant renvoie une notification HTTP contenant l'état de la liaison et, si la liaison réussit, les informations identifiant l'instance de travail distant liée. Ces informations apparaissent dans les détails de l'instance du travail reflet.

Selon le type de moteur distant, les informations suivantes sur l'instance de travail distant apparaissent dans les propriétés de travail reflet :

Le type de moteur distant est distribué

- Nom du flot de travaux
- Heure planifiée
- Poste de travail du flot de travaux
- Nom du travail

Le type de moteur distant est z/OS

- ID d'application
- Heure planifiée
- Numéro d'opération
- Poste de travail
- Nom de travail, s'il a été défini sur le moteur distant.

Lorsque l'instance de travail reflet est mappée vers une instance de travail distant existante, des notifications sur les modifications d'état de travail sont envoyées de façon asynchrone à partir du moteur distant. Elles servent à mapper la transition d'état de travail distant vers la transition d'état de travail reflet. Le mécanisme de stockage et d'acheminement garantit la livraison des messages et la récupération en cas de panne. La figure 49 présente comment l'état d'un travail reflet distribué change, depuis le moment où une liaison est établie jusqu'à ce que l'état du travail reflet devienne SUCC ou ERROR. Seuls les états SUCC et ERROR sont considérés comme l'état final du travail reflet.

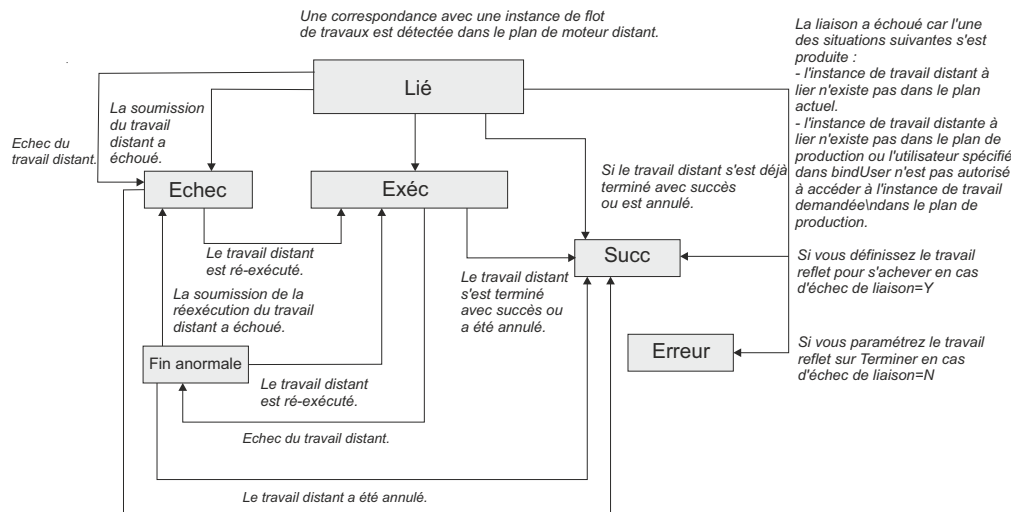


Figure 49. Chaîne de transition d'état de travail reflet après que la liaison a été établie

Si l'instance de travail distant est déjà terminée lorsque la correspondance est trouvée, l'état du travail reflet devient immédiatement SUCC.

Pour plus d'informations sur la raison de l'échec (FAIL) de l'état du travail reflet, voir «Affichage des raisons de l'état d'échec du travail reflet (FAIL)», à la page 859.

Lorsque l'état du travail reflet satisfait à la règle de dépendance, la dépendance du travail local sur le travail reflet est résolue et la dépendance croisée du travail local sur le travail distant est également résolue.

Affichage des raisons de l'état d'échec du travail reflet (FAIL)

L'état du travail reflet peut être FAIL dans une des situations suivantes :

- La soumission du travail reflet a échoué.
- La soumission du travail distant a échoué.

Pour déterminer la raison pour laquelle le statut du travail reflet est FAIL, consultez le journal du travail reflet soit en exécutant la commande **showjobs** avec l'option **;stdlist**, soit en cliquant sur **Journal du travail...** pour l'instance du travail reflet dans la vue **Surveillance des travaux** de Dynamic Workload Console.

Etat d'un travail reflet lors de la récupération ou la réexécution d'un travail distant

Une fois que la liaison est établie, il peut arriver que le travail distant lié soit réexécuté. Dans ce cas, l'état du travail reflet reflète l'état du travail réexécuté. L'état du travail reflet reste EXEC tant que la récupération du travail distant est en cours.

L'état du travail reflet est mis à jour uniquement lorsque le travail distant atteint l'un des états suivants :

ABEND

Lorsque le travail distant ne réussit pas à s'exécuter.

SUCC Lorsque le travail distant réussit.

FAIL Lorsque la soumission du travail distant échoue.

Vous pouvez consulter plus de détails sur le travail distant dans les propriétés du travail reflet. Pour afficher ces détails :

- Exécutez la commande **conman showjobs** avec l'option **props** sur le travail reflet.
- Accédez au panneau des propriétés du travail reflet dans Dynamic Workload Console.

Application du report aux dépendances croisées

Le report fonctionne de la même manière avec les travaux reflet qu'avec les autres types de travail. Les travaux reflet avec le statut **WAIT** et **BOUND** sont traités de la même manière que les travaux avec le statut **EXEC**. Les travaux reflets avec le statut **ERROR** sont traités de la même manière que les travaux avec le statut **FAIL** ou **ABEND**.

L'état d'un travail reflet, lié à un travail distant qui n'est pas reporté, est défini sur **ERROR** lorsque le plan de production distant est étendu.

Remarque : Une meilleure pratique consiste à utiliser les dépendances croisées avec le report de flots de travaux sur les environnements de planification distribués et distants.

Pour plus d'informations sur l'option globale **carryStates**, voir le *Guide d'administration*.

Gestion des travaux reflet dans le plan de production

Selon l'état du travail reflet, vous pouvez exécuter les commandes suivantes :

- Kill** Vous pouvez arrêter un travail reflet dont l'état est BOUND, EXEC ou WAIT. L'association établie par la liaison avec le travail distant est annulée automatiquement et l'état du travail reflet est défini sur ABEND avec le code retour 0.
- Rerun** Vous pouvez réexécuter un travail reflet dont l'état est ABEND, ERROR, SUCC ou FAIL. Lorsque vous réexécutez un travail reflet, une nouvelle demande de liaison est déclenchée.

Chapitre 24. Gestion d'un environnement dynamique IBM i

Gestion des agents IBM i dans un environnement dynamique et planification des travaux avec options avancées sur les agents IBM i.

Définition des agents sur les systèmes IBM i

Pour commencer à planifier des travaux sur les agents IBM i, l'agent doit se trouver dans le réseau IBM Workload Scheduler. A la fin du processus d'installation, l'agent est automatiquement enregistré dans la base de données IBM Workload Scheduler.

Vous pouvez vérifier l'existence de la définition de poste de travail de l'agent installé sur un système IBM i soit à l'aide de Dynamic Workload Console, soit à l'aide de la ligne de commande **composer**.

Pour plus d'informations sur l'utilisation de la console pour visualiser les définitions de poste de travail, voir *Dynamic Workload Console - Guide d'utilisation*, section sur la modification de définitions de poste de travail.

Pour plus d'informations sur l'utilisation de l'interface de ligne de commande pour visualiser des définitions de poste de travail, voir «Définition de poste de travail», à la page 166.

Pour inclure l'agent IBM i dans le plan, voir *IBM Workload Scheduler - Guide de planification et d'installation : Partie 3. IBM Workload Scheduler sur IBM i - Configuration d'un agent dynamique*.

Définition des travaux sur les systèmes IBM i

Sur les agents IBM i, vous pouvez définir les types de travail suivants avec options avancées :

Travaux de services Web

Pour définir des travaux de services Web, voir «travaux de services Web», à la page 618.

Travaux de transfert de fichiers

Pour définir des travaux de transfert de fichier, voir «Travaux de transfert de fichiers», à la page 598.

Travaux J2EE

Pour définir des travaux J2EE, voir «Travaux J2EE», à la page 625.

Travaux de base de données

Pour définir des travaux de base de données, voir «Travaux de base de données», à la page 612.

Travaux Java

Pour définir des travaux Java, voir «travaux Java», à la page 624.

Travaux exécutables

Pour définir des travaux exécutables, voir «Travaux exécutables», à la page 588.

Travaux IBM i

Pour définir des travaux IBM i qui exécutent des commandes natives des systèmes d'exploitation IBM i, voir «Travaux IBM i», à la page 585.

Travaux de commande distante

Pour définir des travaux de commande distante, voir «Travaux de type commande distante», à la page 582.

Travaux TPM (Provisioning)

Pour définir des travaux TPM (Provisioning), voir «Travaux IBM SmartCloud Provisioning», à la page 592.

Travaux IBM WebSphere MQ

Pour définir des travaux IBM WebSphere MQ, voir la section sur le plug-in de travail pour IBM WebSphere MQ dans *Planification des applications avec IBM Workload Automation*.

Travaux IBM Sterling Connect:Direct

Pour définir des travaux IBM Sterling Connect:Direct, voir la section sur le plug-in de travail pour IBM Sterling Connect:Direct dans *Planification des applications avec IBM Workload Automation*.

Vous pouvez définir des travaux avec options avancées sur un agent IBM i soit à l'aide de Dynamic Workload Console, soit à l'aide de la ligne de commande **composer**.

Pour plus d'informations sur la procédure de définition des travaux IBM i, voir les sections sur les étapes préalables de création des types de travaux avec options avancées et sur la création de définitions de travail dans *IBM Dynamic Workload Console - Guide d'utilisation*.

Pour plus d'informations sur l'utilisation de l'interface de ligne de commande pour créer des définitions de travail, voir «Définition de travail», à la page 187.

Gestion des agents sur les systèmes IBM i

Vous pouvez utiliser IBM Workload Scheduler sur les agents IBM i pour démarrer et arrêter uniquement les processus d'agent. Pour plus d'informations sur le démarrage et l'arrêt des agents IBM i, voir «Démarrage et arrêt des agents sur les systèmes IBM i».

Pour gérer l'agent IBM i, utilisez les utilitaires décrits dans «Utilisation des commandes d'utilitaire pour les agents sur les systèmes IBM i», à la page 863.

Démarrage et arrêt des agents sur les systèmes IBM i

Vous pouvez utiliser IBM Workload Scheduler sur les agents IBM i pour démarrer et arrêter uniquement les processus d'agent.

Démarrage des agents sur les systèmes IBM i :

Utilisez l'utilitaire **StartUpLwa**.

Pour plus d'informations sur l'utilitaire permettant de démarrer les agents sur IBM i, voir «StartUpLwa - Démarrage de l'agent», à la page 723.

Arrêt des agents sur les systèmes IBM i :

Utilisez l'utilitaire **ShutDownLwa**.

Pour plus d'informations sur l'utilitaire permettant d'arrêter les agents sur IBM i, voir «ShutDownLwa - Arrêt de l'agent», à la page 722.

Utilisation des commandes d'utilitaire pour les agents sur les systèmes IBM i

Pour les agents sur les systèmes IBM i, vous pouvez utiliser les utilitaires suivants :

param Pour utiliser l'utilitaire param, voir «param», à la page 738.

twstrace

Pour utiliser l'utilitaire twstrace, voir «twstrace», à la page 753.

resource

Pour utiliser l'utilitaire resource, voir «resource», à la page 741.

cpuinfo

Pour utiliser l'utilitaire cpuinfo, voir «cpuinfo», à la page 682.

version

Pour utiliser l'utilitaire version, voir «version», à la page 723.

Planification des travaux sur les systèmes IBM i

Lors de la planification d'un travail sur les systèmes IBM i, le travail lance une commande native qui peut être soit une commande système, soit une commande utilisateur. La commande native contient une commande système SBMJOB, qui lance un travail par lots. La commande native démarre un ou plusieurs travaux par lots. Les travaux par lots peuvent être surveillés seulement s'ils sont démarrés par la commande native. L'agent de surveillance IBM i peut surveiller un maximum de 130 travaux par lots.

Vous pouvez indiquer le nom de la file d'attente dans laquelle le composant agent de surveillance s'exécute à l'aide de la propriété **MonitorQueueName** dans la section du programme de lancement de travaux natif du fichier JobManager.ini. Si vous ne spécifiez pas cette propriété, la file d'attente par défaut (QBATCH) est utilisée.

Pour plus d'informations, reportez-vous à la section relative à la configuration des propriétés du programme de lancement de travaux natif [NativeJobLauncher] dans le *Guide d'administration*.

Vous pouvez utiliser Dynamic Workload Console et la ligne de commande **conman showjobs** pour vérifier si un travail IBM i attend une réponse à un message. Un travail IBM i qui attend une réponse de message est doté du statut SUSP (suspendu). Ce statut indique que le travail s'exécute tout en attendant une entrée. Une fois l'entrée reçue, le statut du travail passe à EXEC (en cours d'exécution).

Pour plus d'informations sur les statuts de travail, voir la section relative à la description des statuts et au mappage des travaux répartis dans *Dynamic Workload Console - Guide d'utilisation*.

Si un travail IBM i attend une réponse, vous pouvez visualiser le texte du message et la réponse associée. Ces informations sont écrites dans le journal des travaux correspondant de IBM Workload Scheduler, de sorte que l'opérateur IBM Workload Scheduler connaisse le message exact que le travail IBM i attend.

Si un travail IBM i attend une réponse à un message, vous pouvez répondre à ce message directement depuis **Monitor Workload** de Dynamic Workload Console. Le travail à l'état SUSP (suspendu) nécessite votre attention et souhaite afficher des informations supplémentaires. Une fenêtre en incrustation affiche le message qui attend votre réponse. Répondez à ce message, puis sélectionnez l'une des actions suivantes :

Action Transférer

Permet de transférer votre réponse. Un message dans la fenêtre en incrustation confirme que votre réponse a été envoyée avec succès.

Action Annuler

Permet d'annuler votre réponse. La fenêtre en incrustation se ferme.

Remarque : Pour un affichage correct de la fenêtre en incrustation qui affiche le message attendant votre réponse, votre gestionnaire de domaine dynamique doit être à la version 9.3.0.2.

Vous pouvez même définir les règles standard d'automatisation de la réponse aux messages en attente. Lors de la définition d'un travail IBM i à l'aide de Dynamic Workload Console ou de la ligne de commande **composer**, spécifiez la liste des messages dont vous souhaitez configurer une réponse automatique. Pour chaque message, spécifiez :

ID message

Identificateur du message.

Texte du message

Le texte du message.

Réponse au message

Réponse automatisée que vous souhaitez définir.

Dans le journal des travaux de sortie, vous pouvez voir les messages mis en file d'attente par le système IBM i et les réponses envoyées automatiquement par le travail.

Pour plus d'informations, voir la section relative à la définition des travaux IBM i dans *Guide d'utilisation et de référence*.

L'agent journal des travaux et la variable d'environnement TWSASPOOLS

Par défaut, toutes les informations relatives à l'exécution des travaux sont stockées dans l'agent journal des travaux. La plupart de ces informations se composent de fichiers spoules. Pour sélectionner les types de fichier spoule que vous souhaitez inclure dans le journal des travaux de l'agent, utilisez la variable système **TWSASPOOLS**, qui fonctionne au niveau de l'agent IBM i pour n'importe quel travail à soumettre.

La variable système **TWSASPOOLS** force l'agent IBM i à ignorer tous les fichiers spoule ou à inclure un ou plusieurs de ces fichiers.

Sur l'agent IBM i, créez une nouvelle variable d'environnement au niveau système nommée **TWSASPOOLS** et définissez-la dans une liste des types de fichier spoule à inclure. La liste doit commencer par le marqueur **SPOOLS:**.

Par exemple, pour forcer l'agent IBM i à ignorer tous les fichiers spoules, créer la variable **TWSASPOOLS** comme suit.

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE(SPOOLS:) LEVEL(*SYS)
```

où la liste après le marqueur **SPOOL:** est vide. Dans ce cas, tout rapport de journal des travaux de l'agent pour l'agent IBM i est limité au rapport d'activités que le

moniteur d'agent produit pour suivre la soumission et la surveillance de l'action, ainsi qu'au journal des travaux IBM i du moniteur d'agent qui est toujours ajouté à la fin du journal des travaux de l'agent.

Pour autoriser l'agent IBM i à inclure seulement les types de fichier spoule QPRINT et QJOBLOG, c'est-à-dire, tout fichier spoule produit par des instructions **printf** dans tout programme ILE-C et tout journal des travaux produit, créez la variable TWSASPOOLS comme suit :

```
ADDENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QJOBLOG') LEVEL(*SYS)
```

Si la variable TWSASPOOLS existe déjà, modifiez-la de la manière suivante :

```
CHGENVVAR ENVVAR(TWSASPOOLS) VALUE('SPOOLS: QPRINT QJOBLOG') LEVEL(*SYS)
```

Si tout paramètre VALUE parameter est défini dans une chaîne incorrecte, l'agent IBM i ignore l'option de variable environnement TWSASPOOLS. Vous pouvez créer et modifier la variable d'environnement tandis que l'agent IBM i est actif mais aucune charge de travail ne doit être en cours d'exécution.

Surveillance des travaux enfants sur les agents IBM i

Lorsque vous soumettez une commande sur un agent IBM i, la commande risque de démarrer un ou plusieurs travaux par lots. L'agent IBM i surveille ces travaux par lots, également appelés travaux enfants.

Lors de la recherche et de la surveillance de travaux enfants démarrés, l'agent IBM i utilise un pourcentage élevé de son temps de traitement.

Si vous savez que votre planification des travaux ne démarre aucun travail enfant ou si vous ne vous préoccupez pas de la surveillance des travaux enfants, vous pouvez ordonner à l'agent IBM i de ne pas chercher et surveiller des travaux enfants, et donc de ne pas améliorer les performances de l'agent.

Vous pouvez exclure la surveillance des travaux enfants soit au niveau de l'agent pour toutes les commandes, soit au niveau de la définition de travail pour une commande unique. Si vous souhaitez activer la surveillance des travaux enfants pour certaines commandes soumises uniquement, vous pouvez définir cette option au niveau de la définition de travail pour une commande unique.

Vous pouvez exécuter l'une ou les deux procédures suivantes pour exclure ou inclure la surveillance des travaux enfants :

Exclure les travaux enfants de la surveillance des travaux au niveau de l'agent

Par défaut, les travaux enfants sont surveillés. Vous pouvez exclure des travaux enfants de la surveillance des travaux pour toutes les commandes soumises en créant la variable d'environnement système TWS_NOCHILDS à l'aide de la commande de système IBM i suivante :

```
ADDENVVAR ENVVAR(TWS_NOCHILDS) LEVEL(*SYS)
```

Si l'agent IBM i trouve TWS_NOCHILDS sur le système IBM i, il ne surveillera pas les travaux enfants pour aucune commande soumise.

Exclure ou inclure les travaux enfants de la surveillance des travaux au niveau de la définition de travail

Vous pouvez exclure ou inclure des travaux enfants de la surveillance des travaux pour un travail spécifique en utilisant :NOCHILDS ou :CHILDS comme jetons de fin de la chaîne de commande pour la commande en question.

- Si vous ajoutez le jeton de fin :NOCHILDS à la fin de la commande native que vous soumettez, l'agent IBM i ignore tous les travaux enfants démarrés par la commande.
- Si vous ajoutez le jeton de fin :CHILDS à la fin de la commande que vous soumettez, l'agent IBM i trouve et surveille tous les travaux enfants démarrés par la commande.

Remarque : Le paramètre au niveau de la définition de travail remplace le paramètre au niveau de l'agent.

Lorsqu'elle est soumise, la commande de système **SBMJOB** démarre toujours un travail par lots. N'essayez pas d'exclure la surveillance des travaux, car si l'agent IBM i trouve la commande SBJJOB dans la définition de travail, il supprime et ignore le jeton de fin :CHILDS ou :NOCHILDS dans la définition de travail et ignore également le paramètre de la variable système TWS_NOCHILDS.

Exemples

Pour surveiller les travaux enfants démarrés lors de l'exécution du programme PAYROLL, définissez la commande suivante dans la définition de travail :

- Si la variable système TWS_NOCHILDS est définie sur le système IBM i :
CALL PGM(MYLIB/PAYROLL):CHILDS
- Si la variable système TWS_NOCHILDS n'est pas définie sur le système IBM i :
CALL PGM(MYLIB/PAYROLL)

Pour ne pas surveiller les travaux enfants démarrés lors de l'exécution du programme MYSCHEDULE, définissez la commande suivante dans la définition de travail :

- Si la variable système TWS_NOCHILDS n'est pas définie sur le système IBM i :
CALL PGM(MYLIB/MYSCHEDULE):NOCHILDS
- Si la variable système TWS_NOCHILDS est définie sur le système IBM i :
CALL PGM(MYLIB/MYSCHEDULE)

Remarque : La commande SBJJOB démarre toujours un travail enfant. L'agent IBM i surveille le travail enfant même si vous définissez une commande SBJJOB, comme dans la définition de travail suivante :

```
SBMJOB CMD(CALL PGM(MYLIB/USERPGM)):NOCHILDS
```

Informations sur la surveillance des travaux enfants dans les journaux des travaux de l'agent IBM i

Si vous avez inclus la surveillance des travaux enfants sur les agents IBM i, comme décrit dans la section «Surveillance des travaux enfants sur les agents IBM i», à la page 865, vous pouvez visualiser les informations relatives à la surveillance des travaux enfants dans le journal des travaux de l'agent IBM i.

Exemples

L'exemple ci-dessous présente les informations relatives à la surveillance des travaux enfants incluse au niveau du travail pour le travail CHILDLONG_CHILD sur l'agent NC117025 :

```
Travail CHILDLONG_CHILD
```

```
Poste de travail (travail) NC117025
```


Flot de travaux AS400ENVSET

Poste de travail (flot de travaux) NC117025

```
=====
= JOB      : NC117025#AS400ENVSET[(1417 10/30/12),(AS400ENVSET)].CHILDLONG_CHILD
= TASK     : <?xml version="1.0" encoding="UTF-8"?>
<jSDL:jobDefinition xmlns:jSDL="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSDL"
xmlns:jSdLibmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jSdLibmi" name="ibmi">
  <jSDL:variables>
    <jSDL:stringVariable name="twS.jobstream.id">AS400ENVSET</jSDL:stringVariable>
    <jSDL:stringVariable name="twS.job.workstation">NC117025</jSDL:stringVariable>
    <jSDL:stringVariable name="twS.job.iawstz">201210301417</jSDL:stringVariable>
  </jSDL:variables>
  <jSDL:application name="ibmi">
    <jSdLibmi:ibmi>
      <jSdLibmi:IBMIParameters>
        <jSdLibmi:Task>
          <jSdLibmi:command>CALL PGM(MINERMA/SBM5JOBS) :CHILDS</jSdLibmi:command>
        </jSdLibmi:Task>
      </jSdLibmi:IBMIParameters>
    </jSdLibmi:ibmi>
  </jSDL:application>
  <jSDL:resources>
    <jSDL:orderedCandidatedWorkstations>
      <jSDL:workstation>805E5EAC1F5911E2B9DB6F8202778C47</jSDL:workstation>
    </jSDL:orderedCandidatedWorkstations>
  </jSDL:resources>
</jSDL:jobDefinition>
= TWSRCMAP :
= AGENT    : NC117025
= Numéro de travail : 760232858
= Tue Oct 30 14:16:31 CET 2012
=====
```

Le travail de surveillance soumis par l'agent dynamique est qualifié sous la forme :
JobName=DYNAMICMON JobUser=CLAUDIO JobNumber=361743
La chaîne de commande de l'utilisateur est la suivante
<CALL PGM(MINERMA/SBM5JOBS) :CHILDS>
2012/10/30 14:16:28.844 - Le travail de l'agent dynamique a soumis la commande utilisateur
CALL PGM(MINERMA/SBM5JOBS)

LES 5 TRAVAUX SUIVANTS ONT DEMARRE sous la commande utilisateur soumise
JobName=CLAUDIO JobUser=CLAUDIO JobNumber=361765
JobName=CLAUDIO JobUser=CLAUDIO JobNumber=361756
JobName=CLAUDIO JobUser=CLAUDIO JobNumber=361762
JobName=CLAUDIO JobUser=CLAUDIO JobNumber=361775
JobName=CLAUDIO JobUser=CLAUDIO JobNumber=361774

Message CPF1241 (Réussite) reçu sur MsgQueue CLAUDIO QUSRSYS
pour le travail CLAUDIO CLAUDIO 361765
Message CPF1241 (Réussite) reçu sur MsgQueue CLAUDIO QUSRSYS
pour le travail CLAUDIO CLAUDIO 361756
Message CPF1241 (Réussite) reçu sur MsgQueue CLAUDIO QUSRSYS
pour le travail CLAUDIO CLAUDIO 361762
Message CPF1241 (Réussite) reçu sur MsgQueue CLAUDIO QUSRSYS
pour le travail CLAUDIO CLAUDIO 361775
Message CPF1241 (Réussite) reçu sur MsgQueue CLAUDIO QUSRSYS
pour le travail CLAUDIO CLAUDIO 361774

*** Codes de FIN regroupés par le travail de surveillance ***

> Code de statut de FIN (Statut) : 0
> Code retour du PROGRAMME (Prc) : 0
> Code retour de l'Utilisateur (Urc) : 0
Code de référence d'unité récupéré via SYSAPI

2012/10/30 14:21:37.890 - Le travail de l'agent dynamique a mis fin à la surveillance
de la commande utilisateur

*** Le code retour pour la commande soumise est 0 ***

*** La commande utilisateur s'est terminée correctement ***

L'exemple ci-dessous présente le journal des travaux pour le travail CHILDLING_NC sur l'agent NC117025 lorsque la surveillance des travaux enfants est exclue au niveau du travail :

Travail CHILDLING_NC

Poste de travail (travail) NC117025

Flot de travaux AS400ENVSET

Poste de travail (flot de travaux) NC117025

```
=====
= JOB      : NC117025#AS400ENVSET[(1417 10/30/12),(AS400ENVSET)].CHILDLING_NC
= TASK     : <?xml version="1.0" encoding="UTF-8"?>
<jsd1:jobDefinition xmlns:jsdl="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdl"
xmlns:jsdlbmi="http://www.ibm.com/xmlns/prod/scheduling/1.0/jsdlbmi" name="ibmi">
  <jsdl:variables>
    <jsdl:stringVariable name="tw.jobstream.id">AS400ENVSET</jsdl:stringVariable>
    <jsdl:stringVariable name="tw.job.workstation">NC117025</jsdl:stringVariable>
    <jsdl:stringVariable name="tw.job.iawstz">201210301417</jsdl:stringVariable>
  </jsdl:variables>
  <jsdl:application name="ibmi">
    <jsdlbmi:ibmi>
      <jsdlbmi:IBMParameters>
        <jsdlbmi:Task>
          <jsdlbmi:command>CALL PGM(MINERMA/SBM5JOBS) :NOCHILDS</jsdlbmi:command>
        </jsdlbmi:Task>
      </jsdlbmi:IBMParameters>
    </jsdlbmi:ibmi>
  </jsdl:application>
</jsdl:resources>
  <jsdl:orderedCandidatedWorkstations>
    <jsdl:workstation>805E5EAC1F5911E2B9DB6F8202778C47</jsdl:workstation>
  </jsdl:orderedCandidatedWorkstations>
</jsdl:resources>
</jsdl:jobDefinition>
= TWSRCMAP :
= AGENT    : NC117025
= Numéro du travail : 760232857
= Tue Oct 30 14:17:01 CET 2012
=====
Le travail de surveillance soumis par l'agent dynamique est qualifié sous la forme :
  JobName=DYNAMICMON JobUser=CLAUDIO JobNumber=361817
La chaîne de commande de l'utilisateur est la suivante
  <CALL PGM(MINERMA/SBM5JOBS) :NOCHILDS>
2012/10/30 14:16:58.330 - Le travail de l'agent dynamique a soumis la commande utilisateur
  CALL PGM(MINERMA/SBM5JOBS)
Par choix de l'utilisateur, AUCUN travail démarré sous la commande soumise ne sera surveillé
*** Codes de FIN regroupés par le travail de surveillance ***
> Code de statut de FIN (Statut) : 0
> Code retour du PROGRAMME (Prc) : 0
> Code retour de l'Utilisateur (Urc) : 0
  Code de référence d'unité récupéré via SYSAPI

2012/10/30 14:17:10.220 - Le travail de l'agent dynamique a mis fin à la surveillance
de la commande utilisateur
*** Le code retour pour la commande soumise est 0 ***
*** La commande utilisateur s'est terminée correctement ***
```

Récupération du code retour agent

Le modèle de programmation IBM i avait pour origine un vieux modèle à orientation objet dans lequel les programmes communiquaient par transmission de messages et non par utilisation de code retour. L'introduction du modèle ILE

(Integrated Language Environment) a mené la définition de zones communes d'échange de données comme les codes retour dans le même environnement de travail : les codes retour utilisateur.

Pour de plus amples informations sur les codes retour utilisateur, voir «Contrôle de l'environnement de travail à l'aide du code retour utilisateur».

Lorsque l'agent IBM i vérifie qu'une commande soumise ou qu'un travail est achevé, il assigne un code retour au travail en fonction de l'état du travail achevé. Le code retour est défini en fonction du message d'achèvement de la commande ou du travail. Si la commande ou le travail se termine avec succès, le code retour est défini sur 0. Si la commande ou le travail ne s'est pas achevé avec succès, le code retour est réglé en fonction de la valeur de la gravité du message lié à l'exception qui a causé la terminaison anormale du travail. L'agent IBM i peut également définir le code retour sur la valeur du code retour utilisateur lorsqu'il est récupéré par la commande soumise. S'il est récupéré, le code retour utilisateur est utilisé comme valeur pour définir le code retour.

La valeur du code retour affectée au travail est incluse dans le journal des travaux de l'agent IBM i pour le travail et renvoyée à l'interface utilisateur du planificateur (panneaux WEB UI ou z/OS ISPF) sous la forme d'un code retour, pour des raisons de compatibilité avec les agents sur d'autres systèmes d'exploitation.

Contrôle de l'environnement de travail à l'aide du code retour utilisateur

Avec l'introduction du modèle ILE IBM i ILE, il est possible de récupérer une valeur retournée par un programme appelé au sein du même travail.

Lorsque l'agent de surveillance vérifie qu'une commande soumise est terminée, il récupère les codes de fin de travail suivants à l'aide d'un système API IBM i :

Code d'état de fin ou <Status> (0 en cas de succès)

Il indique si le système a envoyé une annulation contrôlée pour le travail. Les valeurs possibles sont les suivantes :

- 1** Le sous-système ou le travail est annulé.
- 0** Le sous-système ou le travail n'est pas annulé.
- vide** Le travail n'est pas en cours d'exécution.

Code retour de programme ou <Prc> (0000 en cas de succès)

Il indique le code achèvement du dernier programme (par exemple, un utilitaire de création-maintenance de fichiers ou un programme RPG ou COBOL appelé par le travail).

Si le travail n'inclut aucun programme, le code retour de programme est 0.

Code retour utilisateur ou <Urc> (0000 en cas de succès)

Il indique le code retour défini par l'utilisateur et mis en place par les constructions du langage évolué ILE. Par exemple, le code retour d'un programme écrit en langage C.

Il représente le code retour le plus récent défini par une unité d'exécution dans un travail.

Si la commande soumise est un appel à un programme ILE utilisateur renvoyant une valeur à la sortie, cette valeur peut être trouvée dans le code de référence d'unité du travail.

Vous pouvez décider des modalités de contrôle de l'environnement de travail des travaux que vous avez soumis en préparant les commandes pour qu'elles soient soumises comme des CALLS pour vos programmes ILS, où le flux interne est contrôlé et le statut de fin est décidé par le biais de valeurs de sortie correctes. Si un programme utilisateur se termine en erreur suite à un contrôle du débit incorrect, sans renvoyer de valeur, l'agent de surveillance ne définit pas le code retour comme un code de référence d'unité (Urc) mais suit les critères décrits dans «Récupération du code retour agent», à la page 868.

L'exemple suivant montre un programme utilisateur ILE C dans lequel sont lancés deux travaux par lots et où une valeur de 10 est renvoyée au demandeur, indépendamment du statut d'achèvement des travaux par lots.

```
=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
    int EnvVarRC=0;
    printf("issuing SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
    system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
    printf("issuing SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
    system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
    exit(10);
    return;
}
=====
```

Méthode alternative de définition du code retour utilisateur

Dans certains environnements IBM i, le système API récupérant le code retour utilisateur (Urc) du code de l'analyse de l'agent ne récupère pas la bonne valeur pour l'Urc. Il est donc déconseillé d'utiliser les systèmes d'interfaces de programme d'application IBM i pour récupérer le code retour utilisateur. Pour recevoir une valeur renvoyée par un programme appelé, il est préférable de fournir un paramètre destiné à recevoir cette valeur.

Même si l'analyse de l'agent parvient à récupérer le code retour utilisateur avec un système d'interfaces de programme d'application, une méthode alternative de récupération de code retour utilisateur a été implémentée dans le code de l'analyse de l'agent. La méthode de récupération alternative possède la logique suivante. La variable d'environnement du travail USERRC est créée et définie sur la valeur *INI* avant de soumettre la commande utilisateur. Lorsque la commande prend fin, l'agent de surveillance récupère le code retour de son utilisateur à l'aide du système APIs, mais elle vérifie également si la variable d'environnement du travail USERRC a été mise à jour au niveau du programme de l'utilisateur. Si une valeur autre que *INI* est trouvée, elle est considérée comme étant le code retour de l'utilisateur et la valeur récupérée à l'aide des API système est ignorée car le programme utilisateur a modifié la valeur de la variable d'environnement du travail USERRC.

La modification de la variable USERRC au niveau du programme utilisateur nécessite de modifier la valeur USERRC avant de quitter l'application du code utilisateur. Dans le cas d'ILE C, vous pouvez réaliser cette tâche à l'aide de l'instruction **putenv**, dans laquelle la récupération du code retour est définie.

L'exemple suivant montre comment le code utilisateur retransmet le code retour utilisateur à l'aide de la variable d'environnement `USERRC` du travail réservé de l'agent IBM i. Ce code a été obtenu à partir du code de l'exemple dans «Contrôle de l'environnement de travail à l'aide du code retour utilisateur», à la page 869 en remplaçant l'instruction **exit** par l'instruction **putenv**.

```
=====
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void main(int argc, char *argv[])
{
    int EnvVarRC=0;
    printf("issuing SBMJOB CMD(CALL MYLIB/DIVBY0)...\n");
    system("SBMJOB CMD(CALL MYLIB/DIVBY0)");
    printf("issuing SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT))...\n");
    system("SBMJOB CMD(WRKACTJOB OUTPUT(*PRINT)) LOG(4 0 *SECLVL)");
    EnvVarRC = putenv("USERRC=10");
    return;
}
=====
```

Annexe A. Automatisation de la charge de travail commandée par les événements, définitions des événements et des actions

Cette annexe documente les fournisseurs d'événements et d'actions utilisables pour l'automatisation de la charge de travail commandée par les événements et donne des définitions détaillées des événements et actions.

Fournisseurs d'événements et définitions

Cette section détaille les types d'événement des fournisseurs d'événement suivants :

- TWSObjectsMonitor
- FileMonitor
- TWSApplicationMonitor

Datetime

Contient à la fois la date et l'heure. Vous pouvez spécifier l'une de ces deux valeurs, ou les deux, dans le filtre.

Plusieurs prédicats de filtrage sont autorisés

Vous pouvez définir plusieurs prédicats de filtrage pour cette propriété. L'événement correspond à la condition d'événement si tous les prédicats sont satisfaits.

Valeurs multiples autorisées

Vous pouvez définir plusieurs valeurs pour cette propriété au sein d'un prédicat de filtrage unique. Le filtre est satisfait lorsqu'il existe une correspondance avec l'une des valeurs.

Caractères génériques autorisés

L'astérisque (*) et le point d'interrogation (?) sont les caractères génériques pris en charge.

Evénements TWSObjectsMonitor

Les événements TWSObjectsMonitor sont :

- Statut du flux de travaux modifié
- Travail Until
- Travail soumis
- Travail annulé
- Travail relancé
- Travail en retard
- Travail promu
- Niveau de risque du travail modifié
- Le travail a dépassé la durée maximale
- Le travail n'a pas atteint la durée minimale
- Statut du flot de travaux modifié
- Flot de travaux terminé
- Flot de travaux Until
- Flot de travaux soumis
- Flot de travaux annulé
- Flot de travaux en retard
- Statut du poste de travail modifié
- Statut du serveur d'applications modifié

- Lien vers le poste de travail enfant modifié
- Lien vers le poste de travail parent modifié
- Statut de l'invite modifié
- ProductAlert

Ces événements sont générés par batchman (ou mailman pour les postes de travail) et écrits dans un fichier de boîte aux lettres nommé monbox.msg. Les objets de planification sont surveillés comme suit : Les objets de planification sont surveillés comme suit :

- Les travaux sont contrôlés par le poste de travail sur lequel ils sont exécutés
- Les flots de travaux sont contrôlés par le gestionnaire de domaine maître
- Les postes de travail se contrôlent eux-mêmes
- Les invites locales sont contrôlées par le poste de travail exécutant le travail ou le flot de travaux qui a une dépendance sur l'invite
- Les invites globales sont contrôlées par le gestionnaire de domaine maître

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de chaque type d'événement.

Remarque : Utilisateurs PDF, les tables de paramètre ci-dessus constituent un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Utilisation des événements WorkstationStatusChanged

L'événement est démarré lorsqu'un poste de travail est démarré ou arrêté. Mais les différences opérationnelles suivantes existent en fonction du type de poste de travail qui est contrôlé :

- Pour un agent tolérant aux pannes, l'événement est envoyé lorsque le poste de travail est démarré ou arrêté.
- Pour un poste de travail Dynamic Workload Broker, l'événement est également envoyé lorsqu'il est lié ou non lié (lorsque ces commandes démarrent ou arrêtent également le poste de travail).
- Pour un poste de travail de pool dynamique, l'événement n'est jamais envoyé (même si Dynamic Workload Broker est arrêté) car il n'y a pas de contrôle pour ce type de poste de travail.

Exemples

La règle dans l'exemple suivant soumet le flot de travaux RJS_102739750 sur le poste de travail NC125102 dès que tous les travaux du flot de travaux RCF_307577430 du poste de travail NA022502 ont le statut RUNNING ou SUCCESSFUL.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="TWS_PLAN_EVENTS_JOB_STATUS_CHANGED" ruleType="filter" isDraft="no">
    <description>Event: Job Status Changed; Action: Submit job stream</description>
    <timeZone>Europe/Rome</timeZone>
    <validity from="2011-04-24" to="2012-04-24" />
    <activeTime start="00:00:00" end="12:00:00" />
    <eventCondition name="jobStatChgEvt1"
      eventProvider="TWSObjectsMonitor"
      eventType="JobStatusChanged">
```



```

<scope>* # JOBSTREAMVALUE . * [RUNNING, SUCCESSFUL]</scope>
  <filteringPredicate>
    <attributeFilter name="JobStreamWorkstation" operator="eq">
      <value>NA022502</value>
    </attributeFilter>
    <attributeFilter name="JobStreamName" operator="eq">
      <value>RCF_307577430</value>
    </attributeFilter>
    <attributeFilter name="JobName" operator="eq">
      <value>*</value>
    </attributeFilter>
    <attributeFilter name="Priority" operator="ge">
      <value>10</value>
    </attributeFilter>
    <attributeFilter name="Monitored" operator="eq">
      <value>true</value>
    </attributeFilter>
    <attributeFilter name="Status" operator="eq">
      <value>Running</value>
      <value>Successful</value>
    </attributeFilter>
    <attributeFilter name="Login" operator="eq">
      <value>TWS_user</value>
    </attributeFilter>
  </filteringPredicate>
</eventCondition>
<action actionProvider="TWSAction" actionType="sbs" responseType="onDetection">
  <description>Launch an existing TWS job stream</description>
  <scope>SBS NC125102#RJS_102739750</scope>
  <parameter name="JobStreamWorkstationName">
    <value>NC125102</value>
  </parameter>
  <parameter name="JobStreamName">
    <value>RJS_102739750</value>
  </parameter>
</action>
</eventRule>
</eventRuleSet>

```

La règle dans l'exemple suivant soumet le travail RJR_30411 sur le poste de travail NC122160 dès que le flot de travaux RJS_102739750 du poste de travail NC125102 est soumis.

```

<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="TWS_PLAN_EVENTS_JOB_STREAM_SUBMITTED" ruleType="filter" isDraft="no">
    <description>Event: Job Stream Submitted; Action: Submit job</description>
    <eventCondition name="jsSubEvt1"
      eventProvider="TWSObjectsMonitor"
      eventType="JobStreamSubmit">
      <scope>WORKSTATIONVALUE # JOBSTREAMVALUE</scope>
      <filteringPredicate>
        <attributeFilter name="JobStreamWorkstation" operator="eq">
          <value>NC125102</value>
        </attributeFilter>
        <attributeFilter name="JobStreamName" operator="eq">
          <value>RJS_102739750</value>
        </attributeFilter>
        <attributeFilter name="Priority" operator="range">
          <value>15</value>
          <value>30</value>
        </attributeFilter>
        <attributeFilter name="LatestStart" operator="le">
          <value>2011-04-26</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="TWSAction" actionType="sbj" responseType="onDetection">

