

IBM Tivoli NetView for z/OS
バージョン 6 リリース 2

自動操作ガイド



IBM Tivoli NetView for z/OS
バージョン 6 リリース 2

自動操作ガイド



お願い

本書および本書で紹介する製品をご使用になる前に、 697 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM Tivoli NetView for z/OS (製品番号 5697-NV6) のバージョン6リリース2 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： SC27-2846-02

IBM Tivoli NetView for z/OS

Version 6 Release 2

Automation Guide

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2014.2

© Copyright IBM Corporation 1997, 2013.

目次

図	xvii
本書について	xxi
対象読者	xxi
資料	xxi
IBM Tivoli NetView for z/OS ライブラリー	xxi
関連資料	xxiii
オンライン用語集へのアクセス	xxiv
NetView for z/OS オンライン・ヘルプの使用	xxv
マニュアルへのオンライン・アクセス	xxv
マニュアルのご注文	xxv
アクセシビリティ	xxv
Service Management Connect	xxv
Tivoli 技術研修	xxvi
Tivoli ユーザー・グループ	xxvi
ダウンロード	xxvi
サポート情報	xxvii
本書の表記規則	xxvii
書体の規則	xxvii
オペレーティング・システム依存の変数とパス	xxviii
構文図	xxviii

第 1 部 自動操作の概要 1

第 1 章 NetView 自動操作の紹介 3

NetView 自動操作とは	3
自動操作の利点	3
システムおよびネットワークの可用性の改善	4
拡張の際の制約の除去	4
オペレーターの生産性の向上	5
一貫した操作手順の確立	5
自動操作のクラス	5
システムの自動操作とネットワークの自動操作	5
単一システムと複数システムの自動操作	7
自動操作の段階	8
単一システムの自動操作の段階	8
オペレーター・インターフェースの改善	13
NetView プログラムが稼働していないシステムおよび非 SNA 装置の自動化	21
段階的アプローチの例	21

第 2 章 自動操作プロダクトの概要 25

NetView プログラム自動操作機能	25
コマンド・リストおよびコマンド・プロセッサ	25
タイマー・コマンド	28
自動タスク	28
自動化テーブル	29
メッセージ改訂テーブル	30
リソース・オブジェクト・データ・マネージャー	30
インストール・システム出口	31
MVS コマンド改訂	32

Automated Operations Network (AON)	32
状況モニター	32
オペレーティング・システム自動操作機能と NetView プログラムとの対話	33
MVS システムにおける自動化	33
I IBM Tivoli System Automation for z/OS	40
NetView インターフェースの使用例	40
分散ネットワーク	40
SNMP を使用する IP ネットワーク	40
IBM 以外のネットワーク	41
自動操作に関連した機能およびサービス	41
ワークロードの管理	41
ネットワーク・パフォーマンスの管理	41
入出力の管理	42
ストレージの管理	43
管理報告	43

第 2 部 自動化環境の作成 45

第 3 章 自動操作プロジェクトの定義 47

プロジェクト定義タスク	47
自動操作チームの編成	48
アプローチの選択	48
運用グループの参加	48
プロジェクト計画の作成	49
組織目標の明確化	49
事業目標の明確化	49
データ処理の要件の識別	50
操作環境について	50
MVS システム・ログとネットワーク・ログ	51
運用プロシージャ・ブック	52
問題管理報告書	52
ヘルプ・デスク・ログ	53
サービス・レベル・アグリーメント	53
ユーザー	53
その他のデータ処理計画	53
情報の解釈	53
自動操作の目標と目的の設定	53
自動操作に関する目標の設定	54
測定可能な目的の設定	54
コストと利益の定量化	54
理解と協力	56

第 4 章 自動操作プロジェクトの設計 57

プロジェクト設計作業	57
自動化すべきプロシージャと機能の確認	57
プロシージャと機能の優先順位付け	57
実施のためのスケジュール段階	57
標準の確立	58
設計のガイドライン	58
拡張と伝搬の設計	58
監査能力の設計	59
自動化セキュリティーの設計	60
可用性の設計	60
ソースに近い自動操作	61
単一システム上での複数 NetView プログラムの使用	61
オペレーター・インターフェースの提供	62

スタッフの教育	62
スタッフの役割変更の予測	63
テストの提供	63
問題および変更管理の提供	64
フォーカル・ポイントの選択	64
バックアップ・フォーカル・ポイントの使用	66
オペレーターの制御範囲の定義	66
第 5 章 自動操作プロジェクトの実施	69
実施作業	69
実動作業	69
第 3 部 選択環境における自動操作の計画	71
第 6 章 MVS 拡張複数コンソール・サポート・コンソールを使用した自動操作	73
NetView プログラムで EMCS コンソールを使用する	73
NetView プログラムで EMCS コンソールを使用することの利点	73
拡張複数コンソール・サポート・コンソールの使用計画	74
拡張複数コンソール・サポート・コンソールを使用可能にする	74
コンソールの命名規則の確立	74
拡張複数コンソール・サポート・コンソールの獲得	75
メッセージ改訂テーブル (MRT) またはメッセージ処理機能 (MPF) テーブルを使用した NetView 自動操作への メッセージの送信	76
拡張複数コンソール・サポート・コンソールの属性値の使用	76
経路コードの使用	76
セキュリティ・アクセスのインプリメンテーション	78
MVS メッセージ・データ・スペースの満杯によるメッセージ消失を防ぐ	78
キュー制限の超過によるメッセージの消失を防ぐ	79
MVS メッセージ・ストレージとメッセージ・キュー制限のバランス	79
サブシステム・インターフェースから拡張複数コンソール・サポート・コンソールへのマイグレーション	80
固有の名前の確立	80
各ホストでの新しいリリースの NetView プログラムへのマイグレーション	80
クロスドメイン通信のための RMTCMD コマンドと LU6.2 セッションの使用	80
MVS VARY コマンドへのオペレーター・アクセスの制限	80
EMCS コンソールの AUTO 属性の制限	81
第 7 章 MVS シスプレックスでの自動操作	83
MVS のシスプレックス	83
シスプレックスでの NetView プログラム自動操作の使用	84
シスプレックスでの自動操作の計画	84
第 1 段階EMCS コンソールと属性のメッセージ経路指定への影響	84
第 2 段階MPF のアクションと EMCS コンソールの定義の調整	85
第 3 段階NetView プログラム自動操作をシスプレックスの 1 つのシステムに集中させるかどうかの決定	85
外部メッセージの処理方法	86
第 8 章 リソース・オブジェクト・データ・マネージャーでの自動操作	87
リソース・オブジェクト・データ・マネージャーの紹介	87
RODM との対話	87
自動操作における RODM の使用	88
RODM を使用する際の利点	88
自動操作における RODM の使用計画	89
RODM で自動応答を生成するイベント・タイプの決定	89
RODM 自動操作機能について	89
第 9 章 自動化のための NetView プログラム情報の経路指定	91
NetView プログラム・インターフェース	91

オペレーティング・システムとのインターフェース	92
その他の NetView プログラムとのインターフェース	93
その他のメッセージおよびコマンド機能	93
ハードウェア・モニター・データおよび MSU 用のインターフェース	93
NetView プログラム・メッセージ経路指定	94
送信請求メッセージ	94
非送信請求メッセージ	95
メッセージ経路指定機能	96
ASSIGN コマンドによる経路指定メッセージ	96
MSGROUTE コマンドによるメッセージの経路指定	102
経路コードに基づいた EMCS コンソールへのメッセージの経路指定	102
メッセージ経路指定の流れ	103
DSIEX17 処理	104
PIPE CORRWAIT	104
ASSIGN PRI/SEC 処理	104
許可レシーバーの処理	105
DSIEX02A 処理	105
待機処理	106
自動化テーブル処理	106
DSIEX16 処理	108
ASSIGN COPY 処理	108
廃棄または表示処理	108
NetView プログラムのハードウェア・モニター・データ、および MSU の経路指定	108
ALERT-NETOP アプリケーション	111
XITCI 処理	111
ハードウェア・モニターの初期処理	111
自動化テーブル処理	112
DSIEX16B 処理	112
ハードウェア・モニターの継続処理	112
NetView プログラム・コマンドの経路指定	113
コマンドとタスクとの互換性	113
コマンド経路指定機能	114
コマンド優先順位	115

第 4 部 NetView プログラム自動操作機能. 117

第 10 章 コマンド・リストおよびコマンド・プロセッサー. 119

使用可能な言語	119
メッセージおよび MSU の取得	120
メッセージの機能	120
MSU の機能	120
情報の保管	121
グローバル変数	121
MVS のデータ・セット	122
特定のイベントの待機	123
NetView コマンド・リスト言語を使用した待機処理	123
REXX を使用した待機処理	123
PL/I および C を使用した待機処理	124
MVS のその他のコマンド・リスト機能	125
MVS コンソールへのメッセージの送信	125
ディスク、テープ、および印刷ファイルの割り振り	125
ストレージへのコマンド・リストのロード	125

第 11 章 タイマー・コマンド. 127

タイマー・コマンドの概要	127
AFTER	127

AT	128
EVERY	128
TIMER	129
CHRON	129
タスクの選択	129
タイマー・コマンドの保管と復元.	130
LIST TIMER および PURGE TIMER	131
LIST TIMER.	131
PURGE TIMER.	131
第 12 章 自動タスク	133
自動タスクの定義	133
自動タスクの活動化	134
AUTOTASK コマンドの使用	134
自動タスクと複数コンソール・サポート (MCS) コンソールの関連付け	134
自動タスクの非活動化	135
自動タスクによる自動化.	135
サブシステムの管理	135
非送信請求メッセージの処理	136
コマンドの処理.	136
タスクの開始	137
EXCMD コマンドによる自動タスクへのコマンドの送信	137
第 13 章 メッセージ改訂テーブル	139
メッセージ改訂テーブルについて.	139
メッセージ改訂テーブル・ステートメントのエレメント	139
メッセージ改訂テーブルの処理	140
メッセージ改訂テーブルの検索	140
メッセージ改訂テーブルのコーディング	141
経路コードおよび記述子コードの変更	141
DoForeignFrom ステートメント	142
END ステートメント.	142
EXIT ステートメント	142
NETVONLY ステートメント	142
OTHERWISE ステートメント	143
REVISE ステートメント.	143
SELECT ステートメント	144
UPON ステートメント	144
WHEN ステートメント	145
メッセージ改訂テーブルの例	145
メッセージ改訂テーブルの使用報告書	146
メッセージ改訂テーブルのテスト.	146
第 14 章 コマンド改訂テーブル	149
コマンド改訂テーブルについて	149
コマンド改訂テーブル・ステートメントのエレメント	149
コマンド改訂テーブルの処理	150
コマンド改訂テーブルの検索	150
コマンド改訂テーブルのコーディング	151
コマンド改訂テーブルのステートメント	151
TRACKING.ECHO ステートメント	152
ISSUE.IEE295I ステートメント	152
UPON ステートメント	153
SELECT ステートメント	154
WHEN ステートメント	154
OTHERWISE ステートメント	155

END ステートメント	155
REVISE ステートメント	155
NETVONLY ステートメント	156
WTO ステートメント	157
編集オーダー	157
コマンド改訂テーブルの例	159
コマンド改訂テーブルの使用報告書	160
コマンド改訂テーブルのテスト	161

第 15 章 自動化テーブル 163

自動化テーブルについて	163
自動化テーブル・ステートメントのエレメント	163
自動化テーブル処理	164
自動化テーブルの検索	164
自動化テーブル・ステートメントのタイプ	165
ステートメントのタイプの判別	165
ステートメントのタイプおよび処理	166
自動化テーブルのコーディング	166
BEGIN-END セクション	168
IF-THEN ステートメント	169
条件項目	175
ビット・ストリングの比較項目	234
解析テンプレートの比較項目	235
処置	240
ALWAYS ステートメント	263
%INCLUDE ステートメント	264
SYN ステートメント	265
自動化テーブルの設計上のガイドライン	266
システム・メッセージ処理の制限	266
読みやすい自動化テーブルの作成	267
BEGIN-END セクションによるステートメントのグループ化	267
複雑な比較項目の分離	269
他の自動化テーブルの組み込み	270
運用のための自動化テーブルの調整	270
同義語の使用	270
ステートメントの正しい配置	271
自動化テーブル・リストの使用	271
ALWAYS ステートメントの使用	271
CONTINUE アクションの正しい使用	272
自動化テーブルのデフォルトの設定	272
コマンド応答の自動化の制限	272
NetView プログラム終了時の自動操作	273
自動化テーブル・リストの例	273
自動化テーブル使用報告書	275
AUTOCNT コマンド	275
使用報告書の出力の例	276
複数の自動化テーブルの管理	287
始めに	287
自動化テーブル管理の使用法	288

第 16 章 ポリシー・サービスの概要 301

ポリシー・サービスの使用	302
DSITBL01 のカスタマイズ (オプション)	302
ポリシー・ファイルの定義	302
必須 NetView タスク	303
ポリシー・ファイル構文	303

ポリシー・ファイル管理.	305
ポリシー API の使用.	306
POLICY 構文.	306
ロードされるポリシー・ファイルの判別.	308
ポリシー・ファイルの構文テスト.	308
ポリシー・ファイルのロード.	309
ポリシー定義の照会.	310
ポリシー定義のグループの.	310
ポリシー定義の変更.	311
ポリシー定義の削除.	312
ポリシー定義の追加.	312
REXX API の使用.	313
タイマー API.	313
EZLETAPI.	313
EZLEQAPI.	322
EZLEDAPI.	324
EZLEQCAL.	326
第 17 章 インストール・システム出口.	329
インストール・システム出口とは?	329
インストール・システム出口 DSIEX02A.	329
BNJDSERV のインストール・システム出口 XITCI.	329
インストール・システム出口 DSIEX16 および DSIEX16B.	330
インストール・システム出口 DSIEX17.	330
第 5 部 単一システムの自動操作.	331
第 18 章 自動操作機能のセットアップ・タスク.	333
NetView システムとオペレーティング・システムの通信の確立.	333
システム自動化のための z/OS システムの準備.	333
自動タスクの定義と活動化.	340
第 19 章 メッセージの抑止とアラートのフィルター操作.	341
システム・メッセージの抑止.	341
ネットワーク・メッセージの抑止.	341
アラートのフィルター操作.	342
記録フィルター.	342
目視フィルター.	346
フィルターの迂回.	346
第 20 章 コンソールの統合.	349
コンソールの統合方法.	349
NetView と複数コンソール・サポート (MCS) コンソールとの相違点.	350
画面の処理およびメッセージの表示位置.	350
メッセージの行形式.	350
表示域機能.	350
画面の最新表示.	351
接頭部コマンド名.	351
メッセージの保留.	351
色およびその他の強調表示属性.	352
NetView コマンド機能画面の利点.	353
自動タスクによる複数コンソール・サポート・コンソールの使用.	353
第 21 章 コマンドの統合.	355
簡単なコマンド・プロシーチャーの作成.	355
将来の自動化への考慮.	356

コマンド・プロシージャーの修正	357
コマンド・プロシージャーの文書化	357

第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化 361

決定、自動化するメッセージおよび MSU の	361
メッセージを自動化する自動化テーブル・ステートメントの作成	362
メッセージ ID による検査	362
自動化アクション・メッセージ	363
他の特定基準の検査	363
包括的な基準の検査	365
解析テンプレートを使用するテキストの比較	366
MSU を自動化する自動化テーブル・ステートメントの作成	367
フィールドの存在の検査	369
フィールド内容の検査	370
RECMS と RECFMS の検査	371
MSU アクション	373
16 進表記、文字表記、およびビット表記	374
フィールドが 2 度以上出現する場合	375
ヘッダー情報の使用	376
アラート以外の主ベクトルの使用	376
リソース階層の使用	377
ドメイン ID の使用	378
メッセージ生成による他のデータの自動化	379
ハードウェア・モニター・レコードの自動化	379
状況変更の自動化	380
自動化ステートメントの有効化	380
相関エンジンを使用したメッセージおよび MSU の関連付け	381
相関の概要	381
ストレージに関する考慮事項	383
相関処理	383
状態相関によるフィルター操作	387

第 23 章 整合性のとれた自動操作の確立 405

状態変数を使う方法	405
初期化、モニター、リカバリー、およびシャットダウンの自動化	407
初期化の自動化	408
モニターの自動化	408
リカバリーの自動化	409
シャットダウンの自動化	409

第 24 章 オペレーター・インターフェースの拡張 411

メッセージの表示	411
状況情報の表示	412
状況モニターによる状況の追跡	412
NetView 管理コンソール・ディスプレイによる状況の追跡	412
ハードウェア・モニターによるアラートのモニター	413
プログラム間インターフェースによるアラートの送信	413
GENALERT コマンドによるアラートの送信	414
MS トランスポートによるアラートの送信	414
NetView 管理コンソールによるアラートのモニター	414
フルスクリーン・パネルの作成	415
電子メールまたは英数字ページの送信	416

第 6 部 複数システムの自動操作 417

第 25 章 他の NetView システムへの自動操作機能の拡張	419
ソースに近い自動操作	419
自動化プロシージャの区別	419
役割の定義	419
自動タスク定義の整合性	420
汎用自動化コマンド・プロシージャの開発	420
可搬自動化テーブルの開発	420
転送の指定	421
機能分散前のプロシージャのインストールおよびテスト	421
システム間自動化のログ記録	421
第 26 章 集中操作	423
データのトランスポート	423
LU 6.2 トランスポート	423
LUC	425
OST-NNT	425
NetView 設計済みフォーカル・ポイントのサポート	425
MS-CAPS アプリケーション	427
設計済みフォーカル・ポイントの制御範囲	430
設計済みフォーカル・ポイントの定義方法 (DEFFOCPT)	434
ALERT-NETOP アプリケーション	435
LINK-SERVICES-NETOP アプリケーション	442
OPS-MGMT-NETOP および EP-OPS-MGMT アプリケーション	443
ユーザー定義のカテゴリおよびユーザー定義のアプリケーション	443
NetView プログラムに固有のフォーカル・ポイント・サポート	444
LUC を使用したアラート転送	444
コマンドとメッセージの転送	445
OST-NNT を使用したメッセージ/アラート転送	448
フルスクリーン機能と端末アクセス機能	448
モニター時の SDOMAIN コマンドの使用	448
ドメイン・シフトのための TAF セッションの使用	448
分散システムへの直接のログオン	448
制限事項	449
転送方式の選択	449
構成の選択	450
専用回線と交換回線	450
持続セッションと非持続セッション	452
複数のフォーカル・ポイントの使用	452
フォーカル・ポイントの変更、除去、およびリスト	453
第 7 部 NetView におけるその他の自動操作	455
第 27 章 その他のシステム、装置、およびネットワークの自動化	457
イベント自動化サービス	457
アラートの転送	458
メッセージの転送	459
NCP フレーム・リレー交換装置のサポート	460
第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作	461
複数の RODM データ・キャッシュの管理	461
DSIQTSK タスクを使用する RODM の管理	462
DSIQTSKI 初期設定メンバーを使用する RODM の定義	462
ORCNTL コマンドを使用する RODM の管理	463
RODM メソッドからのコマンドの発行	464
RODM メソッドから出されたコマンドの検証	464

NetView プログラムから RODM へのアクセス	465
ORCONV コマンド	466
高水準言語およびアセンブリ言語プログラムから RODM へのアクセス	466
RODM 自動化シナリオ	466
シナリオのイベント	467
シナリオのエンティティ	467
シナリオのセットアップ	468
シナリオの実行	471
変更メソッド EKGCPPI のキー・セクション	476
第 29 章 端末アクセス機能を使用した自動操作	485
概説	485
TAF の動作方法	486
TAF のセットアップ	487
VTAMLST 定義の追加	487
CICS 端末定義の追加	488
IMS 端末定義の追加	489
TAF に使用される NetView コマンド	489
TAF を使用したアプリケーションの自動化	490
第 30 章 Common Base Event に関する自動化	493
Common Base Event の紹介	493
Common Base Event の作成	493
メッセージおよび MSU の自動化による Common Base Event の作成	493
ハードウェア・モニター・フィルターの設定による Common Base Event の作成	494
自動操作における Common Base Event の使用	495
Common Base Event の関連処理	495
第 31 章 自動化オペレーション・ネットワークの使用	499
AON の自動化とリカバリーについて	499
自動化テーブル	500
制御ファイル	500
自動化操作プログラムについて	500
通知について	501
自動化追跡について	501
ハードウェア・モニターでの自動化通知ログについて	502
リソース・リカバリーとしきい値	502
AON/SNA 自動化	504
AON/SNA オプションについて	505
AON/SNA サブエリア VTAM リソース自動化サポート	508
拡張対等通信ネットワーク機能のリソースのモニター	508
AON/SNA X.25 モニター・サポート	508
AON/TCP の自動化	509
能動モニター	510
リカバリー・モニター	510
MIB ポーリングおよびしきい値分け (z/OS TCP/IP の場合のみ)	510
オペレーターの注意	511
第 32 章 単一システムでの複数 NetView プログラムの実行	513
複数の NetView プログラムのインストール	514
NetView インターフェースと機能	515
プログラム・オペレーター・インターフェース (POI)	515
通信ネットワーク管理インターフェース (CNMI)	515
ハードウェア・モニターのローカル装置インターフェース	516
MVS サブシステム・インターフェース	517
GENALERT	518

状況モニターおよびログ・ブラウザ	519
インターフェースの使用	519
ネットワーク機能とシステム機能の分離	519
問題判別機能と自動化機能の分離	520
マイグレーション	520
2 つの NetView プログラム間の通信	520
NetView プログラムの自動化リカバリー	522
優先順位	522
第 33 章 自動操作のチューニング	523
ログ分析プログラム	523
リソース制御、タスクの優先順位、およびマルチタスキング	526
リソース制御	526
タスクの優先順位	527
複数の自動タスク	528
複数の NetView プログラム	528
自動化テーブル処理	529
ハードウェア・モニター・アラート	530
第 34 章 自動化テーブルのテスト	531
自動化テーブルのテスト	531
並行テストの開始	531
記録済み AIFR を使用しての、自動化テーブルのテスト	532
AUTOTEST コマンドの報告書例	534
テスト環境の使用	537
アプリケーションの使用	537
シミュレーターの使用	538
自動化機能の段階的实施	539
自動化テーブルの一致条件の検証	540
自動化アクション・パラメーターの検証	540
スケジュールされたコマンドの検証	541
自動化効果のチェック	541
自動タスクがコマンド・プロシーチャーを正しく処理することの確認	542
デバッグ・ツールの使用	543
ログの使用	543
自動化していないメッセージおよび MSU の評価	545
NetView 自動化テーブル・リストの使用	545
NetView 自動化テーブル・トレースの使用	545
第 35 章 ロギング	549
ログ記録に関する考慮事項	549
MVS システム・ログ (SYSLOG)	550
ネットワーク・ログ	551
ユーザー作成ログ	551
NetView のログ機能	552
MVS システム・ログと NetView ネットワーク・ログのレコード	553
第 36 章 ジョブ入力サブシステム 3 (JES3) の自動化	555
JES3 複合体でのメッセージ・フロー	555
グローバル・プロセッサーで生成されるメッセージ	555
ローカル・プロセッサーで生成されるメッセージ	556
JES3 環境におけるコマンド	558
NetView プログラムからの JES3 コマンドの発行	558
JES3 複合体の NetView からの MVS コマンドの発行	559
JES3 複合体におけるオペレーティング・システム・コンソールからの NetView コマンドの発行	559
JES3 環境における NetView プログラム	559

第 37 章 SNMP トラップの自動化	563
SNMP トラップの自動化タスク	563
SNMP トラップの自動化タスクの構成	563
SNMP トラップの自動化 CP-MSU	567
SNMP トラップの自動化の例	571
第 8 部 付録	575
付録 A. NetView プログラムの新しい自動操作機能へのマイグレーションの計画	577
NetView for z/OS V6R2 プログラム	577
NetView for z/OS V5R4 プログラム	578
NetView for OS/390 V1R4 プログラム	578
付録 B. サンプル・プロジェクト計画	579
プロジェクトの定義	580
設計	582
実施	583
実動	584
計画表	584
付録 C. 進捗状況測定 of サンプル	587
付録 D. MVS メッセージおよびコマンド処理	589
MVS におけるメッセージ・フロー	589
メッセージ処理機能	589
メッセージ処理におけるサブシステム	590
複数コンソール・サポート	591
コマンドの流れ	592
処理の判別	592
コンソールから出されるコマンド	593
MVS とのインターフェースとしての NetView	593
NetView プログラムで表示または処理される WTO として出されるメッセージ	594
NetView プログラムによって発行される MVS コマンド	594
MVS コンソールからサブシステム・コマンドとして出される NetView コマンド	594
MODIFY (F) コマンドを使用した MVS コンソールからの NetView コマンドの発行	595
VTAM インターフェースを使用したメッセージおよびコマンド	595
コンソールの操作	597
コマンドおよびコマンド・リストをサブシステム・コマンドとして発行するための MVS オペレーター・コンソールの使用	597
コマンドおよびコマンド・リストを MODIFY (F) コマンドとして発行するための MVS オペレーター・コンソールの使用	598
複数コンソール・サポート・オペレーターによるコマンド・リストの使用	598
NetView オペレーター ID からの MVS コマンドの発行	599
付録 E. VTAM メッセージおよびコマンド処理	601
VTAM のメッセージおよびコマンドの流れ	601
メッセージあふれ防止テーブル	602
VTAM メッセージの抑止基準	602
自動化テーブルによるイベントの認識	602
抑止レベルについて	603
抑止できないメッセージの識別	603
付録 F. NetView メッセージおよびコマンドの詳細な流れ	605
フローチャート	605
流れの説明	613

1. NetView コマンド入力 (VTAM 端末)	613
2. ドメイン間コマンド (OST から NNT へ)	614
3. VTAM (POI) コマンド入力	614
4. 送信請求システム・メッセージ	615
5. NetView コマンド入力 (MVS システム・コンソール)	615
6. NetView WTOR に対する応答	615
7. 非送信請求 VTAM (POI) メッセージ	616
8. 非送信請求 MVS システム・メッセージ	617
9. ドメイン間メッセージおよびコマンド (NNT から OST へ)	618
10. MVS または TAF OPCTL オペレーターとしての PPT	619
11. MVS または TAF OPCTL オペレーターとしての OST または NNT.	619
12. 送信請求 VTAM (POI) メッセージ	619
13. PPT メッセージ・キュー処理	620
14. PPT または NetView 許可レシーバー・メッセージの DSIPSS	620
15. OST または NNT メッセージ・キューの処理	622
16. NetView コンソール出力または SYSOP メッセージ・キューの処理	623
17. OST または NNT DSIPSS	624
18. OST、NNT、または自動タスクの送信請求および非送信請求システム MVS 拡張コンソール・メッセージ	625
19. PPT の送信請求および非送信請求システム MVS 拡張コンソール・メッセージ	625
付録 G. NetView メッセージ・タイプ (HDRMTYPE) の説明	627
付録 H. MVS コマンド管理 (非推奨)	633
NetView 環境での MVS コマンド管理の使用可能化	634
MVS 上の MVS コマンド出口の使用可能化	635
追加のコマンド・エコーおよび IEE295I メッセージの抑止	636
排他または包含リスト	638
論理 PARMLIB メンバー - CNMCAUaa	638
CNMCAUaa ステートメントの構文	638
コンソール排他リストおよびコンソール包含リスト	639
コマンド排他リストおよびコマンド包含リスト	639
CMDTEXT 排他リストおよび CMDTEXT 包含リスト	641
突き合わせの順序	642
MVS コマンド管理の開始	642
MVS コマンド出口の活動化	643
MVS コマンド処理の開始	643
MVS コマンド管理設定の表示	643
MVS コマンド管理の停止	643
MVS コマンド管理の停止および CNMCAUaa メンバーの保持	643
MVS コマンド管理の停止および CNMCAUaa メンバーの保持	643
MVS コマンド出口の呼び出しの停止	644
MVS コマンド出口の非活動化	644
MVS コマンド管理のテスト	645
排他または包含リストの開始	645
排他または包含リストの変更	645
コンソールとコマンドの包含および排他リストの一般的な処理	645
NetView コマンド出口によって除外されるコマンド	646
制限事項	647
NetView プログラムでの MVS コマンド管理の処理	648
MVS コマンド管理処理の保護	650
付録 I. 自動操作のサンプル・セット	651
自動操作サンプル・セットの使用	651
自動操作サンプル・セットの位置および名前変更	652
メッセージ抑止サンプル・セットの使用	653
ログ分析プログラムの使用	653

NetView プログラムと MVS 間の通信のセットアップ	653
基本自動操作サンプル・セットの使用	654
拡張自動操作サンプル・セットの使用	661
拡張自動操作サンプル・セットの使用準備	680
拡張自動操作サンプル・セットのカスタマイズ	685
コマンド・リストおよびサンプルの相互参照リスト	691
基本自動操作サンプル・セット	691
拡張自動操作サンプル・セット	692
メッセージ抑止サンプル	695
ログ分析サンプル	695
セットアップ・サンプル	695
特記事項	697
プログラミング・インターフェース	698
商標	698
プライバシー・ポリシーに関する考慮事項	699
索引	701



1. NetView プログラムが単一システム上にあるときの自動化の追加	8
2. 追加システムへの自動操作の伝搬	17
3. ローカルの自動操作が処理できない例外情報の転送	18
4. ターゲット・システムのリモート初期設定	20
5. サブシステム・インターフェースを使用した z/OS と NetView プログラム間のメッセージの流れ	35
6. z/OS システムと NetView プログラム間のコマンドの流れ	36
7. 複数サイト構成の例	65
8. 自動化で使用する NetView インターフェース	92
9. ASSIGN コマンドを使用した非送信請求メッセージの経路指定	98
10. ASSIGN コマンドを使用した非送信請求メッセージの除去	99
11. ASSIGN コマンドを使用した送信請求メッセージの経路指定	100
12. 汎用メッセージおよび特定メッセージの経路指定	101
13. 自動化テーブルの MSGID ステートメント	107
14. ハードウェア・モニターのデータの流れと自動化の MSU の流れ	110
15. EXCMD コマンドの例	114
16. AFTER コマンドの例	128
17. AT コマンドの例	128
18. EVERY コマンドの例	128
19. CHRON コマンドの例	129
20. スキップされた TIMER コマンドを示すメッセージ	130
21. LIST TIMER コマンドの例	131
22. PURGE TIMER コマンドの例	131
23. AUTO1 の定義ステートメント	133
24. 論理 AND 演算子の使用例	172
25. 追加の論理 AND 演算子の使用例	172
26. 論理 OR 演算子の使用例	173
27. 論理 OR および論理 AND 演算子の使用例	173
28. 論理演算子のグループ化の例	173
29. LABEL および ENDLABEL キーワードの使用例	174
30. GROUP キーワードの使用例	174
31. アクションを指定しない THEN キーワードの使用例	175
32. THRESHOLD キーワードに評価されるステートメント	231
33. ビットの比較の例	234
34. 長さの等しくないビットの比較の例	234
35. ヌルのビット・ストリングの比較の例	235
36. 複数行を使用するリテラル比較項目の例	235
37. 文字リテラルの比較の例	236
38. 文字リテラルにおける単一引用符の使用例	236
39. システム・シンボルを文字リテラルとして使用した例	236
40. 文字変数名の使用例	237
41. 文字変数名 DOMID の使用例	237
42. 16 進変数名の使用例	237
43. 変数 DOM1 の値を使用する例	238
44. プレースホルダーの使用例	238
45. プレースホルダーを使って単一文字を選択する例	238
46. ヌルの値を変数として使用する例	240
47. EXEC で CMD を指定する例	248
48. CMD キーワードと ROUTE キーワードの使用例	249
49. EXEC アクションで ROUTE キーワードを指定する例	251
50. 解析トークンの使用例	251

51. 解析区切り文字が無視される例	251
52. 組み合わせの対応しない解析トークンの例	252
53. 1 つのメッセージまたは MSU に対して複数の EXEC を実行する例	259
54. 複数のアクションを指定する例	259
55. CONTINUE を使用したメッセージの矛盾するアクションの例	260
56. SYN ステートメントの使用例	266
57. 同義語の置き換えが適用されない例	266
58. 同義語の置き換えが適用される例	266
59. ステートメントのグループ化の例	268
60. オカレンス検出条件項目の例	269
61. 複雑な比較項目を分離する例	269
62. 他の自動化テーブルを組み込む例	270
63. CONTINUE キーワードの使用例	272
64. ALWAYS ステートメントで CONTINUE キーワードを使用する例	272
65. 自動化テーブル同義語ステートメントの例	273
66. 自動化テーブルの主要メンバーの例	274
67. 自動化テーブル・リストの例	275
68. 自動化テーブル・メンバー	278
69. サンプル・メンバーの自動化テーブル・リスト	279
70. MSG 詳細報告書	283
71. MSU 詳細報告書	283
72. 使用統計で評価されるステートメント	283
73. メッセージ自動化の MSG 要約報告書	285
74. MSU 自動化の MSU 要約報告書	285
75. 自動化テーブル構造	288
76. 自動化テーブル管理コマンドのポップアップ	290
77. 自動化テーブル管理の挿入オプション	291
78. 自動化テーブル管理の表示オプションのポップアップ・ウィンドウ	293
79. 自動化テーブル管理グローバル・コマンドのポップアップ	294
80. 自動化テーブル管理のグローバル表示オプションのポップアップ	295
81. 自動化テーブル管理の ENABLE/DISABLE パネル	296
82. 自動化テーブル管理の Label/Block/Group パネル	298
83. z/OS システムと NetView サブシステム・インターフェース間のメッセージの流れ	334
84. z/OS と NetView プログラム間のコマンドの流れ	335
85. フィルター階層	344
86. Alerts-Dynamic パネルで受信されたアラート	345
87. NetView プログラムによりアラート用に生成されたメッセージ	345
88. コマンドによる NCP の活動化	356
89. NCP を活動化するためのコマンド・リスト・サンプル	356
90. サンプル・コマンド・プロシーチャー	359
91. メッセージ ID でメッセージを検査する例	362
92. メッセージ ID を使用して MVS WTOR メッセージを検査する例	362
93. ドメイン ID によってメッセージを検査する例	363
94. トークンを使用してメッセージのログを記録する例	364
95. テキスト位置を使用してメッセージのログを記録する例	364
96. プレースホルダーを使用してメッセージのログを記録する例	365
97. 論理 AND の論理を使用してメッセージを経路指定する例	365
98. 論理 OR 論理を使用してメッセージを経路指定する例	365
99. プレースホルダーを使用するメッセージの経路指定の例	366
100. 解析テンプレートでプレースホルダーを使用する例	366
101. 解析テンプレートで変数を使用する例	367
102. CP-MSU の概念図	368
103. NMVT の概念図	368
104. ハードウェア・モニターにおけるデータ・レコードの 16 進表示	369
105. MSU の選択の例	370
106. サブベクトルの選択の例	370

107.	サブフィールドの選択の例	370
108.	MSU サブベクトルの内容を検査する例	371
109.	MSU サブベクトルのある位置の内容を検査する例	371
110.	プレースホルダーを使用して MSU サブベクトルのある位置の内容を検査する例	371
111.	RECMS 82	371
112.	X'1044' にカプセル化された RECMS	372
113.	16 進表記の使用例	374
114.	文字表記の使用例	374
115.	ビット表記の使用例	375
116.	フィールドの複数オカレンスを検査する例	375
117.	フィールドの全オカレンスを検査する例	375
118.	MSU フィールドを詳細に検査する例	376
119.	MDS ヘッダーを検査する例	376
120.	MDS-MU でアラート主ベクトルを検査する例	376
121.	レゾリュション主ベクトルを自動化する例	377
122.	経路指定目標設定命令 GDS を自動化する例	377
123.	リソース階層のリソースを検査する例	378
124.	リソース階層の複数リソースを検査する例	378
125.	リソース階層のすべてのリソースを検査する例	378
126.	DOMAINID キーワードの使用例	378
127.	CNM094I メッセージの形式	380
128.	自動化テーブルを検証する例	380
129.	重複ルールの場合の状態遷移	393
130.	基本的なしきい値ルールの場合の状態遷移	394
131.	forwardEvents を使用するしきい値ルールの場合の状態遷移	395
132.	コレクター・ルールの場合の状態遷移	396
133.	パススルー・ルール (randomOrder=no) の場合の状態遷移	397
134.	パススルー・ルール (randomOrder=yes) の場合の状態遷移	398
135.	リセット・オン・マッチ・ルール (randomOrder=no) の場合の状態遷移	399
136.	リセット・オン・マッチ・ルール (randomOrder=yes) の場合の状態遷移	400
137.	アクションの構造	403
138.	状態変数を使って行われる、整合性のとれた自動操作	407
139.	REGISTER QUERY コマンドからの出力例	424
140.	VTAM APPL ステートメント	425
141.	DSI6INIT における典型的なフォーカル・ポイントとエントリー・ポイントの定義ステートメント	434
142.	LU 6.2 を使用した NetView 中間ノード・フォーカル・ポイントのアラート転送	438
143.	RMTCMD の例	446
144.	交換回線サポート	451
145.	自動化テーブルでの SAVECMD コマンド・リストの使用	465
146.	DSIQTSK タスクの DSIQTSKI 初期設定メンバーのサンプル	469
147.	IST105I をトラップし、ORCONV コマンドを出す自動化テーブル・ステートメント	470
148.	DWO670I をトラップする自動化テーブル・ステートメントのサンプル	471
149.	RODM ローダーへの入力ファイル	471
150.	デフォルト RODM の変更	472
151.	自動化テーブルの活動化	472
152.	DEFAULT SENDMSG パラメーターの設定	473
153.	リソース A01A704 を非活動化する例	473
154.	ASSISCMD コマンドの画面例	474
155.	ASSISCMD コマンドの画面例 -- 詳細を表示する M の入力	475
156.	ASSISCMD コマンドの画面例 -- コマンドの詳細	475
157.	ASSISCMD コマンドの画面例 -- コマンドを実行する E の入力	476
158.	変更メソッド EKGCPPI のプロシージャ・ステートメント	476
159.	変更メソッド EKGCPPI のローカル変数	477
160.	変更メソッド EKGCPPI の定数	479
161.	変更メソッド EKGCPPI の初期設定	480
162.	変更メソッド EKGCPPI のサブフィールドの変更	481

163. 変更メソッド EKGCPPI のフィールドの照会	482
164. 変更メソッド EKGCPPI のオブジェクト名の照会	483
165. 変更メソッド EKGCPPI のオブジェクト独立メソッドの起動	484
166. TAF ソース LU のための VTAMLST 定義サンプル	487
167. CICS への TAF の定義	488
168. IMS への TAF の定義	489
169. 自動化障害ロジック	503
170. AON/SNA により自動化されるリソース	505
171. ログ分析プログラムの出力	524
172. フィルター操作されるメッセージ	525
173. フィルター操作を行ったログ分析プログラムの出力	525
174. 自動化テーブルのコマンド処理の回避	529
175. コメント化されているアクションが指定された自動化ステートメント	540
176. \$HASP098 コマンド・リスト	542
177. &CONTROL CMD ステートメントによって書き込まれる DSI013I メッセージ	544
178. メッセージを LOGSEQ に渡すステートメント	545
179. メッセージ CNM493I の形式	549
180. AUTH=CNM アプリケーションを開始させるステートメント	596
181. 他の NetView アプリケーション・プログラムを開始させるステートメント	596
182. DASD をオンラインにするために使用するコマンド	599
183. NetView コマンド入力のフローチャート (VTAM 端末)	605
184. ドメイン間コマンドのフローチャート	606
185. VTAM (POI) コマンド入力のフローチャート	607
186. 送信請求システム (サブシステム・インターフェース) メッセージのフローチャート	608
187. NetView コマンド入力のフローチャート (MVS)	608
188. NetView WTOR に対する応答のフローチャート	609
189. 非送信請求 VTAM (POI) メッセージのフローチャート	609
190. 非送信請求システム (SSI または MVS 拡張コンソール) メッセージのフローチャート (CNMCSSIR)	610
191. ドメイン間メッセージ (NNT から OST へ) のフローチャート	610
192. メッセージのフローチャート (オペレーターは PPT)	611
193. メッセージのフローチャート (オペレーターは OST/NNT)	611
194. OST、NNT、または自動タスク用の送信請求および非送信請求システム MVS 拡張コンソール・メッセージ のフローチャート	612
195. PPT 用の送信請求および非送信請求システム MVS 拡張コンソール・メッセージのフローチャート	613
196. 自動タスクをシステム・コンソールと関連付けるコマンド	615
197. MVS コマンド管理の流れ	634
198. 基本自動操作サンプル・セット自動化テーブル項目	655
199. 基本自動操作サンプル・セットの自動化テーブルにより自動化されるメッセージ	655
200. ROUTE コマンドでの複数の自動タスクとオペレーターの指定	656
201. 自動化テーブルのテスト	659
202. 自動化テーブルの活動化	660
203. 自動タスク AUTO1 の活動化	660
204. CICS 異常終了メッセージ	662
205. 拡張自動操作サンプル・セットにおける受動モニター	663
206. AUTOJES 自動タスクの能動モニター	664
207. メッセージ DSI039I の能動モニター	665
208. DSI039I メッセージに回答する自動化テーブルの EXCMD コマンド	665
209. CNMS64P0 表示の例	678
210. CNMS64P1 表示の例	679
211. CNMS6408 の抜粋 (AUTOMGR オペレーター定義)	683
212. CNMS6409 の抜粋 (DSIPROFM オペレーター・プロファイル)	684
213. TSO 開始変数の変数の定義	687
214. TSO 開始変数の作成	687
215. 共通グローバル変数として &START を定義するステートメント	688
216. 共通グローバル変数の間接的な更新	688
217. 割り当てでの共通グローバル変数の置き換え	688

本書について

IBM® Tivoli® NetView® for z/OS® 製品の高度な機能により、マルチプラットフォームおよびマルチベンダーの複合ネットワークとシステムを一元的に管理して、高レベルの可用性を維持することができます。本書「*IBM Tivoli NetView for z/OS 自動操作ガイド*」には、自動操作の計画立案についての情報が記載されています。NetView プログラムの自動化機能は、システムとネットワークの効率、およびオペレーターの生産性を向上させるために役立てることができます。NetView 自動化機能によって、オペレーターが実行する日常処理の大部分を除去し、単純化することができます。

対象読者

本書は、システム自動化、ネットワーク自動化、またはその両方を行うために NetView プログラムを使用するシステム・プログラマーを対象としています。本書は、自動化についての初心者と、既存の自動化プロジェクトを更新または拡張する方の両方を対象としています。

資料

このセクションでは、IBM Tivoli NetView for z/OS ライブラリーに収められている資料、およびその他の関連資料を取り上げます。また、Tivoli オンライン資料へのアクセス方法と、Tivoli の資料の注文方法についても説明します。

IBM Tivoli NetView for z/OS ライブラリー

IBM Tivoli NetView for z/OS ライブラリーでは、以下の資料が入手可能です。

- 「アドミニストレーション・リファレンス」(SA88-4383) では、システム管理に必要な NetView プログラム定義ステートメントについて記述しています。
- 「アプリケーション・プログラマーズ・ガイド」(SA88-4384) では、NetView プログラム間インターフェース (PPI)、および NetView アプリケーション・プログラミング・インターフェース (API) を使用する方法について記述しています。
- 「自動操作ガイド」(SA88-4387) では、自動化された操作を使用してシステムとネットワークの効率性およびオペレーターの生産性を改善する方法を説明しています。
- 「*Command Reference Volume 1 (A-N)*」(SC27-2847) および「*Command Reference Volume 2 (O-Z)*」(SC27-2848) では、ネットワークとシステム操作およびコマンド・リストとコマンド・プロシージャーで 사용할 ことができる NetView コマンドについて記述しています。
- 「カスタマイズ・ガイド」(SA88-4388) では、NetView 製品をカスタマイズする方法が記述されており、関連情報のソースを参照できるようになっています。
- 「*Data Model Reference*」(SC27-2850) では、Graphic Monitor Facility host subsystem (GMFHS)、SNA トポロジー・マネージャー、およびマルチシステム・マネージャーのデータ・モデルについて説明しています。

- 「インストール:追加コンポーネントの構成」(GA88-4389) では、NetView の基本機能だけでなく追加機能を構成する方法について記述しています。
- 「インストール:グラフィカル・コンポーネントの構成」(GA88-4390) では、NetView グラフィックス・コンポーネントをインストールおよび構成する方法について記述しています。
- 「*Installation: Configuring the GDPS Active/Active Continuous Availability Solution*」(SC14-7477) では、GDPSアクティブ/アクティブ継続的可用性ソリューションとともに使用される NetView 機能を構成する方法について記述しています。
- 「インストール: *NetView Enterprise Management Agent*の構成」(GA88-4401) では、NetView for z/OS Enterprise Management Agent をインストールおよび構成する方法について記述しています。
- 「インストール:概説」(GI88-4261) では、基本 NetView プログラムをインストールおよび構成する方法について記述しています。
- 「インストール:マイグレーション・ガイド」(GA88-4391) では、NetView 製品の現行リリースによって提供される新規機能および前のリリースからの基本機能のマイグレーションについて記述しています。
- 「IP管理」(SA88-4386) NetView プロダクトを使用して IP ネットワークを管理する方法について説明しています。
- 「*Messages and Codes Volume 1 (AAU-DSI)*」(GC27-2856) および「*Messages and Codes Volume 2 (DUI-IHS)*」(GC27-2857) では、NetView 製品のメッセージ、NetView異常終了コード、NetView メッセージに含まれるセンス・コード、および総称アラート・コード・ポイントについて記述しています。
- 「プログラミング:アセンブラー」(SA88-4392) では、アセンブラー言語を使用して NetView 製品の出口ルーチン、コマンド・プロセッサ、およびサブタスクを作成する方法について記述しています。
- 「プログラミング:パイプ」(SA88-4393) では、NetView パイプラインを使用して NetView インストール済み環境をカスタマイズする方法について記述しています。
- 「プログラミング: PL/IおよびC」(SA88-4394) では、PL/I または C を使用して NetView 製品のコマンド・プロセッサおよびインストール・システム出口ルーチンを作成する方法について記述しています。
- 「プログラミング: REXXおよびNetViewコマンド・リスト言語」(SA88-4395) では、再構造化拡張実行プログラム言語 (REXX) または NetView コマンド・リスト言語を使用して NetView 製品のコマンド・リストを作成する方法について記述しています。
- 「*Resource Object Data Manager and GMFHS Programmer's Guide*」(SC27-2862) では、NetView リソース・オブジェクト・データ・マネージャー (RODM) (非 SNA ネットワークを RODM へ定義する方法やネットワーク自動化とアプリケーション・プログラミングで RODM を使用する方法を含む) について記述しています。
- 「セキュリティ・リファレンス」(SA88-4397) では、NetView 環境の許可検査を実装する方法について記述しています。
- 「SNAトポロジー・マネージャー インプリメンテーション・ガイド」(SA88-4398) では、サブエリアを管理するのに使用する NetView SNA トポロジ

ー・マネージャー、拡張対等通信ネットワーキング (APPN)、および TN3270 リソースの計画および実装について記述しています。

- 「*Troubleshooting Guide*」(GC27-2865) では、NetView 製品で発生する問題の文書化、診断、および解決についての情報を提供しています。
- 「*チューニング・ガイド*」(SA88-4399) では、NetView 製品およびネットワーク環境での一定のパフォーマンス目標を達成するために役立つチューニング情報を提供しています。
- 「*Automated Operations Network* ユーザーズ・ガイド」(SA88-4385) では、イベント・ドリブン・ネットワーク自動化機能を提供してシステムとネットワークの効率を向上させる NetView Automated Operations Network (AON) コンポーネントを使用する方法について説明しています。また、AON コンポーネントの自動操作機能を調整および拡張する方法についても説明しています。
- 「*ユーザーズ・ガイド: NetView*」(SA88-4400) では、NetView 製品を使用して複雑なマルチベンダーのネットワークとシステムを一元的に管理する方法について説明しています。
- 「*NetView Enterprise Management Agent* ユーザーズ・ガイド」(SA88-4402) では、NetView Enterprise Management Agent を使用する方法について説明しています。
- 「*NetView 管理コンソール ユーザーズ・ガイド*」(SA88-4396) では、NetView 製品の NetView 管理コンソール・インターフェースについて情報を提供しています。
- 「*Licensed Program Specifications*」(GC31-8848) では、NetView 製品のライセンス情報を提供しています。
- 「*Program Directory for IBM Tivoli NetView for z/OS US English*」(GI11-9444) には、IBM Tivoli NetView for z/OS 製品のインストールに関する資料と手順についての情報を記載しています。
- 「*Program Directory for IBM Tivoli NetView for z/OS Japanese*」(GI11-9445) には、IBM Tivoli NetView for z/OS 製品のインストールに関する資料と手順についての情報を記載しています。
- 「*Program Directory for IBM Tivoli NetView for z/OS Enterprise Management Agent*」(GI11-9446) には、IBM Tivoli NetView for z/OS Enterprise Management Agent のインストールに関する資料と手順についての情報を記載しています。
- 「*IBM Tivoli NetView for z/OS V6R2 Online Library*」(LCD7-4913) には、NetView for z/OS ライブラリーにある資料が含まれています。資料は、PDF フォーマットおよび HTML フォーマットで入手可能です。

関連資料

追加の製品情報は、NetView for z/OS Web サイト (<http://www.ibm.com/software/tivoli/products/netview-zos/>) 上で検索できます。

NetView ブリッジ機能については、「*Tivoli NetView for OS/390 Bridge Implementation*」(SC31-8238-03、V1R4 ライブラリーからのみ入手可能) を参照してください。

オンライン用語集へのアクセス

IBM Terminology Web サイトには、多数の IBM プロダクト・ライブラリーからの用語が 1 つの便利なロケーションに統合されています。 Terminology Web サイトには <http://www.ibm.com/software/globalization/terminology/> でアクセスできます。

NetView for z/OS の用語と定義については、IBM Terminology Web サイトを参照してください。以下の用語は、このライブラリーで使用されます。

NetView

以下のプロダクト:

- Tivoli NetView for z/OS バージョン 6 リリース 2
- Tivoli NetView for z/OS バージョン 6 リリース 1
- Tivoli NetView for z/OS バージョン 5 リリース 4
- Tivoli NetView for z/OS バージョン 5 リリース 3
- Tivoli NetView for OS/390® バージョン 1 リリース 4
- サポートされなくなった NetView リリース

CNMCMD

CNMCMD メンバー、および %INCLUDE ステートメントを使用してその中に組み込まれるメンバーに関する用語

CNMSTYLE

CNMSTYLE メンバー、および %INCLUDE ステートメントを使用してその中に組み込まれるメンバーに関する用語

DSIOPF

DSIOPF メンバー、および %INCLUDE ステートメントを使用してその中に組み込まれるメンバーに関する用語

PARMLIB

連結シーケンスでの SYS1.PARMLIB およびその他のデータ・セットに関する用語

MVS™ z/OS オペレーティング・システムに関する用語

MVS エlement

z/OS オペレーティング・システムの基本制御プログラム (BCP) Element に関する用語

VTAM®

Communications Server - SNA Services に関する用語

IBM Tivoli Network Manager

以下のいずれかのプロダクトに関する用語

- IBM Tivoli Network Manager
- IBM Tivoli OMNIbus and Network Manager

IBM Tivoli Netcool/OMNIbus

以下のいずれかのプロダクトに関する用語

- IBM Tivoli Netcool/OMNIbus
- IBM Tivoli OMNIbus and Network Manager

特に断りのない限り、トピックでプログラムに言及する場合は、そのプログラムの最新のバージョンとリリースを指します。トピックでバージョンのみが示されている場合は、そのバージョンのすべてのリリースを指します。

トピックでパーソナル・コンピュータまたはワークステーションの使用に言及する場合は、プログラマブル・ワークステーションであればいずれも使用できます。

NetView for z/OS オンライン・ヘルプの使用

インストール済み環境と構成に応じて、以下の種類の NetView for z/OS メインフレーム・オンライン・ヘルプが用意されています。

- 一般ヘルプおよびコンポーネント情報
- コマンド・ヘルプ
- メッセージ・ヘルプ
- センス・コード情報
- 推奨処置

マニュアルへのオンライン・アクセス

以下は英語のみの対応となります。資料 DVD「*IBM Tivoli NetView for z/OS V6R2 Online Library*」には、製品ライブラリーにある資料が含まれています。資料は、PDF フォーマットおよび HTML フォーマットで入手可能です。資料へのアクセス方法の説明については、DVD 上の README ファイルを参照してください。

IBM では、この製品の他すべての Tivoli 製品に関する資料が使用可能になった時点および更新された時点で、Tivoli Documentation Central の Web サイト (<https://www.ibm.com/developerworks/mydeveloperworks/wikis/home/wiki/Tivoli%20Documentation%20Central>) に掲載しています。

注: レターサイズ以外の用紙に PDF 文書を印刷する場合は、「ファイル」>「印刷」ウィンドウで、Adobe Reader でレターサイズのページをローカル用紙に印刷できるようにするオプションを設定してください。

マニュアルのご注文

日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは <http://www.ibm.com/jp/manuals/> の「マニュアル・出版物情報」をご覧ください。(URL は、変更になる場合があります)

アクセシビリティ

アクセシビリティ機能は、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア製品を快適に使用できるようサポートします。製品では、標準的なショートカット・キーおよびアクセラレーター・キーが使用され、オペレーティング・システムによって文書化されます。詳しくは、ご使用のオペレーティング・システムが提供する資料を参照してください。

詳しくは、「ユーザーズ・ガイド: NetView」の付録『アクセシビリティ』を参照してください。

Service Management Connect

サービス・マネジメント専門家と情報交換、学習、および共有を行います。これらの専門家は製品サポート技術のエキスパートであり、さまざまな見通しや専門知識を提供します。

Service Management Connect (<http://www.ibm.com/developerworks/servicemanagement/z/>) にアクセスします。Service Management Connect は以下の方法で利用できます。

- Tivoli 製品の他のユーザーと IBM 開発者の間の公開された進行中の取り組みである透過的開発に参加する。初期設計、スプリント・デモ、製品ロードマップ、プレリリース・コードにアクセスすることができます。
- 専門家と 1 対 1 でつながり、Tivoli および NetView コミュニティーに関して共同作業およびネットワーキングを行う。
- ブログを読んで、他の人の専門知識や経験を参考にする。
- WiKi やフォーラムを使用して、より広範囲にわたるユーザー・コミュニティと共同作業を行う。

Tivoli 技術研修

以下は英語のみの対応となります。Tivoli 技術研修の情報については、以下の IBM Tivoli Education Web サイト (<http://www.ibm.com/software/tivoli/education>) を参照してください。

Tivoli ユーザー・グループ

Tivoli ユーザー・グループは、独立した、ユーザーにより運営されたメンバーシップ組織であり、Tivoli ユーザーに対して、Tivoli Software ソリューションをインプリメントする際にユーザーを支援する情報を提供します。このユーザー・グループを介して、メンバーは情報を共有することができ、また、他の Tivoli ユーザーの知識や経験を習得することができます。

ダウンロード

クライアントとエージェント、NetView 製品のデモンストレーション、およびいくつかの無償の NetView アプリケーションは、以下の NetView for z/OS サポート Web サイトからダウンロードできます。

<http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliNetViewforzOS.html>

「サポート・ショートカット」ペインで、「**Tivoli NetView for z/OS**」を展開し、「**Fixes (downloads)**」をクリックして、ダウンロードを検索または選択できるページに移動します。

これらのアプリケーションは、以下の作業に役立ちます。

- カスタマイズ・パラメーターと初期化ステートメントを前のリリースから CNMSTUSR メンバーに、およびコマンド定義を前のリリースから CNMCMDU メンバーにマイグレーションする
- 自動化テーブルの統計情報の入手、および自動化テーブルのリストとの統計情報のマージ
- JES (Job Entry Subsystem) ジョブの状況の表示、または指定された JES ジョブの取り消し

- プログラム間インターフェース (PPI) を使用した、NetView プログラムへのアラートの送信
- PPI を使用した、MVS コマンドの送信および受信
- TSO (Time Sharing Option) コマンドの送信および応答の受信

サポート情報

以下は英語のみの対応となります。IBM ソフトウェアに問題が発生した場合、迅速に解決する必要があります。お客様が必要なサポートを得られるように、IBM は以下の方法を提供しています。

オンライン

Tivoli Software Support サイト (<http://www.ibm.com/software/sysmgmt/products/support/index.html?ibmprd=tivman>) にアクセスします。IBM Software Support サイト (<http://www.ibm.com/software/support/probsub.html>) にアクセスします。

IBM Support Assistant

IBM Support Assistant は、IBM ソフトウェア製品に関する疑問および問題の解決に役立つ無償のローカル・ソフトウェア保守サービス・ワークベンチです。Support Assistant により、問題判別のためのサポート関連の情報および保守サービス・ツールに迅速にアクセスできます。Support Assistant ソフトウェアをインストールするには、<http://www.ibm.com/software/support/isa/> にアクセスします。

トラブルシューティング情報

NetView for z/OS 製品の問題解決について詳しくは、「*IBM Tivoli NetView for z/OS Troubleshooting Guide*」を参照してください。NetView for z/OS 製品の追加サポートは、Yahoo の NetView ユーザー・グループ (<http://groups.yahoo.com/group/NetView/>) で得られます。このサポートの対象は NetView for z/OS ユーザーに限定されており、登録する必要があります。このフォーラムは、質問に答え、ガイダンスを与える NetView 開発者がモニターしています。コードに関する問題が見つかったら、解決策を得るため正式な問題管理レコード (PMR) を開くよう求められます。

本書の表記規則

このセクションでは、本書で使われる規則について説明します。

書体の規則

本書では、書体について以下の規則を使用しています。

太字

- 周囲のテキストと見分けが付きにくい小文字のコマンドおよび大/小文字混合のコマンド
- インターフェース・コントロール (チェック・ボックス、プッシュボタン、ラジオ・ボタン、スピン・ボタン、フィールド、フォルダー、アイコン、リスト・ボックス、リスト・ボックス内の項目、複数列リスト、コン

テナー、メニューの選択項目、メニュー名、タブ、プロパティ・シート)、ラベル (ヒント: およびオペレーティング・システムの考慮事項: など)

- テキスト内のキーワードおよびパラメーター

イタリック

- 引用 (例: 資料、ディスクット、および CD のタイトル)
- テキスト内で定義されている語 (例: 非交換回線は *Point-to-Point* 回線 と呼ばれる)
- 語および文字の強調 (言葉として扱われる語の例: 『制限節を挿入するには、単語 *that* を使用します』。文字として扱われる場合の例: 『LUN アドレスは文字 *L* で始める必要があります』。)
- テキスト中の新規用語 (定義リスト内を除く): *view* は、データが入っているワークスペース内のフレームです。
- 指定する必要がある変数および値: ... ここで *myname* が表すものは ...

モノスペース

- 例およびコード例
- 周囲のテキストと見分けが付きにくいファイル名、プログラミングのキーワード、およびその他のエレメント
- ユーザー宛てのメッセージ・テキストおよびプロンプト
- ユーザーが入力する必要があるテキスト
- 引数またはコマンド・オプションの値

オペレーティング・システム依存の変数とパス

ワークステーション・コンポーネントの場合、本書では、環境変数およびディレクトリー表記に UNIX の規則を使用しています。

Windows コマンド行を使用する場合、環境変数では \$変数 を %変数% に置き換え、ディレクトリーのパスではスラッシュ (/) をそれぞれ円記号 (¥) に置き換えます。環境変数の名前は、Windows 環境と UNIX 環境とで常に同じとは限りません。例えば、Windows 環境の %TEMP% と UNIX 環境で同等なのは、\$TMPDIR です。

注: Windows システムで bash シェルを使用している場合は、UNIX の表記規則を使用できます。

構文図

構文図には、以下の構文エレメントが示されます。水平線 (メインパス) に従い、左から右、上から下に向かって構文図を見てください。

- xxix ページの『記号』
- xxix ページの『パラメーター』
- xxix ページの『句読点と括弧』
- xxx ページの『省略形』

構文の例については、xxx ページの『構文例』を参照してください。

記号

構文図では、以下の記号が使用されます。

- ▶▶ コマンド構文の開始を示します。
- ▶ コマンド構文が続くことを示します。
- | コマンド構文のフラグメントまたは一部の開始および終わりを示します。
- ◀◀ コマンド構文の終わりを示します。

パラメーター

構文図では、以下のタイプのパラメーターを使用しています。

必須 必須パラメーターはメインパス上に表示します。

オプション

オプション・パラメーターは、メインパスの下に示されます。

デフォルト値

デフォルト・パラメーターは、メインパスの上に示されます。パラメーターの説明では、デフォルト・パラメーターには下線が付いています。

構文図では、強調表示、大括弧、または中括弧を頼りにすることはできません。構文図において、主構文線に対するエレメントの相対位置は、エレメントが必須なのか、オプションなのか、またはデフォルト値なのかを示します。

コマンドを発行するときには、コンマなど別の区切り文字が構文内で指定されていない限り、パラメーター間にスペースが必要です。

パラメーターは、キーワードまたは変数に分類されます。キーワードは、大文字で示されます。ユーザーが指定する名前または値を表す変数は小文字で示され、斜体かまたは (NetViewヘルプでは) 異なる色で表示されます。

以下の例では、**USER** コマンドがキーワード、*user_id* パラメーターが必須の変数、そして *password* パラメーターがオプションの変数です。

▶▶—USER—*user_id*—password—▶▶

句読点と括弧

コロン、セミコロン、コンマ、負符号 (-)、および一重引用符と二重引用符など、構文図で示されているすべての句読点を含める必要があります。

オペランドに複数の値がある場合、一般にそれらの値は、括弧で囲んでコンマで区切ります。単一の値の場合は一般に、括弧を省略できます。詳しくは、xxxi ページの『複数のオペランドまたは値』を参照してください。

コマンドにキーワードと変数を区切る定位置コンマを入れる必要がある場合は、キーワードまたは変数の前にコンマを置きます。

コマンドの例を示す場合は、定位置オペランドが存在しないことを示すためにもコンマを使用します。例えば、2 番目のコンマはオプションのオペランドが使用されていないことを示します。

COMMAND_NAME *opt_variable_1*,,*opt_variable_3*

末尾の定位置コンマを指定する必要はありません。定位置、非定位置にかかわらず、末尾のコンマは無視されるか、コマンドがリジェクトされる原因となります。末尾のコンマによってコマンドがリジェクトされるかどうかについては、各コマンドの制約事項を参照してください。

省略形

コマンドおよびキーワードの省略形は、各コマンドの説明の後の同義語表を参照してください。

構文例

次の例では、構文エレメントのさまざまな使用法を示します。

- 『必須構文エレメント』
- 『オプションの構文エレメント』
- 『デフォルトのキーワードおよび値』
- xxxi ページの『複数のオペランドまたは値』
- xxxi ページの『1 行より長い構文』
- xxxi ページの『構文のフラグメント』

必須構文エレメント:

必須のキーワードおよび変数は、構文の主線上に示されます。必須のキーワードと変数をコーディングする必要があります。

▶—REQUIRED_KEYWORD—*required_variable*—▶

必要な選択項目 (2 つ以上の項目) は、メインパスの上側にある垂直スタックに表示されます。項目は英数字順に表示されています。

▶—REQUIRED_OPERAND_OR_VALUE_1—
—REQUIRED_OPERAND_OR_VALUE_2—▶

オプションの構文エレメント:

オプションのキーワードおよび変数は、構文の主線より下に示されます。オプションのキーワードと変数は、コーディングしないことを選択できます。

▶—OPTIONAL_OPERAND—▶

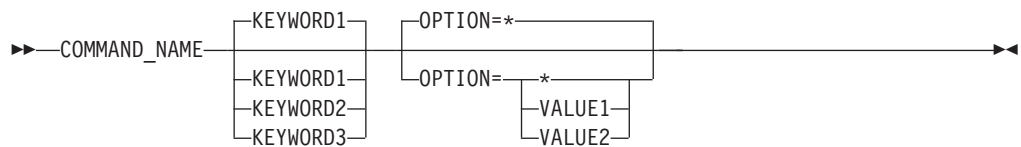
必要な選択項目 (2 つ以上の項目) は、メインパスの下側にある垂直スタックに表示されます。項目は英数字順に表示されています。

▶—OPTIONAL_OPERAND_OR_VALUE_1—
—OPTIONAL_OPERAND_OR_VALUE_2—▶

デフォルトのキーワードおよび値:

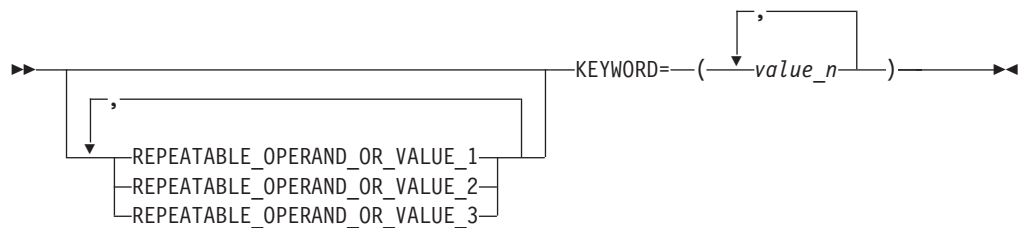
デフォルトのキーワードおよび値は、以下のいずれかの方法で、構文の主線より上に示されます。

- デフォルトのキーワードは、構文の主線より上にのみ示されます。このキーワードを指定することも、または、指定せずにデフォルトにすることもできます。以下の構文例では、構文の主線より上にデフォルト・キーワード **KEYWORD1** が、構文の主線より下に残りのオプションのキーワードが示されています。
- オペランドにデフォルト値がある場合、そのオペランドは構文の主線より上と下の両方に示されます。構文の主線より下に値がある場合は、オペランドを指定するときに、デフォルト値または表示されている値のいずれかを指定する必要があります。オペランドを指定しない場合は、構文の主線より上にあるデフォルト値が使用されます。以下の構文の例は、構文の主線の上下にオペランド **OPTION=*** のデフォルト値が示されています。



複数のオペランドまたは値:

一群のオペランドまたは値の上にある左に戻る矢印は、複数選択が可能か、または 1 つの値を繰り返すことができることを示しています。



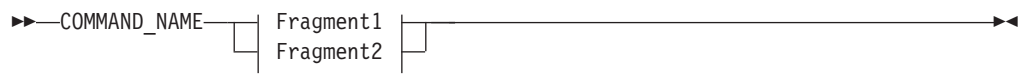
1 行より長い構文:

図が 1 行より長い場合は、続きのある各行が 1 つの矢印で終わり、次の行の先頭が 1 つの矢印で始まります。

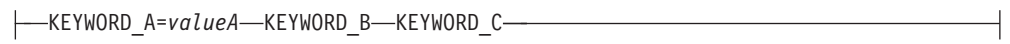


構文のフラグメント:

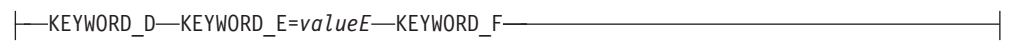
構文図によっては、構文の長い、複雑な、または繰り返されるセクションを表すために使用する構文のフラグメントが含まれています。構文のフラグメントは、メインの構文図の後に記述されます。それぞれの構文フラグメント名は、大/小文字が混合しており、メインの構文図およびフラグメントの見出しに示されます。以下の構文例は、**Fragment1** および **Fragment2** という 2 つのフラグメントを含む構文図を示しています。



Fragment1



Fragment2



第 1 部 自動操作の概要

第 1 章 NetView 自動操作の紹介

本章では次の点について説明しながら、NetView 自動操作について紹介していきます。

- NetView 自動操作 という用語の意味
- 自動操作の利点
- 自動操作のクラス
- 自動操作の段階

NetView 自動操作とは

NetView 自動操作 とは、NetView プログラム (およびいくつかの関連プロダクト) を使用して、通常なら人間の介入を要するような情報システムおよびネットワーク操作を自動化することを意味します。NetView プログラムは、システムおよびネットワークの自動操作を援助するための専門のサービスを提供します。NetView プログラムは、これらのサービスを通して、多くの日常的なオペレーター作業を実行することができます。

NetView プログラム、および NetView 自動操作に使用するその他のプロダクトの具体的な機能については、25 ページの『第 2 章 自動操作プロダクトの概要』を参照してください。

自動操作の利点

NetView 自動操作は、操作環境の簡素化を図ることによって、システムおよびネットワークの全域に対して利益をもたらします。オペレーティング・システムをはじめサブシステム、アプリケーション・プログラム、ネットワーク装置、そのほか多くのプロダクトを管理するにあたって、要求される手操作介入の数を減らすことができます。

操作の簡素化に対する必要性は、データ・センターへのハードウェアおよびソフトウェア・プロダクトの追加、ネットワークへのデータ・センターの追加、およびデータ処理スタッフ人員の追加にともなって増します。NetView 自動操作により操作が簡素化されれば、必要なサービス水準の達成、コスト削減、オペレーターの有効活用といった効果が望めます。

NetView 自動操作は次のことに役立ちます。

- システムおよびネットワークの可用性の改善
- 拡張の際の制約の除去
- オペレーターの生産性の向上
- より一貫した操作手順の確立

システムおよびネットワークの可用性の改善

自動操作はシステムおよびネットワークの可用性の向上をもたらします。操作を自動化すれば、予期しない事態に直面した場合でも、迅速かつ正確な対応が可能になります。障害が発生した場合、それが計画したものか計画外のものかにかかわらず、自動操作によって、リカバリー時間を短縮することができます。

自動化は、オペレーターの実ミスの可能性を減らします。オペレーターによるミスは、ときには障害やリカバリー時間の遅延をもたらします。例えば、オペレーターによるメッセージの見落としやコマンドの誤入力などが考えられます。また、長いコマンドのシーケンスを入力する際には、オペレーターは、いくつものプログラムまたはコンポーネントのコマンド構文を覚えなければなりません (あるいは、時間をかけてそれらの構文を探し出さなければなりません)。オペレーターがミスをする可能性はいたるところにあります。

自動操作では、オペレーターによるコマンド入力の代わりに自動応答が行われます。オペレーターの介入が要求される場合は、自動操作プロシージャがタスクを簡素化するので、ミス発生が減り、同様のイベントに対し確実に同様の対応を講じることができます。さらに、自動操作によってダウン時間が縮小され、シャットダウン、初期設定、およびリカバリーなどのプロシージャの迅速化が期待されます。

拡張の際の制約の除去

自動操作は、システム、およびネットワークの拡張にともなう制約を除去するのに役立ちます。例えば、絶えず増大するデータ転送速度などが、拡張の制約となる場合があります。

システムが迅速化し、ネットワークが拡大する環境になるにつれて、オペレーターが受け取るメッセージとアラートの数も増大します。通常の実作条件の下では、大部分のオペレーターにとって、1 つ 1 つのメッセージを読んで理解することは可能でも、それらすべてに対応するのは困難な場合があります。自動操作を行えば、次のような仕組みによって、表示されるメッセージやアラートの数を大幅に減らすことができます。

- ルーチン・メッセージの抑制
- ルーチン・アラートのブロック
- メッセージおよびアラートへの自動応答

拡張の妨げになっているものの 1 つに、コンソールの増加が挙げられます。新規のプロダクトを導入するたびに、管理しなければならないコンソールが増えることになります。オペレーティング・システムにかかわらず、システムの数が増えれば、その分だけ必要なコンソールの数も増えます。自動操作によって個々のシステム上のコンソールを統合すれば、集中化された 1 点から多数のシステムを操作できるようになります。

また、拡張を妨げるもう 1 つの要因として、ネットワークの複雑化を挙げることができます。相互接続ネットワークには、しばしば膨大な数のリソースとさまざまなタイプのプロダクトが含まれ、さらに TCP/IP リソースとシステム・ネットワーク体系 (SNA) リソースの組み合わせが存在します。これらすべてを管理できるような熟練オペレーターを見つけ出すのは困難であり、コストもかかります。

自動操作を行えば、指定した規則に従って複合ネットワークを管理し、オペレーターの作業の複雑性を軽減することもできます。このように、自動操作は、システムおよびネットワークの拡張に役立ちます。

オペレーターの生産性の向上

自動操作はオペレーターの生産性の向上をもたらします。これは、自動操作によって、メッセージやアラートを読む時間や反復作業を行う時間が節約されるからです。オペレーターはその分、新たな問題の解決など、オペレーターの介入を真に必要なとする作業に集中することができます。

一貫した操作手順の確立

自動操作プロシージャを作成し、それらを文書化することによって、操作を体系立てるとともに効果的に確認を行うことができます。自動操作は、組織全体の一貫した操作手順を確立する土台を提供するものです。自動操作を採用することによって、新規の操作手順を速やかに、一貫性を維持しつつインプリメントし、さらに簡単かつ効果的に変更内容を管理することができるようになります。

自動操作のクラス

NetView プログラムを使用すると、さまざまな自動化環境を確立することができます。このセクションでは、自動操作のいくつかのクラスについて説明すると同時に、各クラスを説明する際に本書で使用している用語についても説明します。これらのクラスには、次のものが含まれます。

- 6 ページの『System Automation』
- 6 ページの『ネットワークの自動操作』
- 7 ページの『単一システムの自動操作』
- 8 ページの『複数システムの自動操作』

システムの自動操作とネットワークの自動操作

NetView プログラムでは、プログラム自身の内部処理が自動化できるだけでなく、システムおよびネットワークについても自動操作が可能になります。システムの自動操作とは、オペレーティング・システム、サブシステム、およびアプリケーション・プログラムの自動操作を意味します。一方、ネットワークの自動操作は、VTAM などの通信プログラムを通じた、ネットワーク・リソースの自動操作を意味します。

NetView プログラムを使用すれば、システムの自動操作またはネットワークの自動操作、あるいはその両方を行うことができます。システムおよびネットワークの自動操作を単一の設計で結合すれば、統合された包括的な自動操作を開発することができます。また、問題の原因となり得るすべてのシステム・ハードウェア、システム・ソフトウェア、およびネットワーク装置についての問題判別を行う上で役立てるため、オペレーターに情報の統一ビューを提供することもできます。

メッセージ、管理サービス単位 (MSU)、およびシステム・コマンドの内容は、自動化テーブルで検査されます。その内容に基づいて、自動化テーブルは適切なコマン

ドを送出し、ユーザーのシステムとネットワークを制御します。MSU はデータ構造の一種で、アラートの主ベクトルをはじめとする管理サービス・データを運びます。

NetView プログラム内の MSU に関する詳細については、91 ページの『第 9 章 自動化のための NetView プログラム情報の経路指定』を参照してください。SNA 内の MSU に関する詳細については、「*SNA Management Services Reference*」、および「*Systems Network Architecture Formats*」を参照してください。

System Automation

システムの自動操作とは、システム・メッセージおよび MSU に自動的に応答し、自動的にシステム・コマンドを送出することを意味します。システム・コマンドは、あらかじめスケジュールした時間に送出することも、システム・メッセージや MSU への応答として送出することもできます。

NetView プログラムが、オペレーティング・システム機能を利用してシステムの自動操作を行う際の方法については、33 ページの『オペレーティング・システム自動操作機能と NetView プログラムとの対話』を参照してください。

NetView プログラムの観点からは、システム・メッセージおよびシステム MSU とは、オペレーティング・システム、サブシステム、またはアプリケーション・プログラムが送出するメッセージおよび MSU を意味します。オペレーティング・システム・メッセージのタイプには、以下のものがあります。

- オペレーター宛メッセージ (WTO)
- 要応答オペレーター宛メッセージ (WTOR)

NetView プログラムは、これらのメッセージがコンソールに表示されたり、システム・ログに記録されたりする前に、変更または宛先変更することができます。

システム MSU は、同じシステム上での他のプログラムから、NetView プログラム間インターフェースを通して、あるいは LU 6.2 セッションを通して送信される MSU です。NetView 自動操作においては、システム・メッセージおよび MSU を抑制することも、それらに自動的に応答することもできます。

システム・コマンドとは、オペレーターがシステム、サブシステム、またはアプリケーション・プログラムに対して送出できるコマンドです。自動化環境では、NetView のオペレーター、および自動操作ルーチンは、すべてのシステム・コマンドを使用することができます。例えば、システム・コマンドは、指定の間隔において、あるいは特定のシステム・メッセージに응答して、自動的に送出することができます。

ネットワークの自動操作

ネットワークの自動操作とは、ネットワーク・メッセージおよびネットワーク MSU に自動的に応答し、自動的にネットワーク・コマンドを送出することを意味します。ネットワーク・コマンドは、あらかじめスケジュールした時間に送出することも、ネットワークのメッセージや MSU への応答として送出することもできます。

NetView プログラムは、ネットワーク・リソースのための、ポリシーに基づいて広範囲にわたる自動操作を提供します。AON は、以下のネットワーク・リソースの自動操作を提供します。

- VTAM SNA
- TCP/IP

NetView プログラムは、VTAM を主とする、他のコミュニケーション・ソフトウェアとの対話を通して、ネットワークの自動操作を達成します。ネットワーク管理のために、すでに NetView プログラムを使用している場合は、現在オペレーターが行っている作業の多くをプログラムに任せることによって、ネットワークの自動化を進めることができます。ネットワークの自動操作は、システムの自動操作の場合と異なり、オペレーティング・システムのメッセージ処理機能は使用しません。

ネットワーク・メッセージ およびネットワーク MSU は、VTAM プログラムから、あるいは VTAM を通して、直接ないし間接的に送信されたメッセージおよび MSU です。これらのメッセージには、次のものが含まれます。

- プログラム・オペレーター・インターフェースを介して NetView プログラムに送信される、VTAM メッセージ
- ネットワーク内で発生したハードウェアおよびソフトウェア問題を報告するために、NetView プログラムに送信される MSU

NetView プログラムは、ネットワーク・メッセージおよび MSU を抑制すること、それに自動的に応答することもできます。

ネットワーク・コマンド とは、オペレーターが VTAM に対して、あるいは VTAM を通して、ネットワークの装置に対して送出できるコマンドです。NetView 自動操作機能は、これらのコマンドを数多く使用します。

単一システムと複数システムの自動操作

自動操作は、単一システムあるいは複数システムのいずれに行うかを選択することができます。新規に自動操作を行うときは、単一システムの自動操作から始めてください。つまり、最初にできるだけ多くの操作をローカルで (各システムで) 自動化し、その後、複数システムの自動化に進みます。その際、他のシステムとの間に必要な対話の数を減らして、完全な複数システムの自動操作を実現するようにします。さらに、コミュニケーション機能、フォーカル・ポイント・システム、通信回線などが過剰負荷にならないようにします。

単一システムおよび複数システムの自動操作の段階ごとの説明については、8 ページの『自動操作の段階』を参照してください。

単一システムの自動操作

単一システムの自動操作 では、各ホスト・システムの自動操作は自己完結型です。自動操作できるのは、そのシステムの VTAM プログラムのドメイン内にある、システムとそのサブシステムおよびアプリケーション・プログラム、およびネットワーク装置です。ただし、単一システムの自動操作では、NetView プログラムは、VTAM ドメイン外にある装置を自動操作することはできません。オペレーティング・システムのリカバリーや初期プログラム・ロードなど、ローカルで自動操作できないこれらのタスクの処理は、オペレーターが行います。

複数システムの自動操作

複数システムの自動操作では、2 つ以上のホスト・システムにわたる自動操作の調整を行います。調整を行うことによって、単一システム上でローカルでは自動化できなかったリソースの操作も、自動化できるようになります。複数システムの自動操作は、調整が単一のデータ・センターをまとめるものか、リモート・ロケーションの複数のデータ・センターをまとめるものかによって、単一サイト か複数サイトの いずれかになります。

複数システムの自動操作によって、集中操作 と呼ばれるリモート操作を確立することができます。集中操作では多くの場合システムにオペレーターが存在せず、全面的なオペレーター・インターフェースも必要ありません。つまり、オペレーター不在のシステムをリモートから操作するわけです。この操作では、ローカルで自動操作できない問題報告書とともに、不在システムの状態に関する情報を中央システムに送ります。

自動操作の段階

NetView 自動操作には、さまざまな技術が含まれます。これらの技術は、使用される環境によって次のものに分類することができます。

- 単一システムで使用されるもの
- 複数システム環境で使用されるもの
- NetView プログラムが稼働していないシステムまたは非 SNA 装置専用のもの

自動化の段階の例については、21 ページの『段階的アプローチの例』を参照してください。

単一システムの自動操作の段階

このセクションでは、単一システムにおける自動操作システムおよびネットワーク管理の基本的な方法を紹介します。これらの方法は、ユーザーが実施するおおよその順序に基づいて、7 つの段階に分類することができます。

本章では、単一または複数システムの自動操作を段階的に紹介するために、いくつかの図を設けており、図 1 はその最初の例です。自動操作を複数システムに拡張する方法については、後の図で示します。

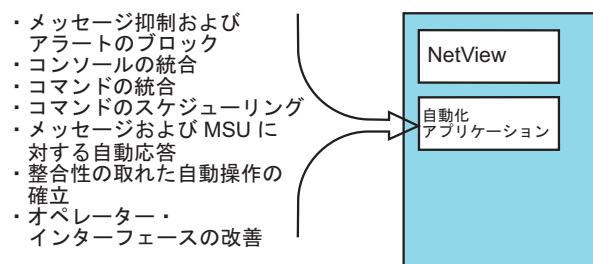


図 1. NetView プログラムが単一システム上にあるときの自動化の追加

単一システムの自動操作における最初の 3 つの段階では、NetView 自動操作を使用して、オペレーターが情報を処理する際のスピードと正確さを向上させます。

- メッセージの抑制およびアラートのブロック
- コンソールの統合
- コマンドの統合

続く 3 つの段階では、NetView プログラムが次の管理タスクを自動的に実行するようにして、オペレーターのワークロードをさらに軽減します。

- コマンドのスケジュール
- メッセージおよび MSU に対する応答
- 整合性のとれた自動操作の確立

オペレーター・インターフェースの改善を図る最後の段階では、新たな環境および軽減されたワークロードにオペレーター・インターフェースを適合させます。

メッセージの抑制または改訂とアラートのブロック

小規模のデータ処理センターでも、おそらく、オペレーターが無視しても構わないメッセージやアラートは数多くあります。規模の大きなセンターであれば、1 秒あたりに受け取るメッセージやアラートの数は何百にものぼります。ところが、その中でオペレーターが決定を下すのに必要なデータを含むものはほんの一部にすぎません。

自動操作への第一歩は、ルーチン・メッセージおよびルーチン・アラートを抑える、つまりブロックすることから始まります。これによって、オペレーターが受け取る不必要な情報の量を減らすことができます。この結果、オペレーターは、必要な情報に集中することができます。

メッセージの数を減らすと、システム全体の負荷も軽減されます。その結果、システムは重要なメッセージを効率的に処理できるようになります。抑制したメッセージは引き続きログに記録することができるので、アプリケーション・プログラムのデバッグや自動操作の監査などを行う際に利用することができます。

メッセージの抑制または改訂は、NetView プログラムの複数の機能を組み合わせることによって行うことができます。

z/OS メッセージ処理機能 (MPF) は引き続きサポートされていますが、MPF の機能の多くは NetView 改訂テーブルまたは NetView 自動化テーブルの機能に取って代わられています。ほとんどのシステム・メッセージの場合は、改訂テーブルを使用して抑制または改訂します。より複雑なアクション (例えば、自動化コマンド・リストの初期化など) を必要とするメッセージの場合のみ、NetView アドレス・スペースに渡して自動化テーブルで処理する必要があります。また、自動化テーブルを使用して、NetView プログラムが直接受信する NetView メッセージと VTAM メッセージのうち、不必要なメッセージを抑制することもできます。

不要なアラートをブロックするには、まず、どの問題レコードがイベントになり、どのイベントがアラートになるかを判別します。そうすると、SRFILTER コマンドを使用して、ハードウェア・モニターの記録フィルターを設定することができます。SRFILTER コマンドによるコマンドのフィルター操作について詳しくは、NetView オンライン・ヘルプを参照してください。

NetView 自動化テーブルに、フィルター・ステートメントを追加することも可能です。NetView 自動化テーブルには、着信メッセージおよび MSU に対する処理オプションと自動応答が含まれています。自動化テーブル・ステートメントは、ハードウェア・モニターの記録フィルターを指定変更することができます。

コンソールの統合

メッセージの流れが減少すれば、いくつかのコンソールを結合することが可能になります。詳しくは、13 ページの『オペレーター・インターフェースの改善』および 423 ページの『第 26 章 集中操作』を参照してください。

不要なメッセージを抑制したあとでは、残りのメッセージを 1 つか 2 つのコンソールに経路指定することができます。これらのメッセージをオペレーターに表示するには、いくつかの方法があります。

コンソールの削減: ハードウェア・モニターから別のインターフェースに情報を移動させることによって、オペレーターが監視する NetView コンソールの数を減らすことができます。ハードウェア・モニターの使用を減らすか、まったく使用しない場合は、別の形式でアラート情報を表示してください。例えば、自動応答で処理できないアラートは、メッセージに変換するか、またはフルスクリーン・パネルに表示してください。自動化テーブルで、この処理を開始することができます。

オペレーターへのメッセージを減らすかまたは除去し、ハードウェア・モニターへの依存度を高くすることによって、オペレーターがモニターする NetView コンソールの数を減らすことができます。自動操作では処理できない問題が発生した場合は、ハードウェア・モニター・アラートを生成して、オペレーターに通知することができます。そのあとで、ハードウェア・モニターを使用すると、オペレーターによる問題解決に役立つ情報を表示することができます。

メッセージの収集によるコンソールの統合: コンソールを統合するには、次に示すようなさまざまなソースからのメッセージを NetView プログラムで収集します。

- オペレーティング・システム
- 情報管理システム (IMS™) プログラムまたは顧客情報管理システム (CICS®) プログラムのマスター・オペレーター・コンソール (NetView の端末アクセス機能を介して取得)
- その他のサブシステムおよびアプリケーション・プログラム
- プロセッサ・ハードウェア・コンソール (Tivoli System Automation for z/OS 製品のプロセッサ・オペレーション・コンポーネントを介して取得)
- VTAM アプリケーション・プログラム
- VTAM

統合によって、中央操作域の 2、3 台、あるいは 1 台だけにコンソールを減らすことができるでしょう。そうすれば、2、3 人ないし 1 人のオペレーターだけで、システムおよびネットワークの制御に必要なメッセージのすべてを受け取れるようになります。オペレーターが複数いる場合は、ある特定のオペレーターに必要なメッセージすべてを、そのオペレーターのコンソールに表示し、それ以外のメッセージは表示しないようにすることができます。このため、1 人のオペレーターが一度にいくつものコンソールを監視したり、他のオペレーターのメッセージに目を移したりする必要もなくなります。

NetView コンソールの専用化: システムおよびネットワークを管理するために使用するコンソールは、統合することができます。

1 つのコンソールを NetView コンソールの管理専用 (コマンド機能使用) にし、別のコンソールをネットワークの管理専用 (ハードウェア・モニター使用) にすることができます。

NetView コンソールをカスタマイズすれば、他のコンソールを利用しているオペレーターでも、NetView プログラムを簡単に扱えるようになります。ただし、NetView コンソールのカスタマイズ機能は、オペレーティング・システムの機能によって異なる場合があります。コンソールのカスタマイズの例としては、メッセージのカラー化やメッセージの接頭部の変更などが挙げられます。(コンソールのカスタマイズは、改訂テーブルを使用して行うこともできます。)

コンソールのカスタマイズについては、「*IBM Tivoli NetView for z/OS* カスタマイズ・ガイド」を参照してください。

コマンドの統合

コマンド・プロシーチャーを単純化することによって、操作性の向上が望めます。オペレーターがどのようなシーケンスでコマンドを最も一般的に出しているかを調べて、そのシーケンスを自動的に出すような、簡潔なプログラム (コマンド・プロシーチャーと呼ばれる) を作成します。オペレーターがそのコマンド・プロシーチャーの名前を入力するだけで、NetView プログラムが、そのシーケンスに含まれているすべてのコマンドを出してくれます。

例えば、次のアクションのいずれかを実行するために、簡単なコマンド・プロシーチャーを作成することができます。

- 直接アクセス記憶装置 (DASD) のバンクをオンラインにして、各ボリュームに正しい属性を与える
- 修復後に一連の通信回線を再確立する
- 必要な DASD の検証を含む、単純なアプリケーションの初期設定を行う
- 満杯になったシステム管理機能 (SMF) データ・セット、またはダンプ・データ・セットのダンプを行う
- 操作チェックリストをモニターする

オペレーター用のコマンド・プロシーチャーを作成すると、各オペレーターの行うべき入力作業は減ります。したがって、オペレーターの生産性は向上し、コマンドの入力ミスによるエラーの発生する確率は減少します。さらに、コマンド・プロシーチャーは後の自動操作のベースともなります。NetView 自動化テーブル、またはタイマー・コマンドを使用すると、同じプロシーチャーのいくつかを自動的に呼び出すことができるからです。

コマンドのスケジューリング

コマンドを特定の時刻に出したい場合、または所定のコマンドを定期的に出したい場合は、コマンド・スケジューリング、および NetView のタイマー・コマンドを使用することができます。例えば、アプリケーションを午後 5:00 にシャットダウンして、テープの転送など特別な活動のためにプロセッサのキャパシティを解放する必要がある場合や、ある作業の状況を 3 分おきにチェックしたい場合などです。

ここで出すコマンドは、コマンド・プロシーチャーでも構いません。例えば、アプリケーション・プログラムを初期設定するための、単純なコマンド・プロシーチャーを作成したとします。そのプログラムを毎日午前 6:00 に初期設定したいのであれば、そのコマンド・プロシーチャーを毎日その時刻に実行させればよいのです。

コマンドをスケジュールリングすることによって、オペレーターは、コマンドを手操作で出す手間を省くことができます。また、オペレーターの手が空いていないときでもアクションを起こすことができ、オペレーターには不可能な頻度で特定のコマンドを繰り返すこともできます。

メッセージと MSU に対する自動応答

オペレーターは、メッセージや MSU のようなイベントに対する応答に、多くの時間を費やすことがよくあります。多くの場合、NetView プログラムは、オペレーターの応答を自動的に出すことができます。

NetView プログラムが提供する自動化テーブルは、着信メッセージおよび MSU を調べてさまざまなアクションを行います。あるメッセージまたは MSU に対して「コマンドの発行」などの対応を指定しておけば、NetView 自動化テーブルがそのアクションを開始します。例えば、すべての IOS150I メッセージ (障害の起こった装置が使用可能になったことを示す) に対し、NetView プログラムを自動応答させることが可能です。NetView 自動化テーブルは、MVS VARY ONLINE コマンドを出して、その装置をオンラインに戻すことができます。

一般的なメッセージおよびアラートの大部分に対して自動応答をするように、NetView プログラムのプログラムを組めば、それらのメッセージやアラートを表示させないですみます。それにより、自動化で解決している問題に関する通知を、オペレーターに表示する必要がなくなります。

整合性のとれた自動操作の確立

整合性のとれた自動操作は、操作の自動化をさらに推し進めた段階です。整合性のとれた自動操作では、NetView プログラムは、各データ処理リソースの望ましい状態と実際の状態とを継続的に追跡します。実際の状態が望ましい状態と異なる場合は、自動操作が訂正を行います。

各リソースの望ましい状態は、プログラマーまたはオペレーターが設定します。リソースに含まれるものとしては、チャンネルなどのハードウェア・コンポーネント、および MVS タイム・シェアリング・オプション (TSO) のデータ・セットやアドレス・スペースなどの、ソフトウェア・コンポーネントが挙げられます。オペレーターがリソースの望ましい状態を検討ないし変更する際に役立つコマンド・プロシーチャーを、作成することも可能です。

自動操作では、各リソースの実際の状態を判別するために、受動モニターおよび能動モニターを採用することができます。受動モニターとは、状況の変化を示すメッセージおよびアラートが出されるのを待つことを意味します。能動モニターとは、状況情報を送るよう請求するコマンドを出すことを示します。例えば、コマンド・プロシーチャーを 10 分ごとに実行するようにセットアップして、重要なリソースの状況をチェックするためのコマンドを出すとします。受動モニターと能動モニターを組み合わせるにより、自動操作で、信頼性のある最新情報を確実に得ることができます。

自動化アプリケーション・プログラムは、リソースの状態に関する情報を受け取ると、グローバル変数を更新することによってその情報を記録します。例えば IMS プログラムに障害が起こった場合、IMS プログラムの状態を表すグローバル変数の値を DOWN に変更することが可能です。

あるリソースにとっての望ましい状態または実際の状態に変化が生じた場合は、修正のためのアクションが必要かどうかを自動化が判別します。もし必要であれば、自動操作では、状態を変更するためのコマンドまたはコマンド・プロシーチャーを出すことができます。また、自動操作により、オペレーターに状態の変更を通知することもできます。

ユーザーは、望ましい状態と実際の状態以外に他の情報を追跡することもできます。例えば、ある変数を使用して、各リソースに対して実行される自動化アクションを指定することができます。あるいは、自動操作が責任を負うリソースを指定することもできます。この場合も、自動操作はすべてのリソースをモニターしますが、問題解決を試みるのはユーザーが指定したリソースに限られます。この手法によって、自動操作の一部を停止するように変数を変更して、任意のリソースを手操作制御に戻すことができます。

NetView プログラムには、自動操作のサンプルが組み込まれています。このサンプルは、NetView のグローバル変数を使用した、整合性のとれた自動操作の例です。整合性のとれた自動操作をインプリメントする前に、サンプルを学習しておいてください。

サンプル・セットに関する詳細については、651 ページの『付録 I. 自動操作のサンプル・セット』を参照してください。

RODM による自動操作の統合

先に述べた技法の他に、NetView プログラムのリソース・オブジェクト・データ・マネージャー (RODM) コンポーネントが提供するいくつかの機能を使って、自動操作を統合する技法もあります。これらの RODM 機能は、リソース情報の追跡および問題解決の自動操作の際に役立ちます。RODM は、リソースとイベントおよびそれらの関係に関するさまざまなタイプの情報を保持することができます。RODM 内の情報間に複雑な関係を指定できるため、NetView プログラムでは、複数のイベントの相互関係を決定し、それらを利用して問題の分析と解決を行うことができます。

オペレーター・インターフェースの改善

自動操作によって、データ処理環境の実行に必要とされる人間の介入量が減ります。しかし、オペレーターが環境のモニターおよびリソース状況の検討を行い、自動操作が正しく機能しているかどうかを検証する必要があります。

そこで必要になってくるのが、例外通知のメカニズムです。例外通知 とは、自動操作のルーチンが、まだ自動化されていないイベントに遭遇したとき、あるいはルーチンが問題解決に失敗したときに、オペレーターにその旨を通知するプロセスのことをいいます。

したがって、オペレーターに必要な情報を与えるためのインターフェースを計画しなければなりません。情報は、オペレーターに対して次の形式で示すことができます。

- コマンド機能からのメッセージ
- ハードウェア・モニターからのアラート
- 状況モニターと NetView 管理コンソール (NMC) による状況情報
- VIEW コマンド・プロセッサで表示されるフルスクリーン表示およびヘルプ・パネル

メッセージによる情報提供

NetView コンソールには多くのメッセージが表示され、NetView プログラムおよびプログラムの管理下にあるプロダクトに関する情報を提供します。これらのメッセージを表示するには、NetView コンソールからコマンド機能进行操作します。自動化されていない環境では、多くの場合、NetView オペレーターは、この機能を頻繁にモニターしなければなりません。

自動操作により、コマンド機能の使用状況を改善することができます。メッセージの抑止により、表示されるメッセージの数が減るため、残ったメッセージの読み取りが容易になります。例外通知用のコマンド機能を利用すると、自動操作ルーチンが問題に遭遇したら必ずメッセージが出るようにすることができます。

コンソールを統合すれば、オペレーターは、オペレーティング・システムと NetView プログラムというように、1 つの画面で複数のプロダクトをモニターできるようになります。さらに、自動化テーブルを使用すると、画面上に重要なメッセージを保持したり、テキストを変更してメッセージを再度出すことができます。

自動操作には、メッセージの重要性がオペレーターによくわかるように、特定タイプのメッセージの表示方法を制御する機能もあります。例えば、システムに、クラスの異なるメッセージを別の色や強調表示で表示させることができます。また、異なるグループのメッセージの情報の配置を換えて形式設定できます。画面表示に関するこれらの変更は、画面フォーマット・メンバーを使用して行います。

画面フォーマット・メンバーに関する詳細については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。

NetView プログラムでは、表示用に指定した数のメッセージを保管できます。メッセージの数が指定数を超えた場合は、最も古いメッセージから破棄されます。しかし、メッセージに基づく自動操作は継続し、すべてのメッセージがログに記録されます。

ハードウェア・モニター・アラートによる情報提供

ハードウェア・モニターはイベントおよびアラートの形で情報を受け取り、その情報を表示します。これらのイベントおよびアラートは MSU やその他のデータ構造で、NetView プログラムへ渡されるものです。アラートは主として、ネットワーク・ハードウェアに問題が発生したことを示します。

注: ハードウェア・モニターが自動化テーブルに提供するのは、非送信請求 MSU に限られます。

非自動化環境の場合と同じようにハードウェア・モニターを他の機能と併用し、リソース情報を表示させることが可能です。これを行うために、ハードウェア・モニターからのオペレーターに対するアラートを、1 つ以上のコンソールに表示させるようにできます。オペレーターはこれらのアラートを、ネットワーク問題の管理に使用することができます。自動操作状況およびその他の情報は、独立したコンソール上に、別の形式（メッセージやフルスクリーン・パネルなど）で表示することができます。

ハードウェア・モニターの活用法: オペレーターは問題記述、考えられる原因のリスト、および推奨アクションのリストを表示することができます。ハードウェア・モニターには、他にも次のような機能があります。

- 報告された問題の記録を維持する
- どのオペレーターがどのアラートを見るかを決定する目視フィルターを提供する
- 情報/管理プログラム、ユーザー定義の外部ログ、またはシステム管理機能 (SMF) 外部ログに、情報を送信できるようにする。

アラートの生成: 独自のアラートを生成するには、NetView プログラムの GENALERT コマンド、プログラム間インターフェース、あるいは管理サービス (MS) トランスポートを利用します。受け取るメッセージの大半を抑止または自動化した後で、アラートを使用して、残りのメッセージと、自動操作に関して発生したすべての問題をオペレーターに通知してください。

GENALERT コマンドは、特定のメッセージを受け取ったときに自動化テーブルから出すことができます。あるいは、コマンド・プロシーチャーから出すこともできます。表示するアラートの内容は、ユーザー自身で制御しなければなりません。アラートの内容には、説明、推奨アクション、連絡先の電話番号、それぞれの環境に合ったその他の情報が含まれています。

アラートを形式化し、プログラム間インターフェース (PPI) PIPE ステージを介してアラートを送信する REXX コマンドを作成することもできます。

ビーパー/電子メール・アクションによる情報提供

INFORM コマンドとそれに関連したポリシー定義を使用すると、ビーパーまたは電子メール・アクションを生成して、主要なイベントまたはアクションを担当者に通知することができます。例えば、勤務時間外に、あるいはリモート・ロケーションをサポートするために、ビーパーまたは電子メール・アクションを使用することができます。

詳しくは、「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」を参照してください。

状況情報の提供

状況モニター、および NetView 管理コンソールで、ネットワークの状況を追跡することができます。したがって、過去のメッセージのシーケンスを見直したり、照会コマンドを出したりしなくても、状況を判断することができます。状況モニターは、状況情報をテキストの形式で状況モニター・パネルに表示します。NetView 管理コンソールは、ワークステーションの画面上に、ネットワークの状況情報をグラ

フィックで表示します。自動操作がイベントに応答してリソースをアクティブに保っているのに対して、オペレーターは、状況表示画面を使ってネットワークを効率的にモニターすることができます。

NetView 管理コンソールは、MVS システムに接続されたワークステーション上に情報を表示することができます。この機能は単一のシステムで実行することもできますが、複数システム環境において非常に役立ちます。状況情報を MVS システムに転送すれば、その他のシステムに関する情報もグラフィックで表示することができます。

状況情報の転送については、423 ページの『第 26 章 集中操作』を参照してください。

情報のフルスクリーン・パネルでの表示

インターフェース設計をより柔軟に行うために、コマンド・プロシーチャーから表示されるフルスクリーン・パネルを作成することができます。フルスクリーン・パネルでは多くのカラー表示および強調表示オプションを提供しており、これらを利用して状況情報や例外通知、あるいはその両方を表示することができます。

フルスクリーン・パネルで表示されている間、NetView プログラムは、自動的にメッセージの表示を据え置きします。ただし自動操作およびメッセージのログ記録は、パネルが表示されている間も継続されます。

対話式システム生産性向上機能 (ISPF) などの標準的なエディターを使用して、フルスクリーン・パネルを作成することができます。

作成したパネルは、VIEW コマンドを使用してコマンド・プロシーチャーから表示することができます。VIEW コマンドを使用すれば、ユーザーは、データのカラー表示および強調表示オプションを指定できるだけでなく、指定したフィールドに入力することも可能です。この入力はその後コマンド・プロシーチャーに送り返されるので、自動操作ルーチンとオペレーターとの対話的通信が可能になります。

フルスクリーン・パネルの表示方法の例については、標準エディターを使用して、VIEW コマンドを使用する NetView コマンド・リストを調べてみてください。これらのコマンド・リストには、BROWSE、TUTOR、および DISG があります。

HELP コマンドも VIEW コマンドを使用します。したがって、独自のヘルプ・パネルを作成したり、既存の NetView のヘルプ・パネルを変更することができます。作成した自動操作を記述する情報、コマンド・プロシーチャーを使用するオペレーターを支援する情報、およびそれぞれのネットワーク環境を反映するカスタマイズされた情報の表示が可能です。

単一システムの自動化の伝搬

複数システムおよびネットワーク全域にまたがる自動操作を実現するためには、まず、単一システムの自動化を NetView システム全体に伝搬させます (17 ページの図 2 を参照)。ただし場合によっては、各システムごとに自動操作を新たに設計しなければならないことがあります。なぜなら、ネットワークの各システムには、異なるアプリケーション・プログラムまたは装置がインストールされている場合があるからです。ただし、1 つのシステムに単一システムの自動操作をインプリメントし

たのであれば、その自動操作の大部分を他のシステムに伝搬させることができるでしょう。

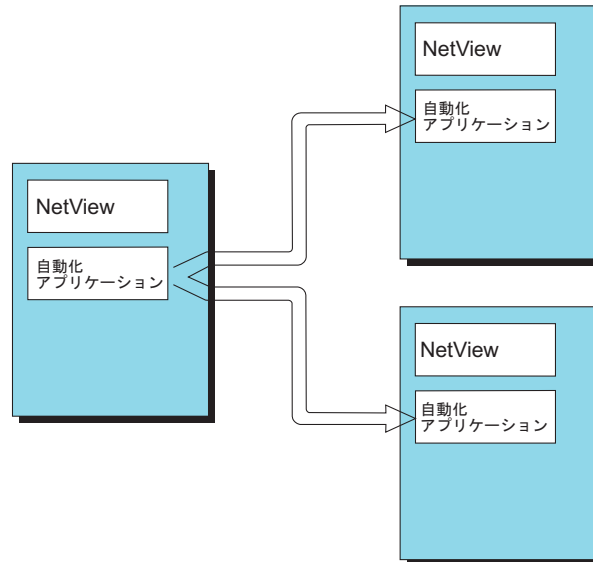


図 2. 追加システムへの自動操作の伝搬

コピーした自動操作を新たなシステム用にカスタマイズする場合、必要な変更の数は、新しい環境が先に自動操作を完了した環境とどれくらい違うかによって決まります。

伝搬のため柔軟な設計を使用します。移送可能な自動操作の設計方法については、58 ページの『拡張と伝搬の設計』を参照してください。

操作の集中化

単一システム統合の結果である、集中化された操作においては、ネットワーク中に散らばった多数のシステムからの情報を、単一コンソールまたはコンソール群に経路指定できます。そうすれば、オペレーターはそれぞれのシステムを、別のコンソールから実行させる必要がなくなります。

システム間の通信機能が過負荷になるのを防ぐため、問題を他のシステムに送信するのは、なるべく多くの問題に対する応答をローカルで自動化してからにしてください。次の 2 種類の情報に限定して転送してください。

- 個々のシステムの条件に関する情報 (オペレーターへの表示を目的とする)
- 個々のシステムでは自動的に解決できず、他のシステムからの支援を必要とする問題に関する情報

これらの問題には、オペレーターの注意が必要なものや、プロセッサやオペレーティング・システム、NetView プログラムの再始動を要求するものが含まれます。

18 ページの図 3 に示すように、1 つのシステムをフォーカル・ポイントに指定して、分散データ・システムから転送される例外を受け取るためのフォーカル・ポイントにすることができます。フォーカル・ポイント・システムで NetView プログラムにログオンすれば、オペレーターは一群のシステム、あるいはデータ・センター

全体、またはいくつかのデータ・センターを管理できるようになります。

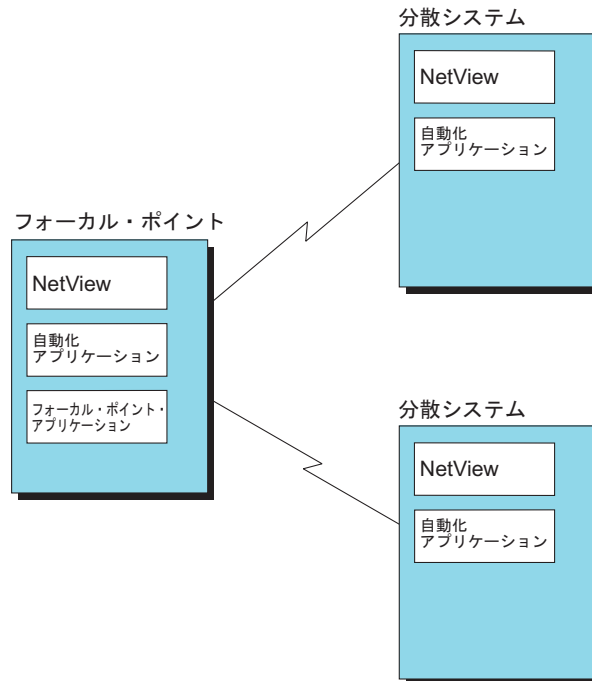


図 3. ローカルの自動操作が処理できない例外情報の転送

集中操作におけるフォーカル・ポイントの使用

自動操作が単一の地点で行われる場合も、複数の地点で行われる場合も、フォーカル・ポイント・システムでは次の 2 種類のアクションを実行します。

- フォーカル・ポイント・システムは、自身のシステムとネットワーク管理を自動化します。したがって、フォーカル・ポイント・システムの自動化を行う際は、他のシステムで使用しているのと同じタイプの単一システム自動操作を使用してください。
- フォーカル・ポイント・システムは、フォーカル・ポイントに報告を行うシステム (分散、ターゲット、またはエントリー・ポイント・システムと呼ばれる) からの情報を自動化します。

ターゲット・システムまたはフォーカル・ポイントで自動化できない情報は、フォーカル・ポイント・システムのオペレーターに提示されます。

フォーカル・ポイントと分散データ・システムをうまく調整すれば、シフト時間外は特定のデータ・センターにはオペレーターを置かずに、そのデータ・センターをリモート操作することが可能になります。このようなシフト体制の間は、オペレーター不在のデータ・センターの分散システムからの情報を、オペレーターのいるデータ・センターのフォーカル・ポイント・システムに転送することができます。しかし、自動化されたデータ・センターをオペレーター不在のまま稼働させるとはいつても、テープのマウントやプリンターの扱いなど、なんらかの手操作介入が必要になる可能性は残ります。

手操作介入の必要を減らす方法については、41 ページの『自動操作に関連した機能およびサービス』を参照してください。

NetView プログラムを利用すれば、メッセージやアラートをはじめ、NetView 管理コンソール (NMC) で使用される状況情報を転送することができます。また NetView プログラムでは、管理サービス・トランスポートを使用するアプリケーション・プログラムについても、そのアプリケーション・プログラムのフォーカル・ポイントを追跡することによって、情報転送を支援することができます。

フォーカル・ポイント・システムが障害で停止すると、他の多くのシステムの管理も中断されることになるので、フォーカル・ポイントには信頼性の高いシステムを選んでください。また、計画または計画外の障害停止の場合には、バックアップ・フォーカル・ポイントによって制御を行うように指定することもできます。

フォーカル・ポイントとして信頼できるシステムを選択する際の基準については、64 ページの『フォーカル・ポイントの選択』を参照してください。バックアップ用のフォーカル・ポイントを選択する際の情報については、66 ページの『バックアップ・フォーカル・ポイントの使用』を参照してください。

リモート操作の確立

先に述べた段階を実施すれば、分散システムは大部分の操作を自動化できます。残りの操作活動に関する情報はフォーカル・ポイントに転送され、そこで、自動操作と集中操作によって、リモートでの手操作介入を必要としない状況が処理されます。

ターゲット・プロセッサのハードウェア・コンソールとシステム・コンソールが関与するアクションを自動化することによって、複数システムの自動化を行うことができます。これらのコンソールにかかわるアクションとしては、ターゲット・プロセッサの初期設定、構成、およびシャットダウンなどが挙げられます。ほとんどの IBM プロセッサに対して、これらのアクションを達成するために、IBM システム自動化支援プログラム (z/OS) を使用することができます。システム自動化支援プログラム (z/OS) を使用して、ターゲット・システムをリモートから初期設定する方法については、20 ページの図 4 を参照してください。

システム自動化支援プログラム (z/OS) の機能の概要については、40 ページの『IBM Tivoli System Automation for z/OS』を参照してください。

I

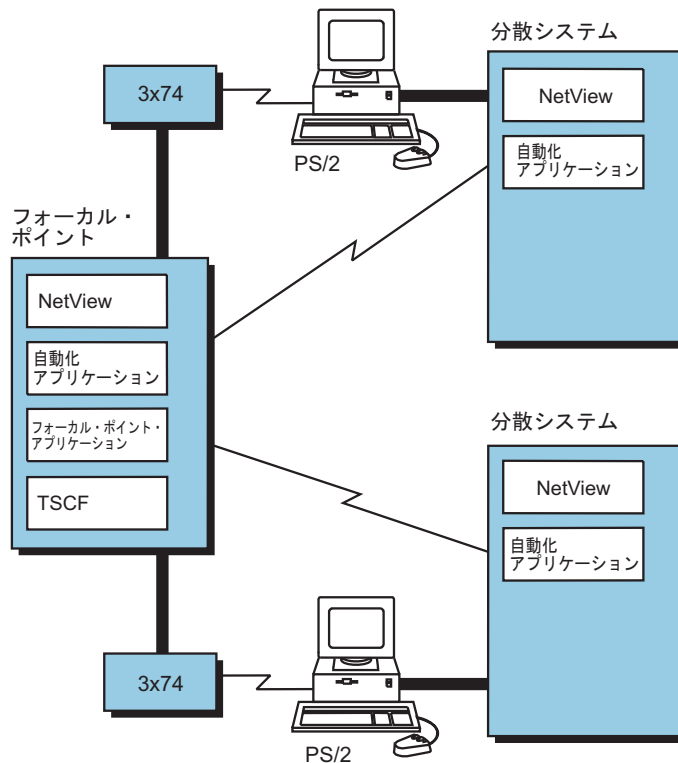


図4. ターゲット・システムのリモート初期設定

システム自動化支援プログラム (z/OS) は、エンタープライズ・システム/4381、エンタープライズ・システム/3080、エンタープライズ・システム/3090、および 大部分のエンタープライズ・システム/9000 (ES/9000) のプロセッサを制御することはできますが、9370 プロセッサをリモートで初期化することはできません。

しかし 9370 の自動電源制御 (APC) 機能を使用すれば、初期プログラム・ロード (IPL) を自動化することができます。9370 の電源が入るようにタイマーをセットしておけば、9370 が IPL を実行し、オペレーティング・システムを始動させます。

続いて、オペレーティング・システムが NetView プログラムを始動させ、これを受けて、NetView が、ユーザーのシステムおよびネットワークの自動化を行います。また APC を利用すれば、リカバリー状態の初期設定に利用されるモデムあるいは RS-232 装置を通して、リモートから電源を入れることもできます。

9370 システムは、リモート・オペレーター機能 (ROF) も提供しています。この機能は、リモート・コンソール機能を提供するものであり、分散 9370 を、中央サイトから制御できるようにします。ワークステーションで ROF を実行させれば、中央サイトにいるオペレーターは、ダイヤル接続を通して、リモート 9370 サービス・プロセッサのハードウェア、およびオペレーティング・システムを制御することができます。

注: システム自動化支援プログラム (z/OS) は、ラック・マウントの ES/9000 プロセッサ (120、130、150、および 170 型) はサポートしていません。NetView の RUNCMD コマンドを使用して、これらのプロセッサをリモートから初期設定するには ES/9000 のプロセッサ・コンソールに初期設定コマンドを送信します。こ

これらの初期設定コマンドを送信するためのコマンド・プロシーチャーを作成すれば、RUNCMD コマンドの正確な入力がさらに確実になります。

ES/9000 プロセッサの詳細については、「Enterprise System/9000 120、130、150、170 型分散プロセッサの管理」を参照してください。

NetView プログラムが稼働していないシステムおよび非 SNA 装置の自動化

ターゲット・システムで NetView プログラムが稼働していない場合でも、NetView プログラムを使って、多くのターゲット・システムを自動化することができます。またネットワーク上にある装置であれば、SNA プロトコルを使用していない装置や、VTAM に報告を行っていない装置についても、NetView プログラムを使って、ネットワーク装置を自動化することが可能です。

NetView プログラムが稼働していないシステム、または非 SNA 装置の場合、NetView 自動化機能が有効か否かは、システムまたは装置の機能によって決まります。システムまたは装置が、問題の報告やその他の情報を NetView プログラムが解釈できるような形式 (メッセージや MSU など) で送信でき、NetView プログラムからのコマンドを受信できればなりません。

プロダクトによっては、NetView プログラムを使用して直接自動化することが可能です。その他のプロダクトについては、既存の NetView インターフェースを使用して、あるいはユーザー自身のインターフェースを作成することによって、間接的に自動化を行うことができます。

段階的アプローチの例

オペレーターが、メッセージやアラートなどのイベント通知の順調な流れをモニターすることによって、システムを管理している一般的な環境においては、オペレーターは各イベントを監視し、イベントがなんらかの問題を示している場合には対応します。このような運用手法は、イベント・モニター環境と呼ぶことができます。

この例では、以下、順を追って、企業内システムの自動化のための段階的アプローチについて説明します。アプローチはイベント・モニター環境から始まって例外モニター環境に移り、さらに集中操作環境へと進みます。集中操作環境では、大多数のイベントおよび問題への自動応答が可能になります。

残りの部分はわずかですがこれについては、単一のフォーカル・ポイント・システムに通知が送られます。NetView システムは、フォーカル・ポイントとして、効率的なインターフェースを用いて問題を記述するので、オペレーターは状況を速やかに理解して、適切なアクションを施すことができます。

オペレーターに新しい環境を理解してもらうために、ネットワークの自動化の方法を文書化し、その後、プロシーチャーまたはラン・ブック (run book) を更新します。

第 1 段階: メッセージの抑止とアラートのフィルター操作

不要な通知は、遮断します。この段階をセットアップした後では、オペレーターが、限定された通知で環境をモニターするのに慣れるまで、時間をとります。この手順を実施する前に、オペレーターに通知してください。

第 2 段階: コンソールの統合

メッセージのモニターに必要なコンソールの数が減り、それぞれのコンソールが受け取るメッセージの割合も少なくなります。オペレーティング・システムから NetView プログラムに、抑止されなかったメッセージを転送してください。

第 3 段階: コマンドの統合

オペレーターその他の情報をもとにして、オペレーターがタスクを実行する際に最も頻繁に利用する、プロシーチャーおよびコマンド群を確認してください。次に、オペレーターがそれらのタスクを効果的に実行できるような、簡単なコマンド・プロシーチャーを作成してください。

第 4 段階: コマンドのスケジューリング

コマンドのスケジューリングを使用して、反復して行うオペレーター作業を実行するためのタイマー・コマンドを発行します。

第 5 段階: メッセージおよび MSU への自動応答の作成

NetView 自動化テーブルを使用して、共通メッセージおよび MSU に対する自動応答を出します。これによって、オペレーターに表示されるメッセージおよびアラートの割合が減り、分刻みのシステムおよびネットワーク操作におけるオペレーターの役割を軽減することができます。

第 6 段階: モニターおよび再活動化の整合

整合性の取れたシステムを作成して、これまでオペレーターが管理してきたプログラクトのモニターと再活動化を行います。この段階では以下のことを行います。

1. 例えば、グローバル変数または RODM を使用して、それぞれのプログラムまたはリソースの状態を追跡する。
2. メッセージおよびアラートを監視して、それぞれのリソースがどの状態にあるかを判別する。
3. コマンド・プロシーチャーを出して違いを正す。

この段階では、従来オペレーターが行っていた機械的な反復作業の最後の部分が省略されるので、イベントのモニターから例外のモニターへ一歩進んだこととなります。オペレーターは、メッセージやアラートの絶え間ない流れを見る必要がなくなります。代わりに、オペレーターは、要約された状況情報と、自動操作では処理できない例外的問題の通知を見るだけで済みます。

第 7 段階: オペレーター・インターフェースの改善

オペレーターは、メッセージやアラートに対して、コマンド機能やハードウェア・モニターを絶え間なくモニターする必要がなくなります。代わりに使用するインターフェースは、VIEW コマンドで表示されるフルスクリーン・パネルなど、状況表示や例外通知により適したものです。

第 8 段階: 複数システムの自動化

単一システムの自動化から、複数システムの自動化へと進みます。複数システムのエンタープライズを自動化する際は、まず、単一システムの自動化を、あらゆる NetView システムに伝搬させます。

第 9 段階: 操作の集中化

まず、1 つのシステムをフォーカル・ポイントに選びます。次に、他のシステムからの例外通知をそのフォーカル・ポイントに転送します。そのフォーカル・ポイントからすべてのシステムを操作できるようになるので、他のシステムにオペレーターを配置する必要がなくなります。企業がいくつかのデータ・センターまたはサイトに分散している場合でも、リモート初期設定を行うことができます。

第 10 段階: 他の機械および装置への自動操作の拡張

NetView プログラム の AON (自動化オペレーション・ネットワーク) コンポーネントを使用して、IBM 以外のシステムと非 SNA 装置を含むほとんどのデータ処理装置を管理することができます。具体的な情報については、499 ページの『第 31 章 自動化オペレーション・ネットワークの使用』を参照してください。

第 2 章 自動操作プロダクトの概要

この章では、NetView プログラムの自動操作で使用する主なプロダクトと、自動化環境におけるそれらの役割、およびプロダクト間の関係について説明します。ここでは特に、次のプロダクトに関する概要を紹介します。

- NetView プログラム自動操作機能
- 33 ページの『オペレーティング・システム自動操作機能と NetView プログラムとの対話』
- 自動化環境を提供する、その他の IBM プログラム
- 40 ページの『NetView インターフェースの使用例』
- 41 ページの『自動操作に関連した機能およびサービス』

NetView プログラム自動操作機能

NetView プログラムは、自動操作の中心となるものです。このプログラムは、企業内にある他のプロダクトから情報を受け取って、指定した方法で情報の処理を行い、自動応答を出します。

自動操作においては、いくつかの NetView プログラム機能が重要な役割を演じます。同じことは、システムの自動操作においても、ネットワークまたは複数の企業の自動操作においてもあてはまります。これらの機能を使用して、NetView プログラムをカスタマイズすることによって、3 ページの『第 1 章 NetView 自動操作の紹介』で述べているような、各種タイプの自動操作が実行できるようになります。NetView プログラムでは、ユーザー独自の自動化アプリケーションを作成するために次の主要機能を提供しています。

- 『コマンド・リストおよびコマンド・プロセッサ』
- 28 ページの『タイマー・コマンド』
- 自動化タスク (28 ページの『自動タスク』)
- 29 ページの『自動化テーブル』
- 30 ページの『メッセージ改訂テーブル』
- 30 ページの『リソース・オブジェクト・データ・マネージャー』 (RODM)
- 自動操作に関しては 31 ページの『インストール・システム出口』
- 32 ページの『MVS コマンド改訂』
- 32 ページの『Automated Operations Network (AON)』
- 32 ページの『状況モニター』

コマンド・リストおよびコマンド・プロセッサ

NetView 製品を使用すれば、独自のプログラムを作成して、そのプログラムを、NetView コマンドと同じように使用することができます。これらのプログラムは、使用される言語によって分類されます。

コマンド・リスト とは、再構造化拡張実行プログラム (REXX) 言語か、NetView コマンド・リスト言語で書かれた、一連のコマンドと特殊な命令のことをいいます。

コマンド・プロセッサ とは、アセンブラー、PL/I、または C 言語で作成した、アセンブル済みモジュール、またはコンパイル済みモジュールです。コマンド・リストおよびコマンド・プロセッサは、自動操作において頻繁に使用されます。

コマンド・リストまたはコマンド・プロセッサは、オペレーターの作業の支援と、オペレーターの介入しないプロシージャーの実行のいずれも行うことができます。複数の NetView コマンドのタスクを実行するようなコマンド・リストを作成すると、オペレーターは、複雑なタスクをたった 1 つのコマンドで実行できるようになります。

また、NetView の自動操作機能を利用して、コマンド・リストまたはコマンド・プロセッサを開始させ、プロシージャーの実行におけるオペレーターの介入を省略することも可能です。コマンドを開始できるものとしては、例えば自動化テーブルやタイマー・コマンドなどがあります。

言語の選択

自動操作を計画するにあたって、まずコマンド・プロシージャーを作成するための言語を、1 つ以上選んでください。各言語の機能については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。

NetView 制御ブロックにアクセスできるのは、アセンブリー言語だけなので、制御ブロック情報の検討や修正をともなう自動操作を行うためには、アセンブリー言語を使用しなければなりません。ただし、他の自動操作ルーチンのほとんどは、アセンブラー以外の 4 つの言語で書いた方が容易です。他の 4 言語も、自動化への特殊値によって各種の機能を提供します。詳細については、『コマンド・プロシージャーによる自動操作』を参照してください。

コマンド・プロシージャーによる自動操作

コマンド・プロシージャー とは、コマンド・リスト、または PL/I ないし C 言語で作成されたコマンド・プロセッサのことです。このセクションでは、コマンド・プロシージャーで利用できる自動機能を要約します。

メッセージおよび管理サービス単位 (MSU) 情報の入手: 自動化テーブルは、コマンド・プロシージャーを呼び出すことによって、メッセージや MSU に応答することができます。この自動化テーブルは、コマンド・プロシージャーに渡すべきメッセージまたは MSU に関する情報を、パラメーターの形式で抽出することができます。例えば、自動化テーブルは、メッセージの MVS システム ID やジョブ名を獲得して、それを応答に使用するためにコマンド・プロシージャーに渡します。

また、コマンド・プロシージャー自体も、メッセージまたは MSU に関する情報を抽出することができます。自動化テーブルから出されたコマンド・プロシージャー(あるいは、管理サービス・トランスポートで MSU が受信されたために出されたコマンド・プロシージャー) は、それが出される原因となったメッセージまたは MSU の内容を獲得することができます。

グローバル変数の使用: 自動操作では、多くのコマンド・プロシージャーの間の協力と、自動化テーブルでの調整が必要なことがよくあります。グローバル変数は、あるコマンド・プロシージャーから、別のコマンド・プロシージャーおよび自動化テーブルに情報を伝送するために便利な方法です。

グローバル変数とは、コマンド・プロシージャーの使用中に値を保持する変数のことです。これは、1つのタスク上で実行されるコマンド・プロシージャー間での情報の共用(タスク・グローバル変数)、または複数の異なるタスク上で実行されるコマンド・プロシージャー間での情報の共用(共通グローバル変数)に、使用することができます。自動化テーブルも、グローバル変数の値を読み取ることができます。値を変更するには、自動化テーブルでコマンド・プロシージャーを呼び出さなければなりません。

NetView プログラムでは、外部データベースにグローバル変数を保管するためのオプションを提供しています。変数を保管しておけば、NetView プログラムを再始動させる際に復元することができるので、障害停止時のリカバリーに役立ちます。

パラメーターの受け入れ: コマンド・プロシージャーは、パラメーターも受け付けることができます。例えば端末からコマンド・プロシージャーを使用するときなど、オペレーターは、そのコマンド・プロシージャーの名前のあとにパラメーター情報を入力することができます。

その他のコマンド・プロシージャーや自動化テーブルなどの自動操作機能を使用しても、コマンド・プロシージャーの使用時にパラメーターを指定することが可能です。例えば、リカバリー・コマンド・リストを作成するときなど、パラメーター変数を使用して、再始動するアプリケーション・プログラムの名前、そのプロダクト用の開始コマンド、およびアプリケーション・プログラムの初期設定のための待ち時間などを指定することができます。

環境情報の入手: コマンド・プロシージャーでは、システムおよび操作環境に関する情報を入手することができます。例えば、コマンド・プロシージャーは次のようなデータを獲得することができます。

- 使用中のオペレーティング・システム
- ドメイン ID
- 現行の日付および時間
- そのプロシージャーを実行するタスクのタイプ

システムおよびネットワークとの対話: コマンド・プロシージャーは、オペレーティング・システムに、コマンドとメッセージを渡すことができます。コマンド・プロシージャーが、コマンドおよびメッセージをオペレーティング・システムに渡す方法については、33ページの『オペレーティング・システム自動操作機能とNetView プログラムとの対話』を参照してください。

コマンド・プロシージャーは、VTAM プログラムに対して、ネットワークを制御するためのコマンドを渡すこともできます。

待ち: コマンド・プロシージャーは、コマンドを出して情報を請求し、その応答を待ってから、さらにアクションを行うことができます。例えば、障害停止したアプ

リケーション・プログラムを再始動させるための自動操作プロシーチャーは、その後で照会コマンドを出して、アプリケーション・プログラムの終結アクションの完了確認を待つことが可能です。

タイマー・コマンド

タイマー・コマンドを使用して自動的なアクションを開始することができます。タイマー・コマンドは、オペレーターと自動操作プロシーチャーのいずれでも出すことができ、他のコマンドおよびコマンド・リスト、コマンド・プロセッサのスケジュールを行います。 NetView プログラムでは、以下のタイマー・コマンドを提供しています。

- AT コマンドは、指定の時刻に別のコマンドを実行するようスケジュールするものです。
- AFTER コマンドは、指定された時間が経過した時点で、コマンドを実行するようスケジュールするものです。
- EVERY コマンドは、あるコマンドが指定した間隔で繰り返し出されるようにスケジュールします。
- CHRON コマンドを使用すると、複合タイマー自動化機能を実行することができます。
- LIST TIMER コマンドおよび PURGE TIMER コマンドを使用すると、スケジュールしたコマンドを調べたり、取り消したりすることができます。
- TIMER コマンドを使用すると、フルスクリーン・パネルを使用して、タイマーの追加、変更、削除を行うことができます。

タイマー・コマンドの使用法については、127 ページの『第 11 章 タイマー・コマンド』、または NetView プログラム・オンライン・ヘルプを参照してください。

自動タスク

自動タスク とは、端末またはオペレーターを必要としない、オペレーター端末タスク (OST) のことをいいます。他の OST と同様、自動タスクはメッセージを受け取り、コマンドを出すことができます。自動タスクの制約は、フルスクリーン・アプリケーション・プログラムを実行できないという点だけです。一方、自動タスクは他の OST と異なり、VTAM プログラムがアクティブでなくても、実行することができます。この機能と、自動タスクがオペレーターの OST から行えるほとんどの作業を行えることにより、自動タスクは自動化に有用なものとなっています。

自動操作では、1 つ以上の自動タスクを定義して、NetView の初期設定中にそれらの自動タスクを開始させることができます。これによって、自動化テーブル、コマンド・リスト、コマンド・プロセッサ、およびタイマー・コマンドは、その自動タスクのもとでコマンドを出せるようになります。自動タスクは受け取ったメッセージを自動化テーブル、またはインストール・システム出口ルーチンに渡します。こうして、自動化のための他の多くの機能が自動タスクを利用できるようになります。

自動タスクは、広範囲に及ぶ自動化に適した優先タスクです。作業を自動タスクに経路指定しておく、オペレーターが OST を使用した場合に起こり得る問題を避けることができます。例えば、オペレーターがログオフしていたり、OST を他の作業で使用している場合などがこれに該当します。

自動化テーブル

NetView 自動化テーブルにより、着信メッセージおよび MSU の処理オプションを指定して、自動応答を出すことができます。テーブルに含まれている一連のステートメントは、さまざまな状況下で、NetView プログラムが取ることができるアクションを定義しています。

プログラムがどのような自動アクションを取ることができるかを判定するために、自動ステートメントは、MSU のあらゆるフィールド、およびメッセージ・テキストのあらゆる部分を検討することができます。(複数行のメッセージでは、先頭行より後の行を検査できるのは、ACQUIRE 条件のみです。) ステートメントはさらに、メッセージの ID、MSU のリソース階層、メッセージまたは MSU のドメイン ID、およびその他多くの属性 (オカレンスのしきい値など) を検討することができます。AND および OR のオペランドを認識できるので、任意の組み合わせでいくつかの比較を指定することができます。

着信メッセージまたは MSU がユーザーの条件に一致するときに NetView プログラムが実行するアクションは、いくつでも指定できます。アクションは、コマンド、コマンド・リスト、およびコマンド・プロセッサのどれであっても差し支えありません。単純な応答であれば、NetView コマンド、VTAM コマンド、またはシステムないしサブシステム・コマンドなど、単一のコマンドで十分です。複雑な応答に関しては、コマンド・リストやコマンド・プロセッサを書くことができます。自動化テーブルにそれらのアクションを行うタスクを指定すると、自動タスク下で自動操作プロシージャーを実行させることができます。

アクションには、メッセージおよび MSU の処理オプションの設定も含まれます。特別のメッセージについては、メッセージ処理オプションを使用して以下のことが指定できます。

- メッセージを抑止するかどうか (しない場合は、どのオペレーターに表示するか)。
- メッセージをオペレーターの表示画面に表示するかどうか (オペレーターに知らせる必要があるメッセージの場合)。
- 自動操作で、特定のアクション・メッセージに対する削除要求を処理するかどうか。
- メッセージをシステムまたはネットワーク、あるいはハードコピーのログに記録するかどうか。
- メッセージに対する注意を喚起するために音響アラームを使用するかどうか。

MSU 処理オプションは、ハードウェア・モニターに出される MSU に適用されます。これらのオプションを使用すると、レコード・フィルタをオーバーライドすることができます。特定の MSU については、MSU 処理オプションを使用して以下のことが指定できます。

- ハードウェア・モニターが MSU をイベント・データベースに記録するかどうか。
- ハードウェア・モニターがイベントをアラート・データベースに記録するかどうか。
- NetView プログラムが、アラートをフォーカル・ポイントに転送するかどうか。

さらに、カラー表示や下線表示などの強調表示オプションを指定して、オペレーターの注意を喚起することもできます。

システムの自動化テーブル使用報告書を作成するには、AUTOCNT コマンドを使用します。この報告書を利用して自動化テーブルのステートメントを分析し、これらのステートメントの一致している頻度を見ることができます。頻繁に一致しているステートメントをテーブルのいちばん上に移動させれば、不一致基準のチェック回数を少なくすることができます。

さらにユーザーは、論理エラーを起こす非適合ステートメントをテーブルから削除するか、あるいは変更するかを決めることができます。自動化テーブル使用報告書に目を通すことによって、自分のシステムで実現済みの自動操作のレベルを判断することができます。これらの統計は、管理目的の報告書として極めて有用です。

AUTOTEST コマンドを使用して、自動化テーブルをテストすることができます。現行メッセージと MSU を使用して、または事前記録メッセージと MSU を使用して、このテストを実行することができます。詳しくは『163 ページの『第 15 章 自動化テーブル』』を参照してください。

AUTOMAN コマンドを使用して、自動化テーブルを管理することができます。このコマンドを使用して、自動化テーブル・ステートメントを使用可能または使用不可にしたり、自動化テーブルをロードまたはアンロードしたり、これらの状況を表示することができます。詳しくは、287 ページの『複数の自動化テーブルの管理』を参照してください。

メッセージ改訂テーブル

メッセージ改訂テーブル (MRT) を使用すると、システム内を流れるメッセージを検査して、それらのメッセージの特定の部分を変更することができます。MRT は、NetView プログラムがアクティブでないときであっても、SSI アドレス・スペースさえアクティブであればアクティブになっています。この機能の使用法についての詳細は、139 ページの『第 13 章 メッセージ改訂テーブル』を参照してください。

リソース・オブジェクト・データ・マネージャー

NetView プログラムプログラムでは、リソース・オブジェクト・データ・マネージャー (RODM) を使用して、ネットワークおよびシステム・リソースに関するさまざまな情報を保管することができます。RODM はこの情報を高速ストレージに保管するので、検索および更新を速やかに行うことができます。自動操作では、メッセージや MSU、状況の変更に対する適切な応答を判断するのに役立てるため、RODM 内の情報を他の自動操作機能とともに用いることができます。

RODM は、メソッド・プロシージャー (または、メソッド) と呼ばれる小さなプログラムを使用して、RODM 内の情報の検索、更新、および操作を含む数多くの機能を実行します。また、RODM はアプリケーション・プログラム・インターフェース (API) を提供しており、アプリケーション・プログラムによる RODM 内の情報へのアクセスを可能にしています。NetView プログラムは、この API およびメソッド・プロシージャーを通して、RODM 内のリソース情報を必要に応じて検索および更新することができます。

自動操作における RODM の使用法については、87 ページの『第 8 章 リソース・オブジェクト・データ・マネージャーでの自動操作』を参照してください。

インストール・システム出口

NetView プログラムでは、独自のルーチンを書いて、特定の場所で処理を制御することができます。これらの場所は、インストール・システム出口 と呼ばれ、NetView プログラムの通常処理の変更を可能にします。自動操作に重要な意味を持つインストール・システム出口は、次のとおりです。

- DSIEX02A
- DSIEX16
- DSIEX16B
- DSIEX17
- XITCI

インストール・システム出口ルーチンをアセンブリー言語で書く方法については、「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」を参照してください。インストール・システム出口ルーチンを PL/I および C 言語で書く方法については、「*IBM Tivoli NetView for z/OS* プログラミング: PL/I および C」を参照してください。

DSIEX02A を使用する場合

DSIEX02A のルーチンを書く場合、そのルーチンはメッセージが自動化テーブルに送られる直前に制御を受け取ります。このルーチンは、メッセージの変更、置き換え、または削除を行うことができます。メッセージを変更または置換した場合、自動化テーブルには新しい方のメッセージが送られます。処理速度を早めたい場合は、このインストール・システム出口ルーチンをアセンブリー言語で書いてください。また、PL/I および C 言語を使用することもできます。

DSIEX16 または DSIEX16B を使用する場合

これらの出口は、メッセージ処理オプションの修正、メッセージの再形式設定、および MSU 情報の変更に使われます。これらのインストール・システム出口ルーチンは、いずれもアセンブリー言語で書かなければなりません。

DSIEX17 を使用する場合

DSIEX17 用にしたルーチンは、メッセージまたはオペレーター・メッセージ削除 (DOM) を MVS システムから受信した直後に、制御権を受け取ります。アセンブリー言語サービス DSIMMDB、または PL/I および C 言語サービス CNMPMDB のユーザー呼び出しから、メッセージまたは DOM を受け取ったときも、ルーチンは制御権を受け取ります。

DSIMMDB に関する詳細については、「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」を参照してください。CNMPMDB に関する詳細については、「*IBM Tivoli NetView for z/OS* プログラミング: PL/I および C」を参照してください。

作成したルーチンによって、メッセージまたは DOM を削除したり、テキストおよびメッセージの属性を修正することができます。このインストール・システム出口を使用するルーチンは、アセンブリ言語でしか作成できません。

ルーチンは、DOM が予期されていない MVS からのアクション・メッセージとして出されたメッセージにマークを付けるために使用することもできます。

XITCI を使用する場合

ハードウェア・モニターの出口 XITCI のルーチンを書いた場合、そのルーチンは、BNJDSERV タスクがデータを受け取ると制御を受け取ります。XITCI では、ハードウェア・モニターに入るどのようなデータでも変更することができます。XITCI 出口ルーチンは、PL/I、C、またはアセンブリ言語のいずれでも書くことができます。

MVS コマンド改訂

NetView MVS コマンド改訂機能を使用すると、MVS コマンドの検査、変更、または拒否を実行できます。詳しくは、149 ページの『第 14 章 コマンド改訂テーブル』を参照してください。

Automated Operations Network (AON)

NetView の自動化オペレーション・ネットワーク (AON) コンポーネントを使用して、VTAM SNA、および TCP/IP リソースに対して、ポリシー・ベースのネットワークの自動化ができます。

AON コンポーネントは、ネットワーク・リソースに問題があることを示すアラートおよびメッセージを代行受信します。AON は、障害の起きたリソースを回復したり、回復するまでリソースをモニターすることができます。AON は、リソースの障害の記録をとって、再発するネットワーク問題を追跡することができます。

AON は本書に記述されている機能の大半を使用して、ドロップインのポリシー・ベース自動操作を提供します。

詳しくは、499 ページの『第 31 章 自動化オペレーション・ネットワークの使用』を参照してください。

状況モニター

NetView の状況モニターを利用すれば、障害停止したネットワーク・ノードを自動的に再活動化することができます。MONON コマンドおよび MONOFF コマンドは、この形式の自動操作の開始および終了を行います。MONON および MONOFF は、端末から出すことも、自動化アプリケーション・プログラムに出させることもできます。状況モニターがどのリソースを自動操作するかを制御するには、VTAMLST データ・セット・メンバー内のステートメントを使用します。

ノードが状況を変更するときに、ユーザー独自の自動操作を行いたい場合は、DSICNM (CNMS5001) に SENDMSG ステートメントを追加することができます。これにより、ノードの状況変更によって CNM094I メッセージが生成されますが、

これは、自動化テーブルで処理することができます。SENDMSG の詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

オペレーティング・システム自動操作機能と NetView プログラムとの対話

システム自動操作において、オペレーティング・システムは、いくつかの自動操作機能を独自に提供する他、NetView プログラムとの対話を通して、さらに多くの自動操作機能を提供します。NetView プログラムは、オペレーティング・システムから情報を受け取り、その情報を NetView 自動操作機能で処理し、コマンドの形でオペレーティング・システムに応答を送ります。また、自動操作とは直接関係のない対話において、オペレーター・コマンドをオペレーティング・システムと NetView プログラムの間で送ることもあります。

MVS システムにおける自動化

NetView プログラムは、オペレーティング・システムおよびアプリケーション・プログラムからの、メッセージおよび MVS に対する応答を自動化することができます。オペレーティング・システムはまず自分自身の自動化タスクを実行してから、NetView プログラムにメッセージを送り、自動操作をさらに進めます。また、NetView コマンドをシステム・オペレーターから NetView プログラムに送信したり、MVS コマンドを NetView プログラムからオペレーティング・システムに送信することもできます。

直接 NetView プログラムに (サブシステム・インターフェース、または NetView プログラムの拡張複数コンソール・サポート・コンソールを通して) 送ることのできるシステム・メッセージには、オペレーター宛メッセージ (WTO) および要応答オペレーター宛メッセージ (WTOR) が含まれます。アプリケーション・プログラム (CICS や IMS プログラムなど) から、コンソールに出されたメッセージの中には、サブシステム・インターフェースや拡張複数コンソール・サポート (EMCS) コンソールを通して処理できないものもあります。そのようなメッセージに自動応答するためには、NetView プログラムの端末アクセス機能を利用します。

メッセージ応答の自動化

システム・メッセージを抑制または改訂する場合は、NetView メッセージ改訂テーブル (MRT) を使用します。メッセージへの応答を自動化するには、NetView メッセージ改訂テーブル内のメッセージに、NetView プログラムへの送信用のマークまたは「NetView のみ」(NETVONLY)のマークを付けます。

送信請求メッセージ (コマンド応答) は拡張 MCS コンソールを介して NetView プログラムに送られます。非送信請求メッセージはサブシステム・インターフェース (SSI) を介して NetView プログラムに送られます。NETVONLY としてマークされたメッセージは、常に SSI を介して NetView プログラムに送られます。

35 ページの図 5 に、サブシステム・インターフェースを使用した場合の、z/OS システムと NetView プログラム間のメッセージの流れを示します。36 ページの図 6 は、コマンドの流れを示しています。

35 ページの図 5 に示されているように、メッセージはまずメッセージ処理機能 (MPF) に流れます。この機能を使用すると、いくつかの処理オプションを設定でき

ます。次に、メッセージは複数コンソール・サポートを通ります。大部分のサブシステムにあてられたメッセージは、サブシステム・インターフェースを通してそれらのサブシステムにブロードキャストされます。

メッセージは、宛先が NetView プログラムでない場合も含め、メッセージ改訂テーブル (MRT) によって処理されます。MRT では、MPF で設定された仕様のオーバーライド、メッセージの抑止または変更、NetView プログラムへのメッセージの転送が可能です。他のサブシステム・インターフェース プログラムも、メッセージの検査と変更を行えます。

NetView プログラムは、受け取ったメッセージを 1 つ 1 つ自動化テーブルの項目と比較し、ユーザーが指定した自動応答があればその応答を出します。

注: CNMSTYLE メンバーの `endcmd.close.leeway` ステートメントを使用すると、CLOSE IMMED、CLOSE STOP、または MVS STOP (P) コマンドが NetView プログラムに入力された後の、コマンドを実行できる時間の長さを指定できます。`endcmd.close.leeway` ステートメントで定義された期間の間、メッセージ自動化はアクティブ状態を維持しますが、新規のコマンドがキューに入ることはありません。CLOSE STOP コマンドが発行されていて、メッセージ・キューイングが SSI に対して有効になっている場合、MVS メッセージ・キューイングは CLOSE STOP コマンドが認識されるとすぐに終了します。

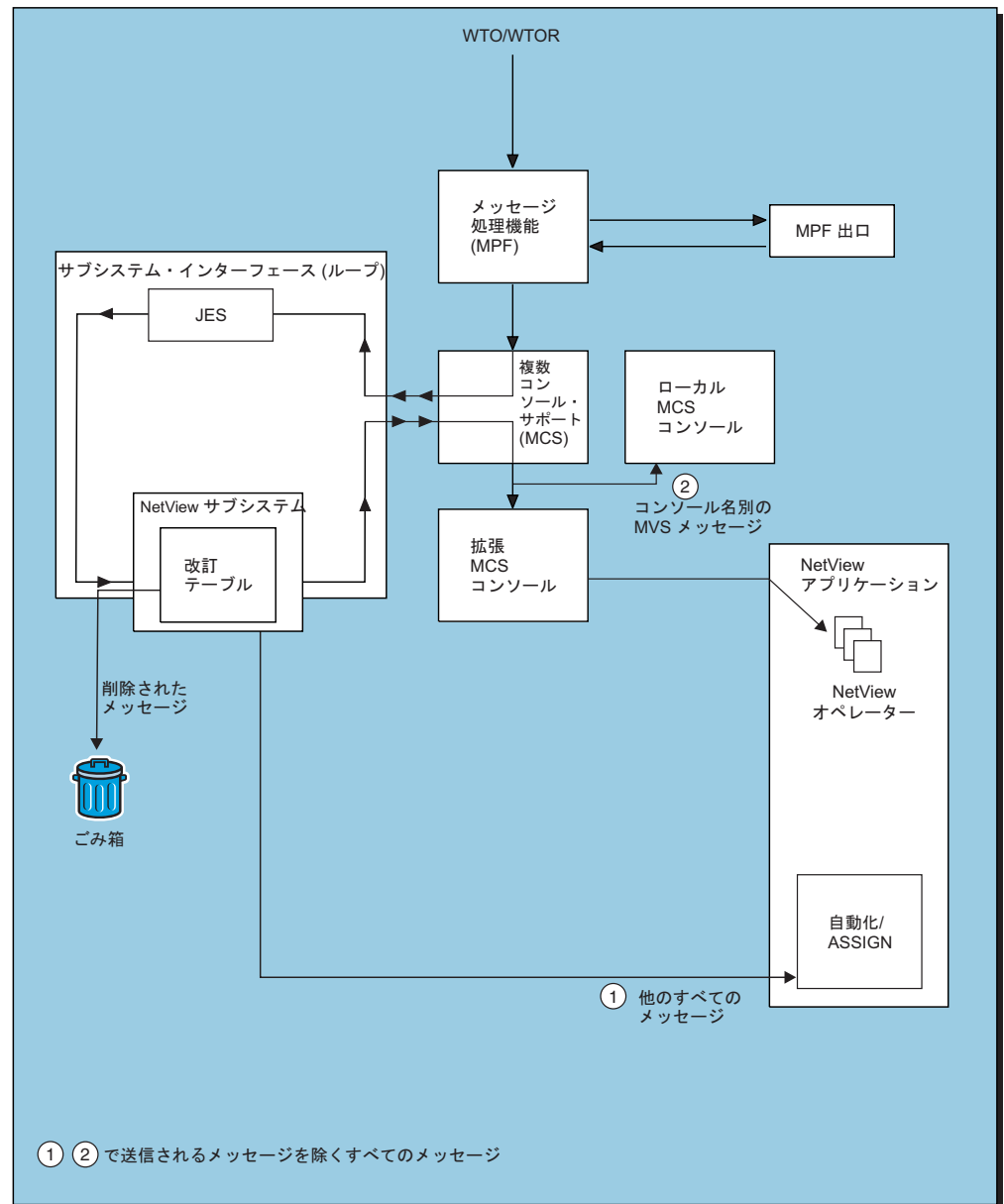


図 5. サブシステム・インターフェースを使用した z/OS と NetView プログラム間のメッセージの流れ

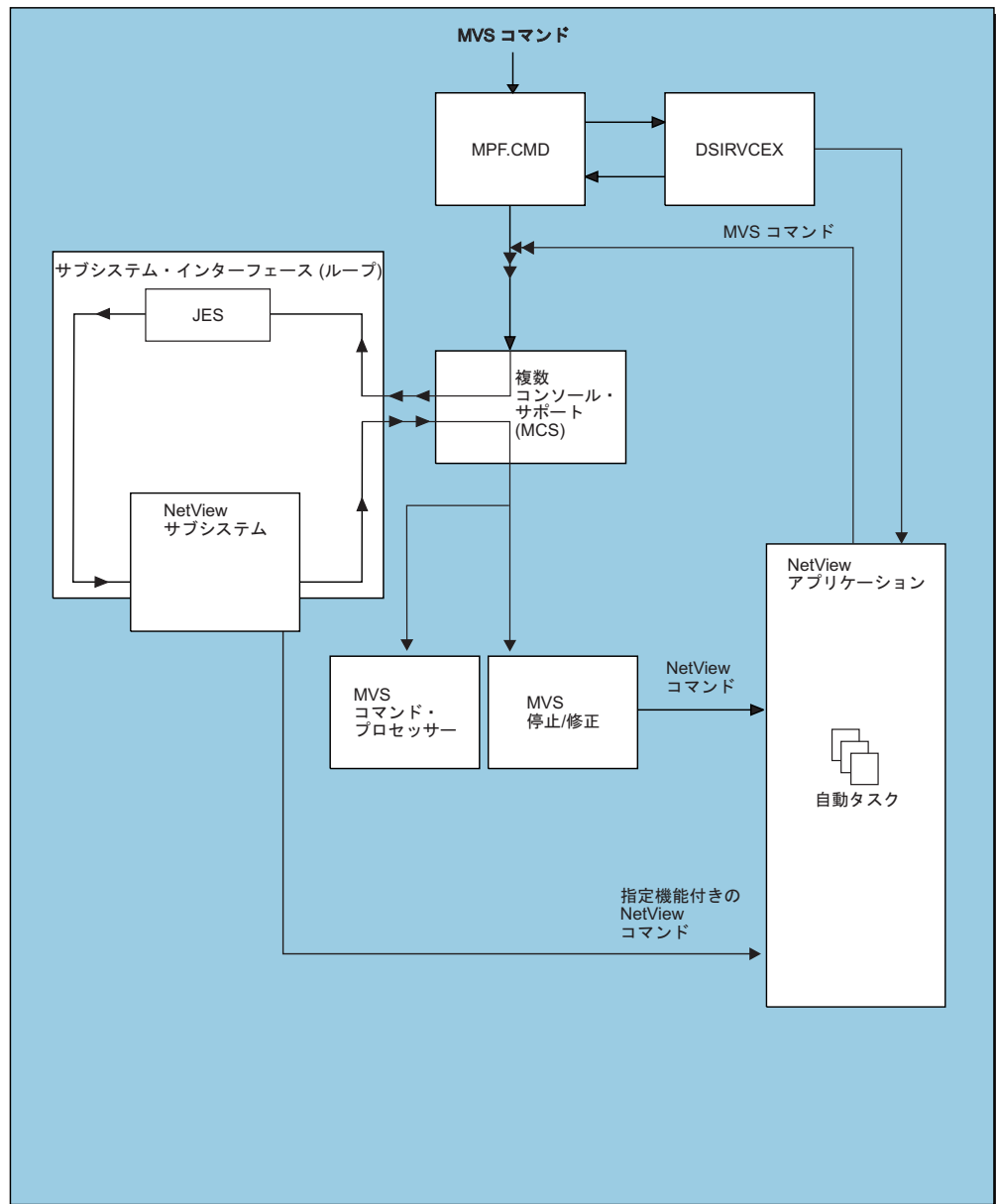


図 6. z/OS システムと NetView プログラム間のコマンドの流れ

メッセージ処理機能 (MPF) またはメッセージ改訂テーブル (MRT) を使用した自動化のためのオプションの設定

メッセージへの応答を自動化する場合に、MPF を使用してオプションを設定できます (例えば、特定のメッセージをオペレーターに対して表示するか、抑制するか、あるいは自動操作に適格であることを示すマークを付けるかどうかなど)。メッセージ改訂テーブルも使用でき、その場合は上記の機能に加えてさらに別の機能も提供できます。

デフォルトでは、NetView プログラムは、ユーザーが自動操作で利用可能とマーク付けしたそれぞれのメッセージを受け取って、それを自動化テーブルに通すようになっています。自動化したいメッセージのみに適格のマークを付けるようにすれば、処理時間を節約できます。

MPF またはメッセージ改訂テーブルを出たメッセージは、サブシステム・インターフェースを経由して NetView プログラムに流れて行くことができます。

MPF および NetView MRT を通過した後、メッセージを NetView プログラムへ受け渡すことができます。この状態は、以下のいずれかの事情で発生します。

- MRT が NETVONLY アクションを指定している。この場合、メッセージへ追加するシステム・アクションはありません。
- 自動化要求が MPF または MRT で出された。この場合、メッセージのコピーが NetView プログラムに送られます。

メッセージが EMCS コンソール経由で EMCS に流れる場合は、追加情報が入手可能です。この情報は、通常はシステム・メッセージに IFRAUVPT のところで付加されるメッセージ・データ・ブロック (MDB) に入っています。例えば、送信元の SYSPLEX 名がこの MDB に入っています。NetView プログラムは、メッセージの送信方法とは無関係に、MPF またはメッセージ改訂によってシステム・メッセージに設定されたあらゆる色情報を保持します。

拡張複数コンソール・サポート・コンソールの詳細については、73 ページの『第 6 章 MVS 拡張複数コンソール・サポート・コンソールを使用した自動操作』を参照してください。

シスプレックスの自動操作

単一の MVS システムの自動操作に加えて、NetView プログラムは、シスプレックス構成で相互接続された MVS システムについても、自動操作を提供することができます。MVS のシスプレックス 構成においては、複数の MVS システムが機能とプログラムを共用することによって単一のシステムとして動作します。

NetView プログラムがシスプレックス環境で操作される場合、NetView プログラムはコンソール名または経路コードによって送られてきたメッセージしか自動化しません。ローカル MVS イメージの外部からのメッセージの後処理については、142 ページの『DoForeignFrom ステートメント』を参照してください。

シスプレックスの自動操作に関する詳細については、83 ページの『第 7 章 MVS シスプレックスでの自動操作』を参照してください。

MSU への応答の自動化

別の MVS アプリケーション・プログラムからの MSU に対する応答を自動化するには、NetView とプログラム間のインターフェースと、管理サービス (MS) LU 6.2 トランSPORTを通して、それらの MSU を NetView プログラムに送信します。このプログラム間インターフェースは、ネットワーク管理ベクトル・トランSPORT (NMVT) と、制御点管理サービス単位 (CP-MSU) の両方を受け取ることができます。NetView 自動化テーブルは、両方のタイプの MSU に自動応答することができます。

MVS コンソールからの NetView コマンドの入力

MVS から NetView コマンドを出すには、3 種類の方法があります。

- MVS MODIFY コマンド
- NetView サブシステム指定文字
- コマンド改訂テーブル (CRT) から

詳細については、『MVS MODIFY コマンドを使用した NetView コマンドの発行』、『指定文字を使用した NetView コマンドの発行』、および 39 ページの『コマンド改訂テーブル (CRT) からの NetView コマンドの発行』を参照してください。

MVS MODIFY コマンドを使用した NetView コマンドの発行: 自動タスクをシステム・コンソールと関連付けてある場合には、そのコンソールから MVS MODIFY コマンドを使用して、NetView コマンドを入力することができます。次のように入力します。

```
f procname,command
```

ここで、*procname* は、CNMCNETV などのようにシステム・プログラマーが割り当てた、NetView プログラムのカatalog式プロシージャーの名前であり、*command* は、送出したい NetView コマンドです。例えば、NetView プログラムが使用している MVS コンソール名と ID を表示させるには、次のように入力します。

```
f procname,disconid
```

指定文字を使用した NetView コマンドの発行: システム・オペレーターが NetView プログラムへコマンドを出せるようにするには、複数コンソール・サポート・コンソールと NetView 自動タスクとの関連付けを行います。複数コンソール・サポート・コンソールと自動タスクの関連付けについては、NetView オンライン・ヘルプで AUTOTASK コマンドを参照してください。

35 ページの図 5 に示すように、複数コンソール・サポート・コンソールから出されたオペレーター・コマンドは、複数コンソール・サポートとサブシステム・インターフェースを通してサブシステムに流れます。サブシステムは、その割り当てられた文字が先頭に付いているコマンドしか処理しません。例えば JES2 は、先頭にドル (\$) 記号の付いたコマンドを処理します。

NetView プログラムは、先頭に指定文字ストリングのついたすべてのコマンドを処理します。NetView プログラムの CNMSTYLE 処理を使用して SSI を開始する場合は、この設定を行う際に MVSPARM.Cmd.Designator ステートメントを使用してください。これ以外の方法については、サンプル CNMSJ010 を参照してください。単一システム上で、複数の NetView プログラムを使用し、それらの NetView プログラムが、複数コンソール・サポート・コンソールで入力された NetView コマンドを処理する場合は、システム上のそれぞれの NetView プログラムに、別の指定文字ストリングを割り当ててください。このサンプルでは、サブシステム名を指定文字ストリングとして使用しています。デフォルトは、パーセント (%) 文字です。

NetView 自動タスクが、複数コンソール・サポート・コンソールと関連付けられており、NetView コマンドがそのコンソールから出される場合、そのコマンドは、そのコンソールと関連付けられている NetView 自動タスクによって呼び出されます。複数コンソール・サポート・コンソールからは、NetView コマンド・プロシージャーおよびコマンドの呼び出しも可能です。

コマンド改訂テーブル (CRT) からの NetView コマンドの発行: システム・オペレーターがコマンド改訂テーブルを使用して NetView プログラムへコマンドを出せるようにするには、送信されるコマンドを CRT が認識し、そのコマンドの NETVONLY アクションを呼び出すように準備します。呼び出される REXX プロシージャでは、検証のために SAF ユーザー名 (およびグループ) をチェックしたり、必要なアクションがあれば実行したりすることができます。例えば、LINKNV コマンドを特定して NETVONLY アクションを使用することを想定すると、それは AUTOTASK コマンドを起動する REXX プロシージャを呼び出して、LINKNV コマンドの発行元である特定のコンソールに関して、MODIFY コマンドまたは指定文字を使用できるようにします。

NetView プログラムからの MVS コマンドの発行

個々の MVS コマンドの前に NetView コマンド MVS を置くことによって、NetView プログラムから MVS システムに、MVS コマンドを出すことができます。NetView オペレーターまたは自動タスクのいずれも、NetView MVS コマンドを出すことができます。

MVS コマンドの前に NetView コマンド MVS を置く以外に、個々のコマンド verb に対してコマンド定義を定義することも可能です。MVS にコマンド定義ステートメントを定義する方法については、CNMS6401 サンプルを参照してください。

権限の与えられていない MVS コマンドの使用に対する防御策として、NetView プログラムのコマンド許可機能を利用することができます。また、IBM リソース・アクセス管理機能 (RACF®) の OPERCMDS クラス、または互換性のあるセキュリティ・プロダクトを利用して、システム・コマンドを保護することができます。システム・コマンド・セキュリティに関する詳細は、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

MVS コマンドの自動化

MVS コンソール、あるいはコンソール・インターフェースから入力する MVS コマンドおよびサブシステム・コマンドは、自動化することができます。そのためには、MVS コマンド出口としてロード・モジュールをインストールし、MPFLSTxx メンバーのいずれかに .CMD ステートメントを追加して、SET MPF=xx コマンドを出し、その出口を活動化しなければなりません。詳しくは、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。

MVS システム・メッセージおよび削除オペレーター・メッセージ (DOM) の発行

NetView WTO および WTOR コマンドを使用すると、MVS システム・メッセージを出すことができます。また、NetView DOM コマンドを使用すれば、MVS DOM を出すことができます。

DOM、WTO、および WTOR コマンドに関する詳細は、「*IBM Tivoli NetView for z/OS*プログラミング: REXXおよびNetViewコマンド・リスト言語」を参照してください。

IBM Tivoli System Automation for z/OS

ユーザーの設計に、システム自動化支援プログラム (z/OS) を組み込むことによって、自動処理の速度を向上させることができます。システム自動化支援プログラム (z/OS) のライセンス・プログラムは、NetView プログラムに基づくアプリケーションです。これは、単一の制御ポイントでシステム管理機能の完全な制御ができるように設計されています。

NetView インターフェースの使用例

NetView プログラムを使用すれば、NetView プログラムにメッセージまたは MSU を送り、NetView プログラムからコマンドを受け取るあらゆるプロダクトについて、その管理を自動化することができます。これらのプロダクトの中には、NetView プログラムへのインターフェースなど、相互接続プロダクトが必要になるものもあります。

相互接続プロダクトを使用すれば、非 SNA ネットワークおよび装置の管理も可能です。これらの非 SNA ネットワークおよび装置の大半は、NetView プログラムへのインターフェースとして、NetView サービス・ポイントを使用します。

このセクションでは、2、3 の例を挙げながら、インターフェース・プログラムとして使用できる相互接続プロダクトについて説明します。

分散ネットワーク

イベント自動化サービスがある NetView プログラムを使用して、分散ネットワークで発生するイベントの処理を自動化できます。イベント自動化サービスは、NetView プログラムおよび分散ネットワークのいずれの環境下においても発生するネットワーク・イベントに対して、両者間のゲートウェイを提供します。イベント自動化サービスは、NetView サブシステム PPI インターフェースを使用して NetView プログラムと、また TCP/IP プロトコルを使用して イベント・サーバーと通信します。

イベント自動化サービスは、NetView のアラートまたはメッセージを Event Integration Facility (EIF) イベントに変換して転送することができます。また、これらのイベントを、NetView アラートに変換して、転送することができます。その後、これらのアラートは、自動応答を開始するための自動操作で使うことができます。

SNMP を使用する IP ネットワーク

イベント自動化サービスは、NetView プログラム、SNMP エージェント、および SNMP マネージャー相互間でイベント・データを管理することができます。NetView アラートは、SNMP マネージャーに転送される前に SNMP トラップに変換されます。SNMP エージェントから到着するトラップは、SNA アラートに変換でき、その後、NetView プログラムハードウェア・モニターに転送できます。そこで、これらのアラートはフィルターに掛けられ、NetView 自動化テーブルに経路指定されます。

SNMP トラップに関する詳細は、457 ページの『イベント自動化サービス』を参照してください。

IBM 以外のネットワーク

NetView プログラムは、その他のタイプのネットワークを管理することもできます (例えば、DECnet)。

自動操作に関連した機能およびサービス

本書では、主にシステム・コンソールとネットワーク・コンソールの自動化の観点から、自動操作について説明します。本書では、自動操作機能を使用してメッセージと MSU に応答するため、およびシステムとネットワーク・ソフトウェアのコンソールでコマンドを出すために、オペレーターのアクションを支援あるいは代行させる方法についても説明します。

システムおよびネットワークでは、自動操作と密接な関係にあるその他の機能やサービスについても利用が可能です。詳細については、自動操作に関連する次のトピックを調べてください。

- 『ワークロードの管理』
- 『ネットワーク・パフォーマンスの管理』
- 42 ページの『入出力の管理』
- 43 ページの『ストレージの管理』
- 43 ページの『管理報告』

ワークロードの管理

実動バッチ・ジョブの管理を自動操作すると、可用性と制御能力が改善され、オペレーターの介入を減らすことができます。

IBM では、ワークロード管理のために、運用計画管理/エンタープライズ・システム体系 (OPC/ESA) ライセンス・プログラムを提供しています。OPC/ESA プログラムを使用することによって、MVS バッチ実動ワークロードの計画、制御、および自動操作が可能になります。このプログラムは、ワークロード処理の計画およびスケジュールを行い、ローカルからリモートにまたがるユーザーのデータ処理環境全体について、作業の流れをモニターおよび制御します。これによって手操作介入の必要が減り、手操作による制御をもっと重要な処理や決定に振り当てることが可能になります。

OPC/ESA プログラムを使用してワークロードを管理することにより、NetView の自動操作はさらに充実します。OPC/ESA プログラムはジョブのスケジュールを行います。スケジュールされたジョブの障害で、オペレーターによるアクションが必要な場合、NetView プログラムによる自動操作によって、そのアクションを行うことができます。

OPC/ESA プログラムについての詳細は、「*Operations/Planning and Control/Enterprise Systems Architecture General Information*」を参照してください。

ネットワーク・パフォーマンスの管理

NetView パフォーマンス・モニター・プログラム (NPM) を使用すると、自動化アプリケーション・プログラムで適用できるパフォーマンス・データの範囲がさらに広がります。また、NPM プログラムは集中操作にとっても有効です。なぜならこの

プログラムは、中央システムと分散システムの対話の速度をモニターする際に役立ちます。中央システムで NPM プログラムを使用することによって、オペレーターは他のシステムで収集したデータを見ることができます。

NPM は、NetView プログラム間インターフェース、およびその他のいくつかのインターフェースを通して、NetView プログラムと通信します。オペレーティング・システムを通して NPM にコマンドを出せば、自動操作ルーチンは特定のネットワーク・リソースに関するデータを要求することができます。例えば、SNA、ローカル・エリア、および X.25 ネットワーク内の通信制御装置、回線、論理装置 (LU)、および物理装置 (PU) に関するデータを要求することができます。

NPM では、非送信請求データについても送信が可能です。例えば、重大なネットワーク・リソースに対するパフォーマンスがユーザー定義のしきい値以下に落ちた場合、NPM プログラムは、NetView プログラムにアラートを送信することができます。このアラートを利用すれば、パフォーマンス上の問題がユーザーに影響を与える前に、問題の発生を自動操作ルーチンに通知することができます。さらにオプションで、問題が解決した時点で NetView プログラムに解決を送信して、自動操作ルーチンにその旨を通知することも可能です。

入出力の管理

入出力 (I/O) 管理には、データ処理複合体への、またはデータ処理複合体からの、データの流れの制御が含まれます。自動化環境において、テープやプリンターの管理など、それまで手操作介入が要求された入出力活動に対するアプローチの変更を思い立ったとします。

テープ管理に対するアプローチの 1 つとして、テープを直接アクセス記憶装置 (DASD) に切り換えることによって、手操作介入を避ける方法があります。DASD なら、テープのように手操作介入を要求することはありません。DASD のコストとテープ処理にかかわる人件費を含めたテープのコストを、ケースごとに比較してください。DASD には高い信頼性と管理能力があります。さらに、入出力に関するユーザーの規則と方針を定期的に確認することが可能です。この場合は、その方針の有効性と、アプリケーションのプログラマーがそれらを忠実に適用しているかどうかを判別してください。

一方、使用回数の少ないデータについては、テープ装置に頼った方がよい場合があります。IBM 3480 磁気テープ・サブシステムは、サイズおよび信頼性の面で他のテープ装置よりも優れていることから、幅広く利用されています。カートリッジ自動装てん機構が用意されていて、これを使用すると、カートリッジの処理を助けて、人手の介入と所要時間を最小限に抑えて、スクラッチ・テープを取り付けることができます。

またプリンターについては、装置の扱いの責任をユーザーの手にゆだねた方がよい場合があります。例えば、印刷の量を減らしたい場合には、IBM の報告書管理および配布システム (RMDS) プログラムが役に立ちます。RMDS プログラムを使用すると、ユーザーに対し、中央ライブラリー・アーカイブからのデータをオンラインで提供することができます。ユーザーは、データをオンラインで表示して、書類にしておきたい一部の情報だけを印刷することができます。RMDS プログラムを使用

すれば、大量の報告書データを定期的に印刷してユーザーに配布する必要がなくなります。たいていの場合、ユーザーが必要としているのはそれらの報告書の一部にすぎないのです。

RMDS プログラムの詳細については、「*Report Management and Distribution System: General Information*」を参照してください。

ストレージの管理

ストレージ管理には、テープや DASD のような、補助記憶装置に保管するデータの保全性と可用性の維持が関係します。ユーザーはこれまで、各自のデータ・セットが存在する可能性のある記憶装置プール内の 1 つ 1 つの装置について、その特性を把握していなければなりません。

記憶管理サブシステム (SMS) を採用し、MVS データ機能記憶管理サブシステム (DFSMS) ファミリー・プロダクトを使用すれば、DASD 記憶管理をユーザーではなく記憶管理責任者の手にゆだねることができます。ストレージの管理者は、ストレージ・クラスと管理クラスの形式で管理方針ステートメントを設定して、これらのクラスに基づいてストレージが割り振られる方法を定義、および管理します。ユーザーはこれらの方針ステートメントによってストレージを割り振るだけでよく、UNIT や VOLSER などの装置特性および構成特性を使用する必要はありません。

SMS を使用すれば、スペース不足によるプログラムの異常終了の回数も減り、実動ジョブ・ストリームへの支障も軽減されます。これは、ジョブが構成の詳細に左右される必然性はないからです。ストレージ管理を OPC/ESA プログラムなどのワークロード管理プロダクトとともに使用すれば、自動操作ジョブのリカバリー機能が得られます。この結果、実動ストリームは安定して実行され、人手の介入を最小限に抑えて、スケジュールされた枠内で終了します。

DFSMS ファミリー・プロダクトの詳細については、「*MVS Storage Management library*」を参照してください。

管理報告

自動化環境に移行するにあたっては、自動操作設計の中に強力な管理報告システムを含めてください。自動化は、さらに多くの操作を処理するため、管理者の注意を必要とする事項、またはリソースに関する変更を必要とする事項を識別する必要があります。ログから情報を入手し、それを管理者に示すために要約するために、ソフトウェア情報/管理およびサービス水準報告プログラム (SLR) 製品を使用することができます。

ソフトウェア情報/管理および SLR 製品について詳しくは、「*Introducing the Information/Family for MVS*」および「*Service Level Reporter General Information*」を参照してください。

第 2 部 自動化環境の作成

第 3 章 自動操作プロジェクトの定義

本章では、自動操作プロジェクトのプロジェクト定義タスクおよびフェーズについて説明します。このフェーズではまず計画チームを編成して、自動操作によって操作がどのように改善されるかを調査し、プロジェクトの目標（ゴール）と目的を設定します。

プロジェクト定義タスク

プロジェクト定義フェーズでは、次の点に焦点を当てます。

- 48 ページの『自動操作チームの編成』またはチーム
- 49 ページの『プロジェクト計画の作成』
- 49 ページの『組織目標の明確化』
- 50 ページの『操作環境について』
- 53 ページの『自動操作の目標と目的の設定』
- 56 ページの『理解と協力』

自動操作は、ほとんどの場合、多くの独立した部門とグループを調整するという、企業全体に及ぶ統合された努力の形で、最大の効果をあげることができます。自動化によって、組織と作業の関係に次のような変化がもたらされる可能性が考えられます。

- 運用組織が再編される
- オペレーターの役割が変わる
- オペレーター、技術サポート要員、およびシステム・プログラマー間の作業関係が変わる

自動操作を行う際には、相当な調整や広範囲に及ぶ変更が必要になることが考えられるので、上級管理者を含む組織全体の理解を得ておくことが大切です。企業の経営管理部門は、自動操作の目標を達成するために必要なリソースを提供しなければなりません。

アプローチを統合することによって、作業の重複を避けることができます。グループや営業所がそれぞればらばらに自動操作の方法を試みて、小規模で相互に関連のないプロジェクトを採用すると、時間が無駄に費やされたり、方法が不適切なものになったり、または各自動化アプリケーション・プログラムを組み合わせで実行させられないという結果を招いたりする恐れがあります。

実施フェーズ中であれば、1 回に作成する自動操作を細分化することができます。自動操作のプロセスを全体的に見るには、これがよいタイミングでもあります。最初から企業規模のアプローチを確立しておく、あとでプロジェクトを再設計する必要の生じる恐れがなくなります。

プロジェクトの最初の部分では、以下の例のように、目標を明確にすることが重要です。

- 自動操作が事業目的にどれだけ役に立つか
- 自動操作によってデータ処理操作をどれだけ改善できるか

自動操作による利益を明確にするとともに、そのコストを見積もることもできます。そうすることにより、投資に対して最大の効果を得られる自動操作のタイプを決定することができます。

自動操作チームの編成

最初のステップは、自動操作の分析および実施を担当する 1 つ以上のチームを編成することです。組織の中でだれが自動操作を担当しているのかが、すでにわかっている場合もあるでしょう。そうでない場合は次の質問を行ってください。

- 組織内で、自動操作の計画に対する責任を持つのはどの部門であるか
- 自動操作チームで必要とする技術はどういったものか、またそれらの技術を提供できるのはだれか

アプローチの選択

使用可能な人的リソースと、必要なスケジュールに応じて、いくつかの方法のうちどれかを使用することができます。まず、プロジェクト専任のプロジェクト・リーダーを任命して、必要に応じて、他の組織のサポートを要求するという方法があります。また、人的リソースが豊富な場合は、一時的なプロジェクト・チームを組むという方法もあります。この場合は、複数の人員が自動化のために、フルタイムで作業することになります。

最後に、これはもっと継続的なものですが、永続的な自動化部門を作成するという方法があります。また、プロジェクトの各段階またはフェーズに対して、個別にチームを編成する必要があるかどうかも考慮してください。組織の多くでは、まず一時的な計画チームからスタートしますが、自動操作の開発が進むに従って、永続的な部門を設定しています。

自動操作計画チームには、自動化によって影響を受けるすべての組織からのメンバーを組み込むとよいでしょう。例えば次のようなメンバーが考えられます。

- 1 人以上のオペレーター
- システム管理の技術サポート・スタッフ 1 人
- ネットワーク管理に携わるスタッフをもう 1 人 (該当する場合)
- 主要なサブシステムおよびアプリケーション・プログラムをサポートする、サブシステム・プログラマー (複数)
- ネットワーク・ユーザーの代表 (複数)

また、Tivoli の営業所からのシステム・エンジニアを、計画チームに迎えることも可能です。営業所のシステム・エンジニアは、自動操作プロダクトに関する情報や、自動化の計画や実施に成功した他の顧客の経験に関する情報を提供することができます。

運用グループの参加

自動操作を成功させるためには、プロジェクトの定義から設計、実施、テスト、実稼働に至る自動操作の各フェーズにおいて、運用グループに参加してもらってくだ

さい。運用グループのメンバーであれば、現行の環境を理解しており、自動操作に適したプロシーチャーを判断することができるはずです。同時に、自動操作を実際に利用する人々でもあります。したがって、この人たちに各自動操作プロシーチャーの設計に携わってもらいと、手順を確実に要求に応じたものにすることができます。

例えば自動化されていない環境では、システムおよびネットワーク・オペレーターは通常、絶え間ないメッセージの流れによって、物事が順調に進んでいるか、あるいは予期されるイベントが予想どおりに起こりつつあるかを判断しなければなりません。特定のプロシーチャー（システムの初期設定など）の自動操作を行って、突然何のメッセージも表示されなくなったら、オペレーターは、物事が予想どおりに進んでいるのかどうかを判断するのが難しくなるでしょう。運用グループの参画が、適切なオペレーター・インターフェースを確保するのに役立ちます。

プロジェクト計画の作成

自動操作プロジェクトを管理するためには、プロジェクト計画によって、各フェーズに必要なステップをリストし、各ステップの担当責任者または担当責任グループを明確にし、各ステップの完了目標日を割り当てます。プロジェクト計画は、プロジェクトを管理し、スケジュールどおりに運行するための手段になります。

プロジェクト計画は、時間とともに発展することが考えられます。完全な詳細を記入することができない場合でも、現時点で想定できる作業、担当グループ、目標日を設定することにより、計画を開始することができます。プロジェクトの発展にともなって、詳細を埋めていくとよいでしょう。

プロジェクトの全フェーズ（定義、設計、実施、および実動）を代表するタスクを識別するための計画については、579 ページの『付録 B. サンプル・プロジェクト計画』を参照してください。もちろん、この計画は例にすぎません。各自のプロジェクトは、かなり違っている場合もあるでしょう。

組織目標の明確化

計画チームのもう 1 つの作業に、自動操作の目標の明確化があります。明確な目標を持つことにより、プロジェクトに焦点を合わせて、結果を評価することができます。また、事業提案書や自動操作プロジェクトの計画書といった、計画書の作成が容易になります。

事業目標の明確化

企業や組織には、いくつかの事業目標があるはずです。例えば、次のような目標です。

- ・ 向こう 2 年間で総業務量を 40 % 増大させる
- ・ 向こう 5 年間、純利益を年間 10 % ずつ増加させる
- ・ コストを抑えて、生産性を向上させることにより、翌年の利益幅を 5 % 増加させる

組織の中のさまざまな分野が、異なる事業目標をもっていることがあり、そのような事業目標を考慮する必要があります。組織の目標を明確に理解することにより、自動化がその目標の達成にどのように貢献できるかを判断することができます。

データ処理の要件の識別

全体的に事業目標をサポートするためには、データ処理部門に独自の要件があるのが普通です。これらの要件には、次のような目的があります。

- 向こう 2 年間でシステムの可用性を 10 % 増大させる
- 向こう 2 年間に、オペレーターを増やすことなく、容量 (1 秒あたり百万単位の命令、つまり MIPS) とネットワーク・リソースの 12 % 成長に対応する
- 向こう 5 年間にわたって、システム・パフォーマンスを年間 15 % ずつ向上させる

データ処理の要件は、通常、システム指向とユーザー指向という 2 つのクラスに分類されます。システム指向の要件では、システムの処理する情報の量を測定します。これらの要件には、次のものが含まれます。

- 予想されるバッチ・スループット
- 各システムにおけるワークロード
- 対話式トランザクションの率
- サポートできるログオン・ユーザーの数

一方、ユーザー指向の要件では、データ処理サービスがユーザーに与えるインパクトを測定します。例えば、対話式作業での推定応答時間や、バッチ作業のターンアラウンド・タイムがあります。

サービス・レベル・アグリーメントは、パフォーマンスに対するこれらの予想を反映するものです。サービス・レベル・アグリーメントは、データ処理部門とその部門のユーザーとの間にかわされる契約と同様のものです。サービス・レベル・アグリーメントでは、提供されるサービス、そのサービス時間、および可用性とパフォーマンスの測定方法に関する同意が規定されます。他の要件では希望の目標を示すことが多いのに対して、サービス・レベル・アグリーメントは、達成する必要がある最小限の条件を指定します。

組織でサービス・レベル・アグリーメントを使用していない場合、またはサービス・レベル・アグリーメントが各自の目標を正確に反映していない場合は、目標に基づいた同意事項の設定を考慮してください。サービス・レベル・アグリーメントがあれば、自動操作によってもたらされたユーザー・サービスの向上を評価する際に役に立ちます。さらにこのアグリーメントは、自動操作の恩恵を受けることになる仕事の問題分野を識別する上でも有用です。

操作環境について

次のような質問が考えられます。

- オペレーターが時間をどのように使っているか
- 生産性を高めるために、どのルーチンと反復作業を自動操作できるか
- スケジュールされていないイベントのうち、オペレーターによるアクションを必要としているものはどれか

- ・オペレーターのアクションを必要とするスケジュールされていないイベントのそれぞれについて、アクションの遅れ、あるいは誤ったアクションがどの程度深刻な結果をもたらすか
- ・こういった種類のイベントが業務に深刻な影響を及ぼすか

要約すれば、現行の運用環境における最も重要な問題および課題は何か、また自動操作により最大の恩恵を受ける部分は何か、という 2 点にまとめることができるでしょう。

現在の環境について調査を行ったら、次に、今後の問題を検討することができます。

- ・来年または今後数年間にわたって、環境にどのような変化が予想されるか
- ・システムまたはネットワークにハードウェアを追加する予定があるか
- ・管理する必要がある新しいアプリケーションを追加する予定があるか
- ・サービスに依存するユーザーの数が増えるかどうか
- ・オペレーターを増やすことなく成長に対応しなければならないという制約があるかどうか
- ・データ・センターを追加するかどうか

こういった要因は、自動操作の戦略と目標に影響を与えます。現在の環境と目的をよく理解することによって、自動操作を最大活用するための包括的な戦略を考案することができます。

自動操作のために操作要件を識別する処理は、操作スタッフとの共同作業から始めてください。オペレーターは、定期的に行っている手順、つまり、スケジュールどおりに実行しているものと、予測可能な応答や反復作業に関するものを識別することができます。こういった情報によって、自動操作する手順を選択して、優先順位付けを行うことができます。

NetView プログラムのメッセージ改訂テーブルは、システム・メッセージのトラフィックの分析に役立てることができます。MSID、PREFIX、または JOBNAME についての UPON ステートメントを組み込むと、他のステートメントを指定していてもいなくても、改訂報告書にはそれぞれの条件ごとに、その条件を満たすメッセージの数が表示されます。MRT についての詳細は、30 ページの『メッセージ改訂テーブル』を参照してください。

MVS システム・ログとネットワーク・ログ

MVS システム・ログとネットワーク・ログを分析することによって、オペレーターが毎日目にするメッセージの数に関する情報が得られます。この情報は、メッセージ抑止による効果を評価するのに役立ちます。ほとんどの MVS システムでは、メッセージ抑止によって、著しい効果が得られます。

単純なアプリケーション・プログラムを作成して、ログの処理に役立てることができます。ログの処理に使用するアプリケーション・プログラムの例については、651 ページの『付録 I. 自動操作のサンプル・セット』を参照してください。サンプル・プログラムでは、SYSLOG (JES2 ログ) または DLOG (JES3 ログ) を分析しています。他のログについては、サンプル・プログラムを変更するか、あるいは独自のプログラムを作成することができます。

このサンプル分析プログラムでは、それぞれのプログラムでできるいくつかのことを説明しています。

- 受け取られた固有のメッセージ ID と、その ID を持つメッセージが発生した回数を記録すること。
- 受け取られた固有のメッセージ ID のリスト (発生頻度順に分類) を提供すること。
- 分析の時間制限、および無視できるメッセージなど指定する入力を受け入れること。

分類されたメッセージ ID のリストは、どのメッセージに関して集中的に対処したらいいかを示しています。各メッセージ ID については、それが、メッセージの総量に占めているパーセンテージを算出してください。通常、少数のメッセージ ID が、メッセージ・トラフィックのほとんどを占めています。わずか 10 個のメッセージ ID がトラフィックの 90 % を占めるという事態も十分に起こり得ます。したがって、ほんの 2、3 のメッセージ ID を抑止、または自動操作するだけで、大幅な節約になります。

これらのログには、オペレーターの入力したコマンドが入っているため、業務上の問題を識別する際にも役立ちます。例えば、ジョブ・キューの開始と停止、ジョブ・クラスの変更、およびジョブ優先順位のリセットのための JES コマンドを、オペレーターが大量に入力していたとします。これはすなわち、オペレーターが作業の流れを制御するために、多くの時間を費やしていることを意味します。したがって、OPC/ESA などのジョブ・スケジューリング・プログラムを導入することもよいでしょう。あるいは、出されたコマンドの大部分が、CICS 端末装置の損失といった、頻繁に発生する状態に対する応答である場合があります。このように、各コマンドの発行頻度に注意することによって、自動操作によって最大の効果を得られる手順を識別することができます。

運用プロシージャ・ブック

運用プロシージャ・ブックまたはラン・ブックは、自動化のための良い情報源です。要件を識別してどのプロシージャを自動化するかを決定したら、オペレーター・プロシージャ・ブックを参照して、どのようにして自動化がそれらのプロシージャを実行できるかを、ステップごとに進めていくことができます。

問題管理報告書

問題管理報告書は、ハードウェアおよびソフトウェアの問題を追跡して、それぞれの問題を解決するために取られたアクションの概要を説明するものです。これらは、頻繁に繰り返し発生してリソースを消費する問題を識別するのに役立つだけでなく、こういった問題に応答するための手順を識別するのに役立ちます。

問題がすぐに検出されなかったり、問題を訂正するために必要なリソースが使用不可であったために長引いた障害停止を探してください。さらに、自動処理であれば、その問題を検出して該当する人々にそれをより迅速に通知することができたかどうか、またはその問題を解決できたかどうかを判断してください。

ヘルプ・デスク・ログ

ヘルプ・デスク・ログは、問題記述のためのもう 1 つの情報源となるものです。問題管理ログと同様、ヘルプ・デスク・ログも、繰り返し発生する状態、および設定済みの手順が不適切である状態を識別するのに役立ちます。

サービス・レベル・アグリーメント

サービス・レベル・アグリーメントおよび同意確認の測定方法を検討することによって、ユーザーへの約束履行が困難な分野を判別することができます。これらの領域は、問題を明確に示すものであり、自動操作は、これらの問題を解決する方法となり得ます。

ユーザー

オペレーターに未報告の問題、あるいは問題管理で追跡されなかった問題を含めて、現行環境において起こり得る問題をユーザーが教えてくれる場合があります。

その他のデータ処理計画

計画期間全体を通して、各自の環境で予想される変更について検討してください。これにはまず、現行のシステムとネットワークの構成を、ハードウェアおよびソフトウェアの両方について文書化してから、今後の計画を検討してください。また、中央のフォーカル・ポイント・システムからの操作を計画するものも含め、自動操作を計画するすべてのデータ・センターの構成を文書化してください。自動操作の計画は、予想される変更を反映したものでなければなりません。

例えば、組織で新しく大規模なシステムを追加する場合は、メッセージの抑止とコンソールの統合が主要な要件となります。データ・センターを追加するか、あるいは分散ネットワークに移行する場合は、ネットワークおよび複数システムの自動操作が主要な要件となります。また、徐々に増加するユーザーをサポートする場合には、オペレーターを増やさずに、システムおよびネットワークにハードウェアとソフトウェアを追加することが、第一の要件となります。

情報の解釈

これらの情報源を検討することによって、次の点が明らかになるはずです。

- オペレーターが、どのように時間を費やしているか
- メッセージ抑止の利点
- 標準化および文書化したいプロシージャは、どれか
- 自動操作を行う上で最も効果的なプロシージャは、なにか
- ユーザーが、どのような問題に遭遇しているか

収集した情報はすべて、自動操作のための要件を定義する上で役に立ちます。

自動操作の目標と目的の設定

自動操作プロジェクトに関して、目標と測定可能な目的を設定することによって、事業およびデータ処理についての要件、および操作環境の向上に対して、プロジェクトがどれほどの効果をもたらすかを判別することができます。

自動操作に関する目標の設定

目標の設定は、計画のプロセスの重要な部分となるものです。事業およびデータ処理の要件に対する理解と、操作に関する問題の領域のリストがあれば、適切かつ長期にわたる自動操作の目標を設定することができます。

自動操作により期待される利点のカテゴリーについては、3 ページの『自動操作の利点』を参照してください。

これらのカテゴリーを確認することによって、各自の組織にとって最も重要なカテゴリーが何であるかを決定できるでしょう。また、各自の要求や環境を反映させた自らの目標を選択することもできるでしょう。自動化に期待する利点の中で最も重要なものを 3 つか 4 つ選択し、それを長期目標にすることによって、自動操作プロジェクトの焦点が明確になります。

測定可能な目的の設定

測定可能な目的を使用して、自動化の目標に向けての進捗度を判断します。長期の目標に対しては、1 つ以上の具体的な測定方法やインディケーターを識別してください。

自動操作プロジェクトにかかるコストと、それによってもたらされる利益を評価する場合、測定方法と計画が重要な役割を果たします。例えば、システムの可用性の向上が目標の 1 つだとすれば、システムの可用性について、現行の水準と達成したいと思う水準をきちんと知っていなければなりません。プロジェクトの特定の部分においてさらにリソースが必要かどうか、それ以外の部分を中断すべきか、それとも範囲を拡張すべきかどうか、および自動操作によりどの程度まで目標を達成するかを評価することができます。

55 ページの表 1 は、主な測定方法を示したワークシートの例です。このワークシートは 5 か年を視野に入れており、3 ページの『自動操作の利点』に示された自動操作による利益をもとに、目標の設定を行っています。主な測定方法は、自動操作の目標を反映し、それぞれの状況に適したものでなければなりません。

55 ページの表 1 にリストされた測定方法により利益を計算する方法については、『コストと利益の定量化』を参照してください。数々の目標について進行度の測定を行う際の指標の例をさらに必要とする場合は、587 ページの『付録 C. 進捗状況測定のサンプル』を参照してください。

コストと利益の定量化

使用できるインディケーター、それらの現在の測定方法、および自動操作のあとで予想される測定方法を識別すると、金額上の値を算出することができます。金額上の値を算出することによって、最大の利益をもたらす自動操作タイプの情報を得ることができます。また、金額上の値を算出することによって、自動操作の各形式を割り振る必要があるリソースの水準を判断することが可能になります。

自動操作プロジェクトに対する計画コストは、実施に必要な人的リソースとシステム・リソースの評価に基づいて算出されます。また、予想される利益は、各自動操作の目標に対して確立した測定方法と推定に基づいて算出されます。

作成したプロジェクト計画には、自動操作の計画、設計、実施に必要な多くのステップが示されます。自動操作計画のサンプルについては、579 ページの『付録 B. サンプル・プロジェクト計画』を参照してください。

作成した計画に基づいて、各ステップに必要なリソースを決定することができます。自動操作の残りの各フェーズに対して必要な人的リソース、およびシステム・リソースを識別してください。

次に、利益を計算してください。自動操作の目標に対して設定した測定方法と計画を使用すると、手操作から自動操作に移行することによって達成される節約について、見積もりを行うことができます。この節約は、自動操作による財務上の利益を示します。表 1 に、利益ワークシートの例を示します。

例えば、目標の 1 つが、ネットワークの拡張にともなうオペレーターの増員を避けることであれば、測定可能な目的では、手操作を続けた場合に追加すると予想されるオペレーターの数を指定する必要があります。同様に、自動操作を実施して、オペレーターの作業を単純化した場合には、追加しなければならないオペレーターの数がどれほど少なくてすむかについても見積もってください。そして、節約可能な金額を計算して、この領域で自動操作を行った場合の価値を評価してください。

可用性の向上も、重要な利益となり得ます。例えば、CICS の可用性の値を算出するには、次のステップを使用することができます。

1. CICS の自動操作なしでの、ユーザー 1 人あたりのダウン時間の年間合計を算出し、この数字から、自動操作で予測されるユーザー 1 人あたりのダウン時間の合計を引きます。
2. ダウン時間の差に、オペレーターやプログラマーなど、各クラスの CICS ユーザーの合計数を掛け合わせます。
3. この数字に、ダウン時間中に各ユーザーが、CICS を必要とする確率 (%) を、掛け合わせます。
4. さらにこの数字に、ユーザーの時間を金額に換算して掛け合わせます。

測定方法の中には、重複するものがあるかもしれません。例えば、データ・センターごとの人材節約の測定方法は、アプリケーションごとの人材節約の測定方法と重複する可能性があります。重複する測定方法がある場合は、節約分の合計に両方の数字を含まないように気を付けなければなりません。

表 1. 利益算出用ワークシートの例

領域	自動操作を行 わない場合	自動操作を行 った場合	年間の節約分	5 年間の合計
システムとネットワークの 可用性				
NetView プログラム				
CICS プログラム				
IMS プログラム				
TSO プログラム				
VTAM プログラム				
通信制御装置				
NCP プログラム				

表 1. 利益算出用ワークシートの例 (続き)

領域	自動操作を行 わない場合	自動操作を行 った場合	年間の節約分	5 年間の合計
拡張に対する制約の除去				
最大容量				
オペレーターの生産性				
要員の数				
現在				
1 年目				
2 年目				
3 年目				
4 年目				
5 年目				
整合性のある操作				
オペレーターが原因で起 った障害				
オペレーターの補充				
合計				

理解と協力

要件、目標、コスト、および利益を調査しておく、経営部門と組織全体のコミットメントを取得して、自動操作プロジェクトを推進する上で役立ちます。

大切なのは、自動操作の影響が及ぶ各部門またはグループから、コミットメントとサポートを取り付けることです。影響を受けるグループとしては、システムおよびネットワーク運用、システム・プログラミング、技術サポート、およびユーザーなどが考えられます。自動操作の設計および実施について、成功を収めるためには、これらのグループの協力が不可欠です。したがって、目標および予測される利益について、必ず各グループの理解を得るように心がけてください。

第 4 章 自動操作プロジェクトの設計

本章では、自動操作プロジェクトの設計フェーズについて説明します。このフェーズでは、自動操作を行う具体的なプロシーチャーを識別し、自動操作に必要な作業の明確化を行います。まず、プロジェクトの有効範囲と、手順を自動操作する順序を定義します。この情報をもとに、自動操作の構造を決定します。次に実施のための基礎作りとして、自動化アプリケーション・プログラム全体の慣例および規則を確立します。

プロジェクト設計作業の紹介に続いて、本章では、設計の目安になるようないくつかの指針について説明します。

プロジェクト設計作業

予備的な計画決定の検討を終えれば、次の設計作業を開始することができます。

自動化すべきプロシーチャーと機能の確認

自動操作を実施すべきプロシーチャーおよび機能の多くは、すでに初期計画の段階で確認済みかもしれません。オペレーターと話し合い、システム・ログおよびネットワーク・ログを検討し、50 ページの『操作環境について』で述べた情報源に立ち返ることが、自動化の追加の候補を見いだす一助となります。自動操作の対象として、例えば次のものを候補に挙げることができます。

- ・ オペレーターの時間を消費するプロシーチャー
- ・ 迅速かつ正確な応答を必要とするイベント
- ・ 機械的に実行できる反復的なプロシーチャー

プロシーチャーと機能の優先順位付け

自動操作の目標を達成するために必要な自動操作手順と機能を選択すれば、スケジュールを作成することができます。このスケジュールでは、手順と機能に優先順位を付けて、最小限の努力で最大の効果が得られる変更を優先することができます。このスケジュールは、組織において自動操作を実施して、それを理解するためにかかると思われる時間を反映するものです。

実施のためのスケジュール段階

自動操作手順と機能の実施をスケジュールする場合は、プロジェクトをいくつかの段階に分けることを考慮してください。そうすることにより、該当の環境において各変更内容をテストし、チューニングし、組み込むために、十分な時間が取れます。一連の段階の例については、8 ページの『自動操作の段階』を参照してください。

目標と目的を反映させて、独自の段階を設定することができます。

標準の確立

スケジュールの作成に加えて、設計チームは、標準を確立して、汎用の自動操作の技法を選択することができます。例えば、次の点について決定を下さなければならないでしょう。

- すべてのルーチンに関してセキュリティ上の問題へのアプローチ
- すべてのルーチンに関してログへのメッセージを書き込む形式
- 問題発生時にオペレーターに通知を行う共通の方法
- グローバル変数を使用してルーチン間で情報を共有するための共通のプロトコル

これらの事項を前もって決定すると、自動化への方法が統一されるため、保守は容易になり、自動化アプリケーションのすべての部分での通信の正確さを高めることができます。『設計のガイドライン』では、考慮する必要がある多くの問題について説明します。これにはプログラミングの問題をはじめ、運用上の変化が各自の環境および組織に与える影響などが含まれます。

設計のガイドライン

設計時には、以下の原則、提案、および指針を考慮します。

- 容易な拡張と伝搬のための設計（『拡張と伝搬の設計』を参照）
- 可聴性のための設計（59 ページの『監査能力の設計』を参照）
- セキュリティの設計（60 ページの『自動化セキュリティの設計』を参照）
- 可用性の設計（59 ページの『監査能力の設計』を参照）
- ソースに近いイベントの自動操作（61 ページの『ソースに近い自動操作』を参照）
- 1 つのシステムにつき、複数の NetView プログラムを使用するかどうかの選択（61 ページの『単一システム上での複数 NetView プログラムの使用』を参照）
- 有効なオペレーター・インターフェースの提供（62 ページの『オペレーター・インターフェースの提供』を参照）
- 自動操作のためのスタッフ教育（62 ページの『スタッフの教育』を参照）
- スタッフの役割変更の予測（63 ページの『スタッフの役割変更の予測』を参照）
- 自動操作のテストの提供（63 ページの『テストの提供』を参照）
- 問題および変更管理の提供（59 ページの『監査能力の設計』を参照）
- 信頼のおけるフォーカル・ポイントの選択（64 ページの『フォーカル・ポイントの選択』を参照）
- バックアップ・フォーカル・ポイントの使用の検討（66 ページの『バックアップ・フォーカル・ポイントの使用』を参照）
- オペレーターの制御範囲の定義（66 ページの『オペレーターの制御範囲の定義』を参照）

拡張と伝搬の設計

自動化環境に新規のソフトウェアまたは新規の装置を追加する場合の時間と労力を節約するために、拡張に向けた自動化を設計します。複数のシステムの自動化を計画している場合は、1 つのシステム用に作ったルーチンを他のシステムに簡単にコピーすることが、確実にできるにします。

拡張および伝搬のための設計方法の 1 つに、システム名とリソース名、および他の重要な情報をコマンド・プロシージャーに正確にコーディングする代わりに、それらに対してグローバル変数を使用するという方法があります。そうすると、単一の自動操作ルーチン・セットを使用して、グローバル変数を再定義することにより、それらを新しい装置、新しいソフトウェア、または新しいシステムに適用することができます。

並行シスプレックス環境では、その環境内の複数の MVS イメージ上で、NetView プログラムのコピーを実行することができます。このため、データ・セット名、区分データ・セット・メンバー名、およびこれらのメンバーの内容は、各 MVS イメージに固有である必要があります。各 MVS イメージの固有性を保持しつつ、このタイプのデータを並列シスプレックス間で共用できるようにすると、クローニング・サポートによって、必要なメンテナンスが少なくなります。別個の NetView 区分データ・セットが、固有のメンバー・データを持つ必要がなくなる可能性があります。この機能を利用するには、バージョン 5 リリース 2 以降の MVS が必要です。

グローバル変数の使用例については、651 ページの『付録 I. 自動操作のサンプル・セット』に記載している自動操作のサンプルを参照してください。また、グローバル変数を利用する代わりに、有益な情報を管理ファイルに保管することも可能です。

拡張および伝搬のためのもう 1 つの設計方法として、保存するシステム名とリソース名、および他の重要な情報をコマンド・プロシージャーに正確にコーディングする代わりに、リソース・オブジェクト・データ・マネージャー (RODM) を使用するという方法もあります。RODM の使用法については、30 ページの『リソース・オブジェクト・データ・マネージャー』、および 87 ページの『第 8 章 リソース・オブジェクト・データ・マネージャーでの自動操作』を参照してください。

自動操作を複数のオペレーティング・システムに伝搬させることを計画している場合には、すべてのオペレーティング・システムで実行できる言語で、コマンド・プロシージャーを作成するようにしてください。どのオペレーティング・システムでどの言語が実行できるかについては、26 ページの『言語の選択』を参照してください。

監査能力の設計

どの自動操作でも、優れた監査証跡が重要になります。NetView プログラムはシステム・メッセージを Canzlog ログに記録します。Canzlog ログには、各メッセージに関して記録された豊富な属性セットが入っており、関係のあるメッセージのみを示すフィルタリング・スキームが備わっています。オプションで、NetView プログラムは、メッセージをネットワーク・ログとシステム・ログのいずれか (あるいはその両方) に記録することもできます。また、順次ログを作成し、選択した基準に従って、いくつかのログに情報を分類することもできます。例えば、複数のサブシステムから送られたメッセージを別々のファイルに記録したり、各 Netview オペレーター、および自動タスクに別々のログを設定したり、または自動操作関連活動を追跡する独立したログを設定したりすることができます。

優れた監査証跡を設計するにあたっては、次のいくつかの基本原則が適用されます。

- 自動操作が開始した各アクションを記録する。
- 記録した各自動操作イベントにインディケーターを付けて、それが自動操作イベントであることを識別できるようにする。例えば、自動操作に関連するすべてのログ項目の前により大記号 (>) を付けるなど。
- できるだけ多くの関連情報を記録する。例えば、ある自動操作プロシーチャーを出す理由や、結果として更新することになるグローバル変数の名前など。
- 自動操作が問題を解決するたびに記録する。
- 自動操作が問題の解決に失敗するたびに記録する。これらの情報は、自動操作をアップグレードして改善するために使用することができます。

自動化セキュリティの設計

自動化環境は、許可されたオペレーターだけが、システムおよびネットワークの制御機能にアクセスできるように設計してください。オペレーターのアクセスは、パスワードを使用したり、制御スパンの決められたコマンド許可やその他の技法を使用して、データ・セットへのアクセスを制限したりすることによって制御することができます。詳しくは、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

例えば、オペレーターまたは自動化タスクが出すことができるコマンドを保護するには、NetView コマンド許可テーブルまたはシステム許可機能 (SAF) プロダクト (RACF (リソース・アクセス管理機能) など) を使用することができます。

自動化テーブルから出されたコマンド・プロシーチャーが、メッセージまたは MSU の発信元の有効性を確認するようにし、正しくないコマンドで応答することがないようにしてください。例えば、あるメッセージについて、それが予測どおりのシステム、ジョブ、またはオペレーティング・システムのコンポーネントからきたものであることが確認できます。自動化環境では、システム・コマンドとネットワーク・コマンドはいずれも、NetView コンソールから入力できるため、NetView コンソールへのアクセスを制御することが特に重要になります。

可用性の設計

それぞれのシステムまたはネットワークを信頼して自動操作を行うわけですから、自動化アプリケーションが確実に、常時機能しており、利用可能であるようにする必要があります。計画的な停止の回数を減らし、計画外の停止を回復させるための手段を考えてください。可用性を設計する際のアプローチは、自動操作を行うシステムが単一システムか複数システムかによって異なる場合があります。

単一のシステムにおいて自動リカバリーを実現するためのアプローチの 1 つとして、そのシステム上で、2 つの NetView プログラムを実行させる方法があります。61 ページの『単一システム上での複数 NetView プログラムの使用』を参照してください。これによって、2 つの NetView プログラムが、互いにモニターし合うという利点が得られます。一方が障害を起こすと、他方がそれを再始動できるのです。

複数システム・ネットワークで、別々のシステム上にある NetView プログラムが相互にモニターするようにし、また必要に応じてリカバリーを開始するようにします。ただし、この方法は、システム間に信頼のおけるリンクがないと使用できません。

フォーカル・ポイント・システムを使用して複数の分散システムを自動操作する場合は、そのフォーカル・ポイント・システムのバックアップを確保してください。これにより、基本フォーカル・ポイントが使用不可になった場合でも、分散システムは外部自動操作またはオペレーターのアクションを必要とする情報の転送を続けることができます。

いずれにしても、NetView プログラム内の自動化アプリケーションは、互いにモニターすることが可能です。これによって、自動タスクおよび自動化テーブルが継続的に機能することが確実にになります。例えば、1 つの自動タスクは、別の自動タスクにメッセージを送り、応答が正しい時間間隔で返されるかどうかを検査することによって、それをモニターすることができます。この種の照会を定期的に行うためには、EVERY コマンドを使用することができます。

ソースに近い自動操作

自動操作の手引きとなる規則の 1 つに、イベントの自動操作は、そのイベントのソースにできるだけ近いところで行うというものがあります。フォーカル・ポイント・システムからいくつかの分散システムを操作したい場合は、まず分散システム自体で自動操作できるものを、すべて自動操作してください。フォーカル・ポイントには、分散システムで処理できない問題だけを転送してください。ソースに近い自動操作を行うと、パフォーマンスと信頼性の両方を最適化することができます。

同様に、システム・メッセージが発信されるアドレス・スペースで MRT アクションを使用して、不要なシステム・メッセージを抑制できます。これにより、システムおよび NetView プログラムの両方で相当量の処理が節減されます。

単一システム上での複数 NetView プログラムの使用

1 つのシステムにつき、1 つの NetView プログラムがあれば、ネットワーク管理、ネットワークの自動操作、およびシステムの自動操作をはじめとする、すべての NetView 機能を実現することができます。ただし、組織によっては、これらの機能を、複数の NetView プログラムに分散させたいという場合があります。例えば各システムにおいて、1 つの NetView プログラムが、ネットワーク問題の判別をはじめとするネットワーク管理操作を行い、もう 1 つの方が自動操作を行う、といった具合です。あるいは、1 つの NetView プログラムで、すべてのネットワーク機能を実行し、もう 1 つのプログラムですべてのシステム機能を実行するという場合も考えられます。

自動操作の NetView プログラムは、それが自動操作するタスクよりも高いディスパッチング優先順位で実行させることが可能です。ネットワーク管理用の NetView プログラムは、自動操作やその他のタスクに干渉しないように優先順位を低く設定することが可能です。

1 つのシステムに、複数の NetView プログラムをインストールすると、組織内の各グループが、それぞれ必要な NetView 機能を、独自に利用できるようになります。例えば、システム運用グループとネットワーク管理グループで、別々の NetView プログラムを利用することが可能になります。

しかし 1 つのシステムで、複数の NetView プログラムを実行させることについては、複雑さが増すなどの欠点もあります。2 つの NetView プログラムを実行させるということは、ライブラリーとログのコピーを 2 つ維持しなければならないから

です。2つの NetView プログラムが、継続的にメッセージをやり取りするというエンドレス・ループが起きないように、注意しなければなりません。また、2つのプログラムを使用すれば、ストレージに関する要件も大きくなります。

NetView メッセージ改訂テーブルを同じシステム上の2つの NetView から実行することにした場合は、必ず SSI の起動順序を検査してください (この検査はコマンド D SSI で行えます)。後の MRT によって、前の MRT で行われた設定がオーバーライドされます。

複数の NetView プログラムを実行させる際の詳細については、513 ページの『第32章 単一システムでの複数 NetView プログラムの実行』を参照してください。

オペレーター・インターフェースの提供

自動操作計画を設計するにあたって、オペレーター・インターフェースは重大な意味をもちます。システムまたはネットワークの操作、自動化アプリケーション・プログラムの運用操作、および自動操作のモニターに必要な情報を、オペレーターが確実に受け取れるようにしてください。

NetView プログラムが提供している、自動化環境モニター用のオプションについては、13 ページの『オペレーター・インターフェースの改善』を参照してください。例えば、インストールされている NetView フィーチャーによって、オペレーターは次の機能が提供する情報をモニターすることができます。

- コマンド機能
- ハードウェア・モニター
- 状況モニター
- NetView 管理コンソール
- 自動化オペレーション・ネットワーク (AON)
- フルスクリーン・パネルとヘルプ・パネル
- Canzlog ログ

ほとんどの環境において、オペレーターは、システムの稼働状態を判断するのに、メッセージを使用することに慣れています。このため、抑止する必要があるメッセージと取る必要がある自動化アクションの決定は、オペレーターを含めて行う必要があります。システムが正常に機能していることを検証するために必要な情報を、確実にオペレーターに提供するようにしてください。

自動操作を始める時点では、なんらかのアクションを必要とするイベントから自動操作コマンド・プロシーチャーの送信に至るまで、すべてをオペレーターに通知することができます。最終的に自動操作が標準の操作モードとなり、操作スタッフが自動操作に満足した時点で、通知を省略し、オペレーターによるアクションが必要な場合のみ、オペレーターに通知を行うようにすることができます。ただし、システムまたはネットワークにおいて自動操作が発生した時間を示すメッセージについては、ログ記録を継続してください。これらのメッセージは、自動操作が障害を起こした場合に問題判別の手掛かりになります。

スタッフの教育

自動化アプリケーション・プログラムを設計および実施する担当者は、実際に自動操作に取り組む前に十分な訓練が必要です。組織の要件と、自動操作に使用しているプロダクトの両方について、理解する必要があります。仕事を分割する場合は、

グループごとに必要な訓練の内容も変わってくるはずです。例えば 1 つのグループが自動操作プロシーチャーを作成し、他のグループが自動操作画面を作成する、という状況が考えられます。

さらに、自動操作のどの段階でも、操作環境の変更について、オペレーターに通知を行う必要があります。オペレーターは、新しいオペレーター・インターフェースと、自分たちの責任の変更について理解する必要があります。自動操作を確実に成功させるためには、常に教育が必要になります。

手操作によるシステム運用の訓練を継続して、自動操作が失敗した場合にオペレーターが再び運用を担えるように備えておくのも、1 つの方法でしょう。しかし通常は、自動操作を再開する訓練を要員に行う方がより効率的です。手操作運用技術の教育に労力を費やすのではなくて、自動操作をテストし、障害を避けるためのバックアップおよびリカバリー計画を実施できます。オペレーターおよびプログラマーが、文書を使用して、必要に応じてプロシーチャーを手操作で実行できるように、自動操作を文書化しておいてください。

スタッフの役割変更の予測

自動操作を行うと、データ処理のスタッフ・メンバーの役割、およびメンバー間の連絡の仕方が、変更されることがあります。これらの変更と、自動操作によって従業員と組織にもたらされる影響について、必ず十分な検討を行ってください。

例えばシステムとネットワークの自動操作を結合することを考えているなら、システムとネットワークのオペレーターをまとめることもできます。システムとネットワークの自動操作には、共通の設計を使用しているため、自動操作で処理できない問題を解決する人々は、システムとネットワークの両方のリソースについて理解する必要があります。

役割の変化の例として、もう 1 つ、オペレーターの経歴に関する変化が予測されます。システムが自動化され、ネットワークのモニターやルーチン、繰り返し要素の強いタスクが自動化されるにつれて、オペレーターは、意思決定や特別な問題の解決をはじめ、自動化アプリケーションそのものの扱いといった作業に、もっと多くの時間を割くことができるようになります。これらの変更に適応するための方法の 1 つには、自動操作専門家といった、オペレーターの新しいジョブ・カテゴリーを設けるというものがあります。こういった専門家は、システムとネットワークを稼働させるために使用される自動化アプリケーション・プログラムだけでなく、システムとネットワークの操作についても理解していなければなりません。自動操作手順の作成または援助を行うオペレーターは、自動操作技術を会得して、今後の環境での操作方法を学ぶことができます。

テストの提供

新規プロダクトやアプリケーションの場合と同様に、自動操作プロシーチャーを実動環境に置く前に、必ずテストを実施してください。実施の各段階において、綿密で周到なテストが必要です。さらに、システムまたはネットワークに変更があった場合も、自動化アプリケーションの再テストを行って、新しいオペレーティング・システムとアプリケーション・プログラム、および新しいハードウェアで、ルーチンが正しく動作することを確認してください。

問題および変更管理の提供

問題管理は、自動操作の重要な部分となるものです。自動操作タスクがリカバリーのためのアクションを起こす前に問題レコードをログ記録することによって、注意が必要なシステムおよびネットワーク問題のレコードを失う危険を、最小限にとどめることができます。

自動操作の実施も、変更管理に影響を及ぼします。自動操作に際して、操作環境に加えられた変更すべてを、以前よりも詳細に追跡するようにしておく有効です。例えば、メッセージ形式のほんのわずかな変更であっても、メッセージが自動操作を起動すると、操作に影響の及ぶ可能性があります。自動操作を起動するメッセージ、アラート、およびその他のデータ・レコードのリストを維持してください。各メッセージについて、メッセージ ID だけを使用するのか、あるいはメッセージのその他の部分も使用するのかを記録してください。メッセージまたはアラートに対する変更がわかったら、それらをこのリストと照らし合わせて、自動操作を更新する必要があるかどうかを調べてください。

フォーカル・ポイントの選択

複数システム環境では、各システム上で独立して実行している単一システムで、多くの自動操作タスクを行うことができます。ローカルで自動操作しないタスクについては、関連データを指定のフォーカル・ポイント・システムへ送信することができます。その後、フォーカル・ポイント・システムの自動操作を通して、そのデータに対して自動応答することが可能です。あるいは、そのデータをオペレーターに表示することもできます。

フォーカル・ポイントを選択するにあたっては、フォーカル・ポイントに実行させたい作業の種類について考慮してください。フォーカル・ポイントをどのように利用するかによって、どのシステムをフォーカル・ポイントに選ぶかが異なってきます。以下に、フォーカル・ポイントの選択に影響を及ぼす可能性のある考慮事項を示します。

- ・ フォーカル・ポイントでは、2 つ以上のシステム間で整合が必要な自動操作活動を行うことができます。
- ・ フォーカル・ポイントでは、他のシステムにおける自動操作機能をモニターし、障害の際にそれらの機能の回復を図ることができます。
- ・ フォーカル・ポイントでは、他のシステムのハードウェアおよびソフトウェアをモニターして、障害の際にそれらのリカバリーを図ることができます。
- ・ フォーカル・ポイントのオペレーターは、自動操作の処理できない例外条件に応答できます。

フォーカル・ポイントには、安定性の高い信頼できるシステムを選んでください。一般に、すでに過負荷になっているシステムは選択しないようにしてください。また、アプリケーション・プログラム開発、新プロダクトのインストールとテスト、または他のテストに使用するシステムも避けてください。

フォーカル・ポイントとなるシステムには、他のシステムで発生した問題をログ記録するための、情報管理プロダクトをインストールしておかなければなりません。また、問題管理、変更管理、および報告のためのプログラムなどの、システム管理アプリケーション・プログラムも必要です。

コミュニケーション管理構成 (CMC) システムがある場合、ユーザーのシステムの中でこの CMC システムの使用可能度は最も高くなります。したがって、CMC システムをフォーカル・ポイントとして使用することができます。CMC システムの容量を検討して、そのシステムが CMC と自動操作の仕事を組み合わせた処理負荷を処理できることを確認してください。

図 7 は、フォーカル・ポイント・システムが分散システムを管理する方法を示したものです。図が示すように、分散システムは複数の場所に設置することも可能です。

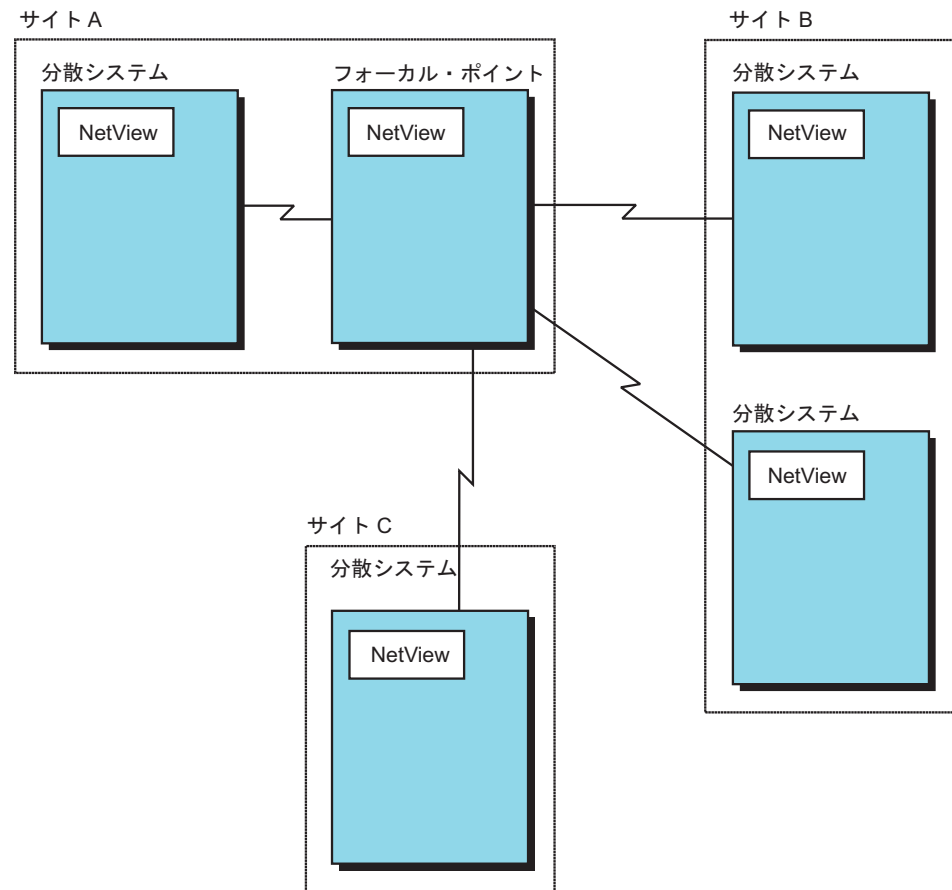


図 7. 複数サイト構成の例

分散システムのさまざまなタイプのデータをフォーカル・ポイントへ転送することができます。これには、メッセージ、アラート、状況情報、および LU 6.2 トランスポート用のユーザー定義の情報クラスなどが含まれます。