```

```

        <description>Launch an existing TWS job stream</description>
        <scope>SBJ NC122160#RJR_30411 INTO NC122160#JOBS</scope>
        <parameter name="JobUseUniqueAlias">
            <value>true</value>
        </parameter>
        <parameter name="JobDefinitionName">
            <value>RJR_30411</value>
        </parameter>
        <parameter name="JobDefinitionWorkstationName">
            <value>NC122160</value>
        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

Événements FileMonitor

Les événements FileMonitor sont :

- FileCreated
- FileDeleted
- ModificationCompleted
- LogMessageWritten

Lorsque vous interceptez des fichiers à l'aide des événements FileCreated, FileDeleted et LogMessageWritten, la mémoire consommée par les processus ssmagent.bin et ssmagent.exe augmente linéairement en fonction du nombre de fichiers interceptés et du nombre d'événements créés. Par conséquent, gardez à l'esprit que plus vous ferez appel aux caractères génériques dans ces types d'événement, et donc, plus vous intercepterez de fichiers, plus la mémoire consommée par les processus ssmagent.bin et ssmagent.exe sera importante.

Les événements FileMonitor ne sont pas pris en charge sur :

- Les pools, pools dynamiques et postes de travail de moteur distant.
- IBM i.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de chaque type d'événement.

Remarque : Utilisateurs PDF, les tables de paramètre ci-dessus constituent un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Utiliser la propriété MatchExpression de la règle d'événement LogMessageWritten

Le plug-in événement LogMessageWritten utilise l'expression régulière spécifiée dans la propriété MatchExpression afin d'effectuer les correspondances de sous-chaîne dans les entrées du fichier journal sous surveillance. La valeur de MatchExpression doit être une expression régulière valide conforme aux règles de syntaxe des expressions régulières de l'agent Netcool/SSM utilisés par le plug-in événement.

Les tables suivantes décrivent la syntaxe des marqueurs d'expressions régulières pris en charge par Netcool/SSM. Remarquez que pour écrire une expression régulière valide pour la propriété MatchExpression, vous devez écrire le caractère d'échappement \ (barre oblique inversée) avant chaque marqueur utilisé dans la

syntaxe d'expression régulière (par exemple, `\^` ou `\$`). Lorsque le marqueur comporte déjà une barre oblique inversée, vous devez en écrire une seconde (par exemple, `\\<` ou `\\b`).

Tableau 147. Syntaxe d'expression régulière.

Marqueur	Correspondances
.	Tout caractère.
^	Le début de ligne (une chaîne nulle).
\$	La fin d'une ligne ; une nouvelle ligne ou la fin du tampon de recherche.
\<	Le début d'un mot (où un mot est une chaîne de caractères alphanumériques).
\>	La fin d'un mot (la chaîne nulle entre un caractère alphanumérique et un caractère non alphanumérique).
\b	Toute frontière de mot (équivalent à <code>(\< \>)</code>).
\d	Un caractère numérique.
\D	Tout caractère non numérique.
\w	Un caractère de mot (alphanumérique ou trait de soulignement).
\W	Tout caractère qui n'est pas un caractère de mot (alphanumérique ou trait de soulignement).
\s	Un caractère blanc.
\S	Tout caractère non blanc.
\c	Caractères spéciaux et échappement. Les caractères suivants sont interprétés selon les conventions du langage C : <code>\0</code> , <code>\a</code> , <code>\f</code> , <code>\n</code> , <code>\r</code> , <code>\t</code> , <code>\v</code> . Pour indiquer un caractère en hexadécimal, employez la syntaxe <code>\xNN</code> syntax. Par exemple, <code>\x41</code> est le caractère ASCII A.
\	A l'exception des caractères décrits au-dessus, tous les caractères peuvent être fermés à l'aide de la barre oblique inversée, qui compose le préfixe d'échappement. Par exemple, pour indiquer un simple crochet gauche, utilisez <code>\[</code> .

Tableau 147. Syntaxe d'expression régulière. (suite)

Marqueur	Correspondances
[]	Tout caractère spécifié dans un jeu. Un jeu de caractères explicite peut être spécifié, comme dans [aeiou], tout comme des intervalles de caractères, comme [0-9A-Fa-f], qui correspondent à n'importe quel caractère numérique hexadécimal. Le trait d'union (-) perd sa signification particulière lorsqu'il est accompagné du caractère d'échappement, comme dans [A\Z] ou lorsqu'il est le premier ou le dernier caractère d'un jeu, comme par exemple dans [-xyz0-9]. Toutes les règles d'échappement à l'aide de la barre oblique inversée peuvent être utilisées à l'intérieur des []. Par exemple, l'expression [\x41-\x45] équivaut à l'ASCII [A-D]. Pour utiliser un crochet de fermeture dans un jeu, vous pouvez soit le fermer à l'aide de [\] ou l'utiliser comme premier caractère d'un jeu, comme dans [xyz]. Les classes de caractères POSIX-style aussi autorisées dans un jeu de caractères. La syntaxe pour les classes de caractères est [:class:]. Les classes de caractères pris en charge sont : • [:alnum:] - caractères alphanumériques. • [:alpha:] - caractères alphabétiques. • [:blank:] - caractères de tabulation et espace. • [:cntrl:] - caractères de commande. • [:digit:] - caractères numériques. • [:graph:] - caractères imprimables et visibles. • [:lower:] - caractères alphabétiques minuscules. • [:print:] - caractères imprimables (qui ne sont pas des caractères de commande). • [:punct:] - caractères de ponctuation (qui ne sont ni des lettres, ni des chiffres, ni des caractères de commande, ni des espaces). • [:space:] - caractères espaces (tel que l'espace, la tabulation et le saut de page). • [:upper:] - caractères alphabétiques majuscules. • [:xdigit:] - caractères numériques hexadécimaux. Les crochets sont autorisés à l'intérieur des crochets de jeu. Par exemple, [a-z0-9!] équivaut à [[:lower:][:digit:]] dans l'environnement local C.
[^]	Inverse le comportement d'un jeu de caractères [] comme décrit au-dessus. Par exemple, [^[:alpha:]] correspond à tout caractère qui n'est pas alphabétique. Le symbole caret ^ n'a cette signification particulière que lorsqu'il occupe la première place d'un jeu de caractères entre crochets.
{n}	Exactement n occurrences de la précédente expression, où 0 <= n <= 255. Par exemple, a{3}à aaa.
{n,m}	Entre n et m occurrences de la précédente expression, où 0 <= n <= m <= 255. Par exemple, un nombre hexadécimal de 32-bit peut être décrit comme 0x[[:xdigit:]]{1,8}.
{n,}	Au moins n occurrences ou plus (jusqu'à une infinité) de la précédente expression.
*	Zéro occurrence ou plus de la précédente expression.
+	Une occurrence ou plus de la précédente expression.
?	Zéro ou une occurrence de la précédente expression.

Tableau 147. Syntaxe d'expression régulière. (suite)

Marqueur	Correspondances
(exp)	Groupement ; toute série d'expression peut être regroupée entre parenthèses afin d'appliquer un opérateur postfix ou barre pour un groupe d'expressions successives. Par exemple : - à ab+ correspondent tous les abbb - à (ab)+ correspondent tous les ababab
	Les autres expressions (logique OR). La barre verticale () possède la priorité la plus basse de tous les marqueurs du langage des expressions régulières. Cela signifie que, à ab cd, correspondent tous les cd mais pas les abd (dans ce cas, utilisez a(b c)d).

Conseil : Lorsque vous définissez des expressions régulières pour la correspondance de caractères multi-octets, entourez chaque caractère multi-octets de parenthèses ().

Le tableau 148 fournit un ensemble d'exemples d'expressions régulières, tout comme des exemples de chaînes, ainsi que les résultats de l'application des expressions régulières à ces chaînes.

Il y a deux cas importants dans la correspondance d'expressions régulières et de chaînes. Une expression régulière peut correspondre à une chaîne entière (cas connu comme un *correspondance de chaîne*) ou seulement à une partie de cette chaîne (cas connu comme un *correspondance de sous-chaîne*). Par exemple, l'expression régulière \<int\> va générer une correspondance de sous-chaîne pour la chaîne int x mais ne générera pas une correspondance de chaîne. Cette distinction est importante parce que certains sous-agents ne prennent pas en charge la correspondance de sous-chaîne. Lorsqu'ils sont applicables, les résultats listés dans les exemples différencient les correspondances de chaîne et de sous-chaîne.

Tableau 148. Exemples d'expressions régulières.

Cette expression...	Appliquée à cette chaîne...	Résulte en...
.	a	Correspondance de chaîne
	!	Correspondance de chaîne
	abcdef	Correspondance de sous-chaîne pour a
	chaîne vide	Pas de correspondance
M..COUNT	MINCOUNT	Correspondance de chaîne
	MXXCOUNTY	Correspondance de sous-chaîne pour MXXCOUNT
	NONCOUNT	Pas de correspondance
.*	chaîne vide	Correspondance de chaîne
	Animal	Correspondance de chaîne
.+	Toute chaîne non vide	Correspondance de chaîne
	chaîne vide	Pas de correspondance
^	chaîne vide	Correspondance de chaîne
	hello	Correspondance de sous-chaîne de longueur 0 en position 0 (position 0 = premier caractère d'une chaîne)

Tableau 148. Exemples d'expressions régulières. (suite)

Cette expression...	Appliquée à cette chaîne...	Résulte en...
\$	chaîne vide	Correspondance de chaîne
	hello	Correspondance de sous-chaîne de longueur 0 en position 5 (position 0 = premier caractère d'une chaîne)
^\$	chaîne vide	Correspondance de chaîne
	hello	Pas de correspondance
\bee	tee	Pas de correspondance
	Paid fee	Pas de correspondance
	feel	Pas de correspondance
	eel	Correspondance de sous-chaîne pour ee
.*thing.*	The thing is in here	Correspondance de chaîne
	there is a thing	Correspondance de chaîne
	it isn't here	Pas de correspondance
	thinxxx	Pas de correspondance
a*	chaîne vide	Correspondance de chaîne
	aaaaaaaa	Correspondance de chaîne
	a	Correspondance de chaîne
	aardvark	Correspondance de sous-chaîne pour aa
	this string	Correspondance de sous-chaîne
((ab)*c)*	chaîne vide	Correspondance de chaîne
	cccccccc	Correspondance de chaîne
	ccccabccabc	Correspondance de chaîne
a+	chaîne vide	Pas de correspondance
	aaaaaaaa	Correspondance de chaîne
	a	Correspondance de chaîne
	aardvark	Correspondance de sous-chaîne pour aa
	this string	Pas de correspondance
(ab)+c)*	chaîne vide	Correspondance de chaîne
	ababababcabc	Correspondance de chaîne
(ab){2}	abab	Correspondance de chaîne
	cdabababab	Correspondance de sous-chaîne pour abab
[0-9]{4,}	123	Pas de correspondance
	a1234	Correspondance de sous-chaîne pour 1234
a{0}	chaîne vide	Correspondance de chaîne
	a	Pas de correspondance
	hello	Correspondance de sous-chaîne de longueur 0 en position 0 (position 0 = premier caractère d'une chaîne)
[0-9]{1,8}	this is not a number	Pas de correspondance
	a=4238, b=4392876	Correspondance de sous-chaîne pour 4238

Tableau 148. Exemples d'expressions régulières. (suite)

Cette expression...	Appliquée à cette chaîne...	Résulte en...
([aeiou][^aeiou])+	Hello	Correspondance de sous-chaîne pour e l
	!!! Supacalafraglistic	Correspondance de sous-chaîne pour upacalaf
[+-]?1	1	Correspondance de chaîne
	+1	Correspondance de chaîne
	-1	Correspondance de chaîne
	.1	Correspondance de sous-chaîne pour 1
	value+1	Correspondance de sous-chaîne pour +1
a b	a	Correspondance de chaîne
	b	Correspondance de chaîne
	c	Pas de correspondance
	Daniel	Correspondance de sous-chaîne pour a
abcd efgh	abcd	Correspondance de chaîne
	efgh	Correspondance de chaîne
	abcdfgh	Correspondance de sous-chaîne pour abcd
[0-9A-F]+	BAADF00D	Correspondance de chaîne
	C	Correspondance de chaîne
	baadF00D	Correspondance de sous-chaîne pour F00D
	c	Pas de correspondance
	G	Pas de correspondance
	g	Pas de correspondance
x = \d+	x = 1234	Correspondance de chaîne
	x = 0	Correspondance de chaîne
	x = 1234a	Correspondance de sous-chaîne pour x = 1234
	x = y	Pas de correspondance
	x^=^ où ^ représente un caractère espace	Pas de correspondance
\D\d	a1	Correspondance de chaîne
	a11	Correspondance de sous-chaîne pour a1
	-9	Correspondance de chaîne
	a	Pas de correspondance
	8	Pas de correspondance
	aa	Pas de correspondance
	4t	Pas de correspondance

Tableau 148. Exemples d'expressions régulières. (suite)

Cette expression...	Appliquée à cette chaîne...	Résulte en...
\s+	Hello_w0rld	Pas de correspondance
	Hello^world où ^ représente un caractère espace	Correspondance de sous-chaîne pour ^ où ^ représente un caractère espace
	Widget^ où ^ représente un caractère espace	Correspondance de sous-chaîne ^ où ^ représente un caractère espace
	^ où ^ représente un caractère espace	Correspondance de chaîne
\S+	Hello_w0rld	Correspondance de sous-chaîne de longueur 11 pour Hello_w0rld
	Hello^world où ^ représente un caractère espace	Correspondance de sous-chaîne pour Hello
	Widget^ où ^ représente un caractère espace	Correspondance de sous-chaîne pour Widget
	^ où ^ représente un caractère espace	Pas de correspondance
\w+	D4n_v4n Vugt	Correspondance de sous-chaîne pour D4n_v4n
	^hello où ^ représente un caractère espace	Correspondance de sous-chaîne pour hello
	blah	Correspondance de chaîne
	x#1	Pas de correspondance
	foo bar	Pas de correspondance
\W	Hello there	Correspondance de sous-chaîne de longueur 1 pour les caractères espaces de séparation
	~	Correspondance de chaîne
	aa	Pas de correspondance
	a	Pas de correspondance
	-	Pas de correspondance
	^444 == 5 où ^ représente un caractère espace	Correspondance de sous-chaîne de longueur 1 pour le premier ^ où ^ représente un caractère espace
\w+\s*=\s*\d+	x = 123	Correspondance de chaîne
	count0=555	Correspondance de chaîne
	my_var=66	Correspondance de chaîne
	0101010=0	Correspondance de chaîne
	xyz = e	Pas de correspondance
	delta=	Pas de correspondance
	==8	Pas de correspondance
[:alnum:]]+	1234	Correspondance de chaîne
	...D4N13L	Correspondance de sous-chaîne pour D4N13L

Tableau 148. Exemples d'expressions régulières. (suite)

Cette expression...	Appliquée à cette chaîne...	Résulte en...
[:alpha:]]+	Bubble	Correspondance de chaîne
	...DANI3L	Correspondance de sous-chaîne pour DANI
	69	Pas de correspondance
[:blank:]]+	alpha^^^^and beta où ^ représente un caractère espace	Correspondance de sous-chaîne pour ^^^^ où ^ représente un caractère espace
	Animal	Pas de correspondance
	chaîne vide	Pas de correspondance
[:space:]]+	alpha^^^^and beta où ^ représente un caractère espace	Correspondance de sous-chaîne pour ^^^^ où ^ représente un caractère espace
	Animal	Pas de correspondance
	chaîne vide	Pas de correspondance
[:cntrl:]]+	...Hello W0rld!	Pas de correspondance
	chaîne vide	Pas de correspondance
[:graph:]]+	hello world	Correspondance de sous-chaîne pour hello
	^^^^ où ^ représente un caractère espace	Pas de correspondance
	^^!^ où ^ représente un caractère espace	Correspondance de sous-chaîne pour !?
[:lower:]]+	Animal	Correspondance de sous-chaîne pour nimal
	ABC	Pas de correspondance
	0123	Pas de correspondance
	foobar	Correspondance de chaîne
	^^0blaH! où ^ représente un caractère espace	Correspondance de sous-chaîne pour bla
[_[:lower:]]+	foo_bar	Correspondance de chaîne
	this_thinG!!!	Correspondance de sous-chaîne pour _thin
[:upper:]]+	OUI	Correspondance de chaîne
	#define MAX 100	Correspondance de sous-chaîne pour MAX
	f00 b4r	Pas de correspondance
[:print:]]+	hello world	Correspondance de chaîne
	^^^^ où ^ représente un caractère espace	Correspondance de chaîne
[:punct:]]+	didn't	Correspondance de sous-chaîne pour '
	Animal	Pas de correspondance
[:xdigit:]]+	43298742432392187ffe	Correspondance de chaîne
	x = bAAdF00d	Correspondance de sous-chaîne pour bAAdF00d
	4327afeffegokpoj	Correspondance de sous-chaîne pour 4327afeffe
c:\\temp	c:\\temp	Correspondance de chaîne

Exemple

La règle dans l'exemple suivant envoie un courrier électronique à une liste de destinataires dès que le fichier /home/book.txt est créé sur le poste de travail editor_wrkstn.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="FILE_MONITOR_FILE_CREATED" ruleType="filter" isDraft="no">
    <description>Event: File Created; Action: Send mail</description>
    <validity to="2012-04-22" />
    <eventCondition name="fileCrEvt1" eventProvider="FileMonitor" eventType="FileCreated">
      <scope>/HOME/BOOK.TXT ON EDITOR_WRKSTN</scope>
      <filteringPredicate>
        <attributeFilter name="FileName" operator="eq">
          <value>/home/book.txt</value>
        </attributeFilter>
        <attributeFilter name="SampleInterval" operator="eq">
          <value>60</value>
        </attributeFilter>
        <attributeFilter name="Workstation" operator="eq">
          <value>editor_wrkstn</value>
        </attributeFilter>
        <attributeFilter name="Hostname" operator="eq">
          <value>ceditor</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="MailSender" actionType="SendMail" responseType="onDetection">
      <description>Send an eMail</description>
      <scope>SAUL.FELLOW@US.IBM.COM, ISAAC.LINGER@US.IBM.COM : THE EXPECTED FILE
        HAS BEEN CREATED!</scope>
      <parameter name="Cc">
        <value>william.waulkner@us.ibm.com</value>
      </parameter>
      <parameter name="Bcc">
        <value>ernest.demingway@us.ibm.com</value>
      </parameter>
      <parameter name="Body">
        <value>The expected file was created!
          The book is ready to be published.</value>
      </parameter>
      <parameter name="To">
        <value>saul.fellow@us.ibm.com, isaac.linger@us.ibm.com</value>
      </parameter>
      <parameter name="Subject">
        <value>The expected file was created!</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>
```

Événements TWSApplicationMonitor

Les événements TWSApplicationMonitor concernent les processus IBM Workload Scheduler, le système de fichiers et la boîte de messages. Il s'agit de :

- MessageQueuesFilling
- TivoliWorkloadSchedulerFileSystemFilling
- TivoliWorkloadSchedulerProcessNotRunning

Les événements TWSApplicationMonitor ne sont pas pris en charge sur les systèmes IBM i.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de chaque type d'événement.

Remarque : Utilisateurs PDF, les tables de paramètre ci-dessus constituent un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec

le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Exemple

La règle dans l'exemple suivant consigne des messages d'avertissement LOGMSG01W dès que les files d'attente des messages de intercom ou de mailbox sur le poste de travail NC122160 atteignent 70 pour cent de leur taille.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="TWS_APPL_MONITOR_MESSAGE_QUEUES_FILLING" ruleType="filter" isDraft="no">
    <description>Event: Message queues filling; Action: Message logger</description>
    <timeZone>America/Los_Angeles</timeZone>
    <validity from="2011-04-25"/>
    <activeTime end="17:00:00"/>
    <eventCondition name="twsmesQueEvt1" eventProvider="TWSApplicationMonitor"
      eventType="TWSMessageQueues">
      <scope>INTERCOM, MAILBOX FILLED UP 70% ON NC122160</scope>
      <filteringPredicate>
        <attributeFilter name="MailboxName" operator="eq">
          <value>intercom</value>
          <value>mailbox</value>
        </attributeFilter>
        <attributeFilter name="FillingPercentage" operator="ge">
          <value>70</value>
        </attributeFilter>
        <attributeFilter name="Workstation" operator="eq">
          <value>NC122160</value>
        </attributeFilter>
        <attributeFilter name="SampleInterval" operator="eq">
          <value>60</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="MessageLogger" actionType="MSGLOG" responseType="onDetection">
      <description>Write a warning message log</description>
      <scope>OBJECT=LOGMSG01W MESSAGE=MAILBOX AND/OR INTERCOM QUEUE
        HAS REACHED 70% OF FILLING</scope>
      <parameter name="ObjectKey">
        <value>LOGMSG01W</value>
      </parameter>
      <parameter name="Message">
        <value>Mailbox and/or Intercom queue has reached 70% of filling</value>
      <parameter name="Severity">
        <value>Warning</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>
```

Fournisseurs d'actions et définitions

Cette section détaille les types d'action des fournisseurs d'actions suivants :

- GenericAction
- MailSender
- MessageLogger
- SmartCloud Control Desk
- TBSMEventForwarder
- TECEventForwarder
- TWSAction
- «TWSForZosAction», à la page 889

Actions GenericAction

Ce fournisseur met en oeuvre une action unique nommée RunCommand qui exécute des commandes non IBM Workload Scheduler. Les commandes sont exécutées sur le même ordinateur que celui où s'exécute le processeur d'événements.

Seul l'utilisateur *TWS* est autorisé à exécuter la commande.

Important : Si la commande inclut la redirection de sortie (par l'utilisation d'un ou de deux signes >), insérez-la dans le fichier exécutable, puis définissez le nom de fichier comme l'argument de la propriété Command.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de RunCommand.

Remarque : Utilisateurs PDF, les tables de paramètre ci-dessus constituent un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Exemple

La règle dans l'exemple suivant exécute la commande **ps -ef** pour lister tous les processus en cours d'exécution sur un poste de travail UNIX lorsqu'un paramètre invalide est découvert sur ce poste de travail. Remarquez que cette règle repose sur un événement personnalisé développé à l'aide du fournisseur d'événements GenericEventPlugIn. Pour de plus amples informations sur les types le développement de types d'événement personnalisable, voir «Définition d'événements personnalisés», à la page 161.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="CUSTOM_EVENT_GENERIC_EVENT" ruleType="filter" isDraft="yes">
    <description>Event: Generic Event; Action: Run Command</description>
    <activeTime start="08:30:00" end="17:30:00"/>
    <eventCondition name="genericEvt3" eventProvider="GenericEventPlugIn"
      eventType="Event1">
      <scope>INVALID PARAMETER ON WORKSTATIONVALUE</scope>
      <filteringPredicate>
        <attributeFilter name="Param1" operator="ne">
          <value>Invalid Parameter</value>
        </attributeFilter>
        <attributeFilter name="Workstation" operator="eq">
          <value>WorkstationValue</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="GenericActionPlugIn" actionType="RunCommand"
      responseType="onDetection">
      <description>Run a command</description>
      <scope>PS -EF</scope>
      <parameter name="Command">
        <value>ps -ef</value>
      </parameter>
      <parameter name="WorkingDir">
        <value>/home</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>
```

Actions MailSender

Ce fournisseur met en oeuvre une action unique nommée SendMail qui se connecte à un serveur SMTP pour envoyer un courrier électronique. Utilisez optman pour personnaliser les attributs associés suivants (pour plus d'informations sur optman, voir *Guide d'administration*) :

- Mail sender
- SMTP server
- SMTP port number
- Mail user name
- Mail user password
- SSL

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de SendMail.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Actions MessageLogger

Ce fournisseur met en oeuvre une action unique nommée MSGLOG qui journalise l'occurrence d'une situation dans une base de données d'audit interne. Le nombre d'entrées de la base de données d'audit est configurable. Il y a un nettoyage automatique basé sur une règle FIFO.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de MSGLOG.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Actions SmartCloud Control Desk

Ce fournisseur met en oeuvre une action unique, nommée OpenTicket, qui ouvre un ticket sur un SmartCloud Control Desk par défaut. Utilisez optman pour spécifier le serveur SmartCloud Control Desk en définissant les options globales sccdUrl, sccdName et sccdUserPassword. Pour plus d'informations sur optman, voir le document *Guide d'administration*.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de OpenTicket.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Actions ServiceNow

Ce fournisseur met en oeuvre une action unique, nommée Open Incident, qui ouvre un incident dans ServiceNow lorsqu'un travail correspondant à une règle définie se termine avec une erreur. Utilisez optman pour spécifier le serveur ServiceNow en définissant les options globales servicenowUrl, servicenowUserName et servicenowUserPassword.

Vous trouverez une rubrique relative à la description détaillée de la commande `optman` et de ces options globales dans le document *Guide d'administration*.

Cliquez ici pour afficher les zones Dynamic Workload Console pour Open Incident de ServiceNow.

Remarque : Utilisateurs PDF, la table de paramètre ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF d'IBM Workload Scheduler Knowledge Center. Vous devez commencer par l'afficher dans le Knowledge Center du produit avant de le sauvegarder ou de l'imprimer.

Actions TBSMEventForwarder

Ce fournisseur met en oeuvre une action unique, nommée TBSMFWD, qui transmet l'événement à un serveur Tivoli Business Service Manager externe ou à toute autre application capable d'écouter les événements au format TBSM (par exemple, Netcool/OMNIBus). Le fournisseur utilise un serveur Sonde EIF par défaut dont le nom d'hôte et le port peuvent être définis dans les options globales `TECServerName` et `TECServerPort` à l'aide de l'utilitaire `optman`.

Remarque : Les options `TECServerName` et `TECServerPort` sont toutes les deux utilisées pour les applications traitant les événements au format TEC ou TBSM. Pour plus d'informations sur `optman`, voir le document *Guide d'administration*.

La sonde IEF utilisée en tant que destinataire peut être remplacée par des paramètres d'action.

Cliquez ici pour afficher les zones Dynamic Workload Console de TBSMFWD.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Configuration de Tivoli Business Services Manager pour la réception d'événements

Pour configurer Tivoli Business Service Manager en vue de la réception d'événements de IBM Workload Scheduler, sur la Sonde EIF, vous devez copier le fichier `utilities/tivoli_eif_tws.rules` fourni avec le DVD de IBM Workload Scheduler. Ensuite, vous devez éditer le fichier `tivoli_eif.rules` en ajoutant la ligne suivante :

```
include "tivoli_eif_tws.rules"
```

Actions TECEventForwarder

Ce fournisseur met en oeuvre une action unique, nommée TECFWD, qui transmet l'événement à un serveur TEC (Tivoli Enterprise Console) externe ou à toute autre application capable d'écouter les événements au format TEC. Le fournisseur utilise un serveur Sonde EIF par défaut dont le nom d'hôte et le port peuvent être définis dans les options globales `TECServerName` et `TECServerPort` à l'aide de l'utilitaire `optman`. Pour plus d'informations sur `optman`, voir le document *Guide d'administration*.

Le TEC utilisé en tant que destinataire peut être remplacé par des paramètres d'action.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de TECFWD.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Actions TWSAction

Les actions TWSAction sont :

- SubmitJobStream
- SubmitJob
- SubmitAdHocJob
- ReplyPrompt

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de chaque type d'action.

Remarque : Utilisateurs PDF, les tables de paramètre ci-dessus constituent un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Utilisation de la propriété SchedTimeResolutionCriteria de l'action SubmitJob

Utilisez cette propriété pour faire correspondre le travail en question avec une instance spécifique du flot de travaux qui le contient (défini avec la propriété JobStreamName) en fonction de l'heure planifiée du flot de travaux. Les valeurs que vous pouvez définir sont les suivantes :

Précédent

Le travail est soumis à l'instance de flot de travaux précédente la plus proche dans le plan.

Suivant

Le travail est soumis à l'instance de flot de travaux suivante la plus proche dans le plan.

Tout Le travail est soumis à l'instance de flot de travaux précédente ou suivante la plus proche dans le plan.

TWSForZosAction

Ce fournisseur implémente une action unique nommée AddJobStream qui ajoute une occurrence d'application (flot de travaux) au plan actuel sur IBM Workload Scheduler for z/OS. Ce fournisseur est à utiliser dans des configurations de planification de bout en bout IBM Workload Scheduler.

La description d'application de l'occurrence à ajouter doit exister dans la base de données AD de IBM Workload Scheduler for z/OS.

Cliquez [ici](#) pour afficher les zones Dynamic Workload Console de AddJobStream.

Remarque : Utilisateurs PDF, la table de paramètres ci-dessus est un fichier html référencé dans le fichier PDF. Il n'est pas sauvegardé localement avec le fichier PDF

dans le centre de documentation. Vous devez dans un premier temps le consulter dans le centre de documentation avant de le sauvegarder ou de l'imprimer.

Exemple

Dans cet exemple, une entreprise pharmaceutique utilise la règle ZOSRULE031 pour produire un planning de distribution des marchandises sous le contrôle de département DISTR07. Dès que la liste des marchandises commandées qui sont à livrer dans le mois à venir est prête et placée dans le fichier MONTHLYORDERS.TXT sur l'agent RU192298 d'une filiale, le système centralisé ajoute l'application (flot de travaux) ADFIRST au plan actuel. ADFIRST contient les opérations (travaux) qui produisent un calendrier de livraison optimisé pour le mois prochain.

```
<?xml version="1.0"?>
<eventRuleSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.ibm.com/xmlns/prod/tws/1.0/event-management/rules"
  xsi:schemaLocation="http://www.ibm.com/xmlns/prod/tws/1.0/
    event-management/rules/EventRules.xsd">
  <eventRule name="ZOSRULE031" ruleType="filter" isDraft="no">
    <eventCondition name="fileCrtEvt19" eventProvider="FileMonitor"
      eventType="FileCreated">
      <scope>/PRODORDER/MONTHLYORDERS.TXT ON RU192298</scope>
      <filteringPredicate>
        <attributeFilter name="Param1" operator="ne">
          <value>/prodorder/monthlyorders.txt</value>
        </attributeFilter>
        <attributeFilter name="SampleInterval" operator="eq">
          <value>60</value>
        </attributeFilter>
        <attributeFilter name="Workstation" operator="eq">
          <value>RU192298</value>
        </attributeFilter>
      </filteringPredicate>
    </eventCondition>
    <action actionProvider="TWSForZosAction" actionType="AddJobStream"
      responseType="onDetection">
      <scope>
        ADD JOBSTREAM ADFIRST[DEADLINE OFFSET: 0001] WITH OWNER DISTR07 IN PLAN
      </scope>
      <parameter name="HoldAll">
        <value>false</value>
      </parameter>
      <parameter name="Priority">
        <value>5</value>
      </parameter>
      <parameter name="JobStreamDeadlineOffset">
        <value>0001</value>
      </parameter>
      <parameter name="JobStreamName">
        <value>ADFIRST</value>
      </parameter>
      <parameter name="OwnerDescription">
        <value>Owner description</value>
      </parameter>
      <parameter name="Owner">
        <value>distr07</value>
      </parameter>
      <parameter name="DependenciesResolution">
        <value>All</value>
      </parameter>
      <parameter name="AuthorityGroup">
        <value>AuthGrpBase</value>
      </parameter>
      <parameter name="Parm_1">
        <value>var1=value1</value>
      </parameter>
      <parameter name="Parm_2">
        <value>var2=value2</value>
      </parameter>
    </action>
  </eventRule>
</eventRuleSet>
```



```

        <parameter name="JCLVariableTable">
            <value>VarTableZos01</value>
        </parameter>
        <parameter name="JobStreamDescription">
            <value>Ce flot de travaux contient des travaux qui traitent
                        des commandes pour le propriétaire DISTR07.</value>
        </parameter>
        <parameter name="Group">
            <value>GroupBase</value>
        </parameter>
    </action>
</eventRule>
</eventRuleSet>

```

Annexe B. Référence de schéma Job Submission Description Language

Cette section de référence spécifie la sémantique et la structure du langage Job Submission Description Language (JSDL) qui est destiné à être utilisé spécifiquement avec Dynamic Workload Broker. Le schéma JSDL permet de décrire les exigences du travail pour la soumission sur les ressources. Dynamic Workload Broker analyse l'environnement informatique et affecte les meilleures ressources disponibles pour exécuter le travail, en fonction des exigences que vous spécifiez.

Introduction

Job Submission Description Language (JSDL) est un langage permettant de décrire les exigences du travail pour la soumission aux ressources. Il contient un vocabulaire et un schéma XML normatif qui facilite l'expression de ces exigences sous forme d'un ensemble d'éléments XML.

Les fichiers JSDL adhèrent à la syntaxe et à la sémantique XML telles qu'elles sont définies dans le schéma JSDL.

Structure de document Job Submission Description Language

Un fichier JSDL est décrit à l'aide de la syntaxe XML et adhère à la syntaxe et à la sémantique XML. La syntaxe XML est une norme informatique et n'est pas expliquée dans ce manuel. Le fichier JSDL adhère également à des règles de syntaxe JSDL, telles qu'elles sont expliquées dans «Types d'éléments Job Submission Description Language», à la page 896 et dans «Eléments JSDL», à la page 899.

Le fichier JSDL est constitué d'éléments (simples ou complexes) et de types. Les éléments complexes contiennent d'autres éléments tandis que les éléments simples n'en contiennent pas. Une spécification de type réalise un contrôle de syntaxe sur la valeur spécifiée pour l'élément auquel elle se réfère. Par exemple, l'élément **physicalMemory** adhère au type `jsdl:NumericRangeType`. Le type `jsdl:NumericRangeType` spécifie que vous pouvez affecter à cet élément soit une valeur numérique spécifique, soit une valeur de plage numérique. Aucun autre type de valeur n'est pris en charge pour l'élément **physicalMemory**.

Le fichier JSDL est organisé en une structure hiérarchique où l'élément **jobDefinition** est l'élément racine. L'élément **jobDefinition** contient tous les éléments qui décrivent le travail et leurs attributs.

La définition du pseudo-schema est du type suivant :

```
< jobDefinition >
  <annotation ... />?
  <category>.../>*
  <variables ... />?
  <application ... />
  <resources ... />?
  <relatedResources .../>*
  <optimization ...>?
  <scheduling ...>?
</jobDefinition>
```

Le tableau 149 fournit une vue en tableau du fichier JSDL en indiquant les relations hiérarchiques entre les éléments contenus dans l'élément **jobDefinition**.

Tableau 149. Structure hiérarchique du fichier JSDL

Premier niveau	Deuxième niveau	Troisième niveau	Quatrième niveau
annotation			
category			
variables	stringVariable		
	uintVariable		
	doubleVariable		
application	script		
	arguments	value	
	environment	nom de la variable	
	credential	username	
		groupname	
		mot de passe	
	j2ee	invoker	type
		jms	connFactory
			destination
			message
		ejb	jndiHome
		credential	userName
			mot de passe
			JAASalias

Tableau 149. Structure hiérarchique du fichier JSDL (suite)

Premier niveau	Deuxième niveau	Troisième niveau	Quatrième niveau
resources	candidateHosts	hostName	
	candidateCPUs	UC	speed
	physicalMemory		
	virtualMemory		
	candidateOperating Systems	operatingsystems	
	fileSystem		
	logicalResource		
	group		
	properties	and	and
			ou
			requirement
	ou		and
			ou
			requirement
	requirement		and
			ou
			requirement
	allocation		
	relationship		
	candidateResources (réservé pour une utilisation interne)	endpointReference (réservé pour une utilisation interne)	
relatedResources	logicalResource		
	group		
	properties	and	and
			ou
			requirement
	ou		and
			ou
			requirement
	requirement		and
			ou
			requirement
	allocation		
	relationship		
	candidateResources (réservé pour une utilisation interne)	endpointReference (réservé pour une utilisation interne)	
optimization	objective		
	ewlm		

Tableau 149. Structure hiérarchique du fichier JSDL (suite)

Premier niveau	Deuxième niveau	Troisième niveau	Quatrième niveau
scheduling	maximumResource		
	WaitingTime		
	estimatedDuration		
	priority		
	recoveryActions	action	parameters
			credential
			tpmaddress
			workflow

La syntaxe JSDL utilise les conventions de style BNF pour les éléments et les attributs :

- ? Indique que l'élément ou l'attribut est facultatif et peut être spécifié une fois.
- * Indique que l'élément ou l'attribut est facultatif et peut être spécifié zéro ou plusieurs fois.
- + Indique que l'élément ou l'attribut est obligatoire et peut être spécifié une ou plusieurs fois.
- [...] Indique que les éléments et attributs contenus dans les parenthèses forment un groupe.
- | Indique que deux ou plusieurs éléments ou attributs s'excluent mutuellement.

Types d'éléments Job Submission Description Language

La spécification JSDL utilise un certain nombre de types de schéma XML standard. Elle utilise également un certain nombre de types spécifiques à la description des exigences de travail.

Ces deux types réalisent une vérification de syntaxe sur la valeur qui peut être affectées à chaque élément dans le fichier JSDL. Par exemple, l'élément **physicalMemory** adhère au type jsdl:NumericRangeType. Le type jsdl:NumericRangeType spécifie que vous pouvez affecter à cet élément soit une valeur numérique spécifique, soit une valeur de plage numérique. Aucun autre type de valeur n'est pris en charge pour l'élément **physicalMemory**.

Types de schéma XML normatifs

La spécification JSDL adopte les types de schéma XML normatifs (xsd) répertoriés ci-dessous. La syntaxe XML est une norme informatique et n'est pas expliquée dans ce manuel.

- xsd:any
- xsd:anyURI
- xsd:boolean
- xsd:double
- xsd:DoubleVariableType
- xsd:duration

- xsd:IDREF
- xsd:NCName
- xsd:PriorityType
- xsd:QName
- xsd:string
- xsd:unsignedInt
- xsd:UnsignedIntVariableType

Types JSDL

Les types suivants sont spécifiques à la syntaxe JSDL :

StringVariableExpressionType

Un type d'expression de variable de chaîne est un type simple dans lequel vous pouvez indiquer une expression de variable qui peut contenir une ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne. Voici le schéma de syntaxe pour ce type :

```
<...>
<xsd:simpleType name="StringVariableExpressionType">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string' />
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string'>
        <xsd:pattern
          value=".*\t*\r*\n*((\\$\\{[a-zA-Z_]+
            [0-9a-zA-Z_\\.\\-]*\\})+[^\\{]*[\\.\\n]*)+" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>
</...>
```

DoubleVariableExpressionType

Un type d'expression de variable double est un type simple dans lequel vous pouvez spécifier une expression de variable qui peut contenir une référence de variable, telle que \${var} ou une valeur double. Voici le schéma de syntaxe pour ce type :

```
<...>
<xsd:simpleType name="DoubleVariableExpressionType">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base='xsd:double' />
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string'>
        <xsd:pattern value="[\n\r\t ]*($\\{[a-zA-Z_]+
          [0-9a-zA-Z_\\.\\-]*\\})[\n\r\t ]*" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>
</...>
```

UnsignedIntVariableExpressionType

Un type d'expression de variable non signé est un type simple dans lequel vous pouvez spécifier une expression de variable qui peut contenir une référence de variable, telle que \${var}, ou une valeur entière non signée. Voici le schéma de syntaxe pour ce type :

```

<...>
<xsd:simpleType name="UnsignedIntVariableExpressionType">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base='xsd:unsignedInt' />
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string'>
        <xsd:pattern value="\n\r\t ]*($\{[a-zA-Z_]+
          [0-9a-zA-Z_\.\\-]*\})[\n\r\t ]*" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>

</...>

```

NotEmptyStringVariableExpressionType

Un type d'expression de variable de chaîne est un type simple qui permet la spécification d'une expression de variable pouvant contenir une ou plusieurs références de variable telles que \${var}, et être facultativement associée à tout caractère ou à une chaîne simple. Cette expression de variable ne peut pas être vide. Voici le schéma de syntaxe pour ce type :

```

<xsd:simpleType name="NotEmptyStringVariableExpressionType">
  <xsd:union>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string'>
        <xsd:minLength value="1"/>
      </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType>
      <xsd:restriction base='xsd:string'>
        <xsd:pattern
          value=".*\t*\r*\n*((\$\{[a-zA-Z_]+
            [0-9a-zA-Z_\.\\-]*\})+[^\\{]*[.\n]*)+" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:union>
</xsd:simpleType>

```

NumericRangeOnlyType

Une valeur de plage numérique est un type complexe qui permet la définition des intervalles et plages supérieurs à, inférieurs à ou contenus dans la valeur spécifiée. Tous les nombres indiqués sont des expressions de variable double. Voici le schéma de syntaxe pour ce type :

```

<...>
  <minimum>jsd1:DoubleVariableExpressionType</minimum> ?
  <maximum>jsd1:DoubleVariableExpressionType</maximum> ?
</...>

```

NumericRangeType

Une valeur de plage numérique est un type complexe qui permet la définition des valeurs ou plages exactes. Tous les nombres indiqués sont des expressions de variable double. Voici le schéma de syntaxe pour ce type :

```

<...>
  <exact>jsd1:DoubleVariableExpressionType</exact> |
  <range>jsd1:NumericRangeOnlyType</range>
</...>

```

StringRangeOnlyType

Une valeur de plage de chaîne est un type complexe qui permet la définition des intervalles et plages supérieurs à, inférieurs à ou contenus

dans la valeur spécifiée. Tous les nombres et chaînes indiqués sont des expressions de variable de chaîne. Voici le schéma de syntaxe pour ce type :

```
<...>
  <minimum>jsd1:StringVariableExpressionType</minimum> ?
  <maximum>jsd1:StringVariableExpressionType</maximum> ?
</...>
```

StringRangeType

Une valeur de plage de chaîne est un type complexe qui permet la définition de valeurs exactes sous forme d'expressions de variable de chaîne ou de plages qui peuvent être appliquées aux types d'entier ou de chaîne. Voici le schéma de syntaxe pour ce type :

```
<...>
  <exact>jsd1:StringVariableExpressionType</exact> |
  <range>jsd1:StringRangeOnlyType</range>
</...>
```

Éléments JSDL

L'élément principal JSDL défini contient la sémantique des éléments qui sont définis par le fichier JSDL.

Le fichier JSDL est constitué d'éléments (simples ou complexes) et de types. Les éléments complexes contiennent d'autres éléments tandis que les éléments simples n'en contiennent pas. Une spécification de type réalise un contrôle de syntaxe sur la valeur spécifiée pour l'élément auquel elle se réfère.

Voici une liste des éléments contenus dans la syntaxe JSDL :

jobDefinition, élément

Définition

Cet élément décrit le travail et ses exigences. Il s'agit de l'élément racine du document JSDL. Cet attribut est obligatoire.

Type Le type de cet élément est jsdl:JobDefinitionType. Il peut contenir les éléments suivants :

- annotation
- category
- variables
- application
- resources
- relatedResources
- optimization
- scheduling

Attributs

name Nom du travail indiqué par l'utilisateur. Le type de cet attribut est **xsd:NCName**. Ce nom doit commencer par un caractère alphabétique et peut comporter les symboles suivants : soulignement (_), moins (-), et point (.). Les espaces, les caractères spéciaux et les caractères accentués et les nombres ne sont pas pris en charge. Cet attribut est obligatoire. Le nom que vous définissez pour cette zone identifie de façon unique la définition de travail lorsqu'elle est sauvegardée dans la base de données de référentiel

des travaux. Après avoir sauvegardé la définition de travail dans la base de données, vous pouvez la soumettre à l'aide de Dynamic Workload Console ou de la ligne de commande.

description

Chaîne spécifiant une courte description de la définition de travail. Le type de cet attribut est **xsd:string**. Cet attribut est facultatif.

targetNamespace

Identificateur URI spécifiant l'espace de nom cible de la définition de travail. Le type de cet attribut est **xsd:anyURI**. Cet attribut est obligatoire.

Pseudo-schéma

```
<jobDefinition
  name="xsd:NCName"
  description="xsd:string"?
  xsd:anyAttribute##other>
<annotation ... />?
<category>.../*
<variables ... />?
<application ... />
<resources ... />?
<relatedResources .../*
<optimization ...>?
<scheduling ...>?
<xsd:any##other/*
</jobDefinition>
```

élément annotation**Définition**

Cet élément fournit des informations descriptives et lisibles sur la définition des travaux. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est **xsd:string**.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<annotation
  xsd:anyAttribute##other>
xsd:string
<xsd:any##other/*
</application>
```

élément category**Définition**

Cette élément décrit la catégorie de travail qui vous aide à classer le travail par catégorie. Un travail peut avoir plusieurs catégories, par exemple : Education_DB, Financial_Dept, Asset_Management. Cette valeur peut être toute valeur de chaîne. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est **xsd:string**.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<category>
  xsd:string
</category>
```

élément variables

Définition

Cet élément décrit la liste des variables définies dans le fichier JSDL. Les trois types de variable suivants sont pris en charge :

- Chaîne
- Double
- Entier

La valeur de variable peut être référencée dans d'autres parties du document JSDL en spécifiant : \${variable name} Une variable référencée peut être l'une des variables définies dans le fichier JSDL avec l'élément **variable** ou elle peut être définie lors de la soumission du travail. La substitution peut être effectuée par le serveur Dynamic Workload Broker dans différentes phases du traitement du travail. Dans chaque phase, le serveur Dynamic Workload Broker tente de faire correspondre toutes les références de variable non encore substituées par les variables définies. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:VariablesType. Il peut contenir les éléments suivants :

- stringVariable
- uintVariable
- doubleVariable

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<variables
  xsd:anyAttribute##other>
  <stringVariable ...>*
  <uintVariable ...>*
  <doubleVariable ...>*
  <xsd:any##other/>*
</variables>?
```

élément stringVariable

Définition

Cet élément décrit une variable en spécifiant le nom de variable et la valeur de chaîne par défaut affectée. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est jsdl:StringVariableType.

Attributs

Les attributs suivants sont définis :

name Cet attribut spécifie le nom de la variable. Le type de cet attribut est xsd:NCName. Cet attribut est obligatoire.

description

Cet attribut spécifie la description de la variable. Le type de cet attribut est xsd:string. Cet attribut est facultatif.

Pseudo-schéma

```
<stringVariable
  name="xsd:NCName"
  description="xsd:string"?
  xsd:anyAttribute##other>
  xsd:string
<xsd:any##other/*>
</stringVariable>
```

élément doubleVariable

Définition

Cet élément décrit une variable en spécifiant le nom de la variable ainsi que la valeur double par défaut affectée. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est xsd:DoubleVariableType.

Attributs

Les attributs suivants sont définis :

name Cet attribut spécifie le nom de la variable. Le type de cet attribut est xsd:NCName. Cet attribut est obligatoire.

description

Cet attribut spécifie la description de la variable. Le type de cet attribut est xsd:string. Cet attribut est facultatif.

Pseudo-schéma

```
<doubleVariable
  name="xsd:NCName"
  description="xsd:string"?
  xsd:anyAttribute##other>
  xsd:double
<xsd:any##other/*>
</doubleVariable>
```

élément uintVariable

Définition

Cet élément décrit une variable en spécifiant le nom de la variable et la valeur d'entier non signé par défaut affectée. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est xsd:UnsignedIntVariableType.

Attributs

Les attributs suivants sont définis :

name Cet attribut spécifie le nom de la variable. Le type de cet attribut est xsd:NCName. Cet attribut est obligatoire.

description

Cet attribut spécifie le nom de la variable. Le type de cet attribut est xsd:string. Cet attribut est facultatif.

Pseudo-schéma

```
<uintVariable
  name="xsd:NCName"
  description="xsd:string"?
  xsd:anyAttribute##other>
  xsd:unsignedInt
<xsd:any##other/*>
</uintVariable>
```

élément application

Définition

Cet élément décrit l'application qui doit s'exécuter avec les paramètres associés. Cet élément est obligatoire et peut être spécifié une fois.

Type Le type de cet élément est jsdl:ApplicationType.

Attributs

Les attributs suivants sont définis :

name Indique le type de l'application. Les valeurs prises en charge sont les suivantes :

Executable

Un fichier utilisé pour effectuer différentes fonctions ou opérations sur un ordinateur.

J2EE Une application basée sur Java 2 Platform Enterprise Edition (J2EE).

Cet attribut est facultatif et peut être spécifié une fois.

description

Indique le nom de l'application. Le type de cet attribut est xsd:string. Cet attribut est facultatif.

version

Indique la version de l'application. Le type de cet attribut est xsd:string. Cet attribut est facultatif.

Pseudo-schéma

```
<application
  name="xsd:NCName"
  description="xsd:string"?
  version="xsd:string"?
  xsd:anyAttribute##other>
<xsd:any##other/>*
```

élément resources

Définition

Cet élément contient les exigences de ressources du travail pour lequel une correspondance doit être trouvée sur l'ordinateur cible afin qu'un travail soit affecté à ce système. Les éléments de l'exigence de ressources contenues sont combinés dans une relation AND. Autrement dit, chaque exigence s'ajoute aux autres pour raffiner l'exigence de ressource correspondante et toutes les exigences doivent être satisfaites pour que l'affectation réussisse. Cet élément est facultatif et peut être spécifié une fois. S'il n'est pas présent, le serveur Dynamic Workload Broker peut choisir n'importe quel ensemble de ressources pour exécuter le travail.

Type Le type de cet élément est jsdl:ResourceType. Les types pris en charge pour cet élément sont répertoriés dans le tableau 150, à la page 935. Il peut contenir les éléments suivants :

- candidateHosts
- candidateCPUs
- physicalMemory
- virtualMemory
- candidateOperatingSystems

- fileSystem
- logicalResource
- group
- propriétés
- allocation
- relationship
- candidateResources

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<resources
  xsd:anyAttribute##other>
  <candidateHosts .../>?
  <candidateCPUs .../>?
  <physicalMemory .../>?
  <virtualMemory .../>?
  <candidateOperatingSystems .../>?
  <fileSystem .../>*
  <logicalResource ...>*
  <group ...>*
  <properties ...>
  <allocation ...>*
  <relationship ...>*
  <candidateResources ...>?
  <xsd:any##other>*
</resources>
```

élément relatedResources

Définition

Cet élément est un identificateur unique pour l'exigence de ressource et doit être unique dans le document. Les exigences définies dans cet élément s'appliquent à la fois aux ressources logiques et aux systèmes d'ordinateurs. Il peut y être fait référence par les éléments **relationship**. Les éléments de l'exigence de ressources contenues sont combinés dans une relation AND. Autrement dit, chaque exigence s'ajoute aux autres pour raffiner l'exigence de ressource correspondante et toutes les exigences doivent être satisfaites pour que l'affectation réussisse. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est jsdl:RelatedResourceType. Il peut contenir les éléments suivants :

- logicalResource
- group
- propriétés
- allocation
- relationship
- candidateResources

Attributs

Les attributs suivants sont définis :

id Indique l'ID interne de la ressource à associer à l'élément de ressources. Cet ID est uniquement utilisé pour référence interne dans le fichier JSDL. Le type de cet attribut est xsd:ID. Cet attribut est obligatoire.

type Indique le type de la ressource requise. Les types pris en charge sont ComputerSystem et Logical Resource. Si cet attribut n'est pas présent, le système prend en compte le type de ressource ComputerSystem. Un type de ressource est identifié par un nom unique et décrit les propriétés fournies par chaque instance de ressources. Pour plus d'informations sur les propriétés de ressource disponibles, voir tableau 150, à la page 935. Le type de cet attribut est xsd:NCName. Cet attribut est facultatif.

Pseudo-schéma

```
<relatedResources
  id="xsd:ID"
  type="xsd:NCName"?
  xsd:anyAttribute##other>
  <logicalResource ...>*
  <group ...>*
  <properties ...>
  <allocation ...>*
  <relationship ...>*
  <candidateResources ...>?
  <xsd:any##other>*
</relatedResources>
```

élément candidateHosts

Définition

Cet élément spécifie l'ensemble des hôtes désignés qui doivent être sélectionnés pour l'exécution du travail. Dynamic Workload Broker affecte le travail à l'un des hôtes de cette liste. Les hôtes spécifiés sont dans la condition OR, autrement dit, au moins l'un d'entre eux doit trouver une correspondance de ressource de système d'exploitation contenue dans la ressource cible. Si aucun des hôtes que vous spécifiez n'est disponible lors de la soumission du travail, le travail attend que l'un d'entre eux devienne disponible. Si aucun hôte ne devient disponible avant l'expiration du délai, le travail échoue. Cet attribut est facultatif.

Type Le type de cet élément est jsdl:CandidateHostsRequirementType. Il peut contenir l'élément suivant :

- hostName

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<candidateHosts
  xsd:anyAttribute##other>
  <hostName>jsdl:StringVariableExpressionType<hostName/>+
  <xsd:any##other>*
</candidateHosts>
```

élément orderedCandidatedWorkstations

Définition

Cet élément indique la liste ordonnée des postes de travail qui sont candidats pour la sélection en fonction des informations spécifiées dans la zone d'exigence de la définition de pool dynamique. Le premier poste de travail qui correspond aux exigences est sélectionné pour le traitement.

Type Le type de cet élément est jsdl:OrderedCandidatedWorkstationsType. Il peut contenir l'élément suivant :

- Poste de travail

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<orderedCandidatedWorkstations
  xsd:anyAttribute##other>
  <workstation>jsdl:StringVariableExpressionType<workstation/>+
  <xsd:any##other>*
</orderedCandidatedWorkstations>
```

élément **hostName**

Définition

Cet élément indique une expression de variable de chaîne contenant le nom d'un hôte unique qui peut être sélectionné pour l'exécution du travail. Si aucun des hôtes que vous spécifiez n'est disponible lors de la soumission du travail, le travail attend que l'un d'entre eux devienne disponible. Si aucun hôte ne devient disponible avant l'expiration du délai, le travail échoue. Pour spécifier le nom d'hôte, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telle que \${var}, tout caractère et toute chaîne. Les caractères génériques sont autorisés. Cet attribut est obligatoire et peut être spécifié une ou plusieurs fois.

Type Le type de cet attribut est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<hostName>jsdl:StringVariableExpressionType<hostName/>

<hostName>lab145674.example.com<hostName/>

<hostName>${my_preferred_host}<hostName/>
```

élément **candidateCPUs**

Définition

Cet élément spécifie l'ensemble des caractéristiques d'unité centrale que doivent satisfaire les hôtes qui peuvent être sélectionnés pour l'exécution du travail. Les combinaisons de caractéristiques spécifiées sont dans la condition OR, autrement dit au moins l'un d'entre elles doit trouver une correspondance dans la ressource cible. Si aucune des caractéristiques d'unité centrale que spécifiez n'est disponible lors de la soumission d'un travail, le travail attend que l'une d'entre elles devienne disponible. Si aucune caractéristique d'unité centrale ne devient disponible avant l'expiration du délai, le travail échoue. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est jsdl:CandidateCPUsRequirementType. Il peut contenir l'élément suivant :

- UC

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<candidateCPUs
  xsd:anyAttribute##other>
  <cpu ...>*
  <xsd:any##other/>*
</candidateCPUs >?
```

élément cpu

Définition

Cet élément spécifie les caractéristiques d'unité centrale que doivent satisfaire les hôtes qui peuvent être sélectionnés pour l'exécution du travail. Les combinaisons de caractéristiques spécifiées sont dans la condition OR, autrement dit au moins l'un d'entre elles doit trouver une correspondance dans la ressource cible. Au moins l'une des unités centrales avec les caractéristiques indiquées doit être disponible pour l'exécution du travail. Si aucun des unités centrales que vous indiquez n'est disponible lors de la soumission du travail, le travail attend que l'une d'elles devienne disponibles. Si aucune caractéristique d'unité centrale ne devient disponible avant l'expiration du délai, le travail échoue. Cet élément est obligatoire et peut être spécifié une ou plusieurs fois.

Type Le type de cet élément est CPURequirementType.

Attributs

Les attributs suivants sont définis :

architecture

Indique l'architecture d'unité centrale requise pour l'exécution du travail. Cet attribut est facultatif. Les valeurs prises en charge sont les suivantes :

- parisc
- powerpc
- powerpc_64
- s390
- s390x
- sparc
- x86
- x86_64

quantity

Indique le nombre de processeurs qui doivent être disponibles sur l'ordinateur. Définir Quantity sur 0 indique que l'exigence est remplie par des ordinateurs comprenant tout nombre de processeurs. Le type de cet attribut est xsd:unsignedInt. Cet attribut est facultatif.

Pseudo-schéma

```
<cpu>
  architecture="xsd:string"?
  speed="jsdl:NumericRangeType"?
  quantity="xsd:unsignedInt"?
  xsd:anyAttribute##other>
  <xsd:any##other>*
</cpu>
```

élément speed

Définition

Cet élément indique la plage de vitesse d'unité centrale en MHz requise pour exécuter le travail. L'unité de mesure est le mégahertz. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:NumericRangeType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<speed>
  xsd:anyAttribute##other>
  <jsdl:NumericRangeType>*
</speed>
```

élément physicalMemory

Définition

Cet élément spécifie une plage ou la valeur exacte indiquant la quantité de mémoire physique disponible requise pour le travail. La quantité est exprimée en octets. Cet attribut est facultatif.

Type Le type de cet élément est jsdl:NumericRangeType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<physicalMemory
  xsd:anyAttribute##other>
  jsdl:NumericRangeType
  <xsd:any##other>*
</physicalMemory>
```

élément virtualMemory

Définition

Cet élément spécifie une plage ou la valeur exacte indiquant la quantité de mémoire virtuelle disponible requise pour le travail. La quantité est exprimée en octets. Cet attribut est facultatif.

Type Le type de cet élément est jsdl:NumericRangeType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<virtualMemory
  xsd:anyAttribute##other>
  jsdl:NumericRangeType
  <xsd:any##other>*
</virtualMemory>
```

élément candidateOperatingSystems

Définition

Cet élément spécifie l'ensemble des caractéristiques de système d'exploitation que doivent satisfaire des hôtes qui peuvent être sélectionnés pour l'exécution du travail. Les combinaisons de caractéristiques trouvent des correspondances OR, autrement dit, au moins l'une d'entre elles doit être satisfaite par la ressource de système d'exploitation contenue dans la ressource cible. Au moins un des systèmes d'exploitation répertoriés doit

être disponible pour que le travail s'exécute. Si aucun des systèmes d'exploitation que vous spécifiez n'est disponible lors de la soumission du travail, le travail attend que l'un d'entre eux devienne disponible. Si aucun système d'exploitation ne devient disponible avant l'expiration du délai, le travail échoue. Cet attribut est facultatif.

Type Le type de cet élément est `jsdl:OperatingSystemsRequirementType`. Il peut contenir l'élément suivant :

- `operatingSystem`

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<candidateOperatingSystems
  <xsd:anyAttribute##other>
  <operatingSystem...>*
  <xsd:any##other/>*
</ candidateOperatingSystems>?
```

élément `operatingSystem`

Définition

Cet élément spécifie les caractéristiques de système d'exploitation qui sont requises pour exécuter le travail. Cet attribut est obligatoire et peut être spécifié une ou plusieurs fois.

Type Le type de cet élément est `OperatingSystemRequirementType`.

Attributs

Les attributs suivants sont définis :

type Cet attribut définit le nom du système d'exploitation requis pour l'exécution du travail. Le type de cet attribut est `xsd:string`. Cet attribut est obligatoire. Les valeurs prises en charge sont les suivantes :

- AIX
- Linux
- Windows 2000
- Windows 2003
- Windows XP
- Windows Vista
- HP-UX
- Solaris

version

Spécifiez la version du système d'exploitation. Vous pouvez indiquer la version ou sous-version exacte du système d'exploitation, par exemple 5.2 ou 5.2.3.30, ou vous pouvez indiquer une partie de la version, par exemple 5.2.3. Dans ce cas, l'exigence s'applique à tous les systèmes d'exploitation possédant la version 5.2.3 et ses sous-versions, tels que les groupes de correctifs et les niveaux de maintenance. Le type de cet attribut est `xsd:string`. Cet attribut est facultatif.