各データ・タイプごとの NetView プログラムの転送機能の概要については、423 ページの『第 26 章 集中操作』を参照してください。メッセージ、アラート、および状況転送のセットアップ方法に関しては、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。詳しくは、「*IBM Tivoli NetView for z/OS*アプリケーション・プログラマーズ・ガイド」を参照してください。

フォーカル・ポイントは、単一でも複数でも構いません。ただし、フォーカル・ポイントが複数ある場合は、通常、各分散システムはすべてのタイプのデータを 1 つ

のフォーカル・ポイントに送信します。つまり、特定のシステムから転送されたアラート、メッセージ、状況情報、および運用管理情報が、すべて同じフォーカル・ポイントに届くことになります。このようなタイプの設計においては、フォーカル・ポイントのオペレーターおよび自動操作は、1つの場所から送られたあらゆるタイプのデータをモニターすることが可能になります。

バックアップ・フォーカル・ポイントの使用

最大 8 つのフォーカル・ポイントをバックアップ用に定義することができます。1 つのフォーカル・ポイント・システムに他の多くのシステムを管理させたい場合、バックアップを使用して、フォーカル・ポイントの障害により自動操作が中断されないようにすることができます。バックアップ用のフォーカル・ポイントは、分散システムの 1 つでもバックアップ専用のシステムでも構いません。ただし障害発生時には、フォーカル・ポイントを引き継ぐことのできるシステムでなければなりません。

多くのタイプのデータにおいて、基本フォーカル・ポイントとの通信が失われた場合に、バックアップのフォーカル・ポイントと分散システムの間で、NetView 対 NetView のセッションを自動的に確立することができます。これは、オペレーターの介入なしで行うことができます。バックアップ・フォーカル・ポイントをサポートしていないのは、状況転送だけです。

その他、バックアップ・フォーカル・ポイントを定義することによって次の利点が期待できます。

- 基本フォーカル・ポイントとバックアップ・フォーカル・ポイントは、相互に相手モニターさせるようにすることができます。通信が失われると、リカバリー・アクションを起動することができます。
- リンクに障害が発生した場合、VTAM プログラム、および NCP によってネットワークのリンクを回復することができます。
- 基本フォーカル・ポイントと分散システムの間、複数の NetView 対 NetView のセッションを確立することができます。このとき、各セッションごとに異なる経路を使用するようにしてください。

オペレーターの制御範囲の定義

制御範囲を定義することによって、フォーカル・ポイントにいるオペレーターは、そのフォーカル・ポイントとエントリー・ポイント (分散ノード) との関係を管理できるようになります。各エントリー・ポイントはタイプと状況によって分類されます。フォーカル・ポイントのオペレーターは、FOCALPT DISPSOC コマンドを使用してこの結果を表示させることができます。

65 ページの図 7 では、フォーカル・ポイントはサイト A にあり、4 つの分散 NetView システムを包括する制御範囲を管理します。エントリー・ポイントは、サイト A に 1 つ、サイト B に 2 つ、またサイト C に 1 つずつあります。

フォーカル・ポイントのオペレーターは、制御範囲マネージャー (SOC-MGR) を通して、制御範囲を管理できます。制御範囲関係の確立とリカバリー、および状況の提供は、フォーカル・ポイントまたはエントリー・ポイント内の MS-CAPS アプリケーションが行います。フォーカル・ポイントのオペレーターは、エントリー・ポ

イントの追加および削除や、制御範囲構成ファイルへの情報の追加を行うことができます。このファイルを、NetView の初期設定時に使用することによって、制御範囲環境を設定することができます。

詳しくは、423 ページの『第 26 章 集中操作』を参照してください。

第 5 章 自動操作プロジェクトの実施

本章では、自動操作プロジェクトの実施、および実動フェーズにかかわる作業について説明します。

広範囲に及ぶ自動操作プロジェクトを想定している場合は、57 ページの『第 4 章 自動操作プロジェクトの設計』で説明しているように、これをいくつかの段階に分けて実施してください。この場合、自動操作の各段階に対応した、実施フェーズと実動フェーズがあります。各段階に対して、この章の作業を繰り返してください。

実施作業

設計フェーズでは、さまざまな機能と手順を実施するためのスケジュールを作成しました。ここでは、指定の順序でこれらの機能を 1 つ 1 つ検討します。自動操作すべき各機能に対して、次の方法を適用します。

1. 手操作による操作方式を分析する。ある機能を自動操作する際は、NetView 機能で、通常オペレーターが実施しているステップ・シーケンスをなるべく忠実に再現させるのが最善策である場合がよくあります。またそうでなくても、自動方式を設定する前には手操作方式を理解する必要があります。
2. その機能を自動操作するための最善のアプローチを判別する。
3. 開発環境で、この機能に使用する予定のプロダクトをインストールする。
4. この機能に使用する予定のアプリケーション・プログラム、およびコマンド・プロシーチャーを開発する。
5. テスト環境で、それらのアプリケーション・プログラムおよびコマンド・プロシーチャーをインストールする。
6. これらのアプリケーション・プログラム、およびコマンド・プロシーチャーをテストし、デバッグする。
7. これらのアプリケーション・プログラムとコマンド・プロシーチャーのパフォーマンスを測定する。調整を行い効率を改善する。

すべての自動操作プロダクト、機能、アプリケーション、およびプロシーチャーについて十分なテストと調整を終えたら、実動フェーズへ進みます。

実動作業

実動フェーズでは、これから実施する変更について、オペレーターを教育することから始める必要があります。

オペレーターの教育を終えたら、自動操作をサポートするために実動システムに加えるプロダクトがあれば、そのインストールを開始してください。これらのプロダクトをテストして、実動システム上で正しく実行することを確認してください。

次に、開発した自動操作機能と自動操作手順をインストールしてください。これらの機能を実動システムに適用するために必要な変更を行ってください。57 ページの『第 4 章 自動操作プロジェクトの設計』で述べたように、伝搬が容易な設計に

なっていれば、ほとんどの場合、管理ファイル内のいくつかのグローバル変数またはデータを変更するだけで済みます。自動操作機能をテストして、必要な訂正や拡張があればそれを行ってください。

プロジェクトをいくつかの段階に分けた場合は、次の段階に進みます。自動操作における各ステージの説明については、8 ページの『自動操作の段階』を参照してください。

すでに導入した自動化を継続的に再調査し、検討してください。これには、結果を測定して、プロジェクト定義フェーズで識別した推定値とそれらを比較してください。測定方法を利用して自動操作の結果を追跡する方法については、54 ページの『測定可能な目的の設定』を参照してください。

自動操作を改善する方法を調査してください。抑止できるメッセージや、自動応答を受け取ることのできる MSU がまだあるはずです。機能やプロシーチャーを積極的に調整および拡張することによって、自動操作から最大の利益を得ることができます。

システムの自動化テーブル使用報告書を作成するには、AUTOCNT コマンドを使用します。この報告書を利用して自動化テーブルのステートメントを分析し、これらのステートメントがどれくらいの頻度で一致しているかを見ることができます。頻繁に一致しているステートメントをテーブルのいちばん上に移動させれば、不一致基準のチェック回数を少なくすることができます。さらにユーザーは、論理エラーを起こす非適合ステートメントをテーブルから削除するか、あるいは変更するかを決めることができます。

自動化テーブル使用報告書に目を通すことによって、自分のシステムで実現済みの自動操作のレベルを判断することができます。これらの統計は、管理目的の報告書として極めて有用です。AUTOCNT コマンドおよび自動化テーブル使用報告書に関する詳細については、163 ページの『第 15 章 自動化テーブル』を参照してください。

第 3 部 選択環境における自動操作の計画

第 6 章 MVS 拡張複数コンソール・サポート・コンソールを使用した自動操作

本章では、33 ページの『MVS システムにおける自動化』で述べた拡張複数コンソール・サポート (EMCS) コンソールについて、さらに詳しい情報を提供します。本章では、以下のことを説明します。

- NetView 自動操作で EMCS を使用する際の、利点、考慮点、および計画時の考慮事項
- MVS サブシステム・インターフェースの代わりに EMCS を使用する場合に得られるいくつかの利点

NetView プログラムで EMCS コンソールを使用する

EMCS コンソールを使用すると、MVS アプリケーション・プログラムは、端末のオペレーターと同じように MVS システムと対話ができるようになります。拡張複数コンソール・サポート・コンソールを使用すると、NetView 自動操作は、NetView オペレーターが MVS オペレーターであるかのように、MVS システムと対話ができるようになります。

拡張複数コンソール・サポート・コンソールを使用すると、NetView 自動操作と MVS システムの対話において、MVS システムの他のバージョンにあった制約のいくつかがなくなります。例えば、拡張複数コンソール・サポート・コンソールを、PARMLIB データ・セットの CONSOLxx メンバーに定義する必要はありません。

NetView プログラムはサブシステム・インターフェースを使用して非送信請求 MVS メッセージを処理し、拡張複数コンソール・サポート・コンソールを使用して送信請求コマンド応答を処理します。これにより、MVS 処理では非送信請求メッセージをより早く抽出でき、同時にオペレーターは EMCS の柔軟性を享受できます。

拡張複数コンソール・サポート・コンソールについての詳細は、MVS ライブラリーを参照してください。NetView で使用する、拡張複数コンソール・サポート・コンソールの属性についての詳細は、「IBM Tivoli NetView for z/OSセキュリティ・リファレンス」を参照してください。

NetView プログラムで EMCS コンソールを使用することの利点

NetView プログラムで、EMCS コンソールを使用する利点は、次のとおりです。

- 使用できる MVS オペレーター・コンソールの定義上の数に制限がない。
- NetView オペレーターに対し、MVS コンソールを動的に定義することができ、オペレーター・コマンドおよび応答が、各オペレーターに割り振られた EMCS コンソールを使用することに注意してください。
- DEFAULTS コマンドの SCRNFMT パラメーターを使用して、NetView コマンド機能画面に、MVS オペレーター・コンソールと同じような形式で情報を表示させることができる。

- コンソールを PARMLIB データ・セットの CONSOLxx メンバーに定義する必要がない。
- オプションにより、システム・メッセージを経路コードに基づいて直接配布することができる。ただし、この方法を使用する場合は、複数のオリジナル・メッセージが NetView プログラムに送られることがあるので、自動操作の重複に注意してください。
- オペレーターの権限をより容易に定義できる。

使用法:

1. データの損失を防ぐため、すべてのクロスドメイン・セッションでは、RMTCMD コマンドを使用してください。これを使用しないと、オペレーター端末タスク (OST) と NetView 間タスク (NNT) の間でセッションが確立された場合、送信されるメッセージから、付属のメッセージ・データ・ブロック (MDB) データ構造が完全に欠落してしまいます。データ構造には、そのメッセージに関する特別情報が含まれています。さらにデータ構造には、そのメッセージに関連するオペレーター・メッセージ削除 (DOM) 情報も含まれています。そのため、OST-NNT セッションでは、MDB データ構造にあるこのような情報は失われます。

MDB データ構造が欠落した状態でメッセージを送信する場合は、MDB 情報を処理できない、初期リリースの NetView プログラムとの互換性があります。

2. 拡張複数コンソール・サポート・コンソールの属性の変更を行った場合、NetView プログラム内の複数のコンソールが、同じ MVS システム・メッセージを請求する可能性があります。そのようなメッセージは NetView プログラムでは送信請求と見なされます。自動操作の重複に注意してください。

拡張複数コンソール・サポート・コンソールの使用計画

このセクションでは、NetView 自動化で、拡張複数コンソール・サポート・コンソールの使用を計画するにあたって、考慮すべき点について説明します。

拡張複数コンソール・サポート・コンソールを使用可能にする

拡張複数コンソール・サポート・コンソールを使用可能にするには、CNMSTYLE メンバーの MVSPARM ステートメントで値を指定します。

コンソールの命名規則の確立

拡張複数コンソール・サポート・コンソールを使用する前に、コンソールの命名規則を確立してください。

注: スタイル指定で ConsMask キーワードを使用すると、固有のコンソール名を選択する作業を単純化できます。

次のものは、コンソールの名前を作成するときの規則です。

- 個々の名前の長さは 2 から 8 文字でなければならない。
- 先頭文字は A から Z、@、#、および \$ のいずれかでなければならない。
- 残りの文字は A から Z、0 から 9、@、#、および \$ のいずれかでなければならない。

コンソール命名規則を使用する場合には、

- 各コンソール名は、1 つのシステムにおいて、またシスプレックス構成では全システムにおいて、それぞれ固有のものでなければならない。
- PARMLIB データ・セットの CONSOLxx メンバーに定義されたコンソール名は、拡張複数コンソール・サポート・コンソールの名前として使用することはできない。
- コンソール名は他のアプリケーション・プログラムで使用しても構わないが、重複してはならない。

拡張複数コンソール・サポート・コンソールの獲得

EMCS コンソールは、MVS コマンド、または NetView GETCONID コマンドにより、獲得できます。CONSOLE キーワードを指定せずに、MVS コマンドまたは GETCONID を出して EMCS コンソールを獲得する場合、当該タスクがまだコンソールを獲得していないと、NetView プログラムは、以下の順にコンソール名を判断します。

1. SETCONID コマンドが使用された場合、それにより指定された名前が使用されます。
2. CNMSTYLE メンバーに組み込まれた CNMSTUSR メンバーまたは CxxSTGEN メンバーの ConsMask ステートメントがアスタリスク (*) として定義されていない場合は、このステートメントの値がデフォルトのコンソール名を判別するためのマスクとして使用されます。詳しくは、「*IBM Tivoli NetView for z/OS*アドミニストレーション・リファレンス」を参照してください。
3. オペレーターがログオンした際にOPERSEC=SAFDEFが有効であった場合、NetViewプログラムは、SAF製品のNetViewセグメントで指定されているCONSNAMEの値を使用します。NetViewセグメントに CONSNAMEがない場合は、ステップ5を参照してください。
4. オペレーターがログオンした際にOPERSEC=SAFDEFが有効でなかった場合、NetViewプログラムは、DSIPRFのオペレーターのプロファイルで指定されているCONSNAMEの値を使用します。オペレーターのプロファイルにCONSNAMEがない場合は、ステップ5を参照してください。
5. CONSNAMEがNetViewセグメントにもオペレーターのプロファイルにも指定されていない場合、NetViewプログラムは、オペレーターのタスク名をコンソール名として使用します。この場合、オペレーター ID は 2 文字以上の長さになっていて、コンソール名の場合と同じルールに従っていなければなりません。

GETCONID コマンドを出すと、EMCS コンソールを獲得することができ、呼び出し側が指定した名前とコンソールの他の属性を指定することができます。

GETCONID コマンドおよび SETCONID コマンドの詳細については、NetView オンライン・コマンド・ヘルプを参照してください。拡張複数コンソール・サポート・コンソールに関連した属性についての詳細は、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

メッセージ改訂テーブル (MRT) またはメッセージ処理機能 (MPF) テーブルを使用した NetView 自動操作へのメッセージの送信

NetView のメッセージ改訂テーブル (MRT) またはシステムのメッセージ処理機能 (MPF) を使用して、メッセージに自動操作のマークを付けることができます。NetView 自動操作へのメッセージ送信についての情報は、590 ページの『メッセージ処理におけるサブシステム』に記載されています。

メッセージ改訂テーブル (MRT) を使用して、メッセージ処理機能 (MPF) によって提供される機能を実行することもできます。メッセージ改訂テーブルについての追加情報は、30 ページの『メッセージ改訂テーブル』に記載されています。

拡張複数コンソール・サポート・コンソールの属性値の使用

拡張複数コンソール・サポート・コンソール用として使用される特定の属性とそれらの値は、デフォルトとして提供されています。デフォルト値の完全なチャートについては、「IBM Tivoli NetView for z/OS セキュリティー・リファレンス」を参照してください。

オペレーターが獲得したコンソールのデフォルト値

オペレーターにより獲得されたコンソールのデフォルト値とその意味は、次のとおりです。

MSCOPE = *ALL

コンソールは、シスプレックスのいずれのメンバーからのメッセージでも受け取ることができ、全システムからのコマンド応答を受け取ることができ

ROUTCODE = NONE

コンソールは、経路コードによるシステム・メッセージの送信請求を行わない。

経路コードの使用

拡張複数コンソール・サポート・コンソールへのメッセージを経路コードを使用して送信請求する場合は、それらのメッセージの宛先コンソール名がタスクで所有される EMCS コンソール名に一致しない限り、メッセージは CNMCSSIR タスクでのみ自動化されることに注意してください。EMCS コンソールを、特定の経路コードを持つメッセージを受け取るように設定した場合、その経路コードが指定されたメッセージは、以下の受信者に送信されます。

- 指定された EMCS コンソール
- メッセージを送信請求した他のすべてのコンソール
- CNMCSSIR タスク

メッセージの中には、複数の経路コードを持つものがあります。経路コードでメッセージを送信請求すると、NetView プログラムで使用している拡張複数コンソール・サポート・コンソールには、1 つのメッセージの複数のインスタンスが配布されることがあります。コンソールの属性を設定するときは、同じメッセージについて、複数のインスタンスを送信請求することはなるべく避けてください。同じメッセージについて複数のインスタンスを送信請求する場合、複数のオペレーターにそ

のメッセージのコピーが 2 つ表示されたり、異なるオペレーターに同じオリジナル・メッセージが表示されたりする可能性があります。

宛先コンソール名が、タスクで所有される EMCS コンソール名に一致するため、メッセージがオペレーター・コンソール端末または自動タスクで受信される場合、メッセージはそのタスクで自動化されます。これは、自動化を要求したのが MPF かメッセージ改訂かには関係なく行われます。ただし、ブロードキャスト・メッセージが自動化されるのは、CNMCSSIR タスクによるか、自動化が要求された場合のみです。

宛先コードの設定により、メッセージがオペレーター端末または自動タスクで受信される場合、そのメッセージが受信タスクによって自動化されることはありません。自動化要求が MPF またはメッセージ改訂から出された場合、メッセージは CNMCSSIR タスクによって自動化されます。この動作を変更するには、インストール・システム出口 17 をコーディングします。詳しくは、「*IBM Tivoli NetView for z/OS プログラミング: アセンブラー*」を参照してください。

以下に、重複したメッセージ送信請求によって、NetView プログラムが自動操作の重複を引き起こす可能性のあるケースを示します。

ケース 1

使用中のコンソール:

- CON4 EMCS コンソールは、経路コードが 4 であるメッセージを受信するように設定されている。この設定は、MVS VARY コマンドで行った場合と、RACF OPERPARM セグメントで行った場合が考えられる。
- CON6 EMCS コンソールは、経路コードが 6 であるメッセージを受信するように設定されている。
- CNMCSSIR タスクは、自動操作作用のマークが付けられているメッセージ、あるいは設定済みの NETVONLY 値または REVISE('Y' AUTOMATE) 改訂テーブル・アクションの対象となるメッセージを受信している。

イベント:

MVS システムの出した IEExxxx メッセージの経路コードが 4 であり、しかもこのメッセージには自動操作作用のマークが付けられています。

結果: CON4 コンソールは、MVS システムからメッセージを受信します。メッセージには経路コードとして 4 が割り当てられているためです。ただし、CON4 はメッセージの宛先コンソールではないため、CON4 を所有するタスクはメッセージの自動操作に対応しません。

CNMCSSIR タスクは、サブシステム・インターフェースのメッセージ・インターフェース・システムからこのメッセージを受信します。メッセージには改訂での自動操作作用のマーク AUTO(YES) が付いているためです。CNMCSSIR タスクのみが自動操作に対応します。自動操作の結果によっては、CON4 を所有するタスクにメッセージの両方のコピーが表示される場合があります。

注: NetView MRT に指定されているのが REVISE('Y' AUTOMATE) ではなく、NETVONLY 値であれば、CON4 コンソールにもその他のコンソールに

も元のメッセージが表示されることはありません。ただし、CNMCSSIR タスクは依然としてメッセージを受け取り、通常どおりに自動操作を行います。

ケース 2

使用中のコンソール:

ケース 1 に同じ

イベント:

MVS システムの出したメッセージ IEEyyyy の経路コードが 4 および 6 であり、しかも MPF テーブルで AUTO(NO) のマークが付けられています。

結果: CON4 は、MVS システムからメッセージを受信します。なぜなら、メッセージには経路コードとして 4 が割り当てられているからです。CON6 は経路コードとして 6 が割り当てられているため、メッセージを受信します。どちらのコンソールもメッセージの宛先ではないため、メッセージがいずれかのコンソールによって自動化されることはありません。

属性のもたらす影響

先の例から、拡張複数コンソール・サポート・コンソールの属性設定が、MVS システム・メッセージの配布に影響を及ぼすことがわかります。拡張複数コンソール・サポート・コンソールの属性についての詳細は、「*IBM Tivoli NetView for z/OS セキュリティー・リファレンス*」を参照してください。

経路コードによりメッセージを送信請求する場合は、メッセージの中には経路コードを持たないものもあることを念頭に置いてください。したがって、コンソールが経路コードを持つメッセージを受信するように定義したとしても、システム内の全メッセージを受信するとは限りません。例えばモニター・タイプのメッセージは、経路コードを持っていません。MVS システム・メッセージとその経路コードのリストについては、MVS ライブラリーを参照してください。

セキュリティ・アクセスのインプリメンテーション

リソース・アクセス管理機能 (RACF) などのセキュリティ・アクセス機能プロダクトをインプリメントすることによって、NetView オペレーター・タスク、および自動タスクにセキュリティを行うことができます。RACF を使用して、拡張複数コンソール・サポート・コンソールの名前に対するアクセスを保護する方法、およびオペレーターと自動タスク用のシステム・コマンドを保護する方法については、「*IBM Tivoli NetView for z/OS セキュリティー・リファレンス*」を参照してください。

MVS メッセージ・データ・スペースの満杯によるメッセージ消失を防ぐ

拡張複数コンソール・サポート・コンソールに書かれるメッセージは、NetView が検索を行うまで、一時的に MVS メッセージ・データ・スペースに保管されます。MVS メッセージ・データ・スペース内のメッセージが、操作中に設定された最大記憶値を上回ると、MVS システムは、NetView が使用している拡張複数コンソール・サポート・コンソールの、メッセージ・データ・スペースへのメッセージ配布を一時停止します。この問題を避けるためには、メッセージ転送用の最大データ・

スペースをセットする、NetView の GETCONID のコマンドのデフォルト値を利用するとよいでしょう。このデータ・スペースは、MVS システムにより管理され、必要がある場合にのみ使用されます。

注: この制限は、自動操作の CNMCSSIR タスクによって取得された非送信請求メッセージには適用されません。

キュー制限の超過によるメッセージの消失を防ぐ

EMCS コンソールにはそれぞれ、QLIMIT と呼ばれる属性があります。この属性は、そのコンソールのデータ・スペースで、一度にキューに入れられるメッセージの数を定義します。キューの数が制限に達すると、MVS システムはそのコンソールへのメッセージ配布を一時停止し、これらのメッセージは失われてしまいます。

各 EMCS コンソールにはもう 1 つ、ALERTPCT と呼ばれる属性があります。この属性は、ある特定のコンソールについて、キュー制限に近づきつつあるかどうかを判断する際に役に立ちます。ALERTPCT 属性は、キュー制限があるパーセンテージに達したときに、警告メッセージを出すように定義します。

あるコンソールでメッセージ・キューが制限に達する場合、送信されたすべてのメッセージを処理するだけの時間がタスクに不足している、といったケースが考えられます。タスクによっては、他のタスクに比べて優先順位が低く、すべてのメッセージを処理するだけの十分な時間が確保できない場合があります。

MVS メッセージ・ストレージとメッセージ・キュー制限のバランス

MVS メッセージ・データ・スペースに確保するストレージの量と拡張複数コンソール・サポート・コンソールを使用するオペレーターの数、および定義したメッセージ・キュー制限の値について、バランスを図る必要があります。メッセージ・データ・スペースのストレージを確保するには、GETCONID コマンドの STORAGE パラメーターを使用します。キュー制限を定義するには、GETCONID コマンドの QLIMIT パラメーターを使用します。

注: RACF OPERPARM セグメントを使用すると、QLIMIT および STORAGE 属性を設定することができます。

データ・スペースの最大許容量を設定するには、STORAGE パラメーターを使用します。この最大ストレージ値は、NetView の中で、最初にアクティブになったコンソールにより設定されます。最初に活動化される EMCS コンソールによって、ユーザーが希望する最大ストレージ値を確実に設定するようにしてください。QLIMIT パラメーターで各コンソールに定義するキュー制限値は、そのコンソールだけに適用されます。

次のメッセージは、拡張複数コンソール・サポート・コンソールにおいてストレージ制限およびキュー制限に達していることを示すメッセージです。したがって、これらのメッセージには特に正しく応答する必要があります。

- DWO201I
- DWO202I
- DWO204I

これらのメッセージの詳細については、NetView オンライン・ヘルプを参照してください。

サブシステム・インターフェースから拡張複数コンソール・サポート・コンソールへのマイグレーション

サブシステム・インターフェースから EMCS コンソールへマイグレーションするには、いくつかの段階を踏む必要があります。以下に、推奨される手順を示します。サブシステム・インターフェースと EMCS コンソールの使用法については、91 ページの『NetView プログラム・インターフェース』を参照してください。

固有の名前の確立

EMCS コンソールを使用する前に、その命名規則を確立してください。コンソール名の付け方については、74 ページの『コンソールの命名規則の確立』を参照してください。

各ホストでの新しいリリースの NetView プログラムへのマイグレーション

コミュニケーション管理構成 (CMC) またはフォーカル・ポイント編成を使用するネットワークの場合は、その CMC またはフォーカル・ポイント・ホストで、新しいリリースの NetView プログラム (V2R4 以降) へのマイグレーションを開始してください。その後、そのマイグレーションをネットワークを通して他のホストに分散します。

クロスドメイン通信のための RMTCMD コマンドと LU6.2 セッションの使用

RMTCMD コマンドおよび LU 6.2 セッションを使用するように、古い NetView ノードを徐々にマイグレーションすることができます。

複数 CMC または複数フォーカル・ポイントを持つ企業の場合は、拡張複数コンソール・サポート・コンソールの使用に向けてこれらのノードをマイグレーションする前に、すべての CMC またはフォーカル・ポイントを更新して、RMTCMD コマンドおよび LU 6.2 セッションを使用してください。また、ネットワークで分散自動操作を使用している場合は、EMCS コンソールの使用に向けてそのプログラムをマイグレーションする前に、メッセージ交換を行うすべての NetView プログラムを更新して、RMTCMD コマンドおよび LU 6.2 セッションを使用してください。強調表示や DOM 情報などの MDB データの消失を防ぐため、いずれの場合も、できる限り RMTCMD コマンドおよび LU 6.2 セッションへのマイグレーションを完了してから、拡張複数コンソール・サポート・コンソールを使用するようにしてください。

MVS VARY コマンドへのオペレーター・アクセスの制限

特に制限がない限り、オペレーターは VARY コマンドを使用して、コンソールが受け取る経路コードなどの、EMCS コンソールの属性を変更することができます。この種の変更は、メッセージ配布の重複を引き起こす可能性があります。したがって、NetView コマンドの許可検査を使用するか、システム許可機能 (SAF) プロダク

トの VARY コマンドを保護して、NetView MVS コマンドの使用を制限してください。例えば、NetView MVS コマンドを保護するには、NetView コマンド許可テーブルまたはシステム許可機能 (SAF) 製品 (RACF (リソース・アクセス管理機能) など) を使用します。コマンド許可については、「*IBM Tivoli NetView for z/OS* セキュリティー・リファレンス」を参照してください。

EMCS コンソールの AUTO 属性の制限

AUTO 属性を指定すると、コンソールは MPF または MRT ステートメント ('Y' AUTOMATE) を使用して自動操作のマークを付けられたメッセージを受信します。どの NetView コンソールにも AUTO 属性が指定されないようにすると、大量のメッセージ・トラフィックを回避できます。

第 7 章 MVS シスプレックスでの自動操作

この章では MVS シスプレックスについて紹介し、その利点のいくつか、シスプレックスでの自動化に関する提案事項、およびシスプレックスでの自動化の計画方法について説明します。

MVS のシスプレックス

MVS シスプレックス とは、各種の機能とプログラムを共用することで単一のシステムとして働く、複数 MVS オペレーティング・システムの構成のことをいいます。これらの複数 MVS システムを、単一のシスプレックスとして動作させるための MVS コンポーネントは、システム間カップリング・ファシリティー (XCF) です。XCF は、ある MVS システムで実行中の許可プログラムが、同じ MVS システムまたは他の MVS システム上のプログラムと対話できるように、あるいはデータの交換を行えるように、結合サービスを提供します。XCF の主な目的は、複数の MVS システムを 1 つのシステムのように使用することです。

XCF では、複数システム環境は、1 つ以上のプロセッサに、複数の MVS システムが常駐している環境と定義されます。プロセッサが複数ある場合、それらのプロセッサは次の条件を満たしていなければなりません。

- 1 つ以上のチャンネル間 (CTC) 接続または 1 つ以上のカップリング・ファシリティーによって相互接続されている。
- 外部時刻参照 (ETR) を使用している。
- XCF 結合データ・セットを共用している。

シスプレックス内の 1 つ以上の結合 MVS システムの集合に対して、XCF シスプレックス名が与えられると、システム内の許可プログラムは XCF の結合サービスを利用できるようになります。XCF はそのシスプレックス内のシステムをモニターし、最小限のオペレーター介入によって、障害の発生したシステムをシスプレックスから除外することができます。

シスプレックスでは、ある MVS システムのコンソールから、同じシスプレックス内の別のシステムへメッセージを経路指定することができます。またコマンドについても、ある MVS システムのコンソールから、同じシスプレックス内の別のシステムへ経路指定を行うことができます。

1 つのシスプレックス内で使用するコンソール名は、すべて固有のものでなければなりません。

シスプレックスの管理計画については、MVS ライブラリーを参照してください。

シスプレックスでの NetView プログラム自動操作の使用

シスプレックスにおいて、NetView 自動操作を使用することの利点としてまず挙げられるのは、NetView プログラムが、同じシスプレックスのすべてのメンバーからのメッセージを受信し、シスプレックス内の適切なメンバーに対し自動応答を返すことができるという点です。

シスプレックスにおいて、NetView プログラム自動操作を使用する方法を次に示します。

- 同一シスプレックス内の別のシステムへのメッセージを処理する際は、NetView プログラム自動化テーブルの別のセグメントを使用するように工夫してください。既存の自動化テーブル・ステートメントに **SYSID** 条件項目を追加すれば、特定のシステムからのメッセージをブロック化することができます。あるいは、システム ID に基づいて、自動化テーブルの特定のアクションを呼び出すことができます。
- コンソールに固有の名前を付けるためには、コンソールの命名方法を工夫するか、または NetView プログラムの **ConsMask** 機能を使用してください。デフォルトでは、コンソール名にオペレーター名と同じ名前が付けられるようになっています。GETCONID コマンドの詳細については、NetView プログラム・オンライン・ヘルプを参照してください。ConsMask 機能の詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

シスプレックスでの自動操作の計画

シスプレックスにおける自動操作を計画するためには、シスプレックス管理の計画に精通していることが条件となります。シスプレックスの管理については、MVS ライブラリーを参照してください。

メッセージ改訂テーブル (MRT) を使用して、メッセージ処理機能 (MPF) によって提供される機能を実行することもできます。メッセージ改訂テーブルについての追加情報は、30 ページの『メッセージ改訂テーブル』に記載されています。

シスプレックスでは複数の MVS システム間で整合のとれた対話が必要になるので、自動操作の計画も複雑なプロセスをたどることになります。計画のプロセスをできるだけ容易なものにするために、本セクションではこれをいくつかの段階に分けて説明します。各段階では、計画の際に考慮する必要がある情報を、実際の順序どおりに提示しています。提示されたすべての情報について、十分な検討を行ってください。

第 1 段階 EMCS コンソールと属性のメッセージ経路指定への影響

73 ページの『第 6 章 MVS 拡張複数コンソール・サポート・コンソールを使用した自動操作』で述べた、拡張複数コンソール・サポート・コンソールについての情報を検討します。次に、以下の項目に留意してください。

- コンソール名はそのシスプレックスにおいて固有のものでなければなりません。
- MSCOPE コンソール属性の値は、コンソールがメッセージを受け取る 1 つ以上の MVS システムを決定します。特に NetView プログラムのデフォルトを使用しない場合には、シスプレックスのこれらの MSCOPE 値を十分に考慮してくだ

さい。ユーザーは任意のコンソールに対し、以下のシステムからのメッセージを受け取るように MSCOPE 値を設定することが可能です。

- そのコンソールが実行中のシステム
- そのシスプレックス内のすべてのシステム
- シスプレックス内のシステム・ソフト
- CMDSYS コンソール属性は、シスプレックス内のどのシステムが MVS コマンドを処理するかを決定します。CMDSYS 属性によって、NetView オペレーターは、そのオペレーターの MVS コマンドをすべて、シスプレックスの特定のシステムに自動的に送信することができます。シスプレックス内の各コンソールにどのような機能があるかを考慮してください。また、次の点についても考慮してみてください。
 - CMDSYS のデフォルト設定は、ローカル MVS システムです。つまり NetView プログラムが稼働しているシステムです。
 - ある NetView プログラムのオペレーターが、シスプレックス内の別の特定のメンバーだけにコマンドを出すこともできます。その場合、CMDSYS 属性はそのシステムに設定しなければなりません。

第 2 段階MPF のアクションと EMCS コンソールの定義の調整

MVS メッセージへの自動応答はメッセージ処理機能 (MPF) テーブルの影響を受けるので、MPF 内で実行されるアクションと、拡張複数コンソール・サポート・コンソール用に予定している定義との調整を図ります。例えば、AUTO(YES) キーワードでマーク付けされたメッセージをどのコンソールで受け取るか、また MSCOPE にどのような値を使用するかを決定する必要があります。以下の情報を考慮してください。

- 各メッセージはただ 1 つの MPF テーブルにより処理されます。MPF テーブルは、そのメッセージの発信元であるシステムの中にあります。ただし、NetView プログラム自動化テーブルなど、シスプレックス内の他のシステムの機能によって、そのメッセージを処理することは可能です。
- シスプレックス内のすべてのシステムで MPF テーブルを同じように定義する必要はありません。しかし、保守を容易にしたい場合や、リカバリー・アクションの実行中にワークロードをあるシステムから別のシステムに移動できた方がよい場合もあります。
- MPF テーブルをシステムごとに定義し、そのシステムで生成されたすべての MVS メッセージが処理できるようにします。メッセージがシスプレックス内の他のシステムで受ける、追加処理または追加自動化の影響を理解し、予期します。

第 3 段階NetView プログラム自動操作をシスプレックスの 1 つのシステムに集中させるかどうかの決定

通常、自動操作はできるだけソースに近いところで提供するのが最も効果的ですが、システム自動操作を集中化することも可能です。JES3 を実行する場合は、JES3 環境で自動操作を集中化する場合の考慮事項が記載されている 555 ページの『第 36 章 ジョブ入力サブシステム 3 (JES3) の自動化』を参照してください。

拡張コンソールに MSCOPE 値を設定すると、シスプレックス内の 1 つのシステムですべてのシステム・メッセージを受け取り、自動応答を返すことができます。拡

張 MCS コンソールに MSCOPE 属性を設定するには、RACF (または、これと同等のシステム許可機能) の OPERPARM セグメント、または MVS VARY コマンドを使用してください。

自動操作アクションを適切なシステムに返すために、自動操作アクション、コマンド・リスト、およびコマンド・プロセッサは、MVS ROUTE コマンドを使用して経路指定を行わなければなりません。そのメッセージがシスプレックス内のどのシステムで出されたかは、自動化テーブルの SYSID 条件項目をチェックするとわかります。REXX および NetView コマンド・リスト言語で書かれたコマンド・リスト、またはコマンド・プロセッサについても、SYSID 情報の利用が可能です。SYSID の使用法についての情報は、「IBM Tivoli NetView for z/OS プログラミング: REXX および NetView コマンド・リスト言語」を参照してください。

メッセージのトラフィックが極端に多い場合は、システム自動操作の集中化が適切でない場合もあります。集中化されたシステム自動操作を使用する場合に、受け入れ可能なパフォーマンスで、どの程度のトラフィックを処理できるかについての基準は確立されていません。

外部メッセージの処理方法

外部メッセージ は、シスプレックス内のローカル・システムとは異なるシステムで発信されたメッセージとして定義されます。外部メッセージがどのように処理されるかを制御する方法は、さまざまです。ローカル・システムによって受信された外部メッセージは、.FORNSSI MPFLSTxx ステートメントによって SSI 出口が外部メッセージを受信できない場合を除き、まずアクティブな SSI 出口を通過します。(詳細については、「z/OS MVS 初期設定およびチューニング解説書 (SA88-8564)」の .FORNSSI ステートメントの説明を参照してください。) メッセージ改訂テーブル (MRT) の処理は SSI 出口の処理時に実行されるため、このときに外部メッセージに対するアクションを取ることができます (このトピックの詳細については、139 ページの『第 13 章 メッセージ改訂テーブル』を参照してください)。MVS は、その後、外部メッセージを受信するコンソールを決定します。NetView システム・オペレーターが所有するコンソールがメッセージの宛先である場合は、メッセージがそこに送信され、メッセージは送信請求メッセージであるとみなされます。メッセージの宛先が NetView システム・オペレーターでない場合は、メッセージが非送信請求であるとみなされ、以下の説明が当てはまります。

- NetView プログラム SSI は、メッセージのコピーを NetView アドレス・スペースに転送するかどうかを判別します。デフォルトでは、外部メッセージは NetView プログラム・アドレス・スペースに転送されません。これは、MRT で AUTOMATE=Y を設定することによりオーバーライドできます。
- メッセージが自動化可能とマークされていれば、CNMCSSIR タスクはそのメッセージを受け取ります。
- メッセージの再改訂または再自動化の可能性を低減するため、MRT 処理およびメッセージ自動化は、できる限りメッセージの発信元の近くで実行してください。
- 非送信請求の外部メッセージが NetView プログラム・アドレス・スペースによって受信されると、自動化はこのメッセージに対して通常どおり実行されます。

第 8 章 リソース・オブジェクト・データ・マネージャーでの自動操作

この章では、リソース・オブジェクト・データ・マネージャー (RODM) を紹介し、RODM を自動操作で使用する際の利点および暗黙の了解について説明するとともに、自動操作を計画する上で特別に考慮すべき点について述べます。

NetView 製品が、RODM およびそのデータ・モデルの記述に使用するオブジェクト指向の用語については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

リソース・オブジェクト・データ・マネージャーの紹介

RODM は、高速ストレージに保管されているデータ・キャッシュです。RODM は、ネットワーク・リソースとシステム・リソースに関するさまざまなタイプの情報を保持することができます。RODM はこれらの情報を高速ストレージに保持しているため、NetView プログラムは、これらの情報を、ほかのタイプのストレージに保持されている情報よりも迅速に検索および更新することができます。

NetView プログラムは、RODM と RODM 内のリソース情報を使用して、ネットワークおよびシステムの自動操作を補助します。

RODM との対話

RODM はメソッドと呼ばれるプログラムを使用して、RODM 内の情報の検索、更新、操作にかかわる多くの機能を実行することができます。RODM に対してユーザー独自のメソッドを作成し、そのメソッドを呼び出すことによって、ユーザーは、RODM においてユーザー定義の特別な機能を行うことができます。メソッドの作成には、PL/I または C 言語のいずれを使用しても構いません。これらのメソッドは、アプリケーション・プログラム (NetView プログラム・コマンド・プロセッサなど) から直接呼び出すことも、RODM のフィールド (オブジェクトの状況など) が変わるときに、自動的に起動することもできます。

NetView プログラムを使用していない RODM のユーザーは、RODM が提供するアプリケーション・プログラム・インターフェースを介して、RODM と対話することができます。アプリケーション・プログラムは API を通して、RODM 内のリソース情報を検索および更新、ならびに RODM メソッドの呼び出しを行うことができます。ユーザーは、PL/I、C、またはアセンブラー言語を使用して、RODM 用にアプリケーション・プログラムを作成することができます。

マルチシステム・マネージャー RODM アクセス機能は、高速かつ効率的な RODM への REXX プログラム・インターフェースを提供します。これによって、RODM でのオブジェクトの作成、更新、照会、および削除を行うことができます。

NetView プログラム・コマンド・プロセッサは、RODM からの値の入手、RODM 内の情報の変更、および RODM メソッドの呼び出しなどの操作を行うことができます。ユーザーは、PL/I、C、またはアセンブラー言語を使用して、NetView プログ

ラム・コマンド・プロセッサを作成することができます。 REXX プログラム、NetView プログラム自動化テーブル・ステートメント、および (ある程度) コマンド・リストは、ORCONV または FLCARODM コマンドを使用して、RODM 内のメソッドの呼び出しおよび情報の変更を行うことができます。

メソッド EKGSPPI および ORCONV コマンドの使用例については、461 ページの『第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作』を参照してください。

EKGSPPI オブジェクト独立メソッドを使用すれば、RODM のメソッドは、あらゆる NetView プログラム・コマンド・リスト、またはコマンド・プロシーチャーを呼び出すことができます。EKGSPPI の使用例については、461 ページの『第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作』を参照してください。

自動操作における RODM の使用

RODM の使用例の 1 つとして、RODM 用のアプリケーション・プログラムは、リソース状況の変更などの外部イベントに対し、RODM 内の関連するリソース情報を更新させることができます。この更新によって、特定の RODM メソッドが開始します。次にそのメソッドは、事前定義されたアルゴリズムに従って、更新された情報を RODM 内の他の情報と比較して、適切な応答を返します。このように、RODM ではストレージ内のリソース情報を維持するとともに、API およびいくつかのメソッドを通して情報への迅速なアクセスを提供しており、ネットワークまたはシステムのさまざまなイベントに対して、適切な自動応答を判断する上で役に立ちます。

また、自動操作の影響を受けるリソースごとに、『自動化進行』インディケーターが RODM 内に保持されます。このインディケーターにより、オペレーターは、NetView プログラム管理コンソールを使用して、そのリソースを調べ、自動化が完了するのを待ってから、リソースの問題を修正することができます。

RODM の詳細および RODM アプリケーション・プログラムとメソッドの作成については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

RODM を使用する際の利点

RODM は、状況情報、ヒストリー情報、構成情報など、リソースに関するさまざまなタイプの情報を受信および保持することができます。このように大量かつ多様な情報を保持できるということは、すなわち、リソース問題の原因分析および解決を図る上で、より多くのデータが利用可能であることを意味します。RODM はストレージ内に情報を保持するので、このリソース情報を速やかに更新および検索することが可能です。

RODM は情報をオブジェクトおよびオブジェクトの集合として保持し、プログラム定義の関係に従って、オブジェクトを関連付けすることができます。RODM では個々の情報の関係を特定することができるので、イベント間の対話を決定し、この情報を問題分析に利用することが可能です。

RODM メソッドを使用して、ネットワーク・イベントおよびシステム・イベントに対して、自動応答を提供することができます。RODM メソッドは、NetView プログラム・ルーチンを開始させることができ、NetView プログラム・ルーチン (自動化テーブル・ステートメントを含む) は、RODM メソッドを開始させることができます。

注: NetView プログラムから、RODM 情報の検索および更新を行う際は、NetView プログラム・コマンド・リスト、NetView プログラム・コマンド・プロセッサ、または NetView プログラム自動化テーブル・ステートメントを作成する必要があります。

自動操作における RODM の使用計画

このセクションでは、ネットワークおよびシステムの NetView プログラムによる自動化における RODM の使用方法を計画するにあたって、考慮すべき項目について説明します。詳しくは、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

RODM で自動応答を生成するイベント・タイプの決定

RODM は、さまざまなタイプのネットワークおよびシステム・イベントに対して、自動応答を生成することができます。ただし、イベントの中には、自動化テーブルだけを使用して自動応答を生成した方がよいものと、自動化テーブルと RODM を組み合わせて自動応答を生成した方がよいものがあります。ユーザーは、各種のイベントへの応答に最も適したメソッドと構成要素 (1 つ以上) を決める必要があります。

例えば、自動化テーブルは、単純かつ速やかな対処を必要とするイベントに対して自動応答する場合に適しています。これは、そのような自動操作タスクについては自動化テーブルの方が RODM よりも迅速に対処でき、コーディングも単純だからです。一方 RODM は、複数のメッセージまたはアラートに起因する複雑なイベントに対して自動応答する場合に適しています。また、次のイベントに対して自動応答を行う場合も、RODM の方が適しています。

- 適切な応答を判断する際に、自動化テーブルで通常利用できる情報よりも多く情報を必要とするもの
- 応答を出す前に条件分析を必要とするもの
- 既存の RODM メソッドまたは RODM アプリケーション・プログラムのアルゴリズムを有効利用できるもの

RODM 自動操作機能について

自動化で RODM を使用する前に、自動化に RODM 機能を使用するイベント (シナリオ) の概要を検討してください。RODM シナリオの例については、461 ページの『第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作』を参照してください。

第 9 章 自動化のための NetView プログラム情報の経路指定

この章では、自動操作の情報を経路指定する方法について説明します。この経路指定の詳細には、以下のトピックが含まれます。

- NetView プログラム・インターフェース
- NetView メッセージ経路指定
- NetView ハードウェア・モニター・データ、および MSU の経路指定
- NetView プログラム・コマンドの経路指定

多くの場合、NetView プログラムのメッセージおよびデータの流れは複雑です。ただし、情報のおおまかな経路を理解しておくことで、自動化の対象になるメッセージおよび MSU が自動化機能へ渡される様子を確認できます。

例えば、自動化テーブルを使用して、VTAM プログラムの通信ネットワーク管理 (CNM) インターフェースから送られるネットワーク管理ベクトル・トランスポート (NMVT) に対する自動化応答を生成させることができます。そのためには、NMVT から自動化テーブルへの流れを妨げるものがない状態を維持してください。

経路指定情報を使用して、2 つの NetView 機能が、1 つのメッセージまたは MSU に相反する属性を指定した場合のアクションを決定することもできます。例えば、MSU はハードウェア・モニターに渡され、次に自動化テーブルに渡され、さらに DSIEX16B インストール・システム出口に渡されることがあります。最後に DSIEX16B 出口でデータが処理されることがわかっているならば、その出口で設定した属性が優先順位を持つことを判断するのに役立ちます。

NetView プログラム・インターフェース

NetView プログラムの自動化により、メッセージ、MSU、およびその他のデータをさまざまなソースから受信することで、データ処理のイベントを監視することが可能になりました。同様に、NetView プログラムから、多くの異なるターゲットおよび宛先に対してコマンドを出すことができます。92 ページの図 8 に、イベント通知の受信や、コマンドの発行のために通常使用されるインターフェースを示します。オペレーティング・システムのインターフェースについて示した図は、25 ページの『第 2 章 自動操作プロダクトの概要』を参照してください。

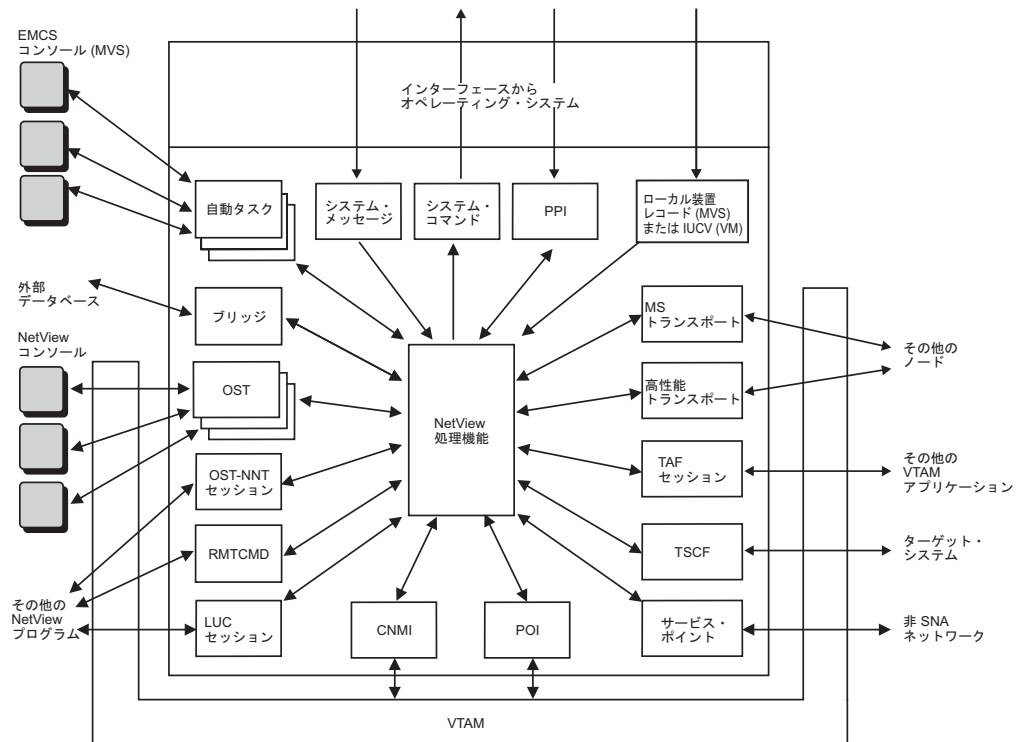


図 8. 自動化で使用する NetView インターフェース

オペレーティング・システムとのインターフェース

システム自動化を実施するには、まず、オペレーティング・システム、サブシステム、およびアプリケーションの状態について記述した情報を、NetView プログラムでアクセスすることから始めます。また、NetView プログラムがシステムへコマンドを送信し、コマンドの応答を受信するようにセットアップすることもできます。サブシステム・インターフェース、または拡張複数コンソール・サポート (EMCS) コンソールを使用して、コマンドの入力および応答の受信を行います。それ以外にも、システム自動化に関しては、次のインターフェースを使用できます。

- システム自動化支援プログラム (OS/390) のプロセッサ・オペレーションは、システム・コンソール上のトラフィックを代行受信できます。
- NetView プログラム端末アクセス機能 (TAF) で、他のアプリケーション (システム・アプリケーションも含む) から、それら独自のコンソールに対して出されたメッセージを代行受信できます。
- MVS のローカル装置では、特定のタイプのシステム問題通知を NetView プログラムに渡して処理することができます。
- MS トランスポートおよび MS トランスポートの高性能オプションを使用すると、2 つのアプリケーション間で LU 6.2 通信が可能になります。トランスポートの使用法の 1 つとして、システム・アプリケーションと NetView プログラム間で情報を渡せます。
- プログラム間インターフェースで、同一システム上の NetView プログラムとともに実行しているシステム・アプリケーションから、MSU を受け入れ、それらを NetView プログラムハードウェア・モニターに渡せます。

システムの自動化を試みている場合、自動化したいメッセージおよびその他の情報が、確実に NetView プログラムに送られるようにする必要があります。

システムの自動化におけるオペレーティング・システムと NetView プログラムの関係の概要に関しては、25 ページの『第 2 章 自動操作プロダクトの概要』を参照してください。

システム通信のセットアップに関する段階別の情報が必要であれば、333 ページの『NetView システムとオペレーティング・システムの通信の確立』を参照してください。

さらに詳しいシステムの流れの情報を必要とする場合は、589 ページの『付録 D. MVS メッセージおよびコマンド処理』を参照してください。

その他の NetView プログラムとのインターフェース

以下のインターフェースを使用すると、NetView プログラムが稼働している 2 つのシステムの間で情報を交換できます。

- RMTCMD コマンド。このコマンドは、他の NetView ヘルプコマンドを送信し、その応答として生成されるメッセージを受信するために使用できます。
- OST-NNT セッション。この方法でも、NetView 間で、メッセージおよびコマンドを送信できます。
- LUC セッション。アラートおよび状況情報をフォーカル・ポイントに転送するために使用します。

NetView プログラム間の通信には、TAF セッション、MS トランスポート、および MS トランスポートの高性能オプションを使用することもできます。NetView プログラム間の通信方法については、423 ページの『第 26 章 集中操作』を参照してください。

その他のメッセージおよびコマンド機能

メッセージの受信やコマンドの送信に使用する NetView プログラム機能には、これ以外にも次のものがあります。

- VTAM メッセージに使用するプログラム・オペレーター・インターフェース (POI)。
- ソフトウェア情報/管理およびその他の拡張データベースに使用する NetView ブリッジ。

ハードウェア・モニター・データおよび MSU 用のインターフェース

NetView プログラムを使用すると、MSU を直接自動化できます。MSU 以外のハードウェア・モニター・データを、最初にメッセージに変換することによって自動化することもできます。ハードウェア・モニター・データおよび MSU は、以下のソースから、NetView プログラムに渡されます。

- 通信ネットワーク管理インターフェース (CNMI)。SNA ネットワークから問題レコードとして渡されます。
- サービス・ポイント。非 SNA ソースから NMVT として渡されます。

- ローカル装置。オペレーティング・システムからの問題記録として渡されます。
- プログラム間インターフェース。同一システムの NetView プログラムで実行している別のアプリケーションから、MSU として渡されます。
- LU 6.2 トランスポート。LU 6.2 アプリケーションから MSU として渡されます。

NetView プログラム・メッセージ経路指定

NetView プログラムがメッセージを受信すると、NetView プログラム経路指定機能により、NetView プログラム内のメッセージの宛先が制御されます。経路指定機能を使用すると、そのメッセージを見るオペレーター、またはそのメッセージを処理する自動タスクを選択できます。メッセージ経路指定を効果的に制御するには、送信請求メッセージと非送信請求メッセージとの違いについて理解しなければなりません。さらに、主要な経路指定機能と、NetView プログラム内におけるメッセージの経路についても、精通していなければなりません。メッセージ経路の詳細については、605 ページの『付録 F. NetView メッセージおよびコマンドの詳細な流れ』を参照してください。

NetView プログラムは、特定の宛先を認識しているメッセージを**送信請求**メッセージと見なします。NetView プログラムは、それ以外のメッセージは、**非送信請求**メッセージと見なします。

送信請求メッセージ

NetView プログラムは、既存の宛先タスク (NetView オペレーター、自動タスク、または NetView プログラム間タスク (NNT)) に対する送信請求メッセージをキューに入れます。次に送信請求メッセージの例を示します。

- NetView コマンドへの応答。
- EMCS コンソールを獲得した NetView オペレーター端末タスク (OST) に直接送られる MVS システム・メッセージ
- NetView オペレーター・コンソールから出されるシステム・コマンドへの応答。送信請求メッセージがシステム・コマンドに応じて戻される方法の詳細については、592 ページの『コマンドの流れ』を参照してください。
- VTAM コマンドへの応答。
- CICS、情報管理システム (IMS) などのアプリケーションとの、端末アクセス機能 (TAF) オペレーター制御セッションから出されるメッセージ (TAF セッションで、それらのアプリケーションから受信するすべてのメッセージを含む)。
- NetView REXX コマンド・リストに、TRACE または SAY 命令を指定した結果出されるメッセージ。
- NetView MSG コマンド、または MVS および TSO の SEND コマンドにより出されるメッセージ。

MVS から出されるシステム・メッセージは、送信請求メッセージの場合も非送信請求メッセージの場合もあります。非送信請求 MVS システム・メッセージの詳細については、95 ページの『MVS システムからの非送信請求メッセージ』を参照してください。

非送信請求メッセージ

メッセージは、特定の宛先タスクが通知されていないと送信されません。例えば、VTAM から 1 次 POI を通して、NetView プログラムに送信されたメッセージが、その NetView プログラムからの要求に関連付けられていない場合があります。NetView プログラムでは、1 次 POI タスク (PPT) に対して指示されたメッセージを非送信請求メッセージと見なします。これは、PPT がメッセージを表示できないためです。サブシステム・インターフェースを通して送られた PPT から出されるコマンドへの応答としての MVS メッセージも、非送信請求メッセージと見なします。

注: ハードウェア・モニターが実行依頼するのは、自動化のための非送信請求 MSU のみです。

許可レシーバー

非送信請求メッセージには特定の宛先がないため、宛先を指定するために ASSIGN コマンドまたは自動化テーブルを使う場合を除いて、すべての非送信請求メッセージは、NetView プログラムにより、許可レシーバーに経路指定されます。**許可レシーバー** とは、単純に、それ以外に宛先を持たない非送信請求の許可メッセージを受け取る許可が与えられた、NetView オペレーターのことを指します。

オペレーター・プロファイルで AUTH ステートメントを使用して、特定のオペレーターの権限を決定してください。プロファイルに AUTH MSGRECVR=YES が指定されているオペレーターは、許可レシーバーになれます。ただし、NetView プログラムでは、一度に 1 つの許可レシーバーしか持てません。

DST からの非送信請求メッセージ

データ・サービス・タスク (DST) から送られてくる非送信請求メッセージは、(アクティブであれば) 許可レシーバーではなく、DST を開始したタスクに送られます。ASSIGN コマンドの場合は、非送信請求 DST メッセージの経路指定に影響を与える可能性はありませんが、自動化テーブルの場合は影響を与える可能性があります。

MVS システムからの非送信請求メッセージ

MVS システムから受信した非送信請求メッセージは、許可レシーバーには送られません。ASSIGN コマンドを使用して、これらのメッセージをほかのタスクに再度経路指定できます。ASSIGN コマンドの詳細については、99 ページの『ASSIGN を使用した送信請求メッセージの経路指定』を参照してください。

ASSIGN コマンドを使用しない場合、CNMCSSIR タスクが、各非送信請求メッセージごとに自動化テーブルをスキャンします。スキャンの結果として、自動化テーブル内で一致が起きる可能性があります。

一致するステートメントに指定されているアクションの 1 つが、コマンド実行である場合、そのコマンドがログオン・タスクに経路指定されていないと、自動化テーブル・ステートメントに指定されたそのコマンドは無視され、Invalid cmd: に関連付けられた DWO050E メッセージがネットログに追加されます。

オペレーターが ROUTE アクションで自動化テーブルに指定されている場合、コマンドは 1 次自動タスク (function.autotask.primary スタイル・ステートメントで定義されたタスク) ではなく、そのオペレーターに送られます。

CNMCSSIR タスクで受信したすべての MVS システム・メッセージは、非送信請求メッセージです。その他の NetView オペレーター・タスクで受信したシステム・メッセージは、送信請求メッセージです。

メッセージ経路指定機能

NetView プログラムにおけるメッセージ経路指定の制御は、インストール・システム出口、自動化テーブル、ASSIGN コマンド、MSGROUTE コマンド、およびパイプ ROUTE コマンドを使用して行えます。EMCS コンソールを持つ MVS システムの場合、メッセージ経路指定の制御に EMCS コンソール属性も設定できます。例えば、メッセージ経路コードに基づいてメッセージを EMCS コンソールに経路指定できます。その経路コードを持つ MVS システム・メッセージは、EMCS コンソールを備えた NetView プログラム・タスクに直接送られます。

163 ページの『第 15 章 自動化テーブル』および 329 ページの『第 17 章 インストール・システム出口』では、それぞれ自動化テーブルおよびインストール・システム出口について説明します。以下のセクションでは、ASSIGN コマンド、MSGROUTE コマンド、および経路コードに基づくメッセージの経路指定について説明します。

ASSIGN コマンドによる経路指定メッセージ

以下のセクションでは、ASSIGN コマンドの使用方法について説明します。ほとんどのメッセージの経路指定では、ASSIGN コマンドではなく、自動化テーブルを使用してください (101 ページの『ASSIGN コマンドと自動化テーブルの経路指定の比較』を参照してください)。ただし、コンソールをグループに割り当てたり、メッセージを自動タスクに経路指定して自動化の速度を上げるような場合には、ASSIGN コマンドが便利です。

EMCS コンソールに直接送られて、NetView プログラム OST に使用される MVS システム・メッセージは、送信請求メッセージであると見なされます。したがって、ASSIGN PRI および ASSIGN SEC では処理されません。1 つ例外があります。すなわち、PPT が EMCS コンソールを持つ場合、PPT に送られる送信請求メッセージは、ASSIGN PRI および ASSIGN SEC で処理できます。CNMCSSIR であるタスクで取得される EMCS コンソールに送られる MVS システム・メッセージは、非送信請求メッセージであると見なされます。

メッセージのオペレーターへの割り当て

MSG オプションを使用すると、送信請求メッセージ、非送信請求メッセージ、および許可メッセージを以下の宛先に送ることができます。

- 特定のオペレーター
- オペレーター・グループ
- システム・オペレーター (SYSOP)
- ネットワーク・ログ (LOG)

ASSIGN コマンドを使用すると、オペレーターは自動化テーブルを編集および再ロードしなくてもメッセージを変更することができます。

オペレーターのグループへの割り当て

GROUP オプションを使用すると、オペレーターのリストを特定のグループに割り当てられます。そうすると、別の ASSIGN コマンド、コマンド・リスト内の MSGROUTE コマンド、および自動化テーブル内での EXEC(ROUTE) アクションにそのオペレーター・グループを使用できます。

メッセージをグループに送るために、自動化テーブル内に ROUTE(+ グループ名) を指定すると、そのグループの内容を変更することで、メッセージを受信するオペレーターのリストを変更できます。特定のグループに属するオペレーターのリストを修正する必要がある場合は、GROUP オプションを指定して ASSIGN コマンドを出します。

注: 割り当ての変更をモニターするのが困難なため、ASSIGN オプションを設定する場合には、オペレーターに、GROUP オプションのみを使用する許可を与えることを考慮してください。NetView の LIST コマンドを使用すると、指定した任意の時点で割り当てられるものをモニターできます。

詳細については、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

ASSIGN を使用した非送信請求メッセージの経路指定

ASSIGN コマンドに PRI オプションを指定すると、非送信請求メッセージ、または許可メッセージを受け取るオペレーターのリストを指定できます。以下を指定できます。

- オペレーター
- 自動タスク
- オペレーターおよび自動タスクのリスト
- グループ ID
- システム・オペレーター (SYSOP)
- ネットワーク・ログ (LOG)

各メッセージは、それぞれ 1 人のオペレーターのみが受信します。リストまたはグループ ID を指定すると、ログオンしているリストまたはグループの最初のオペレーターだけがそのメッセージを受け取ります。

1 次受信先に送られるメッセージの、DOMAINID フィールドの最後の位置には、パーセント記号 (%) のフラグが付けられます。NetView プログラムは、画面上にメッセージとともにそのパーセント記号 (%) を表示し、また、パーセント記号をネットワーク・ログにも記録します。パーセント記号は、BUFHDR の HDRDOMID フィールドには記録されません。

メッセージが 1 次コピーであるかどうかを決定する必要があるインストール・システム出口では、内部機能要求の IFRAUPRI および IFRAUSEC フィールドをチェックしなければなりません。

PRI=SYSOP または PRI=LOG が指定されたメッセージに ASSIGN コマンドを出す場合、そのメッセージは、NetView プログラム自動化テーブルでは処理されません。

ASSIGN コマンドに SEC オプションを指定すると、非送信請求メッセージまたは許可メッセージの 2 次コピーを受信するオペレーターのリストを指定できます。SEC コピーを生成するには、まず、メッセージに PRI 割り当てを行う必要があります。

SEC リスト内のすべてのオペレーター、またはオペレーターのグループは、次の場合にメッセージを受信します。

- ログオンしている場合
- PRI リストのオペレーターの少なくとも 1 人がログオンしている場合

SEC の受信先に送られるメッセージの、DOMAINID フィールドの最後の位置には、アスタリスク (*) のフラグが付けられます。NetView プログラムは、メッセージとともにそのアスタリスクを表示し、また、そのアスタリスクをネットワーク・ログにも記録します。アスタリスクは、BUFHDR の HDRDOMID フィールドには記録されません。

インストール・システム出口は、メッセージが 2 次コピーであるかどうかを判断するために、自動化内部機能要求 (AIFR) の IFRAUSEC フィールドをチェックすることができます。

どの 1 次受信先もログオンしていない場合、NetView プログラムは、割り当ての指定が行われなかったかのように続行します。メッセージの経路指定に変更はなく、メッセージの 2 次コピーは 2 次受信先には送られません。PRI リストに複数のオペレーターを指定するか、または 1 次受信先の 1 つとして永続的な自動タスクを指定すると、メッセージの割り当てが確実に行われ、2 次コピーが確実に 2 次受信先に渡されるようになります。

2 次コピーには、以下の事項が適用されます。

- コピーが別の NetView プログラム・オペレーターにクロスドメインで経路指定されていない場合、これらのコピーは自動化テーブル処理の対象にはなりません。クロスドメインで送信された 2 次コピーは、クロスドメイン NetView プログラムでは、自動化テーブル処理の対象となります。
- これらのコピーは、コマンド・プロシーチャーの WAIT 処理の対象になります。
- 2 次コピーは、メッセージを複数のオペレーターに表示する場合に役立ちます。

ASSIGN コマンドを使用して、非送信請求メッセージを経路指定できます。図 9 に示しているコマンドは、PRI オプションで指定されたログオン済みの最初のオペレーターに、すべての非送信請求メッセージを経路指定します。

```
ASSIGN MSG=*,PRI=(OPER1,AUT01),SEC=(NETOP1,LOG)
```

図 9. ASSIGN コマンドを使用した非送信請求メッセージの経路指定

NetView プログラムは、DSIEX04 インストール・システム出口で特に指定していない限り、各コピーをネットワーク・ログに記録します。上記の例では、オペレータ

一の SEC リストに LOG が指定されているため、OPER1、AUTO1、および NETOP1 でログの抑止が指定されていないと、二重にログが記録されます。

ASSIGN を使用した非送信請求メッセージの除去

ASSIGN コマンドの DROP オプションは、MSG オプションまたは GROUP オプションとともに指定することもできます。MSG オプションとともに指定すると、指定されたメッセージは DROP=AUTH により PRI および SEC 割り当てから除去されます。例えば、図 10 に示すようにコマンドを入力した場合、システムはすべての割り当てを除去するわけではありません。この場合、MSG=* を使用して作成した割り当てが除去されます。(AUTH はデフォルト値であるため、指定する必要はありません。)

```
ASSIGN MSG=*,DROP=AUTH
```

図 10. ASSIGN コマンドを使用した非送信請求メッセージの除去

ASSIGN を使用した送信請求メッセージの経路指定

ASSIGN コマンドに COPY オプションを指定すると、非送信請求メッセージのコピーを受け取るオペレーターのリストを指定できます。以下を指定できます。

- オペレーター
- オペレーターのリスト
- グループ ID
- システム・オペレーター (SYSOP)
- ネットワーク・ログ (LOG)

送信請求メッセージのコピーは、コピー・リストに指定されている、ログオンしたすべての受信者に送られます。

コピーとして送られるメッセージの、DOMAINID フィールドの最後の位置には、正符号 (+) のフラグが付けられます。NetView プログラムは、画面上にメッセージとともにその正符号 (+) を表示し、また、その正符号をネットワーク・ログにも記録します。正符号は、BUFHDR の HDRDOMID フィールドには記録されません。インストール・システム出口は、内部機能要求の IFRAUCPY フィールドをチェックして、送信請求メッセージがコピーであるかどうかを判別することができます。

ASSIGN COPY オプションで生成されるコピーには、以下の項目が適用されます。

- コピーが別の NetView プログラム・オペレーターにクロスドメインで経路指定されていない場合、これらのコピーは自動化テーブル処理の対象にはなりません。これらのコピーは、クロスドメイン NetView プログラムでは、自動化テーブル処理の対象となります。
- これらのコピーは、コマンド・プロシーチャーの WAIT 処理の対象になります。

100 ページの図 11 に示している最初のコマンドは、NETOP1 と OPER1 の両方 (ログオン済みであれば) に、すべての送信請求メッセージのコピーを送ります。

ASSIGN コマンドを DROP=COPY を指定して出すと、指定のメッセージへの COPY 割り当ては除去されます。100 ページの図 11 では、2 番目のコマンドで、

ASSIGN MSG=*,DROP=COPY と入力すると、COPY 割り当てから、MSG=* で割り当てられたメッセージを除去します。

```
ASSIGN MSG=*,COPY=(NETOP1,OPER1)
```

図 11. ASSIGN コマンドを使用した送信請求メッセージの経路指定

ASSIGN を使用したメッセージの自動タスクへの経路指定

1 つのタスクのキューに入ったメッセージが多過ぎて、自動化テーブル処理の待ち時間が長くなり、自動化の速度が低下した場合には、ASSIGN コマンドを使用してメッセージをいくつかのタスク間に分割できます。このような場合であっても、メッセージの最後の経路指定には自動化テーブルを使用できます。

注: ASSIGN コマンドではメッセージを任意のタスクに経路指定できません。詳細は、240 ページの『処置』を参照してください。

自動化論理での ASSIGN の使用

ASSIGN コマンドの宛先の指定とは別途に、自動化論理を適用して、メッセージがそれぞれに割り当てられた宛先に経路指定されているかどうかを判別することができます。MEMBER オプションを指定して使用すれば、ASSIGN コマンドで、自動化テーブル・ステートメントを持つ、DSIPARM メンバーまたは PIPE メッセージ・データを指示することができます。これらのステートメントは、自動化テーブルにコンパイルされます。メッセージがこのテーブルをパススルーするときに、ASSIGN 経路指定基準を満たしているかどうか判別されます。ASSIGN コマンドの詳細については、「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」またはオンライン・ヘルプを参照してください。

REFRESH および ASSIGN コマンドを使用した動的オペレーター制御

REFRESH コマンドを使うと、NetView プログラムにオペレーターを事前定義しなくても、オペレーターの削除および追加を動的に行えます。ASSIGN コマンドは、現時点で NetView プログラムに対して定義されていないオペレーターに、メッセージを割り当てることができます。まだ NetView プログラムに定義されていないオペレーターにメッセージが割り当てられると、ASSIGN コマンドで指定されたオペレーターが、現在 NetView プログラムには定義されていないことを示すメッセージが出ます。その後、割り当ては正常に完了します。

定義されたオペレーターがログオンすると、そのオペレーターにメッセージが送られます。オペレーターが NetView プログラムに定義されているかどうかにかかわらず、ログオンしていないオペレーターに割り当てられたメッセージは、次に割り当てられたオペレーターか、元の宛先に送信されます。

REFRESH コマンドを使用してオペレーター定義を削除した場合、そのオペレーターがログオフするまでの間、オペレーター・セッションは続行します。ログオンされているが、NetView プログラムにはもはや定義されていないオペレーターに割り当てられたメッセージは、依然としてそのオペレーターに送られます。

ASSIGN コマンドと自動化テーブルの経路指定の比較

ASSIGN コマンドを使用して、送信請求メッセージおよび非送信請求メッセージを経路指定できます。ASSIGN は、オペレーターをグループへ割り当てる場合、自動化タスクへメッセージを事前に経路指定して自動化テーブルで得るメッセージを迅速に受け取る場合、さらに、メッセージをシステム・オペレーターに割り当てる場合に最適です。それ以外の場合は、通常、次の理由から、メッセージの経路指定には自動化テーブルを使用する方が適していると思われます。

- ASSIGN コマンドで経路指定を行うメッセージは、ASSIGN コマンドを出した順番には関係なく、特定メッセージから先に処理されます。図 12 に例を示します。

```
ASSIGN MSG=IST*,PRI=(VTAMOPER,AUT01)
ASSIGN MSG=IST5*,PRI=(VTAMOPER,AUT02)
```

図 12. 汎用メッセージおよび特定メッセージの経路指定

2 番目のコマンドに指定された経路指定から先に実行されます。これは、IST5*の方が IST* よりも対象が特定されるためです。3 つ目の ASSIGN コマンド(以下の例のような)が出され、最初の ASSIGN コマンドに指定されたメッセージの経路指定を取り消す場合であっても、2 つ目の ASSIGN コマンドの処理は行われます。

```
ASSIGN MSG=IST*,DROP
```

IST メッセージに出されたすべての ASSIGN コマンドの除去を試みる場合、オペレーターは、IST メッセージに対して出されたすべてのコマンドに加えて、2 つ目のコマンドについても認識していなければなりません。認識していれば、このオペレーターは、適切なコマンドを出して ASSIGN コマンドを除去できます。

複数の異なるオペレーター、コマンド・リスト、およびコマンド・プロセッサが ASSIGN コマンドを発行する場合、自分以外に何が割り当てられるかを必ずしも認識しているわけではありません。したがって、ASSIGN コマンドで送られるメッセージをモニターするのは困難な場合があります。自動化テーブルを使用すると、メッセージの経路指定は集中化され、したがって、モニターしやすくなります。

- すべてのメッセージを自動化テーブルで経路指定した場合、すべての経路指定命令が 1 つのファイルかファイルのセットに入れられているため、テーブルの保守が容易になります。相反する経路指定命令を作成する可能性が低くなり、修正も簡単に行えます。
- NetView プログラム自動化テーブルでメッセージを経路指定すると、ほとんどの場合、メッセージが送信請求メッセージであるのか、非送信請求メッセージであるのかを意識しなくても済みます。しかし、自動化テーブルを使用すると、必要に応じて送信請求メッセージを見分けることができます。IFRAUIND のビット 16 は、メッセージが、NetView プログラムによって、非送信請求メッセージと見なされるかどうかを示します。このビットは、IFRAUIND 自動化テーブル・アクションでチェックできます。

MSGROUTE コマンドによるメッセージの経路指定

MSGROUTE コマンドを使用すると、メッセージのコピーを次の宛先に送ることができます。

- 特定のオペレーターまたは自動タスク
- オペレーター・グループ
- システム・オペレーター (SYSOP)
- NetView プログラム・ハードコピー・ログ
- ネットワーク・ログ (LOG)

MSGROUTE コマンドは、NetView プログラム自動化テーブルで開始されるコマンド・リストから出されます。MSGROUTE は、NetView プログラム自動化テーブルと同様に、BEEP や DISPLAY のようなアクションをメッセージに設定することができます。ただし、MSGROUTE コマンドに指定されたアクションで、特定のメッセージに関して、NetView プログラム自動化テーブルに指定されているアクションの指定は変更できません。NetView プログラムは、メッセージが MSGROUTE コマンドで経路指定されている場合には、そのメッセージを自動化テーブルに再度送信しません。ただし、コピーがクロスドメインで経路指定されている場合、クロスドメイン自動化テーブルでメッセージが処理されます。

MSGROUTE コマンドの使用は、詳細な情報がない状態で、メッセージをどこに経路指定するか、またはどのようなアクションを取るかを決定するのに役立ちます。例えば、メッセージをどこに経路指定するか決定する前に、複数行メッセージの 2 行目をチェックするように、コマンド・リストを検討することができます。

経路コードに基づいた EMCS コンソールへのメッセージの経路指定

経路コードに基づいて MVS システム・メッセージを経路指定する場合には、EMCS コンソールをセットアップして、必要な経路コードを受け取ってください。NetView プログラムでは、経路コードの設定により、通常、メッセージの受信が自動化されることはありません。そのようなメッセージのコピーが CNMCSSIR サブシステム・ルーター・タスクでも受信されるため、自動化の必要はありません。経路コードの詳細については、76 ページの『経路コードの使用』を参照してください。

経路コードの指定

リソース・アクセス管理機能 (RACF) OPERPARM セグメントを使用するか、あるいは ROUT キーワードを MVS VARY コマンド上で使用すると、EMCS コンソールで受信したい経路コードを指定できます。NetView プログラムは、これらのメッセージにオペレーター ECMS コンソール名がないため、非送信請求として扱います。

メッセージ経路指定の流れ

CNMCSSIR サブシステム・ルーター・タスクで受信される非送信請求メッセージの場合、NetView プログラムでのメッセージ経路指定の流れは以下のとおりです。

1. DSIEX17 処理
2. PIPE CORRWAIT
3. ASSIGN PRI/SEC 処理
4. 許可レシーバーの処理
5. DSIEX02A 処理
6. 待機処理
7. 自動化テーブル処理
8. DSIEX16 処理
9. ASSIGN COPY 処理
10. 廃棄または表示処理

表 2 では、次のメッセージ・タイプの経路指定ステップを示します。

- MVS サブシステム・インターフェースから出される非送信請求メッセージ
- その他の非送信請求メッセージ
- 送信請求メッセージ

この表では、メッセージが表の最上部から入力されて、下部に向かって流れると想定してください。メッセージの分類に変更があった場合、メッセージの流れは、すでに行ったステップを繰り返さずに、表の新しい欄で続行します。NetView プログラムは、自動化テーブルと、各インストール・システム出口を、オリジナル・メッセージごとに 1 つだけ呼び出します。

例えば、VTAM から送られてくる非送信請求メッセージには、「その他の非送信請求メッセージ」の欄に記載されているステップが適用されます。つまり、非送信請求メッセージには、ASSIGN(PRI/SEC)、許可レシーバー、DSIEX02A、および自動化テーブル処理が適用されます。ここでは、自動化テーブルは、自動タスクへメッセージを経路指定すると想定します。それ以降は、NetView プログラムは、このメッセージを送信請求メッセージと見なします。メッセージの流れは、すでに行ったステップを繰り返さずに、「その他の送信請求メッセージ」の欄中に説明されているように処理されます。送信請求メッセージは、待機、DSIEX16、ASSIGN(COPY)、ログ記録、および表示処理を経ます。

表 2. NetView プログラム・メッセージ経路指定

ステップ	非送信請求 MVS メッセージ	その他の非送信 請求メッセージ	送信請求 メッセージ	MVS その他の送信請 求メッセージ
DSIEX17	●		●	
PIPE CORRWAIT			注 2	注 2
ASSIGN (PRI/SEC)	●	●		
許可レシーバー		●		
DSIEX02A	●	●	●	●
待機処理	注 1	注 1	●	●

表 2. NetView プログラム・メッセージ経路指定 (続き)

ステップ	非送信請求		送信請求 メッセージ	MVS 送信請求 メッセージ	その他の送信請 求メッセージ
	MVS メッセー ジ	その他の非送信 請求メッセージ			
NetView プログラム自 動化テーブル	●	●	●		●
DSIEX16	●	●	●		●
ASSIGN (COPY)			●		●
ロギング	●	●	●		●
NetView プログラムへ の表示			●		●
システムへの表示		●	●		●
廃棄	●				

注:

1. 非送信請求メッセージの待機処理は、メッセージが待機中のタスクに経路指定されているときにしか行われません。
2. メッセージがパイプライン内のコマンドにより送信請求されると、後続の経路指定はすべて置き換えられ、実施されません。パイプラインからメッセージが再度出されると、そのメッセージは非 MVS 送信請求メッセージとして処理されます。

DSIEX17 処理

DSIEX17 インストール・システム出口が呼び出され、すべてのインバウンド MVS メッセージ、送信請求メッセージ、非送信請求メッセージ、あるいは削除オペレーター・メッセージ (DOM) が処理されます。この出口により、自動化テーブルが呼び出される前にメッセージを変更、置換、または削除できます。この出口を使用して、メッセージまたは DOM を削除できます。また、DSIEX17 出口では、メッセージを自動化するかどうかを制御できます。

PIPE CORRWAIT

NetView PIPE コマンドの CORRWAIT ステージを使用して、次のメッセージを識別できます。

- パイプラインから出されるコマンドに対応しているメッセージ
- パイプラインで処理されなければならないメッセージ

メッセージには、DSIEX02A インストール・システム出口、ASSIGN 経路指定、および自動化で検出しやすいように、印が付けられています。メッセージがこれらのステップのいずれかを経ていると、その後パイプラインに取り込まれて再び出された場合には、再度そのステップを適用することはありません。「IBM Tivoli NetView for z/OS プログラミング: パイプ」に記載された、CONSOLE ステージの ONLY オプションの例外を参照してください。

ASSIGN PRI/SEC 処理

ASSIGN PRI/SEC 処理は、非送信請求メッセージにおいてのみ使用できます。EMCS コンソールを使用している場合、経路コードに基づいて、直接 NetView プ

ログラム OST に送られた MVS システム・メッセージは、非送信請求メッセージと見なされます。ただし、これらのメッセージは ASSIGN PRI/SEC 処理の対象にはなりません。これは、宛先が経路コードに基づいて選択されるためです。これらのメッセージを自動化可能として設定するように DSIEX17 出口をコーディングした場合、それらの経路指定に自動化テーブルを使用することもできます。PPT へ送られる送信請求 MVS システム・メッセージは、ASSIGN PRI および ASSIGN SEC で処理できます。

MPF またはメッセージ改訂で自動操作のマークが付けられたすべての MVS システム・メッセージは、CNMCSSIR サブシステム・ルーター・タスクに送信されます。これらのメッセージは、コマンド応答フラグや、宛先としてのコンソール名の有無に関係なく、非送信請求と見なされます。メッセージがコンソール名に基づいて別のタスクに送信された場合、CNMCSSIR タスクはメッセージを処理しません。それ以外の場合、CNMCSSIR タスクはメッセージをチェックして、1 次受信先に割り当ててどうかを判別します。

ASSIGN コマンドの PRI オペランドを指定したメッセージが割り当てられるオペレーターまたは自動タスクが、1 次受信先になります。1 次受信先がログオンされると、メッセージはそのオペレーター ID に割り当てられます。すると、ASSIGN コマンドの SEC オペランドに指定された任意のオペレーターにあてて、メッセージの 2 次コピーが作成されます。2 次コピーは、自動化テーブルによる処理を対象とはしていません。ただし、クロスドメイン NetView プログラムに送られた 2 次コピーは、クロスドメイン NetView プログラムの自動化テーブルで処理されます。

許可レシーバーの処理

1 次受信先の存在が確認されない非送信請求メッセージは、使用可能な許可レシーバーがあれば、そこに送られます。1 次受信先が見つからない場合、CNMCSSIR タスクはメッセージを自動化テーブルに送信します。自動化テーブルによって有効なアクションが実行されたが、メッセージが ROUTE アクションによって明示的に経路指定されない場合、NetView プログラムはメッセージを 1 次自動タスクに経路指定するかどうかを以下のように決定します。

- DISPLAY(YES) アクションが指定されている場合、メッセージは 1 次自動タスクに経路指定されます。
- DISPLAY(YES) アクションが指定されていない場合、メッセージはログに記録されますが、それ以上は経路指定されません。

DSIEX02A 処理

DSIEX02A インストール・システム出口は、オペレーターの端末に送られる標準出力データを処理するために呼び出されます。この出口を使用して、自動化テーブルが呼び出される前にメッセージを変更、置換、または削除できます。

この出口で (出口から出される USERDROP 戻りコードで、または IFRAUTBA フィールドを 'B'0' に設定することによって) メッセージを削除した場合、NetView プログラムは、そのメッセージの自動化テーブルを検索したり、DSIEX16 出口を呼び出すことはありません。

DSIEX02A がメッセージの IFRAUTB ビットをオンに設定すると、NetView プログラムは、そのメッセージの自動化テーブルを検索しません。ただし、そのメッセ

ージは DSIEX16 によって処理されます。DSIEX02A の詳細については、329 ページの『第 17 章 インストール・システム出口』を参照してください。

待機処理

DSIEX02A 処理のあとで、すべての送信されたメッセージがチェックされ、そのメッセージの送信先のタスクのもとで操作を行っているコマンド・プロシージャーの、未解決の待ち条件が満たされているかどうかが判別されます。

PL/I、C、REXX、および NetView コマンド・リスト言語で作成されたコマンド・プロシージャーでは、特定のメッセージまたはメッセージのグループを待機している間に、処理を保留することが可能です。PL/I、C、および REXX コマンド・プロシージャーでは、この機能に関して TRAP および WAIT コマンドを使用します。NetView コマンド・リスト言語では、&WAIT を使用します。

待機処理の対象になるメッセージには、次のものがあります。

- オペレーターまたは自動タスクによって送信請求されるすべてのメッセージ
- ASSIGN COPY で作成された送信請求メッセージのコピー
- ASSIGN PRI または許可レシーバー処理でオペレーター、または自動タスクに割り当てられた非送信請求メッセージ
- ASSIGN SEC で作成された非送信請求メッセージの 2 次コピー

メッセージが待ち条件を満たした場合、待機コマンド・プロシージャーの処理は再開されます。この時点で抑止されなかったメッセージは、メッセージの流れに従って続行されます。ただし、メッセージを抑止した場合には、NetView プログラムによって、そのメッセージに削除の印が付けられます。この場合、自動化テーブル処理は行われず、NetView プログラムはそのメッセージを表示せず、ログにも記録しません。このメッセージは、DSIEX16 インストール・システム出口に送られます。PL/I または C コマンド・プロセッサ、あるいは REXX コマンド・リストで作成されたメッセージを、TRAP および SUPPRESS で抑止できます。NetView コマンド・リスト言語では、&WAIT SUPPRESS ステートメントを使用できます。

自動化テーブルにより再度経路指定されたメッセージの場合は、新しいタスクで 2 度目の待機処理を受ける可能性があります (『自動化テーブル処理』で説明しています)。

自動化テーブル処理

ログに直接書き込まれたメッセージを除いて、あらゆるソースから出された送信請求メッセージおよび非送信請求メッセージは、そのメッセージの最初のインスタンスのための自動化テーブル処理の対象となります。SEC または COPY オペランドを指定した ASSIGN コマンド、MSGROUTE コマンド、あるいは自動化テーブル自体の ROUTE キーワードによって作成されるメッセージのコピーは、自動化テーブル処理の対象にはなりません。ただし、コピーがクロスドメインで経路指定されている場合、別のドメインの自動化テーブルでメッセージが処理されます。

経路指定メッセージ

自動化テーブル処理において ROUTE キーワードを使用すると、以前に ASSIGN PRI または許可レシーバー処理で経路指定した非送信請求メッセージを、再度経路指定できます。同様に、送信請求メッセージの自動割り当てを変更して、新たに受

信先を追加したり、最初の受信先を消去したりすることも可能です。送信請求メッセージのコピー割り当てには、影響がありません。コピーの送り先は、常に ASSIGN COPY コマンドで指定されたオペレーターになります。

自動化テーブル・ステートメントをコーディングすると、オペレーター、自動タスク、オペレーター・グループ、および PPT を任意に組み合わせて、メッセージやコマンドの送り先に指定できます。経路指定されるコマンドには、コマンド・プロセッサおよびコマンド・リストを含めることができます。メッセージを受け取る予定のオペレーター ID のリストが、メッセージの応答として出すコマンドを処理することになっているオペレーター ID のリストと、一致しなくても構いません。

DSI374A の ID が指定されたメッセージは、自動化テーブル処理を受ける準備が整い、また、図 13 に示すステートメントが、自動化テーブルに入っているとします。

```
IF MSGID='DSI374A' THEN  
    EXEC(ROUTE(ALL OPER1 OPER2 *));
```

図 13. 自動化テーブルの MSGID ステートメント

この例では、メッセージ DSI374A のコピーは、OPER1、OPER2 と、自動化テーブル処理に入力した際にそのメッセージと関連付けられたオペレーターに送られます。自動化テーブルの ROUTE キーワードで作成され、新しいタスクへ送られるメッセージのコピーは、その新しいタスクにおいて待機処理を受ける対象になります (106 ページの『待機処理』で説明しています)。

自動化テーブル内に一致するものがないメッセージは、自動化テーブル処理が行われる際にそのメッセージと関連付けられた受信先に送られます。送信請求メッセージの場合は、そのメッセージで入力データが生成されたタスクが受信先になります。非送信請求メッセージの場合は、(1 次受信先が割り当てられており、そのいずれかがログオンしているならば) そのメッセージに割り当てられた 1 次受信先が受信先になります。

1 次受信先をもたない非送信請求 MVS システム・メッセージの場合は、CNMCSSIR タスクが自動化テーブルをスキャンします。一致するものがあると、EXEC(CMD) を使用して出されるコマンドを、ROUTE キーワードで特定のタスクに経路指定しなければなりません。ROUTE キーワードがない場合、メッセージは CNMSTYLE メンバーの FUNCTION.AUTOTASK.PRIMARY ステートメントで定義した 1 次自動タスクに経路指定されます。

DST から出された割り当ての行われていないメッセージの場合、デフォルトの受信先は次のいずれかになります。

- DST を開始したタスク (そのタスクがログオンされている場合)
- 許可レシーバー (許可レシーバーが存在する場合)
- システム・コンソール・オペレーター

それ以外の非送信請求メッセージ (1 次受信先が割り当てられていないもの) は、許可レシーバーまたはシステム・コンソール・オペレーターに送られます。

メッセージ属性の設定

自動化テーブルで、メッセージの色や強調表示属性をチェックまたは設定できます。自動化テーブルでは、メッセージにログや表示特性のような属性を指定できません。

このような自動化テーブルによる設定値は、NetView DEFAULTS コマンドで指定された属性よりも、優先されます。SCRNFMT キーワードで設定されたメッセージの色や輝度を除いて、NetView OVERRIDE コマンドで指定された属性は、自動化テーブルの設定値よりも優先されます。

DSIEX16 処理

メッセージが自動化の対象になると見なされると、NetView プログラムにより、DSIEX16 インストール・システム出口が呼び出されます。この出口を使用すると、ユーザーはメッセージ・テキストや処理オプションを変更できます。

DSIEX16 の詳細については、329 ページの『第 17 章 インストール・システム出口』を参照してください。

ASSIGN COPY 処理

自動化テーブル処理のあとで、ASSIGN COPY コマンドが有効であれば、NetView プログラムでは、各宛先のオペレーター用に非送信請求メッセージのコピーが作成されます。これらのコピーには、元のメッセージのインスタンスから、DISPLAY、NETLOG、および BEEP などの、表示およびログ属性が取り込まれています。したがって、元のメッセージの自動化テーブル項目が、ASSIGN COPY コマンドを使用して作成されたコピーにも影響を与える可能性があります。

非送信請求メッセージの SEC オペランドで作成される 2 次コピーは、DEFAULTS または OVERRIDE コマンドでデフォルト値が変更される場合以外は、NetView システムのデフォルトを採用します。ASSIGN COPY 処理によって作成されたコピーは、待機処理を受けます (106 ページの『待機処理』で説明しています)。

廃棄または表示処理

NetView プログラムは、経路指定が完了すると、メッセージの廃棄または表示を行います。再度経路指定されないと、すべての非送信請求 MVS システム・メッセージは、NetView プログラムにより廃棄されます。それ以外のすべての非送信請求メッセージおよび送信請求メッセージは、(メッセージの表示オプションがインストール・システム出口または自動化テーブルにより、オフにされている場合を除いて)オペレーティング・システムに関係なく、NetView プログラムにより表示されます。

NetView プログラムのハードウェア・モニター・データ、および MSU の経路指定

このセクションでは、ハードウェア・モニターに送られるデータの流れと、自動操作に送られる MSU の流れを説明します。ハードウェア・モニターにデータを送る方法は数種類あります。次にその方法を示します。

- LUC セッションを通して、ある NetView プログラムから別の NetView プログラムへアラートを転送する方法。
- MS トランスポートを通して、ALERT-NETOP アプリケーションへ、複数ドメイン・サポート・メッセージ単位 (MDS-MU) を送信する方法。
- プログラム間インターフェースを通じて、ハードウェア・モニターへ、制御点管理サービス単位 (CP-MSU) またはネットワーク管理ベクトル・トランスポート (NMVT) を送信する方法。
- 通信ネットワーク管理 (CNM) インターフェースを通じて、ハードウェア・モニターの問題記録 (NMVT、保守用統計レコード [RECMS]、またはフォーマット済み保守用統計レコード [RECFMS]) を受信する方法。
- GENALERT コマンドを使用して、NetView プログラム内部でハードウェア・モニター・レコードを生成する方法。
- ローカル MVS 装置から、ハードウェア・モニターに関するシステム形式のレコード (OBR、MDR、MCH、CWR、または SLH) を受信する方法。

ハードウェア・モニターで受信したレコードの多くが、通常の処理の間に自動化テーブルに送られます。自動化テーブルでは、フィルター属性や強調表示属性を変更したり、自動応答を出したりできます。自動化テーブルに渡されるレコードは、NMVT、CP-MSU、MDS-MU、RECMS、および RECFMS であり、まとめて MSU と呼ばれます。ハードウェア・モニターは、以下のものを含む MSU のみを送信します。

- アラート (キー X'0000')
- リンク・イベント (キー X'0001')
- レゾリューション (キー X'0002')
- PD 統計 (キー X'0025')
- RECMS (X'1044' にカプセル化)
- RECFMS (X'1045' にカプセル化)
- リンク構成データ (キー X'1332')

経路指定およびターゲット命令 GDS 変数 (キー X'154D') は、アラートまたはレゾリューション主ベクトルに付加された自動化テーブルに送ることができます。大部分のリンク・イベント (キー X'0001') など、その他の主ベクトルのうちでも特定のものが、ハードウェア・モニターによってアラート主ベクトルに変換されます。この場合、オリジナルの主ベクトルと変換後のアラート主ベクトルが自動化テーブルに送られます。

NetView プログラムを使用すると、ハードウェア・モニターを介さずに直接 MSU を自動化テーブルに送ることも可能です。この機能は、例えば、自動化テーブルを通して自動的に送られる主ベクトルを持っていない MSU を、自動化したい場合などに役立ちます。

MSU を自動化に直接送る場合、PL/I または C であれば CNMAUTO サービス・ルーチンを、アセンブラーであれば DSIAUTO マクロを使用します。あるいは、MS トランスポート・インターフェースを使用して、MSU を汎用自動化レシーバー (NVAUTO) に経路指定する方法もあります。汎用自動化レシーバーは、MSU を自動化テーブルに提供し、その MSU を廃棄するアプリケーション・プログラムです。

図 14 は、問題レコードをハードウェア・モニターに送信するためのインターフェース、MSU を自動化に送るためのインターフェース、およびそれぞれの場合にデータが取る経路を示しています。この図において、ハードウェア・モニターに送られる各マルチドメイン・サポート・メッセージ単位 (MDS-MU) には、必ず制御点管理サービス単位 (CP-MSU) が含まれています。ハードウェア・モニターから自動化テーブルに送られる CP-MSU には、自動化をサポートする主ベクトルが含まれていなければなりません。主要なステップについては、図 14 で説明します。

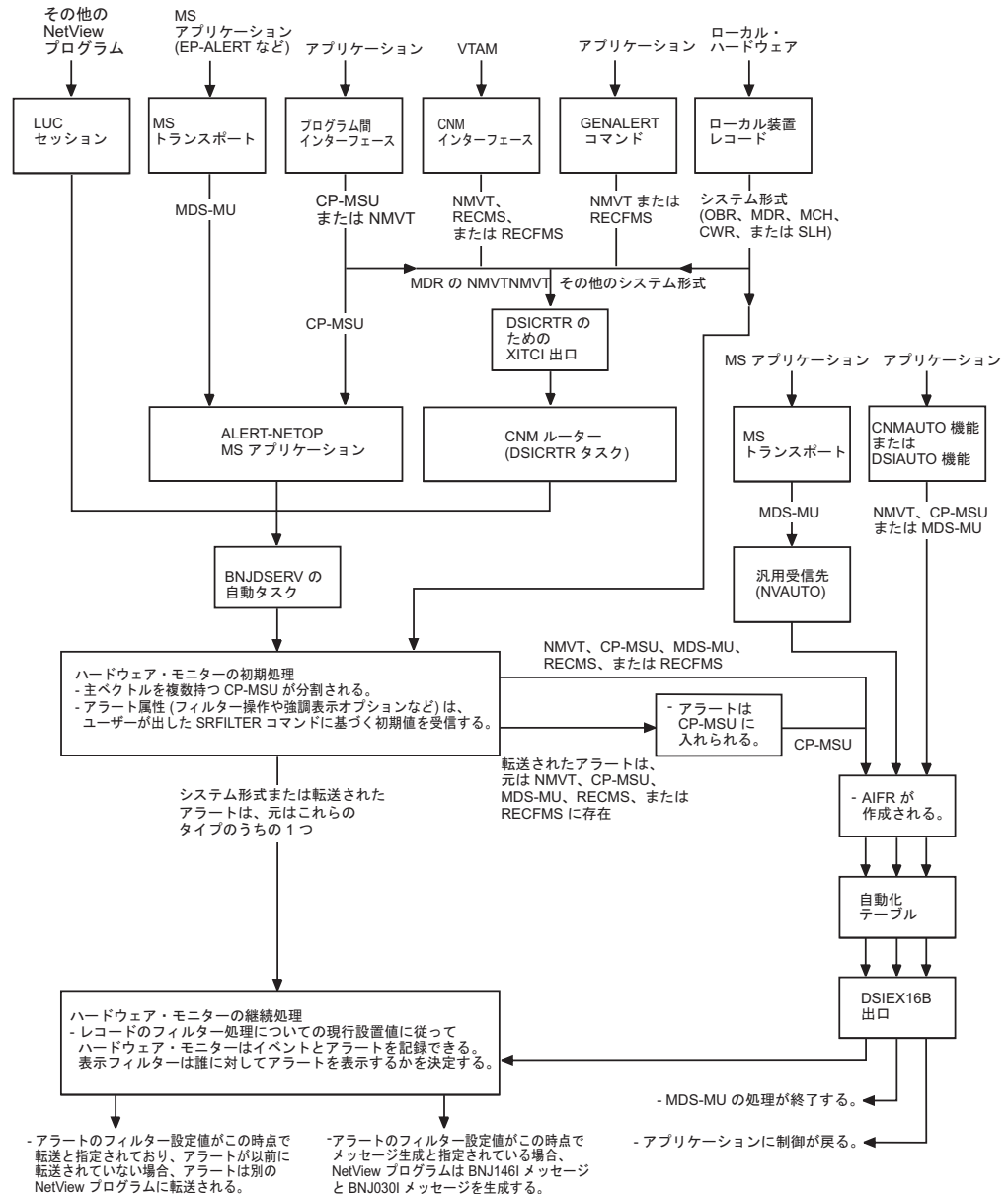


図 14. ハードウェア・モニターのデータの流れと自動化の MSU の流れ

ALERT-NETOP アプリケーション

ALERT-NETOP は NetView プログラム提供の MS アプリケーションであり、MSU を受信して、ハードウェア・モニターに渡します。

XITCI 処理

ハードウェア・モニターが MSU またはその他の問題レコードを受信するごとに、NetView プログラムにより、BNJDSERV タスク用の XITCI インストール・システム出口が呼び出されます。問題レコードが CNM ルーターを通して送られた場合、NetView プログラムは、DSICRTR タスクのための出口 XITCI も呼び出します。

いずれの XITCI インストール・システム出口も、問題レコードを変更、置換、または削除する機能を持ちます。別の NetView ドメインからの LUC セッションにより転送されたアラートは、この時点では特殊な転送形式になっています。XITCI 出口または転送形式の詳細については、「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」および「*IBM Tivoli NetView for z/OS* プログラミング: PL/I および C」を参照してください。

ハードウェア・モニターの初期処理

ALERT-NETOP アプリケーションを通して、CP-MSU をハードウェア・モニターに送る場合 (単独で送る場合であっても、MDS-MU に組み込んで送る場合であっても)、その CP-MSU には複数の主ベクトルが含まれることがあります。その場合、ハードウェア・モニターは、まず、それぞれ 1 つの主ベクトルをもった個別の CP-MSU ヘデータを分割します。次に、NetView プログラムで各主ベクトルを別々に処理します。分割する CP-MSU が MDS-MU に組み込まれている場合、新しい CP-MSU は、それぞれ、元の MDS-MU と同じヘッダー情報をもつ MDS-MU に送られます。この場合、2 つの例外があります。

- 自動化を受けない、基本エンコード規則 (BER) でエンコードされたデータ

特に、X'130F' 主ベクトルの前にある主ベクトル X'000F'、および X'132F' 主ベクトルの前にある主ベクトル X'1330' は、自動化を受けません。

- 経路指定およびターゲット命令 GDS 変数 (X'154D')

経路指定およびターゲットに関する情報は、あとにすぐ主ベクトルが続いている CP-MSU に置かれますが、NetView プログラムは、経路指定および目標情報を新しい CP-MSU の終わりに移動します。

保守用統計レコード (RECMS) および保守用統計様式化レコード (RECFMS) は、ユーザーの作成したアプリケーションによって、X'0000' 主ベクトルを自動化に実行依頼した場合とまったく同じように、自動化へ実行依頼できます。このアプリケーションは、RECMS を X'1044' 主ベクトルにカプセル化するか、あるいは RECFMS を X'1045' 主ベクトルにカプセル化し、次にそれらを再度 X'1212' CP-MSU にカプセル化します。

MDS-MU の CP-MSU 内にある X'132E' 主ベクトルに RECFMS レコードをカプセル化することによって、このレコードを ALERT-NETOP アプリケーションに送ることができます。そうすると、RECFMS はハードウェア・モニターによって通常どおり抽出され、処理されます。

さらに、ハードウェア・モニターに送られるすべてのアラート・タイプのデータは、NetView プログラムにより、ユーザーの SRFILTER 設定値に基づいたフィルターおよび強調表示属性で初期設定されます。

自動化テーブル処理

ハードウェア・モニターに処理される MSU は、その MSU に X'0000'、X'0001'、X'0002'、X'0025'、X'1332'、RECMS、または RECFMS が含まれていれば、すべて自動化テーブル処理の対象となります。最初に分散 NetView システム上の MSU に組み込まれていたアラートが転送されると、自動化のために MSU 形式に戻されます。ハードウェア・モニターは、これらのアラートを CP-MSU 形式にします。外部レコード (OBR)、機械チェック・ハンドラー (MCH)、チャンネル・リカバリー・ワード (CWR)、および 2 次レベル割り込みハンドラー (SLIH) などのようなシステム形式レコードは、自動化テーブル処理を受けません。

自動化テーブルでは、以下の項目を確認または設定することができます。

- 色
- 強調表示
- MSU の属性ハードウェア・モニターのフィルター操作

ハードウェア・モニターの処理を受けない MSU は、PL/I および C の CNMAUTO サービス・ルーチン、アセンブラの DSIAUTO マクロ、あるいは自動化テーブルを呼び出す汎用自動化レシーバー MS アプリケーション (NVAUTO) を通して直接自動化へ送ることができます。この場合、ハードウェア・モニターが MSU を処理しないため、強調表示設定や属性のフィルター操作は機能しません。ただし、自動化テーブルを使用して、MSU への応答として自動化コマンドを初期化することはできます。

MSU への応答を自動化する場合には、そのコマンドを自動タスクに経路指定してください。ハードウェア・モニターのデータ・サービス・タスク (DST) BNJDSERV が、MSU を自動化テーブルに送り、テーブル内の合致したステートメントに、実行するコマンドを指定した EXEC アクションが含まれるが、ROUTE 指定が含まれない場合、そのコマンドは BNJDSERV を開始した OST に送られます。OST がアクティブではない場合には、NetView プログラムでコマンドを経路指定できません。この場合、NetView プログラムからネットワーク・ログあてに問題を提示するメッセージが出されます。したがって、ハードウェア・モニターから MSU へ EXEC アクションを適用する際には、変動のない永続的な自動タスクから BNJDSERV を開始するか、あるいは常に ROUTE を使用するかにしてください。

DSIEX16B 処理

NetView プログラムは、MSU の自動化が検討されたあとで、DSIEX16B インストール・システム出口を呼び出します。この出口から、MSU を変更、置換、または削除することが可能です。詳しくは、329 ページの『第 17 章 インストール・システム出口』を参照してください。

ハードウェア・モニターの継続処理

ハードウェア・モニターにより処理されるタイプの問題レコードは、そのレコードの ESREC および AREC フィルター属性に関する最終的な設定値に従って、イベントおよびアラート・データベースに格納されます。

レコードから ESREC または AREC レコード・フィルターが渡され、そのレコードがイベントまたはアラートとして記録される場合、オペレーターはハードウェア・モニター・パネル上にイベントまたはアラートを表示できます。表示フィルターは、イベントまたはアラートをどのオペレーターに対して表示するかを決定します。ハードウェア・モニター・コンソールの右端にパーセント記号 (%) が表示される場合は、自動化テーブル内の少なくとも 1 つのステートメントに一致したイベントまたはアラートが存在することを示しています。

レコードから ESREC および AREC レコード・フィルターの両方が渡されると、それ以外のフィルター (ROUTE、OPER、TECROUTE、および TRAPROUTE を含む) が適用されます。詳しくは、342 ページの『アラートのフィルター操作』を参照してください。

NetView プログラム・コマンドの経路指定

NetView プログラム・タスクに出されるコマンドの経路指定を制御することができます。以下のセクションで、どのタスクにどのコマンドを経路指定できるかということと、コマンドの経路指定に使用する機能について説明します。

コマンドとタスクとの互換性

経路指定するコマンド、コマンド・プロセッサー、またはコマンド・リストが、確実に宛先タスクのもとで実行できるようにしておいてください。NetView プログラム・メインタスクのもとで実行するタスクのクラスには、次のものがあります。

- メッセージを受信し、コマンド、コマンド・プロセッサー、およびコマンド・リストの処理を制御できるタスク。これらのタスクには、自動タスク、オペレーター端末タスク (OST)、NetView 間タスク (NNT)、および 1 次 POI タスク (PPT) があります。通常のコマンド (TYPE=R)、または即時コマンド (TYPE=I) として実行するコマンド、コマンド・リスト、およびコマンド・プロセッサーは、このタイプのタスクへ経路指定できます。

ただし、いくつかの制限が適用されます。自動タスクは、フルスクリーン・パネルを生成するコマンドを実行できません。また、自動タスクに待機処理を含むコマンド・プロシージャーを実行させる場合には、注意が必要です。無制限の待ち状態になって自動タスクが使用できなくなることを避けるため、WAIT 命令には、タイムアウト値を使用してください。PPT のもとでは実行できないコマンドもあります。これらには、フルスクリーン・パネルを生成するコマンド、待機処理を行うコマンド、さらにその他いくつかのコマンドが含まれます。

待機処理に関する詳細については、「*IBM Tivoli NetView for z/OS* プログラミング : REXX および NetView コマンド・リスト言語」を参照してください。

- BNJDSERV DST または CNMCSSIR タスクで、MSU が自動化テーブルに送られ、そのテーブル内の一致したステートメントでコマンドが実行されるが、ROUTE が指定されていない場合、CMD アクションは 1 次自動タスクに送られます (function.autotask.primary ステートメントで指定します)。
- 入出力操作などのサービスをユーザーに提供する DST。データ・サービス・コマンド (TYPE=D) として実行するコマンドを、DST に経路指定できます。
- ハードコピー・タスク。コマンドをハードコピー・タスクに経路指定することはできません。このタスクにはメッセージのみを経路指定してください。