Pseudo-schéma

```
<operatingSystem
  type="xsd:string"
  version="xsd:string"?
  xsd:anyAttribute##other>
<xsd:any##other>*
</operatingSystem>
```

élément fileSystem

Définition

Cet élément décrit l'ensemble des caractéristiques de système de fichiers qui peuvent être sélectionnées pour l'exécution du travail. Chaque ensemble de caractéristiques spécifie l'emplacement dans lequel le système de fichiers est disponible, la quantité d'espace disque requise et le type du système de fichiers. Le système de fichiers peut se trouver dans le même emplacement que la ressource, par exemple sur un disque local, ou peut être distant, par exemple sur un montage NFS. L'exigence est satisfaite si pour une cible données, toutes les caractéristiques du système de fichiers sont satisfaites. Les combinaisons de caractéristiques sont satisfaites par AND, autrement dit, elles doivent toutes être satisfaites par les ressources de système de fichiers contenues dans la ressource cible. Tous les systèmes de fichiers répertoriés doivent être présents pour que le travail s'exécute. Si l'un des systèmes de fichiers que vous indiquez n'est pas disponible lors de la soumission du travail, le travail attend qu'il devienne disponible. S'il ne devient pas disponible avant l'expiration du délai, le travail échoue. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est jsdl:FileSystemRequirementType. Il contient l'élément **diskSpace**.

Attributs

Les attributs suivants sont définis :

type Jeton spécifiant le type de système de fichiers de l'élément fileSystem conteneur. Le type de cet attribut est jsdl:FileSystemTypeEnumeration. Cet attribut est facultatif. Les valeurs prises en charge sont les suivantes :

Unknown

Le système de fichiers n'est pas spécifié.

No Root Directory

Désigne un système de fichiers local qui n'est pas le répertoire racine.

Removable Disk

Désigne un système de fichiers monté sur un disque dur amovible.

Local Disk

Désigne un système de fichiers monté sur un disque local.

Remote Drive

Désigne un système de fichiers monté sur une unité distante.

CD-ROM

Désigne un système de fichiers monté sur un lecteur de CD-ROM.

RAM Disk

Désigne un système de fichiers monté sur un disque RAM.

mountPoint

Expression de variable de chaîne spécifiant le mappage local sur lequel le système de fichiers est disponible pour le travail. Le type de cet attribut est jsdl:StringVariableExpressionType. Pour spécifier le point de montage, vous pouvez utiliser une expression de variables qui peut contenir une ou plusieurs références de variables, telles que \${var}, tout caractère et toute chaîne. Les caractères génériques sont autorisés. Cet attribut est facultatif.

Pseudo-schéma

```
<fileSystem
  type="jsdl:FileSystemTypeEnumeration"?
  mountPoint="jsdl:StringVariableExpressionType"?
  xsd:anyAttribute##other>
<diskSpace>jsdl:NumericRangeType</diskSpace>?
<xsd:any##other/>*
</fileSystem>
```

élément diskSpace

Définition

Cet élément indique la quantité d'espace disque sur l'élément de système de fichiers conteneur pour l'exécution du travail. La quantité d'espace disque est exprimée en kilooctets. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:NumericRangeType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<fileSystem
  type="jsdl:FileSystemTypeEnumeration"?
  mountPoint="jsdl:StringVariableExpressionType"?
  xsd:anyAttribute##other>
<diskSpace>jsdl:NumericRangeType</diskSpace>?
<xsd:any##other/>*
</fileSystem>
```

élément logicalResource

Définition

Cet élément spécifie une ou plusieurs ressources logiques requises pour exécuter le travail. Les combinaisons de caractéristiques sont satisfaites par AND, autrement dit, elles doivent toutes être remplies par les ressources logiques associées à la ressource cible. Toutes les ressources logiques répertoriées doivent être disponibles pour que le travail s'exécute. Si l'une des ressources logiques que vous spécifiez est indisponible lors de la soumission du travail, le travail attend qu'elle devienne disponible. S'il ne devient pas disponible avant l'expiration du délai, le travail échoue. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est LogicalResourceRequirementType.

Attributs

Les attributs suivants sont définis :

name Expression de variable de chaîne spécifiant le nom de la ressource logique demandée. Pour spécifier le nom de ressource logique,

vous pouvez utiliser une expression de variable qui peut contenir un ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne. Ce nom doit commencer par un caractère alphabétique et peut comporter les symboles suivants : soulignement (_), moins (-), et point (.). Les espaces, les caractères spéciaux et les caractères accentués ne sont pas pris en charge. Cet attribut est facultatif.

subType

Expression de variable de chaîne spécifiant le type de la ressource logique demandée. Pour spécifier le type de ressource logique, vous pouvez utiliser une expression de variable qui peut contenir un ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne. Cet attribut est facultatif.

quantity

Valeur entière spécifiant la quantité de ressource logique requise. La quantité spécifiée est affectée exclusivement au travail en cours d'exécution. Pour spécifier la quantité de ressource, vous pouvez utiliser une expression de variable pouvant contenir une référence de variable, telle que \${var} ou une valeur entière non signée. Cet attribut est facultatif.

Pseudo-schéma

```
<logicalResource
  name="jsdl:StringVariableExpressionType"?
  subType="jsdl:StringVariableExpressionType"?
  quantity="jsdl:UnsignedIntVariableExpressionType"?
  xsd:anyAttribute##other>
</logicalResource>
```

élément group

Définition

Cet élément spécifie que les ressources requises pour l'exécution du travail doivent appartenir au groupe de ressources spécifié. Les groupes sont satisfaits par AND, c'est-à-dire que la ressource cible doit se trouver dans tous les groupes spécifiés. Cet élément est facultatif et peut être spécifié zéro ou plusieurs fois.

Type Le type de cet élément est jsdl:GroupRequirementType.

Attributs

L'attribut suivant est défini :

name Indique le nom du groupe en une expression de variable de chaîne. Pour spécifier le groupe de ressources, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne. Cet attribut est obligatoire et peut être spécifié une fois.

Pseudo-schéma

```
<group
  name="jsdl:StringVariableExpressionType"
  xsd:anyAttribute##other>
</group>
```

élément properties

Définition

Cet élément spécifie les propriétés de la ressource requise pour l'exécution du travail. L'exigence est exprimée comme un ensemble de conditions sur les propriétés de ressources combinées avec des opérateurs AND/OR. Celles-ci sont basées sur le modèle de ressource qui décrit les ressources disponibles dans l'environnement comme des instances des types de ressources. Un type de ressource est identifié par un nom unique et décrit les propriétés fournies par chaque instance de ressources. Utilisez cet élément pour spécifier les exigences avancées. Pour plus d'informations sur les types et propriétés de ressource disponibles, voir tableau 150, à la page 935. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:RequirementCompositorType. Il peut contenir les éléments suivants :

- and
- ou
- exigence

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<properties
  xsd:anyAttribute##other>
<and .../>?
<or .../>?
<requirement .../>?
<xsd:any##other/*>
</properties>
```

élément and

Définition

Cet élément spécifie une condition AND sur les spécifications d'exigence de conteneur. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:RequirementCompositorType. Il peut contenir les éléments suivants :

- and
- ou
- exigence

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<and
  xsd:anyAttribute##other>
<and .../>?
<or .../>?
<requirement .../>?
<xsd:any##other/*>
</and>
```

élément or

Définition

Cet élément spécifie une condition OR sur les spécifications d'exigence de conteneur. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:RequirementCompositorType. Il peut contenir les éléments suivants :

- and
- ou
- exigence

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<or
  xsd:anyAttribute##other>
  <and .../>?
  <or .../>?
  <requirement .../>?
  <xsd:any##other/>*
</or>
```

élément de condition requise

Définition

Cet élément est une valeur de plage spécifiant une exigence sur les fonctions d'une ressource pour l'exécution du travail. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:RequirementType.

Attributs

L'attribut suivant est défini :

propertyName

Chaîne spécifiant la propriété de ressource à laquelle s'applique l'exigence. Les propriétés disponibles varient selon les ressources que vous avez sélectionnées dans l'élément ressources. Pour plus d'informations sur les types et propriétés de ressources, voir tableau 150, à la page 935. Le type de cet attribut est xsd:NCName. Cet attribut est obligatoire.

Pseudo-schéma

```
<requirement
  propertyName="xsd:NCName"
  xsd:anyAttribute##other>
  jsdl:StringRangeType
  <xsd:any##other/>*
</and>
```

élément allocation

Définition

Cet élément spécifie une exigence d'affectation exclusive sur la propriété donnée d'une ressource. Vous pouvez définir une exigence d'affectation sur les attributs de consommable des ressources. Pour plus d'informations sur les attributs de consommables, voir tableau 150, à la page 935. Le travail peut être exécuté sur la ressource si la valeur requise est disponible. Lorsqu'il s'exécute, le travail utilise exclusivement la valeur de propriété de ressource requise. Lorsque le travail se termine, il libère la valeur de propriété. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:AllocationRequirementType.

Attributs

L'attribut suivant est défini :

propertyName

Chaîne spécifiant la propriété de ressource à laquelle s'applique l'exigence. Le type de cet attribut est xsd:QName. Cet attribut est obligatoire.

Quantity

Spécifie la quantité de la propriété qui doit être affectée exclusivement au travail. Pour spécifier la quantité de la propriété qui doit être affectée, vous pouvez utiliser une expression de variable qui peut contenir une référence de variable, telle que \${var} ou une double valeur.

Pseudo-schéma

```
<allocation
  propertyName="xsd:QName"
  xsd:anyAttribute##other>
  jsdl:DoubleVariableExpressionType
<xsd:any##other/*
</and>
```

élément relationship**Définition**

Cet élément spécifie l'exigence que la ressource sélectionnée pour exécuter le travail doit avoir une relation avec d'autres ressources correspondant à certains critères supplémentaires. Une relation est une association directe entre une ressource source et une ressource cible. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:RelationshipRequirementType.

Attributs

Les attributs suivants sont définis :

- type** Chaîne spécifiant le type de relation requis. Le type de cet attribut est xsd:NCName. Cet attribut est facultatif.
- source** Chaîne spécifiant l'ID d'un élément **relatedResources**. Si cet attribut est spécifié, l'exigence de relation indique que la ressource doit avoir au moins une relation dirigée vers elle-même depuis une ou plusieurs ressources satisfaites par l'élément **relatedResources**. Si cet attribut n'est pas spécifié, un attribut cible doit être présent. Le type de cet attribut est xsd:IDREF. Cet attribut est facultatif.
- target** Chaîne spécifiant l'ID d'un élément **relatedResources**. Si cet attribut est spécifié, l'exigence de relation indique que la ressource doit avoir au moins une relation dirigée vers elle-même depuis une ou plusieurs ressources satisfaites par l'élément **relatedResources**. Si cet attribut n'est pas spécifié, un attribut source doit être présent. Le type de cet attribut est xsd:IDREF. Cet attribut est facultatif.

Pseudo-schéma

```
<relationship
  type="xsd:NCName"
  source="xsd:IDREF"?
  target="xsd:IDREF"?
  xsd:anyAttribute##other>
<xsd:any##other/*
</relationship>
```

élément **candidateResources**

Cet élément est réservé pour une utilisation interne.

Définition

Cet élément spécifie l'ensemble des ressources qui peuvent être sélectionnées pour l'exécution du travail. Si cet élément est spécifié, une ou plusieurs ressources de l'ensemble doivent être choisies pour l'exécution du travail. Les ressources sont identifiées au moyen de leur adresse de référence de noeud final (référence de noeud final d'adressage WS) du service de fabrique de travaux gérant la ressource. Les combinaisons d'exigences trouvent des correspondances OR, autrement dit au moins l'une d'entre elles doit être satisfaite par la ressource contenue dans la ressource cible. Au moins une des ressources répertoriées doit être disponible pour que le travail s'exécute. Si aucun des ressources que vous spécifiez n'est disponible lors de la soumission du travail, le travail attend que l'une d'entre elles devienne disponible. Si aucune ressource ne devient disponible avant l'expiration du délai, le travail échoue. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:CandidateResourcesRequirementType. Il contient l'élément **endpointReference**.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<candidateResources
  xsd:anyAttribute##other>
  <endpointReference>wsa:EndpointReferenceType<endpointReference/>+
  <xsd:any##other>*
</candidateResources>
```

élément **endpointReference**

Cet élément est réservé pour une utilisation interne.

Définition

Cet élément spécifie la référence de noeud final d'adressage de services Web du service de fabrique de travaux gérant la ressource. Cet élément est obligatoire et peut être spécifié une ou plusieurs fois. Cet élément est réservé pour une utilisation interne.

Type Le type de cet élément est wsa:EndpointReferenceType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<candidateResources
  xsd:anyAttribute##other>
  <endpointReference>wsa:EndpointReferenceType<endpointReference/>+
  <xsd:any##other>*
</candidateResources>
```

élément **optimization**

Définition

Cet élément spécifie les règles d'optimisation à appliquer au travail. Selon que vous sélectionnez l'objectif ou l'élément ewlm, la règle de sélection de ressources change. Cet élément est facultatif et peut être spécifié une fois.

Si aucune valeur n'est spécifiée, le règle d'équilibrage de charge par défaut consistant à égaliser le nombre de travaux qui s'exécutent sur chaque ressource est appliquée.

Type Le type de cet élément est `jsdl:OptimizationType`. Il ne peut contenir que les éléments suivants :

- `objective`
- `ewlm`

Attributs

L'attribut suivant est défini :

name Indique le nom de la règle d'optimisation. Les valeurs prises en charge sont les suivantes :

JPT_JSDLOptimizationPolicyType

Si vous utilisez cette option, vous devez spécifier l'élément **objective**. A l'aide de l'élément **objective**, vous pouvez spécifier que les propriétés de ressources soient maximisées ou réduites. Lorsque vous définissez l'élément **objective**, Dynamic Workload Broker exécute le travail sur la ressource qui correspond à l'exigence d'optimisation. Il s'agit de la valeur par défaut. Voir «Ressources de la définition de travail», à la page 934 pour plus d'informations.

JPT_BestResource

Utilisez cette option afin que le pool de ressources du travail soit constitué uniquement des meilleures ressources parmi le pool de ressources qui répondent à la règle spécifiée. Si vous utilisez cette option, vous devez spécifier l'élément **objective**. A l'aide de l'élément **objective**, vous pouvez spécifier que les propriétés de ressources soient maximisées ou réduites. Lorsque vous définissez l'élément **objective**, Dynamic Workload Broker exécute le travail sur la ressource qui correspond à l'exigence d'optimisation. Il s'agit de la valeur par défaut. Voir «Ressources de la définition de travail», à la page 934 pour plus d'informations.

JPT_EWLM

Si vous utilisez cette option, l'élément **ewlm** est automatiquement inséré.

Pseudo-schéma

```
<optimization
  name="xsd:NCName"
  xsd:anyAttribute##other>
  <objective .../> |
  <ewlm .../>
  <xsd:any##other>*
</optimization>
```

élément **objective**

Définition

Cet élément spécifie l'objectif à atteindre lorsque vous effectuez l'optimisation d'un travail. Par exemple, si vous sélectionnez l'utilisation d'unité centrale **resourcePropertyName** pour le système informatique **resourceType** avec une réduction **propertyObjective**, Dynamic Workload

Broker tentera d'exécuter le travail sur la ressource où l'utilisation d'unité centrale est la plus faible. Cet élément et l'élément **ewlm** s'excluent mutuellement.

Type Le type de cet élément est PropertyObjectiveType.

Attributs

Les attributs suivants sont définis :

propertyObjective

Chaîne spécifiant le type d'objet. Cet attribut est obligatoire. Les valeurs prises en charge sont les suivantes :

minimize

Les ressources sont affectées au travail avec l'objectif de réduire la valeur de la propriété de type de ressource spécifiée. Par exemple, vous pouvez choisir cet objectif pour attribuer le travail à la ressource dans laquelle l'utilisation de l'unité centrale est la plus faible.

maximize

Les ressources sont affectées au travail avec l'objectif de maximiser la valeur de la propriété de type de ressource spécifiée. Par exemple, vous pouvez choisir cet objectif pour affecter le travail à la ressource dans laquelle la vitesse de traitement est la plus élevée.

minimize.utilization

Les ressources sont affectées au travail avec l'objectif de réduire l'utilisation de la propriété de type de ressource spécifiée. Cet attribut est uniquement disponible pour les propriétés de consommables. Pour une liste de toutes les propriétés de consommable, voir tableau 150, à la page 935. Si vous choisissez de minimiser l'utilisation de la consommation de la propriété, le travail est affecté à une ressource sur laquelle une quantité moindre de la propriété est utilisée.

maximize.utilization

Les ressources sont affectées au travail avec l'objectif de maximiser l'utilisation de la propriété du type de ressource spécifiée. Cet attribut est uniquement disponible pour les propriétés de consommables. Pour une liste de toutes les propriétés de consommable, voir tableau 150, à la page 935. Par exemple, vous aurez peut-être besoin d'effectuer un test de charge sur un poste de travail en créant des travaux dans lesquels la propriété de ressource d'utilisation d'unité centrale est définie par Maximize Utilization. Dans ce cas, tous les travaux présentant ce paramètre seraient affectés au poste de travail sur lequel l'utilisation de l'unité centrale est supérieure, créant une boucle.

resourceType

Chaîne spécifiant le type de la ressource à laquelle la règle s'applique. Si cet élément n'est pas spécifié, le système prend en compte le type de ressource ComputerSystem. Le type de cet attribut est xsd:QName. Cet attribut est facultatif.

resourcePropertyName

Chaîne spécifiant la propriété de ressource à laquelle la règle s'applique. Le type de cet attribut est `xsd:QName`. Cet attribut est obligatoire.

Remarque : Lorsque vous indiquez une exigence d'optimisation sur une propriété d'un type de ressource, vous devez avoir défini préalablement une exigence sur ce type de ressource. Par exemple, si vous souhaitez optimiser la mémoire physique totale sur un système d'exploitation, vous devez préalablement définir une exigence sur le type de ressource de système d'exploitation. Cette procédure ne s'applique pas au type de ressource `Computer System`, car celui-ci est le type de ressource par défaut.

Pseudo-schéma

```
<objective
  propertyObjective="minimize" | "maximize"
  resourceType="xsd:QName"?
  resourcePropertyName="xsd:QName"
  xsd:anyAttribute##other>
<xsd:any##other>*
</objective>
```

élément ewlm

Définition

Cet élément spécifie l'optimisation en fonction du calcul du poids de ressource d'Enterprise Workload Manager. Dynamic Workload Broker exécutera le travail sur les meilleures ressources disponibles telles qu'elle sont indiquées par Enterprise Workload Manager. Cet élément est facultatif et peut être spécifié une fois. Cet élément et l'élément **objective** s'excluent mutuellement.

Type Le type de cet élément est `jsdl:PropertyObjectiveType`.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<ewlm>
  xsd:anyAttribute##other>
<xsd:any##other>*
</ewlm>
```

élément scheduling

Définition

Cet élément spécifie les paramètres de planification à appliquer lors de l'exécution du travail. Cet élément est facultatif et peut être spécifié une fois.

Dans Dynamic Workload Broker, cet élément correspond à la page Planification. Pour plus d'informations sur la page Planification, voir la documentation de l'aide en ligne.

Type Le type de cet élément est `jsdl:SchedulingType`. Il peut contenir les éléments suivants :

- `maximumResourceWaitingTime`
- `estimatedDuration`
- `priority`

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<scheduling
  xsd:anyAttribute##other>
  <maximumResourceWaitingTime>xsd:duration<maximumResourceWaitingTime>?
  <estimatedDuration>xsd:duration<estimatedDuration>?
  <priority>xsd:unsignedint<priority>?
  <xsd:any##other>*
</objective>
```

élément maximumResourceWaitingTime

Définition

Cet élément spécifie combien de temps le serveur Dynamic Workload Broker doit attendre depuis la soumission du travail avant de décider qu'aucune ressource ne correspond aux exigences. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est xsd:duration.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<maximumResourceWaitingTime>?
  xsd:anyAttribute##other>
  <xsd:duration>*
<maximumResourceWaitingTime>
```

élément estimatedDuration

Définition

Cet élément spécifie la durée estimée d'exécution du travail. Celle-ci peut être utilisée par le serveur Dynamic Workload Broker pour planifier l'affectation des ressources. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est xsd:duration.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<estimatedDuration>?
  <xsd:anyAttribute##other>
  <xsd:duration>*
<estimatedDuration>
```

élément priority

Définition

Cet élément spécifie la priorité du travail sous forme d'un entier compris entre 0 et 100. Plus les valeurs sont hautes, plus la priorité est haute. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est xsd:PriorityType. Il peut contenir les éléments suivants :

- maximumResourceWaitingTime
- estimatedDuration
- priority

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<scheduling
  xsd:anyAttribute##other>
  <maximumResourceWaitingTime>xsd:duration<maximumResourceWaitingTime>?
  <estimatedDuration>xsd:duration<estimatedDuration>?
  <priority>xsd:unsignedint<priority>?
  <xsd:any##other>*
</objective>
```

recoveryActions

Définition

Cet élément décrit la liste des actions de reprise que Dynamic Workload Broker doit effectuer si l'intervalle de temps spécifié dans l'élément `maximumResourceWaitingTime` arrive à expiration et qu'aucune ressource correspondant aux exigences n'a été trouvée. Cet élément est facultatif et peut être spécifié une fois. Les actions de reprise définies dans cet élément sont effectuées en démarrant un flux de travaux Tivoli Provisioning Manager.

Type Le type de cet élément est `jsdl:RecoveryActionList`. Il peut contenir l'élément `action`.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<recoveryActions
  xsd:anyAttribute##other>
  <action...>+
  <xsd:any##other/>*
</recoveryActions >
```

action

Définition

Cet élément spécifie les actions de reprise à effectuer. Dynamic Workload Broker exécute les actions répertoriées de façon séquentielle en fonction de l'ordre dans lequel elles ont été spécifiées. Toute action de reprise suivante est exécutée uniquement si l'action précédente s'est terminée avec succès. Si l'élément **recoveryActions** est spécifié, au moins un élément **action** doit être spécifié.

Type Le type de cet élément est `jsdl:RecoveryActionType`.

Attributs

Les attributs suivants sont définis :

name Spécifie le nom de l'action de reprise à effectuer. Cet attribut est facultatif.

additionalTimeOnCompletion

Spécifie l'intervalle de temps pendant lequel Dynamic Workload Broker doit attendre pour que l'action de reprise devienne effective après achèvement. Si cet attribut est spécifié pour une action de reprise, l'action de reprise suivante est effectuée seulement après expiration de l'intervalle de temps spécifié. Si la ressource requise devient disponible avant que l'intervalle expire, Dynamic Workload Broker peut décider d'exécuter le travail avant que l'action se termine.

maximumExecutionTime

Spécifie la période de temps attendue pendant laquelle Dynamic Workload Broker attend que l'action de reprise se termine. Si l'action de reprise n'est pas terminée à l'expiration de ce délai, la procédure de reprise échoue et la séquence de reprise est arrêtée.

Pseudo-schéma

```
< action>
  name="xsd:NCName"
  additionalTimeOnCompletion ="xsd:duration"?
  maximumExecutionTime ="xsd:duration"?
  xsd:anyAttribute##other>
<xsd:any##other/*
</ action >
```

élément tpmaction

Définition

Cet élément spécifie les paramètres requis pour exécuter une action de reprise Tivoli Provisioning Manager. Cet attribut est facultatif et peut être spécifié une fois.

Type Le type de cet attribut est jsdltpm:TPMActionType. Il peut contenir les éléments suivants :

- parameters
- credential
- tpmaddress
- workflow

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<tpmaction
  <parameters... />?
  <credential.../>?
  <tpmaddress.../>?
  workflow="jsdl:StringVariableExpressionType"
</tpmaction>
```

élément parameters

Définition

Cet élément indique les arguments à utiliser lors de l'exécution du flux de travaux Tivoli Provisioning Manager. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet attribut est jsdltpm:ParametersType.

Attributs

Les attributs suivants sont pris en charge :

name Indique le nom de l'attribut à utiliser lors de l'exécution du flux de travaux Tivoli Provisioning Manager. Cet attribut est obligatoire et peut être spécifié une ou plusieurs fois.

Pseudo-schéma

```
<parameters
  <parameter... />+
</parameters>
```


élément credential

Définition

Cet élément spécifie les données d'identification requises pour l'exécution du flux de travaux Tivoli Provisioning Manager. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:CredentialType. Il peut contenir les éléments suivants :

- userName
- mot de passe

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential>  
  <userName> jsdl:NotEmptyStringVariableExpressionType</userName>  
  <password> jsdl:StringVariableExpressionType </password>  
</credential>
```

élément userName

Définition

Cet élément indique un nom d'utilisateur défini sur le système cible, qui est utilisé pour exécuter le flux de travaux Tivoli Provisioning Manager. Cet élément est obligatoire si vous utilisez l'élément credential et peut être spécifié une fois.

Type Le type de cet élément est jsdl:NotEmptyStringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<userName>  
  xsd:anyAttribute##other>  
<jsdl:NotEmptyStringVariableExpressionType*  
</userName>
```

élément password

Définition

Cet élément spécifie le mot de passe de l'utilisateur spécifié utilisé pour exécuter le flux de travaux Tivoli Provisioning Manager sur le système cible. Cet élément est obligatoire si vous utilisez l'élément credential et peut être spécifié une fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<password>?  
  xsd:anyAttribute##other>  
<jsdl:StringVariableExpressionType  
</password>
```

élément tpmaddress

Définition

Cet élément spécifie l'adresse Tivoli Provisioning Manager qui doit être

utilisée pour appeler le service Web Tivoli Provisioning Manager nécessaire à l'exécution du flux de travaux. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:tpm:TPMAddressType.

Attributs

Les attributs suivants sont définis :

- host** Spécifie le nom d'hôte du serveur Tivoli Provisioning Manager à utiliser lors de l'exécution de l'action de reprise.
- port** Spécifie le numéro de port du serveur Tivoli Provisioning Manager à utiliser lors de l'exécution de l'action de reprise.

Pseudo-schéma

```
< tpmaddress  
  <host... />  
  <port... />  
</ tpmaddress >
```

élément workflow

Définition

Cet élément indique le nom du flux de travaux Tivoli Provisioning Manager à exécuter. Pour spécifier le nom de flux de travaux, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne. Cet élément est obligatoire.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<workflow>?  
  xsd:anyAttribute##other>  
<jsdl:StringVariableExpressionType  
<workflow>
```

élément exécutable

Définition

Cet élément spécifie les paramètres pour l'exécution d'une commande système native, c'est-à-dire des fichiers exécutables et des scripts. Vous pouvez également imbriquer des scripts dans cet élément. Cet élément est obligatoire.

Remarque : Les restrictions suivantes s'appliquent :

- Sur les systèmes Windows, vous pouvez exécuter des scripts contenant des commandes de traitement par lots. Les formats de script pris en charge sont :
 - .cmd
 - .bat
- Sur les systèmes UNIX et Linux, seuls les scripts de shell sont pris en charge. Au début du script de shell, vous devez indiquer l'interpréteur de commandes.
- Sur les systèmes UNIX et Linux, les commandes contenues dans les scripts doivent s'exécuter en avant-plan. Cela signifie que vous ne pouvez pas utiliser le paramètre "&" en association avec la commande.

- Sur aucune des plateformes prises en charge, vous ne pouvez inclure dans les travaux des commandes démarrant un programme au moyen d'une interface graphique.

Type Le type de cet élément est `jsdle:ExecutableType`. Il peut contenir les éléments suivants :

- `script`
- `arguments`
- `environment`
- `credential`

Attributs

Les attributs suivants sont définis :

path Expression de variable de chaîne spécifiant le nom de chemin du fichier exécutable à exécuter. Si l'élément **script** n'est pas présent, l'attribut **path** doit être spécifié. Si l'élément **script** est présent, l'attribut **path** ne peut pas être spécifié. Pour indiquer le chemin, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne.

Vous devez indiquer l'extension de fichier. Si vous souhaitez exécuter un fichier exécutable sans spécifier son extension, vous pouvez spécifier le nom du fichier exécutable dans l'élément **script**, de sorte que le fichier soit exécuté dans le shell.

input Expression de variable de chaîne spécifiant l'entrée standard de la commande. Cet attribut est le nom de chemin absolu ou le nom de chemin relatif du répertoire de travail. Pour indiquer le chemin, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne. Cet attribut est facultatif.

output

Expression de variable de chaîne spécifiant la sortie standard de la commande. Cet attribut est le nom de chemin absolu ou le nom de chemin relatif du répertoire de travail. Pour indiquer le chemin, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne. Cet attribut est facultatif.

error Expression de variable de chaîne spécifiant l'erreur standard pour la commande. Cet attribut est le nom de chemin absolu ou le nom de chemin relatif du répertoire de travail. Pour indiquer le chemin, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne. Cet attribut est facultatif.

workingDirectory

Expression de variable de chaîne spécifiant le répertoire de travail requis par le travail à exécuter. Pour spécifier le répertoire, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne. Cet attribut est facultatif. Si vous ne spécifiez pas cet attribut, le travail s'exécute dans les répertoires suivants, selon le système d'exploitation :

Sous UNIX

Les cas suivants s'appliquent :

- Le travail s'exécute dans le répertoire \$HOME_DIRECTORY de l'utilisateur qui soumet le travail, s'il existe.
- Si le répertoire n'existe pas, le travail s'exécute sur /root, si l'utilisateur qui soumet le travail possède les droits requis.
- Si l'utilisateur ne possède pas les droits requis, le travail s'exécute dans le répertoire de travail de l'agent IBM Workload Scheduler.

Sous Windows

Le travail s'exécute dans le répertoire de travail de l'agent IBM Workload Scheduler.

script Spécifie le code de script à exécuter. Pour spécifier des caractères spéciaux requis par les langages de script, le contenu de l'élément script peut être indiqué à l'aide d'une section CDATA.

Pseudo-schéma

```
<executable
  path="jsdl:StringVariableExpressionType"
  input="jsdl:StringVariableExpressionType"?
  output="jsdl:StringVariableExpressionType"?
  error="jsdl:StringVariableExpressionType"?
  workingDirectory="jsdl:StringVariableExpressionType"?
  xsd:anyAttribute##other>
  <script ... />?
  <arguments .../>?
  <environment .../>?
  <credential .../>?
  <xsd:any##other>*
</executable>
```

élément script

Définition

Cet élément spécifie le code de script à exécuter. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est xsd:string.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<script>?
  xsd:anyAttribute##other>
  <xsd:string
</script>
```

élément arguments

Définition

Cet élément spécifie la liste des arguments sous forme d'expressions de variable qui sont concaténées pour produire la chaîne d'argument à transmettre à la commande. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdle:ArgumentsType. Il peut contenir l'élément suivant :

- value

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<arguments
  xsd:anyAttribute##other>
  <value>jsdl:StringVariableExpressionType</value>+
  <xsd:any##other>*
</arguments>
```

élément value

Définition

Cet élément spécifie la valeur de l'élément **arguments**. Pour spécifier la valeur, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que $\{var\}$, tout caractère et toute chaîne. Cet élément est obligatoire et peut être spécifié une ou plusieurs fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<arguments
  xsd:anyAttribute##other>
  <value>jsdl:StringVariableExpressionType</value>+
  <xsd:any##other>*
</arguments>
```

Remarque : Si vous avez besoin de spécifier qu'une valeur consiste en un espace blanc, vous devez l'entourer de guillemets.

élément environment

Définition

Cet élément indique une expression de variable de chaîne de variables d'environnement qui seront définies pour le travail dans l'environnement en cours d'exécution. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est jsdl:EnvironmentType. Il peut contenir l'élément suivant :

- variable

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
environnement <
  xsd:anyAttribute##other>
  <variable name="xsd:string">jsdl:StringVariableExpressionType</variable>+
  <xsd:any##other>*
</environnement>
```

Remarque : Si vous avez besoin de spécifier qu'une valeur consiste en un espace blanc, vous devez l'entourer de guillemets.

élément variable

Définition

Cet élément indique une expression de variable de chaîne de variables d'environnement qui seront définies pour le travail dans l'environnement en cours d'exécution. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est `jsdl:StringVariableExpressionType`.

Attributs

Les attributs suivants sont définis :

name Spécifie le nom de la variable.

value Spécifie la valeur de la variable. Pour spécifier la valeur de la variable, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne.

élément credential

Définition

Cet élément spécifie les données d'identification pour l'exécution de la commande. Incluez cet élément lorsque vous voulez spécifier un nom d'utilisateur ou de groupe avec lequel l'exécutable ou le script s'exécute sur le système cible, qui est différent du nom d'utilisateur ou de groupe avec lequel l'agent de charge de travail s'exécute. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est `jsdle:CredentialType`. Il peut contenir les éléments suivants :

- `userName`
- `groupName`
- mot de passe

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential
  xsd:anyAttribute##other>
  <userName> jsdl:StringVariableExpressionType </userName>
  <groupName> jsdl:StringVariableExpressionType </groupName>
  <password> jsdl:StringVariableExpressionType </password>
  <xsd:any##other>*
</credential>
```

élément userName

Définition

Expression de variable de chaîne qui spécifie le nom d'utilisateur d'un utilisateur défini sur le système cible. La commande s'exécute à l'aide de ce nom d'utilisateur. Cet élément est obligatoire si vous utilisez l'élément `credential` et peut être spécifié une fois. Il peut être un ID utilisateur UNIX ou Windows. Pour spécifier le nom d'utilisateur, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références telles que `${var}`, que vous pouvez associer facultativement à tout caractère ou à une chaîne simple. Si l'application s'exécute sur un système Windows en tant qu'utilisateur de domaine Windows, spécifiez le nom d'utilisateur comme suit :

nom_domaine\nom_utilisateur

Si l'application s'exécute en tant qu'utilisateur local, vous pouvez utiliser le format suivant :

nom_utilisateur

Type Le type de cet élément est jsdl:NotEmptyStringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential
  xsd:anyAttribute##other>
  <userName> jsdl:StringVariableExpressionType </userName>
  <groupName> jsdl:StringVariableExpressionType </groupName>
  <password> jsdl:StringVariableExpressionType </password>
  <xsd:any##other>*
</credential>
```

élément groupName

Définition

Expression de variable de chaîne qui spécifie le nom du groupe auquel appartient l'utilisateur, qui est défini sur le système cible sur lequel la commande s'exécute. Cet élément est facultatif et peut être spécifié une fois. Cet élément est ignoré sur les systèmes cible Windows. Pour spécifier le nom de groupe, vous pouvez utiliser une expression de variable qui peut contenir une ou plusieurs références de variable, telle que \${var}, tout caractère et toute chaîne.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential
  xsd:anyAttribute##other>
  <userName> jsdl:StringVariableExpressionType </userName>
  <groupName> jsdl:StringVariableExpressionType </groupName>
  <password> jsdl:StringVariableExpressionType </password>
  <xsd:any##other>*
</credential>
```

élément password

Définition

Expression de variable de chaîne qui définit le mot de passe du nom d'utilisateur spécifié utilisé pour exécuter la commande sur le système cible. Cet élément est facultatif et peut être spécifié une fois. Cet élément est ignoré sur les systèmes cible UNIX. Pour spécifier le mot de passe, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que \${var}, tout caractère et toute chaîne.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential
  xsd:anyAttribute##other>
  <userName> jsdl:StringVariableExpressionType </userName>
  <groupName> jsdl:StringVariableExpressionType </groupName>
  <password> jsdl:StringVariableExpressionType </password>
  <xsd:any##other>*
</credential>
```

élément j2ee

Définition

Cet élément spécifie les informations d'application J2EE nécessaires au travail. Cet élément est facultatif et peut être spécifié une fois. Les opérations J2EE que vous pouvez effectuer varient selon le type de planificateur (direct ou indirect) que vous sélectionnez et selon que vous activez ou non WebSphere Application Server ou la sécurité J2EE.

Type Le type de cet élément est jsdlj:J2EEType. Il peut contenir les éléments suivants :

- invoker
- jsn
- ejb
- credential

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<j2ee>?
  xsd:anyAttribute##other>
  <jsdlj:J2EEType
</j2ee>
```

élément invoker

Définition

Cet élément spécifie si un auteur d'appel direct ou indirect doit être utilisé pour l'application J2EE. Cet élément est obligatoire et peut être spécifié une fois. Sélectionner un auteur d'appel direct signifie que l'agent IBM Workload Scheduler achemine immédiatement le travail aux composants d'instance WebSphere Application Server (EJB ou JMS). Sélectionner un auteur d'appel indirect signifie que l'agent IBM Workload Scheduler optimise une infrastructure de planification WebSphere existante déjà configurée sur le WebSphere Application Server cible.

Type Le type de cet élément est jsdlj:InvokerType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<invoker>?
  xsd:anyAttribute##other>
  <jsdlj:InvokerType
</invoker>
```

élément jms

Définition

Cet élément spécifie la file d'attente Java Message System (JMS) cible et le

message à envoyer. Cet élément est facultatif et peut être spécifié une fois. Cet élément et l'élément **ejb** s'excluent mutuellement.

Type Le type de cet élément est jsdlj:JMSActionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<jms>?
  xsd:anyAttribute##other>
  <jsdlj:JMSActionType
</jms>
```

élément ejb

Définition

Cet élément spécifie les caractéristiques du répertoire de base JNDI de l'EJB à appeler. Cet élément et l'élément **jms** s'excluent mutuellement. L'EJB doit être déjà installé dans le système WAS cible et doit implémenter l'interface TaskHandler.

Type Le type de cet élément est jsdlj:EJBActionType. Il peut contenir les éléments suivants :

- jndiHome
- credential

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<ejb>?
  xsd:anyAttribute##other>
  <jsdlj:EJBActionType
</ejb>
```

élément jndiHome

Définition

Cet élément spécifie le répertoire de base de l'interface de programmation Java Naming and Directory Interface (JNDI). Cet élément est obligatoire et peut être spécifié une fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<jndiHome>?
  xsd:anyAttribute##other>
  <jsdl:StringVariableExpressionType
</jndiHome>
```

Elément d'action JMS

Définition

Cet élément spécifie les caractéristiques de l'action JMS.

Type Le type de cet élément est jsdlj:JMSActionType. Il peut contenir les éléments suivants :

- connFactory
- destination

- message
- credential

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<jms>?
  xsd:anyAttribute##other>
  <jsd1:JMSActionType
<jms>
```

élément connFactory

Définition

Cet élément spécifie un objet administré utilisé par un client pour créer une connexion au fournisseur JMS. Cet élément est obligatoire et peut être spécifié une fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<connFactory>
  xsd:anyAttribute##other>
  <jsd1:StringVariableExpressionType
<connFactory>
```

élément destination

Définition

Cet élément spécifie un objet administré qui encapsule l'identité d'une destination de message, qui est l'emplacement où les messages sont fournis et consommés. Cet élément est obligatoire et peut être spécifié une fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

élément message

Définition

Cet élément spécifie un objet qui est envoyé d'une application à une autre. Cet élément est obligatoire et peut être spécifié une fois.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<message>
  xsd:anyAttribute##other>
  <jsd1:StringVariableExpressionType
<message>
```

élément credential

Définition

Cet élément spécifie les données d'identification requises pour l'exécution

de l'application J2EE. Incluez cet élément lorsque vous voulez spécifier un nom d'utilisateur avec lequel l'application s'exécute sur le système cible, qui est différent du nom d'utilisateur avec lequel l'agent de charge de travail s'exécute. Cet élément est facultatif et peut être spécifié une fois.

Type Le type de cet élément est `jsdl:CredentialType`. Il peut contenir les éléments suivants :

- `userName`
- mot de passe
- `JAASAuthenticationAlias`

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<credential>?
  xsd:anyAttribute##other>
  <jsdl:CredentialType
<credential>
```

élément `userName`

Définition

Cet élément spécifie le nom d'utilisateur d'un utilisateur défini sur le système cible. L'application J2EE s'exécute à l'aide de ce nom d'utilisateur. Cet élément est obligatoire si vous utilisez l'élément `credential` et peut être spécifié une fois. Pour indiquer le nom d'utilisateur, vous pouvez utiliser une expression de variable qui contient une ou plusieurs références telles que `${var}`, que vous pouvez associer facultativement à tout caractère ou à une chaîne simple. Si vous choisissez un auteur d'appel indirect, utilisez cet élément pour spécifier le nom d'utilisateur requis pour vous connecter au planificateur WebSphere Application Server.

Type Le type de cet élément est `jsdl:NotEmptyStringVariableExpressionType`.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<userName>
  xsd:anyAttribute##other>
  <jsdl:NotEmptyStringVariableExpressionType
<userName>
```

élément `password`

Définition

Cet élément spécifie le mot de passe du nom d'utilisateur spécifié utilisé pour exécuter l'application J2EE sur le système cible. Cet élément est facultatif et peut être spécifié une fois. Pour spécifier le mot de passe, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable, telles que `${var}`, tout caractère et toute chaîne. Si vous choisissez un auteur d'appel indirect, utilisez cet élément pour spécifier le mot de passe requis pour vous connecter au planificateur WebSphere Application Server.

Type Le type de cet élément est `jsdl:StringVariableExpressionType`.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<password>  
  xsd:anyAttribute##other>  
<jsd1:StringVariableExpressionType  
<password>
```

élément JAASAuthenticationAlias

Définition

Cet élément spécifie l'alias d'authentification JAAS. Cet élément est facultatif et peut être spécifié une fois. Il est obligatoire uniquement lorsque vous utilisez un auteur d'appel indirect. Pour spécifier l'alias, vous pouvez utiliser une expression de variable pouvant contenir une ou plusieurs références de variable telles que \${var}, que vous pouvez associer facultativement à tout caractère ou à une chaîne simple.

Type Le type de cet élément est jsdl:StringVariableExpressionType.

Attributs

Aucun attribut n'est défini.

Pseudo-schéma

```
<JAASAuthenticationAlias>  
  xsd:anyAttribute##other>  
<jsd1:NotEmptyStringVariableExpressionType  
<JAASAuthenticationAlias>
```

Ressources de la définition de travail

Cette rubrique présente comment les ressources et leurs propriétés sont utilisées dans la définition de travail afin d'identifier les cibles possibles, de réserver les affectations des ressources consommables et d'optimiser l'équilibrage de charge entre les ressources disponibles.

Il est essentiel de comprendre les ressources physiques et logiques ainsi que leurs propriétés pour créer une définition de travail qui cible précisément les ressources adaptées pour l'exécution du travail, détermine l'exigence d'affectation de ressources et contribue à l'équilibrage de charge entre les ressources disponibles. Chaque ressource possède une ou plusieurs propriétés qui lui sont associées. Les propriétés peuvent avoir les caractéristiques suivantes :

Elles sont consommables

Les propriétés des ressources qui sont consommables possèdent une quantité finie qui leur est associée et peut être consommée par les travaux affectés à la ressource. Par exemple, un système informatique possède un nombre fini de processeurs.

Elles peuvent être optimisées

Certaines propriétés peuvent être utilisées pour définir des objectifs d'optimisation, qui déterminent comment la charge doit être équilibrée lorsque les travaux sont affectés à un groupe de ressources. Par exemple, vous pouvez choisir d'affecter un travail aux ressources correspondantes qui ont l'utilisation d'unité centrale la plus faible.

Elles prennent en charge les caractères génériques

Certaines propriétés peuvent être spécifiées dans la définition de travail à l'aide de caractères génériques. Par exemple, une exigence pour une série spécifique de modèles d'ordinateurs peut être définie en spécifiant le modèle à l'aide de caractères génériques.

Le tableau 150 présente les différents types de ressources qui peuvent être incluses dans une définition de travail ainsi que leurs propriétés disponibles.

Tableau 150. Types de ressource et propriétés

Type de ressource	Propriétés disponibles	Elles sont consommables	Elles peuvent être optimisées	Elles prennent en charge les caractères génériques
Système informatique	Utilisation d'unité centrale	Non	Oui	Non
	HostName	Non	Non	Oui
	Fabricant	Non	Non	Oui
	Modèle	Non	Non	Oui
	Nombre de processeurs	Oui	Oui	Non
	Vitesse de traitement	Non	Oui	Non
	Type de processeur	Non	Non	Non
LogicalResource	Nom d'affichage	Non	Non	Oui
	SubType	Non	Non	Oui
	Quantité	Oui	Oui	Non
OperatingSystem	Nom d'affichage	Non	Non	Oui
	Mémoire physique disponible	Non	Oui	Non
	Espace de permutation disponible	Non	Oui	Non
	Mémoire virtuelle disponible	Non	Oui	Non
	Type de système d'exploitation	Non	Non	Non
	OperatingSystem Version	Non	Non	Non
	Mémoire physique totale	Oui	Oui	Non
	Espace de permutation total	Oui	Oui	Non
	Mémoire virtuelle totale	Oui	Oui	Non
FileSystem	Nom d'affichage	Non	Non	Oui
	Racine du système de fichiers	Non	Non	Oui
	Type de système de fichiers	Non	Non	Non
	Mémoire disponible	Non	Oui	Non
	Mémoire totale	Oui	Oui	Non
Système réseau	Adresse réseau	Non	Non	Non
	Système réseau HostName	Non	Non	Oui

Annexe C. Aide-mémoire pour les commandes

Cette annexe est composée de quatre sections :

- «Gestion du plan»
- «Gestion des objets dans la base de données», à la page 938
- «Gestion des objets dans le plan», à la page 948
- «Commandes d'utilitaire», à la page 954
- «Commandes de génération d'états», à la page 957

Gestion du plan

Cette section décrit les opérations que vous pouvez effectuer sur le plan à l'aide du script **JnextPlan** et de la ligne de commande **planman** :

Tableau 151. Commandes utilisées sur le plan

Syntaxe de la commande ou du script	Action effectuée
JnextPlan [-from mm/jj/aa]aa[hh[:]mm[tz <i>timezoned</i> nom_fh]] {-to mm/jj/aa]aa[hh[:]mm[tz <i>fuseau horaire</i> nom_fh]] -for [h]hh[:]mm [-days n] -days n}	Crée ou étend le plan de production.
planman [paramètres_connexion] crt [-from mm/jj/aa]aa [hh[:]mm [tz <i>fuseau horaire</i> nom_fh]] {-to mm/jj/aa]aa[hh[:]mm[tz <i>fuseau horaire</i> nom_fh]] -for [h]hh[:]mm [-days n] -days n}	Crée un plan de production intermédiaire.
planman [paramètres_connexion] deploy [-scratch]	Déploie toutes les règles ne se trouvant pas à l'état de brouillon.
planman [paramètres_connexion] ext {-to mm/jj/aa]aa[hh[:]mm[tz <i>fuseau horaire</i> nom_fh]] -for [h]hh[:]mm [-days n] -days n}	Crée un plan intermédiaire pour une extension de plan.
planman [paramètres_connexion] showinfo	Extrait les informations du plan de production.
planman [paramètres_connexion] crttrial nom_fichier [-from mm/jj/aa]aa [hh[:]mm [tz <i>fuseau horaire</i> nom_fh]] {-to mm/jj/aa]aa[hh[:]mm[tz <i>fuseau horaire</i> nom_fh]] -for [h]hh[:]mm [-days n] -days n}	Crée un plan d'essai.

Tableau 151. Commandes utilisées sur le plan (suite)

Syntaxe de la commande ou du script	Action effectuée
planman [<i>paramètres_connexion</i>] exttrial <i>nom_fichier</i> {-to <i>mm/jj/aa</i> [<i>hh</i> [:] <i>mm</i> [<i>tz</i> <i>fuseau_horaire</i> <i>nom_fh</i>]] -for [<i>h</i>] <i>hh</i> [:] <i>mm</i> [-days <i>n</i>] -days <i>n</i> }	Crée un plan d'essai avec une extension du plan de production.
planman [<i>paramètres_connexion</i>] crtfc <i>nom_fichier</i> [-from <i>mm/jj/aa</i> [<i>hhmm</i> [<i>tz</i> <i>timezone</i> <i>fuseau_horaire</i>]]] {-to <i>mm/jj/aa</i> [<i>hh</i> [:] <i>mm</i> [<i>tz</i> <i>fuseau_horaire</i> <i>nom_fh</i>]] -for [<i>h</i>] <i>hh</i> [:] <i>mm</i> [-days <i>n</i>] -days <i>n</i> }	Crée un plan prévisionnel.
planman [<i>paramètres_connexion</i>] unlock	Déverrouille le plan de production.
ResetPlan [<i>paramètres_connexion</i>] [-scratch]	Réinitialise le plan de production.
planman reset -scratch	Supprime le plan de préproduction durant la maintenance du fichier Symphony.
planman [<i>paramètres_connexion</i>] resync	Réplique les données de plan entre le fichier Symphony et la base de données.
planman [<i>paramètres_connexion</i>] checksync	Surveille la progression et l'issue du processus de réplication du plan de données dans la base de données.

où les valeurs de *paramètres_connexion* sont les suivantes :

```
[-file nom_fichier]
[-host nom_hôte]
[-port nom_port]
[-protocol nom_protocole] [-proxy nom_proxy]
[-proxyport numéro_port_proxy]
[-password mot_de_passe_utilisateur]
[-timeout secondes]
[-username nom_utilisateur]
```

Pour plus d'informations, voir «Création et extension d'un plan de production», à la page 91.

Gestion des objets dans la base de données

La présente section comprend les sous-sections suivantes :

- «Commandes universelles», à la page 939
- «Objets de planification», à la page 939
- «Commandes composer», à la page 945

Commandes universelles

Cette section décrit le nom et la syntaxe des commandes universelles exécutées à partir du programme **composer**, ainsi que l'autorisation utilisateur requise, le cas échéant, pour les exécuter.

Tableau 152. Commandes universelles

Commande	Syntaxe	Autorisation utilisateur
continue	continue &argument_de_commande&argument_de_commande	Autorisation d'utilisation de composer
edit	edit nom_fichier	Autorisation d'utilisation de composer
exit	exit	Autorisation d'utilisation de composer
aide	help nom_commande	Autorisation d'utilisation de composer
redo	redo instructions	Autorisation d'utilisation de composer
validate	validate nom_fichier [;syntax]	Autorisation d'utilisation de composer
version	version	Autorisation d'utilisation de composer

Objets de planification

La présente section contient la syntaxe de définition de tous les objets de planification.

Dans le tableau qui affiche la liste des commandes que vous pouvez exécuter sur l'objet de planification, la variable *nom_fichier* représente un fichier existant lorsqu'elle est utilisée dans la syntaxe des commandes **add** et **replace**, mais elle représente un fichier non existant lorsqu'elle est utilisée dans la syntaxe de la commande **create/extract**.