コマンド経路指定機能

経路指定コマンドの基本機能は、次のとおりです。

- コマンドを自動化テーブルから出す際に、タスクを選択する、自動化テーブル ROUTE キーワード。
- コマンドをあるタスクから別のタスクに送る、NetView EXCMD コマンド。
- コマンド・プロセッサからコマンドを初期化する、CNMSMSG サービス・ルーチンおよび DSIMQS マクロ。
- コマンドを、他の NetView ドメインに送る、NetView RMTCMD コマンド。
- コマンドを RMTCMD および EXCMD と同じ方法で経路指定する、コマンド・ラベル接頭部。

自動化テーブル ROUTE キーワード

EXEC(CMD) アクションを指定した自動化テーブル内に ROUTE キーワードを指定することによって、コマンドを経路指定することができます。入力メッセージまたは MSU がエントリー・ポイントと一致し、かつ、NetView プログラムが応答コマンドを出す場合、そのコマンドは、ROUTE キーワードで指定されたタスクへ送られます。EXEC(CMD) アクションにおいて ROUTE キーワードを使用しない場合、NetView プログラムは、6 (249 ページ) で説明される規則を使用してコマンドのタスクを選択します。

CNMSMSG サービス・ルーチンおよび DSIMQS マクロ

PL/I または C で作成された CNMSMSG サービス・ルーチンと、アセンブラーで作成された DSIMQS マクロを使用して、コマンドを特定のタスク、ログ、およびその他の宛先に送ることができます。

EXCMD コマンド

EXCMD コマンドを使用すると、コマンド、コマンド・リスト、またはコマンド・プロセッサを、指示したタスク宛てに経路指定し、実行できます。コマンドは、必ず、経路指定したタスク・タイプのもとで実行できるようにしておいてください。例えば、データ・サービス・コマンド・プロセッサが実行できるのは、データ・サービス・コマンドだけです。

図 15 では、LOGOFF コマンドが、AUTO1 タスクに経路指定され、ここでコマンドが処理されます。その結果、AUTO1 はログオフされます。

```
EXCMD AUTO1 LOGOFF
```

図 15. EXCMD コマンドの例

注: サーバー・タスク DSIIPLLOG、DSIRXEXC、および DSIRSH の下で実行するコマンドは、キューに入れないでください。これらのタスクは、自由に TCP/IP 要求を処理するため使用可能な状態にしておかなければなりません。

RMTCMD コマンド

RMTCMD コマンドにより、システム、サブシステム、およびネットワーク・コマンドを、そのネットワーク内の任意の場所にある別の NetView プログラムに送ることができます。コマンドは、他の NetView プログラムにより処理されます。

ROUTE コマンドの代わりに RMTCMD コマンドを使用してください。なぜなら、RMTCMD の場合は OST-NNT セッションを開始する必要がないからです。

コマンド・ラベル接頭部

コマンド・ラベル接頭部 を使用すると、NetView RMTCMD、または EXCMD コマンドと同様に、コマンドを経路指定したり、応答を相関させることができます。応答の相関は、CORRCMD パイプ・ステージで有効です。ラベル付きコマンドの説明については、「*IBM Tivoli NetView for z/OS* ユーザーズ・ガイド: NetView」を参照してください。

コマンド優先順位

通常のコマンドを処理する NetView タスク (自動タスク、その他の OST、NNT、および PPT) は、それぞれ、キューに入れられたコマンドの、NetView における **コマンド優先順位** を認識します。キューに入れられたコマンドには、低優先順位か、高優先順位のいずれかが割り当てられています。優先順位は、NetView プログラムでコマンドが実行される順位を決定する際に役立ちます。

コマンド優先順位は、DEFAULTS コマンドを指定してグローバルに設定できます。また、OVERRIDE コマンドを使用して各タスクに、あるいは CMD コマンドを使用して単一のコマンドに優先順位を設定できます。コマンドをキューに入れるそれ以外の方法に関しては、優先順位指定方法があります。

コマンドの優先順位は、オペレーターが出す通常のコマンド (例えば次のようなもの) に影響を与えます。

- MVS システム・コンソールからオペレーターが入力する NetView コマンド
- EXCMD コマンドによって中継されるコマンド

コマンドの優先順位は次のものには影響を与えません。

- コマンド・リスト内のコマンド

このようなコマンドは、キューに入れられるのではなく、連続して実行されます。

- 自動化テーブルから送出するコマンド

この種のコマンドは、常に低い優先順位でキューに入れられます。

自動化テーブルから CMD 接頭部を使用して、優先順位を高いものに変更することはできません。コマンドを AT、EVERY、または AFTER タイマー・コマンドを指定してスケジュールすると、スケジュール済みのコマンドに適用される DEFAULTS および OVERRIDE の設定値は、タイマーが満了になると有効になります。

自動化アプリケーションのコマンドが、低優先順位と高優先順位の両方に割り当てられてキューに入れられている場合、高優先順位のコマンドを、低優先順位のコマンドよりも優先して順不同に処理できることに注意してください。優先順位の低いコマンドは、それぞれの順番に従って処理されます。実行するタスクのキューに最初に入れられたコマンドが最初に処理されます。高優先順位のコマンドも、コマンド・プロシージャールの場合を除いて、それぞれの順番に従って実行されます。

コマンド・プロシージャは、タスクの高優先順位待ち行列のサービスを使用可能にするために、いくつかの地点で制御を放棄します。そのため、コマンド・プロシージャ自体に高優先順位が割り当てられている場合であっても、コマンド・プロシージャは高優先順位のコマンドによる割り込みを受ける場合があります。コマンド・プロシージャは、長時間実行コマンド群を実行しているとき、および待機または一時停止処理 (例えば、REXX の WAIT または PARSE PULL など) を実行しているときには、割り込みを受け入れます。さらに、REXX または NetView コマンド・リスト言語により作成されているプロシージャを使用すると、呼び出し時にすぐ (最初の命令より前) に割り込みを受け入れることも、コマンド・リスト内の各コマンドのあとに割り込みを受け入れることもあります。

コマンド・プロシージャを、送出された順序で処理するには、コマンド・プロシージャ全体を低優先順位でキューに入れてください。コマンド・プロシージャは、長時間実行コマンドを処理している場合にのみ、低優先順位のコマンドの割り込みを受け入れます。

コマンドの優先順位の詳細と、CMD コマンドおよび DEFAULTS コマンドの構文図については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」を参照してください。OVERRIDE コマンドに関する情報については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

第 4 部 NetView プログラム自動操作機能

第 10 章 コマンド・リストおよびコマンド・プロセッサ

コマンドを 1 つ発行するだけで複雑なアクションを実行できるようにする目的で、コマンド・リストおよびコマンド・プロセッサを使用して、自動化プロシーチャーを作成します。

コマンド・リスト は、REXX、または NetView コマンド・リスト言語のいずれかで作成された、コマンドおよび特殊な命令のセットです。NetView コマンド・リスト言語で作成されたコマンド・リストは、実行時解釈され、REXX で作成されたコマンド・リストは実行時解釈されるか、あるいはコンパイルされます。 コマンド・プロセッサ は、アセンブラー、PL/I、または C で作成されたモジュールです。(PL/I または C で作成された) コマンド・プロセッサとコマンド・リストは、コマンド・プロシーチャー と呼ばれることもあります。コマンド・リストまたはコマンド・プロセッサは、NetView プログラムコマンドであるかのように、送出することができます。

オペレーターのジョブを簡略化したり、自動化を援助したりするために、次のものでコマンド・リストおよびコマンド・プロセッサを使用することができます。

- オペレーター
- 自動化テーブル
- タイマー・コマンド
- EXCMD コマンド
- その他のコマンド・リスト
- その他のコマンド・プロセッサ

NetView プログラムの初期化、および OST の初期化の間に、初期コマンド・リストを処理するよう指定することもできます。コマンド・リストおよびコマンド・プロセッサで実行できる以下の機能があります。

- オペレーターから正常に出された一連の照会、応答、およびコマンドを、1 つのコマンドで置き換える操作。
- 入力基準に従い、さまざまな応答を出す操作。
- 実行される機能が、長いか複雑である場合に、オペレーターの応答の整合性を確実に保つようにする操作。
- 自動タスクのもとで実行する操作。

使用可能な言語

NetView プログラム・コマンド・リスト、およびコマンド・プロセッサを作成するのに使用可能な言語は、以下のとおりです。

- NetView プログラム・コマンド・リスト言語
- REXX
- PL/I
- C

- アセンブラー

各言語の機能の詳細については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。

メッセージおよび MSU の取得

自動化テーブルがメッセージまたは管理サービス単位 (MSU) を受け取ったときに、自動的にコマンド・プロシーチャーを発行するには、NetView プログラム自動化テーブルを使用します。この方法で出されたコマンド・プロシーチャーは、そのコマンド・プロシーチャーを出したメッセージまたは MSU に関連している情報にアクセスします。LU 6.2 トランスポートを介して、NetView プログラムが MSU を受信すると、その NetView プログラムは指定されたコマンド・プロシーチャーを出すことができます。このコマンド・プロシーチャーは、受信した MSU 用の情報にアクセスすることもできます。

メッセージまたは MSU 情報は、コマンド・プロシーチャーに使用することが可能であるため、このメッセージまたは MSU に関連付けられるデータの大部分は、自動化テーブル・ステートメントで解析してから、明示的にコマンド・プロシーチャーに送信する必要はありません。メッセージまたは MSU の属性には、コマンド・プロシーチャー内のさまざまな機能を使用してアクセスできます。詳細については、「*IBM Tivoli NetView for z/OS*プログラミング: REXXおよびNetViewコマンド・リスト言語」を参照してください。

また、自動化テーブル内の EDIT アクションを使用して、自動化されたメッセージと MSU に対する変更を行うことができます。これらの変更は、PIPE EDIT ステージで提供される構文と機能を使用して行われます。EDIT 指定の詳細については、PIPE EDIT のオンライン・ヘルプを参照してください。

メッセージの機能

コマンド・プロシーチャー言語からは、さまざまなメッセージ属性にアクセスするためのキーワードが提供されます。例えば、メッセージ ID、メッセージ・タイプ (HDRMTYPE)、およびメッセージ・テキストを検査できます。MVS システム・メッセージの場合には、ジョブ名または応答 ID を検査することも可能です。

MSU の機能

コマンド・プロシーチャーで、MSU について検査し、処理を行うことが可能です。REXX では、HIER 関数によりアラートのハードウェア・モニターのリソース階層がわかります。また、MSUSEG 関数を使用すれば、MSU の内容についても調べられます。MSU には、MDS-MU のヘッダー情報が含まれる場合があります。HIER および MSUSEG REXX 関数は、(構文の詳細は異なりますが、) 自動化テーブルの HIER および MSUSEG 比較項目と同じです。NetView コマンド・リスト言語では、同様の &HIER および &MSUSEG 制御変数が提供されます。さらに、REXX では、ハードウェア・モニター汎用アラートのコード・ポイントを、指定したテキスト・ストリングに変換できる CODE2TXT 関数が提供されます。この関数は、PL/I または C で作成された CNMC2T (CNMCODE2TXT) サービス・ルーチンでも使用できます。

情報の保管

情報は、グローバル変数、または MVS データ・セットを使用して保管できます。

グローバル変数

コマンド・リストおよびコマンド・プロセッサは、操作手順を自動化するための機能を提供します。そのうちの 1 つに、情報をコマンド・リスト、コマンド・プロセッサ、および自動化テーブル間で送り渡すために使用するグローバル変数を作成および更新するという機能があります。グローバル変数は、次のような目的で自動化の手順を作成するのに役立ちます。

- 自動化でユーザーの環境をモニターしているときに、システムおよびネットワーク・エレメントの現行の状況を維持する。
- 自動化プロシージャにシステム名をコーディングする必要をなくす。これにより、さまざまな個所でコーディングを変更するのではなく、グローバル変数を再定義するだけで、このプロシージャを別のシステムに適用できるようになります。
- パラメーター駆動の処理を自動化するときに、特定のパラメーター値をコーディングする必要をなくす。これにより、コマンド・リストおよびコマンド・プロセッサを再コーディングしなくても、パラメーター値を変更することができるようになります。
- 必要に応じて出されるジョブ名およびサブシステム・コマンドを保持する。

グローバル変数には、タスク・グローバル変数と共通グローバル変数の 2 つのタイプがあります。

QRYGLOBL コマンドを使用すると、共通グローバル変数およびタスク・グローバル変数の数と、それらの値を調べられます。QRYGLOBL コマンドに関する詳細については、NetView プログラム・オンライン・ヘルプを参照してください。

タスク・グローバル変数

そのタスクのもとで実行する各コマンド・リストまたはコマンド・プロセッサは、タスク・グローバル変数の値を設定、検査、および更新することができます。その他の NetView プログラム・タスクは、この変数に直接アクセスできません。したがって、さまざまな NetView プログラム・タスクが、タスク・グローバル変数に同じ名前を使用でき、同じ値は参照しません。NetView プログラムから、2 つのタスクが同じ名前を使用していることを示すことはありません。

ほかのタスクに属しているタスク・グローバル変数を検査または更新するためのタスクは、所有しているタスクへ要求を出されなければなりません。したがって、タスクはそれぞれが所有している変数の制御権を保持できます。各タスクごとに、タスク・グローバル変数を保管するためのタスク・グローバル・ディクショナリーを保有しています。タスク・グローバル変数は、保管、復元および除去できます。

共通グローバル変数

コマンド・リストまたはコマンド・プロセッサを実行できるタスクであれば、どのタスクも、共通グローバル変数を使用することができます。1 つの共通グローバル・ディクショナリーで、すべての共通グローバル変数を保管します。

共通グローバル変数は、保管、復元、および除去することができます。グローバル変数を保管する場合、これらグローバル変数は、NetView プログラムによって、VSAM データベースに入れられます。これらの変数は、保管された場所からグローバル・ディクショナリーに復元できます。保管しておいたグローバル変数で不要になったものは、データベースから除去します。重要なグローバル変数を保管しておく、障害が発生した場合、または計画的にシステムを停止する場合に、リカバリが容易にできます。

変数のタイプの選択

タスク・グローバル変数は、単一の箇所からローカルに参照するのに最も適しています。1 つのタスクにのみ変数が必要な場合には、タスク・グローバル変数を使用することによって、名前の重複を避けられます。ただし、複数のタスクからチェックまたは更新したい情報には、共通グローバル変数を使用してください。自動化テーブルに情報を渡したいときには、共通グローバル変数が最適です。共通グローバル変数を使用すれば、どのタスクが自動化テーブルを使用するかということに注意を払わなくて済みます。

グローバル変数の詳細については、「*IBM Tivoli NetView for z/OS*プログラミング: REXXおよびNetViewコマンド・リスト言語」、*「IBM Tivoli NetView for z/OS*プログラミング:アセンブラー」、および「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。

ATF('DSICGLOB') および ATF('DSITGLOB') を指定して、自動化テーブルからグローバル変数の値を読み取る方法については、DSICGLOB を参照してください。

テーブルが Data REXX を使用してロードされるときに、グローバル変数の値を挿入できます。例えば、次のステートメントでは、メッセージのソースとサブシステム・ルーター・タスクの名前を比較するために、行が挿入されます。

```
%> 'IF IFRAUSDR =' CGLOBAL('CNMSTYLE.SSINAME') 'THEN'
```

Data REXX に関する詳細は、「*IBM Tivoli NetView for z/OS*プログラミング: REXX およびNetViewコマンド・リスト言語」を参照してください。

MVS のデータ・セット

データ・セットを使用して、コマンド・リストやコマンド・プロセッサからのデータを保管する方法もあります。REXX EXECIO および PIPE QSAM コマンドを使用して順次データ・セットからの読み取りおよび順次データ・セットへの書き込みを行うことができます。この機能は、さまざまな目的のために使用できます。

PL/I および C で書かれたコマンド・プロセッサは、NetView 区分データ・セット (CNMMEMO、CNMMEMR、CNMMEMC) への読み取りアクセス権を提供し、VSAM 入出力 (CNMKIO) を要求する高水準言語のサービス・ルーチンを使用できます。PL/I および C 入出力サービスは、データ・セットからの読み取りおよびデータ・セットへの書き込みのために使用することもできます。

アセンブラーで作成されたコマンド・プロセッサは、NetView ファイル (DSIDKS) への読み取りアクセス権を提供し、VSAM 入出力 (DSIZVSM) を要求する、NetView プログラム・マクロを使用できます。

特定のイベントの待機

NetView プログラムにより、メッセージおよびその他のイベントの受信を待機したり、受信した情報に基づいて処理を修正したりすることができます。PIPE コマンドの CORRWAIT ステージを使用して、パフォーマンスを最適化してください。詳しくは、「*IBM Tivoli NetView for z/OS*プログラミング:パイプ」を参照してください。

また、NetView プログラムにより、オペレーターにデータ (NetView コマンド・リスト言語の場合は &PAUSE、REXX の場合は PARSE PULL と PARSE EXT、および高水準言語の場合は WAIT FOR OPINPUT) の入力を請求することもできます。ただし、自動タスクは不在操作なので、自動タスクのもとで実行している自動化コマンド・リストおよびコマンド・プロセッサの入力請求機能は使用しないようにしてください。

イベントを待機するために使用されるコマンドは、コマンド・リスト用とコマンド・プロセッサ用とは異なります。この違いについては、以下のセクションで説明します。自動タスクのもとで実行している自動化コマンド・リストおよびコマンド・プロセッサは、自動タスクがイベントの発生を待機することのないようにしてください。待機機能を使用する場合には、自動タスクが無制限に待ち状態になることを確実に避けるようにタイムアウト値を指定しなければなりません。

NetView コマンド・リスト言語を使用した待機処理

&WAIT 制御ステートメントの基本形式を使用すると、指定のイベントが発生するまでの間、コマンド・リストの処理が中断されます。この &WAIT 制御ステートメントは、2 つの部分から構成されています。最初の部分はオプションであり、コマンド・リストの処理中に、&WAIT ステートメントに到達したときに処理される、コマンドまたは別のコマンド・リストを指定します。2 番目の部分は、イベントとラベルがペアになったリストであり、指定されたイベントが発生したときに処理が転送されるべき場所を指定します。指定できるイベントには、次のものがあります。

- NetView プログラム・コンソールに表示されるメッセージの受信
- 呼び出されたコマンドまたはコマンド・リストからのゼロ以外の戻りコードの受信
- 指定された時間の満了
- オペレーターによる GO コマンドの入力

メッセージの受信により &WAIT ステートメントの条件が満たされた場合、NetView プログラム制御変数を使用して、そのメッセージの内容を調べます。

NetView プログラム・コマンド・リスト言語によるイベントの待機に関する詳細については、「*IBM Tivoli NetView for z/OS*プログラミング: REXXおよびNetViewコマンド・リスト言語」を参照してください。

REXX を使用した待機処理

REXX は、メッセージの待機方法や、メッセージおよびその他のイベントの分析方法を提供するために対話を行う複数の命令を使用します。TRAP 命令により、トラップされるメッセージを指定し、かつ、そのトラップされたメッセージがオペレー

ターに対して表示されるかどうかを指定します。トラップされるメッセージは、メッセージ待ち行列に入れられるため、複数のメッセージを処理することが可能になります。WAIT 命令により、コマンド・リストでは、指定されたイベントが発生するまでの間、処理が中断されます。次のイベントを指定できます。

- トラップするメッセージ
- 秒単位または分単位で指定するタイムアウト値
- オペレーターによる GO コマンドの入力

MSGREAD 命令により、NetView プログラムで現在トラップされているメッセージの中から、1 つのメッセージを読み取ります。その場合、コマンド・リストは、受信したメッセージに基づいてアクションを行うことができます。FLUSHQ 命令は、トラップされたメッセージすべてをメッセージ待ち行列から破棄するために使用します。

REXX でのイベントの待機に関する詳細については、「*IBM Tivoli NetView for z/OS プログラミング: REXX*および*NetView*コマンド・リスト言語」を参照してください。

PL/I および C を使用した待機処理

高水準言語アプリケーション・プログラム・インターフェース (API) からは、メッセージの待機方法、およびメッセージやその他のイベントの分析方法 (REXX で使用されたのと同様の方法です) を作成するために対話を行う、コマンドおよびサービス・ルーチンがいくつか提供されます。TRAP コマンドにより、トラップされるメッセージを指定し、かつ、そのトラップされたメッセージがオペレーターに対して表示されるかどうかを指定します。トラップするメッセージは、コマンド・プロセッサのメッセージ待ち行列に入れられます。それによって、複数のメッセージが処理できるようになります。WAIT コマンドを使用すると、コマンド・プロセッサでは、指定されたイベントが発生するまでの間、処理が中断されます。発生可能なイベントは、以下のとおりです。

- Netview プログラム・コンソールへのメッセージの表示
- 秒単位または分単位で経過したタイムアウト値に指定された時間間隔のセット
- オペレーターが GO コマンドを入力
- CNMSMSG サービス・ルーチンによるデータの送信

CNMGETD サービス・ルーチンからは、データ待ち行列 (そのうちの 1 つは、TRAP および WAIT コマンドを使用してトラップされたメッセージすべてを含むメッセージ待ち行列) へのアクセス権が提供されます。CNMGETD サービス・ルーチンからは、REXX MSGREAD および FLUSHQ 命令と同等の機能やその他の機能が提供されます。

高水準言語のコマンド・プロセッサにおけるイベントの待機に関する詳細は、「*IBM Tivoli NetView for z/OS プログラミング: PL/I*および*C*」を参照してください。

MVS のその他のコマンド・リスト機能

MVS システムでは、コマンド・リストが MVS コンソールにメッセージを送信できます。コマンド・リストから、ディスク、テープまたは印刷ファイルの割り振りを行うことが可能です。オペレーターが最近操作したコマンドおよびテキストをコマンド・リストに保存することもできます。

MVS コンソールへのメッセージの送信

メッセージを MVS コンソールへ送信したり、MVS コンソールから除去したりするには、自動化コマンド・リストで次の 3 つの NetView プログラム・コマンドを使用します。

WTO MVS コンソールにメッセージを送信します。例えば、オペレーターの介入 (プリンターの用紙の補充、またはいくつかある処理方法のうち 1 つの選択など) が必要な場合、WTO コマンドを使用することができます。

WTOR

MVS コンソールにメッセージを送信し、応答を要求します。WTOR を使用するコマンド・リストは、オペレーターの応答があるまでは完了しません。

DOM MVS コンソールから WTO メッセージを除去します。DOM を使用して、アクションが実行された時点で、アクション・メッセージを除去することができます。

これらのコマンドの詳細については、「*IBM Tivoli NetView for z/OS* プログラミング : REXX および NetView コマンド・リスト言語」を参照してください。

ディスク、テープ、および印刷ファイルの割り振り

ALLOCATE コマンドを、REXX EXECIO またはコマンド・プロセッサのデータ・セット・アクセス機能と一緒に使用すれば、ディスク、テープ、印刷ファイル、および内部読み取りプログラムを割り振りできます。これらの機能を使用すると、自動化されたプロシージャで JCL を作成し、それを実行依頼できます。例えば、NetView プログラムで、システム管理機能 (SMF) データ・セットがいっぱいであることを示すメッセージを受信した場合、SMF データ・セット名を、適切なコマンド・リストまたはコマンド・プロセッサに渡すように、自動化テーブルを定義してください。データ・セット名は JCL に組み込まれ、このデータ・セットをダンピングするジョブが、NetView SUBMIT コマンドを使用して実行依頼されます。

注: ジョブ入力サブシステム (JES) が開始される前に開始された NetView プログラムの下で実行している場合は、JES データ・セット (内部読み取りプログラムまたは SYSOUT) を割り振ることはできません。

ストレージへのコマンド・リストのロード

システムのパフォーマンスを向上させるため、処理を行う前にコマンド・リストを主記憶域にロードしておくことができます。事前にロードされていないコマンド・リストを呼び出す場合、コマンド・リストは主記憶域にロードされ、処理されてから、主記憶域から除去されます。したがって、コマンド・リストは、処理のたびに

常駐先の補助記憶装置から検索されます。コマンド・リストを事前にロードしておくと、その度ごとに補助記憶装置から検索しなくても、何回か続けて処理を行うことができます。

次の NetView コマンドで、コマンド・リストを主記憶域に入れたり、主記憶域から出したり、現在主記憶域に入っているコマンド・リストを識別することができます。

LOADCL

すべてのオペレーターが共用する主記憶域にコマンド・リストをロードします。

DROPCL

LOADCL コマンドを使用して主記憶域に事前にロードされていたコマンド・リストを除去します。

MAPCL

現在主記憶域に常駐しているコマンド・リストを示します。

NetView プログラムは、LOADCL コマンドを使用して今までにストレージにロードされたコマンド・リストを管理するのに役立つ、コマンド・リストのサンプル (CNMS8003) を提供します。このサンプルは、MAPCL および DROPCL コマンドを使用して、主記憶域から条件付きでコマンド・リストを削除します。また、MEMSTORE コマンドを使用すると、ストレージにロードされているコマンド・リストや、他の NetView プログラム・データ・セット・メンバーを管理することができます。

これらのコマンドに関する詳細については、NetView プログラム・オンライン・ヘルプ、および「*IBM Tivoli NetView for z/OS*プログラミング: REXXおよびNetViewコマンド・リスト言語」を参照してください。

第 11 章 タイマー・コマンド

NetView プログラムの自動化では、タイマー・コマンドを使用して、他のコマンドの処理をスケジュールできます。タイマー・コマンドを使用すると、タスクから出された任意のコマンドまたはコマンド・リストをスケジュールできます。本章では、タイマー・コマンドおよびいくつかの関連するコマンドについて説明します。

タイマー・コマンドの概要

タイマー・コマンドは、ほかのコマンド (コマンド・リストやコマンド・プロセッサを含む) を発行したいことを、NetView プログラムに通知するコマンドです。タイマー・コマンドを出すと、何日も前から活動をスケジュールしたり、1 日に 1 回、または 1 カ月に 1 回定期的に行われる活動をスケジュールすることができます。タイマー・コマンドは、別のコマンドを次の時点でスケジュールするのに使用します。

- 指定された時間の経過後
- 指定された時刻
- 指定された時間間隔で繰り返す

タイマー・コマンドは AFTER、AT、EVERY、および CHRON です。関連するコマンド LIST TIMER および PURGE TIMER の 2 つは、コマンドのスケジュールを管理する際に役立ちます。AT、EVERY、AFTER、CHRON、および PURGE コマンドは、常に Canzlog ログにエコー出力されます。コマンドがログにエコー出力されない場合 (例えば、REXX プロシージャーによって実行される場合)、これらのコマンド・エコーは、NetView プログラム・メッセージとしてタグ付けされる代わりに TRACE としてタグ付けされます。

本セクションでは、AFTER、AT、EVERY、および CHRON コマンドについて説明します。これらは、オペレーターから直接出されることも、あるいはその他の自動化機能 (コマンド・リストおよびコマンド・プロセッサなど) で使用することも可能です。これらのコマンドの構文およびパラメーターに関する説明については、NetView プログラム・オンライン・ヘルプを参照してください。

注: AFTER、AT、EVERY、および CHRON コマンドは、カスタマイズされた日時形式をサポートします。本章に示す例は、すべてデフォルトの形式を前提としています。

注: 対話式コマンドのスケジューリングは、それらがオペレーターのタスクとして実際にオペレーターによって実行されるのでない限り、行わないでください。

AFTER

AFTER コマンドによって、指定した時間が経過した後にコマンドまたはコマンド・プロシージャーを実行するようにスケジュールすることができます。

発生することが予定されるある活動を一定時間待ったあとで、実際に発生したことをチェックしたい場合には、AFTER コマンドが有用です。例えば、NetView プログ

ラムを使用して、プロダクトを 5 分以内に初期設定したい場合、コマンド・リストをスケジュールして 5 分後にそのプロダクトが正常に開始されたかどうかをチェックできます。

図 16 に示す AFTER コマンドは、5 分後に MVS D A,L コマンドを出し、システム・エレメントに関する状況情報を送信請求するようにスケジュールしたものです。

```
AFTER 00:05:00,ID=DISPSTAT,MVS D A,L
```

図 16. AFTER コマンドの例

AFTER コマンドは、DELAY コマンドの代わりにも使えます。コマンド・リストまたはコマンド・プロセッサから DELAY コマンドが出された場合、そのコマンド・リストまたはコマンド・プロセッサと、その実行タスクは、指定した時間待機するので、その間そのタスクではほかの作業を実行できません。それに比べ AFTER コマンドは、コマンドをスケジュールした後でコマンド・リストまたはコマンド・プロセッサ、およびタスクを解放するため、指定した時間間隔の間でも他の作業を行えます。

AT

AT コマンドは、特定の時刻にコマンドまたはコマンド・プロシージャーを実行するようにスケジュールします。

例えば、図 17 に示す AT コマンドは、STOPSYS コマンド・リストによって 12 月 24 日の午後 6:00 にシステムをシャットダウンし、保管・復元データベースにコマンドを保管するようにスケジュールします。

```
AT 12/24 18:00:00,ID=EVESSAVE,SAVE,STOPSYS
```

図 17. AT コマンドの例

特定の時間もしくは特定の日に実行させたいコマンドをスケジュールするには、AT を使用します。

EVERY

EVERY コマンドは、指定された時間間隔でコマンドまたはコマンド・プロシージャーを繰り返し処理するようにスケジュールします。時間間隔は、秒、分、時間、日のいずれの単位でも指定できます。コマンドまたはコマンド・プロシージャーは、EVERY コマンドが除去されるまで、指定された時間間隔で処理されます。

図 18 の EVERY コマンドは、このタイマー・コマンドが実行された 1 時間後から、コマンド・リスト CHEKSTAT を毎時実行するようにスケジュールします。

```
EVERY 01:00:00,ID=CHKST,CHEKSTAT AUTOVTAM
```

図 18. EVERY コマンドの例

自動タスクがログオンされていて、待ち状態のためにほかの作業の実行を妨げているのではないことを確認するために、その状態をチェックするには、図 18 の中のコマンドと同様の EVERY コマンドを使用します。 NetView 製品とともに提供され

ている、自動タスクのサンプル・セットには、自動タスクをチェックする方法の例が含まれています。この例では、自動化テーブルおよびコマンド・リストと、タイマー・コマンドが使用されています。このサンプルについては、651 ページの『付録 I. 自動操作のサンプル・セット』に説明があります。

TIMER

TIMER コマンドは、スケジュールされたタイマーの表示、追加、変更、テスト、または削除ができるようにするパネルを表示します。このコマンドは、フルスクリーン・モードだけで動作します。

CHRON

CHRON コマンドを使用すると、効率よく時間設定されたコマンドのスケジュールリングができ、例外ケースとタイム・シフトを決定する際に使用される REXX プロシージャ内のコーディングの量を減らすことができます。CHRON はまた、以前に複数のタイマーかまたは AT コマンドと EVERY コマンドの組み合わせを要した基準を結合することによって、タイマー・エレメントの数を削減します。

CHRON EVERY コマンドでは、現行の時間よりも早い開始時間を指定することができます。この機能は、1 つのシフトの間に複数の日に対して時間設定されたイベントをスケジュールしたり、当該シフトの間に最初のタイマーを開始したりする場合に有用です。また、プロシージャ内で CHRON EVERY を使用している場合、時間間隔はそのシーケンス内では次のシフトで開始されるので、この機能が役立ちます。

例えば、コマンドがある一定の日に出されるようにスケジュールできます。図 19 に示す CHRON コマンドは、現在から 2011 年の最終日まで休日を除くすべての平日の 8:00 a.m. から 5:00 p.m. まで、1 時間ごとに LOGTSTAT コマンドを出します。LOGTSTAT コマンドは PPT タスク上で実行されます。この CHRON が 8:00 A.M. から 5:00 P.M. の間に入力されると、LOGTSTAT は次の正時ごとに実行されます。これによって、後続日に対するシフトを指定し、部分シフトをその日に実行することができます。次に示すのは、このようなコマンドの例です。

```
CHRON AT=(08:00:00) EVERY=(INTERVAL=(01:00:00 OFF=17:00:00)
      REMOVE=(12/31/11 00:00:00) DAYSWEEK=(WEEKDAY)
      CALENDAR=(NOT HOLIDAY)) COMMAND=LOGTSTAT ROUTE=PPT
```

図 19. CHRON コマンドの例

タスクの選択

スケジュール済みコマンドは、基本プログラム・オペレーター・インターフェース・タスク (PPT) を指定する PPT オプションが指定されている場合以外は、そのタイマー・コマンドを出したタスクと同じタスクで実行されます。タイマー・コマンドを出したタスクが活動状態でなくなっている場合は、スケジュール済みコマンドは実行できません。そのため、タイマー・コマンドは自動タスクから出した方がよいでしょう。タイマー・コマンドを自動タスクから出すには、EXCMD などのコマンド経路指定機能を使って、タイマー・コマンド (AT、EVERY、もしくは AFTER) を自動タスクに送信します。

スケジュール済みコマンドを、継続的に使用可能な自動タスクで実行すると、そのコマンドが確実に実行できるようになります。PPT の代わりに自動タスクを使用すると、PPT が過負荷状態になることがなくなります。また、PPT 上で実行可能なコマンドについての制限を考慮する必要もありません。

タイマー・コマンドの保管と復元

NetView プログラムが終了すると、保管されていないスケジュール済みタイマー・コマンドはすべて打ち切られます。タイマー・コマンドをデータベースに保管しておくと、NetView プログラムの停止と再始動が行われた場合に、重要なスケジュール済みコマンドが打ち切られることはありません。保管されたタイマー・コマンドを再入力する必要はなく、RESTORE コマンドで復元できます。RESTORE コマンドは、DSISVRT (保管・復元) タスクが活動化されたあとに出します。

RESTORE コマンドを出すと、スケジュール済みのコマンドまたはコマンド・リストのうち、NetView プログラムがダウンしていた間に実行される予定であったものは、複数行のメッセージ CNM465I に示されます。ユーザーまたは自動化では、このメッセージからスケジュール済みコマンドの情報を得られます。この情報から、NetView プログラムがダウンしていたためにスキップされたスケジュール済みコマンドを、実行するかどうか判断できます。

図 20 は、RESTORE を出したときに、スキップされたタイマー・コマンドについて示す複数行メッセージです。

```
CNM465I TIMER EVENT CANNOT BE RESTORED - CURRENT TIME PAST EXECUTION
TYPE: AFTER      TIME: 12/15/10 16:42:17
COMMAND: MAJNODES
OP: OPER1              ID: AFTMAJ
```

図 20. スキップされた *TIMER* コマンドを示すメッセージ

サンプル・コードのメッセージには、以下の情報が含まれています。

- 1 行目には、メッセージ ID とテキストが含まれ、NetView プログラムが、そのタイマー・イベントを復元できない理由が示されます。
- 2 行目には、タイマー・コマンドのタイプ (AT、EVERY、または AFTER) が、コマンドの実行された日時とともに示されます。
- 3 行目には、スケジュール済みコマンドが示されます。
- 4 行目には、コマンドを出したオペレーターの ID が示されます。

オペレーターがそのコマンドで PPT パラメーターを使用していた場合、2 行目と 4 行目には、そのことも示されます。

DSISVRT タスクが活動化されたあとには、コマンド・プロシージャで RESTORE コマンドを出して、CNM465I メッセージを待つことができます。メッセージが受信されると、コマンド・プロシージャで各メッセージの情報を調べて、タイマー・コマンドを再度出すかどうかを判断できます。

LIST TIMER および PURGE TIMER

LIST TIMER および PURGE TIMER コマンドは、タイマー・コマンドの管理に役立ちます。LIST TIMER を使用すると、保留状態のタイマー・コマンドのリストを表示できます。PURGE TIMER を使用すると、それを取り消すことができます。これらのコマンドの構文およびパラメーターに関する説明については、NetView プログラム・オンライン・ヘルプを参照してください。

LIST TIMER

LIST TIMER は、処理のために現在時間設定されているすべてのコマンドおよびコマンド・プロシーチャーを関連情報とともにリストします。例えば、図 21 に示している最初のコマンドは、オペレーター OPER1 が DISPSTAT というタイマー ID を指定し、AT、EVERY、および AFTER を使用してスケジュールしたコマンドまたはコマンド・プロシーチャーがあれば、それを表示します。

図 21 に示している 2 番目のコマンドは、スケジュールしたオペレーター またはタイマー ID には関係なく、AT、EVERY、または AFTER によってスケジュールされたすべてのコマンドおよびコマンド・プロシーチャーのリストを表示します。

```
LIST TIMER=DISPSTAT,OP=OPER1
```

```
LIST TIMER=ALL,OP=ALL
```

図 21. LIST TIMER コマンドの例

PURGE TIMER

PURGE TIMER は、現行のスケジュール済みタイマー・コマンドを取り消します。例えば、図 22 に示している最初のコマンドは、OPER1 が DISPSTAT というタイマー ID を指定してスケジュールしたコマンドがあれば、それを除去します。

図 22 に示している 2 番目のコマンドは、OPER1 がスケジュールしたすべての AT、EVERY、および AFTER コマンドを取り消します。全体を取り消すパージ・コマンドを使用する際には、注意が必要です。

```
PURGE TIMER=DISPSTAT,OP=OPER1
```

```
PURGE TIMER=ALL,OP=OPER1
```

図 22. PURGE TIMER コマンドの例

第 12 章 自動タスク

自動タスク は、特殊なオペレーター端末タスク (OST) であり、オペレーターの介入も、NetView 端末も必要としません。他のオペレーターの OST 同様、自動タスクはメッセージを受信し、コマンドおよびコマンド・プロシージャーを処理し、NetView プログラム間セッションを確立します。自動タスクは、NetView プログラム・フルスクリーン自動化機能を使用して、フルスクリーン・コマンドを実行することができます。自動タスクは端末と関連付けられていないので、VTAM がアクティブでないときも実行できます。この理由により、またオペレーターが実行できる作業と同様な作業を実行することができるので、自動タスクは、システムおよびネットワークの多くの自動操作を実現するのに理想的なものです。

自動タスクの定義

自動タスクの ID を定義するための要件は、NetView プログラム・オペレーター ID 定義の要件と同じです。自動タスクは OST なので、DSIOPF またはシステム許可機能 (SAF) 定義を編集し、次に REFRESH コマンドを使用することで、自動タスク OST を NetView プログラムに動的に定義することができます。

サンプル DSIOPF には、自動タスクも含めた NetView プログラム OST の定義ステートメントの例が示されています。ステートメントは、各オペレーターのプロファイルを定義します。次の図 23 に、AUTO1 (NetView プログラム初期設定処理で使用される自動タスク) の定義ステートメントを示します。

```
AUTO1      OPERATOR      PASSWORD=AUTO1
           PROFILEN      DSIPROFC
```

図 23. AUTO1 の定義ステートメント

自動タスクのパスワードによって、不当なユーザーが、自動タスクのオペレーター ID にログオンして NetView にアクセスすることを防止できるようになっています。リソース・アクセス管理機能 (RACF) などの SAF プロダクトを使用して、MVS システムにログオンする前に、パスワードまたはパスワード・フレーズを要求することができます。SAF プロダクトを使用しない場合には、DSIOPF を使用して各自動タスクにパスワードを定義できます。

パスワードまたはパスワード・フレーズを定義して機密とし、お客様の自動タスク ID を保護してください。パスワード検査またはパスワード・フレーズ検査に SAF プロダクトを使用していない場合は、DSIOPF にパスワードを定義しないことによって、他のユーザーが自動タスクのオペレーター ID にログオンできないようにすることもできます。パスワードを定義しない場合にそのオペレーター ID を開始させられるのは、AUTOTASK コマンドのみです。その場合、NetView の AUTOTASK コマンドのコマンド許可を使用して、その使用を制限できます。

自動タスクの活動化

自動タスクとそれ以外の NetView プログラム OST との違いは、オペレーターによる開始方法にあります。オペレーター OST は、NetView オペレーターが端末にログオンすると開始されますが、自動タスクは、オペレーターから AUTOTASK コマンドが発行されたときに開始されます。オペレーターも自動タスクも単一のオペレーター ID で開始できるため、NetView プログラムに定義されたすべての ID について適切なセキュリティー・レベルを保守することが重要です。セキュリティーの問題に関する説明は、「*IBM Tivoli NetView for z/OS*セキュリティー・リファレンス」を参照してください。

CNMSTYLE メンバーの中で AUTOTASK ステートメントを使用して、NetView プログラムの初期化時に自動タスクを開始することができます。詳細については、「*IBM Tivoli NetView for z/OS*アドミニストレーション・リファレンス」を参照してください。

AUTOTASK コマンドの使用

単一の基本プログラム・オペレーター・インターフェース (POI) タスク (PPT) は、NetView プログラムの始動時に開始されます。CNMSTYLE メンバーに自動化テーブルと AUTOTASK が指定されていれば、PPT は NetView プログラムの初期化時にそれらを開始することができます。詳細については、「*IBM Tivoli NetView for z/OS*アドミニストレーション・リファレンス」を参照してください。

オペレーターに正しいレベルの権限が割り当てられていれば、端末から、あるいはコマンド・リストまたはコマンド・プロセッサを使用して、AUTOTASK コマンドを発行できます。

自動タスクと複数コンソール・サポート (MCS) コンソールの関連付け

CNMSTYLE メンバーで AUTOTASK コマンドまたは AUTOTASK ステートメントを使用している場合は、自動タスクと複数コンソール・サポート・コンソールを関連付けることができます。これは、後でタスクをアクティブにしてから行うこともできます。この関連付けを行うと、MCS コンソールで、自動タスクが受信したすべてのメッセージを表示できるようになり、NetView コマンドを受信して、それらを自動タスクに転送することができます。

例えば、自動タスク AUTOMVS を MVS システムと NetView システムの間のインターフェースとして使いたい場合は、次のことを行ってください。

1. NetView プログラムにアクセスする際に使用する MVS コンソール名を決定します。この例では、コンソール名は netvsys2 です。
2. 自動タスクをコンソール netvsys2 と関連付けるために、次のコマンドを発行します。

```
AUTOTASK OPID=AUTOMVS,CONSOLE=netvsys2
```

複数コンソール・サポート (MCS) コンソールがオンラインでなくても、自動タスクとコンソールを関連付けることができます。コンソールがまだアクティブになっていない場合は、コンソールがオンラインに変更されたときに関連付けが完了します。

この目的のために定義した自動タスクを、それ以外の自動化にも使用する場合には、自動タスクに送られたメッセージがすべて MCS コンソールに表示されることを覚えておいてください。

オペレーター宛 (WTO) メッセージが MVS システムからサブシステム・インターフェースを介して NetView システムに送られ、関連する自動タスクを使用して、複数コンソール・サポート (MVS) コンソールにメッセージを送り返す場合、メッセージはシステム・ログおよび Canzlog ログに 2 回表示されます。1 回は元の形式のまま示され、もう 1 回は NetView から複数コンソール・サポート (MCS) コンソールに送られたときの形式で示されます。重複を防ぐために、複数コンソール・サポート (MCS) コンソール専用で使用する専用自動タスクを定義します。

AUTOTASK コマンドについての詳細は、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」を参照してください。

自動タスクの非活動化

以下のいずれかのコマンドで、自動タスクを非活動化できます。

- EXCMD *autoid*,LOGOFF
- %LOGOFF (自動タスクと関連付けられた複数コンソール・サポート (MCS) コンソールから出す場合)

ここで、% はデフォルトの NetView プログラム・サブシステム記述子を表します。このコマンドを実行するには、サブシステムのアドレス・スペースを活動化しておかなければなりません。

注: 複数コンソール・サポート (MCS) コンソールから入力され、接頭部に記述子が指定されているすべてのコマンドは、MCS コンソールの関連付けを解除するために AUTOTASK コマンドまたは DISC コマンドを使用する場合を除いて、自動タスクを自動的に再始動します。

自動タスクが無限ループでスタックしている場合には、EXCMD *autoid*,RESET を発行して、実行中のコマンド・リストを停止してからログオフを試みてください。場合によっては、STOP FORCE=*autoid* を使用して、無限ループに陥った自動タスクを非活動化することもできます。EXCMD および %LOGOFF コマンドは、自動タスクのもとでその他のキューに入れられたコマンド・リストやコマンドと一緒に、LOGOFF コマンドをキューに入れます。これに対して、STOP FORCE は即時コマンドです。

自動タスクによる自動化

このセクションでは、自動タスクを使用して自動化を行うためのさまざまな方法の中からいくつかを選んで説明します。

サブシステムの管理

自動タスクは VTAM プログラムに依存していないため、システムが VTAM を使用せずに稼働している場合に便利です。例えば、NetView プログラムを初期化しているときに、自動タスクを開始して、VTAM などのサブシステムを管理させること

ができます。自動タスクは、必要に応じて、VTAM の活動化や障害からのリカバリーを支援します。自動タスクはオペレーターのコンソールとは関連付けられていませんが、VTAM 定義で APPL ステートメントを必要としており、かつ VTAM にコマンドを発行することもできる点に留意してください。

重要: NetView プログラムが、VTAM よりも前に始動される場合、VTAM がアクティブでないときに開始された自動タスクには、16 進数による採番方式を使用した、特定の VTAM アプリケーション ID (APPLID) が割り当てられます。NetView プログラムでは、割り当てられた APPLID が VTAM の始動時に使用可能であるかどうか判断しないため、APPLID が使用可能であることが前提となっている必要があります。したがって、これらの自動タスクにはそれぞれ、連続番号の付いた VTAM APPL ステートメントを定義しておかなければなりません。採番には 16 進方式が使用され、いずれかの POS 端末で予約済みの番号の次の番号から開始します。例えば、12 台の POS 端末を定義済みで、VTAM を始動する前に 6 つの自動タスクを開始することになっていて、ドメイン名が CNM01 である場合、これらの自動タスクには、CNM0100C、CNM0100D、CNM0100E、CNM0100F、CNM01010 および CNM01011 という APPL 名を定義する必要があります。

非送信請求メッセージの処理

自動タスクは、すべての非送信請求メッセージおよび応答として発行されたコマンドを処理することができます。この方法には、メッセージの処理に関して、以下の 2 つの利点があります。

- ログオンされている特定のユーザーに依存しない
- 処理速度が向上する

例えば、実行時間の長いコマンドを実行しているオペレーターが、非送信請求メッセージを受信した場合、そのメッセージに応答してオペレーターから発行されたコマンドは、実行時間の長いコマンドが処理を終了するまでキューに入れられます。自動タスクが非送信請求メッセージを受信した場合、それに応答するコマンドは即時に実行されます。

自動タスクを継続的に使用できるようにするため、自動化で自動タスクをモニターさせることができます。拡張自動化サンプル・セットでは、自動タスクをモニターする方法の 1 つが示されています。

詳しくは、「661 ページの『拡張自動操作サンプル・セットの使用』」を参照してください。

コマンドの処理

自動化テーブルから送られたコマンドまたはコマンド・プロシーチャーは、自動タスクで処理できます。コマンドの宛先を明示的に選択するために ROUTE キーワードを使用する時に、自動タスクを使用できます。ROUTE キーワードを使用しないで、デフォルトの経路指定を通してコマンドを自動タスクに送ることもあります。これは、例えば、自動タスクによりコマンドを起動しているメッセージが送信請求されるような場合に起こります。

自動タスクのもとで、AFTER、AT、CHRON、または EVERY コマンドを使用してスケジュールされたコマンドおよびコマンド・プロシーチャーは、自動タスクで処

理できます。さまざまな、タイプの異なるリソースをモニターする場合には、いくつかの自動タスクを定義し、開始できます。自動タスクでは、モニター用にそれぞれ異なる時間間隔を使用し、情報記憶用にタスク・グローバル変数のさまざまな集合を使用できます。

NetView コマンド・プロシーチャーは、処理を続行する前にメッセージまたは別のイベントを受け取るのを待つことができます。これは、WAIT 処理とも呼ばれます。自動タスクのもとで WAIT 処理を含んだコマンド・プロシーチャーを実行する際には、注意が必要です。このようなコマンド・プロシーチャーを自動タスクのもとで実行する場合には、プロシーチャー内の WAIT コマンドに、必ずタイムアウト値を指定しておかなければなりません。また、このようなコマンド・プロシーチャーを実行する自動タスクを制限することができます。

WAIT 処理の詳細については、123 ページの『特定のイベントの待機』を参照してください。

タスクの開始

表 3 は、メッセージまたは MSU が NetView 自動化テーブルで自動化されていて、ROUTE キーワードを使用しないか、または宛先に * を指定した場合のコマンドの宛先を示しています。

表 3. 自動タスクを使用してタスクを開始する場合のコマンドの宛先

応答コマンドが発行された原因	コマンドの宛先
非送信請求サブシステム・インターフェース・メッセージ	1 次自動タスク
BNJDSERV から出された MSU	1 次自動タスク

NetView プログラムの初期化時に特定の機能を実行するための自動タスクを定義するには、CNMSTYLE メンバーで function.autotask ステートメントを使用します。詳細については、「IBM Tivoli NetView for z/OS アドミニストレーション・リファレンス」を参照してください。

EXCMD コマンドによる自動タスクへのコマンドの送信

それ以外のタスクでは、EXCMD コマンドを使用して、コマンドを自動タスクに送信することができます。

- オペレーターはその OST 上で自動タスクを使用して、時間がかかる可能性がある作業を行います。
- 自動タスク相互間でコマンドを送信し合い、論理的に順次処理を必要とする作業を行うこともできます。
- 実行速度が遅いコマンドは、自動タスクに送り、より重要な活動を行っているタスクのスループットや応答時間を妨害しないようにすることが可能です。
- 自動タスクは、他のオペレーターが EXCMD コマンドを使用して自動タスクに送ったコマンドを処理できます。

この方法を使用すると、自動タスクにより、以下の作業をサポートする、ある種のバックグラウンド・プロセッサーが作成されます。

- 単一のタスク下で、論理的に順次処理を必要とする作業

- より重要なオペレーター・タスクを妨害するおそれのある作業

ただし、結果を表すメッセージが、NetView プログラムから、元のオペレーターに対して自動的に戻されることはありません。

第 13 章 メッセージ改訂テーブル

本章では、以下のことを説明します。

- NetView メッセージ改訂テーブル (MRT)
- メッセージ改訂テーブルで使えるステートメント
- メッセージ改訂テーブルのコーディング方法
- メッセージ改訂テーブル・ステートメント
- メッセージ改訂テーブル・リストの例
- メッセージ改訂テーブルの使用報告書
- 複数のメッセージ改訂テーブルの管理

メッセージ改訂テーブルの論理のテストについては、146 ページの『メッセージ改訂テーブルのテスト』を参照してください。

WHEN ステートメントおよび REVISE ステートメントは、PIPE EDIT オーダーで構成されます。これらの編集オーダーの使用法については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS*プログラミング:パイプ」を参照してください。

メッセージ改訂テーブルについて

メッセージ改訂テーブル (MRT) を使用すると、MVS メッセージが表示されたり、ログに記録されたり、自動化されたり、シスプレックスを介して経路指定されたりする前に、それらを代行受信できます。メッセージ ID、ジョブ名、およびその他多数のプロパティに基づいて、メッセージについての判断を行うことができます。

次のような、メッセージのさまざまな面を変更できます。

- メッセージ・テキスト
- 色
- 経路コード
- 記述子コード
- 表示属性とシステム・ログ属性

MRT は、NetView プログラムがアクティブでないときであってもアクティブのままです。ただし、SSI アドレス・スペースを必要とします。また、MRT のロード、MRT に対する照会、または統計の収集は、利用できる NetView プログラム・アドレス・スペースがアクティブになっていなければ行えません。

メッセージ改訂テーブル・ステートメントのエレメント

メッセージ改訂の自動化テーブルには以下のエレメントがあります。

- *DoForeignFrom* ステートメント (142 ページの『*DoForeignFrom* ステートメント』)。外部メッセージを MRT で処理することを示すために使用します。
- *END* ステートメント (142 ページの『*END* ステートメント』)。SELECT ステートメントで開始されたセクションをクローズするのに使用します。

- **EXIT** ステートメント (142 ページの『EXIT ステートメント』)。一致するアクションが検出された時点で、以降のメッセージ改訂を停止するのに使用します。
- **NETVONLY** ステートメント (142 ページの『NETVONLY ステートメント』)。表示、ロギング、およびシスプレックス経由の経路指定は抑制しつつ、NetView プログラム自動操作を提供するのに使用します。
- **OTHERWISE** ステートメント (143 ページの『OTHERWISE ステートメント』)。表示、ロギング、およびシスプレックス経由の経路指定は抑制しつつ、NetView プログラム自動操作を提供するのに使用します。
- **REVISE** ステートメント (143 ページの『REVISE ステートメント』)。改訂アクションを組み込むのに使用します。
- **SELECT** ステートメント (144 ページの『SELECT ステートメント』)。一連の **WHEN** ステートメントの導入に使用します。
- **UPON** ステートメント (144 ページの『UPON ステートメント』)。各セクションの導入に使用します。
- **WHEN** ステートメント (145 ページの『WHEN ステートメント』) と **OTHERWISE** ステートメント。これらは、どちらも条件の指定に使用します。
- **%INCLUDE** ステートメント (264 ページの『%INCLUDE ステートメント』)。別々にコーディングされ、保守されているメッセージ改訂テーブルのセクションを組み込むために使用します。これにより、メッセージ改訂テーブルの保守を、いくつかのグループまたは個人で分割して行えます。ユーザーは、自動化テーブル管理機能 (AUTOMAN) を使用して **INCLUDE** 構造体を表示することができます。この機能の詳細については、サンプル CNMSMRT1 を参照してください。

メッセージ改訂テーブルの処理

REVISE MSG コマンドを使用して、メッセージ改訂テーブルの活動化、非活動化、テスト、リスト、または状況の検査を行うことができます。**REVISE MSG** コマンドの使用法についての詳細は、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

メッセージ改訂テーブルの検索

MVS メッセージが発行されると、NetView プログラム SSI コードは高速検索アルゴリズムを使用して、そのメッセージと等価の特定の **UPON** ステートメントを見つけます。次に、その **UPON** の下の条件とアクションが順次に適用されます。あるメッセージが特定の **UPON** 条件のどれとも一致しない場合、このことは迅速に判別されて、そのメッセージは次に **UPON(OTHERMSG)** の下の条件およびアクション (存在する場合) の対象となります。

従属条件もアクションも指定していない **UPON** ステートメントを組み込めば、一致するメッセージの数のみが含まれている **MRT** 報告書を得られます。従属条件もアクションも持たない **UPON** ステートメントはヌル **UPON** と呼ばれます。

UPON(OTHERMSG) も含め、**UPON** ステートメントに従属する条件は順次に検査されます。したがって、共通メッセージと一致する特定のヌル **UPON** ステートメントを組み込んで、それらのメッセージが **UPON(OTHERMSG)** 条件による検査の対象にならないようにすれば、パフォーマンスが向上する可能性があります。

NetView プログラムがアクティブであるときに SSI アドレス・スペースが再始動すると、再始動の時点でアクティブなメッセージ改訂テーブルおよび存在していた改訂テーブルの変数定義が復元されます。

メッセージ改訂テーブルのコーディング

メッセージ改訂テーブルのコーディングには、次の指示および制限が適用されます。

- 1 桁目をアスタリスク (*) にすれば、その後ろにコメントを書くことができます。また、感嘆符 (!) の後ろに続ける場合は、ファイル内のどの場所にでもコメントを記入できます。
- 行を字下げする場合や、キーワード、論理演算子、および括弧を区切る場合には、ブランクを使用できます。

ただし、比較ストリング内でブランクを使用すると、そのブランクはそのストリング内の文字であると見なされます。

- 比較テキストおよび同義語の値の場合の区切り文字には、単一引用符または二重引用符を使用しなければなりません。リテラル内でこの 2 種類の引用符のどちらかを使用している場合は、区切り文字としてはもう一方の引用符を使用してください。
- SELECT、WHEN、または OTHERWISE ステートメントを使用しなくても、REVISE および NETVONLY などのアクションを直接 UPON ステートメントの下に組み込めます。MRT 報告書では、このようなアクションはタイプ OTHERWISE としてラベル付けされます。
- WHEN ステートメントおよび REVISE ステートメントは、PIPE EDIT オーダーで構成されます。これらの編集命令の使用法については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS* プログラミング:パイプ」を参照してください。
- シスプレックス内を必要もないのに伝送されているメッセージがある場合は、NETVONLY および REVISE('N' DELETE) MRT アクションを使用してそのような伝送が行われないようにすると、パフォーマンスを大幅に改善することができます。

経路コードおよび記述子コードの変更

FLGRTCD1 から FLGRTCD16 という名前の 16 の宛先コード FLG オーダーがあり、これらは WQE に定義されている拡張宛先コードの 16 バイトに対応しています。FLGDSCD1 から FLGDSCD4 という名前の 4 つの記述子コード FLG 命令があります。

ROUTEZERO は、すべての経路コードをゼロに設定するのに使用できます。以下に、いくつかの例を示します。

```
* zero out all route codes and set route codes 8 and 16
REVISE (ROUTEZERO "xxxxxxx1" FLGRTCD1 "00000001" FLGRTCD2)
* set descriptor code 2 meaning immediate action required and retain message
* in AMRF
REVISE ("x1xxxxxx" FLGDSCD1 'Y' AMRF)
```

DoForeignFrom ステートメント

外部メッセージは、デフォルトでは MRT で処理されません (このトピックの詳細については、86 ページの『外部メッセージの処理方法』を参照してください)。

DoForeignFrom ステートメントを使用すると、外部メッセージを MRT で処理することを示すことができます。DoForeignFrom ステートメントを指定する場合は、UPON ステートメントの前に置く必要があります。DoForeignFrom ステートメントの形式は、以下のとおりです。

DoForeignFrom = *ALL | *NONE

- DoForeignFrom を *ALL に設定すると、MRT は、(ローカル・システムで発信されたメッセージの他に) シスプレックス内の他のシステムで発信された外部メッセージを処理します。MRT 処理は、WHEN ステートメントで SYSNAME 編集オーダーを使用することにより、特定のシステム名に制限できます。REVISE ステートメントを使用して AUTOMATE の値を Y に設定し、メッセージが NetView プログラムのアドレス・スペースに送信されるようにすることができます。
- DoForeignFrom を *NONE に設定すると、MRT は、ローカル・システムで発信されたメッセージのみを処理します。NetView プログラム SSI では (したがって MRT では)、外部メッセージが MPFLSTxx MVS PARMLIB メンバーの FORNSSI ステートメントで許可されていない場合、外部メッセージを受け取りません。

END ステートメント

END ステートメントは、対応する SELECT ステートメントから始まったセクションをクローズします。

EXIT ステートメント

通常は、メッセージは特定の UPON グループ内のすべてのものと突き合わされます (各 SELECT-WHEN-END エントリーは 1 項目としてカウントされます)。ただし、EXIT アクションとの突き合わせが行われた場合は、同じ WHEN または OTHERWISE の下の残りのアクションは実行されますが、その UPON グループ内の後続の SELECT ステートメントは迂回されます。

NETVONLY ステートメント

NETVONLY ステートメントを使用すると、メッセージには表示の抑制を指示するマークが付けられますが、そのメッセージは NetView プログラムに送信可能になります。NetView プログラム・アドレス・スペースで NETVONLY ステートメントが受信されると、メッセージは自動化タスクに実行依頼されるか、またはコマンド応答として経路指定されます。NETVONLY ステートメントは、SSI を介して、メッセージを常に NetView プログラムのキューに入れます。

OTHERWISE ステートメント

OTHERWISE ステートメントは WHEN ステートメントと似ていますが、異なる点は、条件を持たないこと、およびある特定の SELECT ステートメント下のすべての WHEN ステートメントの後ろになければならないことです。

REVISE ステートメント

REVISE ステートメントの後ろには、単一の編集スクリプトを囲む括弧のセットが続きます。このようなスクリプトは改訂スクリプトと呼ばれます。この場合に限り(つまり、WHEN、EDIT、または ACQUIRE 条件などの場合とは異なり)、テキスト処理に関して次のような例外的な処理が行われます。すなわち、メッセージ・テキストを変更する出力オーダーがない場合は、テキスト全体の複製が出力メッセージに入れられます(他のスクリプトの場合は、結果としてヌル・テキストが出力されます)。複数の REVISE ステートメントを任意のグループに入れることができ、各ステートメントの動作は直前の改訂結果に基づきます。後続の SELECT グループがメッセージを再検査するときには、先行する REVISE アクションのアクション結果が確認されます。

以下に、REVISE ステートメントの例を示します。

```
REVISE("Cr" COLOR)                * turn msg red
REVISE("CY" COLOR "ABCDEFGH" autotoken) * turn msg yellow and set autotoken
REVISE(ROUTEZERO)                 * set all routecodes to false/zero
REVISE('1xxxxxxx' FLGDSCD2)       * set descriptor code 9 to true
REVISE('1xxxxxx0' FLGDSCD3)       * set descriptor code 17 to true
                                   * and descriptor code 24 to false

REVISE('1xxxxxxx' FLGRTCD4)        * set route code 25 to true
REVISE("cr hr" color )             * turn msg color to red with reverse video
REVISE("ct hu" color )             * turn msg color to turquoise and underline
REVISE ('N' automate)              * do not automate this message
REVISE ('Y' automate)              * automate this message
REVISE ('Y' AMRF)                  * retain Action message in AMRF
REVISE ('N' DISPLAY)               * do not show message at the console
REVISE ('Y' DISPLAY)               * show message at the console
REVISE ('Y' BROADCAST)             * send message to all active consoles
REVISE ('Y' PROG)                  * for programmer information (route code 11)
REVISE ('N' SYSLOG)                * do not write this message to the system log
REVISE ('80'x SYSLOG)              * write this message to the system log
REVISE ('Y' DELETE)                * totally delete message

* turn msg blue and append "SHOULD BE BLUE" to the end of message
REVISE("CB" COLOR 1.* 1 "SHOULD BE BLUE" NW)
* same as previous example except using ALL
REVISE("CB" COLOR ALL 1 "SHOULD BE BLUE" NW)
REVISE ( ALL UPCASE )               * UPPER case the entire message
revise (MSGID 1 "changed message" NW) * Keep msgid and append "changed message"
REVISE("0000000x" FLGRTCD1)        * turn off routecodes 1-7 leave 8 as before
REVISE ( "WHOKNOWS" CONSNAME)      * send message to console with the name WHOKNOWS
```

PIPE EDIT オーダーの使用法については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS*プログラミング:パイプ」を参照してください。

注: MVS には、テキスト出力は 127 文字までという制限があります。これより長いメッセージが切り捨てられた場合、MRT は警告を出さず、条件も提供しません。

SELECT ステートメント

SELECT ステートメントは、一連の WHEN ステートメントを導入します。WHEN ステートメントの後ろには、OTHERWISE ステートメントと END ステートメントがなければなりません。SELECT ステートメントには、引数は含まれません。

UPON ステートメント

UPON ステートメントは、各セクションを導入する最上位の条件ステートメントです。条件には、次の 4 つのタイプがあります。

- MSGID (可能な文字数の範囲は 1 から 12 文字)
- JOBNAME (可能な文字数の範囲は 1 から 8 文字)
- PREFIX (常に 3 文字)
- OTHERMSG

これらは上記の順序でテストされ、最初の 3 つの条件は常にリテラルと比較されます。可能な文字数は、MSGID リテラルは 1 文字から 12 文字まで、JOBNAME リテラルは 1 文字から 8 文字までです。PREFIX リテラルは常に 3 文字です。

以下に、条件の例を示します。

```
UPON (msgid = 'CNM233I' | JOBNAME='VTAM' | prefix = 'IST')
UPON (MSGID="TST102A" |
      MSGID="TST102B" |
      MSGID="TST102C" |
      MSGID="TST102D" )
```

UPON ステートメントのタイプのいずれかと一致したメッセージは、下位の UPON ステートメントとは比較されません。例えば、メッセージ CNM233I が上記のテーブルに渡された場合、このメッセージに対する JOBNAME または PREFIX 条件の比較は実行されず、UPON(OTHERMSG) セクションの下にリストされているステートメントにこのメッセージが実行依頼されることもありません。MSGID の UPON ステートメントがテーブル定義内で物理的に最初のステートメントでなかったとしても、MSGID が最初にテストされることに注意してください。

性能低下を避けるために、UPON(OTHERMSG) ステートメントの使用は制限してください。

ある特定の UPON ステートメント内で、複数の条件を OR シンボル (|) で結合することは可能ですが、AND 結合は不可です。

各 UPON ステートメントの従属ステートメントとして、タイプが SELECT、REVISE、NETVONLY、および EXIT のステートメントをゼロ個以上指定できます。このステートメント・グループは *UPON グループ* と呼ばれ、指定されたときの順序と同じ順序で評価されます。

メッセージがある UPON ステートメントと一致し、その UPON グループのアクションの対象となった場合、MRT はそのメッセージに対してはそれ以上のアクションを実行しません。特に、このようなメッセージは他の UPON 条件とは比較されません。

MLWTO メッセージの場合、最初の行しか検査されないことに注意してください。

WHEN ステートメント

WHEN ステートメントは SELECT ステートメントに従属し、その後ろには常に括弧で囲まれた式が続きます。それぞれの WHEN ステートメントの後ろには、ゼロ個以上のアクション・ステートメントのセットが続き、さらにその後ろに次の WHEN または OTHERWISE ステートメントが続きます。これは *WHEN* グループと呼ばれます。式は、等号 記号 (=) または不等号 を表す記号のセット (≠) のいずれかで区切られる編集スクリプトの対です。この 2 つのスクリプトがメッセージに対して実行され、結果が比較され、先行ブランクと末尾ブランクまたはヌル値が除去されます。両者が等しい (または等しくない (分離文字の値による)) と、メッセージはその WHEN ステートメントと一致したと見なされます。このようなメッセージは、その WHEN グループに属すアクション・ステートメントの処理対象となり、同じ SELECT ステートメント下の他の WHEN ステートメントとは比較されず、OTHERWISE ステートメントとは突き合わされません。

以下に、WHEN ステートメントの例を示します。

```
WHEN (msgid right 1 = 'A')          * when action message
WHEN (MSGID SUBSTR 5.* RIGHT 1 = 'E') * when error message
WHEN (SYSLOG yesno = 'Yes')         * when SYSLOG is on. 'Yes' is case sensitive.
WHEN (SYSLOG yesno = 'No')         * when SYSLOG is off. 'No' is case sensitive.
WHEN (SYSLOG = '80'x)              * when SYSLOG is on
WHEN (SYSLOG = '00'x)              * when SYSLOG is off
WHEN (FLGRTCD1 SUBSTR 2.1 = '1')    * when routing code 2 is on
```

PIPE EDIT オーダーの使用法については、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS*プログラミング:パイプ」を参照してください。

メッセージ改訂テーブルの例

以下に、MRT の例を示します。

```
UPON ( MSGID = 'IEA404A'          ! SEVERE WTO BUFFER SHORTAGE - 100% FULL
      | MSGID = 'IRA200E'          ! AUXILIARY STORAGE SHORTAGE
      | msgID = 'DSI125I')         ! CRITICAL STORAGE SHORTAGE FOR NCCF

      REVISE('CR HR' COLOR)       ! make msgs red/reverse
* note, when adding to text, be sure to put text in there first!
      SELECT
        WHEN (MSGID = 'DSI125I')   !
          REVISE("N" AUTOMATE)     ! do not try to automate dsi125
          EXIT                     ! skip further revision, too.
        OTHERWISE
      END
      REVISE('11xx0xxx' FLGRTCD1   ! send to Rt Cd 1,2 but not 4 ...
            1.* 1 "919-555-5677") ! and add my phone number to text

* Some VTAM related messages ...
UPON (MSGid = 'CNM233I' |MSGID = 'CNM234I' |MSGID = 'CNM235I'
      |MSGID = 'CNM385I' |MSGID = 'CNM386I' |MSGID = 'CNM435I'
      |MSGID = 'CNM439I'
      |JOBNAME = 'VTAM' | preFix = 'IST')
      SELECT
        WHEN (MSGID LEFT 3 = 'CNM') ! like "prefix" for WHEN statment
          REVISE("CP" COLOR)        ! nv msgs above turn pink
        OTHERWISE
          REVISE("CP HR" COLOR)     ! others also underscored
          NETONLY                   ! sys consoles not to see these
      END
```

```

UPON (PREFIX = 'DSI' | prefix = 'CNM' | prefix = 'DWO')
    revise ('xxxx11xx' flgRtCd1) ! in addition to route codes already
                                ! set, add 5 and 6

! Despite being specified first, the prefix condition above is
! evaluated AFTER all MSGID conditions. Due to the following,
! DSI802A & 803A are not affected by the Rt Cd 5 & 6 revision.
UPON (MSGID = 'DSI802A'          ! changing text of these msgs
      | MsgID = 'DSI803A')
    revise(w1 1 msgid nw          ! put in reply ID and msgid
           "reply CLOSE or MSG" nw)! list valid cmds
! note: above revision sets text, so text placed by the edit is the
! ONLY text in the resulting message.
SELECT
    WHEN(MSGID RIGHT 2 = '3A')    ! for one of the above MSGIDs,...
        REVISE(1.* 1 'ONLY!')    ! make additional text changes
    OTHERWISE
END
SELECT
    WHEN(W3 ^= '&DOMAIN.')        ! msg from other NetView?
        NETVONLY                  ! steal msg from MVS, give only
                                ! to this NetView, for automation
! Be sure your automation does something with these msgs!
    OTHERWISE
END
UPON(OTHERMSG)                  ! more performance cost for these tests...
SELECT
    WHEN (MSGID RIGHT 1 = 'A')    ! action msg
        REVISE("HB" COLOR)       ! keep same color, add blink
    WHEN (MSGID RIGHT 1 = 'E')    !
        REVISE('xx1xxxxx' FLGRtCd2 ! add Rt Cd 11 to any present
               'CY' color)        ! and color
    OTHERWISE
END

```

メッセージ改訂テーブルの使用報告書

REVISE MSG REPORT コマンドを使用すると、アクティブな改訂テーブルについての統計および使用状況の情報を収集できます。正常に実行された場合は、BNHRVaaaI メッセージが発行されます。REPORT キーワードを MEMBER オペランドと一緒に使用した場合は、表示される情報は、置き換えられたテーブルとそのテーブルが置き換えられた時刻に関する情報となります。

REVISE MSG コマンドの使用法についての詳細は、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

メッセージ改訂テーブルのテスト

以下のステップに従って、メッセージ改訂テーブルをテストし、REVISE ステートメントに指定した経路コード、記述子コード、およびコンソール名が予想通りに機能していることを検証できます。

1. WTO コマンドを使用してメッセージを発行します。例えば、以下のようになります。

```
WTO TST125A First test message
```

2. メッセージ改訂テーブルでのメッセージの処理を許可し、コンソール名、記述子コード、および経路コードを変更します。

3. NetView プログラム自動化テーブルを使用して REXX ルーチンを呼び出し、記述子コード、経路コード、およびコンソール名についての情報を印刷します。以下に、自動化テーブル項目、メッセージ改訂テーブルのセクション、および自動化テーブル項目が呼び出す REXX の例をそれぞれ示します。

- 自動化テーブル項目

```
IF MSGID = 'TST' . THEN
  HOLD(N) EXEC(CMD('RexxExec') ROUTE(ONE ConsoleName));  * where ConsoleName
  *is the NetView console name obtained using GETCONID
```

- メッセージ改訂テーブルのセクション

```
UPON (MSGID="TST125A")
SELECT
When (MSGID="TST125A")
  REVISE('1xxxxxxx' FLGDSCD1)
  REVISE('xxxxxxx1' FLGRTCD2)
  REVISE ("ConsoleName" CONSNAME)  * where ConsoleName is a valid Console Name
Otherwise
END
```

- 自動化テーブル項目によって呼び出される REXX の例

```
/* RexxExec */
say ROUTCDE()
say "desc code =" DESC()
say "CONSNAME=" SYSCONID()
exit
```

WTO コマンドについての詳細は、NetView プログラムのオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

第 14 章 コマンド改訂テーブル

本章では、以下のことを説明します。

- NetView コマンド改訂テーブル (CRT)
- コマンド改訂テーブルのコーディング方法
- コマンド改訂テーブル・ステートメント
- コマンド改訂テーブル・リストの例
- コマンド改訂テーブルの使用報告書
- コマンド改訂テーブルの論理のテスト

注: MVS コマンド改訂機能は、既存の MVS コマンド管理機能を置き換えるものです。MVS コマンド改訂機能へのマイグレーションについては、「*IBM Tivoli NetView for z/OS*インストール:マイグレーション・ガイド」を参照してください。

コマンド改訂テーブルについて

コマンド改訂テーブル (CRT) により、MVS コマンドが処理される前に MVS コマンドをインターセプトできます。コマンド・ソースには、MVS コンソールおよび NetView MVS コマンドが含まれます。

CRT は、JCL COMMAND パラメーターを使用することによって、または MGCRC マクロまたは直接の SVC 34 を使用する任意のプログラムによって、MVS コンソールのコマンド行に入力されたすべてのテキストを SDSF システム・コマンドとしてインターセプトします。入力されたテキストは、CRT 処理による変更または宛先変更が行われるまでは、有効な MVS コマンドであるかどうか明確ではありません。ただし、REISSUE コマンドを CRT 処理の一部として使用できます。その後のコマンドは CRT アクションから免除されます。NETVONLY ステートメントについては、156 ページの『NETVONLY ステートメント』を、REISSUE ステートメントについては、オンライン・ヘルプを参照してください。コマンドに関する決定を、ソース、コマンド verb、およびコマンド・パラメーターに基づいて行うことができます。

コマンド・テキストに変更を加え、コマンド発行者に対するメッセージを作成し、コマンドを実行するか、コマンドを抑制できます。より複雑なアクションの場合は、コマンドを NetView プログラムに転送することもできます。CRT は、NetView プログラムがアクティブでないときであってもアクティブのままでいられますが、SSI アドレス・スペースを必要とします。ただし、CRT のロード、CRT に対する照会、または統計の収集は、利用できる NetView アドレス・スペースがアクティブになっていなければ行えません。

コマンド改訂テーブル・ステートメントの要素

コマンド改訂自動化テーブルには以下の要素があります。

- UPON ステートメント。各セクションの導入に使用します。
- SELECT ステートメント。一連の WHEN ステートメントの導入に使用します。

- *WHEN* ステートメント と *OTHERWISE* ステートメント。これらは、どちらも条件の指定に使用します。
- *END* ステートメント。*SELECT* ステートメントで開始されたセクションをクローズするのに使用します。
- *REVISE* ステートメント。改訂アクションを組み込むのに使用します。
- *EXIT* ステートメント。一致するアクションが検出された時点で、以降のコマンド改訂を停止するのに使用します。
- *NETVONLY* ステートメント。*NetView* 自動操作を提供するのに使用します。
- *WTO* ステートメント。コマンドの発行元コンソールへのメッセージを生成します。*INTERNAL*、*INSTREAM*、*INTIDS*、または *HC* コンソールから発信されたコマンドの場合、メッセージは *SYSLOG* にのみ書き込まれます。
- *%INCLUDE* ステートメント。別々にコーディングされ、保守されているコマンド改訂テーブルのセクションを組み込むために使用します。これにより、コマンド改訂テーブルの保守を、いくつかのグループまたは個人で分割して行えます。ユーザーは、自動化テーブル管理機能 (*AUTOMAN*) を使用して *INCLUDE* 構造体を表示することができます。詳細については、サンプル *CNMSCRT1* を参照してください。

コマンド改訂テーブルの処理

REVISE CMD コマンドを使用して、コマンド改訂テーブルの活動化、非活動化、テスト、リスト、または状況の検査を行うことができます。*REVISE CMD* コマンドの使用法についての詳細は、*NetView* のオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

コマンド改訂テーブルの検索

MVS コマンドが発行されると、*NetView SSI* コードは高速検索アルゴリズムを使用して、そのコマンドと等価の特定の *UPON* ステートメントを見つけます。次に、その *UPON* の下の条件とアクションが順次に適用されます。コマンドが特定の *UPON* 条件のどれとも一致しない場合、このことは迅速に判別されて、そのコマンドは *UPON(OTHERCMDS)* 条件に基づく条件およびアクションの対象となり、次に *UPON(ALLCMDS)* 条件に基づく条件およびアクションの対象となります。

従属条件もアクションも指定していない *UPON* ステートメントを組み込めば、一致するコマンドの数のみが含まれている *CRT* 報告書を得られます。従属条件もアクションも持たない *UPON* ステートメントはヌル *UPON* と呼ばれます。

UPON ステートメントに従属する条件は順次に検査されます。したがって、共通コマンドと一致する特定のヌル *UPON* ステートメントを組み込んで、それらのコマンドが *UPON(OTHERCMDS)* または *UPON(ALLCMDS)* 条件による検査の対象にならないようにすれば、パフォーマンスが向上する可能性があります。

NetView プログラムがアクティブであるときに *SSI* アドレス・スペースが再始動すると、再始動の時点でアクティブなコマンド改訂テーブルおよび存在していた改訂テーブルの変数定義が復元されます。

コマンド改訂テーブルのコーディング

コマンド改訂テーブルのコーディングには、次の指示および制限が適用されます。

- 1 桁目をアスタリスク (*) にすれば、その後ろにコメントを書くことができます。また、感嘆符 (!) の後ろに続ける場合は、ファイル内のどの場所にでもコメントを記入できます。
- 行を字下げする場合や、キーワード、論理演算子、および括弧を区切る場合には、ブランクを使用できます。比較ストリング内でブランクを使用すると、そのブランクはそのストリング内の文字であると見なされます。
- 比較テキストおよび同義語の値の場合の区切り文字には、単一引用符または二重引用符を使用しなければなりません。リテラル内でこの 2 種類の引用符のどちらかを使用している場合は、区切り文字としてはもう一方の引用符を使用してください。
- SELECT、WHEN、または OTHERWISE ステートメントを使用しなくても、REVISE、NETVONLY、および WTO などのアクションを直接 UPON ステートメントの下に組み込めます。CRT 報告書では、以下のアクションはタイプ OTHERWISE としてラベル付けされます。
- PIPE EDIT オーダーは、WHEN、REVISE、および WTO ステートメントと一緒に使用できます。詳しくは、「157 ページの『編集オーダー』」を参照してください。
- 一度に処理できる最大バイト数は 256 です。C2X 編集命令を使用する場合、16 進数に変換できるのは 128 バイトのデータのみです。操作の結果が 256 バイトを超えるデータになる場合、256 バイトを超えるデータは切り捨てられます。
- デフォルトのワード区切り文字は以下のとおりです。PIPE EDIT 句のワード区切り文字について詳しくは、「IBM Tivoli NetView for z/OS プログラミング:パイプ」資料の PIPE EDIT のグローバル命令 PARSE の説明を参照してください。
(space) , = ()

コマンド改訂テーブルのステートメント

CRT では、以下のステートメントを使用できます。

- 152 ページの『TRACKING.ECHO ステートメント』
- 152 ページの『ISSUE.IEE295I ステートメント』
- 153 ページの『UPON ステートメント』
- 154 ページの『SELECT ステートメント』
- 154 ページの『WHEN ステートメント』
- 155 ページの『OTHERWISE ステートメント』
- 155 ページの『END ステートメント』
- 155 ページの『REVISE ステートメント』
- 156 ページの『NETVONLY ステートメント』
- 157 ページの『WTO ステートメント』

TRACKING.ECHO ステートメント

デフォルトでは、CRT によってコマンドが変更されると、z/OS オペレーティング・システムは追加のコマンド・エコーを発行します。このことは、システム・ログに記録されます。この新しいメッセージは、CRT による変更後のコマンドを反映します。

注:

1. このステートメントを有効にするには、システム APAR OA28464 が必要です。
2. TRACKING.ECHO ステートメントを指定する場合は、すべての UPON ステートメントの前に指定する必要があります。

TRACKING.ECHO ステートメントでは、以下の構文を使用します。

TRACKING.ECHO



パラメーター

説明

YES

追加のエコーを許可するには、YES を指定します。これがデフォルトです。

NO 追加のエコーを抑止するには、NO を指定します。

ISSUE.IEE295I ステートメント

デフォルトでは、z/OS オペレーティング・システムは、CRT で直接行った文書変更に対して、追加の IEE295I (複数行メッセージ) を発行します。このことは、システム・ログに記録されます。メッセージは、NETVONLY アクションを指定した場合には発行されません。この新しいメッセージは、CRT による変更前後のコマンドを反映します。

注:

1. このステートメントを有効にするには、システム APAR OA28464 が必要です。
2. ISSUE.IEE295I ステートメントを指定する場合は、すべての UPON ステートメントの前に指定する必要があります。

ISSUE.IEE295I ステートメントでは、以下の構文を使用します。

ISSUE.IEE295I



パラメーター

説明

YES

トラッキング・メッセージを許可するには、YES を指定します。これがデフォルトです。

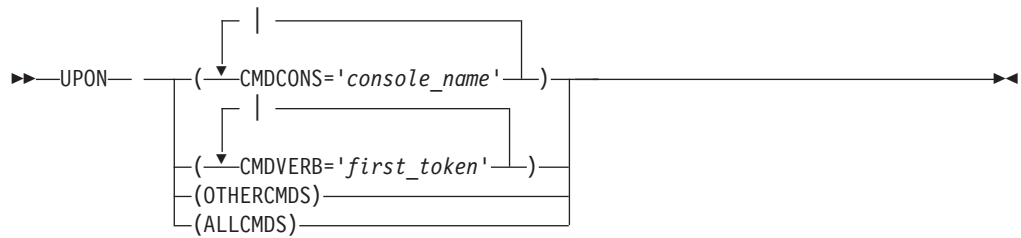
NO トラッキング・メッセージを抑止するには、NO を指定します。

UPON ステートメント

UPON ステートメントは、各セクションを導入する最上位の条件ステートメントです。

UPON ステートメントでは、以下の構文を使用します。

UPON



パラメーター

説明

CMDCONS='console_name'

コマンドを発行しているコンソールの名前。

CMDVERB='first_token'

ブランクまたはコンマで区切られた最初のトークンの値。可能な文字数の範囲は 1 から 12 文字です。コマンド同義語は解決されません。

OTHERCMDS

先行の条件に一致しないすべてのコマンド。

ALLCMDS

すべてのコマンド。

使用法:

1. CMDCONS および CMDVERB 条件には、複数の値を使用できます。あるいは、異なる値を使用して複数回コーディングすることもできます。
2. OTHERCMDS および ALLCMDS 条件をコーディングするのは、テーブル内で一度のみです。
3. パラメーターは以下の順序で検査されます。
 - a. CMDCONS
 - b. CMDVERB
 - c. OTHERCMDS
 - d. ALLCMDS
4. UPON ステートメントのタイプのいずれかと一致したコマンドは、下位の UPON ステートメントとは比較されません。

5. 性能低下を避けるために、UPON(OTHERCMDS) および UPON(ALLCMDS) ステートメントの使用は制限してください。
6. ある特定の UPON ステートメント内で、複数の条件を OR シンボル (|) で結合することは可能ですが、AND 結合は不可です。
7. 各 UPON ステートメントの従属ステートメントとして、タイプが SELECT、REVISE、NETVONLY、および WTO のステートメントをゼロ個以上指定できます。このステートメント・グループは *UPON グループ* と呼ばれ、指定されたときの順序と同じ順序で評価されます。
8. OTHERCMD は OTHERCMDS の同義語です。ALLCMD は ALLCMDS の同義語です。

例:

以下に、UPON 条件の例を示します。

```
UPON (CMDVERB ='CONTROL' | CMDVERB='K' | CMDCONS='SAMSOWN')
UPON(CMDVERB='FORCE')
UPON(OTHERCMDS)
```

SELECT ステートメント

SELECT ステートメントは、一連の WHEN ステートメントを導入します。WHEN ステートメントの後ろには、OTHERWISE ステートメントと END ステートメントがなければなりません。SELECT ステートメントには、引数は含まれません。

SELECT ステートメントでは、以下の構文を使用します。

SELECT

▶—SELECT—▶

WHEN ステートメント

WHEN ステートメントは SELECT ステートメントに従属し、その後ろには常に括弧で囲まれた式が続きます。それぞれの WHEN ステートメントの後ろには、ゼロ個以上のアクション・ステートメントのセットが続き、さらにその後ろに次の WHEN または OTHERWISE ステートメントが続きます。これは *WHEN グループ* と呼ばれます。式は、等号 記号 (=) または不等号 を表す記号のセット (≠) のいずれかで区切られる編集スクリプトの対です。この 2 つのスクリプトがコマンドに対して実行され、結果が比較され、先行ブランクと末尾ブランクまたはヌル値が除去されます。両者が等しい (または等しくない (分離文字の値による)) と、コマンドはその WHEN ステートメントと一致したと見なされます。このようなコマンドは、その WHEN グループに属すアクション・ステートメントの処理対象となり、同じ SELECT ステートメント下の他の WHEN ステートメントとは比較されず、OTHERWISE ステートメントとは突き合わされません。

WHEN ステートメントでは、以下の構文を使用します。

WHEN

▶▶—WHEN— ——(—*action_statement*—)——▶▶

パラメーター

説明

action_statement

指定可能な編集オーダー。指定可能な編集オーダーのリストについては、158 ページの表 4 を参照してください。

例:

以下に、WHEN ステートメントの例を示します。

```
WHEN(CONSAUTH = 'I')           ! console's authority is "I/O"
WHEN(CONSNAME left 3 = 'MST')  ! console's name begins with "MST"
WHEN(WORD 2 = '')              ! command entered with no arguments
WHEN(ASTYPE = 'D')             ! command from USS persistent procedure
WHEN(CMDVERB = 'SWITCH')       ! switch command issued -- or ....
WHEN(CMDVERB = 'I')            ! switch command issued
```

PIPE EDIT オーダーの使用法については、NetView のオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS プログラミング:パイプ*」を参照してください。

OTHERWISE ステートメント

OTHERWISE ステートメントは WHEN ステートメントと似ていますが、異なる点は、条件を持たないこと、およびある特定の SELECT ステートメント下のすべての WHEN ステートメントの後ろになければならないことです。

OTHERWISE ステートメントでは、以下の構文を使用します。

OTHERWISE

▶▶—OTHERWISE——▶▶

END ステートメント

END ステートメントは、対応する SELECT ステートメントから始まったセクションをクローズします。

END ステートメントでは、以下の構文を使用します。

END

▶▶—END——▶▶

REVISE ステートメント

REVISE ステートメントは、コマンド・ストリング (テキスト) を変更します。

REVISE ステートメントでは、以下の構文を使用します。

REVISE

▶▶—REVISE— ———(—*action_statement*—)————▶▶

パラメーター

説明

action_statement

指定可能な編集オーダー。指定可能な編集オーダーのリストについては、158 ページの表 4 を参照してください。

使用法:

1. REVISE ステートメントの後ろには、単一の編集スクリプトを囲む括弧のセットが続きます。このスクリプトは、改訂スクリプトと呼ばれます。コマンド・テキストを変更する出力オーダーがない場合は、テキスト全体が出力コマンドに複製されます。
2. 複数の REVISE ステートメントを任意のグループに入れることができ、各ステートメントの動作は直前の改訂結果に基づきます。後続の SELECT グループがコマンドを再検査するときには、先行する REVISE アクションのアクション結果が確認されます。

制限:

MVS では、コマンド当たりの文字数が 126 文字までに制限されています。これより長いコマンドが切り捨てられた場合、CRT は警告を出さず、条件も提供しません。

例:

以下に、REVISE ステートメントの例を示します。

```
REVISE(ALL 1 " ,AREA=BLD410" N) ! adding a parameter to a command
REVISE('Y' DELETE) ! command is deleted (forbidden)
REVISE(ALL 1 " ,L" NEXT ) ! adding a parameter
```

PIPE EDIT オーダーの使用法については、NetView のオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS プログラミング:パイプ*」を参照してください。

NETVONLY ステートメント

NETVONLY ステートメントは、NetView アドレス・スペースで実行される REXX プロシージャ (コマンド) を指定します。プロシージャのパラメーターは、CRT に実行依頼されるコマンド・テキストになり、NETVONLY アクションの前に CRT で行われるすべての改訂が含まれます。

NETVONLY ステートメントでは、以下の構文を使用します。

NETVONLY

▶▶—NETVONLY=*procedure*————▶▶

パラメーター

説明

procedure

1 から 8 文字の REXX プロシージャー名

使用法:

1. NetView アドレス・スペースが *procedure* (コマンド) を受け取ると、このコマンドはコマンド改訂環境の自動化タスクに実行依頼されます。自動化タスクは、CNMSTYLE メンバーの ?MVSCmdRevision ステートメントで識別されます。NETVONLY ステートメントは、SSI を介してコマンドを NetView プログラムのキューに入れます。
2. 各 WHEN ステートメントまたは OTHERWISE ステートメントには、1 つの NETVONLY ステートメントのみ指定できます。
3. コーディング技法の例については、CNMSRVMC サンプルを参照してください。
4. 各 NETVONLY コマンド・プロシージャーでは、コマンドを排他的に実行するため、コマンド改訂環境の自動化タスクによって仮想 OST (VOST) が接続されます。VOST を使用することで、NETVONLY ステートメントで指定されたプロシージャーを同時に実行できます。したがって、コマンドによって出され、Canzlog ログに記録されるすべてのメッセージでは、VOST フラグがオンになります。

WTO ステートメント

WTO ステートメントは、コマンドの発行元コンソールに送信される WTO メッセージのテキストを作成します。

WTO ステートメントでは、以下の構文を使用します。

WTO

▶▶WTO— ——(-*action_statement*-)————▶▶

パラメーター

説明

action_statement

指定可能な編集オーダー。指定可能な編集オーダーのリストについては、158 ページの表 4 を参照してください。

制限事項:

1. 経路コード、記述子コード、および WTO のその他のパラメーターは設定できません。
2. 126 文字を超えるテキストは切り捨てられます。

編集オーダー

158 ページの表 4 に、WHEN または REVISE の指定と組み合わせて使用できる編集オーダーを示します。

表 4. 編集オーダー

編集オーダー	説明																						
ALL	コマンドのテキスト全体を使用することを示します。これは「1.*」と同じです。																						
ASID	コマンドの MVS 発信元のアドレス・スペース ID を示します (2 バイトの 2 進値)。																						
ASTYPE	どのようにアドレス・スペースが開始されたかを示します (ジョブ・タイプ)。 <table> <tr> <th>値</th><th>説明</th></tr> <tr> <td>D</td><td>USS永続プロシージャー。 アドレス・スペースに、JOB に該当する、開始されたプログラムの名前が含まれています。ただし、OpenMVSアドレス・スペース・ブロックが存在している場合、それは特殊目的USS永続プロシージャーを示します。</td></tr> <tr> <td>J</td><td>アドレス・スペースは JOB です。</td></tr> <tr> <td>N</td><td>アドレス・スペースは、オペレーティング・システム初期化 (NIP) 処理の間に開始されたシステム・アドレス・スペースです。</td></tr> <tr> <td>S</td><td>アドレス・スペースは開始タスク (STC) です。</td></tr> <tr> <td>T</td><td>アドレス・スペースはタイム・シェアリング・ユーザー (TSO) です。</td></tr> <tr> <td>U</td><td>アドレス・スペースは、USS フォークまたは作成プロシージャーです。</td></tr> <tr> <td>*</td><td>エラー: コマンドを発信したアドレス・スペースがクローズしました。</td></tr> <tr> <td>?</td><td>エラー: データが矛盾します (一時的な状態である可能性があります)。</td></tr> <tr> <td>!</td><td>エラー: データが矛盾します。</td></tr> <tr> <td>></td><td>エラー: 発生してはならないエラーが発生しました。</td></tr> </table>	値	説明	D	USS永続プロシージャー。 アドレス・スペースに、JOB に該当する、開始されたプログラムの名前が含まれています。ただし、OpenMVSアドレス・スペース・ブロックが存在している場合、それは特殊目的USS永続プロシージャーを示します。	J	アドレス・スペースは JOB です。	N	アドレス・スペースは、オペレーティング・システム初期化 (NIP) 処理の間に開始されたシステム・アドレス・スペースです。	S	アドレス・スペースは開始タスク (STC) です。	T	アドレス・スペースはタイム・シェアリング・ユーザー (TSO) です。	U	アドレス・スペースは、USS フォークまたは作成プロシージャーです。	*	エラー: コマンドを発信したアドレス・スペースがクローズしました。	?	エラー: データが矛盾します (一時的な状態である可能性があります)。	!	エラー: データが矛盾します。	>	エラー: 発生してはならないエラーが発生しました。
値	説明																						
D	USS永続プロシージャー。 アドレス・スペースに、JOB に該当する、開始されたプログラムの名前が含まれています。ただし、OpenMVSアドレス・スペース・ブロックが存在している場合、それは特殊目的USS永続プロシージャーを示します。																						
J	アドレス・スペースは JOB です。																						
N	アドレス・スペースは、オペレーティング・システム初期化 (NIP) 処理の間に開始されたシステム・アドレス・スペースです。																						
S	アドレス・スペースは開始タスク (STC) です。																						
T	アドレス・スペースはタイム・シェアリング・ユーザー (TSO) です。																						
U	アドレス・スペースは、USS フォークまたは作成プロシージャーです。																						
*	エラー: コマンドを発信したアドレス・スペースがクローズしました。																						
?	エラー: データが矛盾します (一時的な状態である可能性があります)。																						
!	エラー: データが矛盾します。																						
>	エラー: 発生してはならないエラーが発生しました。																						
C2B	入力の 2 進数をプール値ストリングに変換します (EBCDIC の値「0」および「1」)。																						
C2D	入力を10進数を表すストリングに変換します。																						
C2X	入力を16進表記を表すストリングに変換します。																						
CMDX	IEZVX101制御ブロックの先頭から88 (X'58')バイトを入力します。																						
CONSAUTH	コマンドを実行するコンソールの権限を示します: <table> <tr> <th>値</th><th>説明</th></tr> <tr> <td>M</td><td>マスター</td></tr> <tr> <td>I</td><td>入出力</td></tr> <tr> <td>S</td><td>システム</td></tr> <tr> <td>C</td><td>Console</td></tr> <tr> <td>X</td><td>情報を使用できないか、矛盾するデータが見つかりました。</td></tr> </table>	値	説明	M	マスター	I	入出力	S	システム	C	Console	X	情報を使用できないか、矛盾するデータが見つかりました。										
値	説明																						
M	マスター																						
I	入出力																						
S	システム																						
C	Console																						
X	情報を使用できないか、矛盾するデータが見つかりました。																						
CONSNAME	発行コンソール名を返します。																						
D2C	符号付き整数をフルワードに変換します。																						
D2X	符号付き 10 進数を 16 進表記に変換します。																						

表 4. 編集オーダー (続き)

編集オーダー	説明
DELETE	出力オーダー。2 進数入力の「Y」または「I」は、コマンドを削除することを示します。
JOBNAME	入力オーダー。コマンドの発信元の 8 文字の JES ジョブ名を指定します。
LEFT	入力を指定された長さに切り捨てるか、埋め込みます。文字は入力の先頭から、すなわち左からカウントされます。
NEXT	ブランクを入れずに入力を出力に配置するように指定します。
NEXTWORD	ブランクを入れて入力を出力に配置するように指定します。
NVABLE	NETVONLY アクションを正常に実行できる場合は「Yes」を、そうではない場合は「No」を返します。
ONTO	すべての入力オーダーについてコマンド・テキストの論理終了位置を設定します。
PAD	後続のオーダーで使用される埋め込み文字を指定します。埋め込み文字を使用するオーダーの例には、LEFT 変換オーダーおよび position 出力オーダーがあります。
PARSE	WORD 入力命令がワードをカウントする方法を指定します。
PREFIX	変換オーダー。入力テキストの先頭にリテラル・ストリングを追加します。
RESET	以前のすべての SKIPTO および UPTO オーダーを取り消します。元の入力行は、RESETのあとに指定された入力命令では使用可能です。
RIGHT	入力を指定された長さに切り捨てるか、埋め込みます。文字は入力の最後から、すなわち右からカウントされます。
RVAR	入力改訂変数名から、現行値またはヌル・ストリングを返します。
SKIPTO	入力オーダー position length および WORD に対して、行の論理開始位置が行の最初の文字以外のポイントになるように設定します。
STRIP	データの先頭または最後の埋め込み文字を除去することを指定します。
STRIPL	データの先頭の埋め込み文字を除去することを指定します。
STRIPR	データの最後の埋め込み文字をすべて除去することを指定します。
SUBSTR	入力データのサブセットが選択されることを指定します。
SYSNAME	コマンドの発信元システムの 8 文字の名前を指定します。
UPTO	入力行の論理終了位置を再定義します。
WORD	入力行のサブセットを処理することを指定します。サブセットは、開始ワードおよびワードの総数を指定することによって定義されます。
X2C	16 進 EBCDIC ストリングを 2 進表記に変換します。
YESNO	1 バイトのフィールドを文字ストリングの Yes または No に変換します。

PIPE EDIT オーダーについての詳細は、NetView のオンライン・ヘルプまたは「IBM Tivoli NetView for z/OS プログラミング:パイプ」を参照してください。

コマンド改訂テーブルの例

以下に、CRT の例 (CNMSCRT1 サンプル) を示します。

```
UPON(CMDVERB = 'V' | CMDVERB = 'VARY')
  SELECT
    WHEN(WORD 2 = 'NETVIEW')
      NETVONLY=CNMSRVMC
      ! V NetView,(anytext)
      ! handle in NetView
```

```

        OTHERWISE                                ! all other VARY cmds untouched
    END

UPON (CMDVERB='T' | CMDVERB='SET')
    SELECT
        WHEN (W2 next W3 = 'MPF NO')                ! SET MPF=NO ?
            WTO("What do you think you're doing, Dave?") ! Think about it
        OTHERWISE
    END
UPON (CMDCONS='ROOT')                            ! For special console...
* Note this "empty" UPON is tested first by cmd revision. For this
* special console is thus exempt from all CMDVERB & OTHERCMD actions
UPON (ALLCMD)                                     ! THIS applies even to console ROOT
    SELECT
        WHEN (CMDVERB ^= 'SEND')
    * The above means that the following is only for the SEND cmd.
        WHEN (SKIPTO /USER=/ 1 FOUND ^= 'Yes')        ! default = "ALL"
            WTO("TLH447E Please do not broadcast to all.") ! explain to op
            REVISE('Y' DELETE)                        ! disallow default
        OTHERWISE
    END
    SELECT      ! Note: second, independent SELECT under the ALLCMD
        WHEN (CMDVERB ^= 'IMSTDIS')                    ! special to IMS cmd
        WHEN (W2 = 'CCTL')
            WTO("TLH851I Please review guidelines for SYSTAR1.")
            !! Adding a message to the command response
        OTHERWISE
    END
    * NOTE: The following depends on NetView having set a CHRON command
    * to issue SETRVAR (or clist containing SETRVAR) to establish a
    * value for SHIFT: NORMAL (working hours), NIGHT (other times of
    * day, or HOLIDAY (non work days)
UPON (CMDVERB='S' | CMDVERB='START')
    SELECT
        WHEN (W2 ^= /STATCOMP/)                ! starting special proc? see next WHEN
        WHEN ("SHIFT" RVAR = "NORMAL")        !
            NETVONLY=CNMSRVMC                    ! double use of sample clist!
        WHEN ("SHIFT" RVAR = "HOLIDAY")
            ! cmd is allowed, no action here
        WHEN (SKIPTO "LIMIT=" 1 FOUND = 'Yes')    ! limit keyword present?
            ! at NIGHT, proc is allowed with LIMIT keyword specified
        OTHERWISE                                ! SHIFT assumed to be NIGHT, no LIMIT
            WTO('TLH722E Do not start' 1 WORD 2 NW
                'without LIMIT keyword, except holidays.')
            REVISE('Y' DELETE)
    END
    * START T620EESS.SS,SUB=MSTR,PPIOPT=PPI

```

コマンド改訂テーブルの使用報告書

REVISE CMD REPORT コマンドを使用すると、アクティブな改訂テーブルについての統計および使用状況の情報を収集できます。正常に実行された場合は、CNM014I メッセージが発行されます。REPORT キーワードを MEMBER オペランドと一緒に使用した場合は、表示される情報は、置き換えられたテーブルとそのテーブルが置き換えられた時刻に関する情報となります。

REVISE CMD コマンドの使用法についての詳細は、NetView のオンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。

コマンド改訂テーブルのテスト

以下の手順に従うと、コマンド改訂テーブルをテストできます。

1. MVS コマンドを使用してコマンドを発行します。例えば、以下のようになります。

```
MVS SET MPF=NO
```

2. NetView コマンド改訂テーブルを使用して、コマンドを実行する代わりに WTO を発行します。これを行うためのコマンド改訂テーブルの一部を以下に示します。

```
UPON (CMDVERB='T' | CMDVERB='SET')
SELECT
  WHEN (W2 next W3 = 'MPF NO')                ! SET MPF=NO ?
    WTO("This command has been blocked by the NetView program.")
  OTHERWISE
END
```

第 15 章 自動化テーブル

本章では、以下のことを説明します。

- NetView 自動化テーブル
- 自動化テーブルで利用できるステートメント
- 自動化テーブルのコーディング方法
- 自動化テーブル・ステートメントの構文
- 自動化テーブルの設計上のガイドライン
- 自動化テーブルの使用報告書

自動化テーブルを使用してメッセージおよび MSU を自動化する方法については、361 ページの『第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化』を参照してください。

自動化テーブルの論理のテストについては、531 ページの『第 34 章 自動化テーブルのテスト』を参照してください。

自動化テーブルについて

自動化テーブルには、メッセージや管理サービス単位 (MSU) に自動的に応答する機能があります。このテーブルには、NetView プログラムが、特定のメッセージおよび MSU を受信したときに取るアクションを定義するステートメントが含まれています。例えば、コマンド、コマンド・リスト、またはコマンド・プロセッサのいずれの形式でも応答を出すことができます。

さらに、属性や処理オプションを設定することも可能です。例えば、メッセージを抑止したり、ログを取ったり、または経路指定したり、あるいは MSU を遮断したり、記録したり、または強調表示したりできます。

さらに、自動化テーブルでは、画面にエコーされたコマンドを、メッセージと見なしして処理することもできます。メッセージ・タイプにアスタリスク (HDRMTYPE = *) またはその他のコマンド関連のメッセージ・タイプが指定されている場合に、処理を終了させるメッセージをテーブルの最初に加えると、自動化テーブルによるコマンドの処理を停止させることができます。

自動化テーブル・ステートメントの要素

自動化テーブルには以下の要素があります。

- *IF-THEN* ステートメント。NetView プログラムを使用して自動化するメッセージおよび MSU を指定します。IF-THEN ステートメントには、一連の条件と、それに続いて、メッセージまたは MSU がこれらの条件を満たしたときに NetView プログラムが実行する一連のアクションを含みます。
- *BEGIN-END* セクション。ステートメントをひとまとめにして処理できるようにするものです。BEGIN-END セクションは、IF-THEN ステートメントの BEGIN オプションで開始され、END ステートメントで終了します。

- *ALWAYS* ステートメント。テーブル内でこのステートメントの処理の対象となるすべてのメッセージと MSU に関してアクションが取られるようにするため指定します。
- *%INCLUDE* ステートメント。別々にコーディングされ、保守されている自動化テーブルのセクションを組み込むために使用します。これにより、自動化テーブルの保守を、いくつかのグループまたは個人で分割して行えます。ユーザーは、自動化テーブル管理機能 (AUTOMAN) を使用して *INCLUDE* 構造体を表示することができます。
- *SYN* ステートメント。テーブル内で、あとで使用する同義語を定義できるようにします。各 *SYN* ステートメントには、名前と関連する値を含みます。

自動化テーブル・ステートメントは、メンバー *DSIPARM* に保管します。自動化テーブルを構成するステートメントは、1 つのメンバー内、またはメンバーのセットに保管しておき、*%INCLUDE* ステートメントを使用して主自動化テーブル・メンバーに組み込むことができます。

自動化テーブル処理

AUTOMAN コマンドまたは AUTOTBL コマンドのいずれかを使用して、1 つまたはセットになっている自動化テーブルの活動化、非活動化、テスト、リスト、または状況のチェックを行うことができます。また、上記を行えるように定義されている、自動化テーブルの個々のステートメントまたはステートメントのグループを、使用可能にしたり使用不可にすることもできます。

AUTOMAN の詳細については、287 ページの『複数の自動化テーブルの管理』を参照してください。

AUTOTBL コマンドの構文などの詳細については、NetView プログラム・オンライン・ヘルプを参照してください。273 ページの『自動化テーブル・リストの例』は、自動化テーブルのリストに AUTOTBL コマンドを使用した結果を示しています。

自動化テーブルを活動化する場合、NetView プログラムは、まず、組み込まれるすべてのメンバーを結合し、同義の名前を同義の値に置き換えることによって、すべての *%INCLUDE* および *SYN* ステートメントを解決します。メッセージおよび MSU の処理に直接影響を与えるのは、*IF-THEN* ステートメント、*BEGIN-END* セクション、および *ALWAYS* ステートメントに限られます。

自動化テーブルの検索

NetView プログラムでメッセージまたは MSU を受信したときに、自動化テーブルがアクティブであれば、NetView プログラムは、アクティブな自動化テーブルを順次に検索して、以下のものを探します。

- 受信したメッセージまたは MSU に該当する条件
- 無条件に一致する *ALWAYS* ステートメント

一致するものが見つければ、一致するステートメントで指定されたアクションが NetView プログラムにより実行されます。一致するステートメントで *CONTINUE(Y)* が指定されている場合、NetView プログラムはそれ以外に一致する

ものがないかどうか検索を続行します。一致するステートメントで CONTINUE(Y) が指定されていない場合、NetView プログラムは自動化テーブルにおけるメッセージまたは MSU の検索を終了します。

自動化テーブル・ステートメントのタイプ

自動化テーブル・ステートメントのすべてが、入力されるデータすべてに一致するというわけではありません。メッセージが処理されるときに NetView プログラムがチェックするのは、メッセージに適用される自動化ステートメントのみです。MSU が処理されるときに NetView プログラムがチェックするのは、MSU に適用される自動化ステートメントのみです。

IF-THEN または ALWAYS ステートメントのタイプは、『メッセージ』、『MSU』、またはその『両方』の 3 つのうちいずれかです。

- 『メッセージ』タイプのステートメントはメッセージにのみ適用されます。
- 『MSU』タイプのステートメントは、MSU にのみ適用されます。
- 『両方』タイプのステートメントは、メッセージにも MSU にも適用されます。

IF-THEN ステートメントは、そのステートメントに指定されている条件項目とアクションのタイプに従って決まります。ALWAYS ステートメントは、ステートメントに指定されているアクションのタイプに従って決まります。