Agenda

Syntaxe de définition du fichier :

```
$calendar
nom_agenda ["description"]
date [...]
```

Domaine, zone

Syntaxe de définition du fichier :

```

/
/      domain nom_domaine[description "description"]
/      * manager poste_de_travail
/      [parent nom_domaine | ismaster]
/      end

```

Règle d'événement

Syntaxe de définition XML :

```

/
/      • eventRule name=" " ruleType=" " isDraft=" " (1, 1)
/      - description (0, 1)
/      - timeZone (0, 1)
/      - validity from=" " to=" " (0, 1)
/      - activeTime start=" " end=" " (0, 1)
/      - timeInterval amount=" " unit=" " (0, 1)
/      - eventCondition eventProvider=" " eventType=" " (1, n)
/      - scope (0, 1)
/      - filteringPredicate (0, 1)
/      • attributeFilter name=" " operator="eq" (0, n)
/      - value (1, n)
/      • attributeFilter name=" " operator="ne" (0, n)
/      - value (1, n)
/      • attributeFilter name=" " operator="le" (0, n)
/      - value (1, 1)
/      • attributeFilter name=" " operator="ge" (0, n)
/      - value (1, 1)
/      • attributeFilter name=" " operator="range" (0, 1)
/      - value (1, 2)
/      - correlationAttributes (0, 1)
/      - attribute name=" " (1, n)
/      - action actionProvider=" " actionType=" " responseType=" " (0, n)
/      - description (0, 1)
/      - scope (0, 1)
/      - parameter name=" "(1, n)
/      - value (1, 1)

```

Travail

Syntaxe de définition du fichier :

```

/
/      $jobs
/      [poste_de_travail#]nom_travail
/      {scriptname nom_fichier streamlogon nom_utilisateur |
/      docommand "commande" streamlogon nom_utilisateur |
/      task définition_travail }
/      [description "description"]
/      [tasktype type_tâche]
/      [interactive]
/
/      [succoutputcond nom_condition "valeur_condition"]
/      [outputcond nom_Condition "valeur_Condition"]
/
/      [recovery
/      {stop
/      [after [poste_de_travail#]nom_travail]
/      [abendprompt "texte"]}
/      |continue
/      [after [poste_de_travail#]nom_travail]
/      [abendprompt "texte"]] |rerun [same_workstation]
=

```

```
= [[repeatevery hmmm] [for nombre attempts]]
/ [after [poste_de_travail#]nom_travail]
/ | [after [poste_de_travail#]nom_travail]
/ [abendprompt "texte"]
```

/ Remarque :

/ 1. Ce mot clé est uniquement disponible sur les plates-formes Windows.

/ Flot de travaux

/ Syntaxe de définition du fichier :

```
/ schedule [poste_de_travail#][folder/]nom_flot_travaux
/ # commentaire
/ [validfrom date]
/ [timezone|tz fuseau_horaire]
/ [description "texte"]
/ [draft]
/ [variable nom_table]
/ [freedays nom_agenda [-sa] [-su]]
/ [on [runcycle nom
/ [validfrom date] [validto date]
/ [description "texte"]
/ [variable nom_table]]
/ {date|jour|agenda|demande|"icalendar"|groupe_cycle_execution} [...]
/ [fdignore|fdnext|fdprev]
/ [{(at heure [+n day[s]] |
/ schedtime heure [+n day[s]]}
/ [until | jsuntil heure [+n day[s]] [onuntil action]]
= [every fréquence {everyendtime heure[+n day[s]]}
/ [deadline heure [+n day[s]]]]
/ [...]
/ startcond filecreated | filemodified nom_poste_de_travail#nom_fichier user nom_utilisateur interval
/ secondes [(alias nom_travail_condition_démarrage rerun batch outfile nom_fichier_sortie
/ params "paramètres filemonitor supplémentaires")] |
= startcond job nom_poste_travail#nom_travail outcond condition_sortie_travail interval
= secondes [(alias nom_travail_condition_démarrage rerun)]
/ :
/ [except [runcycle nom]
/ [validfrom date] [validto date]
/ [description "texte"]
/ {date|jour|agenda|demande|"icalendar"|groupe_cycle_execution} [...]
/ [fdignore|fdnext|fdprev]
/ [{(at heure [+n day[s]] |
/ (schedtime heure [+n day[s]]))}
/ [...]
/ [{at heure [timezone|tz fuseau_horaire] [+n day[s]] |
/ schedtime heure [timezone|tz fuseau_horaire] [+n day[s]]}
= [until | jsuntil heure [timezone|tz fuseau_horaire] [+n day[s]] [onuntil action]]
/ [deadline heure [timezone|tz fuseau_horaire] [+n day[s]]]
/ [carryforward]
/ [matching {previous|sameday|relative from [+ | -] heure to [+ | -] heure |
/ from heure [+ | -n day[s]] to heure [+ n day[s]] [...]}]
/ [follows {[netagent::][poste_de_travail#][folder/]nom_flot_travaux[.nom_travail |
/ @] [previous|
/ sameday|relative from [+|-] heure to [+|-] heure |
/ from heure [+|-n day[s]] to heure [+|-n day[s]]
```

```

/          ][if <condition> [ | <condition>...]]
/          } ] [,...]] [...]
/ [join nom_condition [nombre | nombre_conditions | all] of
/ description "..."]
/
/      ....
/      endjoin
/      [keysched]
/      [limit nombre_max_travaux]
/      [needs { [n] [poste_de_travail#]nom_ressource } [,...] ] [...]
/      [opens { [poste_travail#]"nom_fichier" [ (qualificatif) ] [,...] }} [...]
/      [priority nombre | hi | go]
/      [prompt {nom_invite|":!|texte"} [,...] ] [...]
/      [onoverlap {parallel|enqueue|donotstart}]
/
/      :
/      instruction-travail
/      # commentaire
/      nom_travail [alias_travail]
/      [outcond condition_sortie_travail interval secondes]
/      [[at heure [timezone|tz fuseau_horaire] [+n day[s]] |
/      schedtime heure [timezone|tz fuseau_horaire] [+n jour[s]]][,...]
/      [until heure [timezone|tz fuseau_horaire] [+n day[s]] [onuntil action]
/      [deadline heure [timezone|tz fuseau_horaire] [+n day[s]] [onlate action] ]
=      [maxdur heure | pourcentage % onmaxdur action]
/      [mindur heure | pourcentage % onmindur action]
/      [every fréquence]
/      [follows {[agent_reseau::][poste_de_travail#]nom_flot_travaux{.nom_travail @} [previous|
/      sameday relative from [+|-] heure to [+|-] heure|
/      from heure [+|-n day[s]] to heure [+|-n day[s]]
/      } } ][if <condition> [ | <condition>...]] [,...] ] [...]
/      [join nom_condition [nombre | nombre_conditions | all] of
/      description "..."]
/      ....
/      endjoin
/      [confirmed]
/      [critical]
/      [keyjob]
/      [needs { [n] [poste_de_travail#]nom_ressource } [,...] ] [...]
/      [opens { [poste_travail#]"nom_fichier" [ (qualificatif) ] [,...] }} [...]
/      [priority nombre | hi | go]
/      [prompt {nom_invite|":!|texte"} [,...] ] [...]
/      [nop]
/      [statistictype custom]
/
/      [instruction-travail...]
/      end

```

Paramètre

Syntaxe de définition du fichier :

```

/      $parm
/      [nom_table.]nom_variable "valeur_variable"

```

Invite

Syntaxe de définition du fichier :

/

/

/

/

/

/

/

/

/

/

/

/

/

/

/

/

/

/

/

=

1

/

/

/

/

/

/

/

/

/

/

/

/

/

•

1

1

/

/

/

/

/

/

/

/

1

/

/

Pour plus d'informations, voir Chapitre 6, «Personnalisation de votre charge de travail à l'aide des tables de variables», à la page 129.

Poste de travail

Syntaxe de définition du fichier :

```

cpuname poste_de_travail [description "description"]
[vartable nom_table]
os type_sys_expl
[node nom_hôte] [tcpaddr port]
[secureaddr port][timezone | tz fuseau_horaire]
[domain nom_domaine]
[for maestro [host poste_de_travail_hôte [access méthode]]
[type fta | s-agent | x-agent | manager | broker | agent |
pool | d-pool | rem-engine]
[ignore]
[autolink on | off]
[behindfirewall on | off]
[securitylevel enabled | on | force]
[fullstatus on | off]
[server id_serveur]
[protocole http | https]
[membres [poste_travail] [...]]
[exigences définition_jsdl]
end

```

```

cpuname poste_de_travail [description description]
[vartable nom_table]
os type_sys_expl
node nom_hôte [tcpaddr port]
[secureaddr port][timezone | tz fuseau_horaire]
[domain nom_domaine]
[for maestro [host poste_de_travail_hôte [access méthode]]
[type fta | s-agent | x-agent | manager]
[ignore]
[autolink on | off]
[behindfirewall on | off]
[securitylevel enabled | on | force]
[fullstatus on | off]
[server id_serveur]
end

```

Classe de postes de travail

Syntaxe de définition du fichier :

```

cpuclass classe_poste_de_travail [description "description"]
[ignore]
members [poste_de_travail | @] [...]
end

```

Définition d'utilisateur

Syntaxe de définition du fichier :

```

nom_utilisateur[poste_de_travail#][domaine\]nom_utilisateur
mot_de_passe "mot_de_passe" end

```

Commandes composer

Cette section décrit les opérations que vous pouvez effectuer dans la base de données à l'aide du programme d'interface de ligne de commande **composer** en utilisant la syntaxe suivante :

```
composer [paramètres connexion] [-defaultws uc_tws]  
["commande [&[commande]] [...]"
```

où *paramètres connexion*, s'ils ne sont pas fournis dans les fichiers localopts ou useropts, sont les suivants :

```
[-file nom_fichier] |  
[-host nom_hôte]  
[-port nom_port]  
[-protocol nom_protocole]  
[-proxy nom_proxy]  
[-proxyport numéro_port_proxy]  
[-password mot_de_passe_utilisateur]  
[-timeout secondes]  
[-username nom_utilisateur]
```

Pour plus d'informations, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

Ces opérations peuvent uniquement être exécutées à partir de toute ligne de commande de client composer installée.

Dans le tableau 153 qui affiche la liste des commandes que vous pouvez exécuter sur l'objet de planification, la variable *nom_fichier* représente un fichier existant lorsqu'elle est utilisée dans la syntaxe des commandes **add** et **replace**, mais elle représente un fichier non existant lorsqu'elle est utilisée dans la syntaxe de la commande **create/extract**.

Tableau 153. Commandes composer

Commande	Syntaxe	Autorisation utilisateur
add	{add a} <i>nom_fichier</i> [:unlock]	add ou modify
authenticate	{authenticate au} [username= <i>nom_utilisateur</i> password= <i>mot_de_passe</i>]	
continuer	{continue c}	
create extract	{create cr extract ext} <i>nom_fichier</i> from {[calendars calendar cal= <i>nom_agenda</i>] [eventrule erule er= <i>nom_règle_événement</i>] [parms parm vb=[<i>nom_table</i> .] <i>nom_variable</i>] [vartable vt= <i>nom_table</i>] [prompts prom= <i>nom_invite</i>] [resources resource res=[<i>poste_de_travail</i> #] <i>nom_ressource</i>] [runcyclegroup rcg= <i>nom_groupe_cycle_exécution</i>] [cpu=[<i>nom_poste_travail</i> classe_poste_de_travail <i>nom_domaine</i>]} [workstation ws= <i>nom_poste_travail</i>] [workstationclass wscl= <i>nom_classe_poste_de_travail</i>] [domain dom= <i>nom_domaine</i>] [jobs jobdefinition jd=[<i>nom_poste_travail</i> #] <i>nom_travail</i>] [sched jobstream js= [<i>nom_poste_travail</i> #] <i>nom_flot_travaux</i> [valid from date \valid to date \valid in date date] [:full]] [users user=[<i>nom_poste_travail</i> #] <i>nom_utilisateur</i> [:password]] [accesscontrollist acl for <i>nom_domaine_sécurité</i>] [securitydomain sdom= <i>nom_domaine_sécurité</i>] [securityrole srol= <i>nom_rôle_sécurité</i>]} [:lock]	display

Tableau 153. Commandes composer (suite)

Commande	Syntaxe	Autorisation utilisateur
delete	<pre>{delete de} {[calendars calendar cal=nom_agenda] [domain dom=nom_domaine] [eventrule erule er=nom_règle_événement] [parms parm vb=[nom_table.]nom_variable] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [table vt=nom_table] [wat=workloadapplicationtemplatename] [cpu={workstationname [:force] workstationclassname [:force] domainname}] [workstation ws=workstationname] [:force] [workstationclass wscl=workstationclassname] [:force] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [début de validité date fin de validité date valid in date date]] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité]} [:noask]</pre>	supprimer
display	<pre>{display di} {[calendars calendar cal=nom_agenda] [eventrule erule er=nom_règle_événement] [parms parm vb=nom_variable.]nom_variable] [table vt=nom_table] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [cpu={nom_poste_de_travail nom_classe_poste_de_travail nom_domaine}] [wat=nom_modèle_application_charge_travail] [workstation ws=nom_poste_travail] [workstationclass wscl=nom_classe_poste_de_travail] [domain dom=nom_domaine] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [valid from date valid to date valid in date date] [:full]] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité]} [:offline]</pre>	display
edit	{edit ed} nom_fichier	
exit	{exit e}	

Tableau 153. Commandes composer (suite)

Commande	Syntaxe	Autorisation utilisateur
list print	<pre>{list l} {[calendars calendar cal=nom_agenda] [eventrule erule er=nom_règle_événmt] [parms parm vb=[nom_table.]nom_variable] [vartable vt=nom_table] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [cpu={nom_poste_travail nom_classe_poste_de_travail nom_domaine}] [wat=nom_modèle_application_charge_travail] [workstation ws=nom_poste_travail] [workstationclass wscl=nom_classe_poste_de_travail] [domain dom=nom_domaine] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [valid from date valid to date valid in date date] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité]} [;offline]</pre>	display
lock	<pre>{lock lo} {[calendars calendar cal=nom_agenda] [eventrule erule er=nom_règle_événmt] [parms parm vb=[nom_table.]nom_variable] [vartable vt=nom_table] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [cpu={nom_poste_travail nom_classe_poste_de_travail nom_domaine}] [workstation ws=nom_poste_travail] [workstationclass wscl=nom_classe_poste_de_travail] [domain dom=nom_domaine] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [début de validité date fin de validité date valid in date date]] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité]}</pre>	modify
modify	<pre>{modify m} {[calendars calendar cal=nom_agenda] [eventrule erule er=nom_règle_événmt] [parms parm vb=[nom_table.]nom_variable] [vartable vt=nom_table] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [cpu={nom_poste_travail nom_classe_poste_de_travail nom_domaine}] [wat=nom_modèle_application_charge_travail] [workstation ws=nom_poste_travail] [workstationclass wscl=nom_classe_poste_de_travail] [domain dom=nom_domaine] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [valid from date valid to date valid in date date] [;full]] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité]}</pre>	modify ou add

Tableau 153. Commandes composer (suite)

Commande	Syntaxe	Autorisation utilisateur
new	new [calendar domain eventrule job jobstream parameter prompt resource runcyclegroup user variable wat workstation workstationclass accesscontrollist securitydomain securityrole]	add ou modify
rename	{rename rn} {calendars calendar cal parms parm vb variable vt prompts prom resources resource res runcyclegroup rcg workstation ws workstationclass wscl domain dom jobs jobdefinition jd jobsched jb eventrule erule er sched jobstream js users user } ancien_identificateur_objet nouvel_identificateur_objet	add et delete
replace	{replace rep} nom_fichier [:unlock]	modify ou add
unlock	{unlock u} {[calendars calendar cal=nom_agenda] [eventrule erule er=nom_règle_événement] [parms parm vb=[nom_table.]nom_variable] [variable vt=nom_table] [prompts prom=nom_invite] [resources resource res=[poste_de_travail#]nom_ressource] [runcyclegroup rcg=nom_groupe_cycle_exécution] [cpu={nom_poste_travail nom_classe_poste_de_travail nom_domaine}] [workstation ws=nom_poste_travail] [workstationclass wscl=nom_classe_poste_de_travail] [domain dom=nom_domaine] [jobs jobdefinition jd=[nom_poste_travail#]nom_travail] [sched jobstream js= [nom_poste_travail#]nom_flot_travaux [début de validité date fin de validité date valid in date date]] [users user=[nom_poste_travail#]nom_utilisateur] [accesscontrollist acl for nom_domaine_sécurité] [securitydomain sdom=nom_domaine_sécurité] [securityrole srol=nom_rôle_sécurité] [:forced]	modify et unlock
validate	{validate val} nom_fichier [:syntax]	

Gestion des objets dans le plan

Cette section décrit les opérations que vous pouvez effectuer sur le plan à l'aide du programme d'interface de ligne de commande **conman** en utilisant la syntaxe suivante :

```
conman ["command[&[command]...] [&"]
```

Commandes conman

La présente section répertorie les commandes que vous pouvez exécuter à partir du programme **conman**.

Voici comment accéder à la ligne de commande **conman** :

```
conman [paramètres_connexion] ["commande[&[commande]...] [&"]
```

où *paramètres_connexion*, s'ils ne sont pas fournis dans les fichiers *localopts* ou *useropts*, sont les suivants :

```
[-file nom_fichier]
[-host nom_hôte]
[-port nom_port]
[-protocol nom_protocole]
[-proxy nom_proxy]
[-proxyport numéro_port_proxy]
[-password mot_de_passe_utilisateur]
[-timeout secondes]
[-username nom_utilisateur]
```

Pour plus de détails, voir «Configuration des options pour l'utilisation des interfaces utilisateur», à la page 65.

Voici comment sélectionner des travaux dans les commandes :

```
[poste_de_travail#]
{nom_flot_travaux(hhmm[date]) travail|numéro_travail}
[{|~}qualifiant_travail[...]]
```

ou :

```
[poste_de_travail#]
ID_flot_travaux.
travail
[{|~}qualifiant_travail[...]]
;schedid
```

Voici comment sélectionner des flots de travaux dans les commandes :

```
[poste_de_travail#]
nom_flot_travaux(hhmm[date])
[{|~}nom_flot_travaux[...]]
```

ou :

```
[poste_de_travail#]
ID_flot_travaux
;schedid
```

Vous pouvez exécuter ces commandes à partir de différents types de poste de travail. Dans ce tableau :

- F** représente les gestionnaires de domaine et les agents tolérants aux pannes.
- S** représente les agents standard.

Vous trouverez le nom et la syntaxe de chaque commande, ainsi que les types de poste de travail à partir desquels vous pouvez l'émettre et, le cas échéant, l'autorisation utilisateur requise.

Tableau 154. Commandes pouvant être exécutées à partir de conman

Commande	Syntaxe	Types de poste de travail	Autorisation utilisateur
adddep job	{adddep job adj} travail_sélectionné ;dépendance[;...] [;noask]	F	adddep - (à utiliser lorsque les mots clés prompts et needs sont utilisés)
adddep sched	{adddep sched ads} flot_travaux_sélectionné ;dépendance[;...] [;noask]	F	adddep - (à utiliser lorsque les mots clés prompts et needs sont utilisés)
altpass	altpass [poste_de_travail#] nom_utilisateur ["mot_de_passe"]	F	altpass
altpri	{altpri ap} travail_sélectionné flot_travaux_sélectionné [;pri] [;noask]	F	altpri
bulk_discovery	{bulk_discovery bulk}	F	display
cancel job	{cancel job cj} travail_sélectionné [;pend] [;noask]	F	cancel
cancel sched	{cancel sched cs} flot_travaux_sélectionné [;pend] [;noask]	F	cancel
checkhealthstatus	{checkhealthstatus chs} [poste_de_travail]	M,F,S	
confirm	{confirm conf} travail_sélectionné ;succ abend} [;IF 'nom_condition_sortie[, nom_condition_sortie][, ...]'] [;noask]	F	confirm
console	{console cons} [sess sys] [;level=niveau_message]	F-S	console
continuer	{continue cont}	F-S	
deldep job	{deldep job ddj} jobselect ;dépendance[;...] [;noask]	F	deldep
deldep sched	{deldep sched dds} jstreamselect ;dépendance[;...] [;noask]	F	deldep
deployconf	{deployconf deploy} [domain!]poste de travail	F, S	Droit de démarrer (start) des actions sur des objets cpu
display	{display file df} nom_fichier [;offline] {display job dj} jobselect [;offline] {display sched ds} jstreamselect [valid {at date in date date}] [;offline]	F-S ¹	display

Tableau 154. Commandes pouvant être exécutées à partir de conman (suite)

Commande	Syntaxe	Types de poste de travail	Autorisation utilisateur
exit	{ exit e }	F-S	
fence	{ fence f } <i>poste_de_travail</i> ; <i>pri</i> [; noask]	F	<i>fence</i>
help (UNIX uniquement)	{ help h } { <i>commande</i> <i>mot_clé</i> }	F-S	
kill	{ kill k } <i>jobselect</i> [; noask]	F	<i>kill</i>
limit cpu	{ limit cpu lc } <i>poste de travail</i> ; <i>limite</i> [; noask]	F	<i>limit</i>
limit sched	{ limit sched ls } <i>flot_travaux_sélectionné</i> ; <i>limite</i> [; noask]	F	<i>limit</i>
link	{ link lk } [<i>domaine!</i>] <i>poste de travail</i> [; noask]	F-S	<i>link</i>
listsym	{ listsym lis } [trial forecast] [; offline]	F	
recall	{ recall rc } [<i>poste de travail</i>] [; offline]	F	<i>display</i>
redo	{ redo red }	F-S	
release job	{ release job rj } <i>travail_sélectionné</i> [; <i>dépendance</i> [;...]] [; noask]	F	<i>release</i>
release sched	{ release sched rs } <i>flot_travaux_sélectionnés</i> [; <i>dépendance</i> [;...]] [; noask]	F	<i>release</i>
reply	{ reply rep } { <i>nom_invite</i> [<i>poste_de_travail</i> #] <i>num_message</i> } ; <i>reply</i> [; noask]	F	<i>reply</i>
rerun	{ rerun rr } <i>travail_sélectionné</i> [; from = <i>[poste_de_travail#]travail</i>] [; at = <i>heure</i>] [; pri = <i>pri</i>] [; sameworkstation =] [; step = <i>étape</i>] [; noask]	F	<i>rerun</i>
resource	{ resource reso } [<i>poste_de_travail</i> #] <i>ressource</i> ; <i>num</i> [; noask]	F	<i>resource</i>
setsym	{ setsym set } [essai prévision] [<i>num_fichier</i>]	F	

Tableau 154. Commandes pouvant être exécutées à partir de conman (suite)

Commande	Syntaxe	Types de poste de travail	Autorisation utilisateur
showcpus	{showcpus sc} [[domaine!]poste_de_travail] [;info ;link] [;offline]	F-S	list ²
showdomain	{showdomain showdom sd} [domaine] [;info] [;offline]	F-S	list ²
showfiles	{showfiles sf} [[num_poste_de_travail]fichier] [;état[;...]] [;keys] [;offline] {showfiles sf} [[poste de travail#]fichier] [;état[;...]] [;deps[;keys info login]] [;offline]	F	
showjobs	{showjobs sj} [travail_sélectionné] [;deps[;keys info login]] [;short single] [;offline] [;showid] [;props] {showjobs sj} [travail_sélectionné [poste_de_travail#]jobnumber.hhmm] [;stdlist[;keys]] [;short single] [;offline] [;showid] [;props]	F	list ²
showprompts	{showprompts sp} [invite_sélectionnée] [;keys] [;offline] {showprompts sp} [invite_sélectionnée] [;deps[;keys info login]][;offline]	F	list ²
showresources	{showresources sr} [[poste de travail#]nom_ressource] [;keys] [;offline] {showresources sr} [[poste de travail#]nom_ressource] [;deps[;keys info login]] [;offline]	F	list ²
showschedules	{showscheds ss} [flot_travaux_sélectionné] [;keys] [;offline] [;showid] {showscheds ss} [flot_travaux_sélectionné] [;deps[;keys info login]] [;offline] [;showid]	F	list ²

Tableau 154. Commandes pouvant être exécutées à partir de conman (suite)

Commande	Syntaxe	Types de poste de travail	Autorisation utilisateur
shutdown	{shutdown shut} [;wait]	F-S	shutdown
start	start [domaine!]poste de travail [;mgr] [;noask] [;demgr]	F-S	start
startappserver	startappserver [domaine!]poste de travail [;wait]	F-S	Droit de démarrer (start) des actions sur des objets cpu
startevtp	{starteventprocessor startevtp} [domaine!]poste de travail	M ⁴	Droit de démarrer (start) des actions sur des objets cpu
startmon	{startmon startm} [domaine!]poste de travail [;noask]	F-S	Droit de démarrer (start) des actions sur des objets cpu
status	{status stat}	F-S	appserver
stop	stop [domaine!]poste de travail [;wait] [;noask]	F-S	stop
stop ;progressive	stop ;progressive		stop
stopappserver	{stopappserver stopapps} [domaine!]poste de travail [;wait]	F-S	Droit d'arrêter (stop) des actions sur des objets cpu
stopevtp	{stopeventprocessor stopevtp} [domaine!][poste_de_travail]	M ⁴	Droit d'arrêter (stop) des actions sur des objets cpu
stopmon	{stopmon stopm} [domaine!]poste de travail [;wait] [;noask]	F-S	Droit d'arrêter (stop) des actions sur des objets cpu
submit docommand	{submit docommand sbd} [poste de travail#"cmd" [;alias[=nom]] [;into=[poste de travail#] {ID_flot_travaux;schedid nom_flot_travaux ([hhmm[date]])}] [;option_travail[;...]]	F-S	submit - (à utiliser lorsque les mots clés prompts et needs sont utilisés)
submit file	{submit file sbf} "nom_fichier" [;alias[=nom]] [;into=[poste_de_travail#]{ID_flot_travaux ;schedid nom_flot_travaux([hhmm[date]])}] [;option_travail[;...]] [;noask]	F-S	submit - (à utiliser lorsque les mots clés prompts et needs sont utilisés)
submit job	{submit job sbj} [poste de travail#]nom_travail [;alias[=nom]] [;into=[poste_de_travail#]{ID_flot_travaux ;schedid nom_flot_travaux([hhmm[date]])}] [;option_travail[;...]] [;varable=nom_table] [;noask]	F-S ³	submit - (à utiliser lorsque les mots clés prompts et needs sont utilisés)

Tableau 154. Commandes pouvant être exécutées à partir de *conman* (suite)

Commande	Syntaxe	Types de poste de travail	Autorisation utilisateur
submit sched	{ submit sched sbs } [poste de travail#]nom_flot_travaux [;alias[=nom]] [;option_flot_travaux[;...]] [;variable=nom_table] [;noask]	F-S ³	submit - (à utiliser lorsque les mots clés prompts et needs sont utilisés)
switchevt	{ switcheventprocessor switchevt } poste de travail	M ⁴	Droit de démarrer et arrêter des actions sur des objets <i>cpu</i>
switchmgr	{ switchmgr switchm } domaine;nouv_gest	F	start stop
system	[: !] commande-système	F-S	
tellop	{ tellop to } [texte]	F-S	
unlink	unlink [domaine!]poste de travail [;noask]	F-S	unlink
version	{ version v }	F-S	

où :

- (1) Indique que vous pouvez uniquement afficher des fichiers sur un agent standard.
- (2) Vous devez avoir un accès **list** à l'objet qui est affiché si l'option *enListSecChk* a été définie sur **yes** sur le gestionnaire de domaine maître lorsque le plan de production a été créé ou étendu.
- (3) Indique que vous pouvez utiliser les commandes submit job (**sbj**) et submit sched (**sbs**) sur un agent standard en utilisant les paramètres de connexion ou en spécifiant les paramètres dans le fichier *useropts* lors de l'appel de la ligne de commande **conman**.
- (4) Vous pouvez exécuter cette commande sur des gestionnaires de domaine maîtres et maîtres de sauvegarde, ainsi que sur des postes de travail installés comme maîtres de sauvegarde, mais utilisés comme des agents tolérants aux pannes ordinaires.

Commandes d'utilitaire

Cette section contient la liste de commandes d'utilitaire que vous pouvez exécuter à partir de l'invite de commande du système d'exploitation. Les commandes d'utilitaire sont divisées en trois groupes : celles que vous pouvez exécuter à la fois sur les systèmes d'exploitation UNIX et Windows, celles que vous ne pouvez exécuter que sur UNIX, et celles que vous ne pouvez exécuter que sur Windows.

Commandes d'utilitaire disponibles à la fois pour les systèmes d'exploitation UNIX et Windows

Tableau 155. Commandes d'utilitaire disponibles à la fois pour UNIX et Windows

Commande	Syntaxe
cpuinfo	cpuinfo -V -U cpuinfo poste_de_travail [type_info] [...]

Tableau 155. Commandes d'utilitaire disponibles à la fois pour UNIX et Windows (suite)

Commande	Syntaxe
datecalc	datecalc -V -U datecalc <i>date_base</i> [<i>décalage</i>] [<i>pic format</i>][<i>freedays nom_agenda</i> [-sa] [-su]] datecalc -t <i>heure</i> [<i>date-base</i>] [<i>décalage</i>] [<i>pic format</i>] datecalc <i>aaaammijhhmm</i> [<i>décalage</i>] [<i>pic format</i>]
delete	delete -V -U delete <i>nom_fichier</i>
evtdef	evtdef -U -V evtdef [<i>paramètres de connexion</i>] dumpdef <i>chemin d'accès au fichier</i> evtdef [<i>paramètres de connexion</i>] loaddef <i>chemin d'accès au fichier</i>
evtsize	evtsize -V -U evtsize <i>nom_fichier</i> <i>taille</i> evtsize -compact <i>nom_fichier</i> [<i>taille</i>] evtsize -info <i>nom_fichier</i> evtsize -show <i>nom_fichier</i> evtsize -info -show pobox
filemonitor	filemonitor -V -U filemonitor -path <i>chemin_à_surveiller</i> -event <i>événement_à_surveiller</i> {fileCreated fileModified [-modificationCompletedTime <i>secondes</i>]} [-repositoryName <i>nom_référentiel</i>] [-repositoryPath <i>chemin_référentiel</i>] [-recursive] [-outputFile <i>nom_fichier_sortie</i>] [-scanInterval <i>intervalle_analyse</i>] [-maxEventsThreshold <i>nbre_max_événements</i>] [-minFileSize <i>taille_min_fichier</i>] [-timeout <i>secondes</i>] filemonitor -reset
jobinfo	jobinfo -V -U jobinfo <i>option-travail</i> [...]
jobstdl	jobstdl -V -U jobstdl [-day <i>num</i>] [{-first -last -num <i>n</i> -all}] [-tw slog] [{-name [" <i>nom_flot_travaux</i> [(<i>hhmm date</i>),(<i>ID_flot_travaux</i>)]. <i>nom_travail</i> " <i>num_travail</i> -schedid <i>ID_flot_travaux.nom_travail</i>]}]
maestro	maestro [-V -U]

Tableau 155. Commandes d'utilitaire disponibles à la fois pour UNIX et Windows (suite)

Commande	Syntaxe
makecal	makecal [-c <i>nom</i>] -d <i>n</i> -e {-f 1 2 3 -s <i>date</i> } -l -m -p <i>n</i> {-r <i>n</i> -s <i>date</i> } -w <i>n</i> [-i <i>n</i>] [-x -z][-freedays <i>nom_calendrier</i> [-sa] [-su]]
morestdl	morestdl -V -U morestdl [-day <i>num</i>] [-first -last -num <i>n</i> -all] [-twsllog] [{-name [" <i>nom_flot_travaux</i> [(<i>hhmm date</i>),(<i>ID_flot_travaux</i>).] <i>nom_travail</i> " <i>num_travail</i> -schedid <i>ID_flot_travaux.nom_travail</i>]}]
param	param -u -V param {-c -ec} [<i>fichier.section.</i> <i>fichier.</i> <i>section.</i>] <i>variable</i> [<i>valeur</i>] param [<i>fichier.section.</i> <i>fichier.</i> <i>section.</i>] <i>variable</i> param {-d -fd} [<i>fichier.section.</i> <i>fichier.</i> <i>section.</i>] <i>variable</i>
parms	parms {[-V -U] -build} parms {-replace -extract} <i>nom_fichier</i> parms [-d] <i>nom_paramètre</i> parms -c <i>valeur_nom_paramètre</i>
release	release -V -U release [-s] [<i>poste_de_travail#</i>] <i>nom_ressource</i> [<i>count</i>]
rmstdlist	rmstdlist -V -U rmstdlist [-p] [<i>age</i>]
sendevent	sendevent -V ? -help -U -usage sendevent [-hostname <i>nom_hôte</i>] [{-port -sslport} <i>port</i>] <i>type_événement</i> <i>source</i> [[<i>attribute=valeur</i>]...] Dans les environnements dynamiques : sendevent [-hostname <i>nom_hôte</i>] [-port <i>port</i>] <i>type_événement</i> <i>source</i> [[<i>attribute=valeur</i>]...]
showexec	showexec [-V -U INFO]
ShutDownLwa	ShutDownLwa
StartUp	StartUp [-V -U]
StartUpLwa	StartUpLwa
twsl_inst_pull_info	twsl_inst_pull_info -twsluser <i>ID_utilisateur</i> -log_dir_base <i>chemin</i> [-u [-run_db2_module [<i>y n</i>] -extract_db_defs [<i>y n</i>] -date <i>aaaammjj</i>]

Commandes d'utilitaire disponibles uniquement pour le système d'exploitation UNIX

Tableau 156. Commandes d'utilitaire disponibles uniquement pour UNIX

Commande	Syntaxe
at	at -V -U at -s <i>flot_travaux</i> -q<i>file_attentes</i>spéc-heure
batch	batch -V -U batch [-s <i>flot_travaux</i>]
showexec	showexec [-V -U -info]
version	version -V -U -h version [-a] [-f <i>fichier</i>v] [<i>fichier</i> [...]]

Commandes d'utilitaire disponibles uniquement pour le système d'exploitation Windows

Tableau 157. Commandes d'utilitaire disponibles uniquement pour Windows

Commande	Syntaxe
listproc (NON PRIS EN CHARGE)	listproc
killproc (NON PRIS EN CHARGE)	killproc <i>pid</i>
shutdown	shutdown [-V -U] [-appsrv]

Commandes de génération d'états

La présente section contient une liste et la syntaxe des commandes de génération d'états et des programmes d'extraction d'états. Ces commandes sont exécutées à partir de l'invite de commande du système d'exploitation.

Commandes de génération d'états

Tableau 158. Commandes de génération d'états

Nom	Sortie produite	Syntaxe
rep1	Etat 01 - Liste détails travaux	rep[x] [-V -U]
rep2	Etat 02 - Liste des invites	rep[x] [-V -U]
rep3	Etat 03 - Liste des agendas des utilisateurs	rep[x] [-V -U]

Tableau 158. Commandes de génération d'états (suite)

Nom	Sortie produite	Syntaxe
rep4a	Etat 04A - Liste des paramètres des utilisateurs	rep [x] [-V -U]
rep4b	Etat 04B - Liste des ressources TWS	rep [x] [-V -U]
rep7	Etat 07 - Liste des historiques des travaux	rep7 -V -U rep7 [-c <i>poste_de_travail</i>] [-s <i>nom_flot_travaux</i>] [-j <i>travail</i>] [-f <i>date</i> -t <i>date</i>] [-l]
rep8	Etat 08 - Histogramme des travaux	rep8 -V -U rep8 [-f <i>date</i> -b <i>heure</i> -t <i>date</i> -e <i>heure</i>] [-i <i>fichier</i>] [-p] rep8 [-b <i>heure</i> -e <i>heure</i>] [-i <i>fichier</i>] [-p]
rep11	Etat 11 - Calendrier de production planifiée	rep11 -V -U rep11 [-m <i>mm[aa]</i> [...]] [-c <i>poste_de_travail</i> [...]] [-s <i>nom_flot_travaux</i>] [-o <i>sortie</i>]
reptr	Etat 09A - Récapitulatif de la production planifiée Etat 09B - Détail de la production planifiée Etat 10A - Récapitulatif de la production réelle Etat 10B - Détails de la production réelle	reptr [-V -U] reptr -pre [-{summary detail}] [<i>fichier_sym</i>] reptr -post [-{summary detail}] [<i>fichier_journal</i>]
xref	Etat 12 - Etat de références croisées	xref [-V -U] xref [-cpu <i>poste_de_travail</i>] [-s <i>nom_flot_travaux</i>] [-depends -files -jobs -prompts -resource -schedules -when [...]]

Programmes d'extraction d'états

Tableau 159. Programmes d'extraction d'états

Programme d'extraction	Utilisé pour générer	Syntaxe
jbxtract	Etat 01 Etat 07	jbxtract [-V -U] [-j <i>travail</i>] [-c <i>poste_de_travail</i>] [-o <i>sortie</i>]
prxtract	Etat 02	prxtract [-V -U] [-o <i>sortie</i>] prxtract [-V -U] [-m <i>mm[aaaa]</i>] [-c <i>poste_de_travail</i>] [-o <i>sortie</i>]

Tableau 159. Programmes d'extraction d'états (suite)

Programme d'extraction	Utilisé pour générer	Syntaxe
caxtract	Etat 03	caxtract [-V -U] [-o <i>sortie</i>]
paxtract	Etat 04A	paxtract [-V -U] [-o <i>sortie</i>]
reextract	Etat 04B	reextract [-V -U] [-o <i>sortie</i>]
r11xtr	Etat 11	r11xtr [-V -U] [-m <i>mm[aaaa]</i>] [-c <i>poste_de_travail</i>] [-o <i>sortie</i>] [-s <i>nom_flot_travaux</i>]
xrxtrct	Etat 12	xrxtrct [-V -U]

Annexe D. Définition et gestion des travaux de branche génériques

IBM Workload Scheduler fournit une large gamme de fonctions de planification. Vous pouvez étendre ces fonctions pour des besoins spécifiques en utilisant une solution personnalisée supplémentaire, nommée travail de branche générique.

Pour plus d'informations sur l'utilisation de cette solution, consultez les sections suivantes :

- «Introduction»
- «Exemples de scénarios», à la page 967
- «Utilisation du travail de branche», à la page 1006
- «Indication des paramètres de travail de branche», à la page 1010
- «Remarques importantes sur le travail de branche», à la page 1022

Remarque : Le travail de branche générique n'est pas une composante native de IBM Workload Scheduler mais est tout de même pris en charge par le service de support logiciel IBM. Fourni avec des messages en anglais uniquement, il suit des conventions d'attribution de nom qui lui sont propres (pour de plus amples informations, voir «Placement du travail de branche dans le flot de travaux», à la page 1009).

Il est vivement recommandé d'utiliser le travail de branche dans votre environnement de test préalablement. La même recommandation s'applique également pour les exemples de scénarios. C'est seulement après avoir produit une exécution sans erreur de travail de branche ou d'exemple de scénario que vous pouvez effectuer le déplacement vers l'environnement de production.

De plus, si vous devez définir des flux alternatifs basés sur différentes sorties d'un travail ou d'un flot de travaux prédécesseur, vous pouvez tirer parti des dépendances conditionnelles. Pour plus d'informations, voir Chapitre 22, «Application d'une logique de branchement conditionnelle», à la page 827.

Introduction

Définir un travail de branche générique afin d'évaluer le statut ou la sortie d'un travail particulier et, selon les conditions indiquées, décider quels sont les travaux à exécuter dans le flot de travaux.

IBM Workload Scheduler propose des fonctions de génération de rapports, la planification basée sur les événements, et plusieurs autres fonctions. Toutefois, certaines tâches ne peuvent être exécutées qu'avec une programmation supplémentaire, tels que l'évaluation logique d'un flux de processus dans un flot de travaux.

Utilisez un travail de branche générique pour évaluer le statut ou la sortie d'un travail particulier et décidez quels sont les travaux à exécuter dans le flot de travaux. Les conditions déterminant quels sont les travaux à exécuter peuvent être très simples (par exemple, le prédécesseur s'est terminé avec l'état SUCC ou ABEND) ou être le résultat d'une opération booléenne ou arithmétique complexe.

La figure 50 illustre les concepts essentiels d'un travail de branche générique.

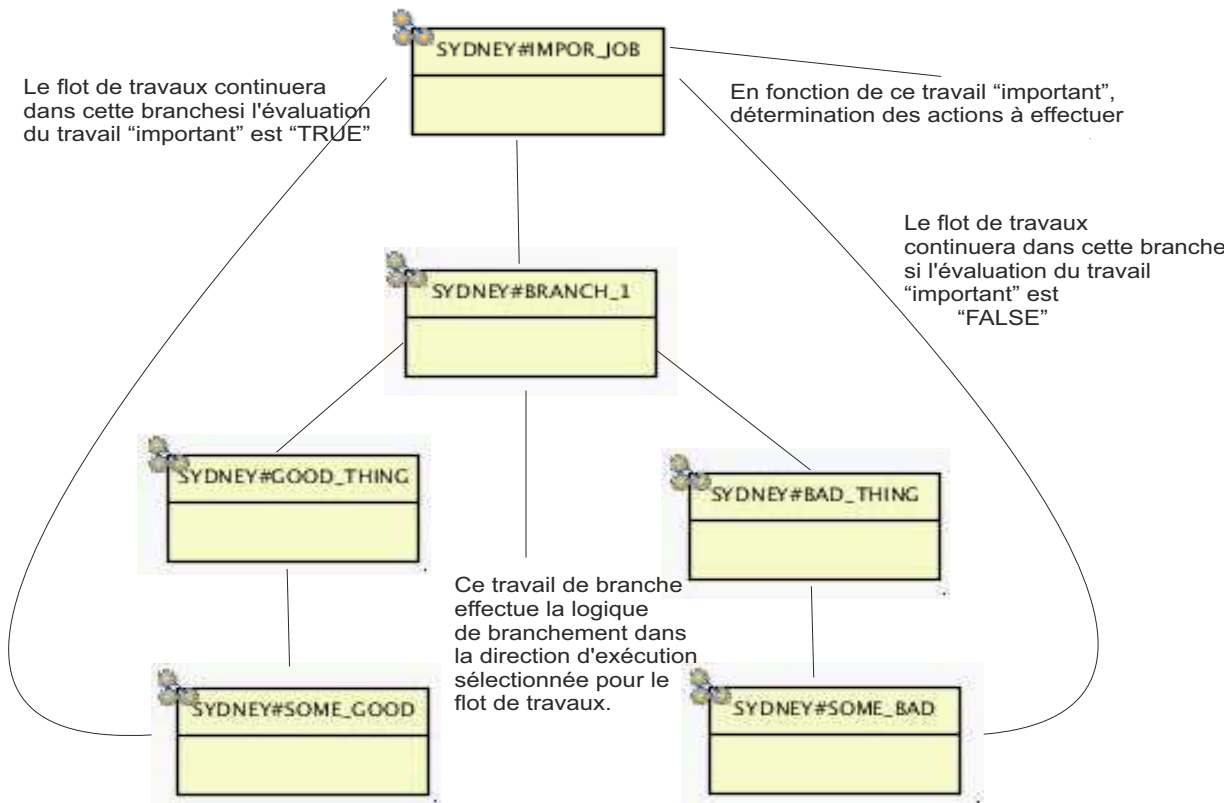


Figure 50. Objectif du travail de branche

La fonction de travail de branche générique n'est pas incluse dans IBM Workload Scheduler, mais a été développée par les spécialistes IBM pendant les différentes implémentations de IBM Workload Scheduler. Le travail de branche générique utilise l'interface ouverte de IBM Workload Scheduler.

Vous pouvez implémenter le simple branchement, selon le statut du travail prédécesseur (SUCC ou ABEND), en un temps très court. Toutefois, pour les scénarios plus complexes, par exemple rechercher un modèle et effectuer des comparaisons numériques, vous devez spécifier des paramètres d'entrée.

Terminologie

Les termes suivants permettent de décrire des travaux de branche :

Parent (parfois désigné sous le nom de travail évalué)

Travail dont vous souhaitez évaluer le statut ou d'autres propriétés.

Condition

Condition que vous exécutez sur le travail parent. La condition peut être aussi simple que *look at parent status* (examiner le statut parent) ou être plus complexe.

Enfant correct

Successeur le plus proche du travail de branche qui doit être exécuté si la condition est évaluée comme TRUE (vraie).

Enfant incorrect

Successeur le plus proche du travail de branche qui doit être stoppé si la condition est évaluée comme FALSE (fausse).

Branche

Séquence de travaux qui sont contrôlés à l'aide de la dépendance FOLLOWS.

Branche d'exécution

Branche sélectionnée à exécuter, en fonction du résultat de condition. La branche d'exécution démarre l'enfant correct si la condition est évaluée comme TRUE (vraie) ou avec l'enfant incorrect si la condition est évaluée comme FALSE (fausse).

Branche d'arrêt

Branche sélectionnée à arrêter, en fonction du résultat de condition. La branche d'arrêt s'arrête avec l'enfant incorrect si la condition est évaluée comme TRUE (vraie) ou avec l'enfant correct si la condition est évaluée comme FALSE (fausse).

Paramètre d'entrée

Argument transmis au travail spécifique de branche. Pour certains paramètres, si vous ne précisez aucune valeur, la valeur par défaut est utilisée.

Suffixe du travail de branche

Différencie plusieurs occurrences de travaux de branche dans le même flot de travaux.

La figure 51, à la page 964 illustre les termes suivants, qui sont utilisés pour définir un flot de travaux géré par un ou plusieurs travaux de branche :

- Parent
- Enfant correct et branche correcte
- Enfant incorrect et branche incorrecte
- Evaluation du statut du travail par rapport à la condition

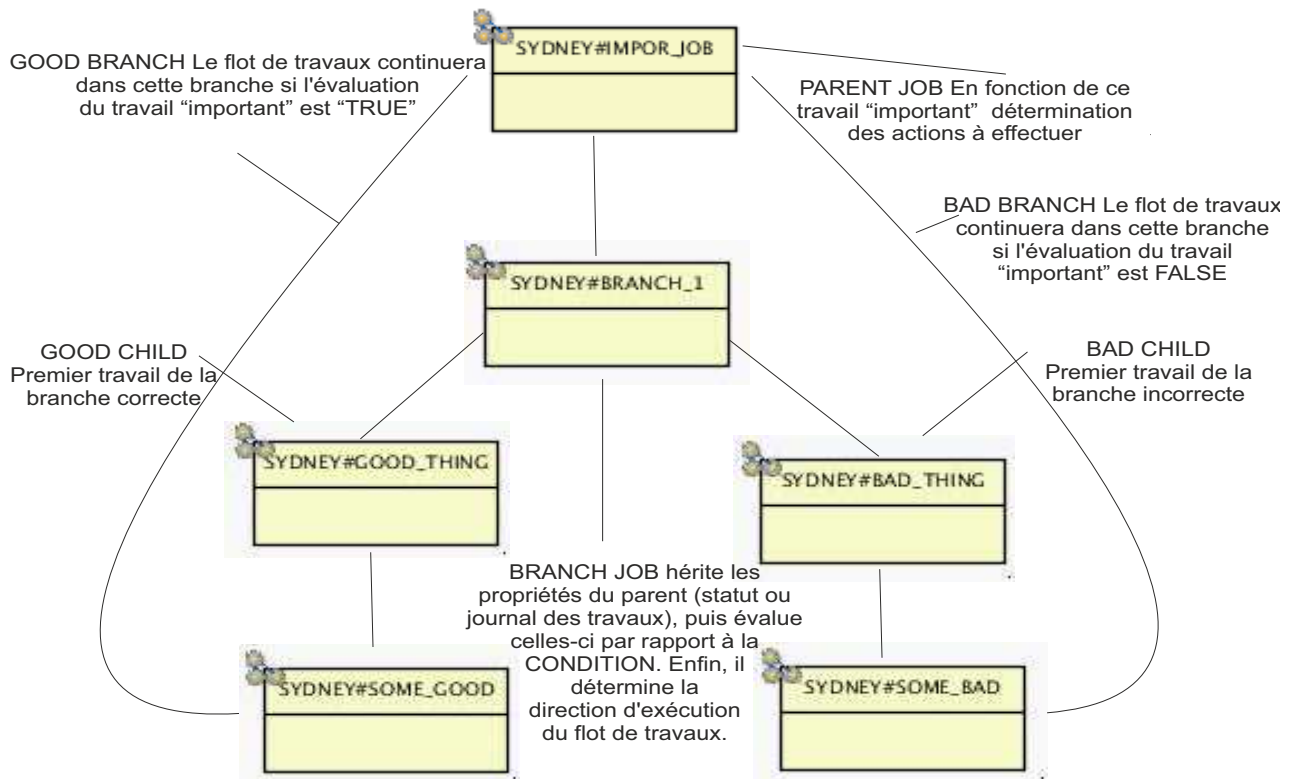


Figure 51. Termes associés à la définition de flot de travaux

La figure 52, à la page 965 illustre les termes supplémentaires qui sont utilisés pour l'exécution du flot de travaux :

- Branche d'exécution
- Branche d'arrêt

Affiche également la différence entre les termes corrects et incorrects, l'exécution et l'arrêt. Les termes sont identiques uniquement lorsque `CONDITION=TRUE`. Si `CONDITION=FALSE`, la branche d'exécution correspond à la branche incorrecte et la branche d'arrêt correspond à la branche correcte. Il s'agit du concept de logique d'évaluation de travail de branche.