条件項目またはアクションのタイプは、『メッセージ』、『MSU』、またはその『両方』の 3 つのうちいずれかです。条件項目またはアクションのタイプを判別するには、本章で所定の項目またはアクションの説明を参照してください。175 ページの『条件項目』では、条件項目について説明しており、240 ページの『処置』では、アクションについて説明しています。

ステートメントのタイプの判別

IF-THEN または ALWAYS ステートメントのタイプを判別する方法は、以下のとおりです。

- すべての条件項目およびアクションのタイプが『メッセージ』であれば、ステートメント・タイプは『メッセージ』になります。
- すべての条件項目およびアクションのタイプが『MSU』であれば、ステートメント・タイプは『MSU』になります。
- すべての条件項目およびアクションのタイプが『両方』であれば、ステートメント・タイプは『両方』になります。
- 一部の条件項目およびアクションのタイプが『両方』であり、一部が『メッセージ』であれば、ステートメント・タイプは『メッセージ』になります。
- 一部の条件項目およびアクションのタイプが『両方』であり、一部が『MSU』であれば、ステートメント・タイプは『MSU』になります。
- 一部の条件項目およびアクションのタイプが『メッセージ』であり、一部が『MSU』であるステートメントは、無効です。
- ALWAYS のように、条件項目もアクションももたないステートメントのタイプは、『両方』です。
- ステートメントに有効でない部分があれば、そのステートメントは無効です。

ステートメントのタイプおよび処理

ステートメント・タイプは、BEGIN-END セクションの処理にも影響を与えます。BEGIN-END セクションのタイプは、BEGIN キーワードが組み込まれており、セクションを開始するステートメントと同じになります。

- ステートメント・タイプ『メッセージ』で開始される BEGIN-END セクションのタイプは『メッセージ』であり、ここにはタイプが『メッセージ』または『両方』であるステートメントまたは別の BEGIN-END セクションを組み込むことができます。
- ステートメント・タイプ『MSU』で開始される BEGIN-END セクションのタイプは『MSU』であり、ここにはタイプが『MSU』または『両方』であるステートメントまたは別の BEGIN-END セクションを組み込むことができます。
- ステートメント・タイプ『両方』で開始される BEGIN-END セクションのタイプは『両方』であり、ここにはタイプが『メッセージ』、『MSU』、または『両方』であるステートメント、あるいは別の BEGIN-END セクションを組み込むことができます。
- 『MSU』タイプのステートメントが組み込まれた『メッセージ』タイプの BEGIN-END セクションや、『メッセージ』タイプのステートメントが組み込まれた『MSU』タイプの BEGIN-END セクションは、無効です。

無効なステートメントや BEGIN-END セクションが組み込まれた自動化テーブルを活動化することはできません。

自動化テーブルでメッセージを受信した場合、NetView プログラムで処理されるステートメントおよび BEGIN-END セクションのタイプは『メッセージ』または『両方』に限られます。自動化テーブルで MSU を受信した場合、NetView プログラムで処理されるステートメントおよび BEGIN-END セクションのタイプは『MSU』または『両方』に限られます。

自動化テーブルのコーディング

自動化テーブルのコーディングには、次の指示および制限が適用されます。

- 自動化テーブルは、固定長 80 文字形式のメンバーに保管しなければなりません。1 から 72 桁目までにステートメントをコーディングできます
- 73 から 80 桁目までには、シーケンス番号を指定します。

シーケンス番号はオプションですが、使用するときには次の条件を満たす必要があります。

- 73 桁目から始まらなければならない。
- 英数字で構成する必要がある。ただし、@、\$、および # の文字は含めることができます。
- %INCLUDE ステートメント以外の各ステートメントの終わりには、セミコロン (;) を指定しなければなりません。
- 自動化テーブルは、大文字小文字混合でコーディングすることができます。大文字は、以下のものに予約されています。
 - コメント
 - 文字リテラル (引用符付きストリング)

- 同義語の名前
- 同義語の値
- %INCLUDE ステートメントに指定するメンバー名

ステートメントのその他のコンポーネントは、テーブルの処理中に内部で大文字に変更されます。この結果、ステートメントが大文字になっていることを示すエラー・メッセージが出されることがあります。

- 行を字下げする場合や、キーワード、論理演算子、および括弧を区切る場合には、ブランクを使用できます。

ただし、比較ストリング内でブランクを使用すると、そのブランクはそのストリング内の文字であると見なされます。

- ステートメントは、1 桁から 72 桁目までの間で、必要なだけ続けることができます。1 行の 72 桁目までステートメントをコーディングし、同じステートメントを次の行の 1 桁目に続けることができます。そうすると、行が連結されます。これにより、長いシンボルを複数行にまたがって分けることができます。

行は、論理演算子、括弧、完全に指定された条件、あるいはオペランドのあとで区切り、次の行の 1 桁から 72 桁目までの間の任意の箇所から再開できます。

- 比較テキストおよび同義語の値の場合の区切り文字には、単一引用符を使用しなければなりません。
 - 同義語の値や比較テキストに単一引用符 (') が組み込まれている場合には、その単一引用符を 2 つ続けて (") 表記しなければなりません。
 - 2 つの単一引用符の代わりに、二重引用符で置き換えることはしないでください。
- コメントは自動化テーブル・メンバーの別の行に置いてください。
 - 継続する自動化テーブル・ステートメントの始めから終わりまでの間には、コメント行を入れないでください。
 - 各コメント行の 1 桁目にはアスタリスク (*) を指定しなければなりません。
- DSIPARM データ・セットの自動化テーブル・メンバーから読み取られた自動化テーブル・ステートメントでは、システム・シンボルが置換されます。

NetView プログラムと共に提供される &DOMAIN 記号も置換プロセスに含まれます。この処理は、コメントの除去、置換、レコード処理の順に行われます。コメントは、置換後にも除去されます。&DOMAIN シンボルでは常に置換が行われます (NetView プログラムの開始時に置換が禁止されない場合)。

MVS およびユーザー定義のシステム・シンボルについては、MVS 第 5 版リリース 2 以前の MVS システムを実行している場合は、置換は行われません。

- 自動化テーブルでは、日本語の 2 バイト文字はサポートされません。

BEGIN-END セクション

BEGIN-END セクションには、一連の自動化テーブル・ステートメントを指定します。これらのステートメントは、IF-THEN ステートメントまたは ALWAYS ステートメントの BEGIN オプションを使って開始され、END ステートメントで終了します。BEGIN-END セクションを使用すると、自動化テーブルを論理的にセグメント化したり、自動化テーブル処理のパフォーマンスを向上させることができます。

BEGIN-END セクションの構文は、以下のとおりです。

BEGIN-END セクション

```
▶▶ IF— conditions — THEN — BEGIN—; ————— END—; —————▶▶  
    ALWAYS— ————— statements—;
```

項目の意味:

IF 169 ページの『IF-THEN ステートメント』で説明するように、IF-THEN ステートメントを開始します。

conditions

以前に説明したように、THEN で示されているアクションを処理するかどうかを判別する条件です。

THEN 以前に説明したように、IF-THEN ステートメントの THEN 部分を開始します。

ALWAYS

263 ページの『ALWAYS ステートメント』で説明するように、ALWAYS ステートメントを開始します。ALWAYS ステートメントで BEGIN-END セクションを開始するのは、BEGIN-END セクションを使わずに単に *statements* をコーディングする場合と同じ意味をもちます。

BEGIN

一連のステートメントの始まりを示します。BEGIN ステートメントを END ステートメントと同じ行に指定することはできません。

statements

任意のステートメント群を指定します。これには、SYN、%INCLUDE、IF-THEN、および ALWAYS の各ステートメントや、その他の BEGIN-END セクションを指定することができます。

END 一連のステートメントの終わりを示します。END ステートメントを BEGIN ステートメントと同じ行に指定することはできません。

使用上の注意:

1. BEGIN とアクションを同時に、同じ IF-THEN ステートメントに指定することはできません。
2. 各 BEGIN ステートメントには、対になる END ステートメントを指定しなければなりません。
3. 条件が真であると、自動化テーブルは、そのセクション内の最初のステートメント (BEGIN のあとのステートメント) から処理を続けます。

条件が真でない場合、自動化テーブルは、そのセクションを終了させる END ステートメントのあとに続くステートメントから処理を続けます。

4. BEGIN-END セクションは、ネストすることができます。つまり、1 つの BEGIN-END セクションの中に別の BEGIN-END セクションを入れることができます。
5. BEGIN-END セクションの中で使用されるステートメントのタイプは互いに整合性がとれていなければならず、IF-THEN ステートメントに対しては、ステートメントの IF 部に指定された条件と整合性がとれている必要があります。

MSU タイプのステートメントとメッセージ・タイプのステートメントを混在させることはできません。ただし、両タイプのステートメントは、MSU タイプのステートメントかメッセージ・タイプのステートメントのいずれかと混在していても構いません。

詳しくは、「165 ページの『自動化テーブル・ステートメントのタイプ』」を参照してください。

6. (BEGIN-END セクションを開始する IF-THEN ステートメントの *conditions* 部) の変数のセットにアクセスして、その BEGIN-END セクション内のすべての EXEC アクションで 사용할 ことができます。

それと同じ変数名を、(このセクション内にある低レベルの IF-THEN ステートメントの) 条件部の値に割り当てた場合、その変数名の値は一時的にオーバーライドされます。その低レベルの IF-THEN ステートメント (もしくはその BEGIN-END セクション) の処理が終わると、変数は、高レベルの IF-THEN ステートメントで定義されている値に戻ります。

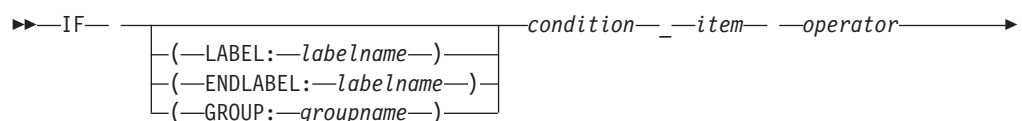
IF-THEN ステートメント

IF-THEN ステートメントを使用すると、NetView 自動操作で代行受信および処理させたいメッセージや MSU を指定できます。このステートメントは、自動操作のために選択されるメッセージまたは MSU が満たさなければならない条件や、メッセージまたは MSU がこれらの条件を満たしている場合に NetView プログラムで行うアクションをコーディングするために使用できます。

NetView プログラムは、IF ステートメントの演算子の前後に示された式を評価します。この条件が真であれば、NetView プログラムはそのステートメントの THEN 部分を処理します。論理 AND (&) 演算子、論理 OR (|) 演算子、および括弧を使用すると、複数の条件を組み合わせられます。

IF-THEN ステートメントの構文は、次のとおりです。

IF-THEN ステートメント



► —compare—_—item— —THEN— —actions—
 BEGIN—

項目の意味:

IF 各 IF-THEN ステートメントの始めにコーディングするキーワードです。

LABEL:*labelname*

LABEL キーワードは、AUTOTBL コマンドの DISABLE または ENABLE 機能を用いて指定される、自動化テーブル・ステートメントまたは BEGIN-END セクションを識別します。

- *labelname* は、英数字で指定する必要があります。@、#、および \$ は含めることができます。
- *labelname* の長さは最大 16 文字です。

ENDLABEL:*labelname*

ENDLABEL キーワードは、AUTOTBL コマンドの DISABLE または ENABLE 機能を用いて指定される、自動化テーブル・ステートメントまたは BEGIN-END セクションを識別します。

注:

- *labelname* の値は、同じメンバーの中にある直前の LABEL キーワードにある値に、一致させる必要があります。
- ENDLABEL が BEGIN-END セクション内にある場合、関連する LABEL は、同じ BEGIN-END セクション内になければなりません。
- LABEL-ENDLABEL のペア上で使用される名前は、自動化テーブルの中では固有でなければなりません。
- ENDLABEL は、英数字で指定する必要があります。@、#、および \$ は含めることができます。
- *labelname* の長さは最大 16 文字です。

labelname の値は、同じメンバーの中にある直前の LABEL キーワードにある値に、一致させる必要があります。

GROUP:*groupname*

GROUP キーワードは、AUTOTBL コマンドの DISABLE または ENABLE 機能を用いて指定される、自動化テーブル・ステートメントまたは BEGIN-END セクションを識別します。

注:

- 1 つ以上の自動化テーブル・ステートメントは、AUTOTBL コマンドの DISABLE または ENABLE 機能を用いて指定される、名前付きグループのステートメントの一部を構成することができます。
- GROUP 名によって識別されるステートメントは、必要であれば複数のメンバーの中に入れることができます。
- *groupname* は、英数字で指定する必要があります。@、#、および \$ は含めることができます。
- *groupname* の長さは最大 16 文字です。

condition_item

比較される項目は、ビット・ストリング、文字ストリング、または解析テンプレートを指定することができます。

条件項目の詳細については、175 ページの『条件項目』を参照してください。

operator

条件項目を比較項目と比較する方法を示します。

- = 条件項目が比較項目と同じである場合、条件が真であることを指定します。
- ≠ 条件項目が比較項目と同じでない場合、条件が真であることを指定します。
- < 条件項目が比較項目より小である場合、条件が真であることを指定します。

注:

- 変数とプレースホルダーはサポートされません。
- 比較値は長さが違っていても構いません。
- ヌル・ストリングは他のどのストリングより小であると見なされます。

- <= 条件項目が比較項目より小か等しい場合、条件が真であることを指定します。

注:

- 演算子に => を指定することができます。
- 変数とプレースホルダーはサポートされません。
- 比較値は長さが違っていても構いません。
- ヌル・ストリングは他のどのストリングより小であると見なされます。

- > 条件項目が比較項目より大である場合、条件が真であることを指定します。

注:

- 変数とプレースホルダーはサポートされません。
- 比較値は長さが違っていても構いません。
- ヌル・ストリングは他のどのストリングより小であると見なされます。

- >= 条件項目が比較項目より大か等しい場合、条件が真であることを指定します。

注:

- 演算子に <= を指定することができます。
- 変数とプレースホルダーはサポートされません。
- 比較値は長さが違っていても構いません。

- スル・ストリングは他のどのストリングより小であると見なされます。

compare_item

NetView プログラムが比較項目と比較する対象には、ビット・ストリング、文字ストリング、または解析テンプレートを指定することができます。

詳しくは、「234 ページの『ビット・ストリングの比較項目』」を参照してください。

THEN IF-THEN ステートメントの 2 番目の部分にコーディングするキーワードを示します。

actions IF-THEN ステートメントの IF 条件が真である場合に、NetView プログラムが行うアクションを指定します。

詳しくは、「240 ページの『処置』」を参照してください。

BEGIN

BEGIN-END セクションの開始を示します。

詳しくは、168 ページの『BEGIN-END セクション』を参照してください。

IF-THEN 構文使用上の注意:

- 1 つのステートメントに、複数の条件を指定することができます。論理 AND (&) または論理 OR (|) 演算子で条件を結合してください。

- 論理 AND (&) の前後にはブランク・スペースを入れてください。

論理 AND (&) で他のデータと連結する場合は、SYSCONE サポートによって IF-THEN ステートメントの論理が変更される場合があります。

- 論理 AND 演算子で条件を結合した場合、すべての結合条件が真でなければ、指定されたアクションが行われません。
- 論理 OR 演算子で式を結合した場合、少なくとも 1 つの結合条件が真でなければ、指定されたアクションが行われません。

図 24 の IF-THEN ステートメントは、論理 AND 演算子で結合された 2 つの条件を示しています。この条件が真になるには、メッセージがドメイン CNM01 から出され、しかもそのテキストが PURGE DATE IS LATER THAN TODAY'S DATE となっていなければなりません。

```
IF DOMAINID='CNM01' &
  TEXT='PURGE DATE IS LATER THAN TODAY'S DATE' THEN
  EXEC (CMD('CLISTA') ROUTE (ONE * OPER1));
```

図 24. 論理 AND 演算子の使用例

論理 AND 演算子で結合された 2 つの条件については、図 25 にも例を示しています。この例では、ドメイン ID は CNM02、MSU 主ベクトル・キーは X'0000' (アラートを示します) でなければなりません。

```
IF DOMAINID='CNM02' & MSUSEG(0000) ^= '' THEN
  COLOR(YEL)
  CONTINUE(Y);
```

図 25. 追加の論理 AND 演算子の使用例

図 26 の IF-THEN ステートメントは、論理 OR 演算子で結合された 2 つの条件を示しています。メッセージ ID が IST051A または IST1311A であれば、NetView プログラムは、指定されたアクションを行います。

```
IF MSGID='IST051A' | MSGID='IST1311A'  
  THEN EXEC (CMD('CLISTA') ROUTE (ONE * OPER1));
```

図 26. 論理 OR 演算子の使用例

2. NetView プログラムでは、論理 AND 演算子で結合された式をグループ化し、次に論理 OR 演算子で結合された式をグループ化します。

例えば、図 27 に示す IF-THEN ステートメントは、論理 OR 演算子と論理 AND 演算子で 3 つの条件を指定しています。

```
IF DOMAINID='CNM01' |  
  TEXT='PURGE DATE IS LATER THAN TODAY''S DATE' &  
  SYSID='MVS1' THEN  
  EXEC (CMD('CLISTA') ROUTE (ONE * OPER1));
```

図 27. 論理 OR および論理 AND 演算子の使用例

NetView プログラムでは、TEXT 条件と SYSID 条件が (論理 AND 演算子でこれら 2 つの条件を結合しているため) 一緒に評価されます。TEXT 条件と SYSID 条件の両方、あるいは DOMAINID 条件が真でなければなりません。次に、このプログラムは、その結果を DOMAINID 条件と結合します。

3. NetView プログラムによる評価を一括して行いたい比較項目の回りを括弧で囲むと、NetView プログラムが条件をグループ化する順序を制御できます。

次の例は、論理演算子のグループ化を示します。DOMAINID 条件と TEXT 条件を、NetView プログラムで一括して評価させたい場合、図 28 に示すように、これらの条件をまとめて括弧で囲みます。

```
IF (DOMAINID='CNM01' |  
  TEXT='PURGE DATE IS LATER THAN TODAY''S DATE') &  
  SYSID='MVS1' THEN  
  EXEC (CMD('CLISTA') ROUTE (ONE * OPER1));
```

図 28. 論理演算子のグループ化の例

前の例の IF-THEN ステートメントを処理するとき、NetView プログラムは、DOMAINID 条件と TEXT 条件を (括弧でグループ化されているため) 一度に評価します。次に、この結果は、NetView プログラムによって SYSID 条件と結合されます。DOMAINID か TEXT 条件のいずれかが真でなければなりません。さらに、SYSID 条件も真でなければなりません。

MLWTO (複数行のオペレーター宛て) メッセージの先頭にブランク行がある場合、NetView プログラムは、ブランク行を無視します。ブランク行は、表示用に保持されており、コマンド・プロシーチャーで GETMLINE を使用しているときには行の配置に影響を与えるおそれがあります。

MVS に対して表示される MLWTO メッセージには、制御行 (IEE932I)、順次のメッセージ ID、またはその両方が組み込まれている場合があります。メッセー

ジをさらに有効に使用するために、IEE932I は NetView プログラムによって除去されますが、順次のメッセージ ID が除去されることはありません。

4. 次の例の中の一連の IF-THEN ステートメントは、VTAM という名前のブロックを構成する、自動化テーブル・ステートメントを示しています。

以下を指定した AUTOTBL ENABLE または DISABLE コマンドを使用すると、さまざまなステートメントを使用可能または使用不可にすることができます。

- **LABEL=VTAM** は、図 29 の中の最初のステートメントのみを指定する場合
- **ENDLABEL=VTAM** は、図 29 の中の最後のステートメントのみを指定する場合
- **BLOCK=VTAM** は、図 29 の中のブロック全体 (3 つのすべてのステートメント) を指定する場合

以下の一連の IF-THEN ステートメントは、LABEL および ENDLABEL キーワードの使用例です。

```
IF (LABEL:VTAM) MSGID = 'IST051A'  
  THEN EXEC (CMD('CLISTA') ROUTE (ONE * OPER1));  
IF MSGID = 'IST052A'  
  THEN EXEC (CMD('CLISTB') ROUTE (ONE * OPER1));  
IF (ENDLABEL:VTAM) MSGID = 'IST053A'  
  THEN EXEC (CMD('CLISTC') ROUTE (ONE * OPER1));
```

図 29. LABEL および ENDLABEL キーワードの使用例

図 30 は、GROUP キーワードの使用法を示すもので、複数の自動化テーブル・ステートメントを組み込んでいます。このなかには、VTAMX という名前のステートメントのグループに含まれているものもあります。AUTOTBL コマンドを使用し、GROUP=VTAMX のキーワードを指定することによって、このグループの一部となっている自動化テーブルの中のすべてのステートメントを、使用可能にしたり、または使用不可にしたりすることができます。この例では、最初と 3 番目のステートメントが該当します。

```
IF (GROUP:VTAMX) MSGID = 'IST054A'  
  THEN EXEC (CMD('CLISTX') ROUTE (ONE * OPER1));  
IF MSGID = 'IST055A'  
  THEN EXEC (CMD('CLISTY') ROUTE (ONE * OPER1));  
IF (GROUP:VTAMX) MSGID = 'IST056A'  
  THEN EXEC (CMD('CLISTZ') ROUTE (ONE * OPER1));
```

図 30. GROUP キーワードの使用例

5. 無許可プログラムから出される MVS メッセージには、正符号が加えられています。

NetView プログラムは、この正符号を除去し、自動化内部機能 (AIFR) 内のフィールドを設定して、メッセージが無許可プログラムにより出されたことを示します。

6. アクションも BEGIN-END セクションも指定せずに THEN キーワードを使用すると、自動化テーブルが、そのメッセージまたは MSU に関してそれ以上のアクションを行わないように指示できます。175 ページの図 31 は、アクションも

BEGIN-END セクションも指定しない THEN キーワードの使用法を示しています。

```
IF HDRMTYPE = '*' THEN ;
```

図 31. アクションを指定しない THEN キーワードの使用例

条件項目

本セクションでは、IF-THEN ステートメントで使用できる条件項目について説明します。以下の 3 つの表に、タイプごとの条件項目を示します。

- メッセージについては、表 5 を参照。
- MSU については、177 ページの表 6 を参照。
- メッセージおよび MSU については、178 ページの表 7 を参照。

以下の複数の表で、各条件項目をアルファベット順に説明します。ただし、以下の 2 点の例外があります。

- メッセージの表示アクションは、そのメッセージが表示のためコマンド機能に送られる場合を除いて無視されます。
- MSU の表示アクションは、表示のためハードウェア・モニターに送られるアラートがその MSU に組み込まれている場合を除いて無視されます。

MSU のタイプは、次の 5 つです。

- 制御ポイント管理サービス単位 (CP-MSU)
- 複数ドメイン・サポート・メッセージ単位 (MDS-MU)
- ネットワーク管理ベクトル・トランスポート (NMVT)
- 保守用統計レコード (RECMS)
- 保守用統計様式化レコード (RECFMS)

表 5. メッセージに対する IF 条件項目

条件項目	比較項目	最大長	説明
ACTIONDL	解析テンプレート	7 文字	メッセージが削除された理由
ACTIONMG	ビット・ストリング	1 ビット	アクション・メッセージの表示
AREAID	解析テンプレート	1 文字	MVS メッセージ域 ID
AUTOTOKE ¹	解析テンプレート	8 文字	MVS メッセージ処理機能自動化トークン
CART ¹	解析テンプレート	8 バイト	コマンドおよび応答トークン
CORRELATED	ビット・ストリング	1 ビット	メッセージが関連処理済みであることを示す。
CORRFAIL	ビット・ストリング	1 ビット	関連処理が失敗したことを示す。
DESC	ビット・ストリング	16 ビット	MVS メッセージ記述子コード
IFRAUWF1	ビット・ストリング	32 ビット	MVS WTO 情報
JOBNAME	解析テンプレート	8 文字	MVS 発信元ジョブ
JOBNUM	解析テンプレート	8 文字	MVS に割り当てられたジョブ番号

表 5. メッセージに対する IF 条件項目 (続き)

条件項目	比較項目	最大長	説明
KEY	解析テンプレート	8 文字	メッセージに関連付けられたキー
MCSFLAG	ビット・ストリング	16 ビット	MVS 複数コンソール・サポート・フラグ
MSGAUTH	ビット・ストリング	2 ビット	許可プログラム・インディケーター
MSGCATTR	ビット・ストリング	16 ビット	MVS メッセージ属性フラグ
MSGCMISC ¹	ビット・ストリング	8 ビット	MVS その他の経路指定フラグ
MSGCMLVL	ビット・ストリング	16 ビット	MVS メッセージ・レベル・フラグ
MSGCMSGT ¹	ビット・ストリング	16 ビット	MVS メッセージ・タイプ・フラグ
MSGCOJBN	解析テンプレート	8 文字	起点ジョブ名
MSGCPROD ¹	解析テンプレート	16 文字	MVS プロダクト・レベル
MSGCSPLX	解析テンプレート	8 文字	メッセージを発信するシスプレックスの名前
MSGDOMFL ¹	ビット・ストリング	8 ビット	MVS オペレーター・メッセージの削除 (DOM) フラグ
MSGGBGPA ¹	解析テンプレート	4 バイト	バックグラウンド表示属性
MSGGDATE	解析テンプレート	7 文字	メッセージに関連付けられたデータ
MSGGFGPA ¹	解析テンプレート	4 バイト	フォアグラウンド表示属性
MSGGMFLG ¹	ビット・ストリング	16 ビット	MVS 汎用メッセージ・フラグ
MSGGMID	解析テンプレート	4 文字	MVS メッセージ ID
MSGGTIME	解析テンプレート	11 文字	メッセージが出された時間
MSGID	解析テンプレート	255 文字	メッセージ ID
MSGSRCNM ¹	解析テンプレート	17 文字	ソース・オブジェクトのソース名
MVSRTAIN	ビット・ストリング	3 ビット	MVS 自動化メッセージ保存機能 (AMRF) AMRF 保存フラグ
NVDELID	解析テンプレート	24 文字	NetView プログラム削除 ID
ROUTCDE	ビット・ストリング	128 ビット	MVS 経路コード
SESSID	解析テンプレート	8 文字	端末アクセス機能セッション ID
SYSCONID	解析テンプレート	8 文字	システム・コンソールの名前
SYSID	解析テンプレート	8 文字	発信元 MVS システムの ID
TEXT	解析テンプレート	255 文字	メッセージ・テキスト
TOKEN	解析テンプレート	255 文字	メッセージ・テキストにおいて、ブランクで区切られるストリング
注: ¹	処理されるメッセージが元はメッセージ・データ・ブロック (MDB) であった場合にのみ、この条件項目に値が含まれます。これが発生するのは、MVS メッセージが NetView オペレーター宛てである場合です。		

表 6. MSU に対する IF 条件項目

条件項目	比較項目	最大長	説明
CORRELATED	ビット・ストリング	1 ビット	MSU が相関処理済みであることを示す。
CORRFAIL	ビット・ストリング	1 ビット	相関処理が失敗したことを示す。
HIER	解析テンプレート	HIER を参照してください。	MSU と関連付けられたリソース階層
HMASPRID ²	解析テンプレート	9 文字	アラート送信元のプロダクト ID が戻される。
HMBLKACT ²	解析テンプレート	5 文字	MSU のブロック ID とアクション・コードが戻される。
HMCPLINK ²	ビット・ストリング	1 ビット	複合リンクが介在しているかどうかを指定するインディケータが戻される。
HMEPNAU ²	解析テンプレート	16 文字	SNA-MDS/LU 6.2 アラート転送プロトコルまたは NV-UNIQ/LUC アラート転送プロトコルを使用して転送された MSU の発信元 MSU であるエントリー・ポイント・ノードのネットワーク・アドレス可能単位 (NAU) 名が戻される。
HMEPNET ²	解析テンプレート	16 文字	SNA-MDS/LU 6.2 アラート転送プロトコルを使用して転送された MSU の発信元 MSU のエントリー・ポイント・ノードの NETID 名が戻される。
HMEPNETV ²	ビット・ストリング	1 ビット	リモート・ノード NetView プログラムである発信元 MSU のエントリー・ポイント・ノードの場所を示すインディケータが戻される。この機能は、SNA-MDS/LU 6.2 アラート転送プロトコルを使用して転送された MSU にのみ適用されます。
HMEVTYPE ²	解析テンプレート	4 文字	MSU のイベント・タイプが戻される。
HMFWDDED ²	ビット・ストリング	1 ビット	ある MSU が NV-UNIQ/LUC アラート転送プロトコルによって、別のノードから転送されたかどうかを指定するインディケータが戻される。
HMFWDSSNA ²	ビット・ストリング	1 ビット	SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント・ノードから MSU が指定されたかどうかを示すインディケータが戻される。
HMGENCAU ²	解析テンプレート	1 文字	MSU の汎用原因コードが 16 進数で戻される。
HMONMSU ²	ビット・ストリング	1 ビット	MSU がハードウェア・モニターによって自動操作に実行依頼されたかどうかを指定するインディケータが戻される。
HMORIGIN ²	解析テンプレート	8 文字	MSU を送信するリソースの名前が戻される。
HMSECREC ²	ビット・ストリング	1 ビット	ハードウェア・モニターが 2 次記録を行うかどうかを指定するインディケータが戻される。
HMSPECAU ²	解析テンプレート	2 文字	MSU の特定のコンポーネント・コードが 16 進数で戻される。
HMUSRDAT ²	解析テンプレート	5 文字	MSU のサブベクトル 33 からユーザー・データが戻される。
MSUSEG	解析テンプレート またはビット・ストリング	MSUSEG を参照してください。	MSU データ

表 6. MSU に対する IF 条件項目 (続き)

条件項目	比較項目	最大長	説明
注: ² MSU がハードウェア・モニターによって自動操作用に実行依頼されていない場合、この条件項目はヌルの値を返します。			

表 7. メッセージおよび MSU に対する IF 条件項目

条件項目	比較項目	最大長	説明
ACQUIRE	解析テンプレート	ACQUIRE を参照してください。	編集指定によって決定される値
ATF	解析テンプレート またはビット・ストリング	ATF を参照してください。	指定された ATF プログラムで判別される値
ATF(DSICGLOB)	解析テンプレート	DSICGLOB を参照してください。	共通グローバル変数の値
ATF(DSITGLOB)	解析テンプレート	DSITGLOB を参照してください。	タスク・グローバル変数の値
ATTENDED	ビット・ストリング	1 ビット	在席タスク・インディケーター
AUTOMATED	ビット・ストリング	1 ビット	有効なアクション・インディケーター
AUTOTASK	ビット・ストリング	1 ビット	自動タスク・インディケーター
CURRDATE	解析テンプレート	8 文字	現行日付
CURRTIME	解析テンプレート	8 文字	現行時刻 (TOD)
CURSYS	解析テンプレート	8 文字	現行 MVS オペレーティング・システム名
DISTAUTO	ビット・ストリング	1 ビット	分散自動タスク・インディケーター
DOMAIN	解析テンプレート	5 文字	現行 NetView プログラムのドメイン名
DOMAINID	解析テンプレート	8 文字	発信元 NetView プログラムのドメイン
HDRMTYPE	解析テンプレート	1 文字	メッセージ・タイプ
IFRAUIND	ビット・ストリング	16 ビット	AIFR インディケーター・フラグ
IFRAUIN3	ビット・ストリング	8 ビット	インディケーター・ビット・フィールド
IFRAUI3X	ビット・ストリング	32 ビット	インディケーター・ビット
IFRAUSB2	解析テンプレート	2 文字	AIFR ユーザー・フィールド
IFRAUSC2	ビット・ストリング	128 ビット	AIFR ユーザー・フィールド
IFRAUSDR	解析テンプレート	8 文字	発信元 NetView プログラム・タスクの名前
IFRAUSRB	ビット・ストリング	16 ビット	AIFR ユーザー・フィールド
IFRAUSRC	解析テンプレート	16 文字	AIFR ユーザー・フィールド
IFRAUTA1	ビット・ストリング	48 ビット	AIFR 制御フラグ

表 7. メッセージおよび MSU に対する IF 条件項目 (続き)

条件項目	比較項目	最大長	説明
INTERVAL	ビット・ストリング	1 ビット	オカレンス間隔検出
LINEPRES	解析テンプレート	4 バイト	最初のテキスト・バッファの表示属性
LINETFLG	ビット・ストリング	16 ビット	表示指定変更フラグ (ビット 16) およびその他のフラグ
MVSLEVEL	解析テンプレート	8 文字	現行 MVS プロダクト・レベル
NETID	解析テンプレート	8 文字	VTAM ネットワーク ID
NETVIEW	解析テンプレート	4 文字	NetView プログラムの版およびリリース
NUMERIC	解析テンプレート	255 文字	変数の数値
NVCLOSE	ビット・ストリング	1 ビット	NetView プログラム CLOSE プロセス・フラグ
OPID	解析テンプレート	8 文字	オペレーターまたはタスク ID
OPSYSTEM	解析テンプレート	7 文字	オペレーティング・システム
SYSPLEX	解析テンプレート	8 文字	ローカル MVS シスプレックス名
TASK	解析テンプレート	3 文字	タスクのタイプ
THRESHOLD	ビット・ストリング	1 ビット	オカレンスしきい値検出
VALUE	解析テンプレート	255 文字	変数の値
VTAM	解析テンプレート	4 文字	VTAM レベル
VTCOMPID	解析テンプレート	14 文字	VTAM コンポーネント ID
WEEKDAYN	解析テンプレート	1 文字	曜日

以下は、条件項目のアルファベット順リストです。

ACQUIRE ('edit_specification')

PIPE EDIT ステージによって提供される構文と機能を使用して、AIFR データを抽出できるようにする条件項目。

戻されたメッセージ・バッファの最初の行だけが、比較のために使用されます。この AIFR パスは、自動操作が変更されるまで継続して使用されます。

edit_specification は、('edit_specification' という形式で) 単一引用符で囲む必要がありますが、*edit_specification* 自体の中で単一引用符を使用することはできません。

edit_specification (編集指定) に関する具体的な情報については、「*IBM Tivoli NetView for z/OS プログラミング:パイプ*」を参照してください。

ACTIONDL [(pos [len])]

NetView プログラム・アクション・メッセージを削除する理由。理由は、1 から 8 文字の長さの EBCDIC ストリングで表されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

ACTIONDL の値がヌル (") でない場合、自動化テーブルは、メッセージまたはアラートと対照させて DOM (オペレーター・メッセージ削除) を処理します。

有効な値は次のとおりです。

" ヌル。メッセージは DOM ではありません。

ASID メッセージを出したアドレス・スペースが終了したため、メッセージが削除されました。

INVALID

認識不能なビット設定の組み合わせが DOM に含まれています。

LOCAL

オペレーターによる重ね打ちまたは CONSOLE DELETE ステージによってメッセージが削除されました。

NETVIEW

NVDELID オプションを使用した NetView プログラム DOM コマンドによって、または内部的に NetView プログラムによって、メッセージが削除されました。

SMSGID

メッセージは、MVS の SMSGID による DOM によって削除されました。単一メッセージが、固有の ID によって削除されました。

TCB メッセージを出したタスクが終了したため、メッセージが削除されました。

TOKEN

メッセージは、MVS のトークンによる DOM によって削除されました。

最大長: 7 文字

タイプ:メッセージ

ACTIONDL の使用上の注意:

1. MVS は、TCB および ASID 条件を SMSGID による DOM に変換する場合があります。
2. SMSGID は、MVS DOM の中でも最も頻度の高いタイプです。
3. 関連する条件項目は ACTIONMG および NVDELID です。また、DOMACTION も参照してください。

ACTIONMG

メッセージが、NetView プログラムによって MVS アクション・メッセージとして処理されるかどうかを示します。ACTIONMG の値は以下のとおりです。

1 メッセージはアクション・メッセージです。

0 メッセージはアクション・メッセージではありません。

最大長: 1 ビット

タイプ:メッセージ

ACTIONMG の使用上の注意:

1. アクション・メッセージは、WTOR であるか、
MVSPARM.ActionDescCodes CNMSTYLE ステートメントで指定されているいずれかのメッセージに一致する記述子コードを使用してマークが付けられているメッセージです。
2. 関連する条件項目は、ACTIONDL および NVDELID です。また、DOMACTION も参照してください。

AREAID [(pos [len])]

メッセージが表示される複数コンソール・サポート (MCS) コンソール上に示される 1 文字の ID (A から Z まで) です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

AREAID に指定された値が 'B' またはブランクであるとき、その値はヌル (") であると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

最大長: 1 文字

タイプ:メッセージ

ATF ([BIT] 'cmdstring')

自動化テーブル機能 (ATF) を実行するために呼び出されるプログラムの ID です。

ユーザー固有の ATF プログラムの作成方法については、「*IBM Tivoli NetView for z/OS プログラミング: アセンブラー*」を参照してください。

条件項目は、プログラムから戻される値です。

比較項目は、ビット・ストリングまたは解析テンプレートです。

BIT 比較項目がビット・ストリングであることを示します。BIT を指定しない場合、解析テンプレートが比較項目になります。

cmdstring

プログラムを呼び出すコマンド・ストリングを指定します。

最初のブランクで区切られるまでのストリング (ブランクがない場合には、そのストリング全体) のテキストが、プログラム名になります。最初のブランクからあとのテキストは、呼び出されたプログラムにパラメーターとして渡されます。

リテラルの引用符付きストリングを使用する場合は、プログラム名を指定する必要があります。しかし、変数値は VALUE (*varname*) 構文を使用して、ATF プログラム・パラメーターとして渡すことができます。

プログラム名が指定されていると、それらのパラメーターはリテラルおよび VALUE 指定の任意の組み合わせで指定できます。

渡される変数は、以下の基準を満たしていなければなりません。

- ATF として渡された変数は、事前にステートメントまたは BEGIN 階層において定義しておかなければなりません。
- 設定されていない変数はすべて、ヌル・リテラルとして扱われます。
- 変数は、位置または長さの添え字付きで使用することはできません。

最大長: 256 バイト

タイプ: 両方

使用上の注意、ATF の:

1. ATF および *cmdstring* には、以下の基準が適用されます。
 - パラメーターを指定した *cmdstring* は、256 バイトから BUFHDR の長さを引いた長さまでに制限されています。
 - *cmdstring* に指定する ATF プログラム名の最大長は 8 文字までです。
 - ATF から戻される値は、256 バイトから BUFHDR の長さを引いた長さまでに制限されています。
2. インターフェースは、レジスター 1 にアドレスが保管されているパラメーター・リストに基づいています。

レジスターには、制御ワーク・ブロック (CWB) や、自動操作される AIFR を指し示すポインターが入っています。

3. レジスター 15 に送られる ATF 戻りコードは、次のとおりです。

コード 意味

0 通常

1-8 その比較が偽とされる原因となったエラーを示します。

9 以上 エラー・メッセージ CNM588E が出され、比較が偽とされる原因となったエラーを示します。

4. AUTOTBL コマンドを発行して、自動化テーブルを正しく活動化できると、そのテーブルが使用するすべての ATF プログラムが NetView プログラムによってロードされます。

メッセージまたは MSU が自動化テーブルを通るごとに、NetView プログラムが ATF プログラムを主記憶域に再ロードするわけではありません。

5. NetView プログラム・サンプルでは、ATF プログラムの例として、OPERID (CNMS4295) プログラムを提供しています。
6. ATF では長さ指定をサポートしていません。

ユーザーは、ある変数に ATF を割り当て、その後 VALUE 条件ステートメントにおいてその変数 (*pos* と *len* を含む) を使用できます。

ATF ([BIT] 'DSICGLOB varname')

DSICGLOB は、NetView プログラム提供の ATF プログラムです。コマンド・リストまたはコマンド・プロセッサが共通グローバル変数の値を事前に設定している場合には、その値が DSICGLOB から戻されます。この値は、256 文字から BUFHDR を引いた長さを超えると切り捨てられます。この変数に値が設定されていなかった場合、DSICGLOB から変数値は戻されません。

比較項目は、ビット・ストリングまたは解析テンプレートです。グローバル変数の値は EBCDIC 文字のストリングであるため、解析テンプレートを使用してください。

varname の指定方法に関しては、汎用 ATF に関する cmdstring の記述を参照してください。

ATF プログラムによってエラーが検出されると、条件項目には誤りがあると見なされ、戻りコードが組み込まれた CNM588E メッセージが出されます。

コード 意味

- 100 変数名が無効です。
- 104 指定された変数名が長過ぎます。
- 108 変数名が指定されていません。
- 112 NetView プログラムのストレージ障害。
- 116 NetView プログラムの内部エラー。
- BIT 比較項目がビット・ストリングであることを示します。BIT を指定しない場合、解析テンプレートが比較項目になります。

varname

共通グローバル変数の名前を指定します。名前の長さは、1 から 31 文字で、その名前は、有効なグローバル変数名でなければなりません。変数名に関する制約事項については、「IBM Tivoli NetView for z/OS プログラミング: REXX および NetView コマンド・リスト言語」を参照してください。

最大長: 256 文字

タイプ: 両方

注: (TVBINXIT ビットがオンになっている) オペレーターの画面の即時メッセージ作業域に送られた NetView プログラムメッセージに対応して、自動化テーブルから DSICGLOB が呼び出された場合、DSICGLOB から変数値は戻されず、その条件は偽とされます。

ATF ([BIT] 'DSITGLOB varname')

DSITGLOB は、NetView プログラム提供の ATF プログラムです。コマンド・リストまたはコマンド・プロセッサがタスク・グローバル変数の値を事前に設定している場合には、その値が DSITGLOB から戻されます。この値は、256 文字から BUFHDR を引いた長さを超えると切り捨てられます。この変数に値が設定されていなかった場合、DSITGLOB から変数値は戻されません。

比較項目は、ビット・ストリングまたは解析テンプレートです。グローバル変数の値は EBCDIC 文字のストリングであるため、解析テンプレートを使用してください。

varname の指定方法に関しては、汎用 ATF に関する *cmdstring* の記述を参照してください。

ATF プログラムによってエラーが検出されると、条件項目には誤りがあると見なされ、戻りコードが組み込まれた CNM588E メッセージが出されます。

コード 意味

- 100** 変数名が無効です。
- 104** 指定された変数名が長過ぎます。
- 108** 変数名が指定されていません。
- 112** NetView プログラムのストレージ障害。
- 116** NetView プログラムの内部エラー。
- BIT** 比較項目がビット・ストリングであることを示します。BIT を指定しない場合、解析テンプレートが比較項目になります。

varname

タスク・グローバル変数の名前を指定します。名前の長さは、1 から 31 文字で、その名前は、有効なグローバル変数名でなければなりません。変数名に関する制約事項については、「IBM Tivoli NetView for z/OS プログラミング: REXX および NetView コマンド・リスト 言語」を参照してください。

最大長: 256 文字

タイプ: 両方

使用上の注意、ATF の:

1. DSITGLOB から戻されるタスク・グローバル変数は、自動化テーブルを呼び出したタスクのグローバル変数です。自動化テーブルを呼び出し、ATF により評価されるタスクが特定できない場合は、共通グローバル変数と DSICGLOB ATF を代わりに使用してください。
2. (TVBINXIT ビットがオンになっている) オペレーターの画面の即時メッセージ作業域に送られた NetView プログラム・メッセージに対応して、自動化テーブルから DSITGLOB が呼び出された場合、DSITGLOB から変数値は戻されず、その条件は偽であると見なされます。

ATTENDED [(pos [len])]

メッセージまたは MSU を自動化する NetView プログラム・タスクを記述します。これは、1 または 0 の値を指定するビット・ストリングです。

ATTENDED の値は、次のとおりです。

- 1** タスクが以下の状態のいずれかであることを示します。
 - 表示装置を持つ OST
 - 対応する OST がある NNT

- 関連する MVS コンソールが AUTOTASK コマンドにより割り当てられている自動タスク
- 分散自動タスク

0 タスクが以下の状態のいずれかであることを示します。

- 関連する MVS コンソールが AUTOTASK コマンドにより割り当てられていない自動タスク
- タイプの異なるタスク (DST タスクや OPT タスクなど)

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: 両方

ATTENDED の使用上の注意:

1. 関連しているオペレーターが自動タスクである場合、その自動タスクがアクティブな MVS コンソールに関連付けられている場合を除いて、表示データを表示できません。
2. DISTAUTO または AUTOTASK 条件項目とともに ATTENDED を指定すると、メッセージまたは MSU を自動化するタスクの特性を追加定義できます。例えば、ATTENDED が 1、DISTAUTO が 0、AUTOTASK が 1 であれば、そのタスクは MVS コンソールと関連付けられた自動タスクです。

AUTOMATED [(*pos* [*len*])]

メッセージまたは MSU が入っている AIFR の自動化インディケータを指します。

その AIFR が、直前の有効なアクションによって自動操作になっているかどうかを指定する 1 ビットのインディケータです。AUTOMATED の値は、次のとおりです。

1 AIFR は自動化されています。

0 AIFR はまだ自動化されていません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

自動操作では、ALWAYS または CONTINUE(YES) ステートメント以外で一致が発生し、これらのデフォルトをオーバーライドするのに AUTOMATED アクションが使用されていない限り、AIFR は AUTOMATED として扱います。

最大長: 1 ビット

タイプ: 両方

AUTOTASK [(pos [len])]

メッセージまたは MSU を自動化する NetView プログラム・タスクについて記述する条件項目です。これは、タスクが自動タスクであるかどうかを指定する 1 ビットのインディケーターです。AUTOTASK の値は、次のとおりです。

- 1 タスクは自動タスクです。
- 0 タスクは自動タスクではありません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: 両方

AUTOTOKE [(pos [len])]

MVS メッセージ処理機能 (MPF) 自動化トークンの名前を 1 から 8 文字で指定します。

AUTO(YES) または AUTO(NO) が MPF テーブルに指定されている場合、値 YES および NO は自動化トークンではありません。

メッセージが元はメッセージ・データ・ブロック (MDB) であった場合にのみ、AUTOTOKE に値が指定されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ:メッセージ

CART [(pos [len])]

8 バイトの MVS コマンドおよび応答トークン (CART) を指定します。CART には、表示不能な文字が含まれている可能性があります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

フィールドに 2 進数の 0 のみが含まれている場合、CART の値はヌル (") であると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

最大長: 8 バイト

タイプ:メッセージ

CORRELATED

メッセージまたは MSU が相関処理済みかどうかを検査する条件項目です。これは 1 ビットのインディケータです。CORRELATED の値は、次のとおりです。

1 メッセージまたは MSU は相関処理済みです。

0 メッセージまたは MSU は相関処理を受けていません。

最大長: 1 ビット

タイプ: 両方

CORRFAIL

相関処理が失敗したかどうかを検査する条件項目です。これは 1 ビットのインディケータです。CORRFAIL の値は以下のとおりです。

1 内部エラーのために相関処理を実行できませんでした。

0 相関処理は問題なく実行されました。

最大長: 1 ビット

タイプ: 両方

CURRDATE [(pos [len])]

1 から 8 文字の現在日付 (yyyy/mm/dd) を指定します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

CURRTIME [(pos [len])]

1 から 8 文字の現在時刻 (hh:mm:ss) を指定します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

CURSIS [(pos [len])]

現行の MVS オペレーティング・システムの名前を 1 から 8 文字で示します。

CURSYS から戻されるシステム名と、SYSID から戻されるシステム名とが異なる場合があります。

- CURSYS は、自動化テーブルが処理しているシステムの名前です。
- SYSID は、メッセージの発信元であるシステムの名前です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

DESC [(*pos* [*len*])]

メッセージに割り当てられる、1 から 16 の MVS 記述子コードを指定します。コード値の詳細については、MVS ライブラリーを参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 16 ビット

タイプ:メッセージ

DISTAUTO [(*pos* [*len*])]

タスクが RMTCMD コマンドで開始された分散自動タスクであるかどうかを示します。DISTAUTO 条件項目は、メッセージまたは MSU を自動化するタスクの内容を記述します。DISTAUTO の値は、次のとおりです。

1 タスクは、分散自動タスクです。

0 タスクは、分散自動タスクではありません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: 両方

DOMAIN [(*pos* [*len*])]

現行の NetView プログラムドメインの名前を、1 から 5 文字で指定します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 5 文字

タイプ: 両方

DOMAINID [(*pos* [*len*])]

メッセージまたは MSU の発信元である NetView システムのドメイン名を、1 から 8 文字で示します。

メッセージの場合、DOMAINID には最初にそのメッセージを処理した NetView システムの名前が指定されます。アラートに基づいて生成されたメッセージ、BNJ030I および BNJ146I の場合、DOMAINID にはこれらのメッセージを生成した NetView システムの名前が指定されます。

ハードウェア・モニターから別の NetView プログラムに転送されたアラートの場合、DOMAINID にはそのアラートを最初に処理して転送した分散 NetView プログラムの名前が示されます。それ以外の MSU の場合、DOMAINID には自動化テーブルの検索を行っているローカル NetView プログラムの名前が示されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

HDRMTYPE [(*pos* [*len*])]

受信したメッセージまたは MSU のバッファ・タイプを 1 文字で指定します。バッファ・タイプについては、627 ページの『付録 G. NetView メッセージ・タイプ (HDRMTYPE) の説明』で説明します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 文字

タイプ: 両方

HIER [(*indexnum*)]

1 つの MSU と関連付けられた、NetView プログラムのハードウェア・モニター階層データを指定します。比較項目は、解析テンプレートです。

indexnum

特定のリソースの名前とタイプの対の索引番号 (1 から 5 まで) を指定します。

HIER は、ハードウェア・モニターから出された MSU を受信した場合にのみ設定されます。 *indexnum* を指定する場合、HIER の値には、
aaaaaaaa1111 形式 (aaaaaaaa は 8 文字の名前、 1111 は 4 文字のタイプ) の名前とタイプの一対を指定します。 名前やタイプが指定の文字数に満たない場合は、右側から必要に応じて空白で埋められます。アラートの値が *indexnum* リソースの値よりも小さい場合、その値はヌルになります。 *indexnum* を指定しないと、HIER の値は既存の名前とタイプの一対をすべて合わせたものに等しくなります。例えば、名前とタイプの一対が 3 組ある場合、その値の形式は以下のとおりです。

aaaaaaaa1111bbbbbbb2222ccccccc3333

名前とタイプの一対は、5 組まで指定できます。MSU には階層情報が組み込まれていないため、HIER の値はヌルになります。 HIER の例については、377 ページの『リソース階層の使用』を参照してください。

HIER では長さ指定をサポートしていません。ユーザーは、ある変数に HIER を割り当て、その後 VALUE 条件ステートメントにおいてその変数 (*pos* と *len* を含む) が使用できます。

最大長: 60 文字

タイプ: MSU

HMASPRID [(*pos* [*len*])]

アラート送信元のプロダクト ID を 9 文字で戻します。SRFILTER コマンドの *prodid* パラメーターにより戻されたものと同じアラート送信元のプロダクト ID です。この ID は次の項目のいずれかを指定できます。

- ハードウェア・プロダクト ID (1 から 4 文字)
- ソフトウェア・プロダクト ID (1 から 9 文字)

後書き空白は切り捨てられません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSU が以下のいずれかであれば、HMASPRID からはヌルの値が戻されます。

- 汎用レコードではない。

注: ここで、汎用とは、サブベクトル 92 を含む MSU すべてを指しています。汎用 MSU には、次のものがあります。

- サブベクトル 92 が組み込まれているアラート
- サブベクトル 92 が組み込まれているレゾリューション

- ハードウェア・モニターによって自動操作に実行依頼されていない。

最大長: 9 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: 装置の検索

```
IF HMASPRID = '3745' THEN  
    EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

この例では、ハードウェア・モニター MSU が汎用であり、3745 装置から出されている場合に、自動化テーブルによって CLISTA コマンド・リストが呼び出され、それがオペレーター AUTO1 に経路指定されるよう指定しています。

例 2: 汎用 MSU の指定

```
IF HMASPRID ^= '' THEN  
    EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

この例では、ハードウェア・モニター MSU が汎用である場合に、自動化テーブルによって CLISTA コマンド・リストが呼び出され、それがオペレーター AUTO1 に経路指定されるよう指定しています。

HMBLKACT[(*pos* [*len*])]

3 文字のブロック ID と 2 文字のアクション・コードが組み合わされた 5 文字の値を戻します。この値は、SRFILTER コマンドの *code* の値と同じになります。値は、非汎用アラート (X'0000') と、統計専用ではない RECMS と RECFMS に関してのみ戻されます。

SRFILTER コマンドに関する詳細については、NetView プログラム・オンライン・ヘルプを参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSU が以下のいずれかであれば、HMBLKACT からはヌルの値が戻されます。

- 汎用アラート (X'0000')

注: ここで、汎用とは、サブベクトル 92 を含む MSU すべてを指しています。汎用 MSU には、次のものがあります。

- サブベクトル 92 が組み込まれているアラート
- サブベクトル 92 が常に組み込まれているレゾリュション

- 解決 (X'0002')
- PD 統計 (X'0025')
- リンク構成データ (X'1332')
- 統計専用 RECMS

注: 統計専用 RECMS とは、統計データのみが組み込まれるレコード保守統計のことを指します。これらのレコードの記録モードは、X'81'、X'86'、および X'87' であり、RECFMS のバイト 8、オフセット 1 に示されます。X'87' の場合、一時エラー (永続エラーではない) を表す RECMS のみが統計専用であると見なされます。

- 統計専用 RECFMS

注: 統計専用 RECFMS とは、統計データのみが組み込まれるレコード形式の保守統計のことを指します。これらのレコードのタイプは、1、4、5 であり、RECFMS レコードのバイト 8、オフセット 1 に示されます。

- ハードウェア・モニターによって自動化テーブルに実行依頼されていない。

最大長: 5 文字

タイプ: MSU

適用の対象: ヌル以外の値が戻されるすべての MSU

例 1: ヌル以外の値を持つブロック ID およびアクション・コードのチェック

```
IF HMBLKACT ^= '' THEN COLOR(RED);
```

この例では、ヌル以外の値を持つブロック ID およびアクション・コードがチェックされ、赤色で示されるよう指定しています。

例 2: 特定のブロック ID およびアクション・コードのチェック

```
IF HMBLKACT = HEX'FFD03' THEN COLOR(RED);
```

この例では、ブロック ID が X'FFD'、アクション・コードが X'03' である MSU がチェックされ、赤色で示されるよう指定しています。

例 3: 特定のブロック ID のチェック

```
IF HMBLKACT = HEX('FFD') . &  
HMBLKACT = MYVAR THEN  
EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1));
```

この例では、ブロック ID が X'FFD' である MSU がチェックされるよう指定しています。特定のアクション・コードはチェックされません。自動化テーブルは、X'FFD' というブロック ID を持つ MSU に関して CLISTA コマンド・リストを呼び出します。そのブロック ID およびアクションは変数 MYVAR で CLISTA コマンド・リストに渡され、そのコマンド・リストはオペレーター AUTO1 に経路指定されます。

HMCPLINK[(pos [len])]

複合リンクが存在しているかどうかを 1 または 0 で示す、1 ビットのインディケーターが戻されます。

インディケーター

説明

- 1 複合リンクが存在しています。複合リンクが存在している場合、HIER 条件項目で戻されるリソース階層には示されないリソース・レベルが存在している可能性があります。複合リンクが存在している場合には、システム図式を使用して、完全な階層構造を判断してください。HIER 条件項目の詳細については、HIER を参照してください。

Most Recent Events パネルのようなハードウェア・モニター・パネルでは、パネルの最上部で図示されたリソース階層にアスタリスク (*) が付けられ、パネルの最下部付近でメッセージ行にメッセージ BNJ1538I が表示されることによって、複合リンクの存在が示されます。

- 0 複合リンクが存在していない、あるいは、ハードウェア・モニターが MSU を自動操作に実行依頼していません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: 複合リンクのチェック

```
IF HMCPLINK = '1' THEN COLOR(RED);
```

この例では、複合リンクが存在しているハードウェア・モニター MSU が赤色で示されるよう指定しています。

例 2: 複合リンクが存在していない MSU のチェック

```
IF HMONMSU = '1' &  
HMCPLINK = '0' THEN COLOR(RED);
```

この例では、ハードウェア・モニターから転送された MSU をチェックして、複合リンクの存在していないものが赤色で示されるよう指定しています。

HMEPNAU[(*pos* [*len*])]

MSU の発信元であるエントリー・ポイント・ノードのネットワーク・アドレス可能単位 (NAU) 名を戻します。ローカル MSU の場合、HMEPNAU はローカル NAU (ドメイン) 名を戻します。リモート・ノード・エントリー・ポイントから転送された MSU の場合、リモート・エントリー・ポイントの NAU 名が戻されます。これは、アラート転送メカニズム、LU 6.2 および LUC の両方に適用されます。

LU 6.2 によって転送されたアラートでは、戻される NAU 名は、ALERT-NETOP アプリケーションに最初にアラートを送信 (転送) した MS アプリケーションが常駐するエントリー・ポイント・ノードの NAU 名です。NetView プログラムが戻された NAU 名がエントリー・ポイント NAU 名であるかどうかを確実に判別できない場合 (例えば、それが中間ノード名の可能性がある) は、*nauname のように、戻された NAU 名には * (アスタリスク) が前に付けられます。

転送の詳細については、423 ページの『第 26 章 集中操作』を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 16 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例: リモート・エントリー・ポイントから転送された MSU の検索 LU 6.2 を使用して NETA.CNM01 から転送された MSU であるかどうかのチェック

```
IF HMFWDNSA = '1' &  
  HMEPNET = 'NETA' &  
  HMEPNAU = 'CNM01' THEN COLOR(RED);
```

この例では、SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント・ノード NETA.CNM01 から転送されたハードウェア・モニター MSU が赤色で示されるよう指定しています。

HMEPNET[(pos [len])]

MSU の発信元であるエントリー・ポイント・ノードの NETID 名を戻します。ローカル MSU の場合、HMEPNET はローカル NETID 名を戻します。LUC アラート転送を使用して転送された MSU の場合は、NetView プログラムが NETID 名を判別できないため、HMEPNET はアスタリスク (*) を戻します。

LU 6.2 アラート転送を使用して転送された MSU の場合、戻される NETID 名は、MS アプリケーションが常駐するエントリー・ポイント・ノードの名前です。NetView プログラムが NETID 名を判断できない場合、HMEPNET はアスタリスク (*) を戻します。NetView プログラムが NETID 名を判断できるが、NETID 名がエントリー・ポイント NETID 名であるかどうかを確信できない場合 (例えば、それが中間ノードの NETID 名である可能性がある)、HMEPNET は前にアスタリスク (*) を付けた NETID 名 (例えば、*netidnam) を戻します。

転送の詳細については、423 ページの『第 26 章 集中操作』を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 16 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例: LU 6.2 を使用して NETA.CNM01 から転送された MSU のチェック

```
IF HMFWDNSA = '1' &  
  HMEPNET = 'NETA' &  
  HMEPNAU = 'CNM01' THEN COLOR(RED);
```

HMEPNETV[(pos [len])]

MSU の発信元であるエントリー・ポイント・ノードがリモート・ノード

NetView プログラムであったかどうかを 1 または 0 で示す、1 ビットのインディケーターが戻されます。この機能は、SNA-MDS/LU 6.2 アラート転送プロトコルを使用して転送された MSU にのみ適用されます。

インディケーター

説明

- 1 エントリー・ポイントが NetView プログラムであったことを示します。
- 0 エントリー・ポイントが NetView プログラムでなかったか、または MSU が SNA-MDS/LU 6.2 アラート転送プロトコルを使用して転送されなかったことを示します。

転送の詳細については、423 ページの『第 26 章 集中操作』を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例: LU 6.2 を使用してリモート・ノード・エントリー・ポイントから転送された MSU の検索。

```
IF HMEPNETV = '1' THEN COLOR(RED);
```

この例では、SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント NetView プログラムから転送された MSU が赤色で示されるよう指定しています。

HMEVTYPE[(*pos* [*len*])]

MSU のイベント・タイプ (4 文字) を戻します。戻された値の後書きブランクは切り捨てられません。

イベント・タイプは、次のとおりです。

AVAL	BYPS	CUST	DLRC
HMV	HELD	IMPD	IMR
INST	INTV	NTFY	PAFF
PERF	PERM	PROC	REDL
RSLV	RSNT	SCUR	SNA
TEMP	UNKN	USER	

詳しくは、NetView オンライン・ヘルプ (HELP NPDA 'event_type') を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSU が以下のいずれかであれば、HMEVTYPE からヌルの値が戻されます。

- ハードウェア・モニターによって自動操作に実行依頼されていない。
- PD 統計 (X'0025')
- リンク構成データ (X'1332')
- 統計専用 RECMS

注: 統計専用 RECMS とは、統計データのみが組み込まれるレコード保守統計のことを指します。これらのレコードの記録モードは、X'81'、X'86'、および X'87' であり、RECFMS のバイト 8、オフセット 1 に示されます。X'87' の場合、一時エラー (永続エラーではない) を表す RECMS のみが統計専用であると見なされます。

- 統計専用 RECFMS

注: 統計専用 RECFMS とは、統計データのみが組み込まれるレコード形式の保守統計のことを指します。これらのレコードのタイプは、1、4、5 であり、RECFMS レコードのバイト 8、オフセット 1 に示されます。

最大長: 4 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: イベント・タイプ PERM の検索

```
IF HMEVTYPE = 'PERM' THEN COLOR(RED);
```

この例では、イベント・タイプが PERM である MSU が赤色で示されるよう指定しています。

例 2: イベント・タイプ SNA の検索

```
IF HMEVTYPE = 'SNA' . THEN COLOR(RED);
```

この例では、イベント・タイプが SNA である MSU が赤色で示されるよう指定しています。後書きブランクをチェックする必要はありません。

例 3: イベント・タイプの抽出

```
IF HMEVTYPE ^= '' &  
HMEVTYPE = MYVAR THEN  
EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1));
```

この例では、ハードウェア・モニター MSU からイベント・タイプが抽出されて、そのイベント・タイプが変数 MYVAR で CLISTA コマンド・リストに渡され、そのコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

HMFWDDED[(*pos* [*len*])]

MSU が別のノードから転送されたかどうかを 1 または 0 で指定する、1 ビットのインディケーターを戻します。

インディケータ

説明

1 MSU は、次のアラートのいずれかで別のノードから転送されました。

- NV-UNIQ/LUC アラート転送プロトコル
- SNA-MDS/LU 6.2 アラート転送プロトコル

0 MSU が別のノードから転送されていないか、LU6.2 を通して転送されているか、あるいはハードウェア・モニターが MSU の自動化を依頼していないかを示します。

次の場合にインディケータ 0 が戻されます。

- ローカル MSU が CNM インターフェースを通して受信される場合。
- ローカル MSU がオペレーティング・システムを通して受信される場合。
- MSU がプログラム間インターフェースを通して受信される場合。
- MSU が SNA-MDS/LU 6.2 アラート転送プロトコルを通して受信される場合。

注: LUC または LU 6.2 を通してエントリー・ポイントから転送される RECMS および RECFMS は、受信側のフォーカル・ポイントでは自動操作に実行依頼されません。RECMS および RECFMS は、エントリー・ポイントで自動操作に実行依頼されますが、受信側のフォーカル・ポイントでは実行依頼されません。

転送の詳細については、423 ページの『第 26 章 集中操作』を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: エントリー・ポイントから転送された MSU の検索

```
IF HMFWD = '1' THEN COLOR(RED);
```

この例では、エントリー・ポイント NetView プログラムから転送されたハードウェア・モニター MSU が、赤色で示されるよう指定しています。

例 2: エントリー・ポイントから転送されなかった MSU の検索

```
IF HMONMSU = '1' &  
HMFWD = '0' THEN COLOR(RED);
```


この例では、エントリー・ポイントの NetView プログラムからではなく、ハードウェア・モニターによって転送された MSU がチェックされ、赤色で示されるよう指定しています。

HMFWDNSNA(*pos* [*len*])

MSU が SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント・ノードから転送されたかどうかを 1 または 0 で指定する、1 ビットのインディケータを戻します。

インディケータ

説明

- 1** MSU は、SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント・ノードから転送されました。
- 0** MSU が SNA-MDS/LU 6.2 アラート転送プロトコルを使用してリモート・エントリー・ポイント・ノードから転送されなかったか、あるいはハードウェア・モニターが MSU を自動操作に実行依頼しませんでした。

転送メカニズムの詳細については、「423 ページの『第 26 章 集中操作』」を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例: LU 6.2 を使用して NETA.CNM01 から転送された MSU のチェック

```
IF HMFWDNSNA = '1' &  
HMEPNET = 'NETA' &  
HMEPNAU = 'CNM01' THEN COLOR(RED);
```

HMGENCAU(*pos* [*len*])

MSU の一般原因コードを、1 文字の 16 進数で戻します。

一般原因コードにより、次のことが示されます。

- 一般の汎用種別
- その MSU が作成される原因となった例外条件

一般原因コードに関する詳細については、システム・ネットワーク体系ライブラリー内の基本アラート (X'91') MS サブベクトルに関する記述を参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

HMGENCAU からは、非汎用アラート (X'0000') と、統計専用ではない RECMS とおよび RECFMS に関する値しか戻されません。MSU が以下のいずれかであれば、HMGENCAU からはヌルが戻されます。

- 汎用アラート (X'0000')

注: ここで、汎用とは、サブベクトル 92 を含む MSU すべてを指しています。汎用 MSU には、次のものがあります。

- サブベクトル 92 が組み込まれているアラート
- サブベクトル 92 が常に組み込まれているレゾリューション

- リンク・イベント (X'0001')
- 解決 (X'0002')
- PD 統計 (X'0025')
- リンク構成データ (X'1332')
- 統計専用 RECMS

注: 統計専用 RECMS とは、統計データのみが組み込まれるレコード保守統計のことを指します。これらのレコードの記録モードは、X'81'、X'86'、および X'87' であり、RECFMS のバイト 8、オフセット 1 に示されます。X'87' の場合、一時エラー (永続エラーではない) を表す RECMS のみが統計専用であると見なされます。

- 統計専用 RECFMS

注: 統計専用 RECFMS とは、統計データのみが組み込まれるレコード形式の保守統計のことを指します。これらのレコードのタイプは、1、4、5 であり、RECFMS レコードのバイト 8、オフセット 1 に示されます。

- ハードウェア・モニターによって自動操作に実行依頼されていない。

最大長: 16 進数 1 文字

タイプ: MSU

適用の対象: ヌル以外の値が戻されるすべての MSU

例 1: ヌル以外の値が指定された一般原因コードのチェック

```
IF HMGENCAU ^= '' &  
HMGENCAU = MYVAR THEN  
EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1));
```

この例では、ヌル以外の値が指定された一般原因コードがチェックされ、そのコードが変数 MYVAR で CLISTA コマンド・リストに渡され、そのコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

例 2: 特定の一般原因コードのチェック

```
IF HMGENCAU = HEX('01') THEN COLOR(RED);
```

この例では、一般原因コードが X'01' であるハードウェア・モニター MSU が赤色で示されるよう指定しています。

HMONMSU[(*pos* [*len*])]

MSU がハードウェア・モニターから自動操作に転送されたかどうかを示す 0 または 1 を戻します。

インディケーター

説明

- 1** MSU がハードウェア・モニターから転送されました。
- 0** MSU がハードウェア・モニターから転送されませんでした。これは、汎用受信先 (NVAUTO) により自動操作に実行依頼されたか、あるいは DSIAUTO または CNMAUTO を出したユーザー・アプリケーションから実行依頼されたかのいずれかである可能性があります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: すべての MSU

例 1: ハードウェア・モニターが実行依頼した MSU のチェック

```
IF HMONMSU = '1' THEN COLOR(RED);  
IF HMONMSU != '1' THEN COLOR(RED);
```

これらの例では、ハードウェア・モニターが実行依頼した MSU が赤色で示されるよう指定しています。

例 2: ハードウェア・モニターが実行依頼しなかった MSU のチェック

```
IF HMONMSU = '1' THEN ;  
IF HMONMSU = '0' THEN ;
```

これらの例では、ハードウェア・モニターが実行依頼していない MSU が自動操作に送られないよう指定しています。

HMORIGIN[(*pos* [*len*])]

MSU を送信するリソースの名前が戻されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

戻された値から後書きブランクは切り捨てられません。HMORIGIN により戻されるリソース名は、BNJMBDST に ALT_ALERT=ORIGIN が指定されているときの、ハードウェア・モニターの Alerts-Dynamic パネル、Alerts-Static パネル、および Alerts-History パネルに表示される名前と同じになります。

ALT_ALERT ステートメントに関する詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

リソース階層に複合リンクが存在していない場合、HMORIGIN で戻される値は HIER 条件項目で戻される値と同じになります。複合リンクが存在している場合、リソース名は同じでない可能性があります。複合リンクが存在するかどうかは、HMCPLINK 条件項目を使用して調べてください。詳しくは、HMCPLINK および HIER を参照してください。

ハードウェア・モニターが MSU を自動操作に実行依頼していない場合、HMORIGIN からはヌルの値が戻されます。

最大長: 8 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: GENALERT からの MSU のチェック

```
IF HMORIGIN = 'GENALERT' THEN COLOR(RED);
```

この例では、GENALERT という名前のリソースから送られた MSU が、赤色で示されるよう指定しています。

例 2: リソース名の抽出

```
IF HMORIGIN ^= '' &  
  HMORIGIN = MYVAR THEN  
  EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1));
```

この例では、ハードウェア・モニター MSU からリソース名が抽出され、その名前が変数 MYVAR で CLISTA コマンド・リストに渡され、そのコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

HMSECREC[(pos [len])]

ハードウェア・モニターが MSU の 2 次記録を取るかどうかを示す、0 または 1 を戻します。

インディケーター

説明

- 1** HIER 条件項目から戻されるリソース・レベルの MSU の 2 次記録を取ることを示します。詳しくは、HIER を参照してください。2 次記録については、NetView オンライン・ヘルプを参照してください。
- 0** 次のいずれかを示します。
 - MSU の 2 次記録は取られません。HMSECREC は、PD 統計 (X'0025') およびフレーム・リレー (X'1332') に関して常に 0 を戻します。これら MSU の 2 次記録が、ハードウェア・モニターによって取られることはありません。
 - ハードウェア・モニターは自動操作に MSU を実行依頼しませんでした。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 1 ビット

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例: 2 次記録のチェック

```
IF HMSECREC = '1' &  
  HIER = MYHIER THEN  
  EXEC(CMD('CLISTA 'MYHIER) ROUTE(ONE AUTO1));
```

この例では、MSU で 2 次記録がチェックされて、変数 MYHIER で HIER リソース階層レベル・データが CLISTA コマンド・リストに渡され、そのコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

HMSPECAU[(*pos* [*len*])]

MSU の特定のコンポーネント・コードを 16 進数 2 文字 (4 文字で表したもの) を戻します。一般原因コードが戻されます。

pos パラメーターは、比較が開始される位置を示します。デフォルトは 1 です。

len パラメーターは、比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

特定のコンポーネント・コードは、MSU が作成される原因となった例外条件と最も密接に関連しているコンポーネント、サブコンポーネント、または論理リソースを示します。具体的なコンポーネント・コードに関する詳細については、システム・ネットワーク体系内の基本アラート (X'91') MS サブベクトルに関する記述を参照してください。

非汎用アラート (X'0000') と、統計専用ではない RECMS および RECFMS にのみ値が戻されます。MSU が以下のいずれかであれば、HMSPECAU からはヌルの値が戻されます。

- 汎用アラート (X'0000')

注: ここで、汎用とは、サブベクトル 92 を含む MSU すべてを指しています。汎用 MSU には、次のものがあります。

- サブベクトル 92 が組み込まれているアラート
- サブベクトル 92 が常に組み込まれているレゾリューション
- リンク・イベント (X'0001')
- 解決 (X'0002')
- PD 統計 (X'0025')
- リンク構成データ (X'1332')
- 統計専用 RECMS

注: 統計専用 RECMS とは、統計データのみが組み込まれるレコード保守統計のことを指します。これらのレコードの記録モードは、X'81'、X'86'、および X'87' であり、RECFMS のバイト 8、オフセット 1 に示されます。X'87' の場合、一時エラー (永続エラーではない) を表す RECMS のみが統計専用であると見なされます。

- 統計専用 RECFMS

注: 統計専用 RECFMS とは、統計データのみが組み込まれるレコード形式の保守統計のことを指します。これらのレコードのタイプは、1、4、5 であり、RECFMS レコードのバイト 8、オフセット 1 に示されます。

- ハードウェア・モニターによって自動化テーブルに実行依頼されていない。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 16 進数 2 文字

タイプ: MSU

適用の対象: ヌル以外の値が戻されるすべての MSU

例 1: ヌル以外の値が指定されたコンポーネント・コードのチェック

```
IF HMSPECAU ^= '' &  
HMSPECAU = MYVAR THEN  
EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1);
```

この例では、ヌル以外の値が指定されたコンポーネント・コードがチェックされ、そのコードが変数 MYVAR で CLISTA コマンド・リストに渡され、そのコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

例 2: 特定のコンポーネント・コードのチェック

```
IF HMSPECAU = HEX('0001') THEN COLOR(RED);
```

この例では、コンポーネント・コードが X'0001' である MSU が、赤色で示されるよう指定しています。

HMUSRDAT[(*pos* [*len*])]

MSU のサブベクトル 33 にある、ユーザーが指定したデータが 5 文字で戻されます。

戻された値から後書きブランクが切り捨てられます。このデータは、ハードウェア・モニターによってフィルター操作されてから使用できます。ハードウェア・モニターでは、サブベクトル 33 にある印刷不可能なデータはすべて下線 () に変換され、小文字が大文字に変換されます。HMUSRDAT で戻される文字は、ハードウェア・モニターによって行われるすべての変換を反映します。また、この文字は、サブベクトル 33 にある文字とは同じでない可能性があります。HMUSRDAT を使用して、サブベクトル 33 にあるすべてのデータが、ハードウェア・モニターにより下線または大文字へ変換されているかどうかを判別してください。

また、MSUSEG を使用して、MSU のサブベクトル 33 にあるユーザーが指定したデータを受信することもできます。ただし、MSUSEG によりデータが変換されることはありません。

サブベクトル 33 データの詳細については、NetView オンライン・ヘルプで、GENALERT コマンドの UDAT オプション、SRFILTER コマンドの U オプションに関する記述を参照してください。

MSU が以下のいずれかであれば、HMUSRDAT からはヌルの値が戻されません。

- サブベクトル 33 が組み込まれていない。サブベクトル 33 が、RECMS または RECFMS レコードに示されることはありません。サブベクトル 33 を組み込むことができるのは、汎用主ベクトルのみです。ハードウェア・モニターは、自動操作に実行依頼された、任意の汎用主サブベクトルのサブベクトル 33 情報を受け取って処理します。
- フレーム・リレー (キー X'1332') である。
- ハードウェア・モニターにより自動操作に実行依頼されていない。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 5 文字

タイプ: MSU

適用の対象: ハードウェア・モニターが自動操作へ実行依頼するすべての MSU

例 1: 特定のユーザー指定データのチェック

```
IF HMUSRDAT = 'MYDAT' THEN COLOR(RED);
```

この例では、サブベクトル 33 にある、ユーザーが指定したデータ MYDAT が組み込まれたハードウェア・モニター MSU がチェックされ、赤色で示されるよう指定しています。

例 2: ユーザー指定データのチェック

```
IF HMUSRDAT ^= '' &  
HMUSRDAT = MYVAR THEN  
EXEC(CMD('CLISTA 'MYVAR) ROUTE(ONE AUTO1));
```

この例では、サブベクトル 33 にあるユーザーが指定したデータが組み込まれたハードウェア・モニター MSU がチェックされ、そのデータが変数 MYVAR で CLISTA コマンド・リストに渡され、このコマンド・リストがオペレーター AUTO1 に経路指定されるよう指定しています。

IFRAUIND [(*pos* [*len*])]

16 ビットの、AIFR インディケーター・フィールド IFRAUIND および IFRAUIN2 を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

ビット 16 は、メッセージが送信請求メッセージと非送信請求メッセージのいずれであったかを示します。

1 非送信請求

0 送信請求

送信請求メッセージおよび非送信請求メッセージの詳細については、91 ページの『第 9 章 自動化のための NetView プログラム情報の経路指定』を参照してください。それ以外のすべてのビットについては、「*IBM Tivoli NetView for z/OS* プログラミング:アセンブラー」を参照してください。

すべてのビットが B'0' である場合、IFRAUIND の値はヌル (") であると見なされます。この条件をテストするときは、ヌル (") キーワードと比較します。

最大長: 16 ビット

タイプ: 両方

IFRAUIN3 [(pos [*len*])]

DSIIFR によりマップされる 8 ビットの AIFR フィールド IFRAUIN3 を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

ドメイン間の優先順位を示すビット 1 および 2 に指定される値は、次のとおりです。

B'00' デフォルト優先順位

B'01' 低優先順位

B'10' 高優先順位

B'11' 受信先の優先順位がテストされます。

最大長: 8 ビット

タイプ: 両方

IFRAUI3X [(pos [*len*])]

DSIIFR によってメッセージに対してマップされる 32 ビットの 2 進フラグを示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

バイト 3 の値 (IFRAUI33) は以下のとおりです。

X'80' メッセージ削除の自動化

X'40' DOM は予期されていません。

X'20' トークンによる DOM が MVS によって出されました。

X'10' ローカル・コピーのみに対して DOM が出されました。

X'08' AUTO(YES) コンソールの所有者に AIFR が送信されました。

最大長: 32 ビット

タイプ: 両方

IFRAUI3X の使用上の注意:

1. 32 ビットの IFRAUI3X フラグの最初の 8 ビットは IFRAUIN3 です (詳細については IFRAUIN3 を参照してください)。
2. すべての IFRAUI3X フィールドに関する詳細については、NetView プログラムとともに出荷されているアセンブラー・マクロ DSIIFR を参照してください。

IFRAUSB2 [(pos [len])]

DSIIFR によりマップされる 2 文字の AIFR ユーザー・フィールド IFRAUSRB を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

フィールドに組み込まれた値がすべてブランクまたは 2 進数のゼロの任意の組み合わせである場合、IFRAUSB2 の値はヌル (") であると見なされます。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 2 文字

タイプ: 両方

注: ビットによる比較を行う場合、IFRAUSRB 条件項目を使用してください。

IFRAUSC2 [(pos [len])]

DSIIFR によりマップされる 128 ビットの AIFR ユーザー・フィールド IFRAUSRC を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

すべてのビットが B'0' である場合、IFRAUSC2 の値はヌル (") であると見なされます。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 128 ビット

タイプ: 両方

注: 文字による比較を行う場合、IFRAUSRC 条件項目を使用してください。

IFRAUSDR [(pos [len])]

メッセージまたは MSU の送信元である NetView プログラム・タスクの名前を指定します。IFRAUSDR は、1 から 8 文字の名前です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

IFRAUSRB [(pos [len])]

DSIIFR にマップされる 16 ビットの AIFR ユーザー・フィールド IFRAUSRB を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

すべてのビットが B'0' である場合、IFRAUSRB の値はヌル (") であると見なされます。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 16 ビット

タイプ: 両方

注: 文字による比較を行う場合、IFRAUSB2 条件項目を使用してください。

IFRAUSRC [(pos [len])]

DSIIFR にマップされる 16 文字の AIFR ユーザー・フィールド IFRAUSRC を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

すべてのバイトが文字ブランクまたは 2 進数ゼロの任意の組み合わせである場合、IFRAUSRC の値はヌル (") であると見なされます。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 16 文字

タイプ: 両方

注: ビットを使用して比較する場合、IFRAUSC2 条件項目を使用してください。

IFRAUTA1 [(pos [len])]

AIFR フィールド IFRAUTA1、IFRAUTA2、IFRAUTA3、IFRAUTA4、IFRAUTA5、および IFRAUTA6 を示します。詳細については、DSIIFR のフィールド IFRAUTA1 から IFRAUTA6 までを参照してください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 説明

1、2、25

HOLD アクション

5、6、26

SYSLOG アクション

7、8、27

NETLOG アクション

9、10、28

HCYLOG アクション

11、12、29

DISPLAY アクション

13、14、30

BEEP アクション

20 メッセージが MVS から出されているか

24 メッセージが (WTOR のような) アクション・メッセージであるか

47 自動化ベクトルの拡張が行われているか

48 データ・バッファーに表示ベクトルがあるか

すべてのビットの説明については、「*IBM Tivoli NetView for z/OS*プログラミング:アセンブラー」を参照してください。

すべてのビットが B'0' である場合、IFRAUTA1 の値はヌル (") であると見なされます。この条件をテストするときは、ヌル (") キーワードと比較します。

最大長: 48 ビット

タイプ: 両方

IFRAUWF1 [(*pos* [*len*])]

DSIIIFR にマップされる AIFR フィールド IFRAUWF1、IFRAUWF2、IFRAUWF3、および IFRAUWF4 を示します。ここには、32 ビットの MVS WTO 情報が含まれています。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

- 6 メッセージが WTOR であるかどうか
- 7 メッセージが抑止されたかどうか
- 8 メッセージがすべてにブロードキャストされたかどうか
- 9 ジョブ名が表示されるかどうか
- 10 統計が表示されるかどうか
- 14 セッションが表示されるかどうか

すべてのビットが B'0' である場合、IFRAUWF1 の値はヌル (") であると見なされます。この条件をテストするときは、ヌル (") キーワードと比較します。

最大長: 32 ビット

タイプ:メッセージ

INTERVAL(*occurrence_number*)

条件項目に対する評価が、*occurrence_number* に指定された数の倍数に当たるオカレンスごとに行われているかどうかのインディケーターを戻します。この条件項目を使用して、その条件が定期的に発生する場合に行われるアクションを指定します。*occurrence_number* パラメーターには、チェックが行われる間隔を指定します。1 から 1000000000 までの値を指定できます。

INTERVAL によって戻される値は、次のとおりです。

- 1 評価を受ける条件項目が、*occurrence_number* に指定された値の倍数であることを示します。
- 0 それ以外のすべてのオカレンスについて戻されます。

例えば、条件項目が 5 番目のオカレンスから 5 つおきのオカレンスごとに (10 番目、15 番目、というように) 評価されるときには INTERVAL(5) から値 1 が戻され、それ以外のオカレンスについて評価されるときには値 0 が戻されます。

評価カウントは、自動化テーブルを通して一致するかどうかを順次検索している間に、INTERVAL 条件項目に達したときにのみ増分されます。評価カウントは、以下のいずれかに該当する場合には、増分されません。

- そのテーブルでこれより前のステートメントの条件に一致したために、INTERVAL 条件項目を持つステートメントに達しない場合。
- BEGIN-END 条件論理の結果、そのステートメントが評価されなかった場合。
- 論理 AND (&) 演算子と連結されている自動化テーブル・ステートメントにおける前の条件が、偽と評価した場合。

最大長: 1 ビット

タイプ: 両方

例: INTERVAL キーワードにより評価されるステートメント

```
IF MSGID = 'XYZ123I' &  
    INTERVAL(5) = '1' THEN  
    <actions>;
```

この例では、自動化テーブルを通しての順次検索がそのステートメントに達し、さらにメッセージ ID が XYZ123I である場合にのみ評価カウントが増分されます。自動化アクションは、このステートメントに達した XYZ123I メッセージのうち、5 の倍数の順番 (5 番目、10 番目、15 番目 ...) のものについてのみ実行されます。

INTERVAL の使用上の注意:

1. **有効な間隔値の選択** - NetView プログラムは、評価カウントが間隔の倍数に達したかどうかを判別する前に、評価カウントを増分します。間隔値に 1 を指定すると条件項目は常に同じであると評価されるため、1 を指定しないでください。例えば、ステートメント INTERVAL(1) = '1' は、常に真になります。
2. **評価カウントのリセット** - 次のいずれかのイベントが起きた場合、評価カウントは 0 にリセットされます。
 - アクティブな自動化テーブルが、AUTOTBL コマンドが発行されて置き換えられた。
 - NetView 自動化テーブル機能がオフに切り替えられた。
 - NetView プログラムは終了します。

JOBNAME [(pos [len])]

JOBNAME は、受信したメッセージの発信元である MVS ジョブの名前を指定する、JOBNAME は、1 から 8 文字の名前です。

JOBNAME は、メッセージを発信したジョブの名前であるため、そのメッセージが参照しているジョブの名前と常に同じであるとは限りません。MVS から NetView プログラム・ジョブに関するメッセージが出されたときは、名前が異なっている可能性があります。ジョブの始動またはシャットダウン操作の間にメッセージが出される場合、JOBNAME には (実際のジョブ名ではなく) 開始プログラムの名前が含まれていることがあるため、ジョブ名は JOBNAME キーワードではなくメッセージ・テキストから抽出するようにしてください。

同じ情報が、MSGCOJBN 条件項目にも使用可能です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

メッセージが MVS システムから受信したものではないか、ジョブ名と関連付けられていない場合、JOBNAME の値はヌル (") であると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

最大長: 8 文字

タイプ:メッセージ

JOBNUM [(*pos* [*len*])]

MVS によって受信したメッセージの発信元であるジョブに割り当てられる番号です。JOBNUM は 8 桁の番号で、英字の接頭部およびブランクを組み込みます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

メッセージが MVS から受信したものではないか、ジョブ番号と関連付けられていない場合、JOBNUM の値はヌル (") であると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

最大長: 8 文字

タイプ:メッセージ

KEY [(*pos* [*len*])]

メッセージに関連付けられたキーを指定します。KEY には表示不可能な値が含まれている可能性があります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

メッセージが元はメッセージ・データ・ブロック (MDB) であった場合にのみ、KEY に値を含みます。

最大長: 8 文字

タイプ:メッセージ

LINEPRES [(*pos* [*len*])]

次の 4 つの表示属性に関する値を指定します。

- アラーム制御
- 色
- 強調表示
- 輝度

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

LINEPRES は、メッセージまたは MSU の最初のバッファから取られる 4 バイトの値です。

LINEPRES がヌル以外の値で、LINETFLG のビット 16 がオンに設定されている場合、メッセージと MSU の表現には、LINEPRES 値が使用されます。LINEPRES 値は、次の 2 つのソースのうちいずれかから取られます。

- メッセージ・データ・バッファ (MDB) に指定された表示のオーバーライド
- 事前に取りられた自動化テーブル・アクションにより指定された表示のオーバーライド

1 つ以上の表示属性が (COLOR、HIGHINT、または XHILITE アクションが指定された) 自動化テーブルによって設定された場合、メッセージまたは MSU に指定された 4 つの表示属性はすべて LINEPRES フィールドにコピーされ、そのメッセージまたは MSU を表示する際に使用されます。自動化テーブルによって設定された以外の属性は、MDB オーバーライド・フィールド、MSGGFGPA 内のフィールド、あるいは OVERRIDE コマンドか DEFAULTS SCRNFMT コマンドで指定された値から取得されたものです。

LINEPRES がヌルである場合、メッセージまたは MSU の属性は、その他 3 つのソースのいずれかから取られます。

- MSGGFGPA 内のフィールド
- OVERRIDE コマンドか DEFAULTS SCRNFMT コマンドで指定された値
- ハードウェア・モニターのデフォルト (MSU の場合)

LINEPRES がヌルであっても、その他の表示属性は、その表示の際にこのメッセージに適用されます。LINEPRES がヌルの場合、MSGGFGPA 内のフィールドで表示属性をチェックできます。

4 つの LINEPRES 文字の意味および指定可能な値は、次のとおりです。

バイト 説明

1 制御フィールド

値	意味
80	MVS アラーム・オン・インディケータ
00	MVS アラーム・オフ・インディケータ

バイト 1 は、MVS インディケータです。このインディケータは、NetView プログラムでは使用しません。このアラームを制御し

ている NetView プログラムインディケータは、IFRAUTA1 のビット 13、14、および 30 にあります。IFRAUTA1 アラーム・インディケータは、自動化テーブル内の BEEP アクションによって設定できます。

2 カラー・フィールド

値	意味

00	フォアグラウンド・カラー

F0	表示バックグラウンド。表示画面上は黒、プリンター上は白。

F1	青

F2	赤

F3	ピンク (紫紅色)

F4	緑

F5	青緑 (シアン)

F6	黄

F7	無色表示。表示画面上は白、プリンター上は黒。

このフィールドは、自動化テーブルの COLOR アクションにより設定できます。値が 00 の場合は、MSGGFGPA のフィールド、あるいは OVERRIDE コマンドか DEFAULTS SCRNFMT コマンドによって、特定のフォアグラウンド・カラーが設定されます。

3 強調表示されたフィールド

値	意味

00	強調表示なし

F1	明滅

F2	反転表示

F4	下線

このフィールドは、自動化テーブルの XHILITE アクションにより設定できます。

4 輝度フィールド

値	意味

E4	通常の輝度

E8	高 (明るい) 輝度

このフィールドは、自動化テーブルの HIGHINT アクションによって設定できます。

最大長: 4 バイト

タイプ: 両方

LINETFLG [(pos [len])]

メッセージまたは MSU の最初のテキスト・バッファから取られる 16 ビット値を示します。

LINETFLG のビット 16 は、LINEPRES に記述された表示属性が、このメッセージまたは MSU に適用されるかどうかを示します。ビット 16 の値は、次のとおりです。

値 意味

0 LINEPRES から戻される属性は、メッセージまたは MSU には適用されません。

1 MSGGFPA の属性を指定変換するために表示属性が指定されており、このメッセージまたは MSU に適用されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、pos で始まるストリングの残りの部分です。

最大長: 16 ビット

タイプ: 両方

MCSFLAG [(pos [len])]

16ビットMVS複数コンソール・サポート・フラグ。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、pos で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

2 メッセージがアクティブであれば、コンソールのキューに入れられます。

3 メッセージは、コマンド応答 WTO です。

5 メッセージは、WTOR への応答です。

6 メッセージは、すべてのアクティブなコンソールにブロードキャストされます。

7 メッセージは、ハードコピーのキューにのみ入れられます。

8 メッセージは、無条件にコンソールのキューに入れられます。

9 メッセージは、タイム・スタンプ処理を受けません。

14 メッセージは、ハードコピーのキューには入れられません。

REXX、高水準言語 (HLL)、NetView コマンド・リスト言語 (CLIST) における MCSFLAG 値は、MCSFLAG の使用可能な 16 ビットのうち 8 ビット

トのみ戻します。自動化テーブルの条件項目 MCSFLAG は、16 ビットすべてを戻します。表 8 に、自動化テーブルの条件項目と、REXX、コマンド・リスト、および HLL 変数との違いを示します。自動化テーブル用として説明されているビットでない方のビットは使用しないでください。

表 8. REXX、コマンド・リスト、および HLL と MCSFLAG 条件項目の比較

ビット	MCSFLAG 条件項目	REXX、CLIST、および HLL
1		REG0
2	REG0	QREG0
3	RESP	RESP
4		REPLY
5	REPLY	BRDCST
6	BRDCST	HRDCPY
7	HRDCPY	NOTIME
8	QREG0	NOCPY
9	NOTIME	
10		
11		
12		
13		
14	NOCPY	
15		
16		

すべてのビットが B'0' である場合、MCSFLAG の値はヌル (") であると見なされます。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 16 ビット

タイプ:メッセージ

MSGAUTH [(pos [len])]

MSGAUTH は、メッセージが許可プログラムから出されたかどうかを示す 2 ビットのインディケーターです。比較項目はビット・ストリングです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGAUTH の値は、次のとおりです。

B'00' メッセージは MVS から出されたものではありません。

B'01' 未使用

B'10' 無許可プログラムから出された WTO

B'11' 許可プログラムから出された WTO

メッセージが MVS から出されたものではない場合、MSGAUTH の最初のビットの値はヌル (") であると見なされます。メッセージが無許可の MVS プログラムから出されたものである場合、2 番目のビットの値はヌル (") であると見なされます。メッセージが MVS から出されたものではない場合、両方のビットの値はヌルであると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

最大長: 2 ビット

タイプ:メッセージ

MSGCATTR [(pos [len])]

MSGCATTR は、MVS メッセージ属性フラグを示す 16 ビットのフィールドです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

- 1** メッセージは、抑止されました。
- 2** メッセージは、コマンド応答です。
- 3** メッセージは、許可プログラムから出されました。
- 4** メッセージは、自動化メッセージ保存機能 (AMRF) により保存されます。

MSGCATTR に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 16 ビット

タイプ:メッセージ

MSGCMISC [(pos [len])]

MSGCMISC は、MVS のその他の経路指定フラグを示す 8 ビットのフィールドです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

- 1** 配布不可能なメッセージが表示されるかどうかを示します。
- 2** 配布不可能なメッセージのみが表示されるかどうかを示します。

- 3 メッセージが ID のみによってキューに入れられるかどうかを示します。
- 4 メッセージが、メッセージ処理機能 (MPF) テーブル内で NetView プログラムの自動操作に適するとして、マークが付けられているかどうかを示します。

MSGCMISC に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 8 ビット

タイプ:メッセージ

MSGCMLVL [(pos [len])]

MSGCMLVL は、MVS メッセージ・レベル・フラグを示す 16 ビットのフィールドです。比較項目はビット・ストリングです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

- 1 WTOR
- 2 即時アクション・メッセージ
- 3 重要な最終アクション・メッセージ
- 4 最終アクション・メッセージ
- 5 通知メッセージ
- 6 ブロードキャスト・メッセージ

MSGCMLVL に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 16 ビット

タイプ:メッセージ

MSGCMSGT [(pos [len])]

MSGCMSGT は、MVS メッセージ・タイプ・フラグを示す 16 ビットのフィールドです。これらのビットは、MVS コンソールに表示されるメッセージに適用されますが、NetView プログラムでは使用されません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットを確認してください。

ビット 意味

- 1 ジョブ名が表示されます。
- 2 状況が表示されます。

MSGCMSGT に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 16 ビット

タイプ:メッセージ

MSGCOJBN [(*pos* [*len*])]

発信元のジョブ名を指定します。MSGCOJBN は、1 から 8 文字の名前です。(同じ情報が、JOBNAME 条件項目でも使用可能です。)

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGCOJBN に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 8 文字

タイプ:メッセージ

MSGCPROD [(*pos* [*len*])]

MSGCPROD は、MVS プロダクト・レベルを示すもので、4 文字の MVS 制御プログラム・オブジェクトのバージョン・レベル、4 文字の制御プログラム名 MVS、および 8 文字の発信元システムの ID から構成された、16 文字のストリングです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGCPROD に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 16 文字

タイプ:メッセージ

MSGCSPLX [(*pos* [*len*])]

受信したメッセージの発信元である MVS SYSPLEX の名前です。

MSGCOJBN は、1 から 8 文字の名前です。*pos* パラメーターには、比較が開始される位置を指定します (デフォルト値は 1 です)。比較項目は、解析テンプレートです。

メッセージが MVS SYSPLEX から受信されたものでないか、関連する SYSPLEX 名をもたないか、またはメッセージが元はメッセージ・データ・

ブロック (MDB) でなかった場合、MSGCSPLX の値はヌル (") であると見なされます。このようなケースを検出するときは、ヌル (") キーワードと比較します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ:メッセージ

MSGDOMFL [(*pos* [*len*])]

MSGDOMFL は、MVS DOM フラグを示す 8 ビットのフィールドです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下のビットをチェックします。

ビット 意味

- 1** メッセージ ID (MSGID) による DOM
- 2** システム ID (SYSID) による DOM
- 3** NetView アドレス・スペース ID (ASID) による DOM
- 4** ジョブ・ステップ TCB による DOM
- 5** トークンによる DOM

MSGDOMFL に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

複数コンソール・サポート・コンソールは、デフォルトが DOM(NORMAL) の受信先としてセットアップされます。この結果、MVS から送られ、これらのコンソールで受信された DOM は、1 ビットのフラグを持ちます。SYSID、ASID、TCB、および TOKEN ビット・フラグは、DOM を MVS プログラムから受信した場合、通常、オンにはセットされません。

最大長: 8 ビット

タイプ:メッセージ

MSGGBGPA [(*pos* [*len*])]

MSGGBGPA は、バックグラウンド表示属性を 4 バイトの 16 進値で示します。比較項目は、解析テンプレートです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

以下にバイトを説明します。

バイト 意味

- 1 バックグラウンドの制御
- 2 バックグラウンドの色
- 3 バックグラウンドの強調表示
- 4 バックグラウンドの輝度

各バイトの値の詳細については、LINEPRES を参照してください。

MSGGBGPA に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 4 バイト

タイプ:メッセージ

MSGGDATE [(*pos* [*len*])]

メッセージの発信元が MDB に入った日付を示します。この日付は、*yyyddd* という形式 (*yyyy* は年を、*ddd* はその年の何日目かを示します) で、7 文字で示されます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGGDATE に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 7 文字

タイプ:メッセージ

MSGGFGPA [(*pos* [*len*])]

フォアグラウンド表示属性を示します。MSGGFGPA には、4 バイトの 16 進数値を指定します。比較項目は、解析テンプレートです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

バイト記述は、次のとおりです。

バイト 意味

- 1 フォアグラウンドの制御

- 2 フォアグラウンドの色
- 3 フォアグラウンドの強調表示
- 4 フォアグラウンドの輝度

各バイトの値の詳細については、LINEPRES を参照してください。

MSGGFGPA に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 4 バイト

タイプ:メッセージ

MSGGMFLG [(pos [len])]

MVS 汎用メッセージ・フラグを指定します。MSGGMFLG は、16 ビット・フィールドです。ビット 1 は、DOM を示します。別のビットはテストしないでください。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGGMFLG に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 16 ビット

タイプ:メッセージ

MSGGMID [(pos [len])]

MSGGMID は、4 文字の MVS メッセージ ID を示します。MSGGMID には、表示不能な文字が含まれている可能性があります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGGMID に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 4 文字

タイプ:メッセージ

MSGGTIME [(pos [len])]

MSGGTIME は、MVS がメッセージと関連付けを行った時間を、*hh.mm.ss.th* という形式 (*hh* は時間、*mm* は分、*ss* は秒、*th* は 100 分の 1 秒を示します) の 11 文字 (ピリオドの数も含む) で示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

MSGGTIME に値が含まれるのは、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限られます。

最大長: 11 文字

タイプ:メッセージ

MSGID [(pos [len])]

MSGID は、受信したメッセージのメッセージ ID を1 から 255 文字で示します。メッセージ ID は通常、メッセージの最初のトークンです。メッセージとともに REPLYID が送信された場合、REPLYID は最初のトークンとして使用されません。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 255 文字

タイプ:メッセージ

MSGSRCNM [(pos [len])]

ソース名を 1 から 17 文字で示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

このソース名は、DSIMMDB または CNMPMDB のアプリケーション・プログラミング・インターフェース (API) 呼び出しによって提供されたソース・オブジェクトの ID です。

DSIMMDB に関する詳細については、「*IBM Tivoli NetView for z/OS*プログラミング:アセンブラー」を参照してください。CNMPMDB に関する詳細については、「*IBM Tivoli NetView for z/OS*プログラミング: PL/IおよびC」を参照してください。

ソース名は、以下の規則に従って、ソース・オブジェクトの中から選択されます。

1. 最初の別名 (ある場合)
2. ネットワーク・アドレス可能単位 (NAU) 名に連結された最初のネットワーク ID。シーケンス内に両方存在する場合は、間にピリオド (.) を入れる。
3. 最初の存在する NAU 名

4. スtring N/A (このリスト内のその他の名前がソース・オブジェクトに指定されていない場合)
5. ヌル (ソース・オブジェクトがない場合)

ソース・オブジェクトの定義方法および DSIAIFRO のマッピングの詳細については、「*IBM Tivoli NetView for z/OS* プログラミング:アセンブラー」を参照してください。

注: メッセージが元はソース・オブジェクトと関連付けられた MDB であった場合にのみ、この機能に値が含まれます。

最大長: 17 文字

タイプ:メッセージ

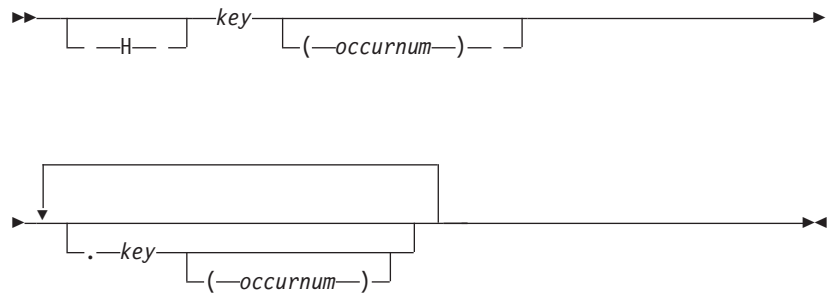
MSUSEG (location [byte [bit]])

MSU の 1 セグメントの内容を示します。比較項目は、ビット・ストリングまたは解析テンプレートです。

location

比較の対象となるデータの位置を指定します。パラメーターの構文は、次のようになります。

Location パラメーター



H MDS-MU の場合、最初の *key* が (主ベクトル・レベルではなく) MDS-MU レベルで取得されることを示します。このパラメーターを使用する場合で、処理される MSU が MDS-MU ではないときは、MSUSEG からヌルの値が戻されます。

key 汎用データ・ストリーム (GDS) 変数の、1 バイトまたは 2 バイトの 16 進数 ID を 2 か 4 文字で表したものあるいは主ベクトル、サブベクトル、サブフィールド、またはサブ・サブフィールドのキーを指定します。

複数の *key* を使用できますが、それらの間はピリオドで区切ってください。追加の *key* はそれぞれ、先行する *key* で示された構造内の下位構造を指定します。

occurnum

GDS 変数、主ベクトル、サブベクトル、サブフィールド、またはサブ・サブフィールドのオカレンス番号 (1 から順にカウントします) を指定します。アスタリスク (*) を指定した場合、任意のオカレンスを意味します。例えば、サブベク

トル・レベルで *occurnum* に 2 を指定した場合、*key* サブベクトルの 2 番目のインスタンスが必要であるということになります。* を指定した *occurnum* は、*key* に指定したキー (あれば) を持つ最初のサブベクトルと指定した比較項目が等しいことを意味します。*occurnum* の最大値は 32767 であり、デフォルトは 1 です。

byte *location* に指定された最下位の *key* 内のバイト位置を指定します。位置 0 ではなく位置 1 が最初のバイトを示します。最大値は 32767 であり、デフォルトは 1 です。

bit *byte* で指定されたバイト内のビット位置を指定します。ビット位置には、1 から 8 までの任意の数を指定できます。位置 0 ではなく位置 1 が最初のビットを示します。ビット位置を指定する場合、比較項目はビット・ストリングになります。それ以外の場合の比較項目は解析テンプレートです。

MSUSEG では長さ指定をサポートしていません。ユーザーは、ある変数に MSUSEG を割り当て、その後 VALUE 条件ステートメントにおいてその変数 (*pos* と *len* を含む) を使用できます。

最大長: 可変

タイプ: MSU

使用上の注意:

1. MSUSEG の使用例については、367 ページの『MSU を自動化する自動化テーブル・ステートメントの作成』を参照してください。
2. MSUSEG 自動化テーブル・ステートメントは、REXX の MSUSEG 関数や、NetView コマンド・リスト言語の &MSUSEG 制御変数に置き換えられません。MSU の位置を指定する形式は同じですが、構文の細部に違いがあります。

MVSLEVEL [(*pos* [*len*])]

現在実行中の MVS レベルを、8 文字のストリングで示します。

ユーザーは、LISTVAR コマンドを使用して、該当システムの MVS レベルを判別することができます。

MVSLEVEL 条件項目と対比させるため、MSGCPROD 条件項目には、メッセージ出力元の MVS システム・レベルを指定します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

現在実行中のシステムが MVS 以外の場合、MVSLEVEL の値はヌルになります。

最大長: 8 文字

タイプ: 両方

NETID [(pos [len])]

VTAM ネットワークの ID を指示します。このフィールドには最大 8 文字の長さまで入ります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

NetView プログラムがアクティブのときに VTAM がアクティブになっていない場合、NETID の値はヌルになります。

最大長: 8 文字

タイプ: 両方

NETVIEW [(pos [len])]

現在実行している NetView プログラムのバージョンとリリースを示します。NETVIEW の値は NVvr の形式の 4 文字フィールドで、*v* はバージョン番号、*r* はリリース番号です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 4 文字

タイプ: 両方

NUMERIC (variable[pos [len]])

テキスト値から数値に変換し、解析テンプレートに指定されたリテラルの数値と比較するための *variable* (変数) を示します。

pos

変数値内で数値に変換するテキストが始まる位置。デフォルト値は 1 です。

len

数値に変換するテキスト値の長さ。この値は正数でも負数でも可能ですが、小数点はサポートされていません。デフォルトは、*pos* で始まる変数の残りの部分です。

最大長: 255 文字

タイプ: 両方

NVCLOSE [(pos [len])]

NetView プログラムが現在 CLOSE 処理を実行中かどうかを示します。1 ビットのインディケータです。NVCLOSE の値は、次のとおりです。

- 1** NetView プログラムが CLOSE 処理を実行している。
- 0** NetView プログラムは CLOSE 処理を実行していない。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

注: NVCLOSE チェックは注意して使用してください。NVCLOSE を自動化ステートメントの唯一の条件項目として使用すると、ループ条件が起きる可能性があります。NVCLOSE 条件項目の使用目的は、次の例に示しているように、他の条件項目と合わせて使用することです。

最大長: 1 ビット

タイプ: 両方

例:

例えば、MYCMD を使用して DSI008I メッセージで参照されているタスクを再始動する場合、次の自動化ステートメントによって、CLOSE 処理中にタスクを再始動するのを妨げます。

```
IF MSGID = 'DSI008I' & NVCLOSE ^= '1' THEN  
  EXEC(CMD('MYCMD')ROUTE(ONE AUTO1));
```

NVCLOSE の使用上の注意::

1. その時点で CLOSE 処理が実行されていないときは、NVCLOSE の値の評価はヌル (") になります。このようなケースをテストするときは、ヌル (") キーワードと比較します。
2. PPT タスクでどのタスクがアクティブかを判別する自動操作を実行している場合に、NetView プログラムが CLOSE STOP 処理を使用すると、ループが起きる可能性があります。このループは、NetView プログラムが CLOSE STOP 処理を完了するうえで妨げになる可能性があります。例えば、PPT がさまざまなタスクに EXCMD を出している場合、NetView プログラムは PPT がそのタスクを完了するまで終了しません。

NVDELID [(*pos* [*len*])]

DOM コマンドによってアクション・メッセージを削除するための入力として使用できるメッセージの 24 文字の EBCDIC 値です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 24 文字

タイプ:メッセージ

OPID [(*pos* [*len*])]

自動化テーブルが処理中であるときの、オペレーターまたはタスクの ID です。OPID は 1 から 8 文字の ID です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ: 両方

OPSYSTEM [(*pos* [*len*])]