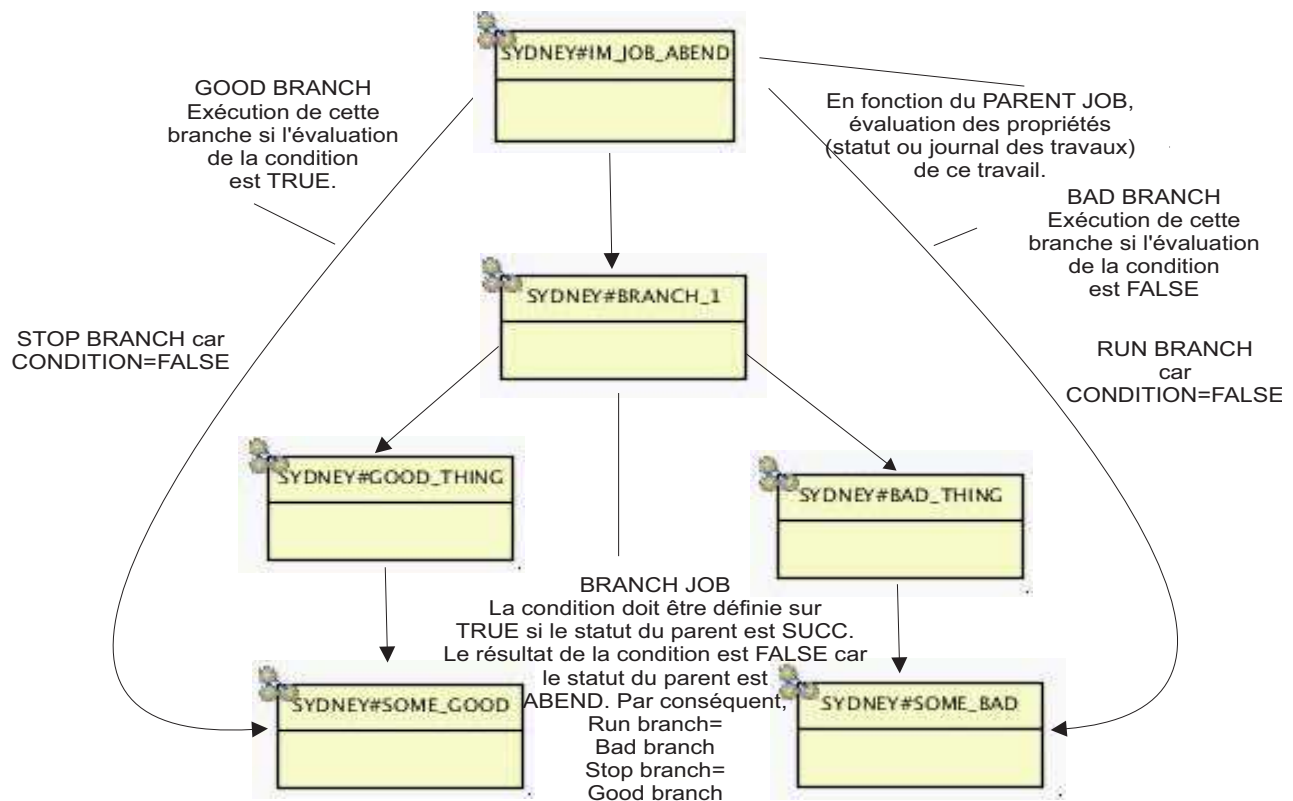


Figure 52. Termes liés à l'exécution de flot de travaux (instance de flot de travaux concrète)

Fonctions de travail de branche

Pour décrire des fonctions de travail de branche générique, le processus de branchement peut être divisé en deux processus :

- **Evaluation** : le travail de branche prend toujours son parent comme entrée. Ensuite, en fonction des paramètres spécifiés, il crée une des sous-conditions suivantes :
 - Vérifie si le parent est terminé avec l'état SUCC ou ABEND.
 - Crée une condition complexe à partir d'une ou de plusieurs sous-conditions parmi les suivantes :
 - Obtient le journal des travaux parent et recherche une ligne comportant le modèle de texte indiqué (transmis au travail de branche comme paramètre d'entrée).
 - (Facultatif) Recherche un autre modèle sur la même ligne.
 - (Facultatif) Recherche une valeur numérique sur la même ligne. Cette valeur numérique est comparée à un nombre défini à l'aide d'un opérateur arithmétique spécifié (le numéro et l'opérateur arithmétique sont transmis au travail de branche en tant que paramètres d'entrée).

Ces sous-conditions sont liées à l'aide des opérateurs booléens AND ou OR. La condition complexe est évaluée immédiatement.

Le résultat de l'évaluation (même si la condition est simple ou complexe) est TRUE ou FALSE.

- **Action** : en fonction du résultat de condition, le travail de branche détermine quelle est la branche à exécuter et quelle est celle à arrêter. Il réalise alors différentes actions sur la branche d'exécution et la branche d'arrêt, comme suit.
 - Les actions possibles sur la branche d'exécution sont :
 - DO NOTHING**
La branche d'exécution est exécutée.
 - RELEASE**
Si le premier travail de la branche d'exécution est suspendu, il est libéré.
 - Les actions possibles sur la branche d'arrêt sont :
 - CANCEL**
Tous les travaux de la branche de travail sont annulés.
 - PAUSE**
Le travail de branche est interrompu. Les travaux de la branche d'arrêt ne peuvent pas s'exécuter car leur prédécesseur est mis en attente.
 - De plus, il existe une action spéciale appelée SIGNAL. Cette action écrit uniquement une recommandation pour vous invitez à confirmer le travail de branche dans le journal de travail. Vous créez un travail de branche SIGNAL en définissant l'indicateur Confirmation requise.
Pour plus d'informations sur le travail de branche SIGNAL, voir scénario d'action signal

Vous pouvez combiner les critères d'évaluation et les actions consécutives à votre gré.

Avantages du travail de branche

Les principaux points forts du travail de branche générique sont :

- Un seul travail de branche est défini dans la base de données IBM Workload Scheduler.
Le travail de branche générique est représenté par une définition de travail qui pointe vers un script de shell. Le script ne prend aucun argument de ligne de commande qui pointe vers le parent et les enfants du travail de branche. Les informations sur le parent (prédécesseur) et les enfants (successeurs) sont évalués automatiquement.
- Il n'est pas nécessaire d'indiquer des paramètres d'entrée si vous utilisez la plupart des scénarios communs de travail de branche.
Vous n'avez pas à spécifier de paramètres lors de l'évaluation de l'état du résultat du travail parent (SUCC ou ABEND). Vous insérez le travail de branche dans le flot de travaux, vous placez les dépendances de prédécesseur/successeur et attribuez aux travaux enfants les noms spécifiques qui identifient l'enfant correct et l'enfant incorrect.
- Le travail de branche renvoie un journal de travail structuré qui contient des informations détaillées sur l'environnement du travail de branche, les paramètres d'entrée, la condition évaluée, les actions exécutées. En lisant le journal de travail, vous pouvez facilement contrôler les activités que le travail de branche a réalisées.
- Le travail de branche utilise la représentation des objets IBM Workload Scheduler dans le plan en cours. Cela signifie que tous les objets du plan (par

exemple, les travaux et flots de travaux) sont référencé à l'aide du mot clé `schedid`. Assurez-vous que toutes les actions lancées sur les objets du pointent vers un objet unique.

Cette fonction est optimisée dans toutes les occurrences de flot de travaux dans le plan courant :

- Toute occurrence du flot de travaux soumise sans spécifier d'alias.
- Toute occurrence du flot de travaux soumise en spécifiant l'alias.
- Le travail de branche générique s'exécute sur les deux gestionnaires de domaines maître UNIX et Windows. Etant donné que le travail de branche est consigné dans le script de shell (par conséquent il s'exécute en mode natif sous UNIX), sur les systèmes d'exploitation Windows vous devez utiliser un interpréteur de shell UNIX.
- Le travail de branche générique s'exécute même si le flot de travaux est défini sur la classe de poste de travail. Le travail de branche lui-même doit être défini sur le gestionnaire de domaine maître.

Exemples de scénarios

Les exemples de scénarios décrivent les différents travaux de branche, les concepts principaux et l'utilisation des travaux de branche.

Le nom des vues possède la structure suivante :

- Utilisation du scénario.
- Définition d'exemple de flot de travaux.
- Journal du travail de branche générique.
- Paramètres requis pour le branchement.
- Positionnement du travail dans le flot de travaux et modification des noms de travaux enfant, si nécessaire.

Un travail générique de branche basé sur le type de condition diffère d'un travail générique de branche basé sur le type d'action, comme suit :

Condition

Indique les critères afin de déterminer la branche d'exécution (travaux qui doivent continuer) et la branche d'arrêt (travaux qui doivent arrêter).

Action

Indique l'action à effectuer sur la branche d'exécution et la branche d'arrêt. Vous pouvez effectuer deux types d'action sur la branche d'arrêt et deux types d'actions sur la branche d'exécution. De plus, il existe une action spéciale de *signalement*.

Scénarios basés sur le type de condition

Utilisez un travail générique de branche en fonction du type de condition pour indiquer les critères qui déterminent la branche d'exécution et la branche d'arrêt.

Pour les scénarios basés sur le type d'action, voir la rubrique consacrée à ces scénarios.

Scénario de branche simple

Utilisez le branchement simple pour évaluer le statut d'un travail.

Utilisation de branche simple

Selon le statut renvoyé par le travail que vous évaluez, une branche spécifique s'exécute : si le travail se termine à l'état SUCC, la branche correcte s'exécute ; si le travail se termine de façon anormale, la branche incorrecte s'exécute.

Branche simple se terminant avec l'état SUCC

La figure 53 affiche la définition de flot de travaux.

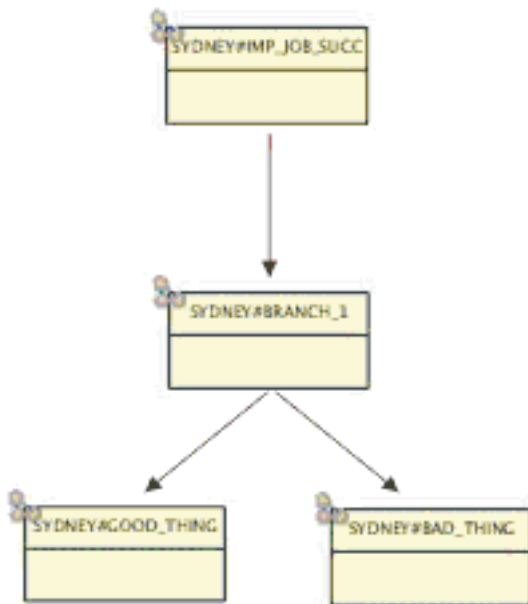


Figure 53. Définition de branche simple (SUCC)

Si le travail parent se termine à l'état SUCC, l'enfant incorrect est annulé et l'enfant correct s'exécute. La figure 54 affiche l'état final du flot de travaux dans le plan courant.

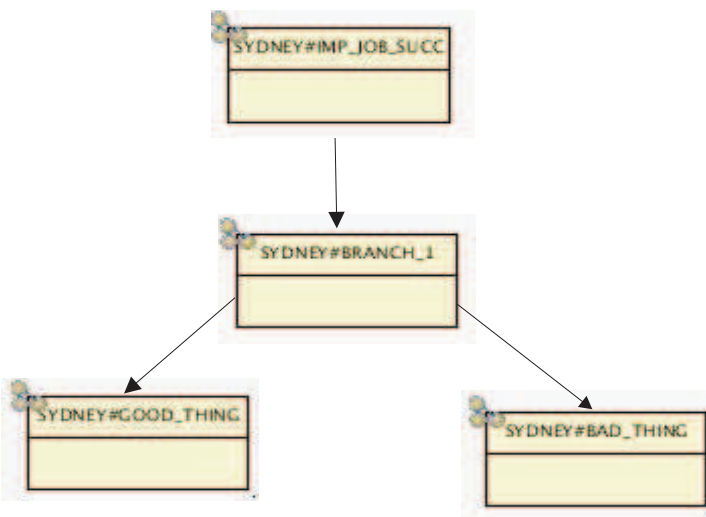


Figure 54. Statut final de branche simple (SUCC)

Fin de branche simple à l'état ABEND

La figure 55 affiche la définition de flot de travaux.

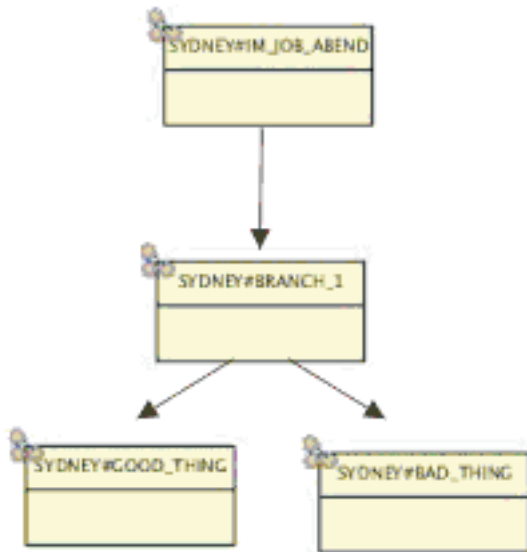


Figure 55. Définition de branche simple (ABEND)

Le travail évalué (c'est-à-dire, le parent) doit avoir l'option de reprise définie sur Continuer, sinon le travail arrêté de manière anormale ne libère pas le travail de branche de la dépendance FOLLOWS et le travail se termine à l'état STUCK. Cela concerne *tous* les travaux parents pour lesquels vous évaluez l'état du résultat, car vous ne devez pas oublier que n'importe quel travail peut se terminer de façon anormale.

L'exemple suivant affiche le journal de l'instance de travail de branche générique :

```

===== START of branch job =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=SIMPLE_BRANCH_B
PARAMETER_PREFIX=
JOB_NAME=BRANCH_1
BRANCH_SUFFIX=1
PARENT=IMPORTANT_JOB_ABEND
=====
===== Input parameters =====
INPUT_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
CONDITION_COUNT=0
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].G_DO_THE_GOOD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].SOME_GOOD_1
  
```

```

Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].SOME_GOOD_2
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
For action CANCEL-RUN_BRANCH=B_DO_THE_BAD_THING and STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING, SOME_GOOD_1, SOME_GOOD_2
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le branchement basé sur l'évaluation du statut du travail parent est l'utilisation la plus courante du travail générique de branche ; il ne nécessite *aucun* paramètre d'entrée.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Scénario de branche longue

le branchement long est l'utilisation récursive du branchement simple.

Utilisation de branche longue

Le but principal du scénario de branche longue est de montrer que le travail de branche générique peut annuler *tous* les travaux dans la branche d'arrêt, même s'il existe une arborescence entière de travaux d'annulation.

Cette fonction est nécessaire car, dans IBM Workload Scheduler, si un travail est annulé, tous les successeurs du travail sont libérés de la dépendance de prédécesseur/successeur, ce qui implique que les travaux qui sont *dépendants* du travail *annulé* démarrent immédiatement. Puisque ce comportement peut être non souhaité dans certains cas, le travail de branche générique annule le premier travail d'arrêt et *tous* ses successeurs.

Remarque : Du point de vue de programmation, le travail générique de branche utilise des appels de fonction récursifs pour parcourir tous les successeurs du premier enfant du travail de branche générique.

Fin de branche longue à l'état SUCC

La figure 56 présente le flot de travaux contenant la structure complexe des successeurs possibles, dans la branche correcte ou incorrecte.

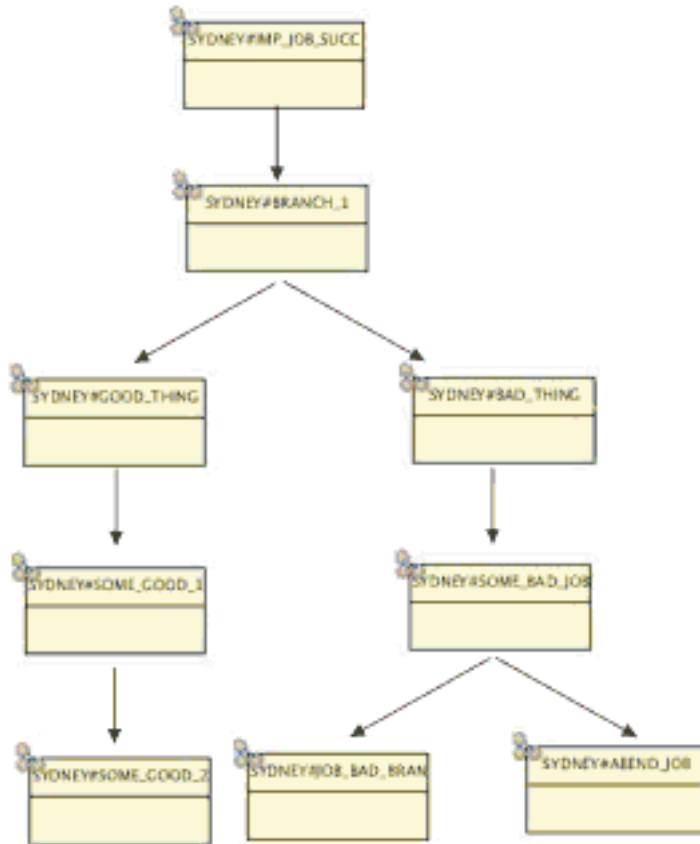


Figure 56. Définition de branche longue (SUCC)

Si le parent s'est terminé à l'état SUCC, le journal de travail montre la sortie suivante :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_LONG_SUCC
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_SUCC
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC3.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
(0AAAAAAAAAAAAEC3)].B_DO_THE_BAD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.SOME_BAD_JOB
%cj SYDNEY#0AAAAAAAAAAAAEC3.SOME_BAD_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),

```

```

(0AAAAAAAAAAAAEC3)].SOME_BAD_JOB
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.ABEND_JOB
%cj SYDNEY#0AAAAAAAAAAAAEC3.ABEND_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
(0AAAAAAAAAAAAEC3)].ABEND_JOB
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC3.ANOTHER_JOB_IN_BAD_BRANCH
%cj SYDNEY#0AAAAAAAAAAAAEC3.ANOTHER_JOB_IN_BAD_BRANCH;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_SUCC[(2154 12/16/07),
(0AAAAAAAAAAAAEC3)].ANOTHER_JOB_IN_BAD_BRANCH
=====
===== Action on RUN Branch =====
Performing action RELEASE on job
SYDNEY#0AAAAAAAAAAAAEC3.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC3.G_DO_THE_GOOD_THING
is NOT NECESSARY,
    because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
TRUE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC.
For action CANCEL-RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING, SOME_BAD_JOB, ABEND_JOB, ANOTHER_JOB_IN_BAD_BRANCH
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Fin de branche longue à l'état ABEND

La figure 57, à la page 973 montre comment le flot de travaux est exécuté si le travail évalué s'est arrêté de manière anormale.

Remarque : Pour libérer les successeurs de la dépendance de prédécesseur/successeur, le travail évalué (travail parent) doit avoir l'option de reprise définie sur Continue. Vous pouvez définir ce paramètre uniquement dans la définition de travail, non dans la définition de flot de travaux.

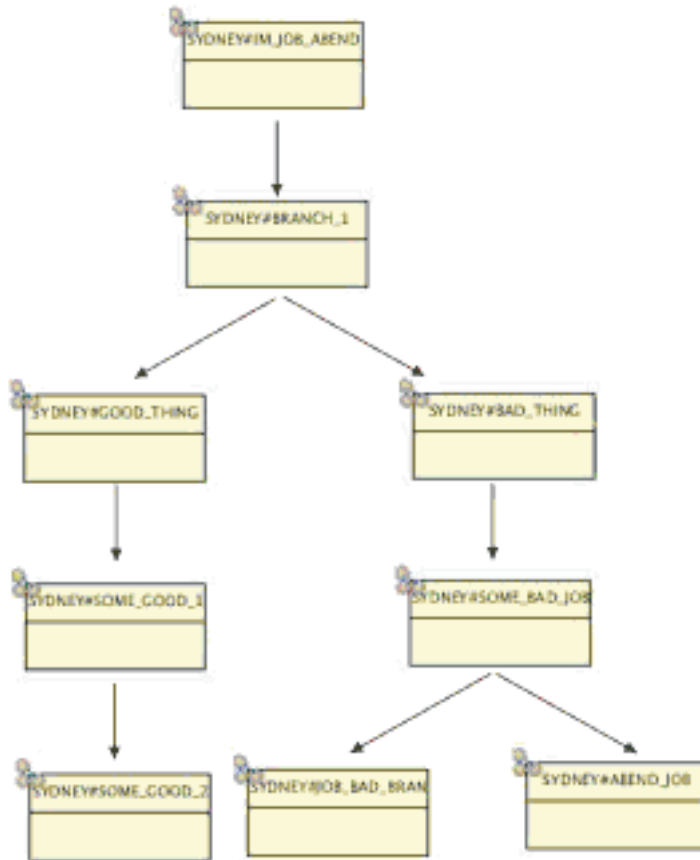


Figure 57. Statut final de branche longue(ABEND)

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_LONG_ABEND
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_ABEND
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEC4.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].G_DO_THE_GOOD_THING
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_1;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].SOME_GOOD_1
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2
%cj SYDNEY#0AAAAAAAAAAAAEC4.SOME_GOOD_2;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_LONG_ABEND[(2159 12/16/07),
(0AAAAAAAAAAAAEC4)].SOME_GOOD_2

```

```

=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEC4.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: Searched for SUCC parent. Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
For action CANCEL-RUN_BRANCH=B_DO_THE_BAD_THING and STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING, SOME_GOOD_1, SOME_GOOD_2
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Ce type de branchement ne nécessite aucun paramètre d'entrée.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Branches multiples

Utilisez la fonction de branche multiple pour disposer de travaux de branche différents dans un même flot de travaux.

Utilisation de branches multiples

Le branchement multiple permet de disposer de plusieurs travaux de branche dans un même flot de travaux. La figure 58, à la page 975 présente un exemple d'un flot de travaux géré par les travaux de branches multiples.

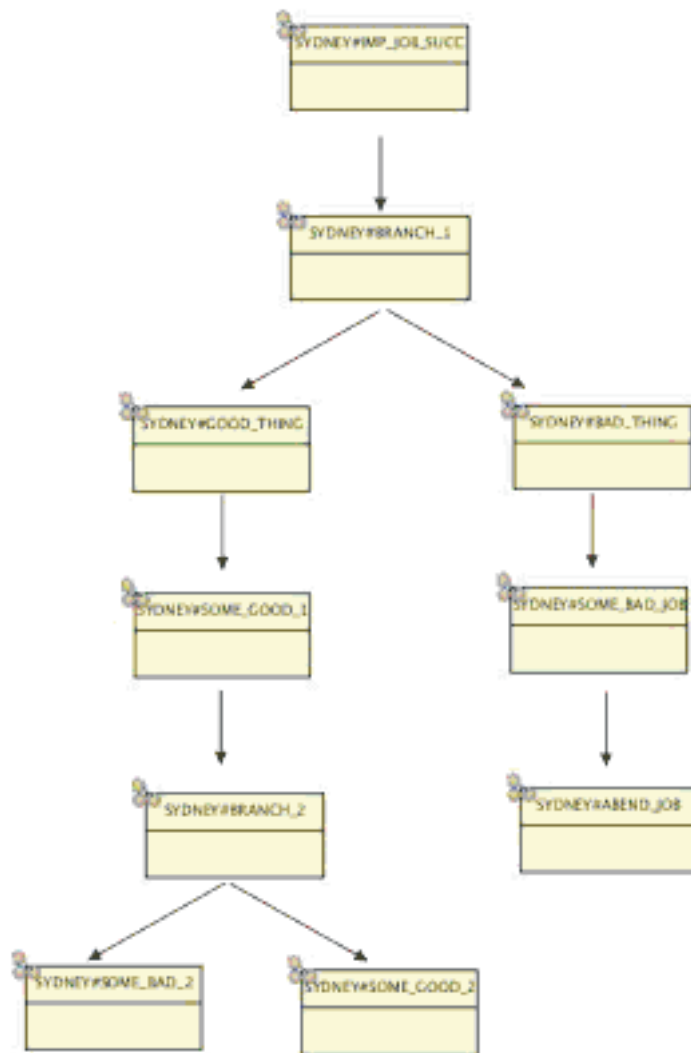


Figure 58. Travaux de branches multiples dans un même flot de travaux

Paramètres d'entrée obligatoires

Il peut s'avérer nécessaire de définir les paramètres d'entrée en fonction des travaux de branche utilisés. Vous n'êtes pas obligé d'indiquer des paramètres pour les travaux basés sur les scénarios suivants :

- Branche simple
- Branche longue

Indiquez les paramètres d'entrée pour les travaux basés sur les scénarios suivants :

- Branche complexe - modèle
- Branche complexe - modèle dans la ligne de modèle
- Branche complexe - comparaison d'une valeur numérique
- Scénario complexe - conditions multiples
- Scénario d'actions de pause/libération
- Scénario d'action de signal

Placement des travaux de branche dans le flot de travaux

Pour chaque travail de branche, insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_". Distinguez les travaux de branche différents par *suffixe de branche*, qui doit comprendre un *caractère de soulignement* et un *numéro*.

La meilleure pratique consiste à créer le suffixe de branche à l'aide de numéros dans l'ordre croissant, de sorte que les travaux de branche multiples dans un flot de travaux sont nommés, par exemple, BRANCH_1, BRANCH_2, et ainsi de suite.

Fin anormale du parent

Le scénario de fin anormale du parent représente le cas inversé du scénario de branche simple.

Utilisation de fin anormale du parent

Le scénario de fin anormale du parent décrit les exigences fonctionnelles suivantes :

- Obtention du statut du travail parent.
- Si le statut du travail parent est SUCC, il est considéré comme INCORRECT.
- Si le statut du parent est ABEND, il est considéré comme CORRECT.

Cette section décrit uniquement le cas où le travail parent se termine à l'état SUCC.

Fin de fin anormale du parent à l'état SUCC

La figure 59, à la page 977 présente le flot de travaux avec la fonction PARENT_ABEND.

Le flot de travaux ressemble à la branche simple, la seule différence est que vous avez spécifié le paramètre CONDITION_SWITCH=PARENT_ABEND pour le travail de branche.

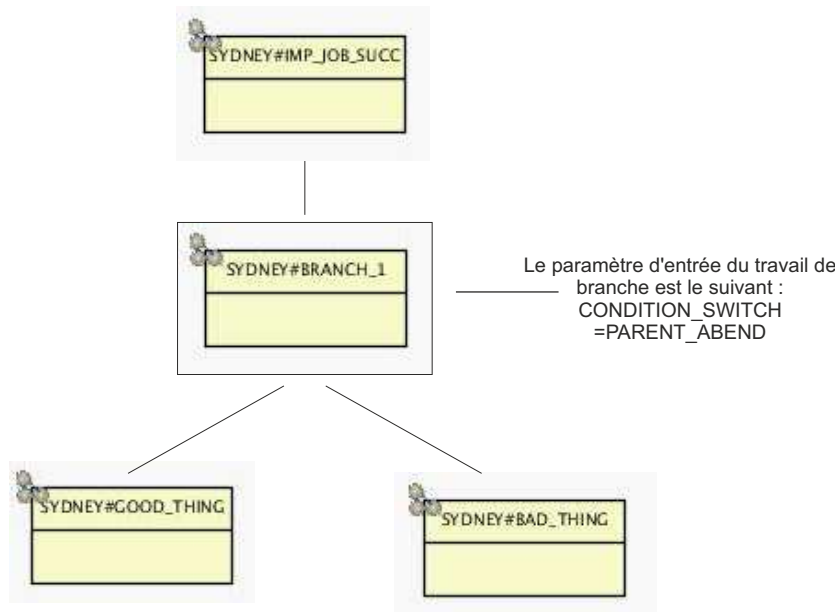


Figure 59. Définition de fin anormale de parent (SUCC)

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PARENT_ABEND
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_SUCC
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_ABEND
ACTION_SWITCH=CANCEL
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_ABEND
FALSE: Searched for ABEND parent.
Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEDA.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEDA.G_DO_THE_GOOD_THING:schedid:noask
Command forwarded to batchman for SYDNEY#GBJ_PARENT_ABEND[(2314 12/16/07),
(0AAAAAAAAAAAAEDA)].G_DO_THE_GOOD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEDA.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEDA.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: Searched for ABEND parent. Status of PARENT JOB(IMPORTANT_JOB_SUCC) is SUCC
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====
  
```

Paramètres d'entrée obligatoires

Le tableau 160 affiche les paramètres requis pour le scénario de branche négatif.

Tableau 160. Paramètres d'entrée pour le scénario de travail de branche négatif

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	PARENT_ABEND

L'exemple suivant illustre la définition de paramètre. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```
BRANCH_1-BEGIN  
CONDITION_SWITCH=PARENT_ABEND  
BRANCH_1-END
```

Pour plus d'informations sur la définition des paramètres de travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Scénarios de branche complexe

les scénarios de branche complexe affichent la flexibilité du travail de branche générique.

Les scénarios précédents ont été basés sur l'évaluation du *statut* du journal parent. Les scénarios de travail de branche complexe sont basés sur l'évaluation du *journal de travail* du travail parent. Pour obtenir une liste des conditions complexes, voir terminologie.

Une condition complexe peut comprendre plusieurs sous-conditions. Par exemple :

1. sous-condition 1
 - Rechercher le modèle de texte (indiqué comme paramètre) dans le journal du travail parent.
 - Isoler la valeur numérique dans la ligne sur laquelle le modèle est trouvé.
 - Comparer la valeur numérique avec un nombre spécifique. Pour cette comparaison, utiliser l'opérateur arithmétique indiqué (indiqué comme paramètre). Si la comparaison arithmétique aboutit, elle renvoie TRUE.

Par exemple, recherchez la ligne contenant le modèle Disque disponible. Si vous la trouvez, recherchez un nombre dans la ligne. Si ce nombre est supérieur à 90, renvoyez TRUE.
2. sous-condition 2
 - a. Rechercher un autre modèle de texte dans le journal du travail parent.
 - b. Si le modèle est détecté, annuler le résultat et renvoyer FALSE (par exemple, renvoyer FALSE si le journal du travail parent contient le modèle Erreur).

3. Connecter les résultats de la première et de la deuxième sous-condition avec un opérateur booléen (l'opérateur booléen est indiqué comme paramètre, les valeurs possibles sont AND et OR).
4. Evaluer le résultat final de toute la condition (TRUE ou FALSE).

La condition définie peut être très flexible et peut couvrir de nombreuses situations standard.

Les sections suivantes décrivent plusieurs scénarios de branche complexe pour expliquer comment :

- Rechercher un modèle spécifique de texte dans le journal du travail parent.
- Rechercher un modèle spécifique de texte dans le journal du travail parent. Rechercher un modèle de texte supplémentaire dans la même ligne si le modèle est trouvé.
- Rechercher un modèle spécifique de texte dans le journal du travail parent. Rechercher une valeur numérique dans la même ligne si le modèle est trouvé. Si la valeur est trouvée, la comparer avec le numéro indiqué à l'aide de l'opérateur arithmétique fourni.
- Combiner plusieurs conditions en une en utilisant les opérateurs booléens AND et OR.

Branche complexe - modèle

Utilisez le scénario de modèle de branche complexe pour rechercher un modèle de texte spécifique dans le journal de travail parent.

Utilisation du scénario de modèle

Vous pouvez rechercher n'importe quelle chaîne, par exemple terminé correctement ou TOUTES LES unités de bande montées. Généralement le texte doit représenter le message positif inclus dans le journal du travail parent.

Remarque : La recherche de modèle recherche généralement le message positif du journal de travail parent. Dans certains cas, vous pouvez être amené à implémenter la logique inversée, par exemple rechercher le modèle Erreur, sans succès ou Espace insuffisant, qui représentent les messages négatifs du journal. Pour utiliser cette méthode, voir «Branche complexe - modèle refusé», à la page 981.

A l'aide de la branche de modèle, si le modèle de texte est détecté, alors
CONDITION=TRUE sinon CONDITION=FALSE.

La figure 60, à la page 980 affiche la définition du flot de travaux pour le scénario de branche de modèle. Le flot de travaux est identique au simple scénario de branche, mais les paramètres supplémentaires sont définis :

- CONDITION_SWITCH=COMPLEX
- PATTERN_1=completed successfully

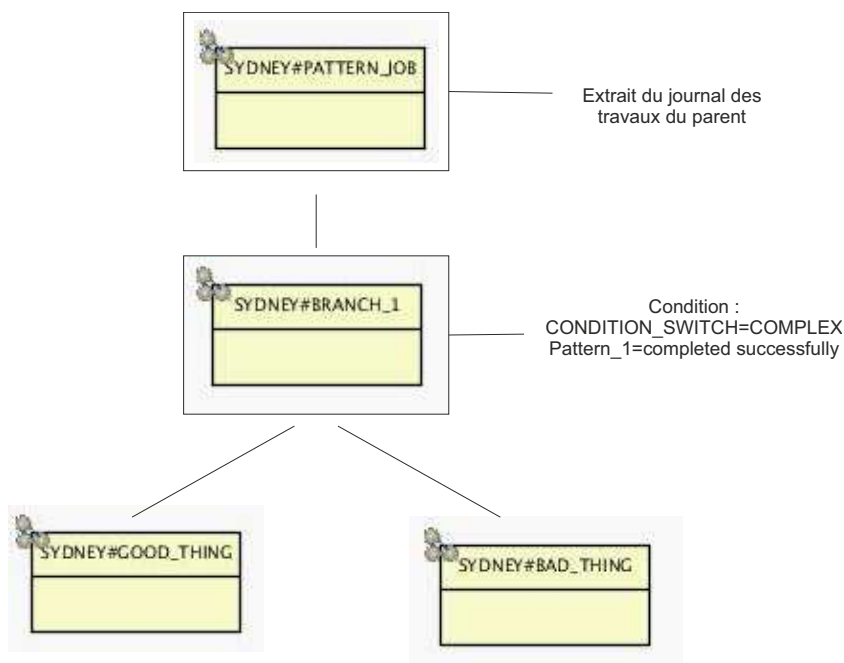


Figure 60. Scénario de modèle - définition

Le journal des travaux du scénario de modèle affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=completed successfully
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "completed successfully" in JOBLOG of PATTERN_JOB
Pattern FOUND, performing further tests.
No additional value defined for specified pattern.
Condition evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-----
----- COMPLEX CONDITION -----
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED4.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAED4.B_DO_THE_BAD_THING:schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN[(1733 12/17/07),
(0AAAAAAAAAAAAED4)].B_DO_THE_BAD_THING
=====
===== Action on RUN Branch =====

```

```

Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAED4.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAED4.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 161 affiche les paramètres requis pour le scénario de modèle.

Tableau 161. Paramètres d'entrée pour le scénario de travail de modèle

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Completed successfully (Terminé correctement)

L'exemple suivant illustre la définition de paramètre. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=completed successfully
BRANCH_1-END

```

Pour une description sur la spécification des paramètres du travail de branche, voir Gestion des paramètres de travail de branche.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Branche complexe - modèle refusé

Il s'agit du cas inversé du scénario de branche de modèle.

Utilisation du scénario de modèle refusé

Avec le scénario de branche de modèle vous trouvez le journal de travail parent d'un modèle considéré comme un message positif (par exemple, terminé correctement). Si, au contraire, vous souhaitez rechercher un message négatif (par exemple, Error), utilisez le travail de branche générique pour refuser chaque

sous-condition définie de la condition complexe. Dans ce scénario, la recherche du modèle a pour conséquence d'activer CONDITION=FALSE. Si le modèle est introuvable, alors CONDITION=TRUE.

La figure 61 affiche la définition du flot de travaux pour le scénario de branche de modèle, avec les paramètres suivants :

- CONDITION_SWITCH=COMPLEX
- PATTERN_1=Error
- NEGATE_CONDITION_RESULT_1=YES

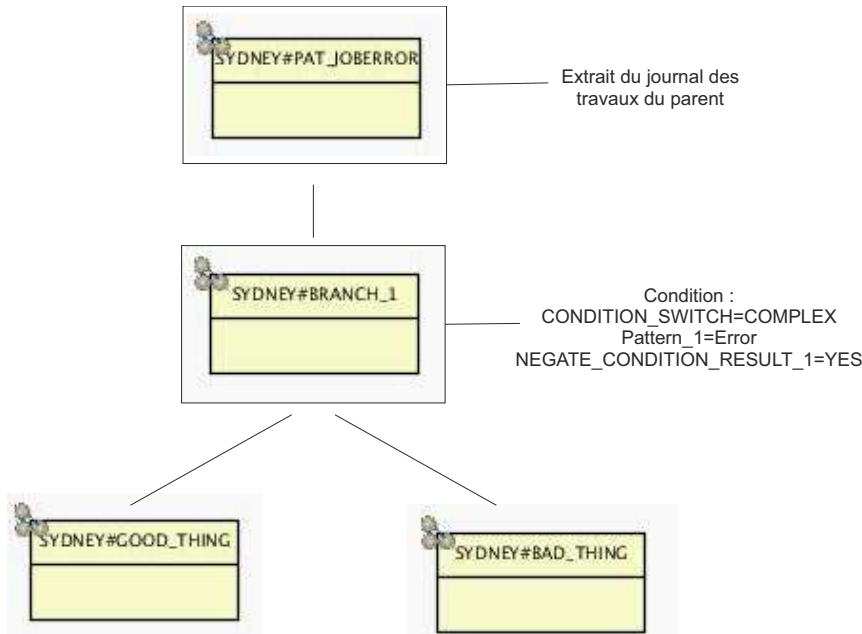


Figure 61. Définition du scénario de modèle annulé

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_NEG
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_JOB_ERROR
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Error
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "Error" in JOBLOG of PATTERN_JOB_ERROR
Pattern FOUND, performing further tests.
No additional value defined for specified pattern.
Condition evaluated as TRUE.
==NEGATED== ATOMIC CONDITION RESULT [1]= FALSE
  
```

```

-----
----- COMPLEX CONDITION -----
[ FALSE ]
CONDITION_RESULT=FALSE
FALSE: The result of complex condition is FALSE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED3.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAED3.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN_NEG[(1730 12/17/07),
(0AAAAAAAAAAAAED3)].G_DO_THE_GOOD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAED3.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAED3.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 162 affiche les paramètres requis pour le scénario de branche refusé.

Tableau 162. Paramètres d'entrée pour le scénario de travail de modèle refusé

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Error
NEGATE_CONDITION_RESULT_1	OUI

L'exemple suivant illustre la définition de paramètre. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_1=YES
BRANCH_1-END

```

Pour une description sur la spécification les paramètres du travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Branche complexe - modèle dans la ligne de modèle

Ce scénario étend la fonction de la recherche de modèle.

Modèle dans l'utilisation de scénario de ligne de modèle

L'objet de ce scénario est de :

1. Obtenir le journal de travail parent.
2. Dans le journal de travail, identifier la ligne contenant un modèle spécifique.
3. Si la ligne est détectée, rechercher un autre modèle dans la ligne.
4. Si le second modèle est trouvé, renvoyer la condition comme TRUE.

L'exemple suivant extrait d'un journal de travail parent affiche ce à quoi ressemble un scénario d'utilisation standard :

```
Sending STOP signal to component XYZ: SUCCESS  
Stopping component XYZ: SUCCESS
```

Selon cet exemple, une recherche simple pour le modèle de texte SUCCESS n'indiquerait pas l'arrêt effectif du composant. Il y a plusieurs occurrences du modèle SUCCESS et l'utilisation de la simple recherche de modèle ne vous permet pas de déterminer le résultat correct.

L'autre exemple proposé concerne un extrait du journal des travaux :

```
About to stop component XYZ...  
Sending STOP signal to component XYZ: SUCCESS  
Stopping component XYZ: FAILED
```

Suivant cet exemple, si vous avez utilisé la recherche simple de modèle, le résultat est CONDITION=TRUE, car le modèle SUCCESS est détecté. Mais dans ce cas, le journal du travail parent n'est pas évalué correctement.

Pour évaluer correctement le journal du travail parent, vous devez utiliser le modèle dans le scénario de ligne de modèle, comme suit :

1. Recherchez le modèle Composant d'arrêt.
2. Si la ligne est détectée, recherchez le modèle SUCCESS.
3. Si les deux recherches sont réussies, alors renvoyez la condition comme VRAIE.

La figure 62, à la page 985 affiche la définition du scénario de branche de modèle.

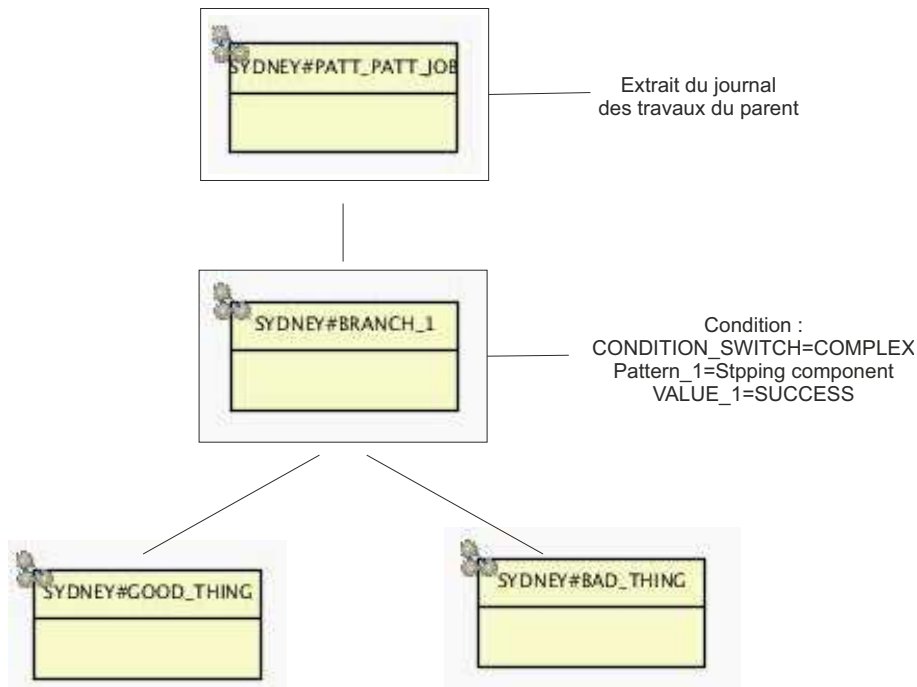


Figure 62. Modèle dans la définition de ligne de modèle

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_PATT
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_PATTERN_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION COUNT=1
PATTERN[1]=Stopping component
VALUE[1]=SUCCESS
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "Stopping component" in JOBLLOG of PATTERN_PATTERN_JOB
Pattern FOUND, performing further tests.
Searching for STRING=SUCCESS within the
  row "Stopping component XYZ: FAILED".
String "SUCCESS" NOT found within the
  row "Stopping component XYZ: FAILED".
ATOMIC CONDITION RESULT [1]= FALSE
-----
----- COMPLEX CONDITION -----
[ FALSE ]
CONDITION_RESULT=FALSE
FALSE: The result of complex condition is FALSE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAED2.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAED2.G_DO_THE_GOOD_THING;schedid;noask
  
```

```

Command forwarded to batchman for SYDNEY#GBJ_PATTERN_PATT[(1634 12/17/07),
(0AAAAAAAAAAAAED2)].G_DO_THE_GOOD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAED2.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAED2.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 163 affiche les paramètres requis pour le modèle dans le scénario de ligne de modèle.

Tableau 163. Paramètres d'entrée pour le modèle dans le scénario de ligne de modèle

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Arrêt du composant
VALUE_1	SUCCESS

La définition de paramètre ressemble à celle de l'exemple suivant. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Arrêt du composant
VALUE_1=SUCCESS
BRANCH_1-END

```

Pour plus d'informations sur la définition des paramètres de travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux juste après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Modèle dans la ligne de modèle - annulé

Dans le scénario de modèle annulé, vous annulez le résultat d'une recherche de modèle simple, dans le modèle annulé au sein du scénario de ligne de modèle, vous utilisez l'approche *annulée* pour *tout* scénario complexe. C'est le cas inversé du modèle dans le scénario de ligne de modèle.

Modèle annulé dans l'utilisation de scénario de ligne de modèle

Si vous effectuez une simple recherche de modèle, par exemple une recherche de l'occurrence ERROR dans le journal de travail parent, ou une condition complexe, vous pouvez utiliser l'approche qui annule le résultat.

Choisissez la façon d'analyser la syntaxe du journal de travail parent. En fonction de votre compréhension du contenu du journal des travaux, indiquez si la sortie des lignes contient un message positif ou négatif. A partir de ces connaissances, choisissez d'annuler le résultat de la sous-condition particulière ou de le conserver en l'état.

La logique d'évaluation du modèle annulé dans le scénario de ligne de modèle est la suivante :

1. Recherche de la ligne de modèle.
2. Recherche de l'identificateur principal, tel que Backup on Primary device.
3. Si la ligne est détectée, recherchez-y le message négatif ERROR.
4. Annulez le résultat.

Cette approche couvre le scénario d'utilisation des sorties telles que SUCCESS, OK et COMPLETED comme messages positifs, et des sorties telles que FAILED et ERROR comme messages négatifs.

La figure 63 affiche la définition du scénario de branche de modèle.

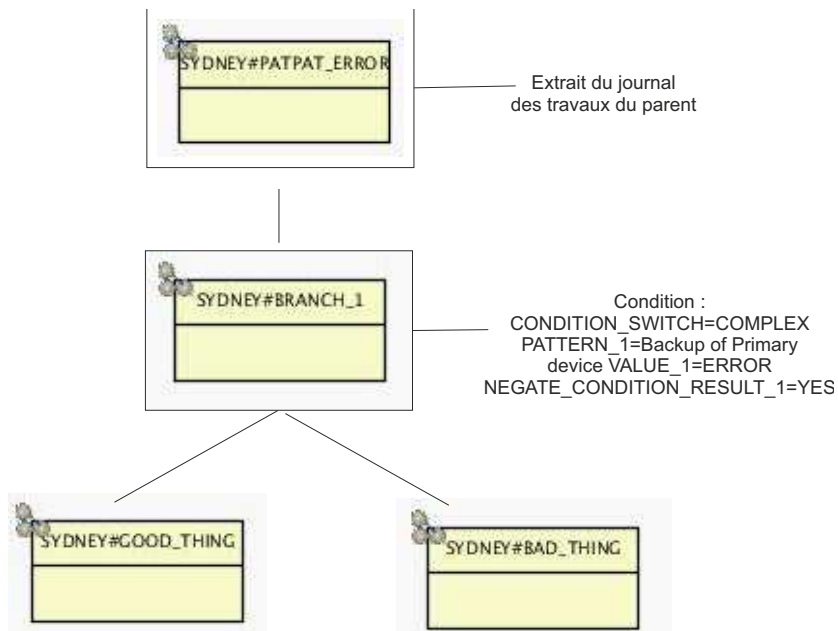


Figure 63. Modèle dans la définition annulée de ligne de modèle

Remarque : Dans *tout* scénario où vous utilisez une condition complexe, vous pouvez également annuler le résultat de *chaque sous-condition particulière*. Vous pouvez également définir des sous-conditions atomiques multiples et annuler uniquement certains d'eux. Pour plus d'informations, voir «Scénario complexe - conditions multiples», à la page 992.

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAT_PAT_NEG
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_PATTERN_ERROR_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Backup of Primary device
VALUE[1]=ERROR
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "Backup of Primary device" in JOBLIST of
PATTERN_PATTERN_ERROR_JOB
Pattern FOUND, performing further tests.
Searching for STRING=ERROR within the
row "Backup of Primary device: ERROR".
String "ERROR" found within the
row "Backup of Primary device: ERROR".
==NEGATED== ATOMIC CONDITION RESULT [1]= FALSE
-----
----- COMPLEX CONDITION -----
[ FALSE ]
CONDITION_RESULT=FALSE
FALSE: The result of complex condition is FALSE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEEA.G_DO_THE_GOOD_THING
%cj SYDNEY#0AAAAAAAAAAAAEEA.G_DO_THE_GOOD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PAT_PAT_NEG[(1941 12/17/07),
(0AAAAAAAAAAAAEEA)].G_DO_THE_GOOD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEEA.B_DO_THE_BAD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEEA.B_DO_THE_BAD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: The result of complex condition is FALSE.
For action CANCEL - RUN_BRANCH=B_DO_THE_BAD_THING and
STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_DO_THE_BAD_THING
CANCELED_JOBS: G_DO_THE_GOOD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 164, à la page 989 affiche les paramètres requis pour le modèle annulé dans le scénario de ligne de modèle.

Tableau 164. Paramètres d'entrée pour le modèle annulé dans le scénario de ligne de modèle

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Sauvegarde d'unité principale
VALUE_1	ERROR
NEGATE_CONDITION_RESULT_1	OUI

La définition de paramètre ressemble à celle de l'exemple suivant. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_1-END
```

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Branche complexe - comparaison d'une valeur numérique

Pour associer la recherche de modèle avec la comparaison d'une valeur numérique.

Modèle dans l'utilisation de branche de ligne de modèle

Ce scénario étend la fonction du scénario de modèle décrit dans Branche complexe - modèle. L'objet de ce scénario est de :

1. Obtenir le journal de travail parent.
2. Dans le journal de travail, identifier la ligne contenant un modèle spécifique.
3. Si la ligne est détectée, rechercher une valeur numérique qu'elle contient.
4. Comparer le numéro avec le numéro qui est indiqué comme paramètre d'entrée, en utilisant l'opérateur arithmétique qui est également indiqué comme paramètre.

L'extrait suivant du journal de travail parent présente un scénario d'utilisation standard :

```
Checking free space...
Free space on volume ABC is 50 %.
```

Vous souhaitez exécuter l'évaluation suivante :

1. Rechercher le modèle Espace disponible sur le volume.
2. Si le texte est détecté, essayez de récupérer une valeur numérique.

3. Comparer la valeur numérique comme suit :

```
If (numeric_value > 30)
Then CONDITION=TRUE
Else CONDITION=FALSE
```

Le nombre 30 et l'opérateur > sont des paramètres transmis au travail de branche. Vous pouvez utiliser tous les opérateurs arithmétiques (par exemple, <, <=, >). Pour une description sur la spécification des paramètres du travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

La figure 64 affiche la définition du scénario de comparaison d'une valeur numérique. Vous pouvez également inverser la logique d'évaluation, bien que généralement cela ne soit pas nécessaire car vous pouvez atteindre le résultat de condition annulé à l'aide de l'opérateur opposé.

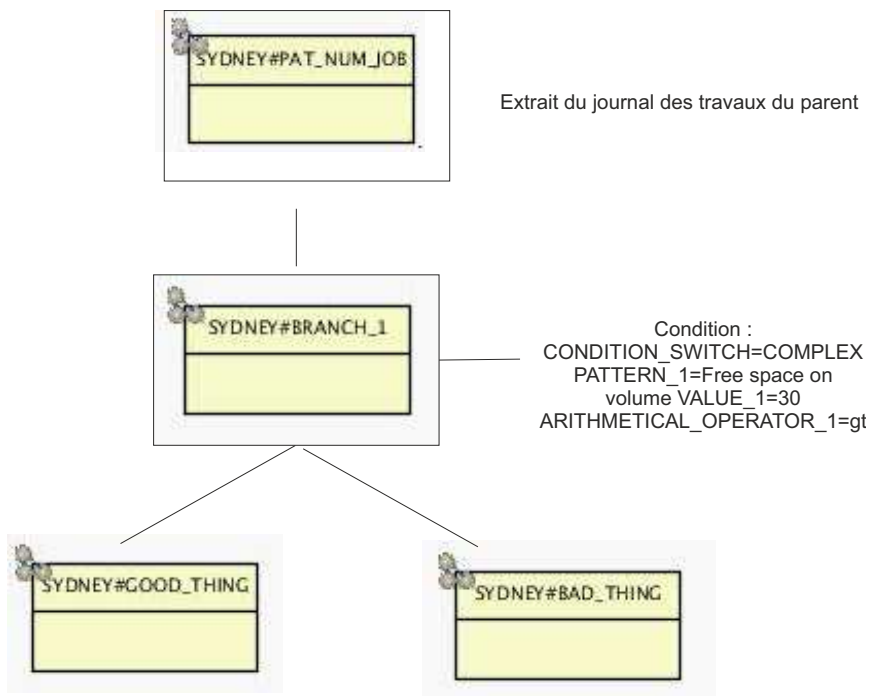


Figure 64. Définition de branche de comparaison numérique

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```
===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PATTERN_NUM
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=PATTERN_NUMBER_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=1
PATTERN[1]=Free space on volume
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[1]=30
ARITHMETICAL_OPERATOR[1]=-gt
NEGATE_CONDITION_RESULT[1]=NO
=====
```

```

===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "Free space on volume" in JOBLIST of PATTERN_NUMBER_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=50. Evaluating arithmetical expression [ 50 -gt 30 ]
Arithmetical expression [ 50 -gt 30 ] evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-----
----- COMPLEX CONDITION -----
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEET.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAEET.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PATTERN_NUM[(2113 12/17/07),
(0AAAAAAAAAAAAEET)].B_DO_THE_BAD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAEET.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAEET.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 165 affiche les paramètres requis pour le modèle dans le scénario de ligne de modèle.

Tableau 165. Paramètres d'entrée pour le scénario de comparaison numérique

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Free space on volume
VALUE_1	30
ARITHMETICAL_OPERATOR	-gt

Il est important de comprendre l'ordre utilisé pour transmettre les nombres à l'expression arithmétique :

number_from_the_joblist compared_against number_supplied_as_parameter

compared_against est l'opérateur arithmétique spécifié comme paramètre. Si vous n'indiquez pas d'opérateur, la valeur par défaut -eq (égal) est utilisée.

L'exemple suivant illustre la définition de paramètre. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```
BRANCH_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
PATTERN_1=Free space on volume  
VALUE_1=30  
ARITHMETICAL_OPERATOR_1=-gt  
BRANCH_1-END
```

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

De plus, il est recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom classique pour le premier travail de branche dans un flot de travaux est BRANCH_1.

Scénario complexe - conditions multiples

Utiliser le scénario de conditions multiples pour définir plusieurs sous-conditions en même temps.

Utilisation de condition complexe

Cette section décrit comment insérer tous les éléments atomiques d'une condition complexe. Le travail de branche obtient le journal de travail parent et exécute une condition complexe sur cette ressource :

- Le journal de travail *ne doit pas* inclure le modèle Error.

et

- L'une des conditions atomiques suivantes doit être satisfaite :
 - L'espace disponible sur l'unité principale est connu, et sa valeur est supérieure à 50.
 - L'espace disponible sur l'unité secondaire est connu, et sa valeur est supérieure à 60.

La figure 65, à la page 993 affiche la définition du scénario de branche complexe.

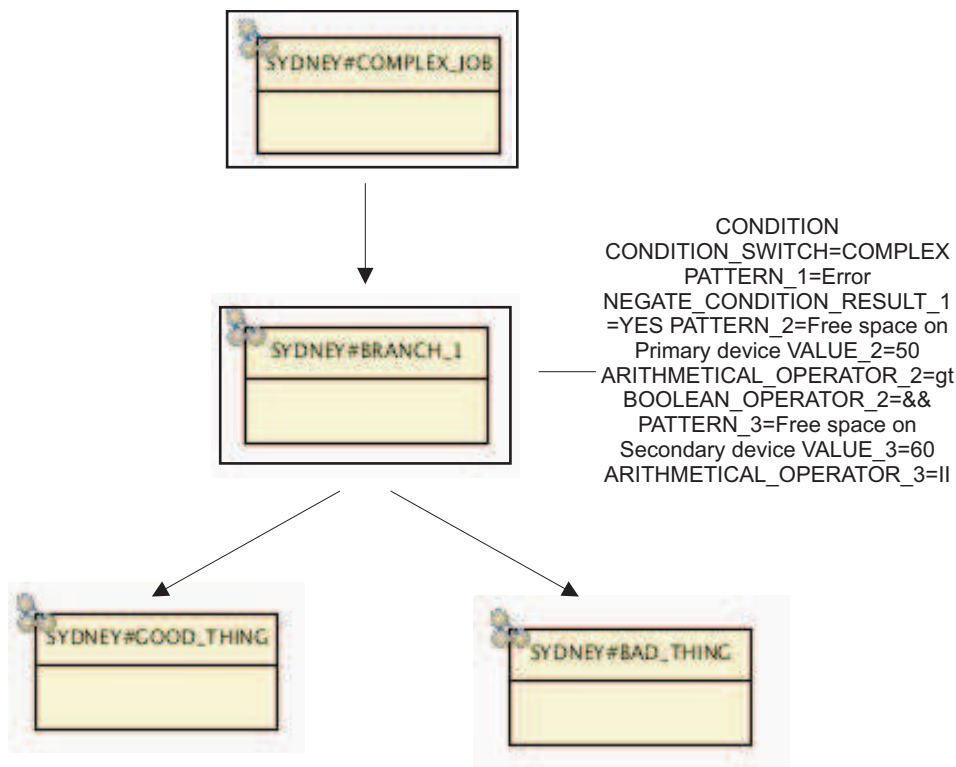


Figure 65. Définition de condition complexe

Le journal de travail affiche la sortie de l'instance de travail de branche générique :

```

===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_COMPLEX
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=COMPLEX_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=CANCEL
CONDITION_COUNT=3
PATTERN[1]=Error
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=YES
PATTERN[2]=Free space on Primary device
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[2]=50
ARITHMETICAL_OPERATOR[2]=--gt
BOOLEAN_OPERATOR[2]=&&
NEGATE_CONDITION_RESULT[2]=NO
PATTERN[3]=Free space on Secondary device
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
VALUE[3]=60
ARITHMETICAL_OPERATOR[3]=--gt
BOOLEAN_OPERATOR[3]=||
NEGATE_CONDITION_RESULT[3]=NO
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "Error" in JOBLIST of COMPLEX_JOB

```

```

Pattern NOT FOUND, condition evaluated as FALSE.
==NEGATED== ATOMIC CONDITION RESULT [1]= TRUE
-----ATOMIC CONDITION 2-----
Searching for "Free space on Primary device" in JOBLLOG of COMPLEX_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=55. Evaluating arithmetical expression [ 55 -gt 50 ]
Arithmetical expression [ 55 -gt 50 ] evaluated as TRUE.
ATOMIC CONDITION RESULT [2]= TRUE
-----ATOMIC CONDITION 3-----
Searching for "Free space on Secondary device" in JOBLLOG of COMPLEX_JOB
Pattern FOUND, performing further tests.
Searching for NUMBER withing row...
Number found=10. Evaluating arithmetical expression [ 10 -gt 60 ]
Arithmetical expression [ 10 -gt 60 ] evaluated as FALSE.
ATOMIC CONDITION RESULT [3]= FALSE
-----
----- COMPLEX CONDITION -----
[ TRUE ] && [ TRUE ] || [ FALSE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAAEW.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAAAAAAAEW.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_COMPLEX[(2219 12/17/07)
(0AAAAAAAAAAAAAEW)].B_DO_THE_BAD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAAAAAAAEW.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAAAAAAAEW.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
TRUE: The result of complex condition is TRUE.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 166 affiche les paramètres requis pour le scénario de branche complexe.

Tableau 166. Paramètres d'entrée pour le scénario de condition complexe

Nom du paramètre	Valeur du paramètre
CONDITION_SWITCH	COMPLEXE
PATTERN_1	Error
NEGATE_CONDITION_RESULT_1	OUI
PATTERN_2	Free space on Primary device
VALUE_2	50
ARITHMETICAL_OPERATOR_2	-gt
BOOLEAN_OPERATOR_2	&&
PATTERN_3	Free space on Secondary device
VALUE_3	60

Tableau 166. Paramètres d'entrée pour le scénario de condition complexe (suite)

Nom du paramètre	Valeur du paramètre
ARITHMETICAL_OPERATOR_3	-gt
BOOLEAN_OPERATOR_3	

L'exemple suivant illustre la définition de paramètre. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
```

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Il est également recommandé de renommer le travail de branche avec un suffixe constitué du caractère de soulignement et d'une valeur numérique. Un nom typique du premier travail de branche dans un flot de travaux est BRANCH_1.

Paramètres de chaîne supplémentaires

Le travail générique de branche exécute la recherche de modèle à l'aide de la commande grep qui accepte plusieurs paramètres d'entrée. Pour affiner la recherche de modèle pour le travail générique de branche, vous pouvez utiliser IS_CASE_SENSITIVE_*i* et IS_REGULAR_EXPRESSION_*i*.

Utilisez les paramètres comme suit :

IS_CASE_SENSITIVE_*i*

Pour activer ou désactiver la distinction entre majuscules et minuscules dans une recherche. La valeur par défaut est YES.

IS_REGULAR_EXPRESSION_*i*

Pour activer ou désactiver une recherche en fonction d'expressions régulières. Par défaut, la valeur est NO.

Remarque : Le suffixe *i* est l'index de la sous-condition particulière.

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Scénarios basés sur le type d'action

Utilisez un travail générique de branche en fonction du type d'action pour indiquer les critères qui déterminent la branche d'exécution et la branche d'arrêt.

Vous pouvez effectuer les actions suivantes :

- Branche d'arrêt
 - CANCEL : la branche d'arrêt est annulée. Il s'agit de l'action la plus fréquente sur la branche d'arrêt.
 - PAUSE : le premier travail de la branche d'arrêt est mis en pause (HOLD). Pour plus de détails, voir Scénario d'actions de pause/libération
- Branche d'exécution
 - Aucune action : la branche d'exécution s'exécute. Il s'agit de l'action la plus fréquente sur la branche d'exécution.
 - Libération : si le premier travail de la branche d'exécution mis en pause (HOLD), augmenter sa priorité (RELEASE). Pour plus de détails, voir Scénario d'actions de pause/libération
- Action spéciale
 - SIGNAL : cette action n'a pas d'effet sur les branches. Elle recommande la confirmation de l'opérateur IBM Workload Scheduler à appliquer. Pour plus de détails, voir Scénario d'action de signal.

Scénario d'actions de pause et de libération

Utilisez ce travail de branche pour gérer un flux de travaux qui est sensible à des résultats de travaux importants, par exemple, pour déterminer quand l'action réalisée par un travail important n'a pas abouti.

Utilisation d'actions de pause et de libération

Utilisez le scénario de pause/libération lorsque, même si le travail de branche a identifié un statut d'erreur, vous ne souhaitez pas annuler les branches. Au lieu d'opter pour l'annulation automatique, vous définissez une séquence d'actions correctives pour l'exécution du travail. Si les actions réussissent, le travail continue comme si l'erreur ne s'était pas produite.

Le flux de processus est similaire à celui présenté dans l'exemple suivant :

1. Vous avez un travail important suivi du premier travail de branche.
2. Le travail de branche est suivi par une bonne branche et une mauvaise branche : la bonne branche (nommé OKbranch) comprend les travaux à exécuter si tout réussit et la mauvaise branche (nommé Correctivebranch) comprend une séquence de tâches destinée à exécuter des actions correctives.
3. Le premier travail de branche évalue la condition exécutée sur le travail parent (votre *travail important*) : si `CONDITION=TRUE`, tout réussit et la branche d'arrêt est annulée. Tous les travaux de la branche corrective sont également annulés, car aucune action corrective n'est nécessaire.
Si `CONDITION=FALSE`, la branche d'arrêt *n'est pas* annulée mais *mise en pause*, ce qui signifie que l'enfant correct (premier travail de la branche d'arrêt) est mis en pause. Dans ce cas, OKbranch est suspendu.
4. Lorsque l'OKbranch est mis en pause, la branche corrective lance les actions correctives.
5. A l'issue de la séquence d'actions correctives, le deuxième travail de branche (placé dans la branche corrective) est soumis pour évaluer le résultat des actions correctives.

Si les corrections réussissent, l'OKbranch est libéré ; si les corrections échouent, l'OKbranch est annulé et le flot de travaux continue d'exécuter la branche incorrecte du deuxième travail de branche.

Généralement, la branche incorrecte du deuxième travail de branche contient un seul travail ABEND (un travail qui exécute une commande *exit 1*). Nous vous conseillons d'exécuter la branche incorrecte de la branche corrective avec le travail ABEND, car cela garantit que le flot de travaux entier s'arrête de manière anormale (les travaux abandonnés précédents possèdent Recovery Option=CONTINUE, ce qui signifie qu'ils n'ont pas propagé l'état ABEND au statut du flot de travaux final).

Remarque : Les deux travaux de branche doivent indiquer le même enfant correct. C'est absolument crucial pour le fonctionnement du processus.

La figure 66, à la page 998 affiche la définition du flot de travaux pour le scénario pause/libération. Le paramètre ACTION_SWITCH=PAUSE est défini pour le premier travail de branche. Aucun paramètre n'est défini pour la deuxième travail de branche.

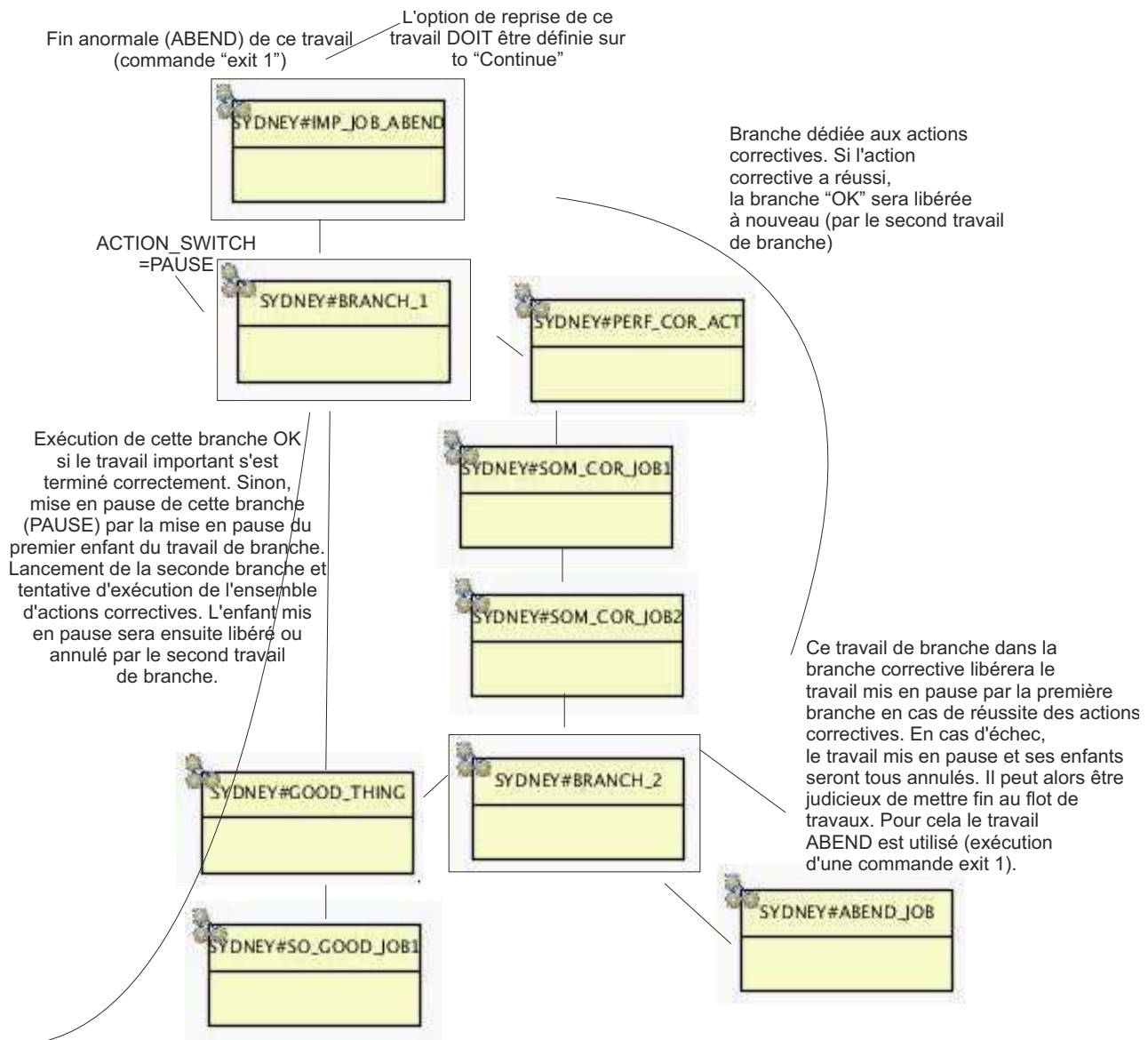


Figure 66. Définition d'actions de pause et de libération

Le journal de travail affiche la sortie de l'instance du premier travail de branche :

```
===== START of branch job BRANCH_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAUSE
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=IMPORTANT_JOB_ABEND
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=PAUSE
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
=====
===== Action on STOP Branch =====
```

```

Performing action PAUSE on job SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING
%altpri SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING;schedid;0;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
  (0AAAAAAAAAAAAEEX)].G_DO_THE_GOOD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job
SYDNEY#0AAAAAAAAAAAAEEX.B_PERFORM_CORRECTIVE_ACTIONS
Releasing of job SYDNEY#0AAAAAAAAAAAAEEX.B_PERFORM_CORRECTIVE_ACTIONS
is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
FALSE: Searched for SUCC parent.
Status of PARENT JOB(IMPORTANT_JOB_ABEND) is ABEND.
For action PAUSE - RUN_BRANCH=B_PERFORM_CORRECTIVE_ACTIONS
and STOP_BRANCH=G_DO_THE_GOOD_THING
BRANCH selected to STOP: G_DO_THE_GOOD_THING
BRANCH selected to CONTINUE: B_PERFORM_CORRECTIVE_ACTIONS
CANCELED_JOBS:
PAUSED_JOB: G_DO_THE_GOOD_THING
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Le journal de travail affiche la sortie de l'instance du second travail de branche :

```

===== START of branch job BRANCH_2 =====
=====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_PAUSE
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_2
PARENT=SOME_CORRECTIVE_JOB_2
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of
PARENT JOB(SOME_CORRECTIVE_JOB_2) is SUCC.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAAAAAAEEX.B_ABEND_JOB
%cj SYDNEY#0AAAAAAAAAAAAEEX.B_ABEND_JOB;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
  (0AAAAAAAAAAAAEEX)].B_ABEND_JOB
=====
===== Action on RUN Branch =====
Performing action RELEASE on job
SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING
Releasing SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING, because priority=0
%altpri SYDNEY#0AAAAAAAAAAAAEEX.G_DO_THE_GOOD_THING;schedid;10;noask
Command forwarded to batchman for SYDNEY#GBJ_PAUSE[(2300 12/17/07),
  (0AAAAAAAAAAAAEEX)].G_DO_THE_GOOD_THING
=====
===== Statistics of branch job BRANCH_2 =====
TRUE: Searched for SUCC parent. Status of
PARENT JOB(SOME_CORRECTIVE_JOB_2) is SUCC.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_ABEND_JOB
BRANCH selected to STOP: B_ABEND_JOB
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_ABEND_JOB

```

```
PAUSED_JOB:
RELEASED_JOB:G_DO_THE_GOOD_THING
===== END of branch job BRANCH_2 =====
```

Paramètres d'entrée obligatoires

Le tableau 167 affiche les paramètres requis pour le premier travail de branche du scénario de pause/libération. Le second travail de branche ne gère pas de paramètre.

Tableau 167. Paramètres d'entrée pour le scénario de pause/libération

Nom du paramètre	Valeur du paramètre
ACTION_SWITCH	PAUSE

Remarque :

- Le parameter ACTION_SWITCH=PAUSE est requis uniquement pour le premier travail de branche dans le flot de travaux. Le second travail de branche doit avoir la valeur ACTION SWITCH=CANCEL (par défaut).
- Les deux travaux de branche doivent indiquer le même enfant correct.
- Chaque travail de branche doit avoir un suffixe différent. Par exemple, dans ce scénario simple de pause et de libération, les deux noms de travail de branche sont utilisés : BRANCH_1 et BRANCH_2.

La définition de paramètre ressemble à celle de l'exemple suivant. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

```
BRANCH_1-BEGIN
ACTION_SWITCH=PAUSE
BRANCH_1-END
```

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement du travail de branche dans le flot de travaux

Insérez le travail de branche générique dans le flot de travaux après le travail parent et renommez l'enfant correct avec le préfixe "G_" et l'enfant incorrect avec le préfixe "B_".

Le premier travail de branche détermine l'OKbranch (suivi de l'enfant correct) et le CorrectiveBranch (suivi de l'enfant incorrect). Le travail représentant l'enfant correct doit posséder le préfixe « G_ », tandis que le travail représentant l'enfant incorrect doit avoir le préfixe « B_ ».

Les deux travaux de branche doivent indiquer le même enfant correct. Cela signifie que le bon enfant du premier travail de branche doit être identique au bon enfant du deuxième travail de branche.

La meilleure pratique est de définir le travail ABEND comme l'enfant incorrect du second travail de branche. Le travail ABEND appelle simplement la commande système *exit 1*, qui a pour effet de mettre fin au travail de façon anormale.

Le fait d'avoir un travail ABEND dans la branche incorrecte assure la propagation de l'état ABEND s'effectue également au niveau du flot de travaux. Aucun travail terminé de façon anormale précédemment ne propagerait l'état ABEND au niveau du flot de travaux si l'option ABEND au niveau de le flot de travaux si l'option de

reprise est réglée sur Continuer. Pour permettre l'exécution du travail de branche, vous devez régler l'option de reprise sur Continuer pour tous les parents du travail de branche.

Scénario d'actions de pause et de libération multiples

Utilisez le scénario d'actions de pause et de libération multiples pour exécuter une séquence d'actions correctives et quitter la branche corrective lorsque l'une d'elles réussit. Lorsqu'une correction aboutit, le travail de branche annule les corrections restantes et libère l'OKbranch.

Utilisation d'actions de pause et de libération multiples

La figure 67, à la page 1002 affiche la définition de flot de travaux.

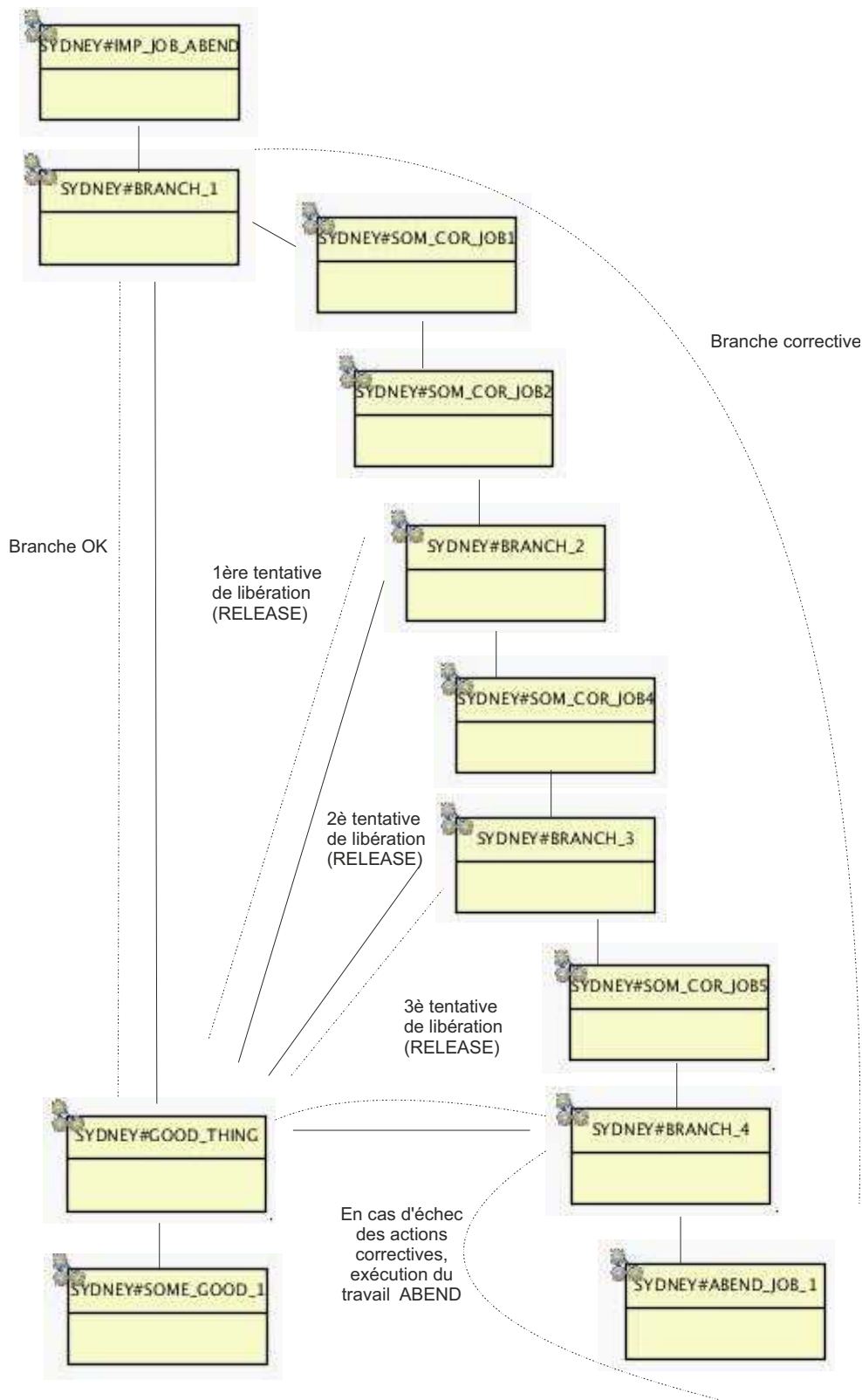


Figure 67. Définition de scénario d'actions de pause et de libération multiples

Vous devez définir le paramètre ACTION_SWITCH=PAUSE pour *tous* les travaux de branche à l'exception du dernier. Par conséquent, dans ce scénario, les travaux de branche BRANCH_1, BRANCH_2 et BRANCH_3 doivent avoir

ACTION_SWITCH=PAUSE. Si vous ne définissez pas ce paramètre, l'enfant approprié est annulé par un travail médian de branche.

Tous les travaux de branche doivent pointer sur le même enfant approprié.

Vérifiez que chaque travail de branche a un suffixe différent. Par exemple, dans ce scénario, les noms BRANCH_1, BRANCH_2, BRANCH_3 et BRANCH_4 sont utilisés.

La meilleure pratique est de définir le travail ABEND comme l'enfant incorrect du dernier travail de branche. Le travail ABEND appelle la commande système exit 1. Cela provoque la fin anormale du travail (TO END ABNORMALLY) et son statut est répercuté au niveau de le flot de travaux.

Scénario d'action de signal

Utilisez ce scénario pour disposer d'un travail de signal qui traite et stocke les informations utiles à la prise de décision dans le journal du travail.

Utilisation de scénario d'action de signal

D'un point de vue logique, le travail de signal et le travail de branche sont différents dans leur étape dernière (c'est-à-dire l'action exécutée). Bien que le travail de branche annule toujours, mette en pause ou libère ses travaux enfant, le travail de signal enregistre uniquement une recommandation pour vous permettre de prendre une décision. Au lieu de bloquer le processus, le scénario de signal sélectionne la branche d'exécution et permet au flot de travaux de continuer. Ce scénario étend l'approche déjà disponible avec les invites de IBM Workload Scheduler ; il représente la combinaison d'invites avec les droits d'accès du travail de branche.

Dans ce scénario, il existe deux travaux dans l'ordre séquentiel suivant :

1. Travail de signal
2. Travail de branche

Pour le travail de branche :

1. Indiquez le paramètre ACTION_SWITCH=SIGNAL et définissez l'option de reprise sur CONTINUE. Dans la définition du flot de travaux, définissez l'indicateur Confirmation requise.

Le travail de signal exécute la logique d'évaluation, ce qui signifie qu'il évalue la condition sur les propriétés du parent, mais n'annule pas et ne met pas en pause *aucun* de ses travaux enfant.

L'indicateur Confirmation requise provoque l'arrêt du flot de travaux par le travail de signal. Une fois terminé, le travail de signal reste à l'état PEND. Le journal des travaux du signal présente l'évaluation complète des conditions, y compris la recommandation de confirmation. Cela signifie que le travail de signal évalue la condition et écrit la recommandation (Confirm SUCC ou Confirm ABEND) dans le journal de travail.

2. Consultez le journal du travail du signal et choisissez de confirmer le travail avec un statut SUCC ou ABEND.

Le travail de branche suivant démarre uniquement lorsque vous confirmez le travail de signal. Le travail de branche évalue le statut défini pour le travail de signal et détermine la branche d'exécution et la branche d'arrêt.

Cette dernière étape représente le branchement simple, qui est décrit dans «Scénario de branche simple», à la page 967.

La figure 68 affiche la définition du flot de travaux pour le scénario de signal.

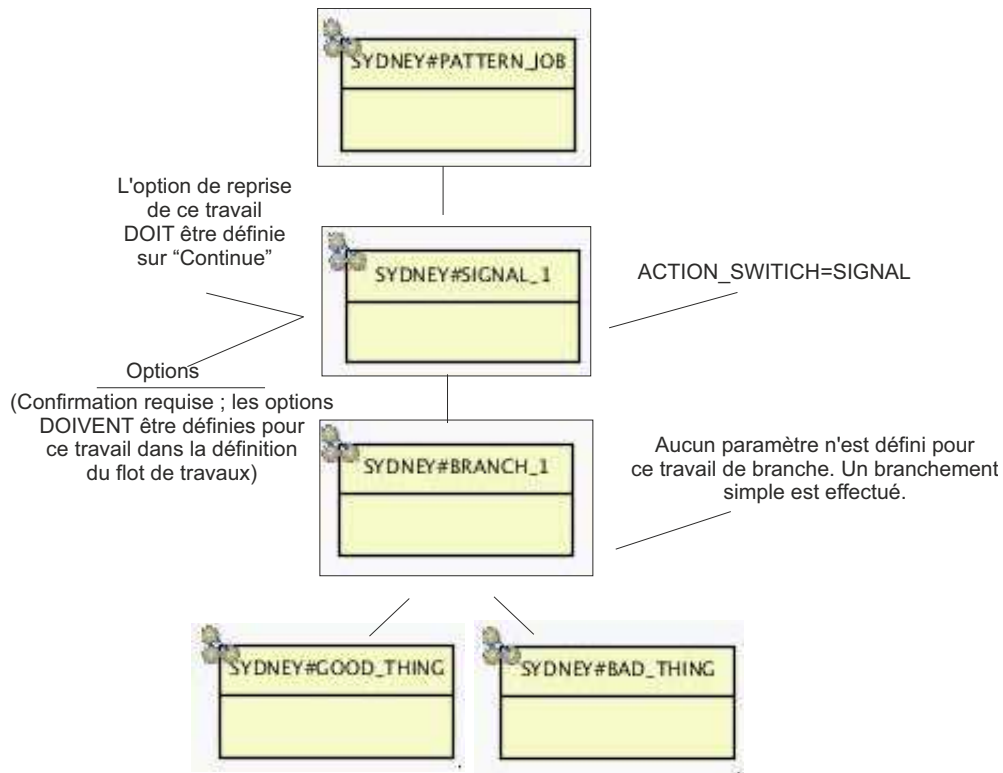


Figure 68. Définition d'action de signal

Le journal du premier travail de branche indique que le travail est mis en attente (HELD), car il est à l'état PEND. Ce statut nécessite votre confirmation, sinon les successeurs du travail ne s'exécutent pas.

```

===== START of branch job SIGNAL_1 =====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_SIGNAL
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=SIGNAL_1
PARENT=PATTERN_JOB
=====
===== Input parameters =====
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=SIGNAL
CONDITION_COUNT=1
PATTERN[1]=completed successfully
IS_CASE_SENSITIVE[1]=YES
IS_REGULAR_EXPRESSION[1]=NO
NEGATE_CONDITION_RESULT[1]=NO
=====
===== MAIN DECISION MAKING =====
COMPLEX condition evaluation
-----ATOMIC CONDITION 1-----
Searching for "completed successfully" in JOBLOG of PATTERN_JOB
Pattern FOUND, performing further tests.
No additional value defined for specified pattern. Condition evaluated as TRUE.
ATOMIC CONDITION RESULT [1]= TRUE
-----

```

```

----- COMPLEX CONDITION -----
[ TRUE ]
CONDITION_RESULT=TRUE
TRUE: The result of complex condition is TRUE.
=====
===== Statistics of branch job SIGNAL_1 =====
*****Recommended confirmation for this job is SUCC.*****
===== END of branch job SIGNAL_1 =====

```

The log of the second branch job shows the processing of the following branch job instance.

```

===== START of branch job BRANCH_1 =====
=====
===== Job environment =====
MASTER_PLATFORM=UNIX
STREAM_NAME=GBJ_SIGNAL
STREAM_CPU=SYDNEY
BRANCH_JOB_NAME=BRANCH_1
PARENT=SIGNAL_1
=====
===== Input parameters =====
CONDITION_SWITCH=PARENT_SUCCESS
ACTION_SWITCH=CANCEL
=====
===== MAIN DECISION MAKING =====
Evaluation dependent on PARENT_SUCCESS
TRUE: Searched for SUCC parent. Status of PARENT JOB(SIGNAL_1) is SUCC.
=====
===== Action on STOP Branch =====
Performing action CANCEL on job SYDNEY#0AAAAAAAIAE16.B_DO_THE_BAD_THING
%cj SYDNEY#0AAAAAAAIAE16.B_DO_THE_BAD_THING;schedid;noask
Command forwarded to batchman for SYDNEY#GBJ_SIGNAL[(0012 12/21/07),
(0AAAAAAAIAE16)].B_DO_THE_BAD_THING
=====
===== Action on RUN Branch =====
Performing action RELEASE on job SYDNEY#0AAAAAAAIAE16.G_DO_THE_GOOD_THING
Releasing of job SYDNEY#0AAAAAAAIAE16.G_DO_THE_GOOD_THING is NOT NECESSARY,
because priority=10
=====
===== Statistics of branch job BRANCH_1 =====
TRUE: Searched for SUCC parent. Status of PARENT JOB(SIGNAL_1) is SUCC.
For action CANCEL - RUN_BRANCH=G_DO_THE_GOOD_THING and
STOP_BRANCH=B_DO_THE_BAD_THING
BRANCH selected to STOP: B_DO_THE_BAD_THING
BRANCH selected to CONTINUE: G_DO_THE_GOOD_THING
CANCELED_JOBS: B_DO_THE_BAD_THING
PAUSED_JOB:
RELEASED_JOB:
===== END of branch job BRANCH_1 =====

```

Paramètres d'entrée obligatoires

Le tableau 168 affiche les paramètres requis pour le travail de signal. Le travail de branche suivant ne gère pas de paramètre.

Tableau 168. Paramètres d'entrée pour le scénario d'action de signal

Nom du paramètre	Valeur du paramètre
ACTION_SWITCH	SIGNAL

La définition de paramètre ressemble à celle de l'exemple suivant. Le texte est saisi dans la zone de commentaires de la définition du flot de travaux.

Remarque : Cette définition indique également certains paramètres qui sont requis *uniquement* lorsque vous utilisez le travail de signal pour une recherche de modèle dans le journal de travail parent, ce qui constitue le principal avantage du scénario de signal. Vous l'utilisez pour exécuter l'analyse syntaxique complexe des résultats du travail, enregistrer l'heure et prendre une décision en fonction des informations collectées.

```
SIGNAL_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
PATTERN_1=completed successfully  
ACTION_SWITCH=SIGAL  
SIGNAL_1-END
```

Pour une description sur la manière de spécifier les paramètres pour un travail de branche, voir «Indication des paramètres de travail de branche», à la page 1010.

Placement des travaux de branche dans le flot de travaux

Insérez le travail de signal après que le travail est évalué. Attribuez au travail de signal un nom incluant la chaîne SIGNAL et une valeur numérique (dans ce scénario, SIGNAL_1 est utilisé).

Le travail de branche suivant est le successeur immédiat du travail de signal. Attribuez-lui un nom avec le suffixe approprié : si c'est le premier travail de branche dans le flot de travaux, le nom peut être BRANCH_1.

Les travaux enfant du travail de branche doivent être nommés selon les instructions communes :

- Le nom de l'enfant correct doit commencer par "G_".
- Le nom de l'enfant incorrect doit commencer par "B_".

Utilisation du travail de branche

Vous pouvez utiliser le travail générique de branche dans votre environnement IBM Workload Scheduler.

Pour comprendre comment utiliser les travaux génériques de branche, voir :

- «Configuration requise pour exécuter des travaux de branche», à la page 617
- «Définition du travail de branche et du travail de signal dans la base de données», à la page 1007
- «Placement du travail de branche dans le flot de travaux», à la page 1009
- «Utilisation du travail ABEND», à la page 1010

Configuration requise pour exécuter des travaux de branche

Pour exécuter un travail de branche générique, vérifiez que la configuration système suivante est respectée.

Sur les systèmes d'exploitation Windows :

- Etant donné que les systèmes d'exploitation Windows ne peuvent pas, en natif, interpréter les scripts de shell UNIX, vous devez installer un interpréteur de shell pour utiliser le script de shell *rép_principale_TWA\TWS\bin\branch.sh* sur le gestionnaire de domaine maître Windows.
- Le répertoire C:\cygwin\bin doit pointer vers le sous-répertoire bin de votre répertoire d'installation Cygwin. Si vous avez installé Cygwin dans un répertoire différent du répertoire par défaut, utilisez le chemin correspondant.

Sur les systèmes d'exploitation UNIX, seuls les interpréteurs de commandes Bash ou Korn sont pris en charge.

Si votre poste de travail UNIX utilise un interpréteur de commandes Bourne, ouvrez le fichier **branch.sh** situé dans *rép_principale_TWA/TWS/bin* sur le gestionnaire de domaine maître et modifiez `#!/bin/sh` de manière à pointer sur un interpréteur de commandes Bash (`#!/bin/bash`) ou Korn (`#!/bin/ksh`).

Définition du travail de branche et du travail de signal dans la base de données

Pour définir des travaux de signal et de branche dans la base de données IBM Workload Scheduler, vous pouvez utiliser Dynamic Workload Console ou le Composer.

Définition de deux travaux pointant vers un script de shell

Pour exécuter un scénario de travail de branche, vous devez définir un travail de branche. Pour exécuter un scénario de signal, vous devez également définir un travail de signal. Les deux travaux indiquent le même fichier de script de shell à la différence que :

- Pour le travail de branche, vous définissez l'option de reprise sur STOP.
- Pour le travail de signal, vous définissez l'option de reprise sur CONTINUER.

Vous avez deux travaux différents qui pointent vers le même fichier de script de shell afin d'éviter le placement incorrect du travail de branche ou du travail de signal dans le flot de travaux. Le travail de branche vérifie s'il est correctement placé dans le flot de travaux ; sinon, il se termine de façon anormale. Pour empêcher les successeurs de s'exécuter, le travail de branche a l'option de reprise définie sur STOP.

Le travail de signal exécute la même vérification ; s'il est mal placé dans le flot de travaux, le travail renvoie un message d'erreur et s'arrête avec un code retour différent de zéro. Même si l'option de reprise est définie sur Continuer, le travail ne libère pas ses successeurs de la dépendance car il a l'indicateur Confirmation requise défini dans la définition de flot de travaux. Le travail de signal reste, par conséquent, à l'état PEND en attendant que vous vérifiez son journal des travaux.

Pour plus d'informations, voir «Scénario d'action de signal», à la page 1003.

Définition du travail de branche à l'aide de Dynamic Workload Console

Pour définir un travail de branche dans la base de données IBM Workload Scheduler à l'aide de Dynamic Workload Console, procédez comme suit :

1. Ouvrez Workload Designer.
2. Cliquez sur **Nouveau -> Définition de travail -> Natif** et, selon votre système d'exploitation, **UNIX** ou **Windows**.
3. Dans le panneau Général, entrez des valeurs dans les zones suivantes :
 - Nom : **BRANCHE**
 - Poste de travail : votre gestionnaire de domaine maître
 - Connexion : nom de l'utilisateur qui soumet et exécute le travail sur le gestionnaire de domaine maître
4. Dans le panneau des tâches :

- Cochez le bouton radio **Script**.
- Selon votre système d'exploitation :

UNIX Dans la zone de tâche, indiquez le chemin complet vers le script de shell `branch.sh`. Utilisez les barres obliques (/).

Windows

Dans la zone de tâche, indiquez deux chemins :

- Le premier chemin pointe vers Cygwin : indiquez le chemin complet vers l'exécutable `bash` de Cygwin. Utilisez la notation standard Windows, c'est-à-dire les barres obliques inversées (\) comme séparateurs de répertoire.
- Le second chemin pointe vers le script de shell : indiquez le chemin complet vers le script de shell `branch.sh`. Utilisez les barres obliques (/). Si le chemin contient des espaces, insérez-le entre guillemets.

Par exemple, la valeur de la zone tâche peut être :

```
c:\cygwin\bin\bash\ "rép_principale_TWA\TWS\bin\branch.sh"
```

5. Sur le panneau des options de reprise, définissez l'action sur **Stop**.

Remarque : Vous pouvez utiliser les paramètres IBM Workload Scheduler comme vous le feriez dans une autre définition de travail.

Définition du travail de signal à l'aide de Dynamic Workload Console

Pour définir un travail de signal dans la base de données IBM Workload Scheduler à l'aide de Dynamic Workload Console, procédez comme suit :

1. Ouvrez Workload Designer.
2. Cliquez sur **Nouveau -> Définition de travail -> Natif** et, selon votre système d'exploitation, **UNIX** ou **Windows**.
3. Dans le panneau Général, entrez des valeurs dans les zones suivantes :
 - Nom : **SIGNAL**
 - Poste de travail : votre gestionnaire de domaine maître
 - Connexion : nom de l'utilisateur qui soumet et exécute le travail sur le gestionnaire de domaine maître
4. Dans le panneau des tâches :
 - Cochez le bouton radio **Script**.
 - Selon votre système d'exploitation :

UNIX Dans la zone de tâche, indiquez le chemin complet vers le script de shell `branch.sh`. Utilisez les barres obliques (/).

Windows

Dans la zone de tâche, indiquez deux chemins :

- Le premier chemin pointe vers Cygwin : indiquez le chemin complet vers l'exécutable `bash` de Cygwin. Utilisez la notation standard Windows, c'est-à-dire les barres obliques inversées (\) comme séparateurs de répertoire.
- Le second chemin pointe vers le script de shell : indiquez le chemin complet vers le script de shell `branch.sh`. Utilisez les barres obliques (/). Si le chemin contient des espaces, insérez-le entre guillemets.

Par exemple, la valeur de la zone tâche peut être :

```
c:\cygwin\bin\bash\ "rép_principale_TWA\TWS\bin\branch.sh"
```

5. Sur le panneau des options de reprise, définissez l'action sur **Continuer**.

Définition du travail de branche et du travail de signal à l'aide du Composer

Pour définir un travail de branche et un travail de signal dans la base de données IBM Workload Scheduler à partir du Composer, procédez comme suit :

1. Connectez-vous au gestionnaire de domaine maître en tant qu'utilisateur disposant des droits permettant d'ajouter des travaux.
2. Suivant votre système d'exploitation, créez une définition de travail comme dans les exemples suivants :

UNIX

```
$JOBS
SYDNEY#BRANCH
SCRIPTNAME "rép_principale_TWA/TWS/bin/branch.sh"
STREAMLOGON tws
TASKTYPE UNIX
RECOVERY STOP

SYDNEY#SIGNAL
SCRIPTNAME "rép_principale_TWA/TWS/bin/branch.sh"
STREAMLOGON tws
TASKTYPE UNIX
RECOVERY CONTINUE
```

Windows

```
$JOBS
HELSINKI#BRANCH
SCRIPTNAME "c:\cygwin\bin\bash\ "rép_principale_TWA\TWS\
bin\branch.sh""
STREAMLOGON tws
TASKTYPE WINDOWS
RECOVERY STOP

$JOBS
HELSINKI#BRANCH
SCRIPTNAME "c:\cygwin\bin\bash\ "rép_principale_TWA\TWS\
bin\branch.sh""
STREAMLOGON tws
TASKTYPE WINDOWS
RECOVERY CONTINUE
```

Placement du travail de branche dans le flot de travaux

Vous pouvez placer le travail générique de branche dans le flot de travaux à l'aide de la console DWC.

Règles générales

Pour placer le travail de branche dans le flot de travaux correctement, vérifiez les éléments suivants :

- Vous disposez d'une seule définition de travail dans la base de données IBM Workload Scheduler, comme décrit dans «Définition du travail de branche et du travail de signal dans la base de données», à la page 1007.
- Lorsque vous insérez le travail de branche dans le flot de travaux, vous lui attribuez un alias décrivant le fichier du travail de branche dans le flot de travaux.

- Nommez le premier travail de branche `BRANCH_1`, le deuxième `BRANCH_2`, et ainsi de suite.

Règles valides pour tous les scénarios à l'exception de l'action de signal

Les règles suivantes s'appliquent à tous les scénarios d'utilisation de travail de branche, à l'exception de l'action `SIGNAL` :

- Le travail de branche doit posséder un seul prédécesseur.
- Le travail de branche doit comporter exactement deux travaux enfant, nommés comme suit :
 - Le nom de l'enfant correct doit commencer par `G_`.
 - Le nom de l'enfant incorrect doit commencer par `B_`.

Règles valides pour le scénario d'action de signal

Les règles suivantes s'appliquent à l'action `SIGNAL` :

- Vous devez utiliser deux travaux contrôlés séquentiellement, comme suit :
 - Travail de signal
 - Travail de branche
- Le travail de signal ne doit posséder qu'un enfant, représentant le travail de branche suivant.
- Pour le travail de signal, vous devez définir le paramètre d'entrée `ACTION_SWITCH=SIGNAL`.
- Pour le travail de signal, vous devez définir l'indicateur Confirmation requise en modifiant les propriétés du travail de branche.
- Le travail de signal doit posséder un seul prédécesseur.
- Le travail de branche doit être conforme aux règles décrites dans «Règles valides pour tous les scénarios à l'exception de l'action de signal ».

Utilisation du travail ABEND

Utilisez le travail `ABEND` lorsque vous souhaitez inclure l'état `ABEND` au statut du flot de travaux.

Si un événement incorrect se produit dans votre flot de travaux et qu'il est détecté par le travail de branche, vous pouvez ainsi propager ce *statut incorrect* au niveau du flot de travaux en insérant le travail `ABEND` dans la branche incorrecte qui suit le travail de branche.

Le travail `ABEND` émet uniquement la commande `exit 1`, qui entraîne la fin anormale du travail et garantit que le statut de fin anormale est répercuté au flot de travaux. Aucun travail terminé précédemment de façon anormale ne propage son état au flot de travaux car son option de reprise est réglée sur `Continuer` (nécessaire pour que le travail de branche s'exécute).

Pour obtenir un exemple de travail `ABEND`, voir «Placement du travail de branche dans le flot de travaux», à la page 1009.

Indication des paramètres de travail de branche

Indiquez les paramètres de travail de branche dans la zone de commentaires du flot de travaux contenant le travail de branche concerné.

Remarque : Veillez à spécifier les paramètres dans la zone de commentaires, et pas la zone de description de l'éditeur de flot de travaux.

Chaque paramètre doit être encadré par un *séparateur de début* et un *séparateur de fin*, même si vous utilisez des paramètres pour un seul travail de branche. Les séparateurs sont construits comme suit :

Séparateur de début

Nom du travail de branche, suivi de la chaîne -BEGIN. Par exemple
BRANCH_1-BEGIN.

Séparateur de fin

Nom du travail de branche, suivi de la chaîne -END, par exemple
BRANCH_1-END.

L'exemple suivant illustre une définition de paramètre correspondant à une branche de travail :

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
```

L'exemple suivant illustre une définition de paramètre correspondant à un flot de travaux avec deux travaux de branche :

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END
BRANCH_2-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_2-END
```

Le nom des travaux de signal doit commencer par la chaîne SIGNAL. Les séparateurs de paramètre sont également requis :

```
SIGNAL_1-BEGIN
ACTION_SWITCH=SIGNAL
SIGNAL_1-END
```

Référence de paramètres

Syntaxe et signification des paramètres de travail de branche.

Types de paramètre

Il existe deux types de paramètre :

Fixe Vous ne pouvez indiquer des paramètres fixes qu'une seule fois dans un travail.

Indexé Vous pouvez spécifier des paramètres indexés plusieurs fois dans un travail, ou ne pas les indiquer du tout.

Paramètres fixes

Vous pouvez indiquer les paramètres fixes suivants :

CONDITION_SWITCH

Type de condition qui s'exécute sur le travail parent. Il peut prendre l'une des valeurs suivantes :

PARENT_SUCCESS

La condition est TRUE lorsque le parent se termine à l'état SUCC. Il s'agit de la valeur par défaut. Pour obtenir une description de cette condition, voir «Scénarios basés sur le type de condition», à la page 967.

PARENT_ABEND

La condition est TRUE lorsque le parent se termine à l'état ABEND. Pour obtenir une description de cette condition, voir «Fin anormale du parent», à la page 976.

COMPLEX

La condition est générée à partir d'une ou plusieurs sous-conditions connectées avec les opérateurs booléens (AND ou OR) et est évaluée avec la logique booléenne. Pour obtenir une description de cette condition, voir «Scénario complexe - conditions multiples», à la page 992.

ACTION_SWITCH

Action à effectuer sur la branche d'arrêt. Il peut prendre l'une des valeurs suivantes :

CANCEL

La branche d'arrêt est annulée. Il s'agit de la valeur par défaut.

PAUSE

La branche d'arrêt est mise en pause. Pour obtenir une description de cette condition, voir «Scénario d'actions de pause et de libération», à la page 996.

SIGNAL

Aucune branche n'est annulée. Une recommandation nécessitant votre confirmation est stockée dans le journal de travail. Pour obtenir une description de cette condition, voir «Scénario d'action de signal», à la page 1003.

Paramètres indexés

Utilisez les paramètres indexés uniquement avec le paramètre `CONDITION_SWITCH=COMPLEX` indiqué (sinon, ils sont ignorés lors du processus de branchement).

Lorsque vous définissez `CONDITION_SWITCH=COMPLEX`, le travail de branche évalue une condition complexe (une condition avec une ou plusieurs sous-conditions). Chaque sous-condition possède son propre *index* qui commence par le numéro 1 et est incrémentiel.

Pour chaque sous-condition, vous devez définir au moins un paramètre `PATTERNi`, où *i* représente la relation entre la sous-condition et le paramètre correspondant. Le suffixe ressemble à *paramètre_indexé_i*, où *i* est le suffixe. Par exemple, les trois paramètres suivants appartiennent à la même sous-condition, qui est la deuxième dans la condition complexe ; leur affinité est exprimée par le suffixe `_2`, qui représente l'index de la sous-condition :

```
PATTERN_2=find this row and number in this row
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-1t
```

Regroupez les paramètres associés à une sous-condition en utilisant le même index. Les sous-conditions sont évaluées séparément, puis connectées les unes aux autres à l'aide des opérateurs booléens. La *condition complexe* est ensuite évaluée.

Exemples d'utilisation d'index

L'exemple suivant présente les paramètres à indiquer si vous souhaitez rechercher trois modèles dans le journal du travail parent. L'index sert de *compteur incrémentiel*. Les paramètres appartiennent à trois sous-conditions distinctes :

```
PATTERN_1=premier modèle à trouver
PATTERN_2=deuxième modèle à trouver
PATTERN_3=troisième modèle à trouver
```

Utilisez la syntaxe suivante pour définir les paramètres liés à même sous-condition (par exemple, recherche d'un modèle, puis d'un numéro sur la même ligne et la comparaison arithmétique ultérieure). Vous indiquez le numéro et l'opérateur arithmétique comme paramètres d'entrée.