NetView プログラムをコンパイルしたオペレーティング・システムです。このフィールドは最大 7 文字の長さまで入ります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 7 文字

タイプ: 両方

ROUTCDE [(*pos* [*len*])]

メッセージに割り当てられている 1 つ以上の MVS 経路コード・ビットを識別します。1 つのメッセージには、最大 128 までの経路コード・ビットを持つことができます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

コード値の詳細については、MVS ライブラリーを参照してください。

最大長: 128 ビット

タイプ:メッセージ

SESSID [(*pos* [*len*])]

受信メッセージを送信した NetView 端末アクセス機能 (TAF) セッションを示す 1 から 8 文字の ID です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

TAF セッションを介してメッセージを受信していない場合、SESSID の値は 16 進ゼロ (X'00') になります。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 8 文字

タイプ:メッセージ

SYSCONID [(*pos* [*len*])]

メッセージに関連する MVS システムの、コンソール名です。システム・コンソール名の長さは 1 から 8 文字です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 8 文字

タイプ:メッセージ

SYSID [(*pos* [*len*])]

メッセージを送信した MVS システムを示す 1 から 8 文字の ID です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

SYSID は、シスプレックスで使用できます。SYSID を既存の自動化テーブル・ステートメントに追加して、特定のシステムからのメッセージを遮断したり、このシステム ID に基づく特定の自動化テーブル・アクションを呼び出したりできます。

この例は、メッセージの発信元が MVS または MVS 以外であるかにかかわらず、システムに対してローカルなメッセージを処理するのに SYSID を使用する方法を示しています。この例では、ローカル・システム名は SYSA です。

```
IF (SYSID = 'SYSA' | SYSID = '') THEN
  BEGIN;
  .
  .
  .
END;
```

システム SYSA の MVS から発信されたメッセージは、SYSID の値が 'SYSA' であるため、SYSID の検査に合格します。発信元が MVS でないメッセージでも、システム SYSA に対してローカルであるメッセージもそれらの SYSID がヌルであるため、検査に合格します。

最大長: 8 文字

タイプ:メッセージ

SYSPLEX [(*pos* [*len*])]

SYSPLEX には、受信したメッセージが自動化されている MVS SYSPLEX の名前を SYSPLEX は、1 から 8 文字の名前です。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

メッセージが MVS SYSPLEX で自動化されていないか、あるいは関連する SYSPLEX 名をもたない場合、SYSPLEX の評価の値はヌル ("") になります。このようなケースを検出するときは、ヌル ("") キーワードと比較します。

最大長: 8 文字

タイプ: 両方

TASK [(*pos* [*len*])]

自動化テーブルが処理中であるときのタスクのタイプです。TASK は、3 文字のストリングです。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

TASK の値は、以下のとおりです。

HCT ハードコピー・タスク。

DST データ・サービス・タスク。DST は、CNMSTYLE メンバーに MOD=DSIZDST が指定されているオプション・タスクです。

OPT 任意指定のタスク。CNMCSSIR のタスクは、常に OPT の値になります。

OST オペレーター端末タスク。自動化タスクは、OST の値になります。自動タスクを他の OST と区別するには、AUTOTASK 条件項目と DISTAUTO 条件項目を使用してください。

NNT NetView 間タスク。

MNT NetView プログラムのメインタスク。

PPT 基本 POI タスク。

最大長: 3 文字

タイプ: 両方

TEXT [(*pos* [*len*])]

受信メッセージのテキストを指定します。TEXT は 1 から 25 文字のストリングで、MSGID を含むメッセージ・テキスト全体が入ります。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

比較項目は、解析テンプレートです。

最大長: 255 文字

タイプ:メッセージ

THRESHOLD(*occurrence_number* [*time_period*])

前回の *time_period* の間、少なくとも *occurrence_number* の回数だけしきい値条件項目に達したかどうかのインディケーターを戻します。

THRESHOLD 条件項目は、指定された時間枠内に、少なくとも指定された回数が発生したときに取るべき特定アクションを指定したいときに使用します。これらのオカレンス検出条件項目の例については、269 ページの図 60 を参照してください。

occurrence_number

指定された時間枠内にしきい値条件に達したオカレンスの数を指定します。指定できる値は、1 から 1000 です。

time_period

しきい値の時間間隔を指定します。デフォルトは 24 時間です。時間の指定形式は *ddd hh:mm:ss* で、それぞれ次のものを表します。

ddd

日数で、0 から 365 の範囲になります。*ddd* を指定する場合は、*hh:mm:ss* も指定する必要があります。

hh:mm:ss

時間 (00 から 23 の範囲)、分 (00 から 59 の範囲)、および秒 (00 から 59 の範囲)。

この時間枠にゼロの値を指定することはできません。 *time_period* に 1 つの数値だけを指定し、コロン区切り文字 (:) が付いていない場合、NetView プログラムはこれを分の値と見なします。

THRESHOLD の正しい指定方法については、表 9 を参照してください。

表 9. THRESHOLD の正しい指定の例

指定	オカレンスの				
	数	日数	時間数	分数	秒数
THRESHOLD(3) = '1'	3	1			
THRESHOLD(4 1 00:00:00) = '1'	4	1			
THRESHOLD(5 1:00) = '1'	5		1		
THRESHOLD(6 0 1:00:00) = '1'	6		1		
THRESHOLD(7 10) = '1'	7			10	
THRESHOLD(8 :30) = '1'	8				30
THRESHOLD(9 10 10) = '1'	9	10		10	
THRESHOLD(10 10 00:10:00) = '1'	10	10		10	

デフォルト時間枠の 24 時間 (1 日) を受け入れる場合以外は、*time_period* の全エレメント (日数、時間、分、および秒) を指定してください。必須条件ではありませんがこうすることで、時間枠の間違いを避けられます。

THRESHOLD によって戻される値は、次のとおりです。

- 1 指定された時間枠内のオカレンスの数が *occurrence_number* の値以上であることを示します。
- 0 それ以外のすべてのオカレンスについて戻されます。

評価カウントは、自動化テーブルの (一致の) 順次検索中に、THRESHOLD 条件項目に到達した場合にのみ増分されます。評価カウントは、以下のシチュエーションのいずれかに該当する場合には、増分されません。

- そのテーブルでこれより前のステートメントの条件に一致したために、THRESHOLD 条件項目を持つステートメントに達しない場合。
- BEGIN-END 条件論理の結果、そのステートメントが評価されなかった場合。
- 論理 AND (&) 演算子と連結されている自動化テーブル・ステートメントにおける前の条件が、偽と見なされた場合。

図 32 で評価カウントが増分されるのは、アクティブな自動化テーブルを介した順次検索がこのステートメントに到達し、かつ メッセージ ID が XYZ123I である場合に限られます。

3 時間の時間枠の自動化テーブル検索中にこのステートメントに達した 5 番目 (以降) の XYZ123I メッセージについてのみ、自動化アクションが発生します。

```
IF MSGID = 'XYZ123I' &  
  THRESHOLD(5 0 3:00:00) = '1' THEN  
  <actions>;
```

図 32. THRESHOLD キーワードに評価されるステートメント

最大長: 1 ビット

タイプ: 両方

THRESHOLD の使用上の注意::

1. **経過時間と時間枠** THRESHOLD 条件の評価を行うたびに、前の時間枠 (*time_period* で指定) 中のオカレンスの数がしきい値に達しているかどうか検査されます。時間が経過するに従って、前のオカレンスが *time_period* 内でなくなっている可能性があります。
2. **実際的なオカレンス値の選択** NetView プログラムは、しきい値に達したかどうかを判別する前に、評価カウントを増分します。オカレンス値に 1 を指定すると条件項目は常に同じであると評価されるため、1 を指定しないでください。例えば、条件項目 THRESHOLD(1 x x:xx:xx) = '1' は常に真となります。
3. **評価カウントのリセット** - 次のいずれかのイベントが起きた場合、評価カウントは 0 にリセットされます。
 - アクティブな自動化テーブルが、AUTOTBL コマンドが発行されて置き換えられた。

- NetView 自動化テーブル機能がオフに切り替えられた。
- NetView プログラムが終了します。

4. **アクションの限界の定義** 終了オカレンス番号 (特定アクションをそれ以上発生させないところ) を定義するには、開始オカレンス番号と終了オカレンス番号を示す 2 つの THRESHOLD 条件を連結したものを 1 ステートメントとします。例えば、任意の 1 時間の時間枠に、メッセージ XYZ123I の 3 番目から 6 番目までのオカレンスについてのみ特定のアクションを発生させたい場合、以下の自動化テーブル・ステートメントを使用します。

```
IF MSGID = 'XYZ123I' &
  THRESHOLD(3 0 01:00:00) = '1' &
  THRESHOLD(5 0 01:00:00) = '0' THEN
  <actions>;
```

ステートメント内の 2 番目の THRESHOLD 条件項目は、1 番目のしきい値に適合 (1 時間の枠内で XYZ123I メッセージの 3 番目のオカレンスで開始) した後にのみ評価されます。2 番目の THRESHOLD 条件項目で 5 番目に評価されるのは、XYZ123I メッセージの 7 番目のオカレンスになります。

5. **即時メッセージの処理** オペレーター画面の即時メッセージ域に送信された NetView メッセージの THRESHOLD が自動化テーブルで評価される (つまり、TVBINXIT がオンになっている) 場合、THRESHOLD オカレンス・カウントは増分されず、条件の評価は偽になります。

TOKEN [(token-number [pos [len]])]

メッセージ内の特定のワードまたはフレーズを示します。NetView プログラムはワード間とフレーズ間にブランク・スペースを入れて、メッセージを複数トークンに分割します。トークンは、隣接していない 2 つのブランク・スペースの間にあるすべての文字で構成されます。比較項目は、解析テンプレートです。

token-number

比較するトークンの番号。これは、数値でなければなりません。デフォルトは 1 です。

pos 指定されたトークン内の比較が開始される位置を指定します。デフォルト値は 1 です。

len 比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 255 文字

タイプ:メッセージ

VALUE (variable [pos [len]])

比較でその値が使用される変数の名前を示します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

最大長: 255 文字

タイプ: 両方

VTAM [(pos [len])]

VTAM のバージョンとリリースを示します。VTAM は 4 文字の文字ストリングで、VTvr または Vvrm の形式を取ります。このうち *v* はバージョン番号を、*r* はリリース番号を表し、*m* は修正レベル番号を表します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

VTAM がアクティブでない場合、VTAM の値の評価はヌル (") になります。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 4 文字

タイプ: 両方

VTCOMPID [(pos [len])]

VTAM コンポーネント ID を示します。VTCOMPID は、14 文字の文字ストリングです。

ユーザーは、LISTVAR コマンドを使用して、該当するシステムでの VTAM コンポーネント ID を判別することができます。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

VTAM がアクティブでない場合、VTCOMPID の値の評価はヌル (") になります。このようなケースをテストするときは、ヌル (") キーワードと比較します。

最大長: 14 文字

タイプ: 両方

WEEKDAYN [(pos [len])]

1 から 7 文字の数値で、曜日を表します。

pos

比較が開始される位置を指定します。デフォルトは 1 です。

len

比較の対象となるストリングの長さを指定します。デフォルト値は、*pos* で始まるストリングの残りの部分です。

WEEKDAYN で得られる文字値は、以下のとおりです。

1 月曜日

2 火曜日
 3 水曜日
 4 木曜日
 5 金曜日
 6 土曜日
 7 日曜日
 最大長: 1 文字
 タイプ: 両方

ビット・ストリングの比較項目

IF-THEN ステートメントに使用できる比較項目には、ビット・ストリングと解析テンプレートがあります。

ビット・ストリングには、比較対象となる 1 つ以上のビット列、もしくはヌルを指定し、単一引用符で囲みます。このストリングは、1、0、および X の任意の組み合わせで構成します。

- 0 NetView プログラムに、B'0' の値のチェックを要求します。
- 1 NetView プログラムに、B'1' の値のチェックを要求します。
- X NetView プログラムにそのビットの値をチェックしないよう要求します。

例えば、0X1 のビット・ストリングでチェックする場合、011 でも 001 でも合致するビット・ストリングとなります。

図 33 の例で、メッセージ経路コードの位置 3 から始まるビットが 10011 である場合、もしくはメッセージ記述コードの位置 6 から始まるビットが 110 である場合に、ID が OPER1、OPER4、および OPER6 であるオペレーターにメッセージを経路指定するとともに、メッセージを表示する際に音響警報を鳴らすことを NetView プログラムに要求します。

```
IF ROUTCDE(3) = '10011' | DESC(6) = '110' THEN
  EXEC (ROUTE(ALL OPER1 OPER4 OPER6)) BEEP(Y);
```

図 33. ビットの比較の例

NetView プログラムは、比較項目のビット・ストリングと、条件項目から取られた長さの等しいビット・ストリングとの比較を、指定された位置 (デフォルトは、位置 1) から開始します。条件項目中で使用できるビット数が十分でない場合は、エラーになります。例えば、図 34 のステートメントでは、NetView プログラムが、10X10 を、ビット位置 14 から 18 まで比較するように指定していますが、記述コードには 16 ビットまでしか含むことができないので、このステートメントはエラーになります。

```
IF DESC(14) = '10X10' THEN ...
```

図 34. 長さの等しくないビットの比較の例

ヌルのビット・ストリング (") の働きは異なっていて、その機能は条件項目によります。開始位置から始まるすべてのビットがゼロの場合、DESC と ROUTCDE は、ヌルになります。図 35 では、DESC ビットのすべてがゼロならば比較は真になります。

```
IF DESC = '' THEN ...
```

図 35. ヌルのビット・ストリングの比較の例

MSUSEG と ATF の場合、ヌルのビット・ストリングはヌルの解析テンプレートと同様の意味をもちます。指定した位置が処理対象の MSU 内に存在しない場合、MSUSEG はヌルになります。呼び出した ATF が、長さゼロの比較項目を送り返す場合、ATF はヌルになります。長さがゼロの比較項目の正確な意味は、ATF によって異なります。値をまだ割り当てていないグローバル変数、またはヌルの値を割り当てたグローバル変数を要求した場合は、NetView プログラム提供の ATF である DSICGLOB および DSITGLOB により、ヌルの値が指定されます。

NetView プログラムで、複数コンソール・サポート (MCS) コンソールが使用されていない場合、メッセージが元はメッセージ・データ・ブロック (MDB) であった場合に限って値が含まれる条件項目は、MVS システム・メッセージとしてはヌルと評価されます。これらの条件項目のリストについては、175 ページの『条件項目』を参照してください。CNMPMDB または DSIMMDB アプリケーション・プログラミング・インターフェースから MDB が受け取られた場合に、これらの条件項目に値が含まれます。

解析テンプレートの比較項目

解析テンプレートには、リテラル、変数名、変数値、およびプレースホルダーを任意に組み合わせて使用することができます。また代わりに、ヌルの値を使用することもできます。

リテラル

リテラルは、指定のストリングとの比較を指定するものです。リテラルは、文字または 16 進数ストリングです。リテラルの最大長は 255 文字です。16 進リテラルの最大長は、255 桁の 16 進数です。

リテラルは、1 行に収まるようにしてください。単一のリテラルに複数行を使用する場合、継続行は字下げしないでください。各行は 72 桁目で終了し、継続行は 1 桁目から開始します。

リテラル比較項目を継続するには、複数行の小さなリテラル比較項目に分割します。連続するリテラル比較項目は、途中でブランクを入れずに連結されるため、各行を必ず 72 桁目で終了して 1 桁目から開始するようにする必要はありません。例えば、図 36 に示すように、リテラル比較項目を複数行にわたって継続できます。

```
IF TEXT='PURGE DATE IS LATER '  
    'THAN TODAY''S DATE'
```

図 36. 複数行を使用するリテラル比較項目の例

文字リテラル は、単一引用符で囲まれた英数字ストリングです。

図 37 の例では、DSII146I メッセージについて、6 番目のトークンの 5 文字目から文字リテラル AUTO と比較しています。

```
IF MSGID = 'DSII146I' & TOKEN (6 5) = 'AUTO' THEN  
  EXEC(ROUTE(ALL * OPER1));
```

図 37. 文字リテラルの比較の例

単一引用符が文字リテラルの一部である場合、図 38 に示すように、1 つ目の単一引用符のあとに 2 番目の単一引用符をコーディングする必要があります。

```
'PURGE DATE IS LATER THAN TODAY''S DATE'
```

図 38. 文字リテラルにおける単一引用符の使用例

図 39 に示すように、システム・シンボルを文字リテラルとして使用することができます。

```
IF DOMAINID = '&DOMAIN' & MSGID = 'DSII146I' THEN  
  EXEC(ROUTE(ALL * AUT01));
```

図 39. システム・シンボルを文字リテラルとして使用した例

16 進リテラル は、HEX() キーワード内の単一引用符で囲まれた 16 進数のストリングで、HEX('A1') のようになります。16 進リテラルは、単一引用符によって 16 進の変数と区別されます。

奇数桁数の 16 進数を指定すると、NetView プログラムによって先行ゼロが加えられます。例えば、NetView プログラムは、HEX('1AB') を HEX('01AB') と同一のものとして解釈します。

変数名

変数名は、(比較を行うときには) NetView プログラムに無視させるが、アクション処理のときに使用するために保管しておきたいメッセージまたは MSU の一部を指定します。保管された変数は CMD 付きの EXEC アクションで、コマンド・ストリング・パラメーターとして使用できます。

比較処理中、変数名で指定されたすべてのコンポーネントを NetView プログラムは無視しますが、無視された各コンポーネントは、ユーザーが指定した変数名に保管されます。IF-THEN ステートメントの IF 部分で設定した変数は、そのステートメントの THEN 部分もしくはその IF-THEN ステートメントの BEGIN-END セクションで使用できます。

変数名には、最大 16 文字の英数字を指定できますが、1 文字目に数値を指定することはできません。IF-THEN ステートメントにコーディング可能な変数名は最大 25 個で、自動化テーブル機能、アクション、もしくはキーワードにない任意の名前を使用できます。どの IF 条件においても、同一の変数名を 2 回以上使用することはできません。

IF-THEN-BEGIN 構造の IF 部分に定義された変数の値は、BEGIN-END セクションでその変数に保存されます。ただし、セクション内の個々の IF-THEN ステートメントで、その中でのみ使用するための変数値を一時的に再定義できます。その場合は同一の変数名との比較を行います。

変数名には、文字または 16 進数を指定します。 **文字変数名** は、値が文字の集合である変数名です。

図 40 の IF-THEN ステートメントには、DATEVAR という名前の文字変数が含まれています。

```
IF DOMAINID='CNM01' &  
  TEXT='DATABASE HASN'T BEEN PURGED SINCE' DATEVAR THEN  
  EXEC (CMD('CLISTA ' DATEVAR) ROUTE (ALL OPERA OPERB));
```

図 40. 文字変数名の使用例

NetView プログラムは、DATABASE HASN'T BEEN PURGED SINCE 12/3/10 というメッセージを受け取った場合、SINCE というワードに続くテキストの値 (12/3/10) を変数 DATEVAR に書き出します。その後、NetView プログラムは DATEVAR の値をパラメーターとして使って、OPERA と OPERB の両方のオペレーターの下で CLISTA を実行します。

図 41 の IF-THEN ステートメントには、DOMID という名前の変数が含まれています。

```
IF TEXT='PURGE DATE IS LATER THAN TODAY''S DATE' &  
  DOMAINID=DOMID THEN  
  EXEC (CMD('CLISTA ' DOMID) ROUTE (ALL OPERA OPERB));
```

図 41. 文字変数名 DOMID の使用例

NetView プログラムがドメイン CNM01 から PURGE DATE IS LATER THAN TODAY''S DATE というメッセージを受け取ったとすると、このステートメントは DOMAINID の値 (CNM01) を変数 DOMID に書き出し、OPERA と OPERB の両方のオペレーターの下で、変数 DOMID をパラメーターとして使用して CLISTA を実行することを意味します。

16 進変数名 は、割り当てられたデータの 16 進表記を値とする変数名です。16 進変数名は、HEX() キーワードで指定してください。

図 42 は、汎用警報データが MSU から変数 GENERICDATA に抽出される例です。この例では、16 進形式で POWEROUT コマンド・リストにデータが渡されます。16 進リテラルの HEX('14') とは異なり、16 進変数 HEX(GENERICDATA) には単一引用符がありません。

```
IF MSUSEG(0000.92 6) = HEX('14') .  
  & MSUSEG(0000.92) = HEX(GENERICDATA) THEN  
  EXEC (CMD('POWEROUT 'GENERICDATA) ROUTE (ONE AUTO1 *));
```

図 42. 16 進変数名の使用例

NetView プログラムは、GENERICDATA に割り当てられたデータを、16 進数字 (0 から 9 および A から F) で表記される EBCDIC 文字のストリングに展開してから、POWEROUT コマンド・リストに渡します。

変数値

ユーザーは、変数の値を VALUE() 関数を用いた解析テンプレートで使用できます。この場合、値はセットされるのではなく使用されます。値をもっていない変数

はすべて、NULL のリテラルとして扱われます。変数の指定は、BEGIN ブロックまたは IF-THEN ステートメントで行わなければなりません。同じ解析テンプレート内で、変数をセットし、同じテンプレート内でその後これを参照することはできません。変数は、位置または長さの添え字付きで使用することはできません。図 43 に示す IF-THEN ステートメントには、変数 DOM1 の値であるドメイン名を使用する解析テンプレートが含まれています。

```
IF DOMAINID=DOM1 &  
  TEXT = 'WORD1 ' VALUE(DOM1) ' WORD3' THEN  
  EXEC (CMD('CLISTA DOM1') ROUTE (ALL OPERA));
```

図 43. 変数 DOM1 の値を使用する例

プレースホルダー

プレースホルダーを指定すると、NetView プログラムは、比較対象外のメッセージまたは MSU の一部分をスキップします。プレースホルダーは変数名に似ていますが、プレースホルダーでスキップしたテキストはアクションに使用しないため、NetView プログラムには保管されません。プレースホルダーは、ピリオド (.) で指定します。.

ピリオドを単一引用符内にコーディングした場合、NetView プログラムはそのピリオドをプレースホルダーではなくストリングの一部として取り扱います。

図 44 の IF-THEN ステートメントには、NetView プログラムにメッセージ・テキストの複数のパーツをスキップさせるプレースホルダーを指定しています。

```
IF TEXT = . 'SENSE CODE=' SENSE . THEN  
  EXEC (CMD('CLIST1 'SENSE) ROUTE (ONE OPERA OPERB));
```

図 44. プレースホルダーの使用例

NetView プログラムが RESOURCE LU1 SENSE CODE=08 NOT ACTIVATED というメッセージを受け取ったとすると、図 44 のステートメントにより、SENSE CODE= に先行するすべてのテキストをスキップして変数名 SENSE に値 08 を格納し、変数名 SENSE に続くすべてのテキストをスキップするという指示が、NetView プログラムに出されます。先行プレースホルダーがなければ、この例のメッセージはステートメントの条件を充足しません。また、後続プレースホルダーがなければ、変数 SENSE は 08 NOT ACTIVATED という値になります。

図 45 の IF-THEN ステートメントは、プレースホルダーを使って、NetView プログラムにメッセージ・テキストから単一文字を選択させるものです。

```
IF MSGID = 'IST105I' &  
  TOKEN(2 4) = 'A' . THEN  
  NETLOG(N);
```

図 45. プレースホルダーを使って単一文字を選択する例

NetView プログラムが IST105I A01A425 NODE NOW INACTIVE というメッセージを受け取ったとすると、メッセージ ID とリソース名 (2 番目のトークン) の 4 文字目が比較されます。自動化テーブル・ステートメントで指定された値に一致する場合は、このメッセージはネットワーク・ログに現れません。

ヌル (null)

ヌル (") の解析テンプレートを使用すると、メッセージまたは MSU に情報が欠落していないかどうかをチェックできます。1 つの比較で、リテラル、変数名、あるいはプレースホルダーと一緒にヌルの値を使用することはできません。論理 AND (&) および論理 OR (|) を使用すると、ヌルの比較を他の比較と組み合わせられます。

処理を行うメッセージまたは MSU に解析テンプレート比較項目に対応する値が、1 つもない場合、解析テンプレート比較項目にはヌルの値が入ります。ヌルの値の正確な意味は、比較項目ごとに異なります。ビット・ストリング比較項目の値は、そのビット・フィールドにあるすべてのビットの値が 'B'0' である場合、ヌルになります。

条件項目によっては、メッセージが元はメッセージ・データ・ブロック (MDB) である場合に限って値が含まれるものがあります (175 ページの『条件項目』を参照)。このような条件項目は、NetView プログラムが、複数コンソール・サポート (MCS) コンソールを使用していない場合に、MVS システム・メッセージとしてヌルと評価されます。これらの条件項目は、MDB を CNMPMDB または DSIMMDB API から受け取った場合に値が含まれます。

テキスト比較項目は、指定された位置が比較項目の長さを超える場合、ヌルの値を返します。例えば、TOKEN(8) の場合、トークンが 6 個しかないメッセージについてはヌルになります。テキスト比較カテゴリにある比較項目には、下記のものなどがあります。

- DOMAINID
- MSGID
- TEXT
- TOKEN

ATF 呼び出しの結果、ATF が長さゼロの比較項目を送り返した場合、値はヌルになります。長さゼロの比較項目の固有の意味は、ATF ごとに異なります。値をまだ割り当てていないグローバル変数、またはヌルの値を割り当てたグローバル変数を要求した場合は、NetView プログラム提供の ATF である DSICGLOB および DSITGLOB により、ヌルの値が指定されます。

指定されたフィールドが処理対象の MSU に存在しない場合、MSUSEG の値はヌルになります。HIER は、MSU にリソース階層情報がない場合、もしくは MSU にない名前タイプの組み合わせが指定された場合、ヌルの値を返します。

一部の機能は、MVS、CNMPMDB API、DSIMMDB API のいずれかからメッセージが受信された場合にのみ設定され、これ以外の場合にはヌルの値を返します。例:

- JOBNAME
- SYSID
- ROUTCDE
- AREAID

SESSID は、処理対象のメッセージが TAF セッションを介して受信されなかった場合、ヌルの値を返します。

HDRMTYPE は、ヌルの値を戻しません。

ヌルの値との比較は、16 進ゼロのストリングとの比較と常に同じとは限りません。

図 46 の例は、ヌル (") キーワードを使用して MSU 内のレゾリュション主ベクトル (キー X'0002') の存在をチェックする方法を示しています。

```
IF MSUSEG(0002) ^= '' THEN
  BEGIN;
  :
  END;
```

図 46. ヌルの値を変数として使用する例

処置

このセクションでは、IF-THEN ステートメントと ALWAYS ステートメントで利用できるアクションについて説明します。表 10 は、これらのアクションを要約したものです。この表においてアクションはタイプ別 (メッセージ、MSU、もしくは両方) に並んでいますが、表のあとの説明はアルファベット順になっています。

表 10. IF-THEN および ALWAYS のアクション

アクション	説明
メッセージ	
DISPLAY	メッセージを表示します。
DOMACTION	オペレーター・メッセージを削除するためのアクションを指定します。
HCYLOG	メッセージをハードコピー・ログに記録します。
HOLD	メッセージを画面に保持します。
NETLOG	メッセージをネットワーク・ログに記録します。
SYSLOG	メッセージをシステム・ログに記録します。
MSU	
SRF	MSU に記録フィルター属性を設定します。
XLO	外部ログのみを指定します。
メッセージおよび MSU	
AUTOMATED	AIFR の有効なアクション・インディケーターを設定します。
BEEP	音響警報を鳴らすように指定します。
CNM493I	該当ステートメントについて、CNM493I メッセージをネットワーク・ログに書き出すかどうかを指定します。
COLOR	フォアグラウンド・カラーを設定します。
CONTINUE	メッセージまたは MSU のテーブル処理の継続を指定します。
EDIT	自動化されている AIFR を変更する編集指定を指定します。

EXEC	コマンドの発行、もしくはメッセージについての経路指定の制御を指定します。
HIGHINT	高輝度 3270 モードを設定します。
TRACE	トレースをオンに設定します。
XHILITE	フォアグラウンドの強調表示を設定します。

AUTOMATED (Yes|No|Ignore)

ステートメントに一致する任意のメッセージまたは MSU に対して有効なアクション・インディケーターの指定値を設定します。可能な値は、以下のとおりです。

YES|Y このステートメントに一致する AIFR の場合、このステートメントのアクションに関係なく強制的に AUTOMATED 状況を ON に設定し、この AIFR が自動化されていることを示します。これがデフォルトです。

NO|N このステートメントに一致する AIFR の場合、そのアクションに関係なく強制的に AUTOMATED 状況を OFF に設定し、この AIFR が自動化されていないことを示します。

IGNORE|I

インディケーターをこのステートメントの前の状態のままにします。

このインディケーターは、AUTOMATED 条件項目によって照会することができます。

タイプ: 両方

BEEP (Y|N)

メッセージまたは MSU が表示されるときに音響警報を鳴らすかどうかを示します。MSU の場合、BEEP は、ハードウェア・モニターによって表示されたアラート主ベクトルのみに適用されます。自動化テーブルその他で BEEP 値を指定しない場合、メッセージのデフォルトは BEEP(N) です。

MSU デフォルトの詳細については、5 (261 ページ) を参照してください。

タイプ: 両方

CBE ('edit_specification')

CBE アクションを使用すると、メッセージまたは MSU からの情報を使用して Common Base Event の XML バージョンを作成し、それを Common Event Infrastructure サーバーに送ることができます。この XML は、PIPE EDIT ステージの CBETEMP グローバル・オーダーによって提供される構文と機能を使用して作成されます。CBE アクションを使用すると、メッセージおよび MSU からのデータを事前定義の XML テンプレートに入力する値として使用できます。結果の XML は Common Event Infrastructure サーバーに送られます。元の AIFR は、その自動操作全体を通して継続します。 *edit_specification* に組み込む内容については、「*IBM Tivoli NetView for z/OS プログラミング:パイプ*」を参照してください。編集指定では、変数はサポートされません。

edit_specification は、('edit_specification' という形式で) 単一引用符で囲む必要がありますが、*edit_specification* 自体の中で単一引用符 (') を使用することはできません。

CNM493I (Y|N)

この自動化テーブル・ステートメントについて、CNM493I メッセージをネットワーク・ログに書き出すかどうかを指定します。CNM493I メッセージが生成されてログに書き出されると、自動化テーブルから実行されるコマンドまたはコマンド・リストの監査証跡になります。自動化テーブル、DEFAULTS コマンド、もしくは OVERRIDE コマンドに CNM493I の値を指定しない場合、デフォルトにより NetView プログラムで CNM493I メッセージが生成されます。

使用上の注意:

1. ステートメントが不安定で、(コマンドまたはコマンド・リストが自動化テーブルからいつ実行されたのかを示すために)、CNM493I メッセージが示すデバッグ機能が必要になる可能性がある場合に、そのステートメントに CNM493I(N) を指定するには、注意が必要です。特定のコマンドまたはコマンド・リストが自動化テーブルから何回実行されたかについては、自動化使用詳細報告書により提供される内容で十分である場合もあります。そのような場合は、次のようにして CNM493I メッセージが生成されないようにしてください。
 - 特定の自動化ステートメントに対し、CNM493I(N) ステートメントを使用します。
 - 特定の NetView プログラム・タスクに対し、OVERRIDE CNM493I=NO を使用します。
 - すべての NetView プログラムに対し、DEFAULTS CNM493I=NO を使用します。
2. CNM493I アクションを指定しても、メッセージまたは MSU に EXEC アクションを CMD キーワード付きで指定しないと、CNM493I アクションは無視され、処理が継続します。

タイプ: 両方

COLOR (BLU|GRE|PIN|RED|TUR|WHI|YEL)

カラー端末で表示するフォアグラウンド・カラーを指定します。MLWTO のすべての行が指定対象となりますが、自動化テーブルに使用できるのは、メッセージの 1 行目だけになります。ハードウェア・モニターは、指定された色を MSU の表示に使用します。コマンド機能は、指定された色をメッセージの表示に使用します。

ハードウェア・モニターの MSU に関するデフォルトについては、8 (261 ページ) を参照してください。

BLU 青を指定します。

GRE 緑を指定します。

PIN ピンクを指定します。

RED 赤を指定します。

TUR 空色を指定します。

WHI 白を指定します。

YEL 黄色を指定します。

タイプ: 両方

CONTINUE (Yes|No|Stop)

ステートメントと一致したメッセージおよび **MSU** が、現行または別のテーブルで別のステートメントに一致するものがあるかどうかを調べるために、自動化テーブル処理を継続する必要があるかどうかを指定します。可能な値は次のとおりです。

YES|Y 一致があった場合、現行の自動化テーブル、およびそれ以降のテーブル (あれば) の次のステートメントに対して、処理が続行されます。

NO|N 一致があった場合、次の自動化テーブル (あれば) の最初のステートメントに、処理が続行されます。これがデフォルトです。

STOP|S

一致があった場合、処理は終了します。現行テーブルまたはそれ以降のテーブル (あれば) のステートメントは、これ以上は評価されません。

タイプ: 両方

DISPLAY (Y|N)

表示機能のあるタスクにメッセージが渡された場合に、**NetView** プログラムにそのメッセージを表示させるかどうかを指定します。自動化テーブルその他で **DISPLAY** の値を指定しなかった場合、デフォルトは **DISPLAY(Y)** になります。

オペレーター画面の即時メッセージ域に送られる **NetView** プログラム・メッセージは、**DISPLAY** アクションの設定値に関係なく常に表示されます。

タイプ:メッセージ

DOMACTION (A|D|N)

NetView プログラムがこのメッセージに関して行う、オペレーター・メッセージの削除 (**DOM**) 処理のタイプを指定します。アクション・メッセージのデフォルト値は **DELMSG** で、その他のメッセージのデフォルト値は **NODELMSG** です。アクション・メッセージは、**WTOR** であるか、**CNMSTYLE** の **MVSPARM.ActionDescCodes** の設定に一致する記述子コードを持つメッセージです。 **DOMACTION** の指定により、以下のように、**DOM** 処理を調整することができます。

A|AUTOMATE

このメッセージについて **DOM** が予期されることを指定します。また、このメッセージについて **DOM** が受信されたときに、元のメッセージに変更を加えたコピーが、**DOM** として識別される変更済みの値とともに自動操作によって送信されるように要求します。この自動操作を実行するには、元のメッセージがオペレーター端末または自動タスクで保留される必要があります。 **DOM** の自動操作は、メッセージが保留される場所には関係なく実行されます。 **DOM** が受信されると、メッセージに対する自動操作は再び実行されます。 **IFRAUDOM** ビットはオンに設定され、その他の **DOM** 関連ビット

は DOM 要求からコピーされます。アクション・メッセージは、内部記憶域から除去され、オペレーター・コンソールから削除されます。

注: 注: IFRAUDOM をチェックするか (ATF で使用する場合)、自動化の値 ACTIONDL をチェックすると、元のメッセージを DOM コピーと区別できます。DOM の自動操作によって実行されるコマンドでは、MSGID を使用して、メッセージのインスタンスと DOM のインスタンスとを関連付けることができます。

DIDELMSG

このメッセージについて DOM を予期することを指定し、このメッセージについて DOM が受信されると、NetView プログラムが内部記憶域およびオペレーター・コンソールからアクション・メッセージを削除するが、自動操作によるメッセージのコピー送信は行わないことを指定します。DELMSG は、アクション・メッセージのデフォルト値です。

NIODELMSG

このメッセージについて DOM を予期しないことを指定します。NetView プログラムが、DOM 要求により今後削除された場合にその内部記録を一切保持されません。メッセージが非送信請求システム・メッセージである場合、NetView プログラムは、DOM が送信されても DOM を処理しません。アプリケーションによって発行されるメッセージの記述子コードが MVSPARM.ActionDescCodes CNMSTYLE ステートメントにリストされているのに、アプリケーションは DOM を発行しないという状態では、この設定は適切です。別の方法として、オペレーターまたは自動タスクが NetView DOM コマンドを使用して、メッセージを削除することもできます。NODELMSG は、アクション・メッセージ以外のメッセージのデフォルト値です。

タイプ:メッセージ

使用上の注意:

1. 自動操作実行後の DOMACTION の値が AUTOMATE または DELMSG である場合、NetView プログラムは、DOM を予期してリソースを割り振ります。こうしたメッセージが多数処理されて DOM が実施されない場合、ストレージと処理時間の両方が悪い影響を受けます。こうした懸念に対応するために、DOM コマンドを使用して、メッセージに関連付けられたレコードを除去できます。
2. アクション・メッセージは、デフォルトでは保留されます。アクション・メッセージ以外のメッセージを自動化すると、DOMACTION(AUTOMATE) または DOMACTION(DELMSG) アクションが、デフォルトで、メッセージを保留 に設定します。HOLD(YES) と HOLD(LOCAL) とを区別するために、HOLD アクションの追加を検討してください。DOM を自動化できるのは、元のメッセージが、何らかの NetView タスクの保留キュー上に留まっている場合に限りです。メ

メッセージの 1 次コピーの送信先となるタスクの **OVERRIDE** 設定を確認して、そのタスクでの **HOLD** の設定でメッセージを保留できるようにしてください。

EDIT ('*edit_specification*')

EDIT アクションでは、**AIFR** が自動化テーブル内にある間はそれに変更を行えるようになりました。これらの変更は、**PIPE EDIT** ステージで提供される構文と機能を使用して行われます。**EDIT** によって、メッセージおよび **MSU** を再フォーマットできます。変更された **AIFR** は、その自動操作全体を通して継続します。元の **AIFR** は使用できません。

edit_specification は、('edit_specification' という形式で) 単一引用符で囲む必要がありますが、*edit_specification* 自体の中で単一引用符 (' ') を使用することはできません。

編集指定では、**AIFR** データに対して行うアクションを定義します。

edit_specification に組み込める内容については、「*IBM Tivoli NetView for z/OS* プログラミング:パイプ」を参照してください。編集指定では、変数はサポートされません。

EXEC ([**CMD**(*cmdstring*)] [**ROUTE**(*routeparms*)])

処理すべきアクションを指定します。**EXEC** アクションには、**CMD** と **ROUTE** の一方または両方を付けて指定することができます。単一のステートメントに複数の **EXEC** アクションをコーディングすると、複数のコマンドを処理したり、複数のメッセージ・コピーを経路指定することができます。

CMD NetView プログラムに実行させたいコマンド、コマンド・リスト、もしくはコマンド・プロセッサを指定します。

cmdstring

2000 文字までのコマンド・ストリングを指定します。このストリングには、リテラル、変数、またはリテラルと変数の組み合わせを使用することができます。リテラルの場合は単一引用符で囲んでください。変数の場合は引用符で囲まないでください。

変数値には先行ブランクまたは末尾ブランクが含まれることがあり、それが原因でコマンドが失敗する場合がありますため、**STRIP** 関数を使用して、先行ブランクおよび末尾ブランクまたは 16 進ゼロを除去できます。**STRIP** 関数の構文は

STRIP(*varname*)

です。ここで、*varname* は変数名であり、先行ブランク、末尾ブランク、および 16 進ゼロを持たないその値がコマンドに挿入されます。

コマンドまたはコマンド・リストは、コマンド名やその他のパラメーターを含む完全なコマンド構文になるように指定してください。自動化テーブル変数をすでに定義している場合は、それをパラメーターに使用できます。235 ページの『解析テンプレートの比較項目』を参照してください。

NetView コマンド・ストリングでは、日本語の 2 バイト文字はサポートされません。

255 文字以上のコマンド・ストリングには、特殊な規則が適用されます。これらの規則についての説明は、9 (251 ページ) を参照してください。

アラートを Event Integration Facility (EIF) イベントまたはトラップに変換して、そのイベントまたはトラップを指定されたイベント・サーバーまたは SNMP マネージャーに転送するために自動化テーブルを使用している場合は、TECROUTE キーワードを接頭部として *cmdstring* の先頭に付けます。TECROUTE の接頭部の付いた 1 つのコマンドのみを、指定されたアラートに対して実行することができます。必要なすべてのコマンド・アクションを同じコマンドにコーディングしてください。

詳しくは、「457 ページの『イベント自動化サービス』」を参照してください。

ROUTE

cmdstring またはメッセージを、*routeparms* に指定されている ID のオペレーターに経路指定するように、NetView プログラムに指示します。

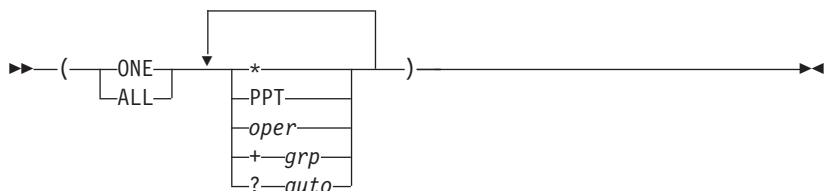
メッセージは、以下のいずれかの方式を使用して経路指定することもできます。

- 自動化テーブルから出すコマンド・リストで MSGROUTE コマンドを使用します。MSGROUTE コマンドは、例えば、メッセージの経路指定先を決める前に、複数行メッセージの 1 行目以外の行をチェックする必要がある場合などに使用します。MSGROUTE コマンドについては、「IBM Tivoli NetView for z/OS プログラミング: REXX および NetView コマンド・リスト 言語」を参照してください。
- EXCMD コマンドを使用します。EXCMD コマンドの詳細については、「IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)」に記載されています。
- PIPE ROUTE ステージ・コマンドを使用します。PIPE ROUTE ステージ・コマンドの詳細については、「IBM Tivoli NetView for z/OS プログラミング: パイプ」に記載されています。

routeparms

メッセージまたは *cmdstring* を処理するための経路指定先であるオペレーターまたはオペレーター・グループを指定します。パラメーターの構文は、次のようになります。

routeparms



項目の意味:

- ONE** リストのうち最初にログオンしたオペレーター、またはリストに載っているグループに割り当てられていてログオンしている最初のオペレーターに、メッセージまたは *cmdstring* を経路指定します。
- ALL** NetView プログラムにログオンしていてリストに載っているオペレーターまたはオペレーター・グループすべてに、メッセージまたは *cmdstring* を経路指定します。
- *** NetView プログラムがメッセージまたは *cmdstring* を、現在のオペレーター・タスク (そのメッセージを自動化テーブルに送信したタスク) に経路指定することを示します。現行タスクがメッセージまたは *cmdstring* を処理できない場合、NetView プログラムは、CNMSTYLE メンバーの FUNCTION.AUTOTASK.PRIMARY ステートメントで定義した 1 次自動タスクに経路指定されます。
- コマンド経路指定についての追加情報は、6 (249 ページ) のメモを参照してください。
- PPT** メッセージまたは *cmdstring* を、PPT に経路指定して処理するよう、NetView プログラムに指示します。
- oper* NetView プログラムが、メッセージまたは *cmdstring* の経路指定先とする、オペレーターの ID です。このオペレーター ID は、OPERATOR 定義ステートメントで NetView プログラムに定義しておく必要があります。オペレーター ID の最大長は、8 文字です。必要な個数をコーディングできます。
- +grp** NetView プログラムがメッセージまたは *cmdstring* の経路指定先とするオペレーター・グループの ID です。最大長は 8 文字で、1 文字目は正符号 (+) でなければなりません。グループ ID は必要な数だけコーディングできます。グループ ID は、ASSIGN コマンドで定義します。
- ASSIGN コマンドの詳細については、「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」を参照してください。
- ?auto* *function.autotask.function_name* で定義される自動タスクの *function_name* で、NetView プログラムによるメッセージまたは *cmdstring* の経路指定先となります。この自動タスクは NetView プログラムに対して定義しておく必要があります。この方法で、必要な数だけコーディングできます。この値は、自動化テーブルのコンパイル時に該当する自動タスク名に解決されます。
- function.autotask* コマンドについては、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

タイプ: 両方 (ただし、EXEC に ROUTE オプションのみが指定されている場合はメッセージ・タイプになります。)

使用上の注意:

1. ROUTE アクションの宛先タスクは、構文図に示されたものだけに限られます。オプションのタスクに経路指定はできません。
2. NetView プログラムは、一致するステートメント上の各 EXEC アクションを個々に処理しますが、ROUTE だけが指定されているアクションに限り、ALL オプションと組み合わせて処理します。重複するタスク ID を除去するため、NetView プログラムは ROUTE だけが指定されたすべてのアクションを ALL オプションと組み合わせます。
3. EXEC アクションで CMD を指定するときには、このコマンドおよびすべてのリテラル・パラメーターを単一引用符で囲んでください。コマンド・ストリングに必要な、スペースなどの区切り文字も、単一引用符で囲む必要があります。

パラメーターのような変数を使用する場合、変数名は単一引用符で囲まないでください。変数の定義は、IF-THEN ステートメントの 条件 部で行います。

ステートメントが BEGIN-END セクションで囲まれている場合は、そのセクションを開始する IF-THEN ステートメントで変数を定義することもできます。定義された変数値がヌルになっている場合は、ヌルの値をコマンド・リストに渡すことができます。しかし、定義を行っていない変数を渡すと、構文エラーになります。

図 47 では、CLISTB のあとにある単一引用符内のスペースは、コマンド名と VARPARM1 のパラメーター値を分けています。また、VARPARM1 と VARPARM2 の間にある単一引用符で囲まれたスペースによって、これら 2 つの変数名が分けられています。さらに、LITPARM の前にある単一引用符内のスペースにより、VARPARM2 とリテラル・パラメーターが区切られています。

```
EXEC (CMD('CLISTB ' VARPARM1 ' ' VARPARM2 ' LITPARM'))
```

図 47. EXEC で CMD を指定する例

VARPARM1 が OPER4 で、VARPARM2 が OPER5 とすると、コマンドは、CLISTB OPER4 OPER5 LITPARM になります。

4. EXEC アクションで CMD を指定した場合、ROUTE も指定してください。

CMD と ROUTE の両方を指定すると、NetView プログラムは *routeparms* に指定されたタスクで CMD アクションを処理します。多くの場合、理想的なコマンド宛先は自動タスクです。SYSOP もネットワーク・ログも、コマンドの経路指定先にすることはできません。BEEP、DISPLAY、HCYLOG、および HOLD のうちのいずれが指定されていても、入力メッセージ以外の CMD アクションには適用されません。

249 ページの図 48 では、NetView プログラムは、ログオンされている +GRP01 グループの 1 番目のタスクに、RUNNING コマンド・リストを送ります。+GRP01 のオペレーターが誰もログオンしていない場合、

コマンド・リストは代わりに AUTOMGR に送られます。いずれの場合も、NetView プログラムは、エントリーを起動するメッセージを表示しません。

```
IF MSGID='DSI530I' & TEXT(10) = TASKNAME ' ' : ' ' . THEN  
  EXEC (CMD('RUNNING ' TASKNAME) ROUTE(ONE +GRP01 AUTOMGR))  
  DISPLAY(N);
```

図 48. CMD キーワードと ROUTE キーワードの使用例

5. 非送信請求メッセージが ASSIGN PRI 条件に一致すると、そのメッセージは最初に自動タスクまたは ASSIGN コマンドで指定されたオペレーター端末に経路指定され、自動操作が継続します。指定されたタスクがアクティブでない場合、経路指定は行われません。これは以下のようなメッセージに影響します。

- MPF または MRT で自動操作が要求され、メッセージが一部の NetView タスクが所有する特定のコンソールに送信されない場合の、非送信請求 MVS メッセージ
- プログラム・オペレーター・インターフェース (POI) を介して受信した非送信請求 VTAM メッセージ
- メッセージの送信元により、「許可された受信先」と指定されているメッセージ
- PPT で実行しているコマンドのコマンド出力
- PPT で生成されるメッセージ
- PPT のキューに入れられたメッセージ

非送信請求 MVS メッセージを除き、メッセージが ASSIGN PRI 条件に一致しない場合、メッセージは許可された受信先に経路指定されます (許可された受信先がログオンしている場合)。それ以外の場合は PPT に経路指定されます。その後、自動化はメッセージが経路指定されたタスクで継続します。ASSIGN PRI 条件によって経路指定されない非送信請求 MVS メッセージは、CNMCSSIR タスクで自動化されます。

6. EXEC ステートメントに CMD を指定するときは、その EXEC に対する ROUTE アクションを必ずコーディングし、自動タスクまたはオペレーターの OST の実行時に自動操作が実行されることが確実でない限りは、ROUTE アクションに対してアスタリスク (*) を記述しないでください。ROUTE アクションにアスタリスクがある場合や、ROUTE アクションがコーディングされていない場合は、コマンドの経路に対して以下のルールが適用されます。

- 自動タスクまたはオペレーターの OST の実行時に自動操作が実行されると、コマンドはそのタスクに経路指定されます。
- 以下のメッセージ・タイプが、ASSIGN PRI の指定内容のいずれにも一致しない場合、これらのメッセージはオプションのタスクで自動化されます。
 - ハードウェア・モニターからの MSU。BNJDSERV タスクで自動化されています。
 - MVS プログラムからの非送信請求メッセージ。CNMCSSIR タスクで自動化されています。

- XCF 送信により生成される非送信請求メッセージ。DSIXCFMT タスクで自動化されています。
- その他のオプション・タスク。DSIPSS TYPE=OUTPUT を使用してタスクからメッセージが発行される場合です。

これらのすべてのタスクでは、明示的に ROUTE を指定していないコマンドは、CNMSTYLE メンバーの FUNCTION.AUTOTASK.PRIMARY ステートメントで定義した 1 次自動タスクのキューに入れられます。

- PPT では、明示的に ROUTE を指定していないコマンドは、CNMSTYLE メンバーの FUNCTION.AUTOTASK.PRIMARY ステートメントで定義した 1 次自動タスクのキューに入れられます。
7. 自動化テーブルでアクティブなタスクが見つからず、コマンドを実行できない場合、NetView プログラムは自動化テーブル処理で DWO032E メッセージをネットワーク・ログに送信してこの問題を報告します。

DWO032E は、自動化ステートメントに障害が起こって予定したとおりに機能できないことを示すメッセージであるため、自動化テーブルには、このメッセージを自動化するステートメントを指定しておく必要があります。DWO032E は常にネットワーク・ログに送られますが、自動化テーブルの NETLOG アクションが影響することはありません。このメッセージが発生するのは、コマンドの経路指定先タスクが 1 つもアクティブになっていない場合です。

デフォルトでは、メッセージ DWO032E はオペレーターに表示されませんが、このメッセージを自動化テーブル・ステートメントからオペレーターに経路指定しておく、自動化に問題があるときにオペレーターに示されるようになります。DWO032E メッセージを自動化テーブル・ステートメントから経路指定する場合、(ループ条件が発生しないように) 指定されたオペレーター・リストのうち、少なくとも 1 オペレーター (PPT など) がログオンされることを確認してください。

コマンドを実行できないという問題を回避するには、自動化テーブルからコマンドを発行するときに、ROUTE リストのどこかに、永続的な自動タスクを入れるようにしてください。

8. EXEC に CMD と ROUTE の両方を指定すると、CMD キーワードで指定されたコマンドの経路指定のみが行われます。

これは、自動化テーブル処理が行われるメッセージの経路指定には影響しません。メッセージそのものの経路指定を変更するには、EXEC アクションに ROUTE を指定し、CMD は指定しないでください。

EXEC アクションに ROUTE を指定して CMD を指定しなかった場合、NetView プログラムは、IF-THEN ステートメントに指定された BEEP、DISPLAY、HCYLOG、および HOLD の各アクションを実行し、メッセージを *routeparms* に指定されたオペレーターに経路指定します。それ以外の場合は、NetView プログラムは現行オペレーター・タ

スク（自動化テーブルにメッセージを送信したタスク）のもとで、BEEP、DISPLAY、HCYLOG、および HOLD の各アクションを処理します。

例えば、図 49 のようなステートメントは、メッセージをオペレーター OPER1 に表示します。

```
IF MSGID='DSI530I' & TEXT(10)= . ':' . THEN
  EXEC (ROUTE(ONE OPER1))
  DISPLAY(Y);
```

図 49. EXEC アクションで ROUTE キーワードを指定する例

9. 自動化テーブルから発行できるコマンドは、最長 2000 文字です。ただし、NetView プログラムのコマンド解析トークンの最大長は、255 文字です。

コマンド解析トークンは、コマンド内の解析区切り文字と解析区切り文字の間の文字すべてで構成されます。この解析区切り文字には、コンマ、等号、括弧、およびブランクがあります。図 50 の解析トークンは、以下のものです。

- MYCMD
- KEYWORD1
- VALUE1
- KEYWORD2
- VALUE1
- VALUE2

個々の解析トークンは、最長 255 文字です。

```
MYCMD KEYWORD1=VALUE1,KEYWORD2=(VALUE1,VALUE2)
```

図 50. 解析トークンの使用例

自動化テーブル・ステートメントでリテラル・ストリング内の 1 つの単一引用符を示すには、2 つの単一引用符を使います。解析トークンに 1 組の単一引用符がある場合、それらの単一引用符の間にある解析区切り文字はすべて無視されます。例えば、図 51 のようなコマンドが発行された場合、'LITERAL, STRING' が 1 つの解析トークンになり、コンマは解析区切り文字として使用されません。

```
IF MSGID='DSIxxxI'
  THEN EXEC(CMD('MYCMD KEYWORD1='LITERAL, STRING',
                 KEYWORD2=(VALUE1,VALUE2)'))
  ROUTE(ALL OPER1));
```

図 51. 解析区切り文字が無視される例

単一引用符の組み合わせが対応していない場合、余分な単一引用符からそのコマンドの終わりまでが全部解析トークンと見なされます。例えば、252 ページの図 52 のようなコマンドが発行された場合、'LITERAL, STRING, KEYWORD2=(VALUE1,VALUE2)' が 1 つの解析トークンになります。

```
IF MSGID='DSIxxxI'
  THEN EXEC(CMD('MYCMD KEYWORD1='LITERAL, STRING,
    KEYWORD2=(VALUE1,VALUE2)')
    ROUTE(ALL OPER1));
```

図 52. 組み合わせの対応しない解析トークンの例

10. REFRESH コマンドを使うと、NetView プログラムにオペレーターを事前定義しなくても、オペレーターの削除および追加を動的に行えます。

自動化ステートメントの ROUTE キーワードでターゲットにされたオペレーターが現在 NetView プログラムに対して定義されていない場合でも、自動化テーブルは正常に活動化されます。

AUTOTBL コマンドを発行して自動化テーブルの活動化のテストを行い、EXEC アクションの ROUTE キーワードで、NetView プログラムに対して定義されていないオペレーターを指定した場合、ROUTE キーワードで指定されたオペレーター ID が無効であることを示すメッセージが受信されます。そのあと、自動化テーブルは正常に活動化されます。

オペレーターが NetView プログラムに定義されている、いないに関係なく、ログオンしていないオペレーターに割り当てられたメッセージは、次に割り当てられているオペレーターか、または最初の宛先に送られます。

オペレーター定義が REFRESH コマンドで削除された場合、そのオペレーターのセッションは、そのオペレーターがログオフするまで続きます。メッセージの経路指定先のオペレーターが、ログオンしていても NetView プログラムにはもはや定義されていない場合も、そのメッセージはまだそのオペレーターに送信されます。

11. CMD パラメーターに指定されているコマンドに制御が渡されるとき、自動化されているメッセージが現行メッセージになります。このメッセージの本文および属性は SAFE * パイプ・ステージを使用して取得でき、メッセージの一部の属性は、WTO または WTOR コマンドが実行されるときにメッセージから継承されます。

HCYLOG(YIN)

ハードコピー・ログ・タスクがアクティブな場合に、メッセージをハード・コピー・ログに取るかどうかを判別します。自動化テーブルなどの場所に HCYLOG 値を指定しなかった場合のデフォルトは、HCYLOG(Y) です。

タイプ:メッセージ

HIGHINT(YIN)

高輝度をサポートする端末の場合に、高輝度 3270 設定を指定します。ハードウェア・モニターは、MSU を表示するときにこの設定を使用します。コマンド機能は、メッセージを表示するときにこの設定を使用します。

ハードウェア・モニターの MSU に関するデフォルトについては、8 (261 ページ) を参照してください。

タイプ: 両方

HOLD(YES|NO|LOCAL)

オペレーターの画面に表示されたメッセージを、NetView プログラムが保持するかどうかを判別します。HOLD パラメーターは、オペレーターがログオフしたときに、キューに入れられたアクション・メッセージを許可レシーバーに再経路指定するかどうかを判別します。

HOLD(YES)

メッセージは NetView 画面に保留され、オペレーターと自動タスクの両方のキューに保持されます。

メッセージがアクション・メッセージである場合は、ログオフ時に再経路指定されます。

HOLD(LOCAL)

メッセージは HOLD(YES) で保留されます。キューに入れられたアクション・メッセージは、ログオフ時に再経路指定されません。

HOLD(NO)

メッセージが画面に保留されないようにします。

HOLD(NO) とマークされたアクション・メッセージは、メッセージが自動タスクに送られる場合を除き、あとの処理のためにキューに入れられることはありません。メッセージが自動タスクに送られたアクション・メッセージである場合、そのメッセージはキューに入れられ、オペレーターがログオフすると許可された受信先に再経路指定されます。

注: HOLD(NO) とマークされた WTOR などのアクション・メッセージは、応答などの後続の DOM では処理されません。したがって、強調表示は変更されませんが、メッセージは、スクロールされて画面からは消えます。

注: すべての HOLD 値は 1 文字に省略できます。例えば、HOLD(LOCAL) と HOLD(L) は同義語になります。

タイプ:メッセージ

NETLOG(Y|N [*indicator-number*] [*] [*oper*[,...]] [+*grp*[,...]])

NetView プログラムがメッセージをネットワーク・ログに入れるかどうか、および指定されたオペレーターまたはオペレーター・グループを示す状況モニター重要メッセージ・インディケーターを、メッセージが活動化するかどうかを判別します。自動化テーブルなどの場所に NETLOG 値を指定しなかった場合のデフォルトは、NETLOG(Y) になります。

indicator-number

状況モニター・メッセージ・インディケーターを示します。

- * NetView プログラムがメッセージまたは *cmdstring* を、現行のタスク (そのメッセージを自動化テーブルに送信したタスク) に経路指定することを示します。現在のタスクが CNMCSSIR である場合、メッセージ経路指定が異なることがあります。

oper [...]

オペレーターの ID を指定します。メッセージはこのオペレーターにとって重要なものとしてログに記録されます。このオペレーター

ID は、OPERATOR 定義ステートメントで NetView プログラムに定義しておく必要があります。オペレーター ID の最大長は、8 文字です。必要な個数をコーディングできます。

+grp [...]

オペレーター・グループのグループ ID を指定します。このグループにとってそのメッセージが重要なものとしてログに記録されます。最大長は 8 文字で、1 文字目は正符号 (+) でなければなりません。グループ ID の定義には、ASSIGN コマンドが使用されます。ASSIGN コマンドについての詳細は、「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」を参照してください。

タイプ:メッセージ

使用上の注意:

1. 重要メッセージ・インディケーターの定義には、F (FILTER) ステートメントを使用することもできます。F (FILTER) ステートメントについては、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。
2. NETLOG で、インディケーター番号だけが指定された場合、許可レシーバーにとって重要なものとしてメッセージがログに記録されます。インディケーター番号、およびオペレーターかオペレーター・グループのリストが指定された場合、そのオペレーターかオペレーター・グループにとって重要なものとしてメッセージがログに記録されます。

これは、コーディング時に NETLOG のインディケーター番号のみを指定した IF-THEN ステートメントです。

```
IF MSGID='IST105I' THEN  
    NETLOG(Y 2);
```

メッセージ IST105I は、状況モニター重要メッセージ・インディケーター番号に 2 を持つ重要メッセージとして定義されています。オペレーターもオペレーター・グループも指定されなかったため、メッセージ IST105I は、NetView プログラムに検出されると、許可レシーバーにとって重要なものとしてログに記録されます。

これは、インディケーター番号、およびオペレーターとオペレーター・グループのリストが NETLOG に指定されたときに、IST105I メッセージがどのようにログに記録されるかを示します。IST105I メッセージは、状況モニター重要メッセージ・インディケーター番号に 2 を持つ重要メッセージとして定義されています。

```
IF MSGID='IST105I' THEN  
    NETLOG(Y 2 * OPER1 +GRP5 OPER6);
```

NetView プログラムは IST105I メッセージを重要なものとしてログに記録し、OPER1、OPER6、+GRP5 グループに割り当てられた全オペレーター、および現行オペレーターに対して、定義されている強調表示を行います。指定対象のオペレーターが、指定対象のグループにも属している場合、強調表示とログは重複して起こらないようになっています。オ

オペレーターの OPER1 と OPER6 がグループ +GRP5 に属している場合でも、それぞれのオペレーターは、メッセージ CNM039I を一度だけ受信します。このメッセージは、オペレーターが状況モニターを表示したりログをブラウズしたりしていない場合に表示されます。

3. 現在 NetView プログラムに定義されていないオペレーターには、自動化テーブルの状況モニター重要メッセージ・インディケーターを割り当てることができます。AUTOTBL コマンドを使用すると、自動化テーブルは正常に活動化します。オペレーターを動的に追加した場合、そのオペレーターはログオンしたあとに NetView 自動操作によって処理されたメッセージのインディケーターを表示できますが、そのオペレーターがログオンする前に処理されたメッセージのインディケーターは表示できません。メッセージ・インディケーターは、オペレーターがログオンしたときに設定されます。

オペレーター定義を動的に削除したにもかかわらず、そのオペレーターがログオンしたままの場合は、状況モニター重要メッセージ・インディケーターをそのオペレーターにまだ割り当てることができます。

SRF (*filterlevel* [*setting*] [*filterscope*])

MSU に対して記録フィルターが設定され、ハードウェア・モニター・データベースの中にあるデータの記録が制御されることを示します。

自動化テーブルなどにある MSU に対してフィルター設定を指定しなかった場合、ESREC のデフォルトは PASS になります。AREC に対するデフォルトは、MSU によって異なります。詳しくは、「*IBM Tivoli NetView for z/OS Command Reference Volume 1 (A-N)*」の SRFILTER コマンドを参照してください。AREC を渡す MSU の場合は、ROUTE のデフォルトは PASS、TECROUTE のデフォルトは BLOCK、TRAPROUT のデフォルトは BLOCK、OPER のデフォルトは BLOCK です。

filterlevel

MSU に設定したい記録フィルターを示します。

ESREC

ハードウェア・モニターが MSU をイベントとして記録するかどうかを判別します。

AREC ハードウェア・モニターがイベントをアラートとして記録するかどうかを判別します。

OPER ハードウェア・モニターがメッセージの BNJ146I と BNJ030I をアラートから生成し、それらを許可レシーバーに送信するかどうかを判別します。

ROUTE

ハードウェア・モニターが、NetView のアラート用フォーカル・ポイントとして機能するシステムに、アラートを経路指定するかどうかを判別します。

TECROUTE

ハードウェア・モニターがアラートを EIF イベントに変換し、そのイベントを指定されたイベント・サーバーに経路指定するかどうかを判別します。

TRAPROUT

ハードウェア・モニターがアラートをトラップに変換し、そのトラップを SNMP マネージャーに経路指定するかどうかを判別します。

- setting* 条件に一致する MSU が、以下のいずれかのイベントからブロックされる (あるいは、いずれかに渡される) かどうかを指定します。
- ハードウェア・モニター・データベース
 - NetView オペレーター (許可された受信先へのメッセージとして)
 - ハードウェア・モニター・フォーカル・ポイント

BLOCK

データは、ブロックされます。これはデフォルトです。

PASS データが渡されます。

filterscope

ハードウェア・モニターが 2 次イベントを記録する場合、フィルター設定値が 1 次イベントと 2 次イベントのどちらに適用されるかを指定します。

PRI フィルター設定値が 1 次イベントに適用されることを示します。

SEC 2 次イベントが存在すれば、フィルター設定値は 2 次イベントに適用されることを示します。

BOTH フィルター設定値が、1 次イベントと 2 次イベントの両方に適用されることを示します。これはデフォルトです。

タイプ: MSU

使用上の注意:

1. ハードウェア・モニター・フィルターが、ハードウェア・モニター SRFILTER (SRF) コマンドで BLOCK に設定されていても、自動化テーブルからの MSU の受信がブロックされることはありません。自動化テーブルの SRF アクションで、SRFILTER コマンドが確立したフィルター設定をオーバーライドできます。

色および強調表示のオプションを SRFILTER コマンドのように設定することはできません。その代わりに BEEP、COLOR、HIGHINT、および XHILITE アクションを使って、自動化テーブルから色と強調表示のオプションを設定してください。

SRFILTER および SRF の詳細については、342 ページの『アラートのフィルター操作』を参照してください。

2. 通常の MSU は *filterscope* のデフォルトとして BOTH を指定すれば、十分です。2 次イベントが記録されるのは、影響されたリソースがその障害を起こしたリソースとは異なることをハードウェア・モニターが判別し、そのために、NetView プログラムが 1 つの問題レコードから 2 つのイベントを作成する場合で、めったに起こりません。2 つのイベントは類似していますが、リソース名が異なり、2 次イベントより 1 次イベントのリソース階層の方が短いものになります。

ハードウェア・モニター SRFILTER コマンドが、それぞれのイベントに別々に影響を与える場合があります。NetView プログラムは 1 次イベントのみを使用して、自動化テーブルの中で一致するものを探しますが、テーブルで指定したフィルター操作オプションは、いずれのイベントにも両方のイベントにも適用することができます。

デフォルトでは、フィルター操作オプションは両方のイベントに適用されます。自動化テーブルで 1 次イベントと 2 次イベントを別々にフィルター操作したい場合、*filterscope* を指定します。

3. SNA-MDS/LU 6.2 アラート転送プロトコルを介して NetView または非 NetView のリモート・ノード・エントリー・ポイントから転送されるアラートには、ESREC または AREC フィルター設定値 (BLOCK または PASS) を設定することもできます。

例えば、SRF アクションが LU 6.2 以外によって転送されたアラート用に ESREC フィルター・レベルを BLOCK に、および AREC フィルター・レベルを PASS に設定するのに使用された場合、ハードウェア・モニターは ESREC/BLOCK および AREC/PASS を不適切な値と見なし、AREC を BLOCK にリセットします。したがって、ESREC および AREC の両方が BLOCK に設定され、データベースに記録されるデータはありません。

しかし、LU 6.2 によって転送されたアラートでは、SRF アクションが ESREC を BLOCK に、および AREC を PASS にするという両方の設定に使用された場合、ハードウェア・モニターはこれらの設定値を受け入れ、アラート・レコードのみがデータベースに記録されます。これはアラートのみ の記録方法で、以下の例で示します。

```
*=====*
* Was the MSU forwarded over LU 6.2, and if so                *
* then record it in the hardware monitor database as alert-only by *
* BLOCKing ESREC and PASSing AREC.                             *
*=====*
IF HMFWDNA = '1' THEN
    SRF(ESREC BLOCK)
    SRF(AREC PASS);
```

423 ページの『第 26 章 集中操作』で説明するように、LU 6.2 を介して NetView エントリー・ポイントから受信されたデフォルト・アラートは、「アラートのみ」としてデータベースに記録されますが、LU 6.2 を介して非 NetView エントリー・ポイントから受信されたアラートは、通常の ESREC および AREC フィルターの処理対象になります。

これらのフィルターをどのようにパスしたかに従って、データはデータベースに記録されます。SRF アクションによってこれらのデフォルトをオーバーライドできます。例えば、非 NetView アラートを「アラートのみ」として記録するために、AREC フィルター・レベルを PASS におよび ESREC フィルター・レベルを BLOCK に設定したり、あるいはエントリー・ポイント NetView システムから転送されたアラートのイベントまたは統計データを記録するために、ESREC フィルター・レベルを PASS に設定できます。

SYSLOG (Y|N)

NetView プログラムがメッセージを MVS システム・ログに送信するかどうかを判別します。自動化テーブルなどに SYSLOG 値を指定していない場合、デフォルトは SYSLOG(N) になります。サブシステム・インターフェースから受信されたメッセージは、SYSLOG パラメーターの影響を受けず、NetView プログラムに送られる前に、無条件に MVS システム・ログに記録されます。

タイプ:メッセージ

TRACE ('tracetag')

メッセージまたは MSU のトレース・インディケーターとタグを設定し、自動化テーブル処理をトレースできるようにします。tracetag タグは、'tracetag' のように引用符で囲む必要があり、16 文字以内でなければならず、さらにブランクを含んではなりません。CONTINUE(Y) アクションが、TRACE アクションを使用する場合に暗黙の指定になります。自動化テーブル全体を通しての AIFR のトレースに関する詳細は、545 ページの『NetView 自動化テーブル・トレースの使用』を参照してください。

タイプ:両方

XHILITE (BLI|REV|UND|NONE)

フォアグラウンドの拡張強調表示を指定します。MSU のデフォルトの詳細については、8 (261 ページ) の注を参照してください。

BLI 明滅を指定します。

REV 反転強調表示を指定します。

UND 下線を指定します。

NONE 拡張強調表示を行わないことを指定します。

タイプ:両方

XLO (Y|N)

外部ログのみを指定します。XLO を N に設定すると、ハードウェア・モニターおよび自動化テーブルに設定された記録フィルタが有効になります。XLO を Y に設定すると、外部ログのみが起り、NetView プログラムは記録フィルタの設定値を無視します。自動化テーブル、BNJDSERV の XITCI インストール・システム出口、もしくはインストール・システム出口 DSIEX16B に XLO 値を指定していない場合、デフォルト値は、N になります (記録フィルタは有効になります)。

タイプ:MSU

使用上の注意:

1. 1 つの自動化ステートメントで、BEGIN とアクションを同時には指定できません。あるアクションを、1 つのメッセージまたは MSU に複数回指定する規則は、そのアクションによって異なります。

EXEC アクションの場合は、1 つのメッセージまたは MSU に対して使用する回数に制限はなく、それぞれの EXEC アクションが行えます。例えば、259 ページの図 53 のステートメントは、着信した DSI530I メッセージをグループ +GRP01 に属するタスクに経路指定するとともに、自動タスク AUTOMGR の

もとで RUNNING コマンド・リストを実行します。

```
IF MSGID='DSI530I' & TEXT(10)= TASKNAME ' ' : ' ' . THEN
  EXEC (ROUTE(ALL +GRP01))
  EXEC (CMD('RUNNING ' TASKNAME) ROUTE(ONE AUTOMGR))
  DISPLAY(Y);
```

図 53. 1 つのメッセージまたは MSU に対して複数の EXEC を実行する例

NetView プログラムは、一致するステートメントそれぞれの EXEC アクションを個別に処理します。ただし、ROUTE キーワードに ALL オプションを指定した (CMD キーワードは指定しない) EXEC アクションは例外です。1 つのメッセージに複数の EXEC(ROUTE(ALL *parm1 parm2 parmx*)) アクションを指定した場合、NetView プログラムはタスク・リストを組み合わせ、メッセージがどのタスクにも 2 度以上経路指定されないようにします。

また、SRF アクションも、1 つの MSU に複数回使用できます。フィルターに行った設定が矛盾する場合、CONTINUE アクションの結果処理されるのが 1 つのステートメント内の設定であるか、複数ステートメントの個々の設定であるかにかかわらず、NetView プログラムは最後に指定された設定を使用します。

単一の自動化ステートメントに複数のアクション (EXEC または SRF 以外) を指定する場合は、アクションの最初のカレンス優先されます。例えば、図 54 のステートメントでは、HOLD アクションの最初のカレンスは Y で、COLOR アクションの最初のカレンスは RED です。したがって、オペレーター OPER1、OPER2、OPER3、および OPER4 は、赤色で保留されたメッセージ IEE136I を受け取ります。

```
IF MSGID = 'IEE136I' THEN
  EXEC (ROUTE(ALL OPER1 OPER2)) HOLD(Y)
  EXEC (ROUTE(ALL OPER3 OPER4)) HOLD(N) COLOR(RED);
```

図 54. 複数のアクションを指定する例

CONTINUE(Y) アクションを、別々のステートメントで任意の回数使用することで、条件に一致したあとも自動化テーブル処理を継続できます。メッセージまたは MSU は、CONTINUE(Y) の指定されていないステートメントの条件に一致するまで、処理を続けます。

EXEC、SRF、および CONTINUE 以外のアクションで、CONTINUE アクションを使用する結果、1 つのメッセージまたは MSU に対して矛盾するアクションが適用される場合、NetView プログラムは最後に指定された値を使用します。例えば、自動化テーブルで BEEP(N) アクションのあとに BEEP(Y) アクションが指定されている場合、BEEP(Y) アクションによって BEEP(N) がオーバーライドされます。しかし、重要メッセージ・インディケーターを持つ NETLOG(Y) アクションのあとに、重要メッセージ・インディケーターのない NETLOG(Y) メッセージが指定されていて、その指定の間に NETLOG(N) が指定されていなければ、260 ページの図 55 に示しているように、最初の NETLOG(Y) からのインディケーターが有効になります。

```

*** First statement for IEE136I ***
IF MSGID = 'IEE136I' THEN
    EXEC (ROUTE(ALL OPER1 OPER2)) HOLD(Y) NETLOG(Y)
    EXEC (ROUTE(ALL OPER3 OPER4)) COLOR(RED)
    CONTINUE(Y)

*** Second statement for IEE136I ***
IF MSGID = 'IEE136I' THEN
    EXEC (ROUTE(ALL OPER5 OPER6))
    EXEC (ROUTE(ALL OPER7 OPER8)) COLOR(BLU) NETLOG(N);

```

図 55. CONTINUE を使用したメッセージの矛盾するアクションの例

メッセージ IEE136I が出されると、最初のステートメントで一致が起きます。CONTINUE(Y) がコーディングされているため、追加の一致を求めて自動化テーブルが検索されます。2 番目のステートメントが検出されると、CONTINUE(Y) がコーディングされているため、最初のステートメントに設定されているアクションがオプションで変更できます。HOLD アクションは最初のステートメントで確立され、2 番目のステートメントでは変更されません。COLOR アクションは最初のステートメントで設定されますが、2 番目のステートメントで変更されます。最後に、2 番目のステートメントによって NETLOG アクションが判別されます。その結果、OPER1 から OPER8 はそれぞれ保留されたメッセージ IEE136I を青色で表示します。メッセージは NETLOG には送られません。

2. Yes または No (YIN) とコーディングするアクションは、Yes の場合は YES か Y を、No の場合は NO か N をコーディングできます。
3. メッセージ・アクションのデフォルトに指定されている値は、NetView システム・デフォルトです。以下のリストは、デフォルトをオーバーライドするかどうかを決めるときに使用してください。リスト内の各項目で、それより前にリストされている項目をオーバーライドできます。
 - a. システム・デフォルト
 - b. DEFAULTS コマンド
 - c. インストール・システム出口 DSIEX02A (ただしこれを DEFAULTS の設定値の代わりに使用する場合)
 - d. 自動化テーブル内で一致するステートメントのアクション
 - e. インストール・システム出口 DSIEX16 (ただしこれを DEFAULTS の設定値の代わりに使用する場合)
 - f. OVERRIDE コマンド。メッセージ・カラーおよび強調表示についての SCRNFMT の指定は、自動化テーブルの指定を無効にしません。
 - g. インストール・システム出口の DSIEX02A または DSIEX16 (ただし、OVERRIDE の設定値の代わりに使用する場合)
4. OVERRIDE コマンドで CNM493I、DISPLAY、HCYLOG、NETLOG、または SYSLOG に YES か NO の値が設定されている場合、自動化テーブルのアクションに指定された設定値 (存在する場合) はオーバーライドされます。OVERRIDE コマンドか DEFAULTS コマンドの BEEP または HOLD に対する設定値 DISABLE は、NetView プログラムが自動化テーブルに指定された設定値を使用しないことを意味します。DEFAULTS コマンドと OVERRIDE コマンドの BEEP キーワードは、メッセージ処理にのみ影響します。

DEFAULTS コマンドと OVERRIDE コマンドについての詳細は、NetView オンライン・ヘルプを参照してください。

5. MSU の場合、フィルター操作および強調表示のアクションが適用されるのは、ハードウェア・モニターの処理で受信されたアラート主ベクトルに限られています。フィルター操作と強調表示のアクションには、SRF、XLO、COLOR、HIGHINT、および XHILITE が含まれます。
6. 以下のリストは、フィルター操作と強調表示のオプションをオーバーライドするかどうかを決めるときに使用します。リスト内の各項目で、それより前にリストされている項目をオーバーライドできます。
 - a. SRFILTER (SRF) コマンドで指定されたハードウェア・モニター・フィルターの設定 (XLO 以外の場合)
 - b. BNJDSERV のインストール・システム出口 XITCI からの戻りコード (XLO の場合)
 - c. 自動化テーブル内で一致するステートメントのアクション
 - d. インストール・システム出口 DSIEX16B

DSIEX16B 処理のあとの最終的な XLO 値が YES の場合、MSU は外部ログにのみ渡されます。それ以外で、最終的な ESREC 値が BLOCK の場合、NetView プログラムは AREC、ROUTE、および OPER の各フィルターを無視します。ハードウェア・モニターは、MSU をイベントとしてもアラートとしても記録しません。ESREC 値が PASS であっても AREC 値が BLOCK の場合、NetView プログラムは ROUTE フィルターと OPER フィルターを無視します。

ハードウェア・モニターでフィルターを BLOCK に設定しても、XITCI で XLO を YES に設定しても、MSU を自動化テーブルに渡されないようにすることはできません。このようにしても自動化テーブルは MSU を処理し、さらには MSU の前の XLO 設定とその他のフィルター設定をオーバーライドすることがあります。

7. MSU の場合、CNM493I メッセージ生成がオプションをオーバーライドするかどうかを決めるときに、以下のリストを使用してください。リスト内の各項目で、それより前にリストされている項目をオーバーライドできます。
 - ・ システム・デフォルト (CNM493I メッセージの生成)
 - ・ DEFAULTS コマンド
 - ・ 自動化テーブル内で一致するステートメントのアクション
 - ・ OST によって自動操作に送信された MSU に対応する OVERRIDE コマンド (DSIAUTO など)。OVERRIDE コマンドは、BNJDSERV などの DST では使用できません。これは、MSU を自動化テーブルに送信してハードウェア・モニターから処理させるための NetView タスクです。
8. アラート主ベクトルが 5 にリストされたすべてのステップを検査してもカラー・オプションや強調表示オプションが割り当てられない場合、ハードウェア・モニターはカラー・マップおよび SRFILTER COLOR DEFAULTS 設定値を使用してアラートの表示を制御します。

初期値を使った場合、アラート動的パネルでの最初の表示時には、アラートの表示は白または高輝度になりますが、それ以外のときには、空色または低輝度

になります。BEEP の設定値は YES、XHILITE の設定値は NONE です。(カラー・マップについては、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照し、SRFILTER COLOR DEFAULTS 設定については、「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照してください。)

アラート主ベクトルに色および強調表示のオプションがいずれか 1 つ割り当てられた場合、SRFILTER COLOR DEFAULTS は適用されませんが、その代わりにアラートは、指定されなかったオプションとして次のデフォルト設定値を受け取ります。

BEEP YES

COLOR

TUR (空色)

HIGHINT

NO

XHILITE

NONE

自動化テーブルなどに値を指定しなかった場合 (5 (261 ページ) を参照)、以下のデフォルト・フィルター設定値が割り当てられます。

- XLO のデフォルトは NO
 - ハードウェア・モニターのデフォルトの SRFILTER 設定は、他のフィルター設定を制御します。
 - ESREC のデフォルトは PASS
 - AREC のデフォルトはアラート・タイプによって異なります (「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」を参照)。
 - アラートが AREC を通過する場合、ROUTE のデフォルトは PASS
 - OPER のデフォルトは BLOCK
9. NetView プログラムは、自動化テーブルに DISPLAY(Y) をコーディングしたときのみ、メッセージの HOLD アクションと BEEP アクションを処理します。HOLD アクションと BEEP アクションは、MVS コンソールに経路指定されたメッセージには適用されません。HOLD、DISPLAY、および BEEP のみが、メッセージが OST-NNT セッションを介して複数ドメインにわたって転送されたときに自動化テーブルに保存されます。
10. 使用しているシステム上の他の NetView プログラムで SYSOP に割り当てられていたメッセージは、自動化しないでください。そのようなメッセージを自動化してしまうと、両方の NetView プログラムにループが発生する可能性があります。

1 つのシステムに属する 2 つの NetView プログラム間の通信に有効な方式については、513 ページの『第 32 章 単一システムでの複数 NetView プログラムの実行』を参照してください。

ALWAYS ステートメント

ALWAYS ステートメントは、テーブル内のその点に達するすべてのメッセージおよび MSU を NetView プログラムに処理させるための、アクションまたは一連のステートメントを指定するために使用します。

ALWAYS ステートメントを自動化テーブルまたは BEGIN-END セクションの先頭で CONTINUE とともに使用すると、指定された値はそのステートメントまたはテーブルでのデフォルトになります。

また、自動化テーブルまたは BEGIN-END セクションの終わりで、そのテーブルまたはセクションの他のステートメントに一致しなかったメッセージおよび MSU を処理するためにも ALWAYS ステートメントを指定することもできます。

ALWAYS ステートメントの構文は、以下のとおりです。

ALWAYS ステートメント

```
▶▶—ALWAYS—┐actions┐;────────────────────────────────────────▶▶  
               └BEGIN┘
```

項目の意味:

ALWAYS

ALWAYS ステートメントの先頭にコーディングするキーワードです。

actions NetView プログラムが行うアクションを指定します。アクションの詳細については、240 ページの『処置』を参照してください。

BEGIN

BEGIN-END セクションの始まりを示します。詳しくは『168 ページの『BEGIN-END セクション』』を参照してください。

使用上の注意:

1. ALWAYS ステートメントは、他のステートメントと同様、メッセージ・タイプ、MSU タイプ、もしくは両方のタイプを指定できます。

メッセージ・タイプのアクションを使うと、ALWAYS ステートメントはメッセージ・タイプのステートメントになり、そのステートメントは MSU に影響しなくなります。また、MSU タイプのアクションを使うと、ALWAYS ステートメントは MSU タイプのステートメントになり、そのステートメントはメッセージに影響しなくなります。

ステートメント・タイプの詳細については、165 ページの『自動化テーブル・ステートメントのタイプ』を参照してください。

2. ALWAYS ステートメントで指定されたステートメントが処理されるのは、自動化テーブルの論理処理および順次処理がすべて終わって、その ALWAYS ステートメントに制御権が渡された場合に限られます。条件に一致したために、自動化テーブル処理が停止した場合、その一致点以後にあるすべての ALWAYS ステートメントは処理されなくなります。

3. コマンドまたはコマンド・リストを出す ALWAYS ステートメントを使用する場合は、注意が必要です。

通常このようなステートメントが処理されるのは、BEGIN-END セクションの中に限られています。こうしたステートメントの使用が不適切であると、大多数のメッセージおよび MSU に影響を及ぼす可能性があります。

%INCLUDE ステートメント

%INCLUDE ステートメントは、自動化テーブルの一部を別のファイルか別のメンバーに保管しておくために使用します。

%INCLUDE ステートメントの構文は、以下のとおりです。

%INCLUDE ステートメント

▶▶ —%—INCLUDE— —*membername*— —————▶▶
 | |
 & *varname*

項目の意味:

%INCLUDE

各 %INCLUDE ステートメントの先頭にコーディングするキーワードです。

membername

組み込まれるメンバーの名前です。このメンバーは DSIPARM データ・セットのメンバーでなければなりません。

&varname

既存のローカル変数またはグローバル変数の名前で、アンパーサンド (&) 文字のあとに指定します。

使用上の注意:

1. %INCLUDE ステートメントは、1 行を超えて指定することはできません。
2. 他の自動化テーブル・ステートメントとは異なり、%INCLUDE ステートメントの終わりにはセミコロン (;) を付けません。
3. 組み込まれたメンバーには、%INCLUDE ステートメントやその他の自動化テーブル・ステートメントが指定されていても構いません。
4. 組み込まれたメンバーには、直接的にも間接的にもそのメンバーそのものが含まれていてはなりません。
5. 変数名を指定した場合は、AUTOTBL コマンドが発行された時点で、指定されたメンバーまたはファイルが NetView プログラムによって組み込まれます。NetView プログラムが変数を検索する順序は、次のとおりです。
 - AUTOTBL コマンドがコマンド・プロシージャーから出された場合、NetView プログラムは *varname* という名前のローカル変数を最初に検索します。
 - AUTOTBL がコマンド・プロシージャーから出されていない場合、もしくは *varname* という名前のローカル変数がない場合、NetView プログラムはタスク・グローバル変数を次に検索し、最後に共通グローバル変数を検索します。

自動化テーブルを活動化したあとに変数の値を変更した場合、組み込まれているメンバーは、AUTOTBL コマンドが再発行されない限り、変更されません。

例えば、*&varname* を使用して、特定の時間帯に予想されるメッセージおよび MSU の通信量をチューニングするためにテーブル・セグメントを組み込む場合、コマンド・プロシーチャーは、時刻を基準にして、自動化テーブルをロードする前に変数を更新する可能性があります。

%INCLUDE ステートメントは、自動化テーブルだけでなく、その他の DSIPARM メンバーでも使用できます。%INCLUDE ステートメントに関する完全な説明については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

SYN ステートメント

SYN ステートメントは、後に自動化テーブルで使う同義語を定義するために使用します。同義語には、名前と値があります。同義語を定義したあとは、その同義語の名前をテーブル内のどこでも使用できます。そのテーブルを活動化すると、NetView プログラムはその名前の代わりにその同義語の値を使用します。

同義語を使うと、繰り返し使用する長いストリングを短く表記できます。また、同義語を使用した場合には、1 箇所の値を修正するだけで、テーブル内に使用されているすべての箇所についての修正が済むので、自動化テーブルの修正と管理も容易になります。

SYN ステートメントの構文は、次のとおりです。

SYN ステートメント

```
▶▶—SYN— —%—synname—%— —==— —'—synvalue—'—;————▶▶
```

項目の意味:

SYN 各 SYN ステートメントの先頭にコーディングするキーワードです。

synname

同義語の名前で、最長 256 文字です。

synvalue

同義語の値です。

使用上の注意:

1. 同義語を自動化テーブルで使用する前に、必ずその同義語を定義しておくようにしてください。同義語の値は、テーブル内で一度しか定義できませんが、その同義語はそのあと何度でも使用できます。同義語はすべて、テーブルの先頭で定義するようにしてください。
2. 同義語を、別の同義語の中に含めることはできません。
3. 同義語の名前と値には、以下のものを除いてブランク、英数字、およびその他の文字を使用できます。
 - 同義語の名前には、パーセント記号 (%) もセミコロン (;) も使用できません。

- 同義語の値には、セミicolon (;) は使用できません。
- 同義語値の区切り文字には単一引用符が使用されるため、同義語値に単一引用符 (') を使用する場合は、代わりに単一引用符を 2 つ続けて使用する (") 必要があります。2 つの単一引用符を用いる代わりに、二重引用符を使用することはできません。同義語に単一引用符が含まれている例を、図 56 に示します。

```

SYN %LogFullCondition% = 'MSGID= 'IFB040I''';
:
IF %LogFullCondition% THEN
    EXEC(CMD('MVS S CLRLOG') ROUTE(ONE AUTO1));

```

図 56. SYN ステートメントの使用例

4. 引用符内に同義語がある場合、同義語値への置き換えは行われません。引用符内にある同義語は、文字ストリングとして処理されます。例えば、図 57 のような SYN ステートメントと自動化テーブル項目が指定されている場合を例にとります。

```

SYN %LDOMAIN% = '''CNM01''';
:
IF MSGID = 'DSI530I' & DOMAINID = '%LDOMAIN%' THEN
    EXEC (CMD('CLIST1 ')ROUTE(ONE AUTO1));

```

図 57. 同義語の置き換えが適用されない例

図 57 のステートメントは正しい構文で指定されているにもかかわらず、同義語 %LDOMAIN% は単一引用符内にコーディングされているために、置き換えは行われません。同義語の一部として単一引用符を組み込みたい場合は、SYN ステートメントと自動化テーブルを図 58 のようにコーディングしてください。

```

SYN %LDOMAIN% = '''CNM01''';
:
IF MSGID = 'DSI530I' & DOMAINID = %LDOMAIN% THEN
    EXEC (CMD('CLIST1 ')ROUTE(ONE AUTO1));

```

図 58. 同義語の置き換えが適用される例

5. 同義語は、命名規則に従うようにしてください。

自動化テーブルの設計上のガイドライン

自動化テーブルの設計またはコーディングを行うときは、以下にリストする技法を使用してください。

システム・メッセージ処理の制限

NetView プログラムが処理するシステム・メッセージの数を、制限するようにしてください。NetView プログラムによってメッセージの自動化および表示を行わない場合は、MPF などのオペレーティング・システム機能を使用すると、NetView プログラムへのメッセージの送信を回避できます。こうすると、NetView プログラムが自動化テーブルを検索しなければならない回数が減って、パフォーマンスが向上します。詳しくは、「341 ページの『システム・メッセージの抑止』」を参照してください。

読みやすい自動化テーブルの作成

以下のアクティビティを実行して、自動化テーブルを読みやすく整合性のとれたものにし、保守しやすくしてください。

- 各自動化テーブルのメンバーまたはファイルの先頭にコメントを付け、そのメンバー内のステートメントの説明を入れます。
- コメントを使用して、どのメッセージまたは MSU に自動化ステートメントが一致する必要があるか、およびどのステートメントのアクションが実行されるかを説明します。
- 字下げしてください。

例えば、IF-THEN ステートメント、BEGIN キーワード、およびそれらの対となる END ステートメントのアクション、ならびに BEGIN-END セクション内のステートメントのアクションは、字下げするようにしてください。

コメントは、1 桁目から始めなければなりませんが、ステートメントにはその必要はありません。1 つのステートメント内にブランクを何個でも使用できます。ただし、ステートメントはセミコロンで終わらなければなりません。

- 自動化メンバーにはブランク行とコメントを入れて、ステートメントとステートメント・グループを区切ります。
- 自動化テーブル・メンバーは、命名規則に従ってください。
- 自動化テーブル論理を複数のメンバーに分離します。そうすれば、この論理を必要に応じて使用可能にしたり、使用不可にすることができます。
- LABEL-ENDLABEL キーワードまたは GROUP キーワードを使用して、自動化テーブル・ステートメントまたはステートメントのグループを定義し、必要に応じて自動化テーブル論理の使用可能または使用不可が行えるようにします。

BEGIN-END セクションによるステートメントのグループ化

BEGIN-END セクションを使用して、容易に識別可能なメッセージ・タイプおよび MSU タイプを、それぞれテーブル内の専用のセクションにまとめて指定すれば、自動化テーブルのパフォーマンスが向上します。また、自動化テーブルの読み取り、理解、および保守が容易になります。

各セクションの末尾に ALWAYS ステートメントを使用すると、該当するステートメントがそのセクション内になかったメッセージおよび MSU の処理を指定できます。268 ページの図 59 では、ALWAYS ステートメントを使用して、該当するステートメントのないメッセージおよび MSU がそれ以上処理されないようにします。


```

* Statements for major-vector X'0000' MSUs
IF MSUSEG(0000) ^= '' THEN
  BEGIN;
  :
  ALWAYS;
END;
* Statements for VTAM messages
IF MSGID = 'IST' . THEN
  BEGIN;
  :
  ALWAYS;
END;
* Statements for JES2 messages
IF MSGID = '$HASP' . THEN
  BEGIN;
  :
  ALWAYS;
END;
* Statements for command-facility DSI messages
IF MSGID = 'DSI' . THEN
  BEGIN;
  :
  ALWAYS;
END;
* Statements for hardware monitor messages
IF MSGID = 'BNJ' . THEN
  BEGIN;
  :
  ALWAYS;
END;

```

図 59. ステートメントのグループ化の例

テーブルを複数セクションに分割すると、一致するメッセージまたは MSU を検索するための処理対象になるステートメントの数を最小限にできます。図 59 では、NetView コマンド機能 DSI メッセージの場合、NetView プログラムは以下の 2 つのステートメントをチェックするだけで、コマンド機能 DSI メッセージのセクションに到達できます。

- MSGID = 'IST' ステートメント
- MSGID = '\$HASP' ステートメント

MSUSEG は MSU のみに関係するため、NetView プログラムは、メッセージに対して MSUSEG(0000) ^= '' ステートメントのチェックは行いません。また、各セクション内の BEGIN-END セクションの数を増やすと、自動化テーブルで比較を行わなければならない回数を減らせます。

BEGIN-END セクションは、使用頻度の高い順にテーブルに並べてください。AUTOCNT コマンドを使うと、自動化テーブル使用報告書を生成して、メッセージおよび MSU の頻度を調べられます。

サンプルの NetView 自動化テーブル DSITBL01 (CNMS1015) では、メッセージのグループ化のために BEGIN-END セクションを使用しています。

269 ページの図 60 のように、BEGIN-END セクション内にオカレンス検出条件項目 (THRESHOLD および INTERVAL) を使用すると、オカレンス検出しきい値に到達したかどうかによって異なるアクションを取るよう指定できます。

```

IF MSGID = 'IST102I' THEN
BEGIN;
  IF THRESHOLD(3 7 00:00:00) = '1' THEN
    EXEC(CMD('MSG SYSOP VTAM DOWN - ASSISTANCE REQUIRED'))
      ROUTE(ONE AUTOVTAM AUTO1 PPT *));
  ALWAYS
    EXEC(CMD('VTAMSTRT'))
      ROUTE(ONE AUTOVTAM AUTO1 PPT *)
    EXEC(CMD('MSG SYSOP AUTOMATION RESTARTING VTAM'))
      ROUTE(ONE AUTO1 PPT *));
END;

```

図 60. オカレンス検出条件項目の例

図 60 の例を使用すると、VTAM がアクティブでなくなった場合に、NetView プログラムは以下の 1 つを実行します。

- 過去 7 日以内にこれと同じことが少なくともほかに 2 回起こっている場合、NetView プログラムはシステム・オペレーターにメッセージを送って、オペレーターに問題を知らせます。
- 過去 7 日以内で、この事態が起こったのが初めてか 2 回目の場合、NetView プログラムは、以下のアクティビティーを実行します。
 - VTAM プログラムの再始動を試行します。
 - 自動化テーブルが、自動タスクに VTAM プログラムを再始動する要求を送信したことをシステム・オペレーターに通知します。

複雑な比較項目の分離

比較項目によっては、他の比較項目より評価に長い時間がかかるものがあります。比較的時間がかかる可能性がある比較項目には、以下のものがあります。

- 複雑な場所を指定する MSUSEG 比較項目
- NetView プログラム提供の DSICGLOB ATF プログラム
- 独自で作成した長い ATF プログラム

これらの項目は、NetView プログラムは、IF-THEN の比較評価が真になった場合にのみ、その項目の評価を行うように、IF-THEN ステートメントで始まる BEGIN-END セクションに入れて分離することができます。

項目を分離するには、分離したい項目を論理 AND 演算子 (&) の後に指定することもできます。この場合、NetView プログラムは新しいメッセージまたは MSU の作成を試行します。例えば、図 61 のステートメントでは DSICGLOB ATF が分離されているため、NetView プログラムは、XYZ123 という ID を持つメッセージが受信された場合にのみ共通グローバル変数 REQUIREDSTATUSB の値を取り出します。

```

IF MSGID = 'XYZ123' &
  ATF('DSICGLOB REQUIREDSTATUSB') = 'ACTIVE' THEN
  EXEC (CMD('RESTARTB'));

```

図 61. 複雑な比較項目を分離する例

他の自動化テーブルの組み込み

自動化テーブルの更新を容易にするには、%INCLUDE ステートメントを使って、複数の自動化メンバーを処理する自動化テーブルを作成します。例えば、メッセージおよび MSU のクラスごとに別々のメンバーを定義するという方法があります。また、複数の自動化テーブルに共通する自動化テーブル・ステートメントを 1 つのメンバーに入れ、そのメンバーを各自動化テーブルに組み込むという方法もあります。

図 62 は、他の自動化メンバーを組み込む自動化テーブルの先頭部分を示しています。

```
* Main automation table
* Synonym definitions
%INCLUDE ATSYNS
* Statements for major-vector X'0000' MSUs
%INCLUDE AT0000
* Statements for VTAM messages
%INCLUDE ATVTAM
* Statements for JES2 messages
%INCLUDE ATJES2
* Statements for NetView command-facility DSI messages
%INCLUDE ATNVDSI
* Statements for NetView hardware monitor messages
%INCLUDE ATNVBNJ
```

図 62. 他の自動化テーブルを組み込む例

図 62 では、自動化テーブル・メンバーの ATSYNS、AT0000、ATVTAM、ATJES2、ATNVDSI、および ATNVBNJ に、同義語を定義するステートメント、X'0000' 主ベクトル、VTAM メッセージ、JES2 メッセージ、NetView DSI メッセージ、および NetView BNJ メッセージがそれぞれ指定されています。

注: 組み込まれたメンバーまたはファイル内に BEGIN-END セクションを使用すると、自動化テーブルの効率を上げられます。

運用のための自動化テーブルの調整

組み込まれるセクションは、運用のフェーズごとに別々に作成してください。例えば、テーブルを時間帯ごとに作成することができます。各テーブルを、特定の時間帯に予想されるメッセージおよび MSU の通信量に合わせてチューニングできます。

もう 1 つのアプローチは、一連の自動化テーブルを並行して使用することです。各テーブルは、AUTOTBL または AUTOMAN コマンドを使用してロードまたは除去することができます。自動化テーブルをロードするときに、他のアクティブな自動化テーブルに関連させて、テーブルを処理する順序を指定することができます。これは、勤務外時間帯および通常勤務時間帯などの、特定時間帯における自動化方針を変更するときに役に立ちます。

同義語の使用

同義語を使用すると、自動化テーブル内の複雑なストリングまたは反復するストリングを定義したり、複数のシステムに使われる自動化テーブルの使用法を標準化したりすることができます。システムに従属する指定はすべて同義語値として定義

し、同義語は自動化テーブルの先頭か組み込まれるメンバーの中に指定できます。そうすれば、自動化テーブルを他のシステムにコピーした場合に変更する必要があるのは、同義語値またはその同義語を含むメンバーだけになります。

注: 同義語は、それを使用しているテーブルと同じテーブルで定義する必要があります。複数のテーブルにまたがって適用することはできません。

ステートメントの正しい配置

自動化テーブルの中または自動化テーブルのセットの中でのステートメントの順序は、慎重に決めなければなりません。ステートメントの位置または指定が正しくないと、以下のような状態が起こることがあります。

- メッセージまたは MSU が、予定外のステートメントに一致した。
- BEGIN-END セクションを誤った位置に指定したために、メッセージまたは MSU が、予定していたステートメントに一致しなかった。
- CONTINUE アクションを誤った位置に指定したために、1 つのステートメントにのみ一致するはずであったメッセージまたは MSU が、複数のステートメントに一致してしまった。

自動化テーブル・リストの使用

自動化テーブル・リスト機能を使用して、既存の自動化テーブル内の新規ステートメントを入れる位置を判別します。また、問題判別用のリストを使用して、単一の自動化テーブル内の構文エラーや不正な位置に配置されたステートメントを検出できます。リストには、組み込まれたすべてのメンバー、同義語名に対応する同義語値、BEGIN-END セクションのレベル、および日付と時刻のスタンプが示されます。また、テーブル内で検出されたすべてのエラーもリストされます。詳しくは、「273 ページの『自動化テーブル・リストの例』」を参照してください。

ALWAYS ステートメントの使用

ALWAYS ステートメントを、以下のように、アクションまたはアクション・リストに使用します。

- テーブル内の特定の箇所で、自動化処理を停止します。

例えば、BEGIN-END セクションの最後のステートメントとして ALWAYS を使用することで、誤ったステートメントに一致しないようにし、パフォーマンスを向上させます。アクションなしの ALWAYS ステートメント (ALWAYS;) をコーディングすれば、メッセージまたは MSU に対する自動化処理を停止できます。

- テーブルの 1 つのセクションにデフォルトを設定します。

例えば、あるメッセージ・グループに対して、特定のアクションを取ることができます。そのような目的で ALWAYS を CONTINUE と同時に指定する例が、272 ページの『自動化テーブルのデフォルトの設定』にあります。

- テストを容易にします。

例えば、メッセージ頻度を分析したり監査証跡を入手したりするために、特定のメッセージ・グループのすべてのインスタンスを、一定時間ログに記憶することができます。

CONTINUE アクションの正しい使用

CONTINUE アクションは、特に慎重に使用してください。CONTINUE を不適切に使用すると、コマンドまたはコマンド・リストが 1 つだけ処理されるべきところを複数処理されてしまうなど、意図しないアクションが行われる場合があります。

1 つのメッセージまたは MSU に対して複数のアクションを行いたい場合には、CONTINUE アクションが便利です。図 63 では、コマンド機能メッセージがすべて順次ログ・ファイルに記録されるため、頻度の分析が容易になります。

```
* Process NetView command facility DSI messages
IF MSGID = 'DSI' . THEN
  BEGIN;
*   Temporary statement to send the message ID to a
*   sequential log file for frequency analysis.
*
*   The statement extracts the message ID in the
*   variable MSGIDVAR and calls the command processor
*   LOGMSGID, which uses NetView program's sequential log
*   facility to write the ID to a sequential file.
  IF MSGID = MSGIDVAR THEN
    EXEC (CMD('LOGMSGID ' MSGIDVAR))
    CONTINUE(Y);
*   Permanent statements that automate various DSI messages.
*
*
  END;
```

図 63. CONTINUE キーワードの使用例

自動化テーブルのデフォルトの設定

CONTINUE および ALWAYS を使って、特定の入力でオーバーライド可能な、自動化テーブルまたは自動化テーブル・セクションのデフォルトを設定します。

```
ALWAYS SYSLOG(Y) NETLOG(N) CONTINUE(Y);
```

図 64. ALWAYS ステートメントで CONTINUE キーワードを使用する例

図 64 のようにステートメントを指定した場合、デフォルトとして、すべてのメッセージはシステム・ログに記録されますが、ネットワーク・ログには記録されません。メッセージが、このテーブルに指定されている位置よりあとにあるステートメントに一致する場合、そのステートメントは明示的に SYSLOG(N) または NETLOG(Y) を指定して、ALWAYS ステートメントで設定されたデフォルトをオーバーライドする必要があります。SYSLOG と NETLOG は、メッセージに限定されるアクションなので、図 64 のステートメントは MSU には影響しません。

コマンド応答の自動化の制限

自動化テーブルでのコマンド応答メッセージのテストは信頼できないので、自動化テーブルでのコマンド応答のテストは限定的に行うようにしてください。応答を処理する PIPE または CLIST からコマンドを発行すれば、それらのコマンドは自動化テーブルからは見えません。また、これらの同じコマンドを、結果としてメッセージが公開されるような別のソースから発行することもできます。

NetView プログラム終了時の自動操作

CLOSE STOP コマンドまたは CLOSE IMMED コマンドを発行すると、既存のグローバル保持は終了し、ENDCMD ステートメントで定義したコマンドは、CNMSTYLE メンバーの endcmd.AutoTask ステートメントで定義したタスクのキューに入れられます。すべてのコマンドは、CNMSTYLE メンバーの endcmd.close.leeway ステートメントで指定した時間内に完了する必要があります。一部の機能は、この期間内にはサポートされません。待ち時間が長いか不確定のコマンド (例えば、NetView TSO サーバーに送信されるコマンド、RMTCMD コマンド、WTOR コマンドなど) は、使用しないようにしてください。メッセージ自動操作は、endcmd.close.leeway ステートメントで定義した期間だけ続行できます。自動操作では、この期間内に新規コマンドのスケジュールを設定することはできません。

自動化テーブル・リストの例

このセクションでは、2 つのメンバーで構成される自動化テーブルの例を示します。図 65 および 274 ページの図 66 を参照してください。

275 ページの図 67 は、AUTOTBL LISTING 機能を使って、メンバーから生成できる自動化テーブル・リストを示します。さらに、275 ページの図 67 では、自動化テーブルの作成、チューニング、デバッグ時に、リストの情報をどのように使用すればよいかを示します。

図 65 に示す第 1 のメンバーには、第 2 のメンバーに組み込まれる自動化テーブル同義語ステートメントが含まれます。

```
* Set table synonyms
SYN %MYDOMAIN% = '''CNM01''';
SYN %NETL3% = 'NETLOG(YES 3 +STATGRP)';
```

図 65. 自動化テーブル同義語ステートメントの例

274 ページの図 66 に示す第 2 のメンバーは、自動化テーブル・メンバーの主要メンバーであり、AUTOTBL コマンドで指定するものです。


```

* Include the member that contains the synonym definitions
%INCLUDE EXSYNS

* Set table defaults and continue processing
ALWAYS SYSLOG(Y) NETLOG(N) DISPLAY(Y)
    CONTINUE(Y);

* All DSI messages go here
IF MSGID = 'DSI' . THEN
    BEGIN;

* Invoke the PDFILTER command list automatically when the hardware
* monitor completes initialization. (Use a synonym to check domain.)
    IF MSGID = 'DSI530I' &
        TEXT = . 'BNJDSERV' . &
        DOMAINID = %MYDOMAIN% &
        TEXT = MESSAGETEXT THEN
        EXEC(CMD('PDFILTER ' MESSAGETEXT));

* Handle the DSI701I message. (Use a synonym to specify the action.)
    IF MSGID = 'DSI701I' THEN
        %NETL3%;

    END;

* Any statements for CNM messages go here
IF MSGID = 'CNM' . THEN
    BEGIN;

* Suppress the CNM094I message
    IF MSGID='CNM094I' THEN
        DISPLAY(N)
        NETLOG(N);

    END;

* This is not a valid statement for handling syntax errors.
IF BADFUNC = 'INFO' THEN
    DISPLAY(N);

```

図 66. 自動化テーブルの主要メンバーの例

図 66 に示したテーブルを生成するには、LISTING キーワードを指定した AUTOTBL コマンドを発行します。この結果は、275 ページの図 67 のリストのようになります。このリストには、自動化テーブルに関する各種情報が、以下のように含まれています。

- 見出し行は、発行された AUTOTBL コマンド、そのコマンドを実行したタスク、および時刻を示します。
- 開始行および終了行には、テーブル中の各メンバーの開始位置と終了位置が表示されます。
- 1 桁目のアスタリスク (*) は、コメント行のマークです。
- 定義した同義語は、解決済みです。つまりこの例では、%MYDOMAIN% および %NETL3% は、それぞれの値で置き換えられています。
- リストには、テーブル中の各ステートメントが以下のように記述されています。
 - 1 桁目から 4 桁目は、ステートメント番号を示します。
 - 6 桁目から 8 桁目は、BEGIN-END のネスト・レベルを示します。例えば 001 は、BEGIN-END セクションに含まれないステートメントを表し、002 は、第 1 レベルの BEGIN-END セクションに含まれるステートメントを表します。
 - 10 桁目から 72 桁目は、ステートメント・テキストです。

- 73 桁目から 80 桁目は、シーケンス番号 (存在する場合) を示します。
- エラー・メッセージは、正しくない各ステートメントのあとに続けて示されます。
- リストの末尾は、エラー総数を示す行になっています。

```
* Set table synonyms
0001 001 SYN %MYDOMAIN% = '''CNM01''';
0002 001 SYN %NETL3% = 'NETLOG(YES 3 +STATGRP)';
----- END OF 'EXSYNS' -----
* Set table defaults and continue processing
0003 001 ALWAYS SYSLOG(Y) NETLOG(N) DISPLAY(Y) CONTINUE(Y);
* All DSI messages go here
0004 001 IF MSGID = 'DSI' . THEN BEGIN;
* Invoke the PDFILTER command list automatically when the hardware
* monitor completes initialization. (Use a synonym to check domain.)
0005 002 IF MSGID='DSI530I' & TEXT = . 'BNJDSERV' . & DOMAINID =
      'CNM01' & TEXT = MESSAGETEXT THEN EXEC(CMD('PDFILTER '
      MESSAGETEXT));
* Handle the DSI701I message. (Use a synonym to specify the action.)
0006 002 IF MSGID = 'DSI701I' THEN NETLOG(YES 3 +STATGRP);
0007 002 END;
* Any statements for CNM messages go here
0008 001 IF MSGID = 'CNM' . THEN BEGIN;
* Suppress the CNM094I message
0009 002 IF MSGID='CNM094I' THEN DISPLAY(N) NETLOG(N);
0010 002 END;
* This invalid statement demonstrates handling of syntax errors
0011 001 IF BADFUNC = 'INFO' THEN DISPLAY(N);
ERROR    CNM505E INVALID FUNCTION NAME "BADFUNC" SPECIFIED IN
          CONDITIONAL
----- END OF 'EXMAIN' -----
1 STATEMENT(S) IN ERROR
```

図 67. 自動化テーブル・リストの例

自動化テーブル使用報告書

NetView 自動化テーブル処理は、1 組のカウンターにより、イベントが特定の基準のセットと比較される回数、および自動化アクションが実行される回数を追跡します。この情報を使うと、自動化テーブルに対して環境に適したチューニングを行えます。例えば、一致する頻度の高いステートメントをテーブルの前の方に移動すると、チェック回数を減らせます。また、一度も一致しないステートメントを調べ、論理エラーのためにそれらを削除したり変更したりする必要があるかを判別することもできます。

さらにこの使用報告書を使用して、システムで自動化を実行するレベルを判別することもできます。

AUTOCNT コマンド

AUTOCNT コマンドは、アクティブな NetView 自動化テーブル、または AUTOTEST コマンドによってテスト中の NetView 自動化テーブルの、自動化テーブル・ステートメントの使用状態を記述する報告書を作成するものです。また、自動化テーブル・ステートメント使用カウンターを、リセットするためにも使用できます。

AUTOCNT コマンドで要求できる情報と統計の自動化ステートメント・タイプは、メッセージ・タイプ、MSU タイプ、もしくは両方です。要求対象には、要約情報と詳細情報があり、詳細情報は、メッセージおよび MSU が各自動化テーブル・ステートメントと比較された回数、および一致した回数を表示します。

『使用報告書の出力の例』には、自動化テーブル使用報告書の例と、要約報告書と詳細報告書の違いを説明します。また、自動化テーブル管理 (AUTOMAN) 機能を使用して、この情報表示をすることができます。

この情報および統計は、複数行メッセージとして表示できます。この情報は、ファイルに保管することもできます。

AUTOCNT コマンドの構文およびパラメーターの説明については、オンライン・コマンド・ヘルプを参照してください。

使用報告書の出力の例

このセクションでは、自動化テーブル使用報告書の例を記載します。まず自動化テーブル・ソースのメンバーがあり、次に詳細および要約の使用報告書があります。自動化のソース・メンバーを使用すると、使用報告書のシーケンス番号 (SEQ NUMBER) のフィールドとメンバー名 (MEMBER NAME) のフィールドを、ソース・メンバーと突き合わせて、詳細な使用統計で参照しているステートメントを特定できます。また、アクティブな自動化テーブルのどのセクションが使用不可になっているかを判別することができます。また、自動化テーブル・リストを使用すると、ステートメント番号 (STMT NUMBER) フィールドとリストされたステートメント番号とを突き合わせて、ステートメントの相関関係を調べられます。

複数行メッセージを別に生成できるのは、以下の各タイプの報告書の場合です。

- アクティブな自動化テーブルにあるすべてのメッセージ・タイプのステートメントに対する詳細報告書
- アクティブな自動化テーブルにある MSU タイプのステートメントに対する詳細報告書
- アクティブな自動化テーブルにあるすべてのタイプのステートメントに対する要約報告書
- アクティブな自動化テーブルにある MSU タイプのステートメントに対する要約報告書

詳細報告書はテーブル内の各ステートメントの使用状況を示すため、以下の目的で使用できます。

- 自動化テーブルのチューニング
- 一度も一致しないか常に一致する事態を引き起こす論理エラーを伴うステートメントの識別
- 新規のステートメントのテスト
- 特定のメッセージおよび MSU の通信量の判別

要約報告書はアクティブな自動化テーブル全体に関する総合的な使用状況を示すため、以下の目的で使用できます。

- 容量計画

- 新規自動化の追加結果の判定
- 傾向分析
- 汎用メッセージおよび MSU の通信量の判定

AUTOCNT コマンドで使用報告書を生成できるステートメント・タイプは、メッセージ・タイプ、MSU タイプ、もしくは両方です。AUTOCNT コマンドで `STATS=SUMMARY` を指定すると、要約のみの報告書が生成されます。AUTOCNT コマンドで `STATS=DETAIL` を指定すると、詳細と要約の両方の報告書が生成されます。

278 ページの図 68 に示す自動化テーブル・メンバー (シーケンス番号付き) の例は、1 時間前に活動化された自動化テーブルに関するものです。また、279 ページの図 69 は、自動化テーブルのリスト (ステートメント番号付き) です。

```

*****
*      BEGINNING OF 'IST' MSGID'S      *
*****
*
IF MSGID = 'IST' . THEN
  BEGIN;
    IF MSGID = 'IST097I' |
      MSGID = 'IST314I' |
      MSGID = 'IST526I' THEN
      DISPLAY(N) NETLOG(N) SYSLOG(N);
    IF MSGID = 'IST259I' THEN
      EXEC(CMD('INOPRU')) ROUTE(ONE AUTOVTAM AUTO1 * PPT));
    IF (LABEL: REACTIVATEVTAM)
      MSGID = 'IST102I' &
      ATF('DSICGLOB VTAMDESIRED') = 'ACTIVE' THEN
      COLOR(PIN)
      EXEC(CMD('VTAMSTRT')) ROUTE(ONE AUTOVTAM AUTO1 * PPT));
    ALWAYS;
  END;
*
*****
*      BEGINNING OF 'DSI' MESSAGES      *
*****
*
IF (LABEL: DSIPREFIX) MSGID = 'DSI' . THEN
  BEGIN;
    IF (LABEL: ALTD SIPREFIX)
      MSGID = 'DSI034I' |
      MSGID = 'DSI201I' |
      MSGID = 'DSI208I' |
      MSGID = 'DSI633I' THEN
      DISPLAY(N) NETLOG(N) SYSLOG(N);
    IF (ENDLABEL: ALTD SIPREFIX)
      MSGID='DSI374A' THEN
      HOLD(Y) BEEP(Y) DISPLAY(Y)
      EXEC(ROUTE(ALL * +GRPOPS));
    ALWAYS;
  END;
*
*****
*      BEGINNING OF MSU SECTION          *
*****
*
IF MSUSEG(0000) = ' ' THEN
  BEGIN;
*   PROBABLE CAUSE: COMMUNICATION CONTROLLER OR TERMINAL CONTROL UNIT
  IF (GROUP: MSUSTATEMENT)
    MSUSEG(0000.93 3) = HEX('3111') . |
    MSUSEG(0000.93 3) = HEX('3121') . THEN
      COLOR(RED) XHILITE(REV)
      SRF(ROUTE PASS);
*   PROBABLE CAUSE: LAN COMPONENT OR LAN ADAPTER
  IF (GROUP: MSUSTATEMENT)
    MSUSEG(0000.93 3) = HEX('37') . |
    MSUSEG(0000.93 3) = HEX('332') . THEN
      COLOR(PIN) XHILITE(REV)
      SRF(ROUTE PASS);
    ALWAYS;
  END;

```

SEQ00001

SEQ00002

図 68. 自動化テーブル・メンバー

```

----- START OF 'AUTOSEG1' -----
*****
*      BEGINNING OF 'IST' MSGID'S      *
*****
*
0001 001 IF MSGID = 'IST' . THEN BEGIN;
0002 002 IF MSGID = 'IST097I' | MSGID = 'IST314I' | MSGID = 'IST526I'
      THEN DISPLAY(N) NETLOG(N) SYSLOG(N);
0003 002 IF MSGID = 'IST259I' THEN EXEC(CMD('INOPRU') ROUTE(ONE
      AUTOVTAM AUTO1 * PPT));
0004 002 IF (LABEL: REACTIVATEVTAM) MSGID = 'IST102I' & ATF('DSICGLOB
      VTAMDESIRED') = 'ACTIVE' THEN COLOR(PIN)
      EXEC(CMD('VTAMSTRT') ROUTE(ONE AUTOVTAM AUTO1 * PPT));
0005 002 ALWAYS;
0006 002 END;
*
*****
*      BEGINNING OF 'DSI' MESSAGES      *
*****
*
0007 001 IF (LABEL: DSIPREFIX) MSGID = 'DSI' . THEN BEGIN;      SEQ00001
0008 002 IF (LABEL: ALTD SIPREFIX) MSGID = 'DSI034I' | MSGID =
      'DSI201I' | MSGID = 'DSI208I' | MSGID = 'DSI633I' THEN
      DISPLAY(N) NETLOG(N) SYSLOG(N);
0009 002 IF (ENDLABEL: ALTD SIPREFIX) MSGID='DSI374A' THEN HOLD(Y)
      BEEP(Y) DISPLAY(Y) EXEC(ROUTE(ALL * +GRPOPS));
0010 002 ALWAYS;
0011 002 END;
*
*****
*      BEGINNING OF MSU SECTION      *
*****
*
0012 001 IF MSUSEG(0000) ^= '' THEN BEGIN;      SEQ00002
*   PROBABLE CAUSE: COMMUNICATION CONTROLLER OR TERMINAL CONTROL UNIT
0013 002 IF (GROUP: MSUSTATEMENT) MSUSEG(0000.93 3) = HEX('3111') . |
      MSUSEG(0000.93 3) = HEX('3121') . THEN COLOR(RED)
      XHILITE(REV) SRF(ROUTE PASS);
*   PROBABLE CAUSE: LAN COMPONENT OR LAN ADAPTER
0014 002 IF (GROUP: MSUSTATEMENT) MSUSEG(0000.93 3) = HEX('37') . |
      MSUSEG(0000.93 3) = HEX('332') . THEN COLOR(PIN)
      XHILITE(REV) SRF(ROUTE PASS);
0015 002 ALWAYS;
0016 002 END;
----- END OF 'AUTOSEG1' -----

0 STATEMENT(S) IN ERROR

```

図 69. サンプル・メンバーの自動化テーブル・リスト

この例におけるメッセージおよび MSU の処理の前提事項

この例では、自動化テーブル使用統計が取られていた過去 1 時間の間に、どのステートメントも使用不可になっていないことが前提となっています。自動化テーブルは、以下のメッセージを処理しました。

メッセージ・タイプ	メッセージの数
IST097I メッセージ	154
IST314I メッセージ	20
IST526I メッセージ	3
IST259I メッセージ	9

メッセージ・タイプ	メッセージの数
IST102I メッセージ	0
その他の IST 接頭部メッセージ	612
DSI034I メッセージ	3
DSI201I メッセージ	3
DSI208I メッセージ	39
DSI633I メッセージ	7
DSI374A メッセージ	1
その他の DSI 接頭部メッセージ	107
接頭部が IST か DSI 以外のメッセージ	1346

また、自動化テーブルは、以下の MSU を処理しました。

MSU タイプ	MSU の数
推定原因が X'3111' のアラート主ベクトル	2
推定原因が X'3121' のアラート主ベクトル	3
推定原因が X'37' のアラート主ベクトル	14
推定原因が X'332' のアラート主ベクトル	3
上記以外のアラート主ベクトル	3211
非アラート MSU	130

詳細な自動化テーブル使用報告書

詳細な自動化テーブル使用報告書には、アクティブな NetView 自動化テーブル内にある各自動化テーブル・ステートメントごとに、以下の情報が示されています。

- ステートメント番号 (STMT NUMBER)

自動化テーブル内のステートメントのシーケンス番号。この番号は自動化テーブル・リストにも表示されているため、突き合わせるリストの相関関係がわかるようになっています。

- ラベル・インディケーター (L I)

この欄には、指定されたラベルのタイプ、またはラベルがない場合には、シーケンス番号が指定されているかどうかを示す文字が入ります。このインディケーターの値は、以下のとおりです。

(blank)

このステートメントには、ラベルもシーケンス番号も指定されていない。

S このステートメントには、ラベルが指定されていない。しかし、シーケンス番号は指定されている。シーケンス番号は、次の欄に表示されている。

L 指定されるシーケンス番号に関係なく、このステートメントには LABEL の指定がある。 LABEL 名は、次の欄に表示されている。

E 指定されるシーケンス番号に関係なく、このステートメントには ENDLABEL の指定がある。 ENDLABEL 名は、次の欄に表示されている。

- B** 指定されるシーケンス番号に関係なく、このステートメントには LABEL の指定があり、これはその後の ENDLABEL の指定と一致しており、したがって、ステートメントの BLOCK を指定することになる。BLOCK 名は、次の欄に表示されている。
- G** 指定されるシーケンス番号に関係なく、このステートメントには GROUP の指定がある。GROUP 名は、次の欄に表示されている。

- シーケンス番号またはラベル名 (SEQUENCE NUMBER/ LABEL NAME)

この欄には、以下の値の 1 つが入ります。

(blank)

このステートメントには、シーケンス番号もラベル名も指定されなかった。

(シーケンス番号)

このステートメントには、ラベル名の指定がなく、シーケンス番号が指定された。

(ラベル名)

このステートメントに対する LABEL、ENDLABEL、または GROUP の仕様に指定された値を示すラベル名。

- メンバー名 (MEMBER NAME)

そのステートメントが指定されているメンバー名。これをシーケンス番号とともに見れば、自動化テーブルのソース・メンバー (またはファイル) との相関関係がわかります。

- 条件比較 (COMPARE COUNT)

対応する条件ステートメントが評価対象になった場合に増分されるカウンター。

- 評価一致 (MATCH COUNT)

対応する条件ステートメントが真と評価された場合に増分されるカウンター。その結果、そのステートメントに指定されたすべての自動化アクションが実行されます。

- 実行対象コマンド (E C)

この欄は、評価一致時にこの自動化ステートメントで実行されるコマンドの数を報告するものです。CMD キーワードの指定された EXEC アクションの数が 99 より多い場合、アスタリスク (*) がこの欄に表示されます。

- 継続インディケーター (C I)

報告欄に X とマーク付けされている場合、CONTINUE アクションが条件ステートメントに含まれているために、NetView プログラムが自動化テーブルの操作を継続することを表します。CONTINUE(Y) アクションを実行すると、後続のステートメントによる条件処理が引き続き発生するため、後続のステートメントの条件に一致する可能性があります。

- ALWAYS ステートメント・インディケーター (A I)

報告欄に X とマーク付けされている場合、そのステートメントが ALWAYS であったことを表します。ALWAYS ステートメントの場合、MATCH/COMP フィールドは常に 100% です。

- 使用不可インディケーター (D I)

この欄は、ステートメントが現在 DISABLE 要求の一部であるのか、または最後に使用統計がリセットされて以降の DISABLE 要求の一部であったのかを記述します。値は以下のとおりです。

(blank)

ステートメントは、最後に使用統計がリセットされて以降、DISABLE 要求の一部ではなかった。

- d** 個々のステートメントは、最後に使用統計がリセットされて以降、使用不可になっていた。しかし現在は使用不可になっていない。
- b** ステートメントのブロックは、最後に使用統計がリセットされて以降、使用不可になっていた。しかし現在は使用不可になっていない。
- S** ステートメントは、そのシーケンス番号を使用していて、現在使用不可になっている。
- L** ステートメントは、LABEL 要求を使用していて、現在使用不可になっている。
- E** ステートメントは、ENDLABEL 要求を使用していて、現在使用不可になっている。
- B** ステートメントは、BLOCK 要求を使用していて、現在使用不可になっている。
- G** ステートメントは、GROUP 要求を使用していて、現在使用不可になっている。

注: 個々の自動化テーブル・ステートメント、ブロック、あるいはグループの状況を表示するために、自動化テーブル管理 (AUTOMAN) 機能を使用してください。

- 比較条件の一致率 (MATCH/COMP)

MATCH COUNT を条件ステートメントの COMPARE COUNT で割って、100 を掛けることにより計算された統計。一致回数も比較回数もともにゼロの場合、この率は -. と記述され、ゼロによる除算を表します。

- 比較率 (COMP/TOTAL)

条件ステートメントの COMPARE COUNT をメッセージ (または MSU) の総数で割り、100 を掛けた結果を計算した統計。このステートメントの比較回数も、自動化で処理されたメッセージまたは MSU の総数もともにゼロである場合、この率は -. と記述され、ゼロによる除算を表します。

- 一致率 (MATCH/TOTAL)

条件ステートメントの MATCH COUNT を、メッセージ (または MSU) の総数で割り、100 を掛けた結果を計算した統計。このステートメントに一致する回数も、自動化で処理されたメッセージまたは MSU の総数もともにゼロである場合、この率は -. と記述され、ゼロによる除算を表します。

数値欄の値が 99999999 を超える場合はすべて無効になり、8 個のアスタリスク (*) が代わりに示されます。

図 70 と図 71 には、詳細報告書の出力を表します。

```
------( AUTOSEG1 MESSAGE DETAILS 09/02/10 14:21:46 )-----
```

STMT NUMB	L I	SEQUENCE LABEL	NUMBER/ NAME	MEMBER NAME	COMPARE COUNT	MATCH COUNT	E C	A I	D I	PERCENTAGES MATCH/ COMP/ MATCH/ TOTAL	COMP/ TOTAL	MATCH/ TOTAL
0001				AUTOSEG1	2304	798	0			34.6	100.0	34.6
0002				AUTOSEG1	798	177	0			22.2	34.6	7.7
0003				AUTOSEG1	621	9	1			1.4	27.0	0.4
0004	L	REACTIVATEVTAM		AUTOSEG1	612	0	1			0.0	26.6	0.0
0005				AUTOSEG1	612	612	0	X		100.0	26.6	26.6
0007	L	DSIPREFIX		AUTOSEG1	1506	160	0			10.6	65.4	6.9
0008	L	ALTD SIPPREFIX		AUTOSEG1	160	52	0			32.5	6.9	2.3
0009	E	ALTD SIPPREFIX		AUTOSEG1	108	1	0			0.9	4.7	0.0
0010				AUTOSEG1	107	107	0	X		100.0	4.6	4.6

```
-----
```

図 70. MSG 詳細報告書

```
------( AUTOSEG1 MSU DETAILS 09/02/10 14:21:46 )-----
```

STMT NUMB	L I	SEQUENCE LABEL	NUMBER/ NAME	MEMBER NAME	COMPARE COUNT	MATCH COUNT	E C	A I	D I	PERCENTAGES MATCH/ COMP/ MATCH/ TOTAL	COMP/ TOTAL	MATCH/ TOTAL
0012	S	SEQ00002		AUTOSEG1	3363	3233				96.1	100.0	96.1
0013	G	MSUSTATEMENT		AUTOSEG1	3233	5				0.2	96.1	0.1
0014	G	MSUSTATEMENT		AUTOSEG1	3228	17				0.5	96.0	0.5
0015				AUTOSEG1	3211	3211		X		100.0	95.5	95.5

```
-----
```

図 71. MSU 詳細報告書

詳細統計には、各ステートメントを自動化処理した結果を示すことができます。比較カウントおよび一致カウントを調べて、自動化ステートメントの最適な順序を判別します。一般には、一致カウントの値が最高のステートメントは、それぞれの BEGIN-END セクションの先頭近くに指定するようにします。同様に、BEGIN-END セクション内のすべてのステートメントに対する一致カウント総数が最高の BEGIN-END セクションは、自動化テーブルの先頭近くに指定するようにします。

ステートメントの位置を移動する際には、テーブルの順次論理に影響しないように注意してください。

```
IF MSGID = 'XYZ123I' & TOKEN(5) = 'DEVICE1' THEN
    EXEC(CMD('COMMAND1'));
IF MSGID = 'XYZ456I' & DOMAINID = 'CNM99' THEN
    EXEC(CMD('COMMAND2'));
IF MSGID = 'XYZ123I' | MSGID = 'XYZ456I' THEN
    EXEC(CMD('COMMAND3'));
```

図 72. 使用統計で評価されるステートメント

ここで、図 72 のステートメントを検討してみます。使用統計で 3 番目のステートメントが一致する頻度が 1 番目、および 2 番目のステートメントより高いことがわかったとしても、3 番目のステートメントを前の 2 つのステートメントより順序を先にした場合、行われる自動化アクションが影響を受けてしまいます。最初に比

較される COMMAND3 を呼び出すステートメントが必ずメッセージに一致してしまうため、COMMAND1 および COMMAND2 を呼び出すステートメントは一度も一致しないようになります。

常に一致する IF-THEN ステートメントがある場合は、そのステートメントの指定に、論理エラーがないかどうか判別します。また、一度も一致しない IF-THEN ステートメントについては、以下のことを判別します。

- ステートメントの指定に論理エラーがないか
- そのステートメントより前にあるステートメントによって、そのステートメントに一致するメッセージまたは MSU が来ないようになっていないか
- もはやそのステートメントの必要はなく、除去する必要があるか

比較カウントおよび一致カウントを調べて、自動化ステートメントの最適な順序を判別します。新しい自動化ステートメントを追加する場合、COMPARE および MATCH COUNTS を調べると、追加により影響した部分がわかります。

一致回数が最高であるものが、1 つ以上のコマンド呼び出しを含むステートメントである場合、そのコマンドを呼び出すための呼び出し処理が過度である場合があります。呼び出されるコマンドがコマンド・リストであれば、LOADCL コマンドを使ってそれらを先にロードしておくようにしてください。

要約自動化テーブル使用報告書: 要約自動化テーブル使用報告書には、以下のような情報が、アクティブな NetView 自動化テーブルにあるメッセージ・タイプもしくは MSU タイプのすべてのステートメントについて記述されます。

- 使用報告書の生成日時

日付の形式は、*mm/dd/yy* です。時刻の形式は *hh:mm:ss* で、*hh* は 24 時間表示に基づくものです。この日時は、SUMMARY 統計 (メッセージ DWO810I および DWO811I) のラベル・メッセージで報告されます。

- 使用カウント・モニターの開始日時

日付の形式は、*mm/dd/yy* です。時刻の形式は *hh:mm:ss* で、*hh* は 24 時間表示に基づくものです。この日時は、メッセージ DWO812I で報告されます。

- 処理対象のメッセージまたは MSU の総数

自動化テーブル全体で処理されたすべてのメッセージもしくは MSU のカウント。

- 一致したメッセージまたは MSU の総数

少なくとも 1 つの自動化テーブル・ステートメントで処理されたメッセージまたは MSU の数。ALWAYS ステートメントでは、メッセージまたは MSU が条件一致になると見なされます。

- コマンドが実行された結果発生したメッセージまたは MSU の数

自動化テーブル・ステートメントから実行される 1 つまたは複数のコマンドの結果生じるメッセージまたは MSU の数のカウント。

- メッセージまたは MSU に対して実行されたコマンドの総数

統計が取られた時点での、一定間隔内にすべての自動化テーブル・ステートメントで実行されたコマンドの総数。EXEC アクションに CMD キーワードを指定した場合、コマンドが自動化テーブルから実行されることを表します。

- メッセージに対して実行された経路指定の総数

統計が取られた時点での、一定間隔中にすべての自動化テーブル・ステートメントによって実行された経路指定の総数。EXEC アクションに ROUTE キーワードを指定する (かつ、CMD キーワードは指定しない) 場合、自動化テーブルから経路指定が実行されたことを表します。

- メッセージまたは MSU 当たりの平均比較回数

比較回数を、自動化テーブル全体で処理されたメッセージまたは MSU の数で割ったもの。

- 分当たりに処理されたメッセージまたは MSU の数の平均

NetView 自動化テーブルで処理されたメッセージまたは MSU の数を、自動化テーブルを最後にリセットまたはロードしてから経過した時間 (分単位) で割ったもの。

- 経過時間 (分単位)

最後に AUTOCNT RESET コマンドが発行されてから、あるいは現在アクティブな自動化テーブルが活動化されてからの、総時間数 (分単位)。

図 73 および図 74 に、要約報告書の出力を表します。

```
- AUTOMATION TABLE MSG SUMMARY REPORT BY OPER1
----- ( AUTOSEG1 MESSAGE SUMMARY 09/02/10 14:21:46 ) -----
STATISTICS STARTED      = 09/02/10 13:58:34
TOTAL MSGS PROCESSED    =      2304
MSGs MATCHED           =       958
MSGs RESULTING IN COMMANDS =        9
TOTAL COMMANDS EXECUTED =        9
TOTAL ROUTES EXECUTED   =         1
AVERAGE COMPARES/MSG   =       2.58
TOTAL MSGS/MINUTE       =       38
MINUTES ELAPSED         =       60
-----
```

図 73. メッセージ自動化の MSG 要約報告書

```
- AUTOMATION TABLE MSU SUMMARY REPORT BY OPER1
----- ( AUTOSEG1 MSU SUMMARY 09/02/10 14:21:46 ) -----
STATISTICS STARTED      = 09/02/10 13:58:34
TOTAL MSUS PROCESSED    =      3363
MSUS MATCHED           =      3233
MSUS RESULTING IN COMMANDS =        0
TOTAL COMMANDS EXECUTED =        0
AVERAGE COMPARES/MSU   =       2.92
TOTAL MSUS/MINUTE       =       56
MINUTES ELAPSED         =       60
-----
```

図 74. MSU 自動化の MSU 要約報告書

要約統計の示す自動化処理の効果と効率、は、以下のように判定します。

- 分当たりのメッセージまたは MSU の数は、自動化処理の負荷を示します。

- メッセージまたは MSU 当たりの平均比較回数は、実行すべき自動化テーブル・アクション (存在する場合) の判別に要した自動化処理時間を示します。

平均比較回数が少ないほど、メッセージおよび MSU の自動化処理による CPU 使用率が低いことを表します。一般には、BEGIN-END セクションを追加するか、複数ステートメントを連結することで、平均比較回数を減らす効果があります。

一致しないメッセージの数が多いと、次のいずれかのアクティビティーを実行する必要がある可能性があります。

- 自動化ステートメントの追加
- オペレーティング・システムのメッセージ処理機能効率の改善。これによって、メッセージの自動化処理を行わなくてもよいようにします。
- より多くのメッセージの抑止

要約統計は、ヒストリーをとることにより、次の処理の影響を把握するのに非常に有効です。

- ネットワークへの装置の追加
- 自動化テーブルへのより多くの自動化ステートメントの追加
- 別の自動化テーブルの使用
- 自動化処理全体の時間帯もしくは日数の変更

ヒストリー統計をとっておくと、容量計画およびシステム・ストレス分析を行うときに役立ちます。

自動化テーブル使用報告書についての一般的な注意事項: 自動化テーブル使用報告書は、以下のような現行の自動化テーブルの使用に基づいています。

- AUTOTBL コマンドによって活動化されているテーブル
- AUTOTEST コマンドを使用してテスト中のテーブル

使用報告書が自動化テーブル・ステートメントと相関がとれていることを確実にするために、次のことを行います。

- ソース・メンバーに対する変更を特定の時刻にのみ許可し、変更を行う前にバックアップ・コピーを保管する。
- 新規の自動化テーブルを活動化するときには、自動化テーブル・リストを生成する。

自動化テーブルを活動化すると、AUTOCNT コマンドで使用するカウンターは、ゼロに設定されます。自動化テーブルを活動化するときに自動化テーブル・リストを生成する場合に、(その自動化テーブルが活動化されてから使用報告書が生成されるまでの間に) AUTOCNT RESET コマンドが発行されていなければ、リストに示された日時は、要約使用報告書の STATISTICS STARTED 日時に一致します。日時の比較は、詳細使用報告書と実際の自動化ステートメントの間が対応しているかどうかを検証する 1 つの方法です。

メッセージ・タイプと MSU タイプの両方を対象とするステートメントもありますが、この場合、そのステートメントはメッセージと MSU の両方の使用統計を対応させており、メッセージ詳細使用報告書と MSU 使用報告書の両方が別々にリストされます。

AUTOCNT コマンドの実行時にはメッセージおよび MSU は継続して自動化テーブル処理の対象になるため、使用報告書統計は正確でない場合もあります。自動化処理全体で現在実行されているメッセージまたは MSU は、詳細使用統計に影響する場合があります。使用統計は、正確なデータを知るためでなく、一般的な傾向を把握するために使用します。

複数の自動化テーブルの管理

AUTOTBL コマンドによって複数の自動化テーブルをロードすることができます。自動化テーブルは、通常は多くの組み込みメンバーで構成されています。自動化テーブル管理 (AUTOMAN) コマンドを使用して、選択されたテーブルまたはすべての自動化テーブルに影響を及ぼす変更項目に対して変更を行うことができます。自動化テーブルの作業時にユーザーを支援するように、AUTOMAN ではフルスクリーン・パネル・インターフェースを備えています。

AUTOMAN およびフルスクリーン・パネル・インターフェースを利用することにより、以下のタスクを行えるようになります。

- 単一または複数の自動化テーブルを表示および管理する
- 個々の自動化テーブルまたはステートメントを使用可能または使用不可にする
- 既存テーブルとその状況を表示する

始めに

AUTOMAN では個別テーブル・コマンドおよびグローバル・コマンドが使用できます。個別テーブル・コマンドは 1 つ以上の選択されたテーブルに適用され、グローバル・コマンドはすべての自動化テーブルに適用されます。それぞれのコマンド・タイプごとの以下のフィーチャーおよびオプションを参照してください。

- 個別テーブル・コマンドを使用して、自動化テーブルを使用可能または使用不可にすることができます。また、自動化テーブル・ステートメントを、以下に基づき使用可能または使用不可にできます。
 - シーケンス番号
 - ラベル
 - 終了ラベル
 - ブロック
 - グループ
 - 組み込み

個別テーブル・コマンドを使用して、以下の要求を発行することもできます。

- 使用不可ステートメントの表示
- ラベル、ブロックおよびグループの表示
- テーブルのロードまたはアンロード
- テーブルのテスト

- %INCLUDE 構造体の表示
- 同義語の表示
- グローバル・コマンドを使用して、自動化テーブルを使用可能、使用不可、またはアンロードにすることができます。ユーザーは使用不可のステートメントを使用可能にしたり、またはブロック、グループおよびラベルを使用可能にしたり、使用不可にすることができます。グローバル・コマンドは、すべての自動化テーブルに影響を及ぼします。

自動化ステートメントは、以下のフィーチャーに基づきテーブル全体にわたって使用可能または使用不可にします。

- ラベル
- ブロック
- グループ

グローバル・コマンドを使用して、以下のタスクの要求を発行することもできます。

- 使用不可ステートメントの位置指定
- ラベル、ブロックおよびグループの表示
- %INCLUDE 構造体の表示

自動化テーブル管理の使用法

AUTOMAN コマンドを使用する際は、本セクションの各ステップとパネル説明に従ってください。

コマンド行から、**AUTOMAN** を入力します。図 75 に示すパネルが、表示されます。このパネルにより、ユーザーの自動化テーブル構造を確認し、必要に応じてアクションを実行することができます。

EZLK8500
Automation Table Management

AUTOMATION TABLE			Enter any character in the selection fields				
SEL	POS	NAME	STATUS	MARKERS	TASK	DATE	TIME
-	1	DISTABLE	ENABLED		NETOP2	03/18/11	13:15:24
-	2	DSITBL01	ENABLED	(AON)	NETOP2	03/18/11	13:11:09

Command ==>

F1=Help	F2=Main Menu	F3=Return	F4=Commands	F5=Refresh	F6=Roll
F7=Backward	F8=Forward	F9=Responses	F10=Global Commands	F12=Cancel	

図 75. 自動化テーブル構造

前図には、すべてのロード済みテーブルの状況が表示されています。本図での各フィールドの説明は、以下のとおりです。

SEL

任意の文字をこのフィールドに入力します。複数のテーブルを選択して、処理す

ることができます。カーソルが 288 ページの図 75 に示す選択フィールドにあり、**F4** または **Enter** (キー) を押すと、COMMANDS ポップアップが表示されます。スラッシュ (/) が、ユーザーの参照のために SEL フィールドに自動的に置かれます。

POS

各テーブルの数値的位置を表示します。

NAME

ロード済みテーブルの名前が入っています。

STATUS

以下の状況の 1 つがこのフィールドに表示されます。

ENABLED

テーブルはロードされ、アクティブです。この選択は緑です。

DISABLED

テーブルはロードされているが使用不可です。この選択は赤です。

ALTERED

テーブルはロードされていて使用可能ですが、使用不可ステートメントが少なくと 1 つ含まれています。この選択は黄色です。

MARKERS

ユーザーがそれぞれのテーブルに指定したマーカを表示します。このフィールドには、テーブルに指定したマークに従って (FIRST)、(LAST)、または (AON) が示されます。このフィールドのインディケータは、AUTOMAN により設定されます。

TASK

テーブルをロードしたタスクの名前です。

DATE

テーブルがロードされた日付です。

TIME

テーブルがロードされた時刻です。

選択されたテーブルに対するコマンドの使用

290 ページの図 76 に示すコマンド・ポップアップでは、1 つ以上の選択済み自動化テーブルを処理する際にユーザーを支援するオプションが備えられています。以下の図において (294 ページの図 79 に示すすべてのテーブルに適用されるグローバル・コマンドと対比して)、オプション 1 から 6 は、1 つ以上の選択済みテーブルに適用されます。オプション 7 から 8 は、1 つのテーブルにのみ適用されます。

288 ページの図 75 に示されている「Automation Table Management」パネルで **F4** を押すと、「Commands」ポップアップ・メニュー (290 ページの図 76) が表示されます。ここで、DSITBL01 を使用不可に選択するとします。

オプション 2 を選択すると、ポップアップ・メニューが表示されます。ここで、選択済みテーブルを使用不可にしてよいかどうかを確認できます。DSITBL01 が使用不可になると、メッセージに、そのコマンドが正常に処理されたのか、あるいは障害が検出されたのかが示されます。288 ページの図 75 において **F9** を押すと、使

用したコマンドの結果が表示されます。

EZLKATBC

AUTOMATION TABLE MANAGEMENT

AUTOMATION TABLE

Enter any character in the selection fields

SEL POS NAME

.....

/ 1 DSITBL0

: COMMANDS (Choose a highlighted command option)

:

:

: - 1 -ENABLE the selected tables

: - 2 -DISABLE the selected tables

: - 3 -RELOAD the selected tables

: - 4 -RELOAD and REINSTATE disabled elements

: - 5 -TEST the selected tables

: - 6 -ENABLE/DISABLE parts of the selected

: tables

: - 7 -UNLOAD selected tables

: - 8 -INSERT a table

: - 9 -DISPLAY options

:

: Enter=Execute Command F3 or F12=Cancel

:

Command ==>

F1=Help F2=Main Menu

F6=Roll

F12=Cancel

図 76. 自動化テーブル管理コマンドのポップアップ

以下のコマンドが、図 76 に示している **COMMANDS** ポップアップ・メニューで使用可能です。

- 1 選択済みテーブルを使用可能にする。
- 2 選択済みテーブルを使用不可にする。
- 3 選択済みテーブルを再ロードする。
- 4 選択済みテーブルを再ロードして、すべての使用不可になっているエレメントを復元する。
- 5 選択済みテーブルをテストする。
- 6 選択済みテーブルの一部を使用可能または使用不可にする。
- 7 選択済みテーブルをアンロードまたは除去する。
- 8 (現在の選択済みテーブルに基づいて) 複数の新規テーブルを挿入できるパネルを表示する。291 ページの図 77を参照してください。
- 9 自動化テーブルで他の表示オプションが使用できるパネルを表示する。293 ページの図 78を参照してください。

自動化テーブルの挿入: 図 76 で **8** を選択すると、以下のパネルが表示されます。ここで、新規自動化テーブルを挿入できます。

EZLKATBI
AUTOMATION TABLE MANAGEMENT

AUTOMATION TABLE INSERT PANEL (Press Enter to process INSERT request)

Preceding Table	Focus Table AT= 1	Next Table
N/A	DSITBL01	N/A

SELECT INSERT OR TEST OPTION

1 - AT(DEFAULT)
2 - AFTER
3 - BEFORE
4 - REPLACE
5 - FIRST
6 - LAST
または
7 - AUTOTBL TEST

Table Name
Listing Name AUTOM819

(Required)
Default listing name
(names can be reused,
but cannot be in use
by another table)

SELECT A MARKER OPTION

- Enter your own marker
- Mark as AON's table

Command ==>
F1=Help F3=Return
F6=Roll
F12=Cancel

図 77. 自動化テーブル管理の挿入オプション

自動化テーブルの INSERT オプションは、290 ページの図 76 で選択した INSERT コマンドと 288 ページの図 75 で選択された focus table に基づいてテーブルを挿入する際に使用します。

挿入パネルでは、フォーカス・テーブルの名前とその数値位置を表示します。フォーカス・テーブルの左側に Preceding Table の名前が、またその右側に Next Table の名前がきます。これらの位置にテーブルがない場合は、N/A が表示されます。この情報を使用して、INSERT オプションを以下のように指定することができます。

1 - AT

新規テーブルをフォーカス・テーブルと同じ位置に挿入します。

そのフォーカス・テーブルは、次の位置に移動します。フォーカス・テーブルが FIRST とマークされている場合は、AT オプションを使ってテーブルを挿入することはできません。

2 - AFTER

新規テーブルをフォーカス・テーブルの次の位置に挿入します。

フォーカス・テーブルが LAST とマークされている場合は、AFTER オプションを使ってテーブルを挿入することはできません。

3 - BEFORE

新規テーブルをフォーカス・テーブルの前に挿入します。この要求は、デフォルト AT と同じ結果になります。

4 - REPLACE

フォーカス・テーブルを新規テーブルで置き換えます。

この機能は、290 ページの図 76 にある RELOAD オプションと同じ結果になります。

フォーカス・テーブルが FIRST または LAST とマークされている場合は、両方のテーブルが同じ名前を持っていない限り、REPLACE オプションを指定することはできません。

5 - FIRST

新規テーブルを挿入し、それに FIRST とマークを付けます。

別のテーブルが FIRST とマークされているか、または現行フォーカス・テーブルが位置 1 に置かれた最初のテーブルでない場合には、このオプションを指定することはできません。

6 - LAST

新規テーブルを挿入し、それに LAST とマークを付けます。

別のテーブルが LAST とマークされているか、または現行フォーカス・テーブルがリストされた最後のテーブルでない場合は、このオプションを指定することはできません。

7 - AUTOTBL TEST

指定されたテーブル名で AUTOTBL テストを実行します。

そのテーブルはロードされません。他の挿入フィールドは無視されます。

291 ページの図 77 では、以下のフィールドに注意する必要があります。

Table Name

挿入する自動化テーブルの名前 (1 から 8 文字) を入力します。

Listing Name

リスト・メンバーの固有 ID。自動化テーブル管理を行う場合に必要です。

指定された自動化テーブルのリスト・メンバーの名前 (1 から 8 文字) を入力します。固有のリスト・メンバー名はデフォルトが提供されますが、オーバーライドできます。リスト名は再使用できますが、現在使用中でないことが必要です。

マーカーまたは ID を入力するには、**Enter your own marker** フィールドに任意の文字を使用します。マーカーのテキストを、そのフィールドに続くスペースに入力します。カスタム・マーカーか AON マーカーのいずれかを選択できますが、両方は選択できません。AON によって使用されているテーブルは、正しくマークしておかなければなりません。AON が存在していない場合は、Mark as AON's table は表示されません。

すべてのフィールドが完了し、新規テーブルを挿入できる状態であれば、**Enter** を押してください。

表示オプションのポップアップ・ウィンドウの使用法: COMMANDS ポップアップ・メニューから 9 を選択すると (290 ページの図 76 を参照)、次の DISPLAY OPTIONS ポップアップ・ウィンドウが表示されます。

EZLKATLD		AUTOMATION TABLE MANAGEMENT	
AUTOMATION TABLE		Enter any character in the selection fields	
SEL	POS NAME		
/	1 DSITBL01	: COMMANDS (Choose a highlighted command option) :	
		:	
		: 9 1 -ENABLE the selected tables :	
			
		: DISPLAY OPTIONS (select option and press ENTER) :	
		:	
		: 1 - Browse table with includes 8 - Browse listing :	
		: 2 - Browse table without includes :	
		: 3 - Display INCLUDE structure :	
		: 4 - Display synonyms :	
		: 5 - Display labels/blocks/groups :	
		: 6 - Display disabled statements :	
		: 7 - Display AUTOCNT statistics :	
		:.....:	
Command ==>			
F1=Help		F2=Main Menu	F3=Return
		F6=Roll	F12=Cancel

図 78. 自動化テーブル管理の表示オプションのポップアップ・ウィンドウ

図 78 において、DISPLAY 機能は単一の自動化テーブルに対してのみ作用します。このパネルでのテーブルは、288 ページの図 75 で選択されたものです。DISPLAY OPTIONS パネルでは、以下のオプションを選択することができます。

- 1 選択済みテーブルに対してデフォルト XINCL オプションを指定して、BROWSE コマンドを起動します。
- 2 選択済みテーブルに対して NOINCL オプションを指定して、BROWSE コマンドを起動します。
- 3 %INCLUDE 構造体 (WINDOW コマンドを使用した) が表示されます。INCLUDE レベルはそれぞれ、字下げされ、色分けされています。WINDOW コマンドの詳細については、NetView オンライン・ヘルプを参照してください。
- 4 オプション 3 と同じ %INCLUDE 構造体を組み込み済み同義語とともに表示します。
- 5 ラベル、ブロック、およびグループの名前のみが表示されている新規パネルを表示します。
この新規パネルにより、追加の機能を使用可能や使用不可にすることができます。
- 6 296 ページの図 81 に示すパネルを表示しますが、選択済みテーブル内の使用不可のステートメントだけを表示します。
- 7 WINDOW での次のコマンドの結果を表示します。
AUTOCNT REPORT=BOTH,STATS=DETAIL,NAME=(selected table)
- 8 選択済みテーブル・リスト・ファイルに対して、BROWSE コマンドを起動します。

グローバル・コマンドの使用法

グローバル・コマンドはすべてのテーブルに適用されます。グローバル・コマンドを使用するには、288 ページの図 75 のパネルで **F10** を押します。以下のポップア

ップ・メニューが表示されます。

EZLKATBG

AUTOMATION TABLE MANAGEMENT

AUTOMATION TABLE

Enter any character in the selection fields

SEL	POS	NAME	
-	1	DSITBL01	: GLOBAL COMMANDS (Choose a command option) :
			:
			: 1 -Enable all tables :
			: 2 -Disable all tables (must confirm) :
			: 3 -Unload all tables (must confirm) :
			: 4 -Display options :
			:
			: Enter=Execute Command F3 or F12=Cancel :
			:.....:

Command ==>

F1=Help F2=Main Menu

F6=Roll
F12=Cancel

図 79. 自動化テーブル管理グローバル・コマンドのポップアップ

GLOBAL COMMANDS ポップアップ・メニューが表示されると、以下のオプションが選択できます。

- 1 すべてのテーブルをオンにします。
- 2 すべてのテーブルをオフにします。(注 2 を参照。)
- 3 メモリーからすべてのテーブルを除去します。(注 2 を参照。)
- 4 GLOBAL DISPLAY OPTIONS ポップアップを表示します。

このポップアップ・メニューから選択された任意のオプションは、すべてのテーブルに適用されます。

使用上の注意:

1. SEL フィールドに入れた文字は、GLOBAL COMMANDS ポップアップ・ウィンドウが表示されているときは無視されます。
2. すべての自動化テーブルを UNLOAD または DISABLE にするグローバル・コマンドを選択すると、ユーザーがそれらのテーブルを使用不可にしたいのか、除去したいのか、あるいは操作をキャンセルしたいのかどうかを尋ねる確認パネルが表示されます。詳しくは、「299 ページの『確認パネル』」を参照してください。

グローバル表示パネルの使用法: 図 79 で、選択オプション 4 を選択すると、以下の GLOBAL DISPLAY OPTIONS ポップアップ・ウィンドウが表示されます。

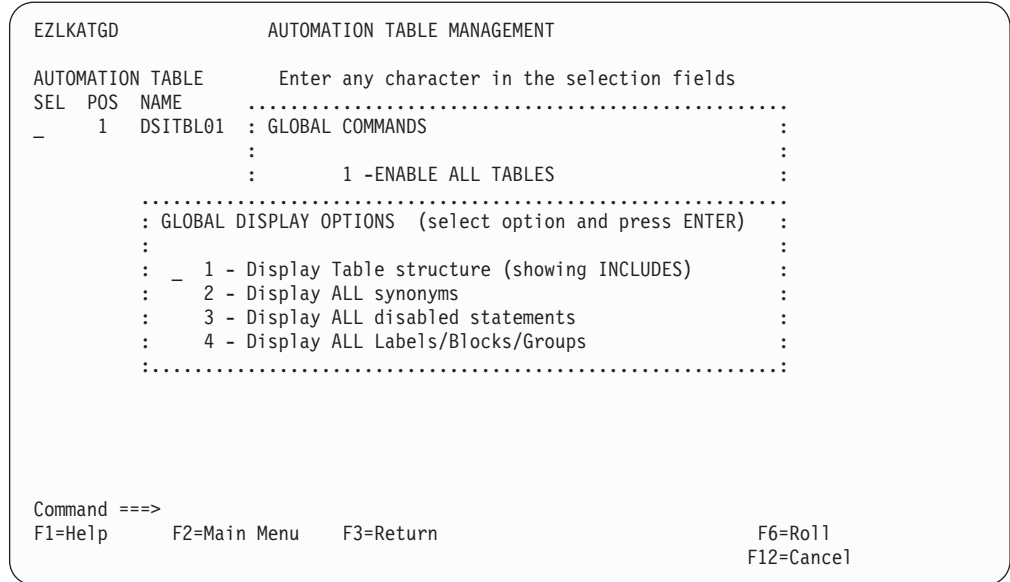


図 80. 自動化テーブル管理のグローバル表示オプションのポップアップ

上記のパネルで、以下のオプションを選択できます。

- 1 %INCLUDE テーブル構造体を WINDOW コマンドを使って表示します。
それぞれのネスティング・レベルは、別の色と字下げによって表示されます。 AUTOTBL コマンドによってロードされる各テーブルは、1 列目に表示され、テキスト *primary table* が続きます。
- 2 %INCLUDE テーブル構造体とその同義語を各レベルごとに表示します。
- 3 296 ページの図 81 に示すパネルを表示しますが、使用不可のステートメントだけを表示します。
- 4 すべてのテーブルに関するラベル名、グループ名、ブロック名を表示します。

表示されたパネルで、すべての自動化テーブルにおいてこれらの名前を使用可能または使用不可にすることができます。このパネルから任意のラベル、グループ、またはブロックを使用可能や使用不可にすると、結果としてグローバル・アクションになります。

自動化テーブル・ステートメントの使用可能化および使用不可化: 296 ページの図 81 において、各コマンドと情報が AUTOCNT STATISTICS とリストを使用して収集されています。このパネルでは、すべてのロードされた自動化テーブルに作用するグローバル表示機能が使用できます。

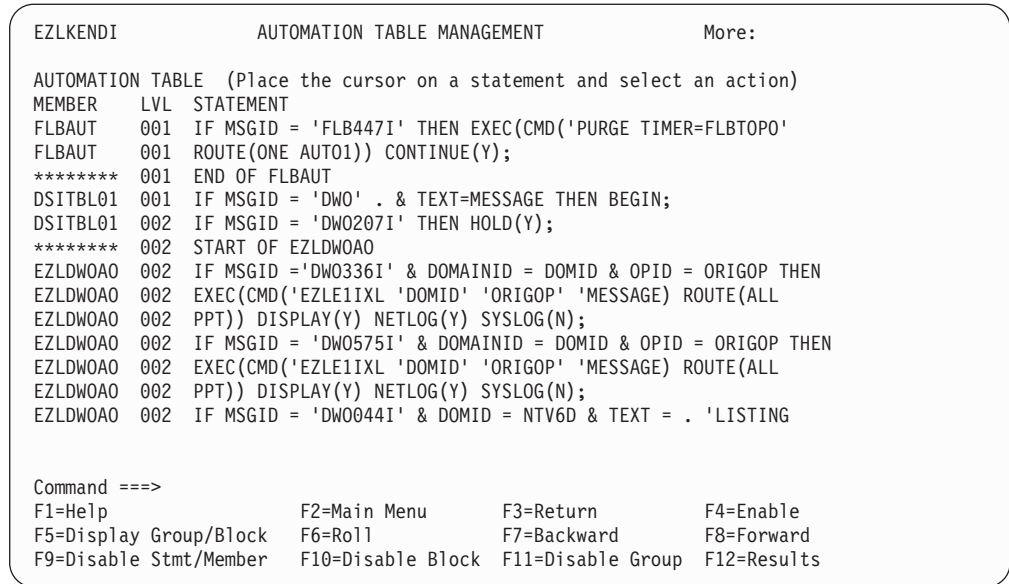


図 81. 自動化テーブル管理の ENABLE/DISABLE パネル

注: このパネルに GLOBAL DISPLAY OPTIONS または DISPLAY OPTIONS (6) ポップアップ・ウィンドウから入ると、許可されないステートメントだけが表示されます。

図 81 において、以下のデータが表示されます。

MEMBER

自動化テーブルの名前で、続いてステートメントが入っています。

STATEMENT

自動化テーブル・リスト・メンバーから検索された自動化テーブル・ステートメント。

各ステートメントの状況は、別々の色で以下のように表示されます。

- 緑** ステートメントは使用可能です。
- 赤** ステートメントは使用不可です。
- 青** ステートメントにラベルもシーケンス番号もないので、ステートメントを個別に使用不可にすることはできません。
- ピンク** このステートメントは、それ自体は使用不可ではありませんが、使用不可のテーブル、%INCLUDE、あるいは開始ブロックの結果として使用できない状態になっています。
したがって、ピンクのステートメントを活動化するには、このステートメントが使用不可となる原因となった先行の赤いステートメント上にカーソルを置き、ステートメントを活動化する必要があります。

このパネルを使用して自動化テーブル・ステートメントを使用可能または使用不可にするには、ユーザーはアクションを行いたいステートメントにスクロールし、以下のファンクション・キーの 1 つを選択します。

- F4** どのステートメントがカレント・フォーカスであり、またそのステートメントを使用不可にする方法に基づき、ラベル、ブロック、またはグループを使用可能にします。
- F5** グループとブロックを表示します。
- F9** ステートメントまたはメンバーを使用不可にします。緑で表示されているステートメントは使用不可にすることができます。
- F10** ブロックを使用不可にします。この機能には、ユーザーの選択が対応する ENDLABEL ステートメントを持つ LABEL ステートメントであることが必須です。
- F11** グループを使用不可にします。この機能には、ユーザーの選択がグループ・ラベルを含むステートメントであることが必須です。
- F12** 発行された最新のコマンドの記述とそれらのコマンドの個別の結果を表示します。

ENABLE/DISABLE (使用可/使用不可) パネルを検索して、特定のステートメント、グループ、ブロック、またはラベルを調べることができます。検索するには、以下のコマンドを使用します。コマンドの省略形を括弧内に示しています。

FIND *anytext* (**F** *anytext*)

ユーザー指定のテキストを検索します。

NEXT TAG (**NT**)

次のグループ、ブロック、またはラベルを検索します。

NEXT IDENTIFIER (**NI**)

次のグループ、ブロック、ラベル、またはシーケンス番号を検索します。

NEXT GROUP (**NG**)

次のグループを検索します。

NEXT BLOCK (**NB**)

次のブロックを検索します。

NEXT SEQUENCE (**NS**)

次のシーケンス番号を検索します。

NEXT ENABLED (**NE**)

次の使用可能ステートメントを検索します。

NEXT DISABLED (**ND**)

次の使用不可ステートメントを検索します。

検索は、ユーザーのカーソル位置から開始します。カーソルは、検索ターゲットが見つけられた行に置かれます。次ページまたは前ページにページ送りする前に別の検索が指定されると、検索は直前の検索ターゲットの後ろから開始されます。

注: ***** のメンバー名は、組み込みメンバーの開始または終了を指します。

Labels/Blocks/Groups パネルの表示: 298 ページの図 82 は、296 ページの図 81 で **F5** を選択した場合に表示されるポップアップ・ウィンドウを図示しています。以下のパネルにおいて、ユーザーのカーソルを Label/Block/Group 上に置き、それを直接に使用可能または使用不可にすることができます。

EZLKATGB		AUTOMATION TABLE MANAGEMENT		
MEMBER	TYPE	LABEL/BLOCK/GROUP NAME	STATUS	NUMBER OF STATEMENTS
-----	----	-----	-----	-----
DISTABLE	LABEL	BOB	ENABLED	1
DISTABLE	LABEL	BOB2	ENABLED	1
DISTABLE	LABEL	JIM	ENABLED	1
DISTABLE	LABEL	NITE2	ENABLED	1
DISTABLE	LABEL	STEVE	ENABLED	1
DISTABLE	BLOCK	NITE	ENABLED	8
DISTABLE	GROUP	KAT	DISABLED	3
COMMAND ==>				
F1=Help		F2=Main Menu	F3=Return	F4=ENABLE
F5=DISABLE		F6=Roll	F7=Backward	F8=Forward
F9=Toggle Display				F12=Cancel

図 82. 自動化テーブル管理の Label/Block/Group パネル

293 ページの図 78 の表示パネルから前のポップアップに入ると、ユーザーのアクションは、単一自動化テーブルにのみ影響を及ぼします。 295 ページの図 80 の表示パネルからこのポップアップに入ると、ユーザーのアクションはすべての自動化テーブルの名前付きラベルに同様に影響を及ぼします。

図 82 には、以下の情報が表示されます。

MEMBER

自動化テーブル・メンバーの名前で、続いてラベル、ブロック、またはグループが入っています。

TYPE タイプがラベル、ブロック、またはグループであるかどうかを示します。

LABEL/BLOCK/GROUP NAME

ラベル、ブロック、またはグループの名前を示します。

STATUS

以下のいずれかの状況が入っています。

- ENABLED (ラベル、ブロック、グループが使用可能にされている。)
- DISABLED (ラベル、ブロック、グループが使用不可にされている。)

NUMBER OF STATEMENTS

ラベル、ブロック、またはグループ内にあるステートメントの数を示します。

ラベルは、常に 1 です。この数値は、グループおよびブロックの場合には異なります。

注: NUMBER OF STATEMENTS 列は、ラベルまたはグループの場合、影響のあるステートメントをすべて含んでいるとは限りません。表示されている合計は、そのラベルまたはグループを含むステートメントの数です。例えば、ラベル付きステートメントが BEGIN で終了していれば、BEGIN と END のブロック内にあるステートメントはカウントされません。すべての影響のある自動化テーブルのステートメントを表示するには、296 ページの図 81 に示す ENABLE/DISABLE パネルを使用してください。

LABEL/BLOCK/GROUP ポップアップにあるファンクション・キーを使用して、以下のアクションを使用可能にすることができます。

- F4** 選択されたラベル、ブロック、またはグループを使用可能にします。
- F5** 選択されたラベル、ブロック、またはグループを使用不可にします。
- F9** ラベル、ブロック、またはグループだけが一度に表示されるように表示域を切り替えます。
- F12** ユーザーのアクションによって発行されたコマンドとそのコマンドの結果を表示します。

注: 295 ページの図 80 からこのポップアップ・ウィンドウをアクセスすると、F4 および F5 の機能は、すべての自動化テーブルに影響するグローバルな使用可能/使用不可機能になります。例えば、グループ NITEOPS が 2 つのテーブルで定義されている場合に、F5 を押すと両方のテーブルにあるそのグループが使用不可になります。

確認パネル

AUTOMAN によって提供される機能の多くは、該当するテーブルを処理する際に自動化テーブル・リスト・ファイルが必須になります。リスト・ファイルがテーブルに関するアクションの要求に続いて利用不可の場合、警告メッセージが表示されます。警告パネルで **F4** を押し、ユーザーの要求を完了するように AUTOMAN が必要に応じてリスト・ファイルを作成するのを許可します。

リスト・ファイルを作成すると、関係のある自動化テーブルを再ロードしなければなりません。それらのテーブルを再ロードすると、以下の状態をリセットします。

- 現行のテーブル、日付、および時間をロードしたオペレーター ID
- AUTOCNT の統計
- インターバルとしきい値の条件項目のカウント
- テーブル内で現在使用不可になっているすべてのステートメント

前にリストされた条件のいずれかがユーザーの環境に悪影響を及ぼす場合は、**F12** を押してリスト要求をキャンセルしてください。

第 16 章 ポリシー・サービスの概要

NetView ポリシー・サービスは、動的なポリシー・ベースの管理およびユーザーのリソースの自動化を可能にする一連の機能です。複数の NetView 機能でポリシー・サービスを活用します。リソースに対してアクションを行う前に、これらの機能は、ポリシー定義を使用して、行うべきアクション (必要であれば) を決定します。

NetView ポリシー・サービスを使用して、独自のポリシー・ベースのアプリケーションを作成できます。このセクションには、独自のポリシー・アプリケーションを作成して、NetView ポリシーを管理する情報を記載しています。

NetView ポリシー・サービスは、以下のものから構成されています。

ポリシー・リポジトリ

ユーザーのポリシー用のストレージの永続データ・ストア

ポリシー API

ポリシー定義の照会、変更、追加、削除、またはロードを行うためにポリシー・リポジトリにアクセスできるようにする機能の集合。

NetView ポリシー・サービスを使用するどのアプリケーションにおいても、ポリシー・リポジトリにロードされるポリシー定義ならびにポリシーを解釈して、適切なアクションを取るためのアプリケーション・コードを用意する必要があります。

NetView プログラムには、NetView ポリシー・サービスを使用する、以下の 2 つのアプリケーションが組み込まれています。

ネットワーク管理コンソール (NMC)

NMCSTATUS ポリシー定義に基づいて、(NetView 管理コンソール・ビューのリソースのための) タイム・スケジュールを定義する機能。このようなスケジュールを使用して、以下の場合に指定する、ビューに対してポリシーを適用できます。

- ビューの中の 1 つ以上のリソースの表示可能状況を、NetView 管理コンソールで使用不可にする。
- ビュー内の 1 つ以上のリソースが、集合体から一時的に停止されている。

Automated Operations Network (AON)

ポリシー定義に基づいたネットワーク・リソースの自動操作を可能にします。

これらのアプリケーションは、それぞれポリシー・リポジトリのデフォルト・ポリシーを定義してから、そのポリシーを解釈し、そのポリシーに基づいて適切なアクションを行います。NetView 管理コンソールおよび AON に組み込まれているデフォルト・ポリシー・ステートメントは変更可能です。NetView 管理コンソールおよび AON が使用するすべてのポリシー定義については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。本章では、ポ

リシー・サービスのインストール、ポリシー・ファイル・ステートメントの構文、ポリシー・リポジトリへのポリシー・ファイルのロード方法、およびポリシーの管理方法について説明しています。

ポリシー・サービスの使用

NetView ポリシー・サービスを使用するには、以下のことを行います。

- DSITBL01 のカスタマイズ
- ポリシーの定義
- ポリシー・ファイルの定義

DSITBL01 のカスタマイズ (オプション)

DSITBL01 をカスタマイズするには、以下のステップを実行します。

1. メッセージ EZL110I に関連したステートメントを変更します。

NetView プログラムには、EZL110I メッセージを受信するたびに、コンポーネント (AON および NetView 管理コンソール) を再同期するステートメントが組み込まれています。

2. オプションで、他の任意のポリシー・ベースのアプリケーションに対して適宜ステートメントを追加します。

注: DSITBL01 サンプル自動化テーブルには、NetView プログラムを正しく操作するために不可欠なステートメントが多数入っています。DSITBL01 サンプルの全部または一部を変更あるいは削除する際は、十分に注意してください。

ポリシー・ファイルの定義

CNMSTYLE メンバーには、ポリシー定義ファイルを定義できるようにするステートメントが含まれています。これらのステートメントは、ポリシー・リポジトリをロードまたは再ロードするたびに使用されます。CNMSTYLE メンバーの詳細については、「*IBM Tivoli NetView for z/OS*インストール:概説」に記載されています。

1 つ以上の個別ポリシー・ファイルを定義できます。これらのファイルはマージされ、1 つの論理ファイル名としてロードされます。デフォルトの論理ファイル名は NVPOLICY です。この場合のステートメントは、以下のとおりです。

POLICY.&domain = NVPOLICY

ポリシー・リポジトリをロードする際に使用する論理ファイル名を定義します。

POLICY.xxx = file_name

ポリシー定義を含む DSIPARM 内の実ファイル名を定義します。必要に応じて、このようなステートメントを 1 つ以上指定できます。"xxx" は、ポリシー・アプリケーションごとに定義されているとおりの任意の文字セットにできます。NetView プログラムには、以下の 2 つのポリシー・ファイル名が含まれています。

POLICY.AON = EZLCFG01 for AON

POLICY.GRAPHICS = DUIPOLCY for NMC status

必須 NetView タスク

初期設定中に、NetView は、CNMSTYLE メンバーの POLICY ステートメントに基づく定義済みのポリシー・ファイルのあるポリシー・リポジトリをロードします。ポリシー・リポジトリは、EZLTCFG タスクがアクティブ状態である間、アクティブでストレージにロードされた状態です。

EZLTCFG は、毎回、NetView により初期設定する必要があります (INIT=YES)。NetView のインストール時に、アクティブ・タスクをカスタマイズするステップがありました。EZLTCFG が当該リストに含まれていることを確認してください。

ポリシー・ファイル構文

ポリシー・リポジトリは、オープンでフレキシブルであるため、アプリケーションは、最小限のコーディングを行うだけでこのポリシー・リポジトリを使用できます。

ポリシー・ファイルは、DSIPARM になければならず、また CNMSTYLE メンバーの POLICY ステートメントにあるファイルを参照しなければなりません。

また、ポリシー・ファイル内の %INCLUDE を使用して、他のメンバーをポリシー・ファイルの一部として組み込むことができます。

いくつかの構文規則があるだけです。どのポリシー構成内の基本構文も、以下の規則に従わなければなりません。

```
Policy_Name Policy_definition, keyword1=value1,  
keyword2=value2,  
keyword3=value3
```

説明：

Policy_Name

ポリシー (RECOVERY または NMCSTATUS など) の名前。これを使用するのは、POLICY 要求で ENTRY= parm と指定されている場合です。この名前は、1 桁目から始めて、1 から 32 文字まででなければならず、組み込みブランク ()、コンマ (,), 単一引用符 (') または二重引用符 (" ")、括弧 (), または等号 (=) は含めません。

注: 独自のポリシーを作成するに当たって、回避する必要があるポリシー名のリストについては、「IBM Tivoli NetView for z/OS アドミニストレーション・リファレンス」の『自動化オペレーション・ネットワーク (AON) 定義』を参照してください。

Policy_Definition

定義対象のポリシー (例えば、HOLIDAY)。これを使用するのは、POLICY 要求で TYPE= parm と指定されている場合です。この名前は、1 から 32 文字まででなければならず、組み込みブランク ()、コンマ (,), 単一引用符 (') または二重引用符 (" ")、括弧 (), または等号 (=) は含めません。

keywordn

keyword=value の組は、ポリシー定義の場合には必須です。キーワードは 1

文字以上で、これに、組み込みブランク (), コンマ (,), 単一引用符 (') または二重引用符 (" "), 括弧 (), または等号 (=) は含めません。

Valuen キーワード値構文は、アプリケーションごとに定義されます。ブランクまたは等号 (=) を含む値は、単一引用符 (') または二重引用符 (" ") 内に含めて使用する必要があります。このような機能には、以下のものがあります。

```
TEXT=A=B    is invalid
TEXT='A=B'  is ok
TEXT="A=B"  is ok
TEXT=A B    is invalid
TEXT="A B"  is ok
```

使用上の注意

- コメントは、1 桁目にアスタリスク (*) を指定してから始めます。

ポリシー定義内には、コメントを組み込まないでください。コメントは、該当するポリシー定義を開始する前に入れてください。

- RECOVERY または NMCSTATUS などのポリシー定義は、1 桁目から始めます。

注: ポリシー・ステートメントは、1 桁目から始める必要があります。ポリシー・ステートメントの継続行は、1 桁目から始めてはいけません。

- 次行への継続がサポートされています。

行を継続するには、現在行をコンマで終了して、次行は 2 桁目以降から始めます。

- テキスト・ストリングまたは括弧を使用したストリングの場合、継続はサポートされていません。
- ポリシー・ステートメントは、1 から 72 桁目までに記述しなければなりません。
- ポリシー・ステートメントに構文エラーが含まれている場合、ポリシー・リポジトリのロードは終了します。
- ユーザー独自のアプリケーションの場合には、Tivoli NetView 事前定義ポリシーは使用しないでください。

Tivoli の NetView ポリシーを変更すると、エラーが起きる可能性があります。詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

例:

ポリシー情報は、アプリケーションで定義され、ポリシー・リポジトリに保管されます。次の AON の RECOVERY ポリシーの例は、ポリシー定義の仕方について示しています。ポリシーの目的を説明するためにコメントを使用しています。

```
* AON RECOVERY Policy for all Physical Units (PUs) :
* automate recovery except from midnight to 6am every day of the week
* Policy_name=RECOVERY
* Policy_definition=PU
* keyword1=AUTO
* value1=Y
```



```

* keyword2=NOAUTO
* value1=(*,00:00,06:00)
  RECOVERY PU,AUTO=Y,NOAUTO=(*,00:00,06:00)

* AON RECOVERY Policy for all "Holidays" :
*   automate recovery except from midnight to 6am every Holiday
* Policy_name=RECOVERY
* Policy_definition=HOLIDAY
* keyword1=AUTO
* value1=Y
* keyword2=NOAUTO
* value1=(*,00:00,06:00)
RECOVERY HOLIDAY,AUTO=Y,NOAUTO=(*,00:00,06:00)

* AON RECOVERY Policy for a resource called NCP10 :
*   automate recovery except from midnight to 2am every Holiday
*   automate recovery except from midnight to 4am on weekends
*   automate recovery except from midnight to 6am on weekdays
* This example shows continuation for lines 2 through 4.
RECOVERY NCP10,AUTO=Y
  NOAUTO=(HOLIDAY,00:00,02:00),
  NOAUTO=(WEEKEND,00:00,04:00),
  NOAUTO=(WEEKDAY,00:00,06:00)

```

キーワード値構文は、アプリケーションで定義することを思い出してください。この場合のアプリケーションは AON です。

AON RECOVERY ポリシーおよびその他の NetView ポリシーについて詳しくは、「*NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

ポリシー・ファイル管理

NetView POLICY コマンドを使用して、ポリシー・リポジトリにロードされるポリシー・ファイルの種類を管理し、このようなポリシー定義に基づいたアクションを実行することができます。

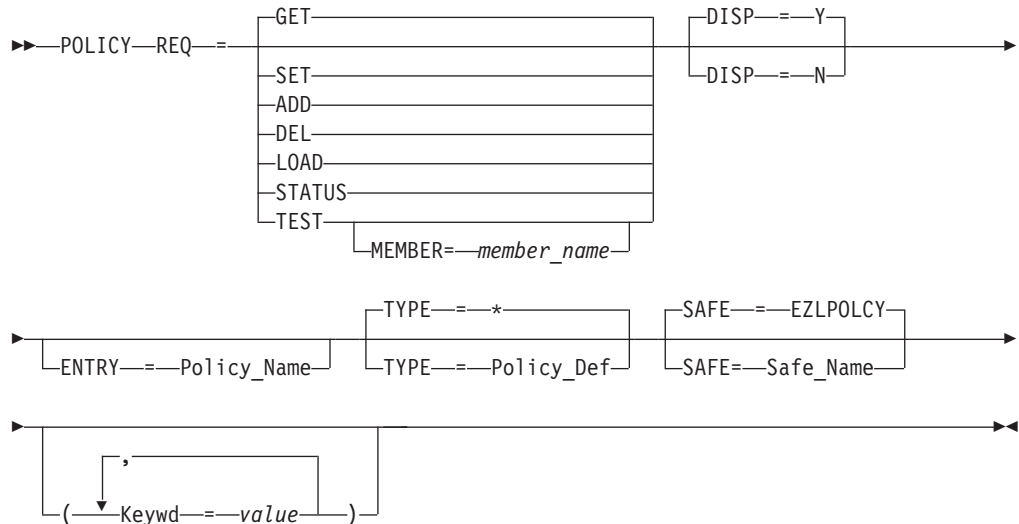
POLICY コマンドは、ポリシー・リポジトリへの多目的で汎用的なアプリケーション・プログラミング・インターフェース (API) です。このコマンドは、ポリシー・リポジトリ内のすべてのポリシー定義への標準化されたアクセスを提供します。

アプリケーションによっては、さらに多くの特定インターフェースを出荷するものもあります。例えば、SETAUTO コマンドを使用して AON RECOVERY ポリシーのみを管理することができます。SETAUTO を使用する際には、他のポリシー定義がロードされても、それらはユーザーには表示されません。アプリケーション固有のインターフェースについては、該当するユーザーズ・ガイドに記載されています。

ポリシー API の使用

POLICY 構文

POLICY



説明：

GET ポリシー・リポジトリから要求されたポリシー定義を取り出します。これがデフォルトです。

SET 要求されたポリシー定義のキーワードの値を更新します。

ADD 指定されたキーワードおよび値を使ってポリシー・リポジトリに新規ポリシー定義を作成します。

DEL ポリシー・リポジトリからポリシー定義を削除します。

LOAD CNMSTYLE メンバーの定義に基づいてポリシー・リポジトリをロードします。

STATUS

ポリシー・リポジトリにロードされているポリシー・ファイルについて照会します。

TEST ポリシー・ファイルの構文チェックを実行します。

MEMBER=member_name

テストする実際のポリシー・ファイルの名前。 **TEST** 要求でなくても指定されていないと、現在アクティブな状態にあるポリシー・ファイルがすべてテストされます。

DISP

Y: 関係のあるメッセージをユーザー・コンソールに表示します。 **Y** がデフォルトです。

N: 関係のあるメッセージをユーザー・コンソールに表示しない オプション。

注: このオプションは、メッセージによる割り込みが入らないように、3270 アプリケーションで使用してください。

ENTRY=*Policy_Name*

RECOVERY または NMCSTATUS などの、有効な任意のポリシー名で、「NetView for z/OS アドミニストレーション・リファレンス」またはその他のアプリケーションで定義されているとおりです。

Type=*

type=* と入力すると、POLICY コマンドは、指定された *Policy_Name* に対するすべてのポリシー定義を戻します。 **Type=*** がデフォルトです。

TYPE=*Policy_Def*

HOLIDAY などの、有効な任意のポリシー名で、「NetView for z/OS アドミニストレーション・リファレンス」またはその他のアプリケーションで定義されているとおりです。

SAFE=*Safe_Name*

要求からの出力を含むセーフの名前。EZLPOLCY がデフォルトです。

keyword

NOAUTO など、ポリシー・アプリケーションで許可されている有効な任意のポリシー・キーワード。

値 ポリシー・アプリケーションで許可されている有効な任意のキーワード値。

戻りコード:

-1	シグナル障害
-5	シグナル停止
0	要求が成功しました
1	要求されたポリシー定義が見つかりません (GET/ADD/SET)
3	パラメーターが欠落しています -- メッセージ EZL203I を参照してください
4	パラメーターが無効です -- メッセージ EZL204I を参照してください
7	シグナルに値がありません -- メッセージ EZL271E を参照してください
8	シグナル構文 -- メッセージ EZL275E を参照してください
9	セキュリティ許可障害 -- メッセージ EZL228E を参照してください
10	要求が処理されませんでした -- 他のエラーが発生しました

使用上の注意

- 1 つ以上の *keyword= value* の組を指定できます。
- **TYPE=*** と指定すると、指定されたポリシー・グループについてのすべてのポリシー定義を取り出すことができます。

例えば、**POLICY REQ=GET ENTRY=RECOVERY TYPE=*** と入力すると、システムはすべてのポリシーからすべての RECOVERY ポリシー定義を戻します。

- REQ=DEL ではキーワードもキーワード値も削除できません。特定のポリシー定義のみ削除することができます。
- REQ=GET ではキーワードもキーワード値も照会できません。特定のポリシー定義のみ照会することができます。
- REQ=STATUS または REQ=LOAD の場合はパラメーターを使用することはできません。
- コマンド行から POLICY と入力すると、DISP= は無視されます。
- REQ=TEST の場合に MEMBER= が指定されていない場合、現在アクティブなポリシー・ファイルについてはすべて、ポリシー・リポジトリにロードされている現行ポリシーに基づいて、構文の検査が行われます。

ロードされるポリシー・ファイルの判別

ポリシー・リポジトリがロードされているかどうか（さらに、ロードされている場合はどのファイルがロードされているのか）を判別するには、次のコマンドを発行してください。

POLICY REQ=STATUS

応答は以下のようになります。

```
EZL005I MEMBER NVPOLICY CURRENTLY BEING USED FOR THE CONTROL FILE
EZL006I NVPOLICY FILE 1 = EZLCFG01
EZL006I NVPOLICY FILE 2 = DUIPOLCY
EZL002I END
```

ここで、NVPOLICY は、ポリシー・リポジトリをロードするために使用する論理ファイル名です。EZLCFG01 および DUIPOLCY は、ポリシー定義をロードする際に NVPOLICY を作成するために使用される実ファイルです。

ポリシー・ファイルの構文テスト

ポリシー・ファイルをロードする前に、それについて構文テストを実施する必要があります。構文テストを実施するには、以下のコマンドを発行してください。

POLICY REQ=TEST, MEMBER=member_name member_name は、DSIPARM 内のポリシー・ファイル名です。このファイルは、変更されたばかりの既存ファイルの場合もあれば、ロードしようとする新規ファイルの場合もあります。

おそらく、ポリシー・ファイル内のいくつかのポリシーには変更が加えられていることでしょう。MEMBER= パラメーターを指定せずに POLICY REQ=TEST を発行すると、現在ロードされているポリシー・ファイルの新規バージョンをテストすることができます。テストが成功したファイルについてはすべて、次のメッセージが表示されます。

```
EZL023I TEST OF CONTROL FILE MEMBER "EZLCFG01" WAS SUCCESSFUL
```

テストが成功しなかったファイルについてはすべて、次のメッセージ、およびエラーを記述したその他の詳細記述メッセージが表示されます。

```
EZL023I TEST OF CONTROL FILE MEMBER "MYPOLICY" WAS UNSUCCESSFUL
```

すべてのファイルのテストが成功すると、ポリシー・リポジトリを再ロードすることができます。『ポリシー・ファイルのロード』を参照してください。

ポリシー・ファイルのロード

ポリシー・リポジトリは、CNMSTYLE メンバーの POLICY ステートメントに基づいて、NetView の初期設定の際にロードされます。NetView プログラムがアクティブであるときに、ポリシー・リポジトリをロードする場合は、POLICY REQ=LOAD コマンドを発行します。次のメッセージが表示されます。

```
EZL110I NVPOLICY BEING USED FOR THE CONFIGURATION TABLE
EZL006I NVPOLICY FILE 2 = EZLCFG01
EZL006I NVPOLICY FILE 2 = DUIPOLICY
EZL002I END
```

ポリシーを再ロードすると、すでにロードされている、これまでのポリシーに加えられた一時的変更は除去されます。

1. ポリシー・リポジトリのロード時および再ロード時には十分に注意してください。

ポリシー・ファイルをロードする前にそのテストを行ってください。 308 ページの『ポリシー・ファイルの構文テスト』を参照してください。

ポリシー・リポジトリを使用するアプリケーションは、ポリシーが再ロードされるたびに再同期する必要があります。

2. ポリシーをロードするタスクを許可します。

ポリシー・リポジトリを使用するアプリケーションは、ポリシーが再ロードされるたびに再同期する必要があります。

3. ポリシー・リポジトリを使用する独自のアプリケーションを作成した場合、以下のステップを行う必要があります。
 - a. EZL110I 自動化ステートメントを検討してください。これらは、ポリシーが再ロードされる際にアプリケーションを再同期させるために使用するものです。
 - b. アプリケーションに同様の機能を用意します。

DSITBL01 により、メッセージ EZL110I を自動化することができます。

ポリシーが正常にロードされたかどうか判別するには、サンプル clist (EZLECKPF) を使用できます。これは NetView プログラムに組み込まれています。

例えば、EZL110I プロセスに以下の行を追加します。

```
'PIPE SAFE * | KEEP EZL110I'      /* save AIFR */
'EZLECKPF ' component              /*ie; AON   */
IF RC <> 0 THEN
  /* HANDLE POLICY NOT LOADED      */
ELSE
  /* POLICY LOADED, CONTINUE        */
```

ポリシー定義の照会

ポリシー定義を照会するには、以下のコマンドを発行します。

POLICY REQ=GET ENTRY=Policy_Name TYPE=Policy_Definition

ポリシーが存在するのであれば、*keyword=value* の組を含む、複数行の応答を受け取ります。例えば、NCP10 用の RECOVERY ポリシーについて照会するには、次のコマンドを発行します。

POLICY REQ=GET ENTRY=RECOVERY TYPE=NCP10

以下に、返される結果の例を示します。

```
EZL115I RECOVERY NCP10 AUTO Y
EZL115I RECOVERY NCP10 NOAUTO (HOLIDAY,00:00,02:00)
EZL115I RECOVERY NCP10 NOAUTO (WEEKEND,00:00,04:00)
EZL115I RECOVERY NCP10 NOAUTO (WEEKDAY,00:00,06:00)
EZL002I END
```

Policy_Name は RECOVERY です。 Policy_Definition は NCP10 です。 4 つの *keyword=value* の組があります。 1 番目の *keyword=value* の組は AUTO=Y です。 2 番目の *keyword=value* の組は NOAUTO=(HOLIDAY,00:00,02:00) です。 3 番目の *keyword=value* の組は NOAUTO=(WEEKEND,00:00,04:00) です。 4 番目の *keyword=value* の組は NOAUTO=(WEEKDAY,00:00,06:00) です。

ポリシー定義のグループの

1 つのコマンドで複数のポリシー定義について照会する場合は、次のコマンドを発行します。

"POLICY REQ=GET ENTRY=TCP390 TYPE=*".

送信者は、次のメッセージを見ます。

```
EZL115I TCP390 DEFAULTS PINGCNT 3
EZL115I TCP390 DEFAULTS PINGRETRY 3
EZL115I TCP390 DEFAULTS PINGLEN 64
EZL115I TCP390 DEFAULTS PINGTIME 10
EZL115I TCP390 DEFAULTS DEBUG 0
EZL115I TCP390 DEFAULTS VERBOSE Y
EZL115I TCP390 DEFAULTS SNMPRETRY 2
EZL115I TCP390 DEFAULTS SNMPTO 3
EZL115I TCP390 DEFAULTS MAXREP 10
EZL115I TCP390 DEFAULTS NONREP 0
EZL115I TCP390 DEFAULTS RPLENGTH 64
EZL115I TCP390 DEFAULTS RPTO 5
EZL115I TCP390 NMPIPL10 IPADDR 9.67.50.52
EZL115I TCP390 NMPIPL10 HIER2 SP-APPL
EZL115I TCP390 NMPIPL10 HIER3 NETSP
EZL115I TCP390 NMPIPL10 DOMAIN LOCAL
EZL115I TCP390 NMPIPL10 UNIXSERV YES
EZL115I TCP390 NMPIPL10 TCPNAME TCP38
EZL115I TCP390 NMPIPL10 FORMAT STACK
EZL115I TCP390 NMPIPL10 SNMP MVS
EZL115I TCP390 NMPIPL10 HOSTNAME NMPIPL10 raleigh.ibm.com
EZL115I TCP390 NMP190 IPADDR 9.67.50.34
EZL115I TCP390 NMP190 HIER2 SP-APPL
EZL115I TCP390 NMP190 HIER3 NETSP
EZL115I TCP390 NMP190 DOMAIN NTV74
EZL115I TCP390 NMP190 UNIXSERV YES
EZL115I TCP390 NMP190 TCPNAME TCP38
```

```
EZL115I TCP390 NMP190 FORMAT STACK
EZL115I TCP390 NMP190 SNMP MVS
EZL115I TCP390 NMP190 HOSTNAME NMP190.raleigh.ibm.com
EZL002I END
```

この場合、ポリシー名は TCP390 です。この照会では、3 つの Policy_Definition 値、DEFAULTS、NMPIPL10 および NMP190 が戻されました。

ワイルドカードがサポートされています。例えば、**POLICY REQ=GET ENTRY=TCP390 TYPE=NMP*** を発行すれば、NMP* についてのみ TCP390 ポリシー定義を取り出します。次のような応答が表示されます。

```
EZL115I TCP390 NMPIPL10 IPADDR 9.67.50.52
EZL115I TCP390 NMPIPL10 HIER2 SP-APPL
EZL115I TCP390 NMPIPL10 HIER3 NETSP
EZL115I TCP390 NMPIPL10 DOMAIN LOCAL
EZL115I TCP390 NMPIPL10 UNIXSERV YES
EZL115I TCP390 NMPIPL10 TCPNAME TCP38
EZL115I TCP390 NMPIPL10 FORMAT STACK
EZL115I TCP390 NMPIPL10 SNMP MVS
EZL115I TCP390 NMPIPL10 HOSTNAME NMPIPL10.raleigh.ibm.com
EZL115I TCP390 NMP190 IPADDR 9.67.50.34
EZL115I TCP390 NMP190 HIER2 SP-APPL
EZL115I TCP390 NMP190 HIER3 NETSP
EZL115I TCP390 NMP190 DOMAIN NTV74
EZL115I TCP390 NMP190 UNIXSERV YES
EZL115I TCP390 NMP190 TCPNAME TCP38
EZL115I TCP390 NMP190 FORMAT STACK
EZL115I TCP390 NMP190 SNMP MVS
EZL115I TCP390 NMP190 HOSTNAME NMP190.raleigh.ibm.com
EZL002I END
```

この場合、照会により 2 つの policy_definitions、NMPIPL10 および NMP190 が戻されました。

ポリシー定義の変更

ポリシー定義を変更するには、1 つ以上のキーワードの値を変更します。ポリシー定義を変更すると、指定したキーワードおよび値が変更され、他のすべてのキーワードおよび値は未変更のままです。ポリシー定義を変更するには、以下のコマンドを発行します。

POLICY REQ=SET ENTRY=Policy_Name TYPE=Policy_Definition keyword=value

NCP10 の場合に、AUTO が N に設定されるように RECOVERY ポリシーを変更するには、次のコマンドを発行します。

POLICY REQ=SET ENTRY=RECOVERY TYPE=NCP10 AUTO=N

次のような応答が表示されます。

```
EZL001I REQUEST "REPL" WAS SUCCESSFUL FOR EZLEPOLY
```

変更内容を検証するには、次のコマンドを発行します。

POLICY REQ=GET ENTRY=RECOVERY TYPE=NCP10

次のような応答が表示されます。

```
EZL115I RECOVERY NCP10 AUTO N
```


ポリシー・リポジトリにロードされるポリシー定義のコピーに対して更新が行われます。DSIPARM の元のポリシー・ファイルは未変更のままです。変更を永久的なものにする場合は、次にポリシーがロードされたときに、その変更内容が失われないように DSIPARM 内の元のポリシー・ファイルを変更してください。

注: 既存のポリシー定義を完全に置き換える場合は、現行のポリシー定義を削除してから、新規ポリシー定義を追加してください。

ポリシー定義の削除

ポリシー要求を削除するには、次のコマンドを発行します。

POLICY REQ=DEL ENTRY=Policy_Name TYPE=Policy_Def

NCP10 の場合に RECOVERY ポリシーを削除するには、次のコマンドを発行します。

POLICY REQ=DEL ENTRY=RECOVERY TYPE=NCP10

コマンドが成功すると、次のような応答が表示されます。

```
EZL001I REQUEST "DEL " WAS SUCCESSFUL
```

ポリシー・リポジトリにあるポリシー・ファイルから削除されます。DSIPARM の元のポリシー・ファイルは未変更のままです。変更を永久的なものにする場合は、DSIPARM 内の元のポリシー・ファイルを変更してください。こうすると、次にポリシーがロードされるたびに、変更内容は失われません。

ポリシー定義の追加

新規ポリシー定義を動的に作成または追加するには、次のコマンドを発行します。

POLICY REQ=ADD ENTRY=Policy_Name TYPE=Policy_Definition keyword=value1 keyword2=value2 ...

MYRES 用に AUTO が yes で、NOAUTO ウィンドウが毎日、真夜中から午前 6 時までというように RECOVERY ポリシーを定義するには、次のコマンドを発行します。

**POLICY REQ=ADD ENTRY=RECOVERY TYPE=MYRES AUTO=Y
NOAUTO=(*,00:00,06:00).**

policy_name は RECOVERY です。policy_definition は MYRES です。1 番目の keyword=value の組は AUTO=Y です。2 番目の keyword1 =value1 の組は NOAUTO=(*,00:00,06:00) です。

次のメッセージが返されます。

```
EZL001I REQUEST "ADD " WAS SUCCESSFUL
```

ここで、ポリシーについて照会して、ポリシー・リポジトリにロードされた内容を参照することができます。ポリシー・リポジトリにロードされているポリシー定義のコピーに対して更新が行われます。DSIPARM の元のポリシー・ファイルは

未変更のままです。変更を永久的なものにする場合は、次にポリシーがロードされたときに、その変更内容が失われないように DSIPARM 内の元のポリシー・ファイルを変更してください。

REXX API の使用

POLICY API を使用すると、複雑なルーチンを作成することができます。詳しくは、306 ページの『ポリシー API の使用』を参照してください。

POLICY API が REXX プロシージャーから呼び出されたときには、その構文は未変更のままです。ポリシー定義を照会 (REQ=GET) しようとする、出力内容は変更されます。応答には、メッセージ ID (EZL115I など) は含まれていません。他のデータはすべて同じです。

例えば、NetView の NCP10 用の RECOVERY ポリシーについて照会するように REXX ルーチンを作成するには、次のコマンドを発行します。

```
/* REXX */

'PIPE NETV I',

'POLICY REQ=GET ENTRY=RECOVERY TYPE=NCP10',

'I STEM POL.' ¥DO I=1 to POL.0

END

-

-

EXIT
```

タイマー API

このセクションでは、NetView のタイマー API を使用可能にする方法について説明します。各セクションごとに、API の概要、API の構文、および API の使用方法の例について説明します。ここには、API を実行したときに受け取ることになる戻りコードもリストします。

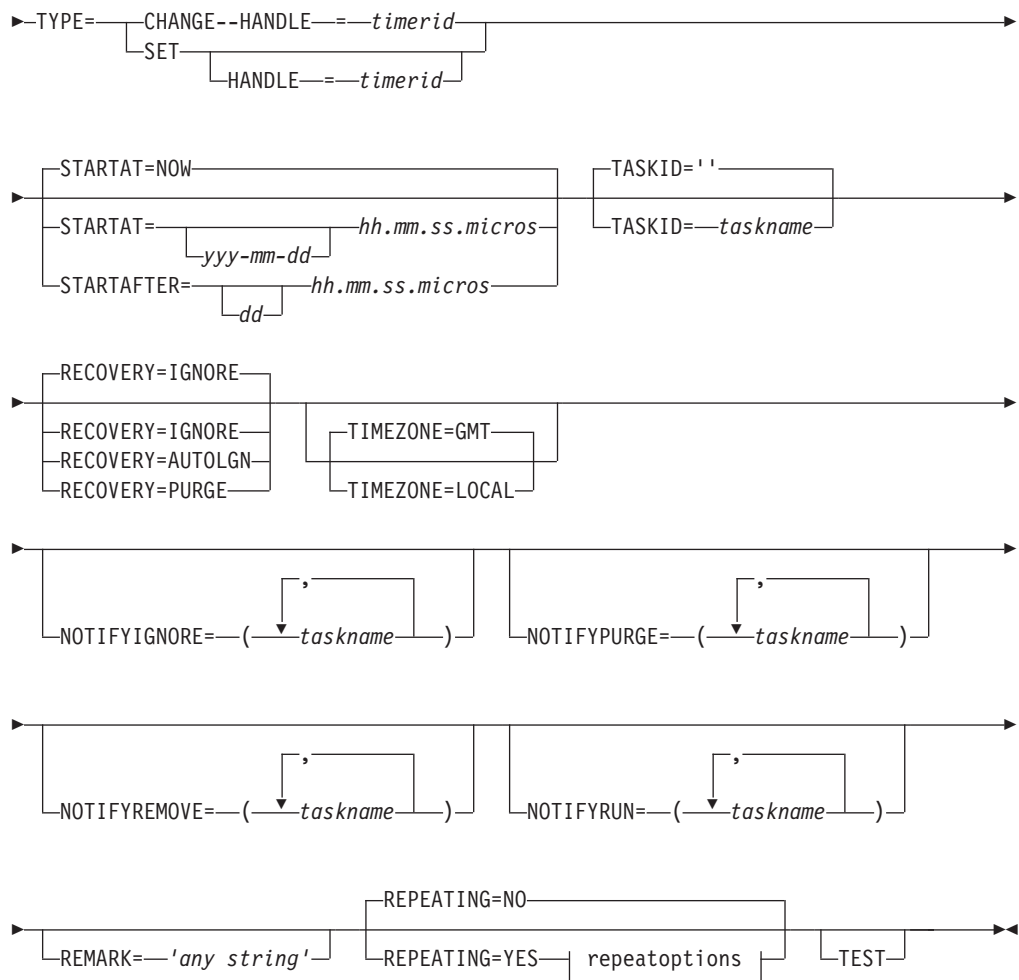
EZLETAPI

EZLETAPI API は、タイマーを設定または変更するようにパラメーターを定義するために使用します。

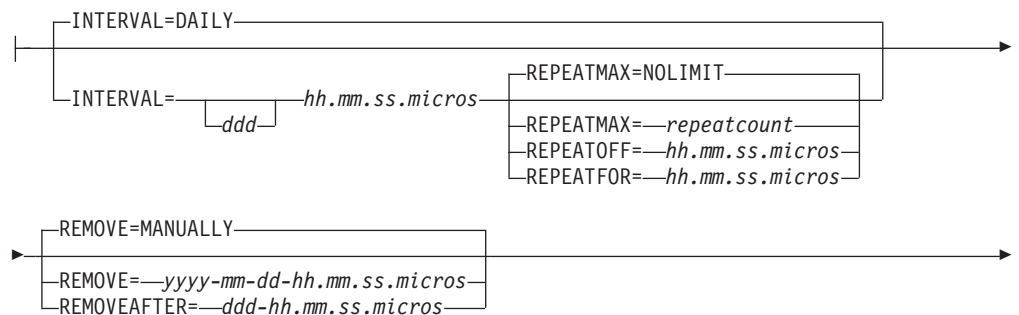
EZLETAPI API の構文は、以下のとおりです。

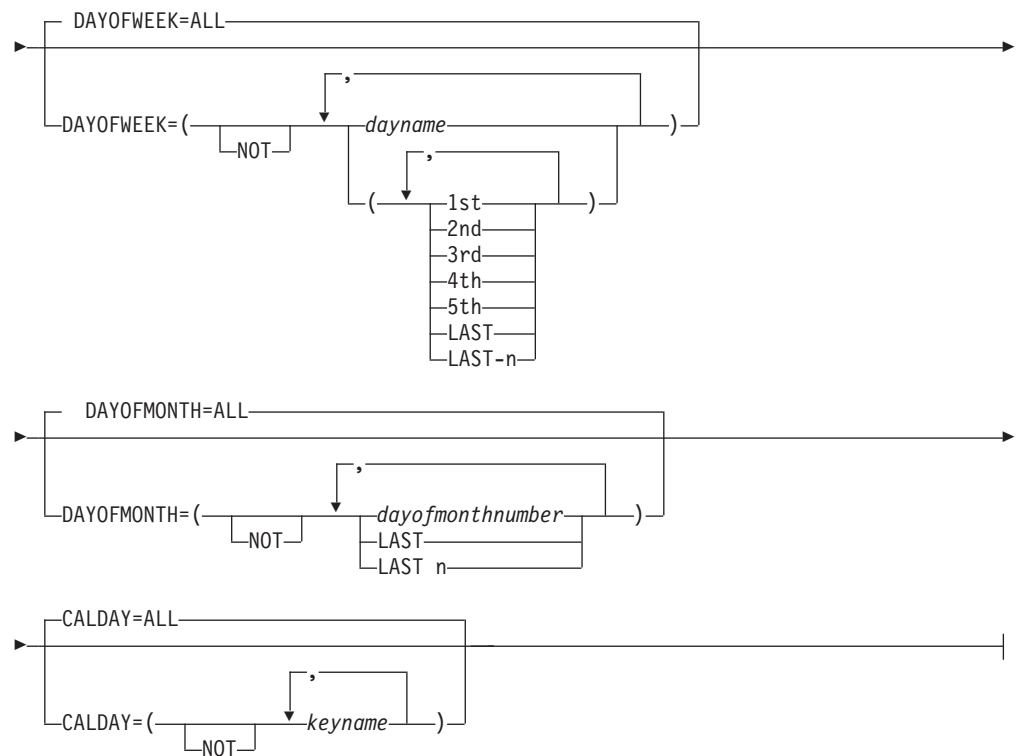
EZLETAPI

→ EZLETAPI { SAFE=EZLTASAF | SAFE=*safename* } ACTION=*'text'* →



repeatoptions:





説明：

SAFE 出力が配置されるセーフの名前です。デフォルトは EZLTASAF です。

ACTION

タイマー・イベントが発生するときに行われるアクション。ストリングは、単一引用符またはアポストロフィで囲まなければなりません。単一引用符もアポストロフィもストリング内に含めることはできません。これらを使用する必要がある場合は、ストリング内でアポストロフィを 2 つ一緒に使用するか、二重引用符を使用してください。

TYPE タイマー要求のタイプを指定します。

Set 新規要求であることを指定します。このオプションを指定して、タイマー・ハンドルを渡すと、そのハンドルを持つタイマーが作成されます。

変更 既存のタイマーへの変更であることを指定します。既存のタイマーを削除するためには、既存のタイマー・ハンドルを渡さなければなりません。

注： TYPE=CHANGE と指定する場合、既存のタイマーで変更されていない パラメーターもすべて、呼び出し時に指定しなければなりません。このようなパラメーターは、EZLEQAPI を使用し、優先パラメーターを変更して取得することができます。すべてのパラメーターを指定しているわけではない場合、タイマーは、指定されたパラメーター、およびパラメーターが指定されていない場合はデフォルトを使用して設定されます。

HANDLE

タイマーの場合は、固有 ID (1 から 8 文字)。ハンドルが指定されていない場合は、固有 ID が生成されます。ハンドルは、ALL、RST、または SYS で始めることはできません。

STARTAT

アクションが開始される日時を指定します。指定されていない場合のデフォルトは、直ちに開始することです。STARTAT および STARTAFTER キーワードは、同時に指定することはできません。

STARTAFTER

アクションが開始されてからの間隔を指定します。STARTAT および STARTAFTER キーワードは、同時に指定することはできません。指定されていない場合のデフォルトは、直ちに開始することです。

TASKID

タイマーがスケジュールされているタスクまたはグループの名前。デフォルトは、EZLETAPI を発行するタスクにタイマーをスケジュールすることです。

RECOVERY

ACTION が実行されるようにスケジュールされていて、指定されたタスクがアクティブでないときの処理方法を指定します。

AUTOLGN

自動タスクが特定のタスク名で開始されることを指定します。
AUTOLGN は、グループのタスクでは指定できません。

IGNORE

タスクがアクティブでない限り、ACTION は実行されないことを指定します。これがデフォルトです。

PURGE

タスクがアクティブでない場合には、タイマーを除去することを指定します。

TIMEZONE

時刻がグリニッジ標準時 (GMT) またはローカル・システム時刻のどちらに対応しているのかを指定します。デフォルトは GMT です。

NOTIFYIGNORE

指定されたタスクがアクティブではなかったため、アクションが実行されないときは、リストされているオペレーターに通知することを指定します。

NOTIFYPURGE

タスクがアクティブではないか、タイマーが削除されているため、アクションがパージされたときは、リストされているオペレーターに通知することを指定します。

NOTIFYREMOVE

REMOVE 時刻になったか、時間間隔を指定せずにコマンドが実行されるようにスケジュールされているために、アクションが除去されたときは、リストされているオペレーターに通知することを指定します。

NOTIFYRUN

アクションが実行されるようにスケジュールされていて、指定されたタスクがアクティブであるときは、リストされているオペレーターに通知することを指定します。

REMARK

タイマーを設定する際に、注釈を指定できるようにします。例えば、タイマーを設定するコマンド・リストを指定できます。注釈は、単一引用符またはアポストロフィで囲まなければなりません。

REPEATING

アクションが反復されることを指定します。デフォルトは NO です。REPEATING=YES と指定すると、以下のキーワードを使用する必要があります。

INTERVAL

STARTAT 時刻から REPEATOFF 時刻までの間で、アクションを反復する時間 (インターバル) を指定します。DAILY は、DAYOFWEEK、DAYOFMON および CALDAY 項目に合わせて、アクションを毎日 1 回ずつ実行するように指定します。'ddd' に指定できる範囲は 1 から 365 までです。

REPEATMAX

毎日の AT 時刻ごとにコマンドが反復され適用される回数を指定します。インターバル指定に反復カウントを乗じた積は、24 時間未満でなければなりません。REPEATMAX=NOLIMIT と指定すると、タイマーは、STARTAT または STARTAFTER 時刻から始まって、一定のインターバルでスケジュールされることになります。各新規タイマーは、すでに算出されている実行時間から、ちょうど INTERVAL の時間が経過すると実行されるように設定されています。それ以降の日については、AT 時刻も AFTER 時刻も因数にはなりません。NOLIMIT がデフォルトです。

REPEATOFF

インターバルが終了する時刻を指定します。この値は 24 時間を超えてはならず、翌日にわたることができます。タイマーは、REPEATOFF 時刻以降は稼働しません。

REPEATFOR

インターバルが継続する時間の長さを指定します。タイマーは、STARTAT 時刻以降 REPEATFOR 時間は稼働しません。このインターバルは 24 時間を超えてはならず、翌日にわたることができます。

REMOVE

タイマー・アクションが削除される時点を指定します。デフォルトは MANUALLY で、この場合、タイマーは自動的に除去されません。

REMOVEAFTER

タイマー・アクションが削除される時点を指定します。dd-hh.mm.ss.micros は、タイマーが除去される時刻を指定するもので、STARTAT または STARTAFTER 時刻よりも後になります。

DAYOFWEEK

曜日の名前です。 DAYOFWEEK と DAYOFMON および CALDAY は互いに影響を与えます。 DAYOFWEEK=ALL がデフォルトです。有効な値は次の通りです。

- SUN
- MON
- TUE
- WED
- THU
- FRI
- SAT
- WEEKDAY
- WEEKEND

NOT を指定して、選択した日を省略すると、選択した日を含んだ長い方のリストはなくなります。例えば、DAYOFWEEK=(TUE,WED,THU,FRI) と指定する代わりに、 DAYOFWEEK=(NOT MON,WEEKEND) と指定すれば同じ結果が得られ、コマンドは火曜日から金曜日までしか実行されません。

コマンドが、その月の該当日に当たると実行されるように指定できます。例えば、DAYOFWEEK=(MON(1ST,3RD),FRI(LAST)) と指定すると、コマンドは、その月の第 1 および第 3 月曜日ならびに最終金曜日にだけ実行されることになります。 LAST または LAST-n を指定すると、その月の特定の曜日の数を考慮する必要はなくなります。有効な値は、次のとおりです。

- 1ST から 5TH まで
- LAST
- LAST -1 から LAST -4 まで

DAYOFMONTH

その月の日数です。 DAYOFMONTH=ALL がデフォルトです。有効な値は次の通りです。

- 1 から 31
- ALL
- LAST
- LAST -1 から LAST -30
- NOT

DAYOFMONTH=ALL がデフォルトです。 DAYOFMONTH と DAYOFWEEK および CALDAY は互いに影響を与えます。

NOT を指定して、選択された日を省略すると、選択した日を含んだ長い方のリストはなくなります。例えば、DAYOFMON=(2,4,5,6,7,8,...29,30) と指定する代わりに、 DAYOFMON=(NOT 1,3,31) と指定しても同じ結果が得られ、このコマンドは、その月の

1 日、3 日、および 31 日には実行されません。LAST または LAST -n を指定すると、その月の日数を考慮する必要はなくなります。

CALDAY

DSISCHED で定義されているようなキーの名前です。このコマンドは、指定された日に実行されます。NOT が指定されている場合、このコマンドは指定されている日には実行されません。リストには、最大で 1,000 までの固有キーを入力できます。この限度を超えると、メッセージ DSI656I が出されます。CALDAY=ALL がデフォルトです。

TEST 作成中のコマンドを検証するためのオプション・キーワードは、実際に CRON タイマーをスケジュールしていなくても、構文上は正しいことになります。

戻りコード:

注: 戻りコードが 0 の場合、メッセージは戻されません。戻りコードがゼロ以外の場合、CHRON により生成されるメッセージはセーフに戻されます。

-8	REXX 構文上の障害
-5	REXX 停止
-1	REXX 障害
0	正常終了
4	TYPE の指定が正しくありません
8	セーフ名が誤りです
12	受け渡されたオペランドのエラー
16	セキュリティチェックが失敗しました
20	タイマー ID が固有ではありません
24	タイマー・コマンドで検出された値が無効です
28	指定された時刻はすでに過ぎています
32	カレンダーが使用できませんでした
36	タイマー・コマンドから応答がありません
40	既存のタイマー削除が失敗しました
44	内部処理エラー

出力:

セーフの内容: 戻りコードがゼロ以外の場合は、すべての生成されたメッセージ。

戻りコードが 0 の場合:

1 桁目から始まるタイマー・ハンドル。

メッセージ

以下のメッセージおよびそれに関連した戻りコードが、EZLETAPI により生成されます。

EZL228

RC 16

EZL984

RC 12

DSI450

RC 12

DSI486

RC 12

DSI649

RC 4、8、12

DSI651

RC 12

DSI654

RC 12

DSI655

RC 12

起動されたタイマー・コマンドにより生成される他のメッセージが戻されます。

例

以下のテスト `exec` では、EZLETAPI、EZLEQAPI および EZLEDAPI を使用しています。

```
/* */
/* Valid parameters to this test exec are TIMERID and OPID */
/* This routine changes the ACTION keyword of an existing timer */
/* and sets the timer using EZLETAPI. */
trace o
parse arg argstring
parse var argstring TMRID OPID .

/* Make OPID valid for the call*/
If OPID = '' Then
  OPID = ' '
  delfailed = 0
/* Build the command to be invoked */
testcmd = 'EZLEQAPI '
testcmd = testcmd || 'TASKID='OPID' '
testcmd = testcmd || 'HANDLE='TMRID' '
testcmd = testcmd || 'SAFE=SOMSAFE '

Address NETVASIS
/* Issue the command that was built */
'PIPE CORRCMD (MOE) 'testcmd,
'| SEP',
'| STEM CHRON1.'

/* Check for messages returned from PIPE */
If CHRON1.0 ^= 0 Then
  Do I = 1 to CHRON1.0 until I = CHRON1.0
  say CHRON1.I
end
```

```

Else                                     /* No PIPE errors      */
Do
    /* Put the EZLEQAPI safe output into stem    */
    'PIPE SAFE SOMSAFE ',
    '| sep ',
    '| STEM SOUTPUT.'
/* Line 1 contains the return code from EZLQAPI */
If soutput.1 = 0 Then                    /* Check return code    */
Do
    /* Line 2 contains the list of keywords that were set */
    temp_var = soutput.2
    /* Prime the EZLETAPE command */
    newtimer = 'EZLETAPE SAFE=NEWSAFE '

    /* Line 3 is the first line of keywords. Loop until the */
    /* end of the stem to retrieve them all. */
    do i = 3 to soutput.0                /* Get the parms */
    'PIPE var soutput.'i' | varload ' /* Save the keywords */
    parse var temp_var var_name ',' temp_var
    Select
    /* NEXT_POP is not a valid keyword for EZLETAPE */
    When var_name = 'NEXT_POP' Then
        nop

    /* Set the value of the ACTION keyword and append it to */
    /* to the command being built */
    When var_name = 'ACTION' Then
        newtimer = newtimer || 'ACTION='DISCONID''
    When var_name = 'HANDLE' Then
        Do
            If substr(HANDLE,1,3) = 'SYS' |,
               substr(HANDLE,1,3) = 'RST' Then
                Do
                    oldhandle = value(handle)
                End
            Else
                Do
                    oldhandle = ''
                    newtimer = newtimer || 'HANDLE='value(HANDLE)' '
                End
            End
        End

    /* Append any other keyword found */
    Otherwise
        newtimer = newtimer || var_name || '=' || value(var_name) || ' '
    End
end                                     /* Get the parms      */
End
Else                                     /* non-zero EZLEQAPI return code */
Do
    do i = 1 to soutput.0
        say soutput.i
    end
End
End

/* If there was an invalid timerid found earlier, delete that */
/* timer before continuing. */
If oldhandle ^= '' Then
Do
    deletcmd = 'EZLEDAPE safe=DELSAFE HANDLE='oldhandle' ' ||,
               'TASKID='value(taskid)
    newtimer = newtimer || 'TYPE=SET'
    'PIPE CORRCMD (MOE) 'deletcmd,
    '| SEP',
    '| STEM Chron1.',
    '| COUNT',