```
PATTERN_1=Free space on Primary device
VALUE_1=50
ARITHMETICAL_OPERATOR_1=-gt
```

Signification des paramètres indexés

la liste suivante décrit les paramètres indexés et leurs valeurs possibles. Vous pouvez utiliser tous ces paramètres pour créer une seule sous-condition.

PATTERN_i

Recherche d'un modèle de texte (par exemple, ended successfully). Si le modèle est trouvé, le résultat de la condition est TRUE.

Pour le paramètre `PATTERNi`, vous pouvez indiquer les paramètres supplémentaires suivants. Si `PATTERNi` n'est pas spécifié, ils sont ignorés.

VALUE_i

STRING ou NUMERIC. Cela est déterminé automatiquement en lisant le paramètre de valeur particulière lors du démarrage du travail de branche.

La valeur spécifiée par `VALUEi` est recherchée *dans la même ligne* identifiée par la recherche de la chaîne indiquée par `PATTERNi`.

Deux types de valeurs sont possibles et sont déterminés automatiquement lors de l'analyse syntaxique du contenu du paramètre `VALUEi` :

Valeur de chaîne

Recherche d'un autre modèle de texte dans la même ligne.

Si les deux modèles sont présents dans la même ligne, le résultat de la condition est TRUE.

Pour obtenir une description de cette fonction, voir «Branche complexe - modèle dans la ligne de modèle», à la page 984.

Valeur numérique

Recherches de la valeur numérique dans la même ligne.

L'opérateur arithmétique que vous avez indiqué est ensuite utilisé pour exécuter la comparaison arithmétique. Si la comparaison arithmétique réussie, la condition est TRUE. Un opérateur arithmétique spécifique est définie pour chaque valeur numérique.

Pour obtenir une description de cette fonction, voir «Branche complexe - comparaison d'une valeur numérique», à la page 989.

ARITHMETICAL_OPERATOR_ *i*

Opérateur utilisé pour la comparaison arithmétique.

NEGATE_CONDITION_RESULT_ *i*

Cet argument annule le résultat de la sous-condition particulière, ce qui signifie qu'il permute le résultat TRUE ou FALSE de la sous-condition.

BOOLEAN_OPERATOR_ *i*

Les sous-conditions définies sont liées les unes aux autres par l'opérateur booléen AND ou OR. Vous pouvez utiliser l'opérateur booléen car *i*=2. Cela signifie que l'index de l'opérateur booléen doit être d'au moins 2.

Par exemple, vous avez deux paramètres dans la liste. Chacun d'eux représente une sous-condition. Chaque sous-condition est évaluée séparément et le résultat est renvoyé comme TRUE ou FALSE. Pour évaluer la condition entière, vous devez joindre les résultats particuliers ensemble.

La signification de BOOLEAN_OPERATOR_ *i* est que le résultat de la connexion de *cette* sous-condition avec le résultat de la sous-condition *précédente* utilise l'opérateur booléen AND ou OR.

Tables de référence

Le tableau 169 décrit les paramètres, leurs valeurs possibles, et les valeurs par défaut.

Tableau 169. Paramètres et valeurs

Nom du paramètre	Valeurs possibles	Valeur par défaut
CONDITION_SWITCH	PARENT_SUCCESS	PARENT_SUCCESS
	PARENT_ABEND	
	COMPLEXE	

Tableau 169. Paramètres et valeurs (suite)

Nom du paramètre	Valeurs possibles	Valeur par défaut
ACTION_SWITCH	CANCEL PAUSE SIGNAL	CANCEL (pour les travaux de branche) SIGNAL (pour les travaux de signal)
PATTERN_ <i>i</i> , où <i>i</i> est l'index incrémentiel	Toute chaîne de caractères	
VALUE_ <i>i</i> , où <i>i</i> est l'index incrémentiel	Toute chaîne de caractères Toute valeur numérique (entier ou réel)	
ARITHMETICAL_OPERATOR_ <i>i</i> , où <i>i</i> est l'index incrémentiel	-lt -le -eq -ne -ge -gt	-eq
IS_CASE_SENSITIVE_ <i>i</i> , où <i>i</i> est l'index incrémentiel	OUI NON	OUI
IS_REGULAR_EXPRESSION_ <i>i</i> , où <i>i</i> est l'index incrémentiel	OUI NON	NO
NEGATE_CONDITION_RESULT_ <i>i</i> , où <i>i</i> est l'index incrémentiel	OUI NON	NO
BOOLEAN_OPERATOR_ <i>i</i> , où <i>i</i> est l'index incrémentiel	&& 	&&

Les valeurs pour les opérateurs arithmétiques et booléens utilisent la syntaxe UNIX. Leurs significations sont décrites dans tableau 170.

Tableau 170. Description des opérateurs arithmétiques

Valeur UNIX du paramètre	Interprétation de la valeur du paramètre	Signification de la valeur du paramètre
-lt	<	Inférieur à
-le	<=	Inférieur ou égal à
-eq	=	Egal à
-ne	!=	Différent de
-ge	>=	Supérieur ou égal à
-gt	>	Supérieur à
&&	ET	ET logique
	OU	OU logique

Respect de la casse

Les règles suivantes s'appliquent lorsque vous indiquez un paramètre :

- Les noms de paramètre ne sont pas sensibles à la casse.
- Les valeurs des paramètres fixes ne sont pas sensibles à la casse.
- Les modèles à rechercher dans le journal de travail parent sont sensibles à la casse. Pour remplacer ce comportement, indiquez `IS_CASE_SENSITIVE_i=NO`, où *i* représente l'index de sous-condition en cours. Par exemple, pour le paramètre suivant :

```
PATTERN_2=text
```

Vous permutez la recherche de modèle sensible à la casse en spécifiant ce paramètre supplémentaire :

```
IS_CASE_SENSITIVE_2=NO
```

Si vous recherchez deux modèles dans la même ligne, ce paramètre les affecte tous les deux.

- Les séparateurs de paramètre requis pour définir le travail de branche sont sensibles à la casse. Vous devez spécifier les séparateurs en majuscules, sinon ils ne sont pas lus et les valeurs par défaut sont utilisées.

Exemples de statut d'exemple

Les sections ci-dessous récapitulent la construction de jeux de paramètres simples ou plus complexes.

Branche simple, scénarios de branche longue

Les valeurs par défaut suivantes sont utilisées :

- `CONDITION_SWITCH=PARENT_SUCCESS`
- `ACTION_SWITCH=CANCEL`

Vous n'avez pas besoin d'indiquer de paramètre d'entrée pour ces scénarios. Les valeurs par défaut sont fournies automatiquement par le travail de branche.

Pause

Cette fonction requiert au moins un paramètre fixe. Vous devez utiliser au moins deux travaux de branche d'implémenter pour implémenter le scénario de pause et de libération.

Pour utiliser la fonction de pause et de libération, vous devez remplacer le comportement par défaut pour le premier travail de branche. Le nom du travail de branche dans cet exemple est `BRANCH_1`. Si le nom de votre travail de branche a un suffixe différent (par exemple, `BRANCH_5`), vous devez ajuster les noms du séparateur.

Le second travail de branche dans l'approche de mise en pause et de libération ne nécessite aucun paramètre d'entrée s'il dépend uniquement du statut de son parent (SUCC).

Voici la syntaxe complète avec les séparateurs de paramètre :

```
BRANCH_1-BEGIN  
ACTION_SWITCH=PAUSE  
BRANCH_1-END
```

Aucun paramètre indexé n'est fourni. Les paramètres répertoriés peuvent être utilisés uniquement en combinaison avec le paramètre fixe `CONDITION_SWITCH=COMPLEX`. Du fait que le paramètre `CONDITON_SWITCH` n'est pas inclus, la valeur par défaut `CONDITION_SWITCH=PARENT_SUCCESS` a été utilisée.

Pour plus d'informations sur la combinaison de l'action de pause avec l'état complexe, voir «Etat complexe avec l'action de pause», à la page 1020.

Recherche simple de modèle

Cette fonction est représentée par une sous-condition et elle a besoin d'un paramètre fixe et d'un paramètre indexé.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est `BRANCH_1`. Si le nom de votre travail de branche a un suffixe différent (par exemple, `BRANCH_5`), vous devez ajuster les noms du séparateur.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=find this text
BRANCH_1-END
```

Il n'y existe qu'un paramètre répertorié représentant la seule sous-condition.

Recherche de modèle annulée

Cette fonction est représentée par une sous-condition et elle a besoin d'un paramètre fixe et deux paramètres indexés.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est `BRANCH_1`. Si le nom de votre travail de branche a un suffixe différent (par exemple, `BRANCH_5`), vous devez ajuster les noms du séparateur.

```
BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=do not find this text
NEGATE_CONDITION_RESULT_1=YES
BRANCH_1-END
```

Les deux paramètres indexés ont le même suffixe car ils appartiennent à la même sous-condition.

Recherche multiple de modèle

Cette fonction est représentée par plusieurs sous-conditions et elle a besoin d'un paramètre fixe et de plusieurs paramètres indexés.

Dans cet exemple, vous trouverez deux modèle indépendants dans le journal du travail parent. Vous créez deux sous-conditions distinctes, chacune avec son propre index(1 et 2). Vous devez également indiquer si *les deux* modèles doivent être recherché (à l'aide de l'opérateur booléen *AND*) ou si le fait de trouver *au moins un* modèle est suffisant (opérateur booléen *OR*).

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.

```
BRANCH_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
PATTERN_1=find this text  
PATTERN_2=find also this text  
BOOLEAN_OPERATOR_2=&&  
BRANCH_1-END
```

Les paramètres indexés possèdent des suffixes différents.

Le paramètre PATTERN_1 appartient à la première sous-condition et le paramètre PATTERN_2 appartient à la deuxième sous-condition. Le paramètre BOOLEAN_OPERATOR_2 indique comment la deuxième sous-condition est connectée à la première sous-condition.

Modèle dans une recherche de modèle

Cette fonction est représentée par une sous-condition et elle a besoin d'un paramètre fixe et deux paramètres indexés.

Dans cet exemple, vous recherchez le modèle de texte dans le journal du travail parent. Vous recherchez ensuite un autre modèle sur la même ligne. Tous les paramètres indexés appartiennent à une même sous-condition.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.

```
BRANCH_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
PATTERN_1=find this text  
VALUE_1=and also this text on the same row  
BRANCH_1-END
```

Les deux paramètres indexés ont le même suffixe car ils appartiennent à la même sous-condition.

Recherche de modèle avec comparaison numérique

Cette fonction est représentée par une sous-condition et elle a besoin d'un paramètre fixe et trois paramètres indexés.

Vous recherchez un modèle de texte dans le journal du travail parent. Vous recherchez ensuite un nombre sur la même ligne. Vous souhaitez comparer ce numéro avec un autre numéro que nous fournissons comme paramètre d'entrée. Pour la comparaison arithmétique, vous utilisez l'opérateur arithmétique spécifié comme paramètre d'entrée.

Tous les paramètres indexés appartiennent à une même sous-condition.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.


```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Total backup size
VALUE_1=500
ARITHMETICAL_OPERATOR_1=-1t
BRANCH_1-END

```

Les trois paramètres indexés ont le même suffixe car ils appartiennent à la même sous-condition.

La signification des fonctions appelées par les paramètres spécifiés peut être représentée comme suit :

1. Obtenir le journal de travail parent.
2. Extraire la ligne contenant la chaîne Total backup size.
3. Si (ligne trouvée) :
4. Alors passer à l'étape suivante
5. Else return FALSE
6. Tenter d'extraire la valeur numérique de la ligne.
7. Si (nombre trouvé) :
8. Alors passer à l'étape suivante
9. Else return FALSE
10. Si (number_from_joblog < 50) :
11. Then return TRUE
12. Else return FALSE

Combinaison de recherche de modèle et d'action de pause

Une condition autre qu'une condition par défaut peut être associée à une action autre qu'une action par défaut. Cela signifie que nous utilisons des entrées pour CONDITION_SWITCH et ACTION_SWITCH dans ce scénario.

Nous expliquons comment combiner une recherche de modèle unique et l'action de pause.

Cette fonction est représentée par une sous-condition et elle a besoin de deux paramètres fixes et d'un paramètre indexé.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.

```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
ACTION_SWITCH=PAUSE
PATTERN_1=find this text
BRANCH_1-END

```

Les deux paramètres fixes sont remplacés. Un paramètre indexé appartient à une seule sous-condition.

Action de signal

Le scénario de signal requiert des séparateurs différents.

Deux travaux gèrent le flux de flot de travaux :

- Travail de signal
- Travail de branche

Seul le signal utilise les paramètres d'entrée.

Le nom du travail de signal se compose du mot SIGNAL et du suffixe, ce qui signifie que les séparateurs de paramètre sont *différents* les uns des autres. De plus, pour le travail de signal, les règles communes sont valides. Les séparateurs de paramètre doivent ressembler au nom de travail connexe.

Ce scénario illustre la différence ainsi que les paramètres de l'action de signal. L'action de signal est combinée à la condition en fonction de la recherche de modèle dans le journal du travail parent.

La fonction de signal est représentée par un paramètre fixe. La recherche de modèle requiert un paramètre fixe pour le type de condition et un paramètre indexé indiquant la recherche de modèle.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.

```
SIGNAL_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
ACTION_SWITCH=SIGNAL  
PATTERN_1=find this text  
SIGNAL_1-END
```

Etat complexe avec l'action de pause

La condition complexe peut être associée à l'action explicite.

Pour plus d'informations sur la condition complexe, voir «Scénario complexe - conditions multiples», à la page 992. Dans ce scénario, l'état complexe est associé à l'action de pause.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Le nom du travail de branche est BRANCH_1. Si le nom de votre travail de branche a un suffixe différent (par exemple, BRANCH_5), vous devez ajuster les noms du séparateur.

```
BRANCH_1-BEGIN  
CONDITION_SWITCH=COMPLEX  
ACTION_SWITCH=PAUSE  
PATTERN_1=Error  
NEGATE_CONDITION_RESULT_1=YES  
PATTERN_2=Free space on Primary device  
VALUE_2=50  
ARITHMETICAL_OPERATOR_2=-gt  
BOOLEAN_OPERATOR_2=&&  
PATTERN_3=Free space on Secondary device
```

```

VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END

```

La signification de la définition se présente ainsi :

- Subcondition_1 : si vous trouvez le modèle Error, renvoyez FALSE, ou TRUE dans le cas contraire. Le résultat inversé est obtenu grâce au paramètre NEGATE_CONDITION_RESULT_1.
- Subcondition_3 : recherchez la ligne contenant Free space on Primary device. Extrayez le nombre de cette ligne. Si ce nombre est supérieur à 50, renvoyez TRUE, FALSE dans le cas contraire.
- Subcondition_3 : recherchez la ligne contenant Free space on Secondary device. Extrayez le nombre de cette ligne. Si ce nombre est supérieur à 60, renvoyez TRUE, FALSE dans le cas contraire.
- Placez ces trois sous-conditions à l'aide d'opérateurs booléens de sorte que la condition complexe soit générée comme suit :
 1. If (subcondition_1 = TRUE) AND (subcondition_2=TRUE) OR (subcondition_3=FALSE)
 2. Then return TRUE
 3. Else return FALSE
 4. If (complex_condition_result=TRUE)
 5. Then CANCEL bad_branch
 6. Else PAUSE the good_branch

Pour plus d'informations sur les concepts de pause et de libération, voir «Scénario d'actions de pause et de libération», à la page 996.

Travaux de branches multiples dans un même flot de travaux

Vous pouvez définir des paramètres pour plusieurs travaux de branche définis dans le même flot de travaux. Le fait de posséder de plusieurs travaux de branche dans un même flot de travaux signifie que les jeux de paramètres différents sont enserrés par des *séparateurs* différents.

L'exemple suivant affiche la syntaxe complète avec les séparateurs de paramètres. Les noms des travaux de branche sont BRANCH_1 et BRANCH-2. Si le nom de vos travaux de branche ont des suffixes différents (par exemple, BRANCH_5 et BRANCH_6), vous devez ajuster les noms du séparateur.

```

BRANCH_1-BEGIN
CONDITION_SWITCH=COMPLEX
PATTERN_1=Error
NEGATE_CONDITION_RESULT_1=YES
PATTERN_2=Free space on Primary device
VALUE_2=50
ARITHMETICAL_OPERATOR_2=-gt
BOOLEAN_OPERATOR_2=&&
PATTERN_3=Free space on Secondary device
VALUE_3=60
ARITHMETICAL_OPERATOR_3=-gt
BOOLEAN_OPERATOR_3=||
BRANCH_1-END

BRANCH_2-BEGIN
CONDITION_SWITCH=COMPLEX

```

PATTERN_1=Backup of Primary device
VALUE_1=ERROR
NEGATE_CONDITION_RESULT_1=YES
BRANCH_2-END

Remarques importantes sur le travail de branche

Cette section met en évidence certaines remarques et suppositions sur la conception du travail de branche.

- Le nom du travail de branche dans la base de données est BRANCH. La propriété Recovery options=STOP doit être définie par le travail.
- Le nom du travail de signal dans la base de données est SIGNAL. La propriété Recovery options=CONTINUE doit être définie par le travail.
- Le nom du travail de branche inséré dans le flot de travaux doit être constitué du nom du travail de branche (BRANCH) et du *suffixe*. Le suffixe correspond à la position du travail de branche dans le flot de travaux. Par exemple, BRANCH_3 est le troisième travail de branche dans le flot de travaux.
- Le nom du travail de signal inséré dans le flot de travaux doit être constitué du nom du travail de signal (SIGNAL) et du suffixe. Le suffixe correspond à la position du travail de signal dans le flot de travaux. Par exemple, SIGNAL_3 est le troisième travail de signal dans le flot de travaux.

Le compteur de suffixe pour les travaux de signal est différent du compteur de suffixe des travaux de branche, c'est pourquoi les travaux BRANCH_3 et SIGNAL_3 peuvent exister dans un flot de travaux en même temps, même si leur suffixe est identique.

- Le travail de branche possède un seul parent. Cela signifie que le travail de branche doit posséder la dépendance de prédécesseur/successeur exactement définie sur un travail dans le même flot de travaux. Ceci est vérifié lors du démarrage du travail de branche. Si cette condition n'est pas satisfaite, le travail de branche se termine anormalement.

La connexion du travail de branche à plusieurs parents est arrêtée par le travail de branche qui s'arrête anormalement.

- Pour le parent qui est évalué par rapport au statut du résultat (SUCC ou ABEND), vous devez définir la propriété suivante dans la définition de travail, et non dans la définition du flot de travaux :

Recovery options=Continue

- Pour les scénarios d'utilisation de travail de branche suivants, le travail de branche doit comporter exactement deux travaux enfants :
 - ACTION_SWITCH=CANCEL (valeur par défaut)
 - ACTION_SWITCH=PAUSE
 - Les travaux enfants doivent être signalés comme enfant correct et enfant incorrect. Les travaux enfants doivent être nommés comme suit :
 - Le travail qui représente l'enfant correct doit avoir un nom commençant par "G_". Il n'est pas nécessaire de renommer la *définition du travail*, renommez l'*alias du travail* dans le flot de travaux.
Par exemple, le nom du travail dans la base de données est START_BACKUP. Pour utiliser ce travail comme l'enfant correct du travail de branche, insérez-le dans le flot de travaux et dans le flot de travaux attribuez au travail l'alias G_START_BACKUP.
 - Le travail qui représente l'enfant incorrect doit avoir un nom commençant par "B_". Il n'est pas nécessaire de renommer la *définition de travail*, renommez l'*alias du travail* dans le flot de travaux.

Par exemple, le nom du travail dans la base de données est RESUME_DATABASE. Pour utiliser ce travail en tant qu'enfant incorrect du travail de branche, insérez ce travail dans le flot de travaux et dans le flot de travaux attribuez au travail l'alias B_RESUME_DATABASE.

Le nombre des travaux enfants et leurs préfixes corrects sont vérifiés au démarrage du travail de branche. Si ces conditions ne sont pas remplies, le travail de branche se termine anormalement.

- L'utilisation des guillemets n'est pas mise en oeuvre dans la version actuelle du travail de branche. Si vous spécifiez des guillemets dans les paramètres PATTERN_i ou VALUE_i, les guillemets sont automatiquement supprimés.
- Lors de l'évaluation de la condition complexe, le paramètre de recherche d'origine est un modèle de texte. Ce modèle texte est recherché dans le journal du travail parent. S'il existe plusieurs lignes comportant le modèle recherché, seule la première ligne est identifiée. Les autres lignes qui correspondent sont ignorées.

Le contenu de la première ligne identifiée est évalué par une autre sous-condition, comme illustré par le modèle dans les scénarios de ligne de modèle et de comparaison de valeur numérique.

- Lors de l'extraction de la valeur numérique de la ligne de modèle, il ne peut y avoir qu'une valeur numérique dans la ligne. Les valeurs numériques de type entier et réel sont acceptées.

La présence de plusieurs nombres dans la ligne de modèle identifiée engendrera l'extraction d'une valeur numérique incorrecte ; tous les nombres de la ligne seront concaténés et ne représenteront pas une valeur pertinente.

- Lorsque vous utilisez le scénario de signal, le travail de signal doit avoir exactement un enfant. L'enfant doit avoir un travail de branche consécutif et son nom doit être conforme aux conventions d'appellation du travail de branche générique.
- Lors de l'utilisation de la paire d'actions PAUSE et RELEASE, le processus est le suivant :
 - Le premier travail de branche définit la priorité du travail qui doit être mis en pause sur 0.
 - Le second travail de branche définit la priorité du travail qui doit être libéré sur 10.

Cette fonction est codée. Il n'y a aucun mécanisme mis en oeuvre qui vous permettrait de vous "rappeler" la priorité du travail d'origine.

- Toutes les informations sur le flot de travaux, dans lequel le travail de branche s'exécute, sont extraites de la *base de données* à l'aide de la commande *composer* lors du démarrage du travail de branche. Cela signifie que le travail de branche ne reflète aucune instance de travail ayant été *soumise dans* le flot de travaux mais qui n'existe pas dans la définition du flot de travaux.

Annexe E. Accessibilité

Les fonctions d'accessibilité permettent aux personnes souffrant d'un handicap physique (par exemple, une mobilité réduite ou une déficience visuelle) de pouvoir utiliser les logiciels. Les principales fonctions d'accessibilité de ce produit permettent aux utilisateurs d'effectuer les opérations suivantes :

- Utilisation des technologies d'assistance, comme le logiciel de lecture d'écran et le synthétiseur de parole numérique, pour entendre la description de l'affichage. Pour plus de détails sur l'utilisation de ces technologies avec ce produit, consultez la documentation produit de la technologie d'assistance.
- Utilisation des fonctions spécifiques ou équivalentes à l'aide du clavier uniquement.
- Agrandissement des éléments affichés.

En outre, la documentation produit comprend maintenant des options pour une meilleure accessibilité :

- Entièrement disponible en HTML et PDF convertible, elle permet la plus large utilisation possible du logiciel de lecture d'écran.
- Toutes ses images sont fournies avec un texte secondaire pour que les utilisateurs visuellement déficients puissent en comprendre la teneur.

Navigation dans l'interface avec le clavier

Des touches de raccourci et de clavier standard sont utilisées par le produit et sont documentées par le système d'exploitation. Pour plus de détails, consultez la documentation du système d'exploitation.

Le panneau relatif à l'éditeur de règle d'événement est le seul panneau qui ne permet pas d'effectuer des opérations à l'aide du clavier uniquement et le sous-système de canaux ne peut pas être désactivé. Cependant, vous pouvez autrement effectuer toutes les opérations disponibles dans ce panneau en l'exécutant la commande composer à partir de l'interface de ligne de commande.

Agrandissement des éléments affichés

Vous pouvez agrandir les informations affichées dans les fenêtres du produit, à l'aide des fonctions des systèmes d'exploitation sur lesquels le produit est exécuté. Par exemple, dans un environnement Microsoft Windows, vous pouvez réduire la résolution de l'écran pour agrandir la taille des polices du texte affiché. Pour plus de détails, consultez la documentation du système d'exploitation.

Remarques

Ces informations ont été développées pour des produits et services proposés aux Etats-Unis. Ce matériel est disponible dans d'autres langues auprès d'IBM. Toutefois, une copie du produit ou de la version du produit dans cette langue peut être nécessaire pour y accéder.

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, logiciel ou service IBM n'implique pas que seul ce produit, logiciel ou service IBM puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit de IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous accorde aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

Pour plus d'informations sur les licences concernant les produits utilisant un jeu de caractères double octet, contactez IBM Intellectual Property Department dans votre pays ou envoyez votre demande par écrit à l'adresse suivante :

*Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan*

INTERNATIONAL BUSINESS MACHINES CORPORATION FOURNIT LE PRESENT DOCUMENT "EN L'ETAT" ET DECLINE TOUTE RESPONSABILITE, EXPLICITE OU IMPLICITE, RELATIVE AUX INFORMATIONS QUI Y SONT CONTENUES, Y COMPRIS EN CE QUI CONCERNE LES GARANTIES DE VALEUR MARCHANDE OU D'ADAPTATION A VOS BESOINS. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US*

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans ce document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux dispositions du document IBM Customer Agreement, des Conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance présentées ici ont été obtenues dans des conditions de fonctionnement spécifiques. Les résultats peuvent donc varier.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Ces informations sont fournies uniquement à titre de planification. Elles sont susceptibles d'être modifiées avant la mise à disposition des produits décrits.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs de personnes, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms de personnes, de sociétés ou des données réelles serait purement fortuite.

LICENCE DE COPYRIGHT :

Le présent logiciel contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM, à des fins de développement, d'utilisation, de

vente ou de distribution de programmes d'application conformes aux interfaces de programmation des plateformes pour lesquels ils ont été écrits. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Les exemples de programmes sont fournis tels quels sans aucune garantie d'aucune sorte. IBM n'est en aucun cas responsable des dommages liés à l'utilisation de ces exemples de programmes.

© (nom de votre société) (année).

Des segments de code sont dérivés des exemples de programme d'IBM Corp..

© Copyright IBM Corp. _entrez l'année ou les années_.

Marques

IBM, le logo IBM et ibm.com sont des marques d'International Business Machines Corp. dans de nombreux pays. Les autres noms de produits et de services peuvent appartenir à IBM ou à des tiers. La liste actualisée de toutes les marques d'IBM est disponible sur la page web "Copyright and trademark information" à l'adresse www.ibm.com/legal/copytrade.shtml.

Adobe, le logo Adobe, PostScript et le logo PostScript sont des marques d'Adobe Systems Incorporated aux Etats-Unis et/ou dans certains autres pays.

IT Infrastructure Library est une marque de AXELOS Limited.

Linear Tape-Open, LTO, le logo LTO, Ultrium et le logo Ultrium sont des marques de HP, IBM Corp. et Quantum aux Etats-Unis et dans certains autres pays.

Intel, le logo Intel, Intel Inside, le logo Intel Inside, Intel Centrino, le logo Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium et Pentium sont des marques d'Intel Corporation ou de ses filiales aux Etats-Unis et/ou dans certains autres pays.

Linux est une marque de Linus Torvalds déposée aux Etats-Unis et/ou dans certains autres pays.

Microsoft, Windows, Windows NT et le logo Windows sont des marques de Microsoft Corporation aux Etats-Unis et/ou dans certains autres pays.



Java ainsi que tous les logos et toutes les marques incluant Java sont des marques d'Oracle et/ou ses sociétés affiliées.

Cell Broadband Engine est une marque de Sony Computer Entertainment, Inc. aux Etats-Unis et/ou dans certains autres pays et est utilisée sous licence.

ITIL est une marque d'AXELOS Limited.

UNIX est une marque enregistrée aux Etats-Unis et dans certains autres pays, dont seule la société The Open Group peut concéder la licence.

Dispositions pour la documentation du produit

Les droits d'utilisation relatifs à ces publications sont soumis aux dispositions suivantes.

Applicabilité

Ces dispositions viennent s'ajouter à celles du site Web IBM.

Usage personnel

Vous pouvez reproduire ces publications pour votre usage personnel, non commercial, sous réserve que toutes les mentions de propriété soient conservées. Vous ne pouvez distribuer ou publier tout ou partie de ces publications ou en faire des oeuvres dérivées sans le consentement exprès de IBM.

Utilisation commerciale

Vous pouvez reproduire, distribuer et afficher ces publications uniquement au sein de votre entreprise, sous réserve que toutes les mentions de propriété soient conservées. Vous ne pouvez reproduire, distribuer, afficher ou publier tout ou partie de ces publications en dehors de votre entreprise, ou en faire des oeuvres dérivées, sans le consentement exprès de IBM.

Droits

Excepté les droits d'utilisation expressément accordés dans ce document, aucun autre droit, licence ou autorisation, implicite ou explicite, n'est accordé pour ces publications ou autres informations, données, logiciels ou droits de propriété intellectuelle contenus dans ces publications.

IBM se réserve le droit de retirer les autorisations accordées ici si, à sa discrétion, l'utilisation des informations s'avère préjudiciable à ses intérêts ou que, selon son appréciation, les instructions susmentionnées n'ont pas été respectées.

Vous ne pouvez télécharger, exporter ou réexporter ces informations qu'en total accord avec toutes les lois et règlements applicables dans votre pays, y compris les lois et règlements américains relatifs à l'exportation.

IBM N'OCTROIE AUCUNE GARANTIE SUR LE CONTENU DE CES PUBLICATIONS. LES PUBLICATIONS SONT LIVREES EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES PUBLICATIONS EN CAS DE CONTREFAÇON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Index

Caractères spéciaux

.jobmanrc, script de configuration 60

A

abenp

état du travail 426

accès lent à la base de données 736

access

agents étendus et agents de réseau 175

définition de poste de travail 175

accessibilité xviii, 1025

action

élément 921

action du déclencheur 19

actions sur les objets de sécurité
indication d'actions sur les objets de sécurité 311

activation

communication SSL 179

fuseau horaire 799

activation de la prévision d'heure de

début 85

add

état du travail 426

flot de travaux, état 434

add, commande 338

adddep job, commande 439

adddep sched, commande 442

affinité

définition 671

syntaxe 671

affinité avec alias de travail 671

affinité avec ID de travail 671

affinité avec nom de travail 671

agent 11, 13

arrêt 43

définition de l'utilisateur

Windows 203

définition de poste de travail 176

démarrage 43

méthode d'accès

syntaxe 806

agent dynamique

définition de poste de travail 176

méthode d'accès

fichier d'options 809

passerelle 13

présentation 805

agent étendu

définition de poste de travail 175

méthode d'accès

exécution 812

fichier d'options 809

identification des incidents 815

messages de réponse 808

présentation 805

agent étendu (*suite*)

tâche d'obtention d'un statut

syntaxe 814

tâche de vérification d'un fichier

syntaxe 813

agent IBM Workload Scheduler for z/OS

méthode d'accès

fichier d'options 809

agent réseau

définition 819

dépendance interréseau 817

création 821

gestion à l'aide de conman 822

exemple de scénario 820

externe 822

EXTERNE, état

ERROR 822

EXTRN 822

méthode d'accès

fichier d'options 819

netmth, méthode d'accès 819

présentation 817

référence 817

agent standard

définition de poste de travail 175

agent tolérant aux pannes 11

agents dynamiques 575, 667

agréger

dépendances 831

allocation

élément 914

altération de fichier Symphony

commande resetFTA 487

altpass, commande 444

altpri, commande 445

amélioration des performances de la base
de données 736

anciens travaux

amélioration 675

anciens travaux avec fonctionnalités

dynamiques 675

and

élément 913

annotation

élément 900

API Java 32

appel vers un service Web 669

modèles de fichiers JSDL 580

appel vers un service Web

générique 669

modèle 580

application

élément 903

application de charge de travail 394, 398

définition 3

ligne de commande 406

applications de charge de travail

correspondance 405

définition 304

fichier de mappage 399

gestion 393

applications de charge de travail (*suite*)

importation 405

journaux et fichiers de trace 406

régulières, expressions 404

Appserverbox.msg 45

appservman

arrêt 531

architecture 37

archivage

instances de travail 736

arguments

élément 926

arrêt

processus de poste de travail 43

WebSphere Application Server 43

association d'ordinateurs

extraction 741

association de travaux

définition 671

association logique de ressources

extraction 741

assurance de service de charge de travail

calcul des heures de début des

travaux 85

plan prévisionnel 85

at, commande 679

ATSCRIPT, variable 679

at, mot clé 228, 233

attente

état du travail 427

attributs d'objet

attributs des types d'objet 316

aucune installation

instances d'agent dynamique 645

auditbox.msg 45

authenticate, commande 340

autolink

définition de poste de travail 178

automatisation conditionnelle 135

automatisation conditionnelle de la

charge de travail

définition 138

automatisation de la charge de

travail 135

autostart monman 544

B

base de données

réplication du plan 28

batch, commande 679

batchman, processus 39

behindfirewall

définition de poste de travail 178

boîte aux lettres, fichiers

Appserverbox.msg 45

auditbox.msg 45

Courier.msg 45

définition de la taille 696

Intercom.msg 45

Mailbox.msg 46

boîte aux lettres, fichiers (*suite*)

- mirrorbox.msg 46
- Monbox.msg 46
- Moncmd.msg 46
- NetReq.msg 46
- PlanBox.msg 46
- Server.msg 46

broker

- définition de poste de travail 176

bulk_discovery, commande 446

C

calendar

- cycle d'exécution 218, 244, 268
- Cycle d'exécution 10
- holidays 10
- jours chômés 10

cancel, commande 446

cancel sched, commande 448

candidateCPUs

- élément 906

candidateHosts

- élément 905

candidateOperatingSystems

- élément 908

candidateResources

- élément 916

caractères génériques

- composer 327

carryforward

- mots clés des flots de travaux 82
- personnalisation 82
- stageman 82
- variable 108

carryforward, mot clé 235

carryStates

- variable 82, 87

category

- élément 900

caxtract, commande 777

cibles de travail

- définition 934

classe

- Poste de travail 15

classe de poste de travail 15

classe de postes de travail,

- définition 184

- cpuclass 184

- ignore 185

- members 185, 308, 309

cloud

- travaux d'application des accès
- SmartCloud 592

- travaux de type services Web
- RESTful 620

Cloud & Smarter Infrastructure,

- formation technique xviii

code retour sur AS/400 868

code retour sur i5/OS 868

code retour sur IBM i 868

code retour utilisateur sur AS/400 868, 870

code retour utilisateur sur i5/OS 868, 870

code retour utilisateur sur IBM i 868, 870

codes retour

- programme d'exécution de
- travail 644

- Travail avec options avancées 644

- travail de base de données 644

- travail de services Web 644

- travail de transfert de fichier 644

- travail Java 644

combiner

- dépendances 831

command

- logman 110

commande

- Cpuinfo 815

- stageman 107

commande checkhealthstatus 450

commande continue (composer) 340

commande continue (conman) 454

commande datamigrate 689

commande exportserverdata 732

commande importserverdata 733

commande param 738

commande resetFTA 487

commande StartUpLwa 723

commande twstrace 753

Commande wappman 406

commandes

- adddep job 439

- adddep sched 442

- altpass 444

- altpri 445

- at 679

- batch 679

- bulk_discovery 446

- cancel job 446

- cancel sched 448

- caxtract 777

- checkhealthstatus 450

- commande d'utilitaire de la
- version 723

- confirm 450

- console 453

- continue (composer) 340

- continue (conman) 454

- cpuinfo 682

- datamigrate 689

- datecalc 685

- deldep job 455

- deldep sched 457

- delete 691

- deployconf 458

- display 459

- evtdef 692

- evtsize 696

- exit 461

- exportserverdata 732

- fence 462

- filemonitor 698

- getmon 490

- help 463

- importserverdata 733

- jbextract 775

- jobinfo 701

- jobprop 735

- jobstdl 703

- kill 464

- limit cpu 465

commandes (*suite*)

- limit sched 467

- link 468

- listsucc 472

- listsym 470

- maestro 706

- makecal 706

- metronome 708, 723

- morestdl 708

- movehistorydata 736

- param 738

- parms 710

- paxtract 778

- prxtract 776

- r1lxtr 779

- recall 474

- redo 475

- release 713

- release job 476

- release sched 478

- rep1 757

- rep11 761

- rep2 757

- rep3 757

- rep4a 757

- rep4b1 757

- rep7 758

- rep8 759

- reply 480

- reptr 762

- rerun 481

- rerunsucc 484

- resource 488, 741

- rextract 779

- rmstdlist 717

- sendevent 719, 751

- setsym 489

- showcpus 490

- showdomain 497

- showexec 720

- showfiles 498

- showjobs 501

- showprompts 520

- showresources 523

- showschedules 525

- shutdown 531

- start 532

- startappserver 534

- startbrokerapp 535

- starteventprocessor 535

- startmon 536

- StartUp 722

- StartUpLwa 723

- status 537

- stop 538

- stop ;progressive 540

- stopappserver 541

- stopbrokerapp 542

- stopeventprocessor 543

- stopmon 544

- submit docommand 545

- submit file 549

- submit job 552

- submit sched 556

- switcheventprocessor 561

- switchmgr 562

- tellop 564

- commandes (*suite*)
 - unlink 565
 - version 567
 - xref 763
 - xrxtct 781
- commandes d'utilitaire 677
 - affichage de la version du produit 723
 - affichage des fichiers de liste standard 717
 - affichage des informations relatives aux travaux 701
 - affichage des travaux en cours d'exécution 720
 - affichage du contenu des fichiers de liste standard 708
 - agents 727
 - at 679
 - at.allow, fichier 681
 - at.deny, fichier 681
 - ATSCRIPT, variable 679
 - batch 679
 - création d'agendas 706
 - création et gestion des variables et mots de passe en local sur des agents dynamiques 738
 - datamigrate, informations 689
 - définition d'événements personnalisés 692
 - définition de la taille des fichiers de boîte aux lettres 696
 - définition de variables localement sur des agents dynamiques 735
 - démarrage de netman 722, 723
 - dynamique 727
 - envoi d'événements personnalisés 719
 - extraction des rapports HTML 708, 723
 - filemonitor 698
 - gestion locale des paramètres 710
 - gestionnaire de domaine dynamique 727
 - libération des unités de ressource 713
 - liste des commandes 677
 - liste des fichiers de la liste standard 703
 - modification du format de date 685
 - obtention des informations relatives aux postes de travail 682
 - obtention du chemin racine_TWS 706
 - shutdown 721
 - ShutDownLwa 722
 - suppression de fichiers 691
 - suppression des fichiers de liste standard 717
- commandes utilitaires pour les agents dynamiques
 - envoi d'événements personnalisés 751
- comment, mot clé 235
- communication réseau 48
 - début de la journée 48
 - traitement des travaux 48
- communication SSL
 - activation 179
- compiler
 - messages 816
- composer
 - messages 816
- composer, programme
 - codes retour de ligne de commande 331
 - éditeur 323
 - éditeur XML 323
- Composer, programme 321
 - caractères de commande 326
 - caractères génériques 327
 - caractères spéciaux 330
 - configuration 322
 - variables 755
 - variables sous UNIX 322
 - variables sous Windows 322
 - délimiteurs 330
 - exécution de commandes 324
 - filtres 327
 - liste des commandes 331
 - paramètres de connexion 324
 - prompt 323
 - sortie offline 322
 - sortie sur le terminal 322
- composer, référence 163
- composeur
 - ligne de commande 30
- COMPUTERNAME, variable 54
- condition de démarrage 135
- configuration
 - génération de rapports de ligne de commande 793
 - propriétés locales 53
- configuration, scripts
 - .jobmanrc 60
 - djobmanrc.cmd 63
 - jobmanrc 57
 - jobmanrc.cmd 61
- configuration de ligne de commande
 - fichier CLIConfig.properties 728
- confirm, commande 450
- confirmed, mot clé 236
- conman
 - ligne de commande 30
- conman, programme 411
 - caractères de commande 415
 - caractères génériques 417
 - caractères spéciaux 417
 - configuration 412
 - variables définies 412
 - délimiteurs 417
 - exécution de commandes 413, 416
 - invites utilisateur 416
 - liste des commandes 436
 - prompt 413
 - sélection d'un travail 419
 - arguments 419
 - at, utilisation 421
 - confirm, utilisation 421
 - critical, utilisation 421
 - critnet, utilisation 421
 - deadline, utilisation 422
 - every, utilisation 422
 - finished, utilisation 422
- conman, programme (*suite*)
 - sélection d'un travail (*suite*)
 - follows, utilisation 423
 - ID_flot_travaux 420
 - logon, utilisation 424
 - needs, utilisation 424
 - nom_flot_travaux 419
 - opens, utilisation 424
 - priority, utilisation 425
 - prompt, utilisation 425
 - recovery, utilisation 425
 - schedid 420
 - scriptname, utilisation 425
 - started, utilisation 426
 - state, utilisation 426
 - until, utilisation 427
 - sélection des flots de travaux 428
 - arguments 428
 - at, utilisation 429
 - carriedforward, utilisation 430
 - carryforward, utilisation 430
 - finished, utilisation 430
 - follows, utilisation 431
 - ID_flot_travaux 429
 - limit, utilisation 432
 - needs, utilisation 432
 - nom_flot_travaux 428
 - opens, utilisation 432
 - priority, utilisation 433
 - prompt, utilisation 433
 - schedid 429
 - started, utilisation 433
 - state, utilisation 434
 - until, utilisation 434
 - sortie offline 412
 - sortie sur le terminal 412
 - traitement 418
- conman, référence 411
- conman startappserver
 - JnextPlan 91
- connFactory
 - élément 932
- consignation des statistiques de travaux 110
- console, commande 453
- Console de définition de courtage de travaux
 - modification de définitions de travail 670, 671
- conventions, typographie xviii
- Courier.msg 45
- cpuclass
 - classe de postes de travail, définition 184
- Cpuinfo
 - command 815
- cpuinfo, commande 682
- cpuname
 - définition de poste de travail 172
- create, commande 352
- CreatePostReports
 - JnextPlan 92
- création
 - application de charge de travail 394
- création d'un plan d'essai
 - ligne de commande planman 99

- création d'un plan prévisionnel
 - ligne de commande planman 101
- création d'un travail 187, 581
- création d'un travail dynamique 187, 581
- création d'une définition de travail
 - application des accès oslc, procédure préalable 606
 - automatisation oslc, procédure préalable 606
 - scp, procédure préalable 592
- credential
 - élément 923, 928, 932
- critères de correspondance
 - dans un intervalle absolu 73, 261
 - dans un intervalle relatif 72, 261
 - follows 248
 - follows absolute to 73
 - follows previous 72
 - follows relative to 72
 - follows sameday 71
 - join 256
 - même jour 71, 261
 - prédécesseur 74
 - prédécesseur suspendu 75
 - successeur 74
 - valeur précédente la plus proche 72, 261
- cycle d'exécution
 - jour 218, 244, 267
- Cycle d'exécution
 - Annuel 4
 - Basé sur des décalages 4
 - Basé sur des règles 4
 - calendar 218, 244, 268
 - date 218, 244, 267
 - exclusif 10
 - Exclusif 4
 - freedays 4
 - Hebdomadaire 4
 - icalendar 219, 245, 268
 - inclusif 10
 - Inclusif 4
 - on 267
 - quotidien 4
 - Simple 4
- cycle d'exécution, règles 4
- cycle d'exécution annuel 4
- cycle d'exécution basé sur des décalages 4
- cycle d'exécution exclusif 4
- cycle d'exécution hebdomadaire 4
- cycle d'exécution inclusif 4
- cycle d'exécution les jours chômés 4
- cycle d'exécution quotidien 4
- cycle d'exécution simple 4
- cycle de production 67
 - gestion 67
 - identification des instances de flot de travaux 70
 - ligne de commande planman 94

D

- d-pool
 - Poste de travail 177

- dans un intervalle absolu
 - critères de correspondance 73
 - follows 248
 - follows absolute to 73
 - join 256
- dans un intervalle relatif
 - critères de correspondance 72
 - follows 248
 - follows relative to 72
 - join 256
- date
 - cycle d'exécution 218, 244, 267
- datecalc, commande 685
- deadline, mot clé 237
- début de la journée
 - établissement de la communication 49
- définition
 - automatisation conditionnelle de la charge de travail 138
 - définition des objets de sécurité dans la base de données 306
- dépendances
 - follows 248
 - join 256
 - needs 265
 - opens 274
 - prompts 278
- dépendances conditionnelles 829
- objets de base de données
 - agendas 205
 - classes de postes de travail 184
 - Domaine de sécurité 307
 - domaines 185
 - flot de travaux 226
 - groupe de cycle d'exécution 216
 - JobManagement 636
 - jobs 187
 - JSR 352 Java Batch 629
 - Liste de contrôle d'accès de sécurité 306
 - MQTT 631
 - postes de travail 166
 - prompts 10, 213
 - règles d'événement 291
 - ressources 215
 - Rôle de sécurité 309
 - services Web 618
 - transfert de fichier 598
 - travail Soumission d'un flot de travaux 641
 - travaux AS/400 585
 - travaux d'application des accès OSLC 610
 - travaux d'automatisation OSLC 608
 - travaux de base de données 612
 - travaux de la méthode d'accès 590
 - travaux exécutables 588
 - travaux i5/OS 585
 - travaux IBM i 585
 - Travaux J2EE 625
 - travaux Java 624
 - travaux JCL 582
 - travaux MSSQL 615
 - travaux reflet 596

- définition (*suite*)
 - objets de base de données (*suite*)
 - utilisateurs Windows 200
 - variables 206
 - VariableTable 633
 - objets de sécurité dans la base de données 306
 - paramètres_connexion 65
 - table de variables 211
 - travaux d'application des accès SmartCloud 592
 - travaux de type commande distante 582
 - travaux de type services Web RESTful 620
- définition d'agenda 205
- définition d'application de charge de travail 394
- définition d'invite 10, 213
- Définition d'objets
 - dans la base de données 163
- définition d'un groupe de cycle d'exécution 216
- définition d'utilisateur 200
 - domaine sécurisé 204
 - utilisation sur les types de travaux avec options avancées 204
- définition de domaine 185
 - ismaster 186
 - manager 186
 - parent 186
- définition de domaine de sécurité
 - domaine de sécurité 308
- Définition de domaine de sécurité 307
- définition de flot de travaux 226
 - exporter 397
- définition de listes de contrôle d'accès
 - liste de contrôle d'accès de sécurité 306
- Définition de listes de contrôle d'accès de sécurité 306
- définition de paramètre 206
- définition de poste de travail 166, 179
 - access 175
 - agent 176
 - agent dynamique 176
 - agent étendu 175
 - agent standard 175
 - autolink 178
 - behinfirewall 178
 - broker 176
 - cpuname 172
 - domaine 174
 - fta 175
 - fullstatus 178
 - hôte 174
 - ID serveur 180
 - manager 176
 - members 180
 - protocole 180
 - requirements 180
 - secureaddr 173
 - tcpaddr 173
 - timezone 174
 - type 175
 - type de système d'exploitation 172
- définition de règle d'événement 291

définition de règle d'événement *(suite)*

- eventRules.xsd 291
- mots clés
 - actionProvider 298
 - actionType 299
 - activeTime 294
 - correlationAttributes 298
 - description 294, 300
 - eventCondition 295
 - eventProvider 295
 - eventRule 293
 - eventType 295
 - filteringPredicate 297
 - heure d'été 294
 - isDraft 293
 - name 293
 - onDetection 300
 - onTimeOut 300
 - opérateur 298
 - responseType 300
 - ruleType 293
 - scope 297, 300
 - timeInterval 294
 - timeZone 294
 - validity 294

définition de ressource 215

- définition de rôles de sécurité
 - rôle de sécurité 309

- définition de table de variables
 - table de variables 172

définition de travail 187

- création 580, 670, 671
- docommand 190
- interactive 192
- option de reprise 195
- scriptname 189
- streamlogon 191
- tâche 190
- tasktype 191
- travaux d'application des accès
 - OSLC 610
- travaux d'application des accès
 - SmartCloud 592
- travaux d'automatisation OSLC 608
- travaux de base de données 612
- travaux de la méthode d'accès 590
- travaux de services Web 618
- travaux de transfert de fichier 598
- travaux de type commande
 - distante 582
- travaux de type services Web
 - RESTful 620
- travaux exécutables 588
- travaux IBM i 585
- Travaux J2EE 625
- travaux Java 624
- travaux JCL 582
- travaux JobManagement 636
- travaux JSR 352 Java Batch 629
- travaux MQTT 631
- travaux reflet 596
- travaux VariableTable 633
- utilisation de variables et de paramètres 200

Définition de travail

- travaux MSSQL 615

définition de travail d'automatisation oslc

- procédure préalable 606
- définition de travaux IBM i 861
- définition de travaux non opérationnels (NOP) dans des flots de travaux 266
- définition de travaux sur AS/400 861
- définition de travaux sur i5/OS 861
- définition de travaux sur IBM i 861
- définition des agents sur les systèmes AS/400 861
- définition des agents sur les systèmes i5/OS 861
- définition des agents sur les systèmes IBM i 861
- Définition du rôle de sécurité 309
- deldep job, commande 455
- deldep sched, commande 457
- delete, commande 341, 691
- démarrage
 - processus de poste de travail 43
 - WebSphere Application Server 43
- démarrage du traitement
 - plan de production 113
- démarrage et arrêt des agents sur IBM i 862
- démarrage et arrêt des agents sur les systèmes AS/400 862
- démarrage et arrêt des agents sur les systèmes i5/OS 862
- démarrage et arrêt des agents sur les systèmes IBM i 862
- dépendance
 - croisée 854
 - interréseau 817, 822
- dépendance croisée
 - ajout au plan 851
 - comme dépendance sur un travail
 - reflet 850
 - définition 850
 - étapes de définition 850
 - flux d'informations 847, 854
 - introduction 845
 - logique 845
 - plan de production 852
 - poste de travail de moteur
 - distant 845
 - surveillance de la résolution dans le plan 851
 - travail distant 845
 - travail reflet 845
- dépendance de fichier
 - définition 672
- dépendance de travail
 - définition 671
- dépendance interréseau
 - création 821
 - gestion à l'aide de conman 822
- dépendances
 - conditionnelle 827
 - joindre, combiner 831
 - orphelines 75
- dépendances conditionnelles
 - définition 827, 829
 - joindre 831
 - réplication de plan 842
- dépendances croisées
 - définition 845

dépendances croisées *(suite)*

- gestion 845
- dépendances de fichier 672
 - définition 672
 - dépendances de fichier 672
- Déploiement de règles
 - ligne de commande planman 102
- deployconf, commande 458
- description, mot clé 238
- destination
 - élément 932
- déverrouillage du plan
 - ligne de commande planman 103
- diskSpace
 - élément 911
- display, commande 345, 459
- docommand
 - définition de travail 190
- domaine 16
 - définition de poste de travail 174
- domaine de sécurité
 - définition de domaine de sécurité 308
- done
 - état du travail 426
- données de plan 28
- doubleVariable
 - élément 902
- draft, mot clé 238
- durée estimée 569
- durée prévue 112
- Dynamic Workload Console
 - accessibilité xviii

E

- edit, commande 351
- ejb
 - élément 931
- ELAB_JOB_STAT_JS 571, 573
- éléments
 - action 921
 - allocation 914
 - and 913
 - annotation 900
 - application 903
 - arguments 926
 - candidateCPUs 906
 - candidateHosts 905
 - candidateOperatingSystems 908
 - candidateResources 916
 - category 900
 - connFactory 932
 - credential 923, 928, 932
 - destination 932
 - diskSpace 911
 - doubleVariable 902
 - ejb 931
 - endpointReference 916
 - environment 927
 - estimatedDuration 920
 - ewlm 919
 - executable 924
 - exigence 914
 - fileSystem 910
 - group 912
 - groupName 929

- éléments (*suite*)
 - hostName 906
 - invoker 930
 - j2ee 930
 - JAASAuthenticationAlias 934
 - jms 930
 - jndiHome 931
 - jobDefinition 899
 - logicalResource 911
 - maximumResourceWaitingTime 920
 - message 932
 - mot de passe 923, 929, 933
 - objective 917
 - operatingSystem 909
 - optimization 916
 - orderedCandidatedWorkstations 905
 - ou 913
 - parameters 922
 - physicalMemory 908
 - priority 920
 - propriétés 913
 - recoveryActions 921
 - relatedResources 904
 - relationship 915
 - resources 903
 - scheduling 919
 - script 926
 - speed 908
 - stringVariable 901
 - tpmaction 922
 - tpmaddress 923
 - UC 907
 - uintVariable 902
 - userName 923, 928, 933
 - value 927
 - variable 928
 - variables 901
 - virtualMemory 908
 - workflow 924
 - enCarryForward
 - variable 82, 87
 - enCFInterNetworkDeps
 - variable 87
 - enCFResourceQuantity
 - variable 87
 - end, mot clé 239
 - endpointReference
 - élément 916
 - enLegacyId
 - variable 88
 - enLegacyStartOfDayEvaluation
 - variable 90, 800
 - enPreventStart
 - variable 88
 - enTimeZone
 - variable 90, 799
 - environment
 - élément 927
 - environnement de travail sur
 - AS/400 869
 - environnement de travail sur i5/OS 869
 - environnement de travail sur IBM i 869
 - error
 - état du travail 427
 - estimatedDuration
 - élément 920
 - état du poste de travail 493, 496
 - événements personnalisés
 - définition 161, 692
 - envoi 161, 719
 - envoi à partir d'agents dynamiques 751
 - every, mot clé 239
 - utilisés dans les définitions de flots de travaux 239
 - utilisés dans les définitions de travaux 240
 - evtdef, commande 692
 - evtsz, commande 696
 - ewlm
 - élément 919
 - EWLM, intégration
 - activation dans les travaux 916
 - optimisation des fonctions 916
 - EWLM, optimisation
 - activation dans les travaux 916
 - except, mot clé 244
 - exec
 - état du travail 427
 - flot de travaux, état 434
 - executable
 - élément 924
 - exécution de commandes système
 - depuis composer 383
 - depuis Conman 416, 563
 - exigence
 - élément 914
 - exit, commande 352, 461
 - exporter
 - définition de flot de travaux 397
 - extension d'essai
 - ligne de commande planman 100
 - externe
 - flot de travaux 822
 - jobs 823
 - extract, commande 352
 - extraction des données de la base de données 580, 669
 - extrn
 - état du travail 427
- ## F
- facteur de fiabilité 112
 - fail
 - état du travail 427
 - fdignore
 - except 247
 - on 221, 270
 - fdnext
 - except 247
 - on 221, 270
 - fdprev
 - except 247
 - on 221, 270
 - fence
 - état du travail 427
 - fence, commande 462
 - fichier CLIconfig.properties
 - configuration de ligne de commande 728
 - fichier d'options
 - agent dynamique
 - méthode d'accès 809
 - fichier d'options (*suite*)
 - agent étendu
 - méthode d'accès 809
 - agent IBM Workload Scheduler for z/OS
 - méthode d'accès 809
 - fichier d'options globales
 - name 809
 - fichier d'options locales
 - name 809
 - fichier de mappage, applications de charge de travail 399
 - fichier de mappage, expressions régulières 404
 - fichier Symphony
 - JnextPlan 91
 - plan de production 82, 91
 - fichiers
 - Appserverbox.msg 45
 - at.allow 681
 - at.deny 681
 - auditbox.msg 45
 - Courier.msg 45
 - Intercom.msg 45
 - Mailbox.msg 46
 - mirrorbox.msg 46
 - Monbox.msg 46
 - Moncmd.msg 46
 - NetReq.msg 46
 - PlanBox.msg 46
 - Server.msg 46
 - filemonitor, commande 698
 - fileSystem
 - élément 910
 - filtres
 - composer 327
 - fin anormale
 - état du travail 426
 - flot de travaux, état 434
 - final, flot de travaux
 - traitement automatique du plan 114
 - flot de travaux 2
 - calcul du temps d'exécution 112
 - externe 822
 - flot de travaux, états
 - add 434
 - exec 434
 - fin anormale 434
 - hold 434
 - ready 434
 - stuck 434
 - succ 434
 - FNCJSI
 - plan de préproduction 70
 - follows
 - critères de correspondance 248
 - follows, mot clé 248
 - follows absolute to
 - critères de correspondance 73
 - dans un intervalle absolu 73
 - follows previous
 - critères de correspondance 72
 - valeur précédente la plus proche 72
 - follows relative to
 - critères de correspondance 72
 - dans un intervalle relatif 72

- follows sameday
 - critères de correspondance 71
 - même jour 71
- fonction miroir 28
- fonctions dynamiques 575, 667
- formation xviii
 - technique xviii
- freedays, mot clé 252
- fta
 - définition de poste de travail 175
- fullstatus
 - définition de poste de travail 178
- fuseau horaire
 - activation 799

G

- génération d'états, commandes 755
 - Calendrier de production planifiée 761
 - exemple de sortie 771
 - configuration 755
 - Détail de la production planifiée 762
 - exemple de sortie 769
 - Détail de la production réelle 762
 - exemple de sortie 770
 - exemples de sorties 764
 - extraction, programmes 774
 - caxtract 777
 - jbextract 775
 - paxtract 778
 - prxtract 776
 - r11xtr 779
 - rextract 779
 - rxrtract 781
 - Histogramme des travaux 759
 - exemple de sortie 769
 - Liste des agendas 757
 - exemple de sortie 767
 - Liste des caractéristiques des travaux 757
 - exemple de sortie 764
 - liste des commandes 756
 - Liste des historiques des travaux 758
 - exemple de sortie 768
 - Liste des invites 757
 - exemple de sortie 767
 - Liste des paramètres 757
 - exemple de sortie 768
 - Liste des ressources 757
 - exemple de sortie 768
 - modification du format de date 756
 - Récapitulatif de la production planifiée 762
 - Récapitulatif de la production réelle 762
 - Références croisées 763
 - exemple de sortie 772
- génération de rapports de ligne de commande
 - configuration 793
- gestion
 - applications de charge de travail 304, 393
 - critères de correspondance 71
 - cycle de production 67

- gestion (*suite*)
 - dépendances externes de prédécesseur/successeur 71
 - objets de la base de données 163
 - travail reflet dans le plan 859
- gestion de plan
 - concepts de base 67
 - logman 110
 - personnalisation 82, 86
 - stageman 107
- gestion des agents sur les systèmes AS/400 862
- gestion des agents sur les systèmes i5/OS 862
- gestion des agents sur les systèmes IBM i 862
- gestion des événements
 - arrêt du moteur de surveillance 544
 - arrêter le serveur de traitement d'événement 543
 - démarrage du moteur de surveillance 536
 - démarrer le serveur de traitement d'événement 535
 - permuter le serveur de traitement d'événement 561
- gestion des fuseaux horaires 799
 - nom des fuseaux horaires à longueur variable 799
- gestion des objets
 - dans le plan 411
 - ligne de commande 321
- gestion des travaux et des agents dans l'environnement dynamique IBM i 861
- gestion des travaux et des agents sur AS/400 861
- gestion des travaux et des agents sur i5/OS 861
- gestion des travaux et des agents sur IBM i 861
- gestion du plan
 - affectation de la console 453
 - affichage de l'aide 463
 - affichage de l'état du plan de production 537
 - affichage de la bannière de conman 567
 - affichage de la liste des successeurs d'un travail 472
 - affichage de travaux ou de flots de travaux 459
 - affichage des dépendances de fichiers 498
 - affichage des informations relatives au domaine 497
 - affichage des informations relatives aux flots de travaux 525
 - affichage des informations relatives aux invites 520
 - affichage des informations relatives aux postes de travail 490
 - affichage des informations relatives aux ressources 523
 - affichage des informations relatives aux travaux 501
 - affichage des invites non résolues 474

- gestion du plan (*suite*)
 - affichage des plans traités 470
 - ajout d'une dépendance à des flots de travaux 442
 - ajout d'une dépendance à des travaux 439
 - annulation de flots de travaux 448
 - annulation de travaux 446
 - arrêt de l'application Dynamic Workload Broker 542
 - arrêt des processus d'un poste de travail 531, 538
 - arrêt des processus d'un poste de travail BEHINDFIREWALL 540
 - arrêt des travaux 464
 - arrêter le serveur d'applications 541
 - changement de gestionnaire de domaine 562
 - commande ignorée 454
 - confirmation de l'achèvement d'un travail 450
 - définition du niveau de message 453
 - Demander une reconnaissance avec accès simultané (bulk discovery) 446
 - démarrage de l'application Dynamic Workload Broker 535
 - démarrage des processus d'un poste de travail 532
 - démarrer le serveur d'applications 534
 - envoi de messages à l'opérateur 564
 - liaison de postes de travail 468
 - libération des flots de travaux de dépendances 478
 - libération des travaux de dépendances 476
 - limitation des travaux s'exécutant dans un flot de travaux 467
 - Mise à jour du fichier de configuration de surveillance 458
 - modification de la priorité 445
 - modification de la priorité minimale 462
 - modification des travaux s'exécutant sur un poste de travail 465
 - modification des unités de ressource 488
 - modification du mot de passe utilisateur 444
 - obtenir les moniteurs actifs 490
 - réexécution de travaux 481
 - réexécution des commandes 475
 - réexécution des successeurs d'un travail 484
 - réponse aux invites 480
 - sélection d'un plan traité 489
 - sortie de conman 461
 - soumission de commandes sous forme de travaux 545
 - soumission de fichiers sous forme de travaux 549
 - soumission de travaux 552
 - soumission des flots de travaux 556
 - suppression d'une dépendance à des flots de travaux 457

- gestion du plan *(suite)*
 - suppression d'une dépendance à des travaux 455
 - suppression d'une liaison entre postes de travail 565
- gestionnaire de console
 - messages 815
- gestionnaire de domaine 11
- gestionnaire de domaine maître 11
- gestionnaire de domaine maître de sauvegarde 11
- group
 - élément 912
- groupe de cycle d'exécution 216
 - syntaxe de définition de travail 943
- groupName
 - élément 929

H

- help, commande 357, 463
- hold
 - état du travail 427
 - flot de travaux, état 434
- hold, état 476
- HOME, variable 54, 55, 62
- HOMEDRIVE, variable 54
- HOMEPAATH, variable 54
- hostName
 - élément 906
- hôte
 - agents étendus 174
 - définition de poste de travail 174

I

- IBM Workload Scheduler
 - architecture 37
 - concepts de base 1
 - contrôle du traitement des travaux 23
 - définition d'activités 22
 - environnement d'exécution 21
 - exécution de commandes sous Windows 37
 - exécution de la gestion des événements 29
 - gestion de la production 28
 - interfaces utilisateur 30
 - mise en route rapide 33
 - object 1
 - présentation 1
 - processus 37
 - réseau 20
- icalendar
 - cycle d'exécution 219, 245, 268
- ID serveur
 - définition de poste de travail 180
 - mailman, processus 38
- identification des instances de flot de travaux
 - dans le plan 70
 - at 70
 - scheddateandtime 70
 - plan de préproduction 70

- ignore
 - classe de postes de travail, définition 185
- importation
 - application de charge de travail 398
- importation, application de charge de travail 398
- informations relatives aux ressources logiques
 - extraction 741
- Installation de plug-in
 - instances d'agent dynamique 645
- Installation des plug-in
 - instances d'agents 645
- instances d'agent dynamique
 - création
 - aucune installation 645
 - Installation de plu-ins 645
 - Installation des plug-in 645
- instances de travail
 - archivage 736
- instructions JSDL 893
- Intégration avec IBM Tivoli Monitoring 6.1
 - bulk_discovery 446
- intégrité des données
 - table de variables 131
- interactive
 - définition de travail 192
- Intercom.msg 45
- interface 806
- interface de ligne de commande
 - définition 65
- interfaces utilisateur
 - API Java 32
 - Application Lab 31
 - composer 32
 - conman 32
 - Dynamic Workload Console 30
 - Interface des services Web 32
 - optman 32
 - planman 32
 - plug-in 32
- intervalle de confiance 112, 569
- intro
 - état du travail 427
- invite ad-hoc 10
- invite de commande Windows
 - niveau de privilège pour l'exécution de commandes IBM Workload Scheduler 37
- invite de fin anormale 10
- invite de reprise 10
- invite globale 10
- invite locale 10
- invite nommée 10
- invoker
 - élément 930
- ismaster
 - définition de domaine 186

J

- j2ee
 - élément 930
- JAASAuthenticationAlias
 - élément 934

- jbextract, commande 775
- jms
 - élément 930
- jndiHome
 - élément 931
- JnextPlan
 - conman startappserver 91
 - CreatePostReports 92
 - MakePlan 91
 - SwitchPlan 92
 - UpdateStats 92
- job statement
 - dans les flots de travaux 254
- Job Submission Description Language (JSDL) 893
- jobDefinition
 - élément 899
- jobinfo, commande 701
- jobman
 - variables d'environnement 54
- Jobman
 - messages 816
- jobman, processus 39
 - limit cpu 39
- jobprop, commande 735
- jobs
 - allocation 934
 - création 934
 - définition 934
 - jobs
 - consommables, propriétés 934
 - optimisables, propriétés 934
 - optimization 934
- jobstdl, commande 703, 708
- join
 - critères de correspondance 256
- join, mot clé 256
- joindre
 - dépendances 831
- jour
 - Cycle d'exécution 218, 244, 267
- JSDL (Job Submission Description Language) 893
- jsuntil, mot clé 257

K

- keyjob, mot clé 259
- keysched, mot clé 260
- kill, commande 464

L

- LANG, variable 54, 55
- langage de planification 226
- LD_LIBRARY_PATH, variable 55
- LD_RUN_PATH, variable 55
- liaison
 - définition 845
- licence
 - licensetype 172
- licensetype
 - licence 172
- ligne de commande
 - compositeur 30
 - conman 30

- ligne de commande (*suite*)
 - optman 30
- ligne de commande planman
 - création d'un plan d'essai 99
 - création d'un plan prévisionnel 101
 - Déploiement de règles 102
 - déverrouillage du plan 103
 - extension d'essai 100
 - extraction des informations du plan 98
 - paramètres_connexion 94
 - plan intermédiaire 95, 97
 - réinitialisation du plan 103
 - réplication des données de plan 105
 - suppression de plan 104
 - surveiller la réplication 106
- limit, mot clé 260
- limit cpu
 - jobman, processus 39
- limit cpu, commande 465
- limit sched, commande 467
- link, commande 468
- list, commande 358
- liste de contrôle d'accès de sécurité
 - définition de listes de contrôle d'accès de sécurité 306
- listsucc, commande 472
- listsym, commande 470
- LOCAL_RC_OK, variable 58, 62
- lock, commande 365
- logicalResource
 - élément 911
- logique conditionnelle 827
- logmanMinMaxPolicy
 - variable 89
- logmanSmoothPolicy
 - variable 89
- LOGNAME, variable 54, 55

M

- maestro, commande 706
- MAESTRO_OUTPUT_STYLE, variable 54, 55
- MAIL_ON_ABEND, variable 58, 59, 62
 - sur un poste de travail Windows 62
- Mailbox.msg 46
- mailman, processus 38
 - ID serveur 38
- maintenance de tables de base de données
 - movehistorydata, commande 736
- makecal, commande 706
- MakePlan
 - JnextPlan 91
- manager
 - définition de poste de travail 176
- matching, mot clé 261
- maxdur, mot clé 262
- maximumResourceWaitingTime
 - élément 920
- maxLen
 - variable 86
- mécanisme de verrouillage
 - table de variables 131
- members
 - classe de postes de travail, définition 185, 308, 309
 - définition de poste de travail 180
- même jour
 - critères de correspondance 71
 - follows 248
 - follows sameday 71
 - join 256
- message
 - élément 932
- messages
 - compiler 816
 - composer 816
 - gestionnaire de console 815
 - Jobman 816
- méthode d'accès 806
 - agent
 - syntaxe 806
 - agent dynamique
 - fichier d'options 809
 - agent étendu
 - fichier d'options 809
 - agent IBM Workload Scheduler for z/OS
 - fichier d'options 809
 - interface 806
 - options de tâche 806
- méthode d'accès pour agent dynamique
 - présentation 805
- méthode d'accès pour agent étendu
 - présentation 805
- metronome, commande 708, 723
- migration 130
- mindur, mot clé 264
- minLen
 - variable 86
- mirrorbox.msg 46
- mise à niveau 130
- modèle d'octroi de licence
 - modèle de tarification 515
 - poste de travail réel 515
- modèles
 - de définition d'objet de planification 166
- modification de définitions de travail 670, 671
- modify, commande 370
- Monbox.msg 46
- Moncmd.msg 46
- mot clé \$MANAGER 174
- mot clé \$MASTER 174
- mot clé critical 236
- mot de passe
 - définition sur agent dynamique 645
 - élément 923, 929, 933
 - résolution sur agent dynamique 645
 - types de travail avec fonctions avancées 645
- mot de passe local
 - gestion sur des agents dynamiques 738
- moteur distant
 - dépendance croisée 854
 - mode de liaison 853
 - poste de travail 177
 - Poste de travail 166

- mots clés
 - at 228, 233
 - carryforward 235
 - comment 235
 - confirmed 236
 - critical 236
 - deadline 237
 - description 238
 - draft 238
 - end 239
 - every 239
 - utilisés dans les définitions de flots de travaux 239
 - utilisés dans les définitions de travaux 240
 - except 244
 - follows 248
 - freedays 252
 - join 256
 - jsuntil 257
 - keyjob 259
 - keysched 260
 - limite 260
 - matching 261
 - maxdur 262
 - mindur 264
 - needs 265
 - on 267
 - onlate 273
 - onoverlap 274
 - opens 274
 - priority 277
 - prompt 278
 - schedtime 279
 - schedule 281
 - startcond 283
 - statisticstype custom 282
 - table de variables 291
 - timezone 286
 - travail nop dans un flot de travaux 266
 - until 287
 - validfrom 290
- mots clés des flots de travaux
 - at 228, 233
 - carryforward 82, 235
 - comment 235
 - confirmed 236
 - critical 236
 - description 238
 - draft 238
 - échéance 237
 - end 239
 - every 239
 - utilisés dans les définitions de flots de travaux 239
 - utilisés dans les définitions de travaux 240
 - except 244
 - follows 248
 - freedays 252
 - job statement 254
 - join 256
 - jsuntil 257
 - keyjob 259
 - keysched 260
 - le 273

mots clés des flots de travaux *(suite)*

- limite 260
- matching 261
- maxdur 262
- mindur 264
- needs 265
- nop 266
- on 267
- onmaxdur 262
- onmindur 264
- onoverlap 274
- onuntil 222, 224, 258, 287
- opens 274
- priority 277
- prompt 278
- schedtime 279
- schedule 281
- startcond 283
- statisticstype custom 282
- table de variables 291
- timezone 286
- until 287
- validfrom 290
- mots clés réservés
 - pour les définitions d'utilisateur 165
 - pour les flots de travaux 165
 - pour les postes de travail 165
- mots réservés
 - pour les définitions d'utilisateur 165
 - pour les flots de travaux 165
 - pour les postes de travail 165
- movehistorydata, commande 736

N

- name
 - fichier d'options globales 809
 - fichier d'options locales 809
- needs, mot clé 265
- netman, processus 38
- netmth, méthode d'accès 819
- NetReq.msg 46
- new, commande 375
- ni veau de journalisation sur i5/OS 863
- niveau de journalisation sur AS/400 863, 864
- niveau de journalisation sur i5/OS 864
- niveau de journalisation sur IBM i 863, 864
- noms de chemin d'accès, notation xix
- noms de répertoire, notation xix
- nop, mot clé 266
- notation
 - noms de chemin d'accès xix
 - type de caractère xix
 - variables d'environnement xix
- nouveau programme d'exécution 669
- nouveaux exécuteurs 190
 - application des accès
 - SmartCloud 592
 - modèle 580
 - planification 667
 - scheduling 13, 575
 - travail de services Web 618
 - travail de transfert de fichier 598
 - travail de type commande
 - distante 582

nouveaux exécuteurs *(suite)*

- travail de type services Web
 - RESTful 620
- travail JobManagement 636
- travail JSR 352 Java Batch 629
- travail MQTT 631
- travail Soumission d'un flot de travaux 641
- travail VariableTable 633
- travaux AS/400 585
- travaux d'application des accès OSLC 610
- travaux d'automatisation OSLC 608
- travaux de base de données 612
- travaux de la méthode d'accès 590
- travaux exécutables 588
- travaux i5/OS 585
- travaux IBM i 585
- Travaux J2EE 625
- travaux Java 624
- travaux JCL 582
- travaux MSSQL 615
- nouveaux plug-in 580, 669
 - application des accès
 - SmartCloud 592
 - modèle 580
 - Services Web RESTful 620
 - travail de services Web 618
 - travail de transfert de fichier 598
 - travail de type commande
 - distante 582
 - travail JobManagement 636
 - travail JSR 352 Java Batch 629
 - travail MQTT 631
 - travail Soumission d'un flot de travaux 641
 - travail VariableTable 633
 - travaux AS/400 585
 - travaux d'application des accès OSLC 610
 - travaux d'automatisation OSLC 608
 - travaux de base de données 612
 - travaux de la méthode d'accès 590
 - travaux exécutables 588
 - travaux i5/OS 585
 - travaux IBM i 585
 - Travaux J2EE 625
 - travaux Java 624
 - travaux JCL 582
 - travaux MSSQL 615

O

- objective
 - élément 917
- objets de base de données 391
 - add, commande 338
 - affichage de la bannière de composer 391
 - agendas 205
 - authenticate, commande 340
 - classes de postes de travail 184
 - commande twstrace 753
 - Commande wappman 406
 - continue, commande 340
 - create, commande 352

objets de base de données *(suite)*

- Définition de domaine de sécurité 307
- Définition de listes de contrôle d'accès de sécurité 306
- Définition du rôle de sécurité 309
- delete, commande 341
- display, commande 345
- domaines 185
- edit, commande 351
- exit, commande 352
- extract, commande 352
- flot de travaux 226
- groupe de cycle d'exécution 216
- help, commande 357
- JobManagement 636
- jobs 187
- JSR 352 Java Batch 629
- list, commande 358
- lock, commande 365
- modify, commande 370
- MQTT 631
- new, commande 375
- postes de travail 166
- print, commande 358
- prompts 10, 213
- redo, commande 377
- règles d'événement 291
- rename, commande 379
- replace, commande 381
- ressources 215
- services Web 618
- table de variables 211
- transfert de fichier 598
- travaux d'application des accès OSLC 610
- travaux d'automatisation OSLC 608
- travaux de base de données 612
- travaux de la méthode d'accès 590
- travaux exécutables 588
- travaux IBM i 585
- Travaux J2EE 625
- travaux Java 624
- travaux JCL 582
- travaux MSSQL 615
- travaux reflet 596
- travaux Soumission d'un flot de travaux 641
- unlock, commande 383
- update, commande 388
- users 200
- validate, commande 390
- variables 206
- VariableTable 633
- on
 - Cycle d'exécution 267
- on, mot clé 267
 - cycle d'exécution 267
- onlate
 - travail dans un flot de travaux 273
- onlate, mot clé 273
 - Cycle d'exécution 273
- onmaxdur, mot clé 262
- onmindur, mot clé 264
- onoverlap, mot clé 274
- onuntil, mot clé 222, 224, 258, 287
- opens, mot clé 274

- operatingSystem
 - élément 909
- opérations de base de données 669
 - modèles de fichiers JSDL 580
- opérations de transfert de fichiers 669
 - modèles de fichiers JSDL 580
- opérations Java 669
 - modèles de fichiers JSDL 580
- optimisation des ressources
 - activation dans les travaux 916
 - EWLM, intégration 916
- optimisation des travaux
 - EWLM, intégration 916
- optimization
 - élément 916
- option locale
 - mm retry link, variable 114
- options
 - untilDays 87
- options de réexécution avancées 26
- options de reprise
 - continuer 26
 - rerun 26
 - stop 26
 - travaux de reprise 26
- options de tâche
 - méthode d'accès 806
- options globales
 - carryforward, variable 108
 - carryStates 82
 - enLegacyStartOfDayEvaluation
 - variable 90, 800
 - enTimeZone, variable 90, 799
 - untilDays 87
 - Variable 'enLegacyId' 88
 - variable carryStates 87
 - variable enCarryForward 82, 87
 - variable enCFInterNetworkDeps 87
 - variable enCFResourceQuantity 87
 - variable enPreventStart 88
 - variable logmanMinMaxPolicy 89
 - variable logmanSmoothPolicy 89
 - variable maxLen 86
 - variable minLen 86
 - variable startOfDay 87, 800
- optman
 - ligne de commande 30
- orderedCandidatedWorkstations
 - élément 905
- ordinateur
 - ressource 934
- ordinateurs associés à une ressource
 - extraction 741
- orphelines
 - dépendances 75
- ou
 - élément 913

P

- parameter 19
- parameters
 - dans les définitions de travail 200
 - élément 922
- paramètre global 130
 - définition 206
 - table de variables 129

- paramètre local
 - database 710
 - définition 206
 - exporter 710
 - gestion 710
 - importation dans 710
- paramètres communs 130
- paramètres_connexion
 - définition 65
- paramètres de journal sur AS/400 863, 864
- paramètres de journal sur i5/OS 863, 864
- paramètres de journal sur IBM i 863, 864
- paramètres des travaux enfants pour les performances sur AS/400 865
- paramètres des travaux enfants sur AS/400 pour les performances 865
- paramètres des travaux enfants sur i5/OS pour les performances 865
- paramètres des travaux enfants sur IBM i pour les performances 865
- parent
 - définition de domaine 186
- parms, commande 710
- PATH, variable 55
- paxtract, commande 778
- pend
 - état du travail 427
- performances de la base de données
 - amélioration 736
- permutation d'agents étendus
 - mot clé \$MANAGER 174
 - mot clé \$MASTER 174
- personnalisation de la charge de travail
 - utilisation de la table de variables 129
- physicalMemory
 - élément 908
- plan
 - mise en route rapide 33
- plan à long terme
 - plan de préproduction 69
- plan d'essai
 - création 99
 - description 84
 - extension 100
- plan de préproduction
 - description 69
 - FNCJSI 70
 - plan à long terme 69
 - suppression de plan 104
- plan de production
 - démarrage du traitement 113
 - description 82
 - déverrouillage du plan 103
 - extraction des informations 98
 - fichier Symphony 67, 91
 - générations 91, 92
 - JnextPlan 67, 91, 92
 - réinitialisation du plan 103
 - réplication de données 105
 - surveiller la réplication 106
 - Symphony, fichier 92
 - traitement automatique 114

- plan intermédiaire
 - extension avec 97
 - générations 95
- plan prévisionnel
 - calcul de première heure de début 85
 - création 101
 - description 85
- PlanBox.msg 46
- planification des types de travaux avec options avancées 575, 667
- planification dynamique 11, 13, 14, 575, 667
 - définition de la tâche de travail 190
 - définition de poste de travail 166
 - définition de travail 187, 581
 - types de travaux avec options avancées 575, 667
- planman deploy 150
- plug-in 32
- plug-in de travaux d'application 190
 - planification 667
 - scheduling 13, 575
- pool 11, 14, 667
 - définition 166
 - définition de l'utilisateur
 - Windows 203
 - Poste de travail 177
- pool dynamique 11, 14, 667
 - définition 166
 - définition de l'utilisateur
 - Windows 203
 - Poste de travail 177
- pools
 - planification des types de travaux avec options avancées 575, 667
- pools dynamiques
 - planification des types de travaux avec options avancées 575, 667
- poste de travail
 - type d-pool 177
 - type de pool 177
 - type de pool dynamique 177
- Poste de travail
 - agent tolérant aux pannes 11
 - boîte aux lettres, fichiers
 - NetReq.msg 45
 - classe 15
 - création 166
 - définition 166
 - gestionnaire de domaine 11
 - gestionnaire de domaine maître 11
 - gestionnaire de domaine maître de sauvegarde 11
 - pool 11
 - pool dynamique 11
 - poste de travail de moteur distant 11
 - processus 37
 - type de moteur distant 166, 177
- poste de travail de moteur distant 11, 15, 21
 - définition 845, 850
- postes de travail dynamiques 11, 166
- prédécesseur
 - critères de correspondance 74
 - successeur 69, 74
- prédécesseur suspendu
 - critères de correspondance 75

- prédécesseur suspendu (*suite*)
 - dépandances orphelines 75
 - successeur 75
- présentation
 - agent dynamique 805
 - agent étendu 805
 - méthode d'accès pour agent
 - dynamique 805
 - méthode d'accès pour agent étendu 805
- prêt
 - état du travail 427
- print, commande 358
- priority
 - élément 920
- priority, mot clé 277
- procédure mémorisée de base de données
 - modèles de fichiers JSDL 580
 - travaux de base de données 669
 - modèles de fichiers JSDL 580
- processus
 - batchman 39
 - jobman 39
 - mailman 38
 - monman 38
 - netman 38
 - ssmagent 38
 - writer 38
- processus de liaison
 - pour un travail reflet distribué 853
- processus de poste de travail 38
 - arborescence des processus sous
 - UNIX 40
 - arborescence des processus sous
 - Windows 41
 - arrêt 43
 - batchman 39
 - communication interprocessus 45
 - début de la journée
 - établissement de la communication 49
 - démarrage 43
 - gestion des changements d'état des
 - travaux 50
 - jobman 39
 - mailman 38
 - ID serveur 38
 - monman 38
 - netman 38
 - writer 38
- processus monman 38
- Programme Conman
 - Codes retour de ligne de commande
 - Conman 436
- promotion d'un travail 118
- promotion d'un travail dynamique 674
- promotion d'un travail dynamique
 - critique 674
- promotion de travail sur des pools
 - dynamiques 674
- promotion de travaux de courtier 674
- promotion des travaux 118
 - variables d'environnement 674
- prompt
 - ad-hoc 10
 - fin anormale 10
 - globale 10

- prompt (*suite*)
 - locale 10
 - nommée 10
 - reprise 10
- prompt, mot clé 278
- propriétés
 - élément 913
- propriétés du poste de travail 496
- propriétés locales 53
- protocole
 - définition de poste de travail 180
- prxtract, commande 776

R

- r11xtract, commande 779
- rapports de traitement par lots
 - exemple de scénario 792
 - fonctions de trace 797
 - journaux 797
- ready
 - flot de travaux, état 434
- recall, commande 474
- recovery
 - définition de travail 195
- recoveryActions
 - élément 921
- redo, commande 377, 475
- réexécution avec les successeurs 26
- règle 19
- règle d'événement 19
- règles d'événement
 - exemples de scénarios 142, 152
 - instances 161
 - option de délai d'attente 151
 - substitution de variables 151
- réinitialisation du plan
 - ligne de commande planman 103
- relatedResources
 - élément 904
- relation d'affinité
 - définition 671
- relationship
 - élément 915
- release, commande 713
- release job, commande 476, 480
- release sched, commande 478
- rem-eng
 - poste de travail 177
- rename, commande 379
- rep1, commande 757
- rep11, commande 761
- rep2, commande 757
- rep3, commande 757
- rep4a, commande 757
- rep4b, commande 757
- rep7, commande 758
- rep8, commande 759
- replace, commande 381
- réplication de plan
 - dépandances conditionnelles 842
- répliquer le plan 28
- reporter
 - travail distant 859
 - travail reflet 859
- reptr, commande 762

- requirements
 - définition de poste de travail 180
- rerun, commande 481
- rerunsucc, commande 484
- résolution
 - variable 133
- resource
 - logique 19
 - ordinateur 934
 - physique 19
 - scheduling 19
- resource, commande 488, 741
 - exécution à partir d'un agent
 - configuration de
 - CLIconfig.properties 750
 - exigence 750
- resources
 - élément 903
 - optimisables 935
- ressource associée
 - ressource logique 934
 - système d'exploitation 934
 - système de fichiers 934
 - système réseau 934
- ressource de planification 19
- ressource logique 19
 - ressource associée 934
- ressource physique 19
- retard, état 237
- retract, commande 779
- rmstdlist, commande 717
- rôle de sécurité
 - définition de rôles de sécurité 309

S

- sched
 - état du travail 427
- schedtime, mot clé 279
- schedule, mot clé 281
- scheduling
 - élément 919
- script
 - élément 926
- script de configuration djobmanrc 63
- script de configuration jobmanrc 57, 61
- scriptname
 - définition de travail 189
- secureaddr
 - définition de poste de travail 173
- sécurité
 - tables de variables 132
- securitylevel 179
 - définition de poste de travail 179
- sendevent, commande 719, 751
- Server.msg 46
- serveur d'applications
 - arrêt 541
- ServiceNow
 - actions liées à des événements 887
- setsym, commande 489
- SHELL_TYPE, variable 59
- showcpus, commande 490
- showdomain, commande 497
- showexec, commande 720
- showfiles, commande 498
- showjobs, commande 501

showprompts, commande 520
 showresources, commande 523
 showschedules, commande 525
 shutdown
 commande d'utilitaire 721
 shutdown, commande 531
 ShutDownLwa
 commande d'utilitaire 722
 soumission d'un flot de travaux
 travaux Soumission d'un flot de
 travaux 641
 speed
 élément 908
 SPSS
 ELAB_JOB_STAT_JS
 exécution 573
 identification des incidents 573
 importation et configuration 571
 installation 569
 marquage des travaux 571
 utiliser pour prévoir la durée du
 travail 569
 stageman
 carryforward 82
 SwitchPlan 107
 start, commande 532
 startappserver command 534
 startbrokerapp, commande 535
 startcond, mot clé 283
 starteventprocessor, commande 535
 startmon, commande 536
 startOfDay
 variable 87, 800
 StartUp, commande 722
 statisticstype custom, mot clé 282
 statistiques avancées 569
 ELAB_JOB_STAT_JS 571, 573
 marquage des travaux 571
 utiliser pour prévoir la durée du
 travail 569
 status
 retard 237
 status, commande 537
 statut de processus du poste de
 travail 493
 statut des liens d'un poste de travail 493
 stop, commande 538
 stop; progressive, commande 540
 stopappserver, commande 541
 stopbrokerapp, commande 542
 stoeventprocessor, commande 543
 stopmon, commande 544
 streamlogon
 définition d'utilisateurs Windows 200
 définition de travail 191
 stringVariable
 élément 901
 stuck
 flot de travaux, état 434
 submit docommand, commande 545
 submit file, commande 549
 submit job
 code retour 436
 submit job, commande 552
 submit sched, commande 556
 succ
 état du travail 427

succ (*suite*)
 flot de travaux, état 434
 successeur
 critères de correspondance 74
 prédécesseur 69, 74
 prédécesseur suspendu 75
 succp
 état du travail 427
 suppression de plan
 ligne de commande planman 104
 switcheventprocessor, commande 561
 switchmgr, commande 562
 SwitchPlan
 JnextPlan 92
 Symphony, fichier
 JnextPlan 92
 plan de production 92
 syntaxe
 agent
 méthode d'accès 806
 agent étendu
 tâche d'obtention d'un statut 814
 tâche de vérification d'un
 fichier 813
 SystemDrive, variable 54
 système d'exploitation
 ressource associée 934
 système d'exploitation Windows
 caractères spéciaux, gestion 111
 système de fichiers
 ressource associée 934
 système réseau
 ressource associée 934
 systèmes d'exploitation Windows
 niveau de privilège pour l'exécution
 de commandes IBM Workload
 Scheduler 37
 SystemRoot, variable 54

T

table de variables 20
 définition 211
 définition de table de variables 172
 intégrité des données 131
 mécanisme de verrouillage 131
 par défaut 131
 utilisation 129
 table de variables par défaut
 utilisation 131
 tables de variables
 migration du fichier de sécurité 130
 sécurité 132
 tâche
 définition de travail 190
 tâche d'obtention d'un statut
 agent étendu
 syntaxe 814
 tâche de vérification d'un fichier
 agent étendu
 syntaxe 813
 tasktype
 définition de travail 191
 tcpaddr
 définition de poste de travail 173
 technique, formation xviii
 tellop, commande 564
 TEMP, variable 54
 temps d'exécution moyen 112
 temps d'exécution prévu 112, 569
 timezone
 dans les flots de travaux 286
 définition de poste de travail 174
 timezone, mot clé 286
 TIVOLI_JOB_DATE, variable 54, 56
 TMPDIR, variable 54
 TMPTEMP, variable 54
 TPM (Provisioning), définition de travail
 procédure préalable 592
 tpmaction
 élément 922
 tpmaddress
 élément 923
 traitement automatique
 plan de production 114
 traitement automatique du plan
 final, flot de travaux 114
 traitement des travaux
 configuration 63
 travail 2
 calcul du temps d'exécution 112
 prévision du temps d'exécution 569
 travail dans un flot de travaux
 onlate 273
 travail de branche
 actions correctives 996
 actions correctives dans l'ordre 1001
 affinage de la recherche de
 modèle 995
 ajout du travail de branche à la base
 de données 1007
 ajout du travail de signal à la base de
 données 1007
 basé sur la condition, définition 967
 basée sur l'action, définition 996
 branchement long 970
 branchement multiple 974
 commande grep 995
 conditions complexes 978
 configuration requise sous
 UNIX 617, 1006
 configuration requise sous
 Windows 617, 1006
 dans un flot de travaux 1009
 demande de confirmation 1003
 évaluation du journal de travail 978
 fin anormale de parent 976
 libération de branchement 996
 mise en pause du branchement 996
 paramètre
 IS_CASE_SENSITIVE_i 1016
 paramètres fixes 1012
 paramètres indexés 1012
 points forts 966
 principaux concepts 961
 processus d'action 966
 processus d'évaluation 965
 propagation du status ABEND au flot
 de travaux 1010
 recherche de modèle 979
 recherche de modèle étendu 984,
 987, 989
 recherche de modèle négatif 981
 signalement d'une action 1003

- travail de branche (*suite*)
 - simple branchement 968
 - sous-conditions 978
 - sous-conditions multiples 992
 - termes 962
 - travail de branche
 - script de shell 1007
 - types de paramètres 1012
 - UNIX, configuration requise 617, 1006
 - Windows, configuration requise 617, 1006
- travail distant
 - définition 845
 - échec 859
 - reporter 859
 - transition d'état lors de la récupération 859
- travail enfant sur i5/OS 865
- travail enfant sur IBM i 865
- travail Java générique 669
 - modèle 580
- travail reflet 2
 - définition 596, 845, 850
 - échec 859
 - état d'echec 859
 - gestion dans le plan en cours 859
 - lors de la récupération d'un travail
 - distant 859
 - modification d'état après liaison 857
 - reporter 859
- travaux, états
 - abexp 426
 - add 426
 - attente 427
 - done 426
 - error 427
 - exec 427
 - extrn 427
 - fail 427
 - fence 427
 - fin anormale 426
 - hold 427
 - intro 427
 - pend 427
 - prêt 427
 - sched 427
 - succ 427
 - succp 427
- travaux critiques
 - fichier de sécurité 121
 - options globales 118
 - options locales 120
- travaux de base de données DB2 612
- travaux de base de données
 - dynamiques 187, 581
- travaux de base de données MSSQL 612
- travaux de base de données
 - Netezza 612
- travaux de base de données Oracle 612
- travaux de la méthode d'accès 580
- travaux de service Web
 - dynamiques 187, 581
- travaux de services Web 669
 - modèles de fichiers JSDL 580
- travaux de transfert de fichier
 - dynamiques 187, 581

- travaux de transfert de fichiers 669
 - modèles de fichiers JSDL 580
- travaux dynamiques 166, 187, 190, 581
 - application des accès
 - SmartCloud 592
 - travail de services Web 618
 - travail de transfert de fichier 598
 - travail de type commande
 - distante 582
 - travail de type services Web
 - RESTful 620
 - travail JobManagement 636
 - travail JSR 352 Java Batch 629
 - travail MQTT 631
 - travail MSSQL 615
 - travail Soumission d'un flot de travaux 641
 - travail VariableTable 633
 - travaux AS/400 585
 - travaux d'application des accès
 - OSLC 610
 - travaux d'automatisation OSLC 608
 - travaux de base de données 612
 - travaux de la méthode d'accès 590
 - travaux exécutables 588
 - travaux i5/OS 585
 - travaux IBM i 585
 - Travaux J2EE 625
 - travaux Java 624
 - travaux JCL 582
- travaux dynamiques essentiels 674
- travaux dynamiques importants 674
- travaux exécutables 580
- travaux existants
 - amélioration 675
- travaux existants avec fonctionnalités dynamiques 675
- travaux IBM i 580
 - travaux AS400 580
- travaux J2EE 580
- travaux Java
 - modèles de fichiers JSDL 580
- Travaux Java 669
- travaux Java dynamiques 187, 581
- travaux MSSQL 580
- travaux standard
 - amélioration 675
- travaux standard avec fonctionnalités dynamiques 675
- trialsked
 - plan d'essai 84
 - plan prévisionnel 85
- TWS_PROMOTED_JOB 812
- TWS_TISDIR, variable 56
- type
 - définition de poste de travail 175
- type de système d'exploitation
 - définition de poste de travail 172
- types de ressource
 - consommables 935
- types de travaux 669
 - modèle 580
- types de travaux avec options
 - avancées 575, 667, 669, 675
 - définition 2
 - modèle 580
 - modèles de fichiers JSDL 580

- types de travaux avec options avancées (*suite*)
 - planification dynamique 2, 575, 667
 - planification statique 2
 - scheduling 13, 190
- types de travaux existants
 - définition 2
- types de travaux spécifiques 669
 - modèles de fichiers JSDL 580
- typographie, conventions xviii
- TZ, variable 54, 56

U

- UC
 - élément 907
- uintVariable
 - élément 902
- UNISON_CPU, variable 54, 56
- UNISON_DATE, variable 55
- UNISON_DATE_FORMAT, variable 56
- UNISON_DIR, variable 54, 56
- UNISON_EXEC_PATH, variable 54, 56
- UNISON_EXIT, variable 58
- UNISON_HOST, variable 55, 56
- UNISON_JCL, variable 57
- UNISON_JOB, variable 55, 56
- UNISON_JOBNUM, variable 55, 56
- UNISON_MASTER, variable 55, 56
- UNISON_RUN, variable 55, 56
- UNISON_SCHED, variable 55, 56
- UNISON_SCHED_DATE, variable 56
- UNISON_SCHED_EPOCH, variable 55, 56
- UNISON_SCHED_IA variable 55, 56
- UNISON_SCHED_ID variable 55, 56
- UNISON_SHELL, variable 55, 56
- UNISON_STDLIST, variable 55, 56, 57
- UNISON_SYM, variable 55, 56
- UNISONHOME, variable 55, 56
- UNIXTASK 191
- UNKNOWN 191
- unlink, commande 565
- unlock, commande 383
- until, mot clé 287
- untilDays
 - option 87
- update, commande 388
- UpdateStats
 - JnextPlan 92
- URI instance Dynamic Workload
 - Broker 732, 734
- USE_EXEC, variable 59
- user 20
- USERDOMAIN, variable 55
- userName
 - élément 923, 928, 933
- USERNAME, variable 55
- USERPROFILE, variable 55
- Utilisateur Windows
 - définition 203
 - exécution de travaux sur un an
 - agent 203
 - exécution des travaux sur un
 - pool 203
 - exécution des travaux sur un pool
 - dynamique 203

- Utilisateur Windows (*suite*)
 - planification sur un agent 203
 - planification sur un pool 203
 - planification sur un pool dynamique 203
- utilisation
 - table de variables 129
 - table de variables par défaut 131
- utilisation des commandes d'utilitaire pour les agents sur les systèmes AS/400 863
- utilisation des commandes d'utilitaire pour les agents sur les systèmes i5/OS 863
- utilisation des commandes d'utilitaire pour les agents sur les systèmes IBM i 863

V

- valeur précédente la plus proche
 - critères de correspondance 72
 - follows 248
 - follows previous 72
 - join 256
- valeurs d'attribut des objets
 - spécification des valeurs d'attribut des objets 317
- validate, commande 390
- validation des données de la base de données 580, 669
- validfrom, mot clé 290
- value
 - élément 927
- variable 20
 - dans les définitions de travail 200
 - définition sur agent dynamique 645
 - élément 928
 - résolution 133
 - résolution sur agent dynamique 645
 - types de travail avec fonctions avancées 645
- Variable
 - définition 206
- variable locale
 - gestion sur des agents dynamiques 738
- variable POSIXHOME 62
- variable TWS_PROMOTED_JOB 54, 56
- variables
 - ATSCRIPT 679
 - carryforward 108
 - carryStates 82, 87
 - COMPUTERNAME 54
 - élément 901
 - enCarryForward 82, 87
 - enCFInterNetworkDeps 87
 - enCFResourceQuantity 87
 - enLegacyId 88
 - enLegacyStartOfDayEvaluation 90, 800
 - enPreventStart 88
 - enTimeZone 90, 799
 - exportées localement par jobmanrc 57, 60, 61, 63
 - exportées sous UNIX 55
 - HOME 54, 55, 62

- variables (*suite*)
 - HOMEDRIVE 54
 - HOMEPATH 54
 - LANG 54, 55
 - LD_LIBRARY_PATH 55
 - LD_RUN_PATH 55
 - LOCAL_RC_OK 58, 62
 - logmanMinMaxPolicy 89
 - logmanSmoothPolicy 89
 - LOGNAME 54, 55
 - MAESTRO_OUTPUT_STYLE 54, 55
 - MAIL_ON_ABEND 58, 59, 62
 - maxLen 86
 - minLen 86
 - PATH 55
 - POSIXHOME 62
 - SHELL_TYPE 59
 - startOfDay 87, 800
 - SystemDrive 54
 - SystemRoot 54
 - TEMP 54
 - TIVOLI_JOB_DATE 54, 56
 - TMPDIR 54
 - TMPTMP 54
 - TWS_PROMOTED_JOB 54, 56
 - TWS_TISDIR 56
 - TZ 54, 56
 - UNISON_CPU 54, 56
 - UNISON_DATE 55
 - UNISON_DATE_FORMAT 56
 - UNISON_DIR 54, 56
 - UNISON_EXEC_PATH 54, 56
 - UNISON_EXIT 58
 - UNISON_HOST 55, 56
 - UNISON_JCL 57
 - UNISON_JOB 55, 56
 - UNISON_JOBNUM 55, 56
 - UNISON_MASTER 55, 56
 - UNISON_RUN 55, 56
 - UNISON_SCHED 55, 56
 - UNISON_SCHED_DATE 56
 - UNISON_SCHED_EPOCH 55, 56
 - UNISON_SCHED_IA 55, 56
 - UNISON_SCHED_ID 55, 56
 - UNISON_SHELL 55, 56
 - UNISON_STDLIST 55, 56, 57
 - UNISON_SYM 55, 56
 - UNISONHOME 55, 56
 - untilDays 87
 - USE_EXEC 59
 - USERDOMAIN 55
 - USERNAME 55
 - USERPROFILE 55
 - variables locales 61, 63, 322
- variables, environnement, notation xix
- variables d'environnement
 - promotion des travaux 674
- variables d'environnement, notation xix
- variables de promotion de courtier 674
- varTable, mot clé 291
- vérification de l'intégrité référentielle 333
- vérification de la santé de la boîte aux lettres 450
- version
 - commande d'utilitaire 723
- version, commande 391, 567

- version, commande (*suite*)
 - affichage de la bannière de composer 391
- virtualMemory
 - élément 908

W

- wappman, commande
 - journaux et fichiers de trace 406
- WAS
 - arrêt 541
- WebSphere Application Server
 - arrêt 43, 541
 - démarrage 43
 - infrastructure 37
- WINDOWSTASK 191
- workflow
 - élément 924
- writer, processus 38

X

- XATASK 191
- xref, commande 763
- rxtrct, commande 781



Numéro de programme : 5698-WSH

Imprimé en France