```

```

'| VAR errorcnt',

If Chron1.0 ^= 0 Then
Do
'PIPE SAFE DELSAFE ',
'| STEM DOUTPUT.'

If Doutput.0 ^= 0 Then
If Doutput.1 ^= '0' Then
Do
delfailed = 1
say Doutput.1
End
End
End
Else
newtimer = newtimer || 'TYPE=CHANGE'
If delfailed = 0 Then
Do
/* Issue the EZLETAPI command just built */
'PIPE CORRCMD (MOE) 'newtimer,
'| SEP',
'| STEM Chron1.',
'| COUNT',
'| VAR errorcnt',
If Chron1.0 ^= 0 Then
Do
'PIPE SAFE NEWSAFE ',
'| STEM NOUTPUT.'

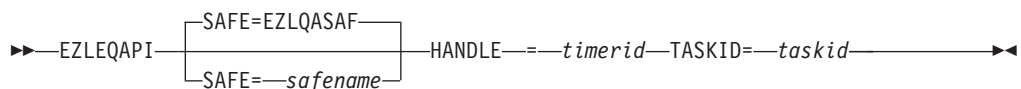
If Noutput.0 ^= 0 Then
If Noutput.1 = 0 Then
Do
say 'The handle is 'Noutput.2
End
Else
say 'Failure in EZLETAPI: Return code is 'Noutput.1
Else
say 'Nothing in the safe'
End
End
Else
SAY 'The delete of timer 'oldhandle' was not successful'
exit

```

EZLEQAPI

EZLEQAPI は、特定のタイマーが設定されているかどうかを判別するため、(CHRON コマンドにより設定された) タイマーについての照会を簡単にすることができる API です。

EZLEQAPI API の構文は、次のとおりです。



説明：

SAFE EZLEQAPI コマンドの出力が配置されるセーフの名前を指定します。デフォルトは EZLQASAF です。

HANDLE

照会対象のタイマーです。有効な値は以下のとおりです。

timerid 指名されたタイマー要求の状況を表示します。 *timerid* は、SETTIMER コマンドの HANDLE オペランドで指定されるか、あるいはシステムで生成されるオプション・ハンドルです。

TASKID

照会対象のタスクです。有効な値は以下のとおりです。

taskid 指名されたオペレーターに対する要求およびタイマー要求のみをリストします。オペレーターが現在ログオンされていなくても、*taskid* は指定できます。

戻りコード:

- 8 REXX 構文上の障害
- 5 REXX 停止
- 1 REXX 障害
- 0 正常終了
- 4 指定された基準に該当するタイマーはありません
- 8 セーフ名が誤りです
- 12 受け渡されたオペランドのエラー
- 16 セキュリティー検査が失敗しました
- 20 内部処理エラー
- 24 要求されたタイマーが CHRON タイマーではありませんでした

出力:

セーフの内容:

戻りコードがゼロ以外の場合:

エラー・メッセージが生成されます

このルーチンにより生成されたメッセージおよびそれに関連した戻りコード:

EZL228

(RC 16)

EZL253

(RC 4)

DSI450

(RC 12)

DSI486

(RC 12)

DSI649

(RC 8、12)

DSI651

(RC 12)

起動されたタイマー・コマンドにより生成される他のメッセージが戻されます。

戻りコードが 0 の場合

戻りコードが 0 の場合、要求されたタイマー情報は以下のフォームで戻されます。

- */* Line 1 contains the return code*/*
- */* Line 2 contains a comma delimited list of variables that are being returned in the safe*/*
- */* Line 3 to the end of the safe contains the values that can be set using the VARLOAD stage.*/*
- */HANDLE/timerid*
- */ACTION/'text'*
- */START/value1/STARTAFTER/value*
- */TASKID/value*
- */RECOVERY/value*
- */TIMEZONE/value*
- */NOTIFYIGNORE/values*
- */NOTIFYPURGE/values*
- */NOTIFYREMOVE/values*
- */NOTIFYRUN/values*
- */REMARK/'text'*
- */REPEATING/*
- */INTERVAL/value*
- */REPEATMAX/value*
- */REPEATOFF/value*
- */REPEATFOR/value*
- */REMOVE/value*
- */REMOVEAFTER/value*
- */REMOVEAFTER/value*
- */REPEATFOR/value*
- */CALDAY/value*
- */NEXTPOP/yyyy-mm-dd-hh.mm.ss*

注: NEXTPOP は、このタイマーが次回ポップする時刻です。

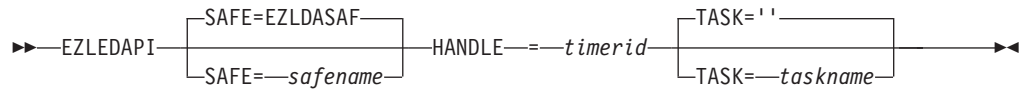
例

EZLEQAPI を使用したテスト `exec` の例については、313 ページの『EZLEAPI』を参照してください。

EZLEDAPI

EZLEDAPI

EZLEDAPI API により、アプリケーションは、それらが設定しているタイマーを削除できます。



説明：

HANDLE

削除対象のタイマーです。有効な値は、*timerid* で、特定のタイマー要求を削除します。

SAFE EZLEDAPI コマンドの出力が配置されるセーフの名前です。デフォルトは EZLDASAF です。

TASKID

削除タイマーを実行するタスクです。有効な値は、次のとおりです。

" ユーザー独自のオペレーター ID のタイマー要求を削除します。

— TASKID が指定されていなければ、これがデフォルトです。

taskid 指定されたオペレーターのタイマー要求だけを削除します。オペレーターが現在ログオンされていなくても、*taskid* は指定できます。

戻りコード：

- 8 REXX 構文上の障害
- 5 REXX 停止
- 1 REXX 障害
- 0 正常終了
- 4 削除されたタイマーはありません
- 8 セーフ名が誤りです
- 12 受け渡されたオペランドのエラー
- 16 セキュリティチェックが失敗しました
- 20 内部処理エラー

セーフの内容：

戻りコードが 0 の場合：

- 1 桁目から始まる削除されたタイマー・ハンドル。

戻りコードが 8 よりも大きい場合：

- エラー・メッセージが生成されます

メッセージ

以下のメッセージおよびそれに関連した戻りコードが、EZLEDAPI により生成されます。

EZL228

RC 16

DSI450

RC 12

DSI486

RC 12

DSI649

RC 8、12

DSI651

RC 12

起動されたタイマー・コマンドにより生成される他のメッセージが戻されます。

注: このルーチンでは、NetView PURGE TIMER=*timerid* OP=*operid* にアクセスできません。

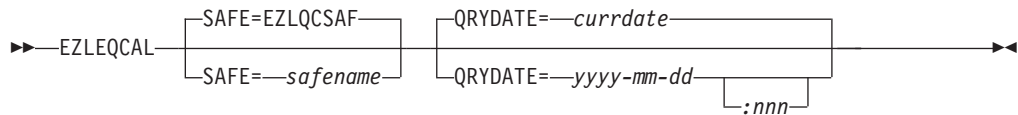
例

EZLEDAPI を使用したテスト `exec` の例については、313 ページの『EZLEDAPI』を参照してください。

EZLEQCAL

EZLEQCAL API により、配置されている定義を判別するためにカレンダーに対する照会を行うことができます。

EZLEQCAL API の構文は、以下のとおりです。



説明:

QRYDATE

照会対象の日付です。これは、日付指定に従い、:nnn と指定して、照会される日数を指定できます。yyyy の範囲は 1 から 9999 までです。nnn の範囲は 1 から 999 までです。照会可能な日付の範囲は (nnn を含む) 0001-01-01 から 9999-12-31 までです。

SAFE EZLEQCAL コマンドの出力が配置されるセーフの名前です。デフォルトは EZLQCSAF です。

戻りコード:

- 8 REXX 構文上の障害
- 5 REXX 停止
- 1 REXX 障害
- 0 正常終了
- 4 セーフ名が誤りです
- 8 受け渡されたオペランドのエラー

セーフの内容:

戻りコードが 0 の場合:

データを含むセーフは、次に記述しているとおりです。データは、1 列目から始まります。

この例では、2010 年 12 月 31 日からの 4 日間を表示します。

EZLEQCAL 2010-12-31:4

```
DEC 31, LAST    2010 FRI 5TH, LAST    (NO SPECIAL CALENDAR DAYS)
JAN  1, LAST-30 2011 SAT 1ST, LAST-4  DAYA, HOLIDAY, NEW_YEARS_DAY
JAN  2, LAST-29 2011 SUN 1ST, LAST-4  DAYB
JAN  3, LAST-28 2011 MON 1ST, LAST-4  DAYC
```

1 列目は月です。 2 列目は日付指定です。 3 列目は年です。 4 列目は曜日です。 5 列目は曜日指定です。最後の列はユーザーにより定義された日です。

注: このルーチンでは、DSIPARM の DSISCHED データ・セットに対して READ アクセスできなければなりません。

第 17 章 インストール・システム出口

本章では、プロダクト・センシティブ・プログラミング・インターフェースおよび関連のガイダンス情報を説明します。

このセクションでは、自動化に使用可能なインストール・システム出口について簡単に説明します。

インストール・システム出口とは？

一部の NetView 出口を使用すると、データに対するプログラミング・アクセスが可能になります。そのような出口を使用すると、オペレーター・コマンド、ログオン、メッセージ、および MSU のテキストを、ユーザーの機能で取り込めます。テキストの発生元や NetView 処理の段階により、これ以外の出口も使われます。自動化に関連して使用される重要な出口には、インストール・システム出口の DSIEX02A、XITCI、DSIEX16、DSIEX16B、および DSIEX17 があります。

自動化のインストール・システム出口およびその他のインストール・システム出口の詳細については、「*IBM Tivoli NetView for z/OS プログラミング: アセンブラー*」および「*IBM Tivoli NetView for z/OS プログラミング: PL/I および C*」を参照してください。

インストール・システム出口 DSIEX02A

NetView プログラムは DSIEX02A 出口を呼び出し、オペレーターの端末への標準出力を処理します。DSIEX02A の処理は自動化テーブルが使用される前に発生するため、DSIEX02A 出口で行われるメッセージに対する変更処理は、すべてメッセージの自動化に影響を与える可能性があります。DSIEX02A 出口は、メッセージの変更や置換以外に、メッセージの削除にも使用できます。メッセージが削除されると、自動化テーブルはそのメッセージを使えません。DSIEX02A 出口は、アセンブラー、PL/I、または C のいずれでも作成できますが、パフォーマンスの点でアセンブラーを使用します。

BNJDSERV のインストール・システム出口 XITCI

BNJDSERV タスクは MSU を受信した後に出口 XITCI を呼び出します。MSU が自動化テーブルもしくはハードウェア・モニター・ログに渡る前に、MSU を修正したり、置換したり、削除することができます。BNJDSERV の XITCI 出口を DSICRTR の出口の代わりに使用すると、BNJDSERV が DSICRTR からの MSU のほかに ALERT-NETOP からの MSU も処理する点で便利です。出口 XITCI は、アセンブラー、PL/I、または C のいずれでも作成できますが、パフォーマンスの点でアセンブラーを使用します。

インストール・システム出口 DSIEX16 および DSIEX16B

NetView プログラムは、自動化テーブル処理が発生した後すぐに、DSIEX16 出口と DSIEX16B 出口を呼び出します。DSIEX16 出口と DSIEX16B 出口は、アセンブラーで作成する必要があります。これらの出口は、自動化テーブルのテストには呼び出されません。

NetView プログラムが DSIEX16 出口を呼び出すのは、自動化テーブルがメッセージを処理した後で、かつ自動化テーブルから発行されるすべてのコマンドが実行される前です。DSIEX16 出口を使用すると、メッセージの処理オプションを修正したり、メッセージを再形式設定したり、メッセージを置換したりできます。自動化テーブルで指定された処理オプション（メッセージをログに記録し表示するかどうかなど）を置換して、NetView の指定変更設定値が有効にならないようにすることもできます。また、メッセージの応答に対して発行する新規の自動化コマンドを指定することもできます。さらに、DSIEX16 出口を使用して、メッセージの抑止および自動化の効果をモニターすることもできます。

NetView が DSIEX16B 出口を呼び出すのは、自動化テーブルが MSU を処理したあとで、かつ自動化テーブルから発行されるすべてのコマンドが実行される前です。DSIEX16B 出口を使用すると、以下のことが可能になります。

- MSU およびその属性の検査と修正
- 自動化処理の結果の変更
- MSU 自動化の効果のモニター

行う機能によっては、DSIEX16 出口と DSIEX16B 出口の両方に同じルーチンを使用できる場合もあります。

インストール・システム出口 DSIEX17

DSIEX17 出口は、NetView プログラムが MVS メッセージとオペレーター・メッセージ削除 (DOM) を自動化内部機能要求 (AIFR) 形式に変換したあとに呼び出されます。この出口を使用して、メッセージまたは DOM を修正したり削除したりすることができます。DSIEX17 出口は、アセンブラーでのみ作成できます。

第 5 部 単一システムの自動操作

第 18 章 自動操作機能のセットアップ・タスク

自動操作を使用するには、その前に以下のセットアップ作業を行う必要があります。

- システムの自動化を使用したい場合は、NetView システムとオペレーティング・システムとの通信をセットアップする。
- 自動処理を行うために自動タスクを定義し活動化する。

NetView システムとオペレーティング・システムの通信の確立

システムの自動化のため、またはオペレーターが NetView プログラムからオペレーティング・システム・コマンドを出すためには、オペレーティング・システムと NetView システムとの間のインターフェースを活動化します。

システム自動化のための z/OS システムの準備

NetView プログラムを使用して、z/OS オペレーティング・システムをシステム自動化のために準備するには、以下のことを行います。

- NetView プログラムを z/OS オペレーティング・システムにサブシステムとして定義します。
- 使用したい z/OS オペレーティング・システム・メッセージ配布オプションを選択します。
 - サブシステム・インターフェース
 - 拡張複数コンソール・サポート (EMCS) コンソール
- オペレーティング・システムから NetView プログラムにシステム・メッセージ (MPF テーブル) を転送します。
- z/OS オペレーティング・システムにサブシステム割り振り可能コンソールを定義します。
- NetView 開始プロシージャを検討します。
- オプションで、z/OS オペレーティング・システム・コマンド用の CMDDEF ステートメントを追加して、NetView プログラムから z/OS オペレーティング・システムおよびサブシステム・コマンドを出しやすいようにします (この指定をしておくと、システム・コマンドの接頭部に z/OS を付ける必要がなくなります)。

上記のステップは、NetView サブシステム・アドレス・スペースと NetView アプリケーション・アドレス・スペースの両方と z/OS オペレーティング・システムとの対話の準備作業です。z/OS オペレーティング・システムで、この 2 つの NetView アドレス・スペースが協調して、自動操作機能が使えるようになります。

334 ページの図 83 に、サブシステム・インターフェースを使用した場合の、z/OS システムと NetView プログラム間のメッセージの流れを示します。335 ページの図 84 は、コマンドの流れを示しています。

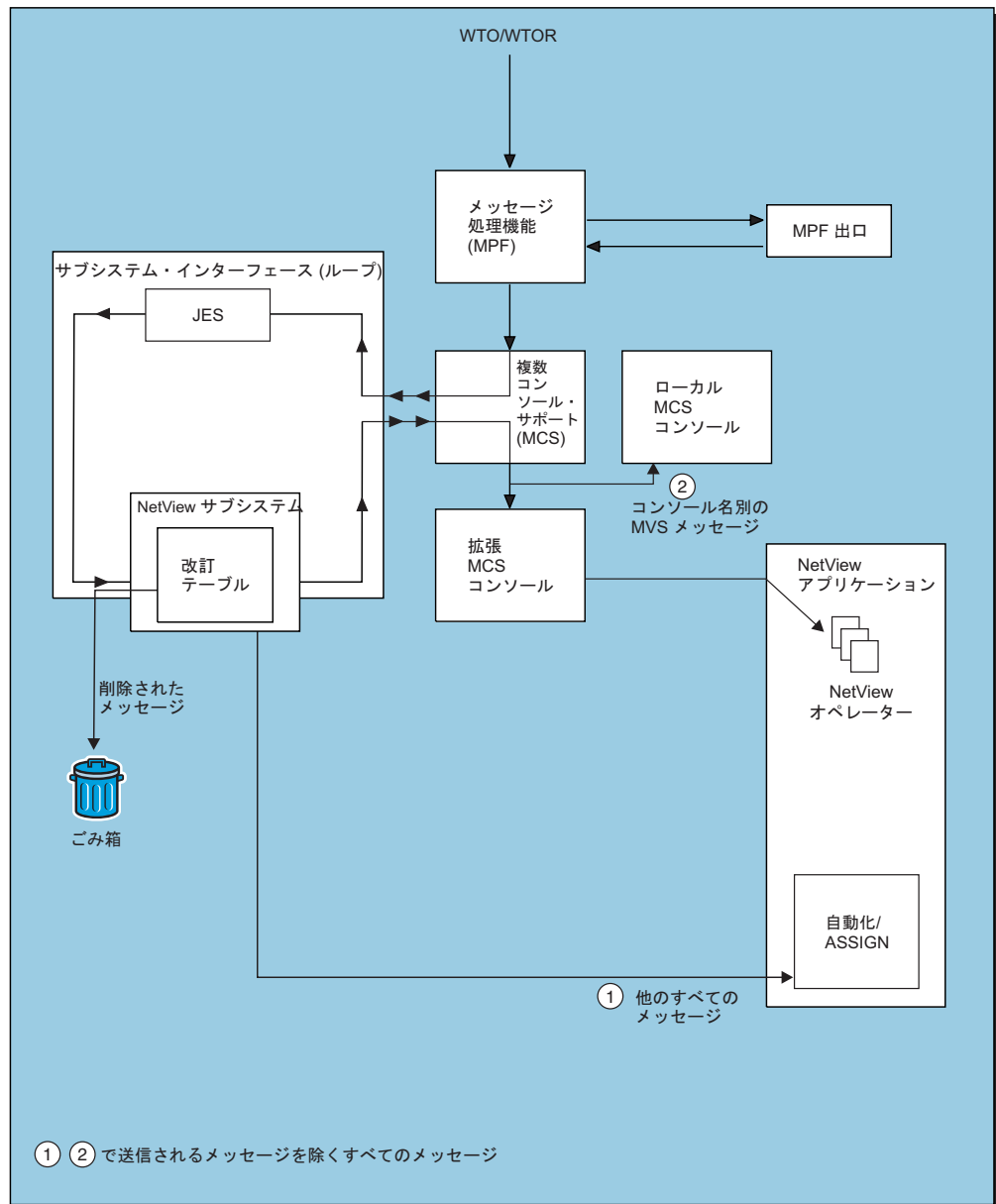


図 83. z/OS システムと NetView サブシステム・インターフェース間のメッセージの流れ

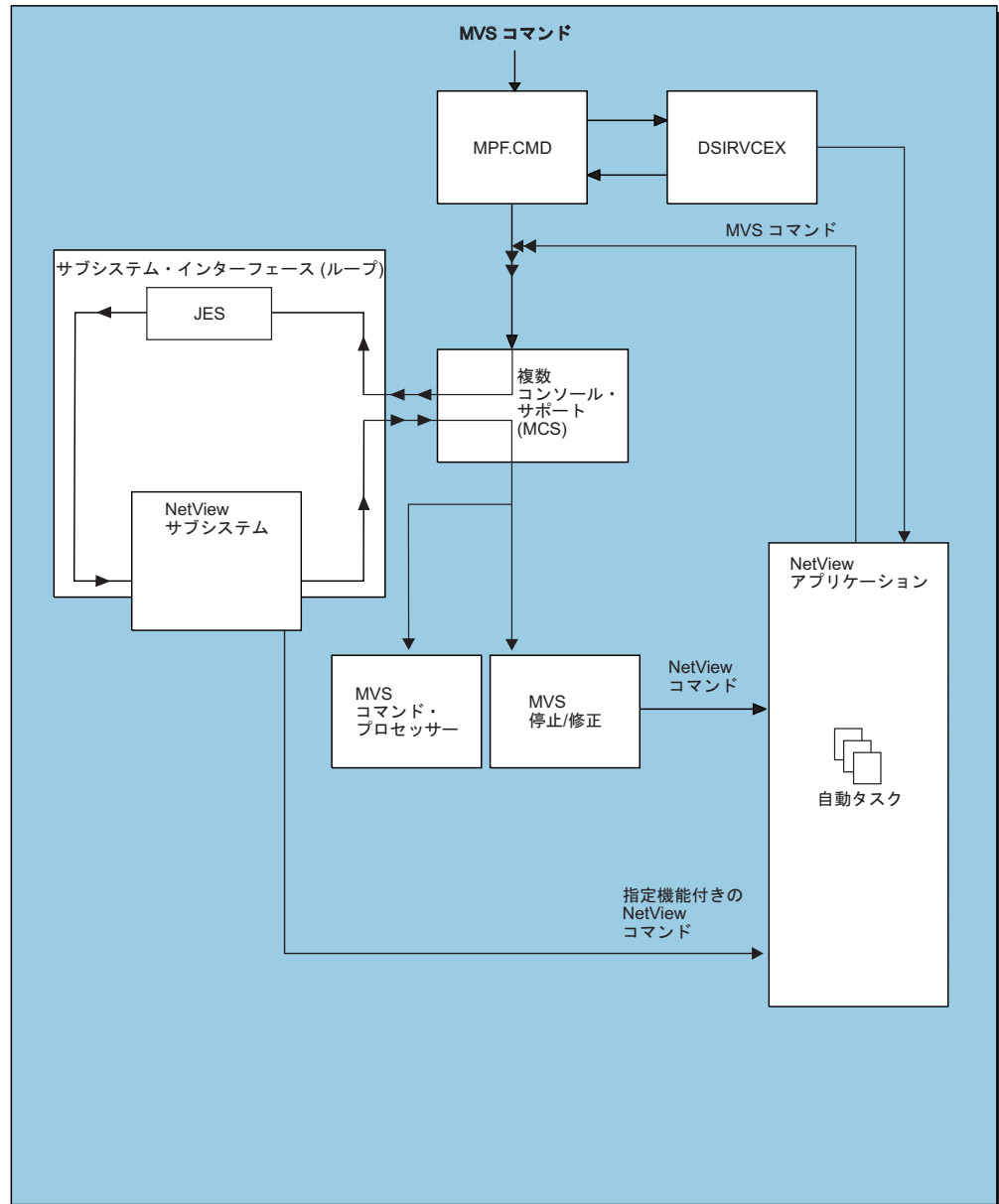


図 84. z/OS と NetView プログラム間のコマンドの流れ

NetView サブシステム・アドレス・スペースは、z/OS サブシステムとして機能します。z/OS サブシステム・インターフェースでブロードキャストされるメッセージを選択し、選択されたメッセージ・コピーを自動化操作のために NetView アプリケーション・アドレス・スペースに転送します。EMCS コンソールを使用する場合、サブシステム・アドレス・スペースは、メッセージでなくコマンドを受信するために使用されます。z/OS システム内のメッセージおよび NetView システムへのサブシステム・インターフェース全体にわたるメッセージの流れの詳細については、589 ページの『付録 D. MVS メッセージおよびコマンド処理』を参照してください。

NetView アプリケーション・アドレス・スペースは、すべてのネットワーク管理機能、メッセージ・システムとネットワーク・メッセージの処理、および NetView プログラムに関する表示サービスを実行します。これには、自動化のために使用する自動化テーブルおよび自動タスクが含まれます。

NetViewプログラムを z/OS システムにサブシステムとして定義する

以下の理由により、NetView プログラムを z/OS オペレーティング・システムのサブシステムとして定義します。

- サブシステム・インターフェースをシステム、サブシステム、およびアプリケーション・メッセージに使用するため
- z/OS コンソールからの NetView コマンドの入力を使用可能にするため
- プログラム間インターフェースを使用可能にするため

NetView プログラムをサブシステムとして定義するには、SYS1.PARMLIB の IEFSSNnn メンバーを更新してください。IEFSSNnn メンバーには、z/OS システムの初期化中に複数の 2 次サブシステムを定義するパラメーターがあります。80 バイトの IEFSSNnn レコードにはそれぞれ、1 つの 2 次サブシステムを定義するパラメーターがあります。

NetView サブシステムのエントリー名は、NetView サブシステムの 4 文字の名前です。アプリケーション・アドレス・スペースとサブシステム・アドレス・スペースの両方に対する開始プロシージャ名の最初の 4 文字は、NetView プログラムに定義するこの 4 文字のサブシステム名と一致していなければなりません。例えば、NetView 製品に付属のサンプルでは、アプリケーション・アドレス・スペースとサブシステム・アドレス・スペースのいずれの開始プロシージャ名も、CNMP で始まっています。このプロシージャを使用する場合、IEFSSNnn メンバーにエントリー CNMP を指定してください。この定義は、z/OS システムを次に IPL したときに有効になります。

インストール時に SYS1.PARMLIB の IEASYSnn メンバーで MAXUSER と RSVNONR に指定した値が、NetView プログラムの停止と再始動に想定した並行処理回数と比べて適切であることを確認してください。詳しくは、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。

z/OS から NetView プログラムへのシステム・メッセージの転送の確認

システム・メッセージを z/OS システムから NetView プログラムへ転送するには、以下の 2 とおりの方法があります。

- サブシステム・インターフェースを介する方法
- EMCS コンソールを介する方法

z/OS はメッセージをオペレーターへの書き込みキュー・エレメント (WQE) の代わりにメッセージ・データ・ブロック (MDB) に送信するため、EMCS コンソールを使用すると、システム・メッセージの詳細情報を入手できます。MDB には、メッセージを表示する色などの追加情報が入っています。

NetView プログラムが z/OS システム、サブシステム、およびアプリケーションのメッセージにアクセスするためには、z/OS プログラムがサブシステム・インターフェースでメッセージをブロードキャストする必要があります。MPF のメッセージに AUTO(NO) を指定しない限り、サブシステム・インターフェースの各メッセージは NetView プログラムにより検査されます。EMCS コンソールを使用する場合、CNMCSSIR タスクはデフォルトで、MPF テーブルで AUTO(YES) または AUTO(token) とマークされているすべてのメッセージを受信します。NetView プロ

グラムが使用中の他の拡張コンソールは、デフォルトで、コマンドの応答のみを受信するか、コンソール ID またはコンソール名により指示された WTO を受信するようセットアップされています。その後、NetView ASSIGN コマンドと自動化テーブル処理が発生し、メッセージの経路指定や自動化が可能になります。NetView プログラムが検査するメッセージがどれであるかを決定するための MPF の AUTO 機能の使用については、341 ページの『システム・メッセージの抑止』を参照してください。自動化に使用するため NetView プログラムに転送したいメッセージには、それぞれ AUTO(YES) を使用してください。

EMCS コンソールの属性は変更可能で、特定の経路コードを持つメッセージを、指定されたコンソールに送信できます。経路コードは、RACF OPERPARM セグメント、もしくは z/OS VARY コマンドの ROUT キーワードで指定できます。拡張コンソールの属性の説明については、「IBM Tivoli NetView for z/OS セキュリティー・リファレンス」を参照してください。

EMCS コンソールの属性を変更する場合は、CNMCSSIR のタスクにより以前に受信されていたメッセージをすべて受信するように、拡張コンソールをセットアップしてあることを確認してください。

z/OS コマンドが NetView プログラムに所有されているコンソールから出され、さらにコマンドの応答に AUTO(YES) と SUP(YES) が指定されている場合、そのメッセージは CNMCSSIR タスクのもとで自動化されます。そのメッセージは、非送信請求の z/OS システム・メッセージとして処理されます。

EMCS コンソールの動的定義

コンソール名は、NetView オペレーター ID と同じです。GETCONID コマンドを使用して拡張 MCS コンソールを取得することもできます。

拡張 MCS コンソール名は、使用しているシステムおよびシスプレックス全体の中で固有のものでなければなりません。SYS1.PARMLIB の CONSOLxx メンバーで定義された名前は、いずれも拡張 MCS コンソール名として使用できません。

GETCONID コマンド: GETCONID コマンドを使用すると、オペレーター、自動タスク、または基本プログラム・オペレーター・インターフェース・タスク (PPT) 用のコンソールを取得できます。GETCONID コマンドで取得したコンソールには、タスクのデフォルト値とは別の名前を指定できます。デフォルト以外の名前を指定することによって、コンソール名がシスプレックス内で固有のものでなければならないという z/OS の制限に従うことになります。GETCONID コマンドとそのパラメーターの詳細な説明については、NetView オンライン・ヘルプを参照してください。

注: オペレーターまたは自動タスクが、z/OS コマンドを発行するか、またはそのコンソール名にあてられた z/OS メッセージを受信する必要があることが確実な場合に、GETCONID コマンドを使用します。z/OS コマンドを入力する可能性の少ないオペレーターまたは自動タスクに固有の名前を割り振らないで、コンソールに固有の名前を割り当てる場合には SETCONID を使用します。

STORAGE パラメーター、QLIMIT パラメーター、ALERTPCT パラメーター、および QRESUME パラメーターを制御するために、個々のオペレーターまたは自動タスクの初期設定コマンド・リストで GETCONID コマンドを使用する場合は、注意が

必要です。これらのパラメーターは GETCONID コマンドで指定できますが、z/OS VARY コマンドを使って修正することはできません。これらのパラメーターは、以下のとおりです。

STORAGE

拡張 MCS コンソール・メッセージ用に z/OS データ・スペースに割り振られる最大メガバイト数を指定します。このストレージは、NetView プログラムに送られる拡張 MCS コンソール・メッセージすべてを記憶するためのものです。単にこのコンソール用のメッセージだけのためのものではありません。定義した 1 番目の拡張 MCS コンソールで、最大のストレージを指定します。このストレージの値を変更するには、すべての拡張 MCS コンソールを解放したあと、GETCONID コマンドを新規の最大ストレージの値で再度出す必要があります。

このストレージに空き容量がなくなると、NetView プログラムはメッセージ DWO201I を出します。

QLIMIT

このコンソールで NetView プログラム用の拡張 MCS コンソール・データ・スペース内で一度にキューに入れられるメッセージの数を指定します。

QLIMIT に達すると、NetView はメッセージ DWO202I を出します。

ALERTPCT

QLIMIT の特定比率 (%) に到達すると、z/OS プログラムはコンソールに警告メッセージ (DWO204I) を発行します。ALERTPCT では、その比率 (%) を指定します。

QLIMIT に達すると、z/OS システムはそのコンソールについてのメッセージのキューイングを停止します。NetView プログラムは、一定のメッセージ・レベル (QRESUME の値) に達するまでキューからメッセージを取り出し、その後 z/OS がメッセージのキューイングを再開します。しかし、キューイングが停止した時点から再開する時点までの、そのコンソールのすべてのメッセージは失われます。このメッセージをトラップして、すぐにアクションを行う (メッセージ・トラフィックを他のコンソールに切り替えるなど) と、メッセージの消失を防げます。

QRESUME

その比率に達した時点でキューイングを再開する QLIMIT の比率 (%) を指定します。

QRESUME の値に達すると、NetView プログラムはメッセージ DWO608I を出します。

GETCONID のデフォルト値が十分な場合、コマンド許可を使用して、オペレーターが STORAGE、QLIMIT、QRESUME、および ALERTPCT に任意の値を入力しないようにできます。キーワードの制約については、「*IBM Tivoli NetView for z/OS* セキュリティー・リファレンス」を参照してください。

デフォルト値以外の値を使用したいときは、拡張 MCS コンソールにアクセスしているオペレーターが使用するそれぞれの初期設定コマンド・リストに、それらの値をコーディングしてください。これにより、最初に実行する初期設定コマンド・リストが適切な値を設定します。

SETCONID コマンド: このコマンドを使用すると、オペレーターが使用するコンソールの名前を動的に選ぶことができます。GETCONID コマンドとは違って、コマンドが発行されるときには、コンソールは取得されていません。オペレーターがログオンするとき、または自動タスクが SYSPLEX 環境で固有のコンソール名の予約を開始するときの CLIST の実行の際に役に立ちます。これにより、システムのオーバーヘッドが軽減されます。というのは、必要になるまでコンソールが取得されないからです (z/OS コマンドは、そのオペレーターが入力します)。

RELCONID コマンド: RELCONID コマンドを使うと、オペレーター、自動タスク、もしくは PPT で取得されていた拡張 MCS コンソールまたはサブシステム・コンソールを解放できます。

RELCONID コマンドの詳細については、NetView オンライン・ヘルプを参照してください。

NetView 開始プロシージャの検討

CNMSAMP ライブラリーにある CNMPSSI (CNMSJ010) は、NetView サブシステム・アドレス・スペースの開始プロシージャのサンプルです。また、CNMPROC (CNMSJ009) は、NetView アプリケーション・アドレス・スペースの開始プロシージャのサンプルです。これらの NetView サンプルで、CNMP は IEFSSNnn の z/OS システムに定義された 4 文字のサブシステム名を指します。

サンプル CNMPSSI プロシージャにあるシンボリック・パラメーターの説明については、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。これらのパラメーターは、インストール・システムの要件に合わせて変えられます。

NetView プログラムは、JES や VTAM を開始する前に開始し、NetView プログラムで JES、VTAM、その他のサブシステムやアプリケーションの開始を自動化できます。初期設定タスクの拡張自動化サンプル・セットは、このような方法を使用しています。最初に NetView プログラムを開始したい場合は、必要なシステム定義変更の詳細について 680 ページの『拡張自動操作サンプル・セットの使用準備』を参照してください。

NetView プログラムからシステム・コマンドを出すための CMDDEF ステートメントの追加

システム自動化は、z/OS のシステム・コマンド、サブシステム・コマンド、およびアプリケーションのコマンドを NetView プログラムから出すことが基本です。

NetView プログラムの z/OS コマンド・プロセッサを使用すると、NetView オペレーターは、コマンドの前に z/OS を付けて、NetView プログラムから z/OS のシステム・コマンド、サブシステム・コマンド、またはアプリケーション・コマンドを入力することができます。追加のアクションは必要ありません。

z/OS のシステム・コマンドまたはサブシステム・コマンドが、NetView プログラムのコマンドまたは VTAM のコマンドと同じでない限り、CMDDEF ステートメントを CNMCMD 内にセットアップできます。これにより、コマンドの前に z/OS を付けなくても、特定の z/OS のシステム・コマンドとサブシステム・コマンドを NetView プログラムから入力できます。

z/OS のシステム・コマンドまたはサブシステム・コマンド用の CMDDEF ステートメント構文は、次のようになっています。

```
CMDDEF.commandname.MOD=CNMCMJC  
CMDDEF.commandname.TYPE=R  
CMDDEF.commandname.RES=Y
```

項目の意味:

commandname

z/OS の任意のシステム・コマンドまたはサブシステム・コマンドの名前

この方式で z/OS、JES2、および JES3 のコマンドを定義する CMDDEF ステートメントの例が、拡張自動化サンプル・セットのメンバー CNMS6401、CNMS6402、および CNMS6403 にあります。

注: 多くの共通システム・オペレーター・コマンド verb のスペルは、NetView コマンドに似ています。例えば、VARY、MODIFY、DISPLAY、および REPLY システム・コマンドの名前と省略形は、NetView プログラムの ACF VTAM コマンドと同じで、HOLD の z/OS 省略形は、HELP コマンドに対する NetView コマンド同義語と同じです。z/OS コマンドの名前は変更できません。これらの z/OS verb の定義を回避するか、該当する CMDDEF ステートメントを名前変更してください。

自動タスクの定義と活動化

自動タスクを使って、コマンドを出したり、メッセージに応答することができます。自動タスクは、システムの自動化にもネットワークの自動化にも非常に重要な働きをします。自動タスクは、オペレーター端末タスク (OST) であるため、NetView プログラムに OST 定義ステートメントが必要です。自動化計画を実施するすべての自動タスクについて、DSIOPF 中に OST 定義ステートメントを含める必要があります。

自動タスクを定義する方法については、133 ページの『自動タスクの定義』を参照してください。

第 19 章 メッセージの抑止とアラートのフィルター操作

メッセージの抑止とアラートのフィルター操作は、自動操作を実現するために行う最初の重要なステップです。これを行うことにより、オペレーター介入の必要がない情報は、オペレーターに表示されないようになります。また、メッセージの抑止を行うと、NetView 自動化機能で多数の通知メッセージ操作を行う必要がなくなります。

抑止対象候補メッセージのリストが、自動化サンプル・セットの中にあります。抑制対象となるかどうかを識別するのに役立つログ分析プログラムも、自動化サンプル・セットに入っています。詳しくは『523 ページの『ログ分析プログラム』』を参照してください。

システム・メッセージの抑止

NetView for z/OS バージョン 5 リリース 2 より前の各リリースでは、多数のシステム・メッセージを抑制するためには、MVS メッセージ処理機能 (MPF) テーブルまたは MPF 出口を使用するのが最良の方法でした。MPF の詳細については、z/OS ライブラリーを参照してください。

NetView for z/OS バージョン 5 リリース 2 では、MPF よりはるかに多くの機能を持つメッセージ改訂テーブルをコーディングできます。次の解説書によって詳細情報が得られます。

- 30 ページの『メッセージ改訂テーブル』に記載されているメッセージ改訂テーブルの説明
- サンプル CNMSMRT1 テーブル
- NetView オンライン・ヘルプまたは「*IBM Tivoli NetView for z/OS Command Reference Volume 2 (O-Z)*」の REVISE コマンド

ネットワーク・メッセージの抑止

ネットワーク・メッセージなどの一部のメッセージは、オペレーティング・システムのメッセージ処理機能の対象となりません。この種のメッセージの中で不要なメッセージを抑止するには、自動化テーブルを使用します。自動化テーブル・ステートメントを作成すると、抑止したいメッセージを、メッセージ ID、メッセージ・テキスト、その他のメッセージ属性に基づいて正確に選択できます。自動化テーブル・ステートメントのコーディング方法については、163 ページの『第 15 章 自動化テーブル』を参照してください。362 ページの『メッセージを自動化する自動化テーブル・ステートメントの作成』に、その例があります。

メッセージを指定したあとは、DISPLAY(NO) アクションを使用して表示を抑止します。また、HCYLOG、NETLOG、SYSLOG の各アクションを使用すると、抑止されたメッセージを NetView にログ記録させる必要があるかどうかを指定できます。240 ページの『処置』では、これらの自動化テーブル・アクションについて説明しています。

アラートのフィルター操作

NetView で用意されている以下の 2 セットのフィルターを使用すると、アラート管理を行うことができます。

- 記録フィルター

NetView がどのレコードをハードウェア・モニター・データベースに記録するかを判別します。これによって、不要なデータが累積されないようにできます。また、記録フィルターを使用すると、アラートからメッセージを生成したり、アラートをフォーカル・ポイントに経路指定したり、アラート・カラーおよび強調表示オプションを選択したりすることもできます。

- 目視フィルター

個々のオペレーターに表示される情報を制限します。目視フィルターを使用すると、オペレーターは自分が関係するアラートだけを表示でき、ハードウェア・モニター・データベースのすべての情報を分類する必要はありません。

記録フィルター

記録フィルターを設定するには、ハードウェア・モニターの **SRFILTER(SRF)** コマンドか、NetView 自動化テーブル・アクションを使用します。

アラート・タイプの問題レコードは、最初にハードウェア・モニターに渡されます。そこで、先に発行した **SRFILTER** コマンドによって、問題レコードの初期フィルター設定方法が決まります。そのレコードが自動操作に適している場合は、次に自動化テーブルに渡されます。自動化テーブルのステートメントに **SRF**、**COLOR**、およびその他のアクション・コマンドを使用すると、アラートのための初期設定値はオーバーライドされます。最後に、最終的な有効な設定値を使って、NetView プログラムは、指定された項目に従ってレコードの処理を行います。経路指定の詳細については、108 ページの『NetView プログラムのハードウェア・モニター・データ、および MSU の経路指定』を参照してください。

注: レコードをブロックするために **SRFILTER** コマンドを使用していても、レコードは自動化テーブルに渡されます。自動化テーブルで、**BLOCK** の設定値をオーバーライドする場合があります。

ハードウェア・モニターに着信した非送信請求レコードがデータベースに送られるのは、記録フィルターを通過した場合に限られます。フィルター操作には、以下のレベルを使用できます。

ESREC

イベントおよび統計記録フィルターです。レコードをイベントとしてログに記録するかどうかを定義します。オペレーターは、レコードをイベント・パネルに表示させることができます。

AREC アラート記録フィルターです。ESREC フィルターを通過したレコードも、アラートとしてログ記録する必要があるかどうかを定義します。オペレーターは、アラート・パネルでレコードを表示させることができます。

OPER オペレーター・フィルターです。アラートに関する情報を含む **BNJ146I** と **BNJ030I** のメッセージをハードウェア・モニターに生成させる必要があるかどうかを定義します。これらのメッセージは許可レシーバーに送られ、通常

のメッセージ処理を通ります。ただし、OPER フィルターが適用されるのは、レコードが AREC フィルターを通過した場合に限られます。

ROUTE

ROUTE フィルターです。NetView プログラムが、アラートをローカルにログ記録するだけでなく、ハードウェア・モニターのアラート・フォーカル・ポイントにも転送するかどうかを定義します。ただし、AREC フィルターを通過しない限り、アラートは転送できません。LUC および LU 6.2 方式で転送されたアラートは両方とも、フォーカル・ポイントで再度 OPER および COLOR フィルターを通過します。LUC で転送されたアラートのフォーカル・ポイントにおける、フィルター操作の詳細については、444 ページの『LUC を使用したアラート転送』を参照してください。LU 6.2 方式で転送されたアラートのフォーカル・ポイントにおけるフィルター操作の詳細は、440 ページの『SNA-MDS/LU 6.2 転送アラート用のフィルターの記録』を参照してください。

TECROUTE

TECROUTE フィルターです。NetView プログラムが、(アラートをローカルにログ記録するだけでなく) 指定イベント・マネージャーにアラートを転送するかどうかを定義します。ただし、AREC フィルターを通過しない限り、アラートは転送できません。

TRAPROUT

TRAPROUT フィルターです。NetView プログラムが、アラートをローカルにログ記録するだけでなく、SNMP マネージャーにも転送するかどうかを定義します。ただし、AREC フィルターを通過しない限り、アラートは転送できません。

COLOR

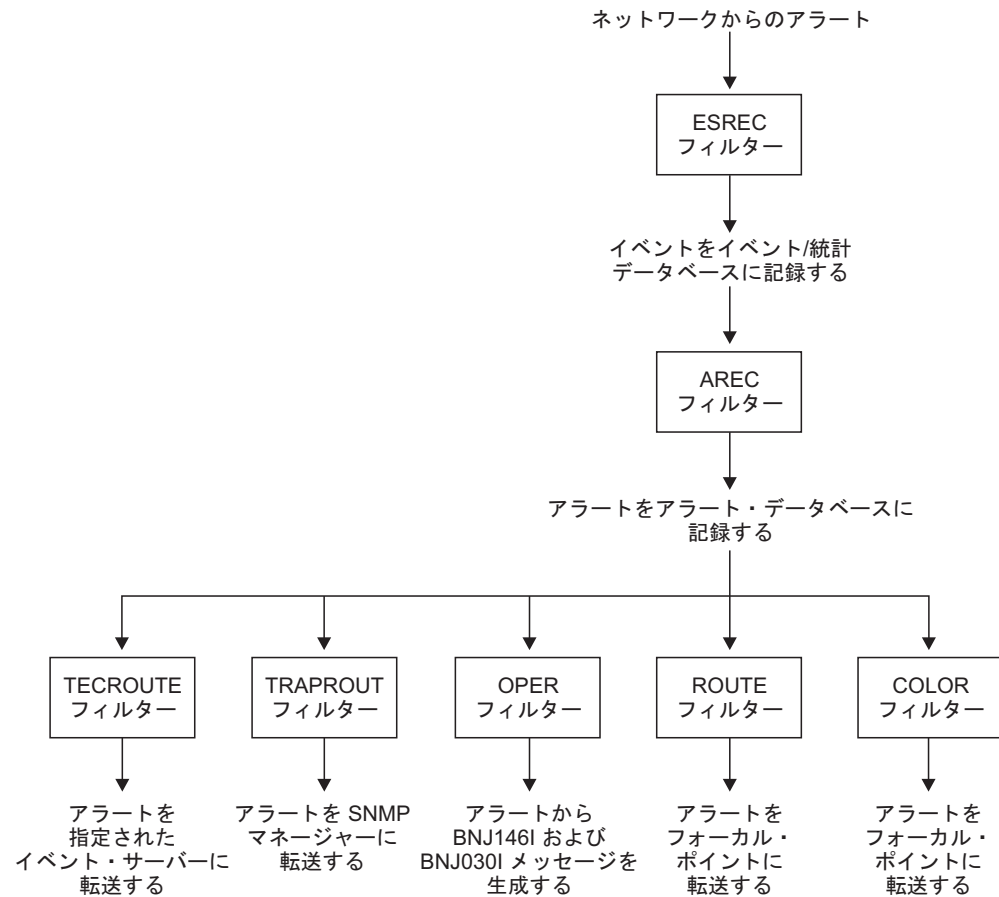
色および強調表示フィルターです。ハードウェア・モニターにレコードを表示する方法を定義します。アラートの色を選択したり、輝度の高さを指定したりできます。また、拡張強調表示オプション (下線、明滅、もしくは反転表示) の選択や、レコードを表示する際にアラームを鳴らすかどうかの指定もできます。色および強調表示フィルターは他のフィルターと違って、BLOCK の値も PASS の値も使用しません。

f

ESREC、AREC、OPER、ROUTE、TECROUTE、および TRAPROUT の各フィルターを設定するには、ハードウェア・モニターからは **SRFILTER** (SRF) コマンドを、自動化テーブルからは **SRF** アクションを使用します。色および強調表示の属性を設定するには、ハードウェア・モニターから **COLOR** パラメーターを指定した **SRFILTER** コマンドを発行します。しかし、自動化テーブルから **SRF** アクションを使用してこれらの属性を設定することはできないので、その代わりに、**COLOR** アクション、**HIGHINT** アクション、および **XHILITE** アクションを使用します。**SRFILTER** コマンドで指定した設定値は、すべて自動化テーブルを使ってオーバーライドできます。

レコードが、**SRFILTER** コマンドおよび自動化テーブルからフィルター設定値を受信すると、ハードウェア・モニターは、最終的な設定値に矛盾がないかどうかを検査します。344 ページの図 85 は、フィルター間の階層を示しています。COLOR

以外のフィルターはいずれも、フィルターの下にあるアクションを停止することにより、レコードをブロックできます。



統計、イベント、およびアラート

ハードウェア・モニターが受信した非送信請求レコードはすべて、イベントか統計かに分類されます。統計レコードは、設定されたしきい値を超えるとイベント・レコードに移される可能性があります。

デフォルトの `SRFILTER` 設定値は、すべてのイベントについて `PASS` になります。そのため、イベントはいずれも `ESREC` データベースに入っています。しかし、特定レコードがハードウェア・モニター・データベースに記録されるのをブロックしたい場合は、`SRFILTER ESREC BLOCK` コマンドか自動化テーブルの `SRF(ESREC BLOCK)` アクションのいずれかを使ってイベントのブロックを選択します。

レコードが `ESREC` フィルターを通過した場合、ハードウェア・モニターはそれをイベントとして記録します。イベントが `AREC` フィルターも通過した場合、ハードウェア・モニターはイベントからアラートを作成します。オペレーターは、`Alerts-Dynamic` パネルおよびその他のパネルにアラートを表示させることができま

す。デフォルトの AREC 設定値は、イベント・タイプとリソース・タイプによって異なります。デフォルトの詳細については、NetView オンライン・ヘルプを参照してください。

COLOR フィルターおよび OPER フィルター

COLOR フィルターは、Alerts-Dynamic パネルおよび Alerts-Static パネルでのアラートの表示方法を判別するために使用します。いずれかのフィルターで、アラートの色または強調表示の値を指定した場合、アラートはその色で表示されますが、それ以外の場合は、アラートはデフォルト処理のままになっています。デフォルト処理を指定するには、カラー・マップ (「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド, SA88-4388」を参照)および SRFILTER COLOR DEFAULT コマンドを使用してください。カラー・マップおよび COLOR DEFAULT の標準設定では、最初アラートは Alerts-Dynamic パネルの最上部に白で表示され、アラームが鳴るようになっていますが、それ以外のときはアラートは空色で表示されます。

アラートが記録されると、ハードウェア・モニターは、そのアラートについての OPER フィルター属性の設定を調べます。OPER フィルターは、NetView プログラムが、アラートを記述するためにメッセージを NetView 許可オペレーターに送信するべきかどうかを決定します。SRFILTER OPER コマンドのデフォルトは、BLOCK です。

レコードが ESREC フィルターおよび AREC フィルターを通過し、ハードウェア・モニターの Alerts-Dynamic パネルに表示される例を、図 86 に示します。

```
DOMAIN RESNAME TYPE TIME   ALERT DESCRIPTION:PROBABLE CAUSE
CNM01  IBMRING LAN  12:30  LOBE WIRE FAULT:RING ADAPTER CABLE
```

図 86. Alerts-Dynamic パネルで受信されたアラート

IBMRING というリソースに対して OPER フィルターが PASS に設定されている場合、NetView プログラムはアラート用のメッセージ (図 87 参照) を生成します。

```
BNJ030I 10/15 12:30 PERM ALERT RECEIVED FROM THE FOLLOWING
          RESOURCE: LAN  IBMRING
BNJ146I 10/15 12:30 N TYPE=PERM PRID=3725 MAJ=01 MIN=01 ACT=10
          HIER=A03NCP,COMC,IBMRING,LAN DOMID=CNM01
```

図 87. NetView プログラムによりアラート用に生成されたメッセージ

記録フィルターに関するその他の情報

ROUTE フィルターを使用すると、ハードウェア・モニターのアラート・フォーカル・ポイントに転送するアラートを選択できます。デフォルトの ROUTE フィルターは PASS で、この場合、分散システムはすべてのアラートをそのフォーカル・ポイントに転送します。アラートの転送に関する詳細は、423 ページの『第 26 章 集中操作』を参照してください。

TECROUTE フィルターを使用すると、指定イベント・マネージャーに転送するアラートを選択できます。デフォルトの TECROUTE フィルターは BLOCK で、この場合、アラートは転送されません。

TRAPROUT フィルターを使用すると、SNMP マネージャーに転送するアラートを選択できます。デフォルトの TRAPROUT フィルターは BLOCK で、この場合、アラートは SNMP に転送されません。

DFILTER コマンドを使用すると、SRFILTER コマンドで確立されたフィルターを表示できます。SRFILTER、DFILTER、およびその他のハードウェア・モニターのフィルター操作のトピックに関する詳細は、NetView のオンライン・ヘルプを参照してください。 367 ページの『MSU を自動化する自動化テーブル・ステートメントの作成』では、特定レコードを選択する自動化テーブル・ステートメントのコーディング方法の例を記載します。自動化テーブルからのフィルター操作を制御するために使用する SRF、COLOR、XHILITE、および HIGHINT の各アクションの詳細については、240 ページの『処置』を参照してください。

目視フィルター

目視フィルターを使用すると、表示するアラートが限定されるので、ハードウェア・モニターのオペレーターは、ネットワークの特定部分または特定タイプのアラートに集中できます。表示する NetView イベントおよびアラートの選択は、ハードウェア・モニターの SVFILTER(SVF) コマンドを使って行います。目視フィルターの設定値は、DFILTER コマンドで表示できます。

SVFILTER コマンドは、このコマンドを実行している OST の実行されているオペレーターの表示にのみ影響します。SVFILTER コマンドを使うと、システムまたはネットワークのモニターを担当しているオペレーターはフィルターを使用して、余分なアラート・レコードを排除できます。それによって、オペレーターは担当している特定分野に集中できるようになります。

SVFILTER コマンドは SRFILTER コマンド同様に、特定のイベント・タイプ、リソース名、もしくはリソース・タイプに影響を及ぼしたり、指定されたリソースに接続されているすべてのリソースに影響を及ぼすことができます。さらに、NetView プログラムがレコードを受信した時刻を目視フィルターの基準として使用できます。目視フィルターのデフォルトは、PASS です。

例えば、システム・パフォーマンスの面でサービス・レベルが契約を満足するかどうかを確認する専門の部署があるとしたします。その部署のオペレーターが、PERF 以外のすべてのイベント・タイプについて、目視フィルターを BLOCK に設定していると、その特定部署に影響するアラート (パフォーマンス・アラート) のみを表示させることができます。

目視フィルターを効果的に使うには、SVFILTER コマンドとそのさまざまなオプションの構文について精通してください。構文については NetView オンライン・ヘルプを参照してください。

フィルターの迂回

異常事態のときには、通常のフィルター操作を迂回する必要も出てきます。その場合、XITCI インストール・システム出口ルーチンを作成して、戻りコードを 252 にするか、自動化テーブルに XLO アクションを使用して、あるレコードについてのみ外部ログを指定してください。このようにすると、NetView プログラムは、そのレコードのすべてのフィルターを BLOCK に設定します。XITCI 戻りコードを 253 にしてもよく、この場合は、NetView プログラムにより ESREC フィルターは

PASS になりますが、その他のすべてのフィルターは BLOCK になります。通常のフィルター操作を変更することのできるインストール・システム出口には、このほかに DSIEX16B があります。インストール・システム出口については、「*IBM Tivoli NetView for z/OS*プログラミング:アセンブラー」および「*IBM Tivoli NetView for z/OS*プログラミング: PL/IおよびC」を参照してください。

第 20 章 コンソールの統合

コンソールを統合することによって、オペレーターがモニターしなければならないコンソールの数を減らせます。NetView、MVS マスター・コンソール、およびサブシステム・コンソールのそれぞれの操作は、1 つの NetView コマンド機能画面上で結合できます。オペレーターは、NetView コンソールから MVS コマンドを発行して、マスター・コンソール操作を実行できます。

例えば、フォーカル・ポイントである 1 つの MVS システムで複数の MVS システムの活動をモニターしている場合、すべてのシステムから来るメッセージを 1 つの画面に表示できます。フォーカル・ポイントの NetView コマンド機能には、複数のターゲット・システムからのメッセージが、オペレーティング・システムの種類にかかわらず、一定形式でオペレーターに表示されます。さらに、モニターに必要なコンソールの台数を、1 台に削減できます。423 ページの『第 26 章 集中操作』では、集中化されたフォーカル・ポイント・システムからのリモート・システムおよびネットワークの操作を説明します。

コンソールの統合方法

メッセージ改訂テーブル (MRT) またはメッセージ処理機能 (MPF) を使用してコンソールを統合することができます。どちらも、システム・メッセージを NetView プログラムに経路指定することができます。これらのメッセージは、サブシステム・インターフェースを介して NetView プログラムに流れます。(メッセージ改訂テーブルについて詳しくは、139 ページの『第 13 章 メッセージ改訂テーブル』を参照してください。)

NetView サブシステム機能は、MPF テーブルで AUTO(YES) または AUTO(token) のマークが付いているメッセージ、あるいは NETVONLY や REVISE("1" AUTOMATE) などの改訂テーブル・アクションの対象となっているメッセージをすべて送信します。メッセージを任意の NetView オペレーター・コンソールに経路指定するには、NetView 自動化テーブルを使用します。

端末アクセス機能 (TAF) を使用すると、特定のアプリケーションまたはサブシステムからそれらのマスター端末またはコンソールへのメッセージを代行受信できます。NetView プログラムをそのアプリケーションのコンソールのように扱い、NetView からコマンドを発行することができます。こうして、単一のコマンド機能画面から、多数のサブシステムまたはアプリケーションの管理が可能です。ここでのサブシステムには、CICS と IMS が含まれていますが、これらに限定されるわけではありません。TAF を使用してコンソールを統合する方法については、486 ページの表 17 を参照してください。

NetView と複数コンソール・サポート (MCS) コンソールとの相違点

自動化環境で NetView を介してシステム・メッセージとネットワーク・メッセージを表示する場合、特に MVS 環境では、コマンド機能画面とオペレーティング・システム・コンソールとで、メッセージの表示方法が異なる点に注意する必要があります。ここからは、これら 2 つの相違点について説明します。

画面の処理およびメッセージの表示位置

コマンド機能画面を使用する場合も、複数コンソール・サポート (MCS) コンソールを使用する場合も、新規の行は最後に表示されたメッセージより下の行に書き出されます。複数コンソール・サポート (MCS) コンソールの画面がいっぱいになると、最新メッセージで最下位行のメッセージ行が書き換えられ、メッセージ域全体を上シフトすることによって画面全体が書き換えられます。そのあと一番古い (削除可能) 行がなくなります。

コマンド機能の場合は、画面がいっぱいのときは、先頭メッセージ行に最新メッセージが表示されます。それは画面の最下位行である必要はありません。循環方式が使用されているため、個々のメッセージの位置は変わらず、破線によって、最新のメッセージが最も古いメッセージから区別されます。新しいメッセージが届くたびに、破線が画面の下方に移動し、一番古いメッセージから上書きされていきます (JES3 コンソールでの方法と同じです)。ENTER を押すと、パネルの破線より上の領域と下の領域が入れ替わります。

メッセージの行形式

複数コンソール・サポート・コンソールには、オプションで各メッセージの前にタイム・スタンプ、ならびに JOB、STC または TSO のいずれかの番号を入れることができます。番号はそれぞれ、そのメッセージを出したジョブ、開始タスク、もしくは TSO ユーザーを示すものです。NetView コマンド機能画面に画面形式 (SCRNFMT) 定義を使用すると、メッセージ接頭部をカスタマイズできます。メッセージ接頭部には、以下の項目も追加することができます。

- 可変長形式の日付
- ドメイン名
- MVS メッセージに対応するジョブ名とジョブ ID
- TAF セッション名

この画面形式定義を活動化するには、DEFAULTS コマンドか OVERRIDE コマンドを使用します。項目の完全なリスト、および DEFAULTS コマンドと OVERRIDE コマンドの詳細については、NetView オンライン・ヘルプを参照してください。

表示域機能

複数コンソール・サポート (MCS) コンソール・オペレーターは、MVS CONTROL (K) コマンドを使用して、コンソール特性を変更できます。NetView コマンド機能画面には行外表示域機能がないので、コマンド応答情報はすべて行内に表示されます (複数コマンド・サポートの場合の K A,NONE と同じ)。そのため、メッセージに複数行を要する場合やメッセージ・トラフィックが多い場合には、応答の一部が上書きされてしまうこともあります。オペレーターは、AUTOWRAP コマンドを使用して、NetView 画面のメッセージ・フローを制御することができます。

画面の最新表示

NetView コマンド機能画面には、時間制御 AUTOWRAP 機能が備わっています。この機能は、複数コンソール・サポート (MCS) コンソールで CONTROL コマンドに RTME パラメーターを使用する場合などに使用されるものです。オペレーターは AUTOWRAP NO コマンドを使用して、最新表示を停止できます。

接頭部コマンド名

CNMCMD メンバーで各コマンドを定義すると、オペレーターは、MVS 接頭部を付けずに MVS コマンドを発行できます。定義のサンプルは、CNMS6401、CNMS6402、および CNMS6403 サンプルの自動化サンプルにあります。ただし、MVSDISPLAY、VARY、および MODIFY は MVS 接頭部を付けずに、また CNMCMD メンバーでの事前設定なしで実行できます。

CNMDEF 定義を使用して、JES2 コマンドが MVS 接頭部なしで入力できるようになっている場合、入力形式は \$D J555 ではなく \$DJ555 でなければなりません。これは、NetView プログラムが、最初のトークンをコマンドとして使用するためです。CNMDEF 処理は、MVS 以外の代替接頭部を提供する場合にも使用できます。

メッセージの保留

アクション・メッセージは、WTOR であるか、MVSPARM.ActionDescCodes CNMSTYLE ステートメントで指定されているいずれかのメッセージに一致する記述子コードを使用してマークが付けられているメッセージです。自動操作が想定されていない場合は、MVS アクション・メッセージが受信され、オペレーターの画面に表示されると、自動化テーブル項目に HOLD(Y) が指定されている場合と同じような処理が行われます。逆に、HOLD(NO) ではなく DOMACTION(DELMMSG) または DOMACTION(AUTOMATE) を使用して、アクション・メッセージ以外のメッセージを自動化した場合にも、メッセージは保持されます。これらのメッセージ・タイプは、いずれも DOM によって削除できます。アクション・メッセージは、表示と同時に強調表示されます。前述のいずれの場合にも、追加のメッセージが到着したときにメッセージが画面をロールオフすることはありません。これらのいずれかのメッセージを保持するタスクで後続の MVS DOM 要求が受信された場合には、メッセージの強調表示は解除されます。また、オペレーターがいずれかの保留状態メッセージまたはアクション・メッセージを NetView 画面から削除したい場合は、メッセージのある行にカーソルを移動して ENTER キーを押してください。

自動化テーブルで実行できるアクションの説明については、163 ページの『第 15 章 自動化テーブル』の『アクション』のセクションを参照してください。

オペレーター・メッセージ削除 (DOM) 要求は、さらに NetView 間タスク (NNT) に渡されます。このため、フォーカル・ポイント・システムの関連するオペレーター端末タスク (OST) は、画面に保留されていたアクション・メッセージを自動的に除去します。そのあと OST は DOM を、その OST のドメイン内にあるすべての OST および NNT に再度経路指定し、いずれの経路指定先においてもメッセージが削除できるようにします。メッセージが経路指定されたあとに NNT セッションが中断した場合、DOM 経路指定は行われません。

注: NNT は、DOM の一部のタイプをサポートしていません。EMCS コンソールを使用する場合は、DOM(NORMAL) EMCS コンソール属性を使用してください。

色およびその他の強調表示属性

コマンド機能で使用する色を指定するには、以下のような方法があります。

注: NetView バージョン 5.2 で追加されたメッセージ改訂テーブル機能を使用すると、MPF テーブルで使用可能な機能のほとんどを、メッセージ改訂テーブルを使用して実行できます。このテーブルについては、30 ページの『メッセージ改訂テーブル』で説明しています。

- インストール・システム出口を使用すると、色を設定できます。
- MVS システム・メッセージは、MPF テーブルまたはメッセージ改訂テーブルに指定した色で送信されます。
- 自動化テーブルで COLOR、XHILITE、および HIGHINT の各アクションを使用すると、メッセージの色、強調表示、および輝度を設定できます。自動化テーブルのアクションで設定された色は、MPF テーブルまたはメッセージ改訂テーブルで指定された色をオーバーライドします。自動化テーブルの値を変更する際には、NetView プログラムを停止して再始動させる必要はありません。
- 画面形式定義を使用すると、コマンド域のような、NetView コマンド機能画面の特定フィールドの色を設定できます。画面形式定義ではさらに、アクション・メッセージ、通常メッセージ、および即時メッセージに対応する色を指定することもできます。画面形式定義は、DEFAULTS コマンドまたは OVERRIDE コマンドで活動化します。

NetView コマンド機能画面のカスタマイズ全般に関する説明については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。特定の画面形式定義ステートメントについては、「*IBM Tivoli NetView for z/OS*アドミニストレーション・リファレンス」を参照してください。項目の完全なリスト、および DEFAULTS コマンドと OVERRIDE コマンドの詳細については、NetView オンライン・ヘルプを参照してください。

注:

1. MPF テーブルの MVS システム・メッセージに関する色と強調表示は、メッセージの色と強調表示に関する画面形式定義をオーバーライドします。
2. 自動化テーブルの色と強調表示は、MPF の色の指定と色および強調表示に関する画面形式定義とをオーバーライドします。自動化テーブルの色と強調表示は、さらにインストール・システム出口 DSIEX02A またはインストール・システム出口 DSIEX17 の色と強調表示をオーバーライドします。
3. インストール・システム出口の色と強調表示は、MPF 色の指定と色に関する画面形式定義とをオーバーライドします。インストール・システム出口 DSIEX16 を使用すると、自動化テーブルの色と強調表示をオーバーライドします。
4. ネットワーク・ログをブラウズする場合、メッセージは、NetView コマンド機能画面に表示された色と同一の色では表示されません。

オペレーターは、個別に色と強調表示の属性を操作できます。例えば、自動化テーブルに一致するものがあり、そこで COLOR アクションが指定されている MVS シ

ステム・メッセージは、MPF テーブルに指定された輝度と強調表示で表示されます。このメッセージの色は、自動化テーブルに指定された色になります。

NetView コマンド機能画面の利点

NetView コマンド機能画面を使用すると、以下の利点があります。

- メッセージの色や接頭部を変更して、NetView コマンド機能画面をカスタマイズできます。
- NetView の BROWSE コマンドを使用して、オペレーターはシステム・ログ、ネットワーク・ログ、操作用コマンド・リスト、および NetView アプリケーションのパラメーター・ライブラリー・メンバーを直接表示させることができます。
NetView オペレーターは、権限を持っていれば SYS1.PARMLIB などの他のシステム・ライブラリーを表示させることができます。また、NetView プログラムが実行されているリモート・システム上で、これらのすべての項目を表示することもできます (BROWSE コマンドのヘルプを参照してください)。
- NetView ヘルプ機能では、コマンド構文、および各コマンドの使用についての情報ならびに VTAM 戻りコード情報が提供されます。オペレーターは、任意の NetView 端末からいつでもヘルプ情報にアクセスできます。特定のオペレーターに対して、カスタム・オンライン・ヘルプ情報を簡単に追加することもできます。
- NetView コマンド・セキュリティ・チェックリストおよびユーザー独自のコマンド・リストとコマンド・プロセッサを使用して、各オペレーターに適切なコマンドと機能を提供できます。
- NetView オペレーターが TSO、CICS および IMS などの他のアプリケーションとの TAF セッションを使用すると、アプリケーションごとにコンソールを分ける必要がなくなります。

自動タスクによる複数コンソール・サポート・コンソールの使用

AUTOTASK コマンドを使用すると、複数コンソール・サポート (MCS) コンソールと NetView 自動タスクを関連付けることができます。次に、複数コンソール・サポート (MCS) コンソールから NetView コマンドを入力して、自動タスク下で実行し、応答メッセージを受信できます。

複数コンソール・サポート (MCS) コンソールは、コンソールとして通常は受信するメッセージもすべて受信します。AUTOTASK コマンドで割り当てられたコンソールについては、次の事項を考慮してください。

- CONSOLxx メンバーにある複数コンソール・サポート (MCS) コンソールの定義、およびアクティブな MPFLSTxx の定義が適用されます。
- 表示域を使えるのは、特定の MVS コマンドに限られます。NetView コマンドによって、複数コンソール・サポート (MCS) コンソール状況表示域に出力が書き出されることはありません。それらの出力は、通常のメッセージ・トラフィックとして表示されます。
- 複数コンソール・サポート (MCS) コンソールで入力される NetView コマンドの先頭には、NetView サブシステムの指定文字ストリングを付けなければなりません。指定文字ストリングは CNMSTYLE メンバーの MVSPARM.Cmd.Designator ステートメントを使用して指定します。

- 自動化テーブルの HOLD(Y) および BEEP(Y) などの NetView 画面管理アクションは、対応する自動タスクのメッセージを表示する複数コンソール・サポート (MCS) コンソールには影響しません。同様に、INPUT や RETRIEVE などの NetView 画面管理コマンドは、EMCS コンソールには影響しません。
- BROWSE およびその他のフルスクリーン・アプリケーションは、自動タスクのもとでは使用できないため、複数コンソール・サポート (MCS) コンソールからは使用できません。
- 他のドメインからの WTOR およびアクション・メッセージにはコンソール特性 (HELD/HIGHLIGHT) がないため、システムからのメッセージと混同されることはありません。

第 21 章 コマンドの統合

コマンド・プロシーチャーで、複雑なプロセスまたは一連のコマンドを置き換えたり補足したりすると、コマンドを統合できます。コマンドを統合すると、あるプロセスを実行するために必要な入力量が減り、入力ミスも減らせます。また、特定アクションの実行や特定問題の解決のためにすべてのオペレーターが同じプロセスを使うようにすることもできます。さらに、自動化テーブルおよびタイマー・コマンドなどの自動化機能には、独自に開発したコマンド・プロシーチャーの一部を使用できるので、後に自動化をさらに進めることもできます。

簡単なコマンド・プロシーチャーの作成

コマンドを統合する最も簡単な方法は、REXX あるいは NetView コマンド・リスト言語でコマンド・リストを作成することです。パフォーマンスの影響を受けやすい長いプロシーチャーでは、PL/I または C で作成したコマンド・プロセッサを使用できます。まず、コマンド・リストの作成から始め、このコマンド・リストのテスト、デバッグ、および調整が済んでから、リストの一部をコマンド・プロセッサに変換します。119 ページの『第 10 章 コマンド・リストおよびコマンド・プロセッサ』では、コマンド・リスト、コマンド・プロセッサ、および各オペレーティング・システムで使用可能な言語について解説しています。コマンド・リストとコマンド・プロセッサの詳細については、NetView のカスタマイズ資料を参照してください。

コマンド・プロシーチャー作成の最初のステップとして、まずオペレーターが繰り返し行う一連のアクションを特定します。一連のアクションには、NetView コマンド、VTAM コマンド、およびシステム・コマンドの実行のいずれを組み込むこともできます。また、あるコマンドの結果のメッセージ受信待ちや、リソース状況の定期的チェックなども組み込めます。共通のオペレーター・アクションに関する情報は、オペレーター・プロシーチャー資料、システム・ログおよびネットワーク・ログ、それにオペレーターから得るようにしてください。

次に、先に特定した一連のアクションを達成するコマンド・プロシーチャーを次のように作成します。

- テキスト・エディター (ISPF など) を使用して、命令をファイルに書き込みます。情報をコーディングする方法については、NetView のカスタマイズ資料を参照してください。
- コマンド・リストの名前と同じメンバー名を持つコマンド・リストを作成している場合には、そのコマンド・リストを DSICLD データ・セットに入れてください。
- PL/I または C でコマンド・プロセッサを作成している場合は、コマンド・プロセッサをコンパイルし、STEPLIB データ・セットに連係編集します。そのプロシーチャーの CMDDEF ステートメントを CNMCMD に追加し、NetView プログラムを停止してから再始動して、この新しいステートメントを有効にします。

す。PL/I または C コマンド・プロセッサを NetView プログラムに定義する方法については、「*IBM Tivoli NetView for z/OS*プログラミング: PL/IおよびC」を参照してください。

簡単な例として、オペレーターが NCP を活動化するとき、図 88 のようなコマンドを使用しているとします。

```
V NET,ACT,ID=NCP1,LOAD=YES,LOADSTA=LINK1
```

図 88. コマンドによる NCP の活動化

もっと短いコマンドで同じ処理を実行するため、NetView コマンド・リスト言語で ACTNCP1 というコマンド・リストを作成すると、図 89 のようになります。

```
ACTNCP1 CLIST
&CONTROL ERR
*****
*                                     *
* ACTNCP1 - Activates NCP1          *
*                                     *
*****
V NET,ACT,ID=NCP1,LOAD=YES,LOADSTA=LINK1
&EXIT
```

図 89. NCP を活動化するためのコマンド・リスト・サンプル

このコマンド・リストを作成すると、オペレーターはコマンド全体を出す代わりに、コマンド ACTNCP1 (または独自に定義した任意のコマンド同義語) を出すだけで済みます。

この変更によりオペレーターが満足していれば、コマンド・リストを拡張できます。例えば、NCP の名前をパラメーターとして受け取るようにすれば、このリストはより汎用的なものとなります。また、NCP が正常に活動化されたことを検証することもできます。

将来の自動化への考慮

オペレーターのために作成するコマンド・プロシーチャーの多くは、後に自動化機能として使用できるようになります。例えば、EVERY というコマンドを使用すれば図 89 の ACTNCP1 コマンド・リストを毎日実行するようにスケジュールすることも可能です。

そのため、コマンド・プロシーチャーを作成する際には、自動化に適しているかどうかを考慮してください。例えば、オペレーターの入力を待つ間休止するものや、オペレーターの入力にフルスクリーン・パネルを使うものを自動タスクから使用するためには、先にコマンド・プロシーチャーを修正する必要があります。同様に、メッセージを待つコマンド・プロシーチャーでは、タイムアウトの値を指定しておくことで、予定していたメッセージが受信されなかった場合に自動タスクが無期限に待ち続けることがないようにする必要があります。自動化アクションの発生を検査できるように、自動化に使用するコマンド・プロシーチャーには監査証跡を組み込む必要があります。例えば、コマンド・プロシーチャーが実行されるたびに、あるメッセージをネットワーク・ログに送ることができます。

コマンド・プロシーチャーの修正

コマンド・プロシーチャーを実動環境で使用したあとに修正したい場合、別の名前にするか、別のデータ・セットにそのプロシーチャーをコピーしてください。そのプロシーチャーにテスト名を付けるかテスト・データ・セットにプロシーチャーを入れて、テストおよび調整をしたあとで、実動環境に移行してください。新規のプロシーチャーは、テストおよび調整が終わってから、実動環境のプロシーチャーと置き換えるようにします。この過程を踏むことにより、未テスト状態のプロシーチャーでのエラーが実動環境に影響を及ぼす事態を避けられます。

NetView プログラムが、コマンド・リストを見つけるために、コマンド・プロシーチャー・データ・セットを検索するときの順序を制御することができます。これは、NetView 開始プロシーチャーの中のデータ・セットを、検索したい順序に連結することによって可能になります。以下の順序を使用することを検討してください。

1. 実動データ・セット
2. テスト・データ・セット
3. 元のコマンド・リスト・データ・セット (CNMCLST)

コマンド・プロシーチャーの文書化

自動化環境では、多くの重要なオペレーター活動に代ってコマンド・プロシーチャーが使用されます。そのため、コマンド・プロシーチャーは理解しやすく保守しやすいように作成する必要があります。

すべてのコマンド・プロシーチャーのプロログには、表 11 に示す情報を一定形式で組み込むようにしてください。

表 11. コマンド・プロシーチャーの文書化

情報	理由
作成日	追跡のため
コマンド・プロシーチャーのソフトウェア・レベル	システム・ソフトウェア変更時の参照用
作成者	追跡のため
機能	そのプロシーチャーの機能がすぐわかるようにするため
予定される入出力	このプロシーチャーおよび他のプロシーチャーの更新時の参照用
使用される変数	このプロシーチャーおよび他のプロシーチャーの更新時の参照用
このプロシーチャーと相互呼び出し関係にあるプロシーチャー	一連の他のプロシーチャーをすぐに参照できるようにするため
変更処理	変更処理の制御を管理するため

例えば、359 ページの図 90 は、AOPIGUPD (CNME6401) コマンド・リスト内の文書化を表しています。AOPIGUPD は、拡張自動化サンプル・セットに組み込まれている共通コマンド・リストで、最初にアスタリスクのある行は、NetView コマン

ド・リスト言語部分の注釈を示します。 REXX 部分のコメントは、区切り文字 /*
で始まり、区切り文字 */ で終わります。

```

/*AOPIGUPD CLIST
&CONTROL ERR
*****
* (C) COPYRIGHT IBM CORP. 2010
* IEBCOPY SELECT MEMBER=((CNME6401,AOPIGUPD,R))
* LAST CHANGE: 12/18/10
*
* DESCRIPTION: SEE COMMENTS AT END OF SAMPLE
*
* CNME6401 CHANGED ACTIVITY:
* CHANGE CODE DATE DESCRIPTION
* -----
*****
* BUILD THE COMPLEX COMMON GLOBAL VARIABLE
* &COMPLEXVAR IS SET TO THE CONCATENATION OF THE FIRST THREE
* INPUT PARAMETERS
*****
&COMPLEXVAR = &CONCAT &1 &2
&COMPLEXVAR = &CONCAT &COMPLEXVAR &3
*****
* SET THE VALUE OF THE CONSTRUCTED GLOBAL VARIABLE
*****
&GLOBAL &COMPLEXVAR
&&COMPLEXVAR = &4
&EXIT
*****
* CLIST NAME : AOPIGUPD
* CATEGORY : INITIALIZATION UTILITY
* DESCRIPTION : SET COMMON GLOBAL VARIABLE VALUE FOR AN
* : AUTOMATED APPLICATION. THE DESIRED COMPLEX
* : GLOBAL VARIABLE IS CONSTRUCTED USING INPUT
* : PARAMETERS &1, &2, AND &3 THE CONSTRUCTED
* : GLOBAL VARIABLE IS THEN SET TO THE VALUE
* : PASSED IN INPUT PARAMETER &4
* INPUT PARMS : &1 - RESOURCE COMMON PREFIX (MAX. 5 CHARS)
* : &2 - FUNCTION IDENTIFIER (MAX. 3 CHARS)
* : &3 - APPLICATION IDENTIFIER (MAX. 3 CHARS)
* : &4 - DESIRED VARIABLE VALUE
* VARIABLES : &COMPLEXVAR IS A TEMPORARY VARIABLE TO HOLD
* : THE NAME OF THE CONSTRUCTED COMPLEX
* : GLOBAL VARIABLE.
* ACTION : VALUE SET FOR COMMON GLOBAL VARIABLE
* CALLING CLISTS : AOPIVARS
* CALLED CLISTS : NONE
*
*****
&EXIT
END OF CLIST */

/* REXX CONVERSIONS */

/* AOPIGUPD Command List */

TRACE E
/*****
/* (C) COPYRIGHT IBM Corp. 2010
/*****
/* Build the complex common global variable COMPLEXVAR and set
/* it to the concatenation of the first three input parameters
/*****
complexvar = MSGVAR(1)||MSGVAR(2)||MSGVAR(3)
/*****
/* Set the value of the constructed global variable
/*****
INTERPRET complexvar '= MSGVAR(4)'
'GLOBALV PUTC 'complexvar
EXIT
/*****
/* Command List Name : AOPIGUPD
/* Category : Initialization utility
/* Description : Set common global variable value for an
/* : automated application. The desired complex
/* : global variable is constructed using input
/* : parameters 1, 2, and 3. The constructed
/* : global variable is then set to the value
/* : passed in input parameter 4.
/* Input Parms : MSGVAR(1) - resource common prefix
/* : (max. 5 chars)
/* : MSGVAR(2) - function identifier
/* : (max. 3 chars)
/* : MSGVAR(3) - application identifier
/* : (max. 3 chars)
/* : MSGVAR(4) - desired variable value
/* Variables : COMPLEXVAR is a temporary variable to hold
/* : the name of the constructed complex
/* : global variable.
/* Action : Value set for common global variable
/* Called By : AOPIVARS
/* Calls : None
/*
/*****

```

図 90. サンプル・コマンド・プロシーチャー

第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化

本章では、NetView 自動化テーブルを使用して、共通メッセージおよび管理サービス単位 (MSU) の処理を自動化する方法について説明します。また、他の情報をまずメッセージまたは MSU に変換してから自動化する方法についても説明します。例えば、メッセージは、MSU 以外のハードウェア・モニター・レコードから作成することも、状況モニターや NetView 管理コンソールで検出された状況変更から作成することもできます。

本章に記載している自動化テーブル・ステートメントのうち、特定ステートメントをコーディングするために必要な構文の詳細については、163 ページの『第 15 章 自動化テーブル』を参照してください。

また、NetView プログラムの自動化オペレーション・ネットワーク (AON) コンポーネントを使用して、共通のメッセージおよび MSU の処理を自動化することができます。詳しくは、「499 ページの『第 31 章 自動化オペレーション・ネットワークの使用』」を参照してください。

決定、自動化するメッセージおよび MSU の

自動化すべきメッセージおよび MSU を識別するには、一般的な勤務時間帯の活動を記録しているシステム・ログおよびネットワーク・ログを、オペレーターとともに検討するのが良い方法です。

オペレーター・コンソールで一連の予測可能なコマンド入力が必要になるメッセージおよび MSU を識別するには、オペレーターの助言が役立ちます。そのようなメッセージには、例えば、REPLY コマンドが必要なメッセージ、ジョブを取り消す時間を示すメッセージ、リカバリー可能な装置障害を示すメッセージなどがあります。最初に、明らかに対象となるメッセージおよび MSU を探して、自動化してください。そのあとでまた実行環境についてオペレーターとともに検討し、別のメッセージおよび MSU のセットを自動化の対象として選択します。

MVS のログ分析プログラムは、メッセージ・ログにあるメッセージを分析するために使用することができます。523 ページの『ログ分析プログラム』を参照してください。自動化の対象となるメッセージを識別する場合、オペレーティング・システムのメッセージ処理機能を使ってシステム・メッセージを処理するようにしてください。ジョブ名とメッセージ ID の両方で特定したい、ネットワーク・メッセージや MVS システム上のメッセージなど、オペレーティング・システムが抑止できないメッセージを抑止するために、NetView プログラムを使用します。

自動化の対象となるメッセージおよび MSU を探す場合、オペレーターに問い合わせたりログを調べたりする以外にも、手順書、問題管理報告書、ヘルプ・デスクのログ、サービス・レベル契約書、およびユーザーなど、有用と思われる他のいくつかの情報源があります。これらの検討については、47 ページの『第 3 章 自動操作プロジェクトの定義』を参照してください。

メッセージには自動化できないものがあります。例えば、TYPE=FLASH を指定した DSIPSS マクロによって発行されるメッセージは表示されず、自動化できません。

自動化の対象となるメッセージおよび MSU を決定したあとは、自動化テーブル・ステートメントをコーディングしてそれらのメッセージおよび MSU を選択します。

メッセージを自動化する自動化テーブル・ステートメントの作成

メッセージの自動化を決定したあと、次のステップは、そのメッセージをいかに選択するか、つまりそのメッセージを自動化テーブルに記述する方法の決定です。このステップでは、対象のメッセージが別の類似メッセージと間違えられないようにする必要があります。自動化したいメッセージまたはメッセージ・クラスによって、メッセージ ID、他の特定基準、もしくは大きなメッセージ・クラスを選択する包括的な基準のいずれかを使用します。自動化に使用できるメッセージ属性には、メッセージが送信請求であるか非送信請求であるかの属性や、メッセージの発行元が許可プログラムであるか非許可プログラムであるかの属性など、数多くあります。

メッセージ ID による検査

多くの場合、メッセージの選択にはメッセージ ID を使用できます。メッセージ ID は、メッセージを他のメッセージと区別するキーとなっていて、通常はメッセージの先頭に付いています。MSGID キーワードを使用すると、メッセージをその ID に基づいて自動化できます。

例えば、NetView メッセージ DSI001I MESSAGE SENT TO taskid を選択して抑止したい場合、図 91 の IF-THEN ステートメントを使用することができます。

```
IF MSGID = 'DSI001I' THEN  
    DISPLAY(N);
```

図 91. メッセージ ID でメッセージを検査する例

メッセージ ID は、常にストリング中の最初のワードまたはトークンになっているとは限りません。MVS WTOR の場合は、メッセージの先頭に応答 ID が付いているために、メッセージ ID は 2 番目のトークンになっています。そのため、WTOR メッセージの 01 AHL125A RESPECIFY TRACE OPTION OR REPLY U では、AHL125A がメッセージ ID です。こうした場合にも、メッセージ ID を使用してメッセージを選択することは可能です。例えば、上記の WTOR メッセージを選択して抑止したい場合、図 92 のステートメントを指定します。

```
IF MSGID = 'AHL125A' THEN  
    DISPLAY(N);
```

図 92. メッセージ ID を使用して MVS WTOR メッセージを検査する例

自動化アクション・メッセージ

NetView によってアクション・メッセージというマークが付けられるメッセージには、DOM が関連付けられていない場合があります。このため、DOM が関連付けられていないメッセージについては、適切な自動化ステートメントを使用して、理由を説明するようにしてください。DOMACTION(NODELMSG) 自動化アクションでは、それに伴うストレージの使用時に DOM を待たずに済みます。

MVSPARM.ActionDescCodes CNMSTYLE ステートメントの詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。DOM に関する対処方法の詳細については、DOMACTION を参照してください。

代行手段としては、DEFAULTS コマンドの MAXCSSIR キーワードについてしきい値を指定する方法があります。これは、REXX プロシージャーを使用して、最も古く、最も重複の多いメッセージを、最も多くのメッセージを含むアドレス・スペースから除去する方法です。詳しくは、CNME1103 のサンプルを参照してください。

他の特定基準の検査

メッセージ ID だけでは、自動化の対象メッセージを識別するのに特定化が十分ではない場合があります。メッセージ・テキストの中に、特定アクションを行うメッセージを選択するための追加情報が含まれていることもあります。以下のセクションでは、DSI039I MSG FROM AUTOMGR : CHECKING AUTOTASK - AUTOJES のような、特定のメッセージ ID を持つメッセージの一部を選択する方法の例を示します。

前の例に出てくる DSI039I は、あるオペレーターまたは自動タスクが、MSG コマンドを出して、別のオペレーターまたは自動タスクに送信することのできるメッセージです。この場合、拡張自動化サンプル・セットの 1 つである AUTOMGR が、MSG コマンドを出しています。このメッセージを ID だけで選択したとしたり、MSG によって生成されたすべてのメッセージを自動化することになってしまいます。しかし、追加基準を使用すれば、自動化したい DSI039I メッセージのみを選択できます。

ドメイン ID によるメッセージの検査

使用可能な基準の 1 つに、ドメイン ID があります。例えば、ドメイン CNM01 から DSI039I メッセージをすべて選択し、そのコピーを図 93 のように MVS システム・ログに書き込むことができます。

```
IF MSGID = 'DSI039I' &  
  DOMAINID = 'CNM01' THEN  
  SYSLOG(Y);
```

図 93. ドメイン ID によってメッセージを検査する例

トークンによるメッセージの検査

トークンを使用して、自動化のためのメッセージを指定することもできます。NetView プログラムは、メッセージ・テキストを、ブランクすなわちスペースのあ

る位置で複数のトークンに分割します。例えば、メッセージ DSI039I MSG FROM AUTOMGR : CHECKING AUTOTASK - AUTOJES には、以下の 9 つのトークンがあります。

TOKEN(1)

DSI039I

TOKEN(2)

MSG

TOKEN(3)

FROM

TOKEN(4)

AUTOMGR

TOKEN(5)

:

TOKEN(6)

CHECKING

TOKEN(7)

AUTOTASK

TOKEN(8)

-

TOKEN(9)

AUTOJES

DSI039I メッセージの場合、TOKEN(4) はメッセージ・コマンドを出したタスクを指しています。図 94 のステートメントは、AUTOMGR から DSI039I メッセージをすべて選択しています。

```
IF MSGID='DSI039I' &  
  TOKEN(4)='AUTOMGR' THEN  
  SYSLOG(Y);
```

図 94. トークンを使用してメッセージのログを記録する例

位置によるメッセージの検査

メッセージ DSI039I では、メッセージの内容が 29 桁目から始まるように配列されています。そのため、メッセージの内容を取得するには、TEXT(29) を使用します。図 95 のように、内容の一部のみを指定すると、メッセージの選択は失敗します。

```
IF MSGID = 'DSI039I' &  
  TEXT(29) = 'CHECKING' THEN  
  SYSLOG(Y);
```

図 95. テキスト位置を使用してメッセージのログを記録する例

図 95 のステートメントでは、メッセージを選択することにはなりません。TEXT(29) は CHECKING AUTOTASK - AUTOJES であり、CHECKING のみではないためです。

プレースホルダーによるメッセージの検査

対照的に、図 96 のステートメントではプレースホルダー (.) を使用しており、サンプル・メッセージを選択しています。

```
IF MSGID = 'DSI039I' &  
  TEXT(29) = 'CHECKING' . THEN  
  SYSLOG(Y);
```

図 96. プレースホルダーを使用してメッセージのログを記録する例

このステートメントでは、29 桁目から始まるテキストと、プレースホルダーで表される任意のテキストが続く、ストリング CHECKING とを比較しています。

包括的な基準の検査

自動化ステートメントに包括的な基準を使用する場合、単一のメッセージ ID やドメイン ID の代わりに大きいメッセージ・グループを選択できます。ここでも、上記で使ったものと同じメッセージの DSI039I MSG FROM AUTOMGR : CHECKING AUTOTASK - AUTOJES のような特定のメッセージ ID を持つメッセージの一部を選択するにはどうしたらよいかの例を示します。

論理 AND 論理を使用する基準の検査

図 97 のステートメントでは、ストリング CNM で始まる名前を持つ任意のドメインからの DSI039I メッセージを選択しています。このステートメントは、これらのメッセージを、OPER1 がログオンしている場合には OPER1 に送信します。OPER1 がログオンしていない場合には、経路指定への影響はありません。このステートメントは、プレースホルダー (.) で示しているように、ドメイン ID とストリング CNM に任意のストリングが続いているものとを比較します。

```
IF MSGID = 'DSI039I' &  
  DOMAINID = 'CNM' . THEN  
  EXEC(ROUTE(ONE OPER1 *));
```

図 97. 論理 AND の論理を使用してメッセージを経路指定する例

論理 OR の論理を使用する基準の検査

論理 OR 論理を使用すると、IF 条件はさらに包括的なものになります。OR 条件を使うと、2 つ (またはそれ以上) の条件のうちあてはまるステートメントを選択することができます。図 98 のステートメントで、DSI039I メッセージが選択されるのは、このメッセージの発信元が AUTOMGR か SYSOP である場合です。

```
IF MSGID = 'DSI039I' &  
  (TOKEN(4) = 'AUTOMGR' | TOKEN(4) = 'SYSOP') THEN  
  EXEC(ROUTE(ONE OPER1 *));
```

図 98. 論理 OR 論理を使用してメッセージを経路指定する例

プレースホルダーを使用する基準の検査

プレースホルダーは、比較用のストリング値の前でもあとでも使用できます。図 99 のステートメントでは、29 桁目以降のどこかにストリング CHECKING が含まれているすべての DSI039I メッセージが選択されます。

```
IF MSGID = 'DSI039I' &  
  TEXT(29) = . 'CHECKING' . THEN  
  EXEC(ROUTE(ONE AUTO1));
```

図 99. プレースホルダーを使用するメッセージの経路指定の例

このように包括的な比較を使用する場合には、比較条件が包括的すぎると、処理するつもりがなかったメッセージまで自動化の対象になってしまうことがある点に注意してください。図 99 の IF 条件は、上記のメッセージの例 DSI039I MSG FROM AUTOMGR : CHECKING AUTOTASK - AUTOJES には一致しますが、同時に別のメッセージ DSI039I MSG FROM OPER1 : I AM CHECKING ON THE STATUS OF THE SPOOL UTILIZATION にも一致してしまいます。

このような事態を避けるため、メッセージのテキストのどこかにあるストリングに包括的な比較を使用する場合には、より特定の条件を必ず一緒に指定するようにしてください。図 99 の例では、MSGID='DSI039I' という条件が付加されているために、29 桁目以降にストリング CHECKING が含まれている他のメッセージで、条件が満たされないようになっています。

解析テンプレートを使用するテキストの比較

自動化したいメッセージを記述するとき解析テンプレートを使うと、柔軟性のあるテキスト比較を行うことができます。

例として、内容がいつも同じとは限らないメッセージを自動化したい場合を考えます。メッセージ内のデータは表示されるたびに異なる可能性があります、そのメッセージ内のデータ位置が、自動化対象メッセージとして識別する手掛かりとなります。

解析テンプレートでのプレースホルダーの使用

例えば、IEE362A SMF ENTER DUMP FOR SYS1.MANx ON volser というメッセージが出たときにはいつでも特定のアクションを起こしたいとします。

図 100 の IF-THEN ステートメントでは、このメッセージが自動化の対象として選択されるようになっています。このステートメントでは、プレースホルダー (ピリオド) を使って、予期できないテキストをスキップするようになっています。

```
IF MSGID='IEE362A' & TEXT= . 'FOR SYS1.MAN' . 'ON' . THEN  
  EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

図 100. 解析テンプレートでプレースホルダーを使用する例

このステートメントは、まずメッセージ ID が IEE362A であるか検査し、メッセージ・テキスト内のどこかにストリング FOR SYS1.MAN があり、次に任意ストリングのあとにストリング ON が続き、また任意ストリングが続いていることを検査します。このようなステートメントを使えば、ストリング全体のデータが正確には

わからなくても、長いテキスト・ストリングの検査が可能になります。プレースホルダーで示されるフィールドのサイズや内容は、認識する必要がありません。

解析テンプレートでの変数の使用

プレースホルダーの代わりに変数を使用すると、メッセージからデータを抽出できます。この場合、ステートメントのアクション部分に変数を使用して、抽出対象のデータを示します。例えば、366 ページの図 100 で IF 条件のコーディングに使用している 2 番目と 3 番目のプレースホルダーの代わりに、LIBIND という名前の変数と VOLSER という名前の変数を使用すると、図 101 のステートメントのようになります。

```
IF MSGID='IEE362A' & TEXT= . 'FOR SYS1.MAN' LIBIND 'ON' VOLSER THEN  
EXEC(CMD('CLISTA ' LIBIND ',' VOLSER) ROUTE(ONE AUTO1));
```

図 101. 解析テンプレートで変数を使用する例

変数 LIBIND には、メッセージの中のストリング FOR SYS1.MAN とストリング ON の間にあるなんらかのデータが格納され、変数 VOLSER には、ストリング ON のあと、メッセージの末尾まで続くなんらかのデータが入ります。

例えば、メッセージの IEE362A SMF ENTER DUMP FOR SYS1.MANX ON CPDLIB が出された場合、変数 LIBIND の値は X になり、変数 VOLSER の値は CPDLIB になります。

複数行メッセージの解析テンプレートの使用

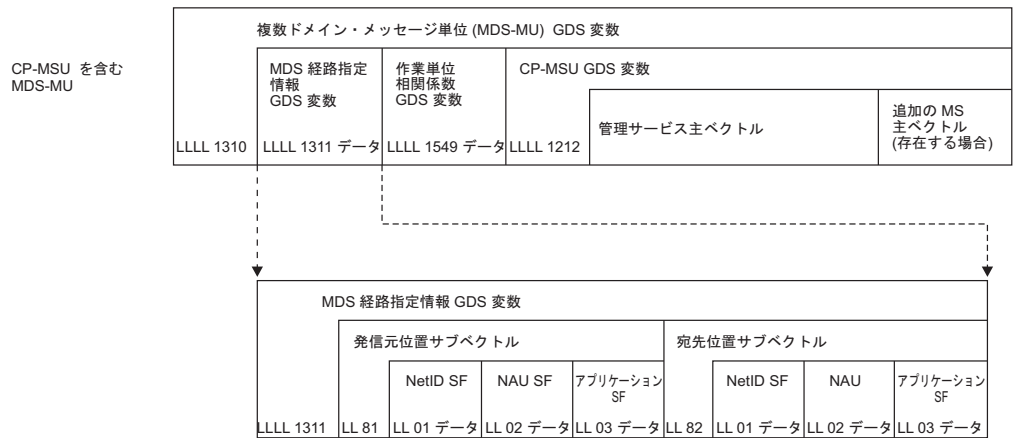
1 行だけのメッセージも、複数行のメッセージも、自動化の対象にできます。複数行メッセージに解析テンプレートを使用すると、メッセージのブランク行以外の最初の行からのみ情報の抽出が行われます。その他のメッセージ行から情報を抽出したい場合は、コマンド・プロシージャーを使う必要があります。複数行メッセージの例としては、NetView MAPCL コマンド、VTAM DISPLAY コマンド、MVS DISPLAY コマンド、いくつかの形式の JES2 \$D コマンドなどで出されるメッセージがあります。これらのメッセージに指定したアクションは、個々のメッセージ行をすべて含むメッセージ全体に適用されます。

MSU を自動化する自動化テーブル・ステートメントの作成

NetView 自動化テーブルを使用すると、管理サービス単位 (MSU) の処理を自動化できます。自動化可能な MSU タイプは、下記の 5 つです。

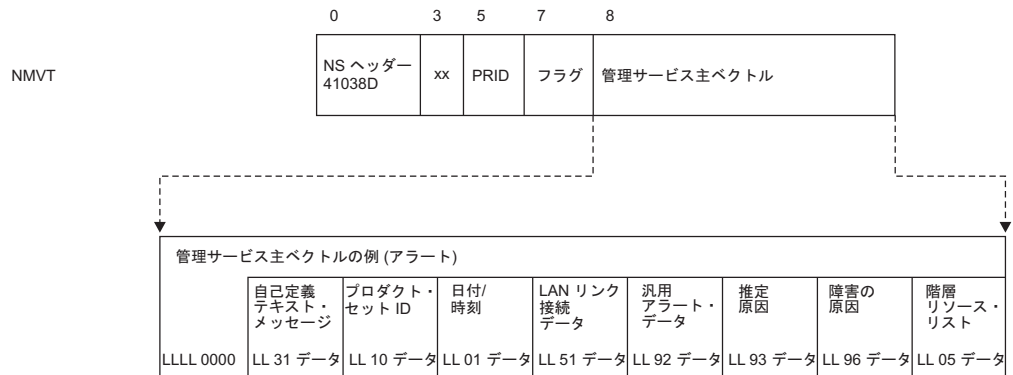
- NMVT
- CP-MSU
- MDS-MU
- RECMS
- RECFMS

368 ページの図 102 は、制御点管理サービス単位 (CP-MSU) を含む複数ドメイン・サポート・メッセージ単位 (MDS-MU) の構造を示しています。368 ページの図 103 は、ネットワーク管理ベクトル・トランスポート (NMVT) の構造を示しています。詳細については、システム・ネットワーク体系ライブラリーを参照してください。



すべての数字は、16 進形式です。
LL または LLLL は、構造体の長さを指す 1 バイトまたは 2 バイトのフィールドを表しています。

図 102. CP-MSU の概念図



すべての数字は、16 進形式です。
LL または LLLL は、構造体の長さを指す 1 バイトまたは 2 バイトのフィールドを表しています。

図 103. NMVT の概念図

運搬されるデータ (この場合はアラート) は、**管理サービス主ベクトル** と呼ばれる構造体にあるという点に注意してください。管理サービス主ベクトルには、以下の主ベクトルが含まれています。

- X'0000'
- X'0001'
- X'0002'
- X'0025'
- X'1332'
- X'1044' (カプセル化された RECMS)
- X'1045' (カプセル化された RECFMS)

CP-MSU の管理サービス主ベクトルの形式は、NMVT の管理サービス主ベクトルの形式と同じです。

RATE ステートメントを使用すると、リソースから反復 MSU を抑止することができます。RATE 機能の結果作成されたフィルターによってブロックされる MSU

は、自動化に渡されることはありません。これらの MSU を自動化したい場合、AUTORATE ステートメントを、BNJDSERV DST 初期設定メンバーに追加します。「IBM Tivoli NetView for z/OSカスタマイズ・ガイド」の RATE および AUTORATE ステートメントの項を参照してください。

以下の例では、ローカル・エリア・ネットワークから受信するアラートを自動化したいとします。アラートは、NMVT に入れられて NetView プログラムに送られますが、この NMVT を自動化の対象として選択することにします。まず、ハードウェア・モニターの Alerts-Static パネルに表示されるアラートのインスタンスを調べます。次に DM と入力して詳細メニューを表示し、1 を選択して NMVT を 16 進表示します。パネルを 2 画面か 3 画面進めると、図 104 に示すように、アラートの内容すべてを表示することができます。

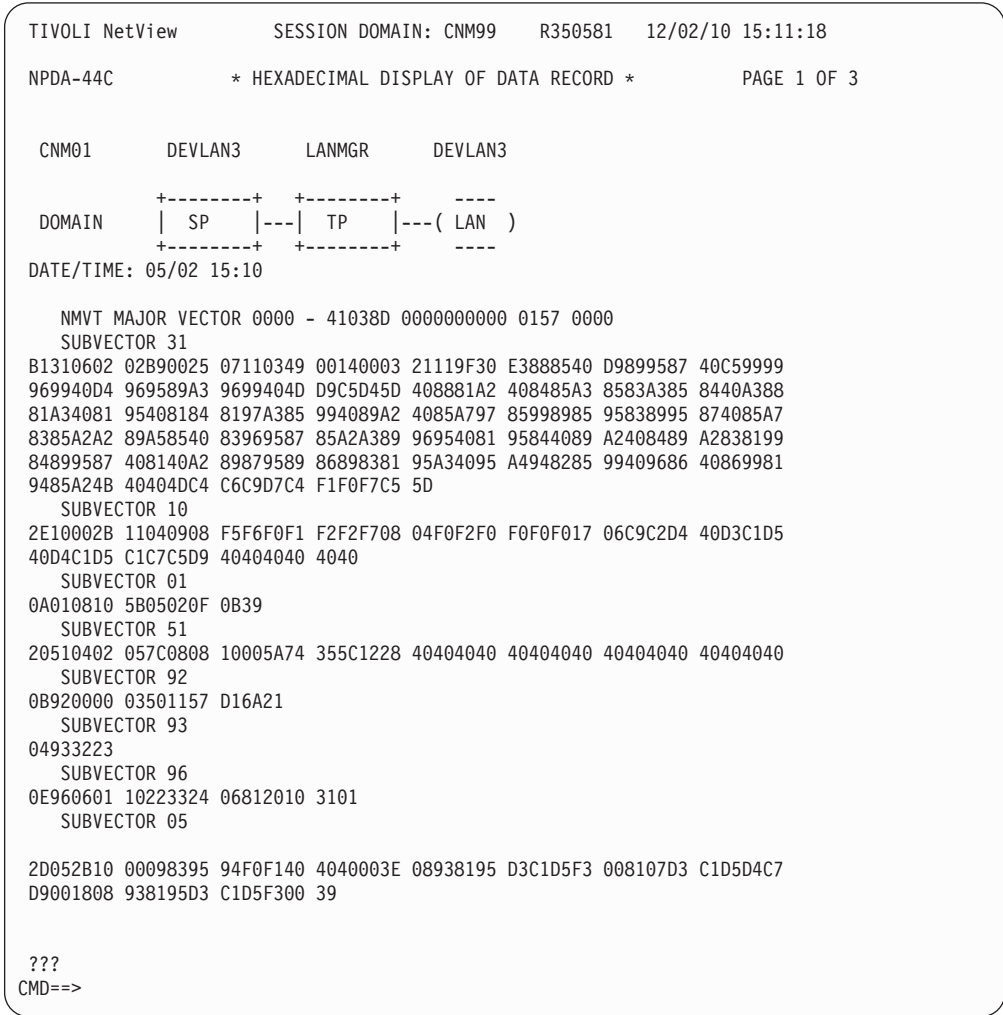


図 104. ハードウェア・モニターにおけるデータ・レコードの 16 進表示

フィールドの存在の検査

MSU を自動化の対象に選択するために、MSUSEG 比較項目を使用できます (構文規則については、MSUSEG を参照してください)。主ベクトル・レベルから開始し、下位レベルに移動します。例えば、ハードウェア・モニターを介して渡される

MSUの中からアラート主ベクトルを含むもの(キー X'0000'を持つ)をすべて選択し、緑色で表示させるには、図 105 の例のように指定します。

```
IF HMONMSU = '1' & MSUSEG(0000) ^= '' THEN  
    COLOR(GRE);
```

図 105. MSU の選択の例

この例では、アラート主ベクトルが存在する場合は MSUSEG(0000) によってその内容がすべて戻されますが、ヌル (") キーワードと結果を比較しているので、アラート主ベクトルの内容は無視して、単にそのような主ベクトルが存在するかどうかの検査のみをしています。

サブベクトルの検査

下位レベルに移動するには、ピリオド (.) 文字を使用します。主ベクトルの下位レベルは、サブベクトルです。X'31' サブベクトル (自己定義のテキスト・メッセージ) を含む X'0000' 主ベクトルのみを選択する場合は、図 106 に示すステートメントを使用することができます。

```
IF MSUSEG(0000.31) ^= '' THEN  
    COLOR(GRE);
```

図 106. サブベクトルの選択の例

サブフィールドの検査

図 107 に示されるように、サブフィールドやサブ・サブフィールドのレベルまで検査することもできます。

```
IF HMONMSU = '1' & MSUSEG(0000.51.02) ^= '' THEN  
    COLOR(GRE);
```

図 107. サブフィールドの選択の例

このステートメントの例では、ハードウェア・モニターを介して渡され、X'02' サブフィールド (リングまたはバスの ID) を含む X'51' サブベクトル (LAN リンク接続サブシステム・データ) を持つ X'0000' 主ベクトル (アラート) が含まれるすべての MSU が選択されます。

フィールド内容の検査

単に存在をテストするだけでなく、さらに特定化するため、特定フィールドの内容をテストすることもできます。例えば、ハードウェア・モニターを介して渡されるアラートのうち、推定原因 (サブベクトル X'93' に示される) が X'3223' になっているものをテストする場合が考えられます。推定原因 X'3223' は、「*Systems Network Architecture Formats*」の説明にあるように、トークンリング・アダプター・インターフェースを意味します。この原因によって出されたアラートを選択する場合、371 ページの図 108 のステートメントのようにコーディングできます。

```
IF HMONMSU = '1' & MSUSEG(0000.93) = HEX('04933223') THEN
    COLOR(GRE);
```

図 108. MSU サブベクトルの内容を検査する例

この例で、MSUSEG は長さバイト (この場合は X'04') とキー (X'93') を含むサブベクトル X'93' の内容全体を返しますが、バイト位置を指定して、長さとキーはスキップすることができます。位置 1 はデフォルトで、長さバイトである 1 バイト目から比較を開始します。「*Systems Network Architecture Formats*」の表記では、0 が 1 バイト目を示すことになっているので、これとは異なります。図 109 のステートメントでは、位置 3 を使用しており、長さバイトとキー・バイトをスキップした残りのデータを、調べる対象にしています。

```
IF MSUSEG(0000.93 3) = HEX('3223') THEN
    COLOR(GRE);
```

図 109. MSU サブベクトルのある位置の内容を検査する例

自動化テーブルのステートメントでは、プレースホルダー (.) を使用したり、変数に値を割り当てることもできます。プレースホルダーと変数は、MSU に対してメッセージの場合と同じ働きをします。例えば、図 109 のステートメントでは、サブベクトル X'93' にデータ X'3223' が、確かに含まれているかどうかを検査しますが、図 110 のように末尾にプレースホルダーを追加すると、サブベクトルが単にデータ X'3223' で始まっているかどうかを検査することができます。

```
IF MSUSEG(0000.93 3) = HEX('3223') . THEN
    COLOR(GRE);
```

図 110. プレースホルダーを使用して MSU サブベクトルのある位置の内容を検査する例

RECMS と RECFMS の検査

ハードウェア・モニターが自動化のため RECMS と RECFMS を実行要求すると、RECMS と RECFMS は設計済み主ベクトルの中にカプセル化されます。X'1044' 主ベクトルは RECMS に使用されます。X'1045' 主ベクトルは RECFMS に使用されます。

RECMS 82

図 111 は、ハードウェア・モニターに受信された RECMS 82 (記録モード) の例です。

```
01038103C4000182F04000zzzz
```

図 111. RECMS 82

項目の意味:

X'010381'

RECMS のヘッダー

X'82' RECMS 82 を示します (バイト 8 を参照)。RECMS 82 は、オフセット 0 ではなくオフセット 1 で始まります。

F04000zzzz

RECMS 82 の残り

RECMS および RECFMS のレコード形式については、ネットワーク制御プログラム・ライブラリーを参照してください。

カプセル化された RECMS

この RECMS 82 は、ハードウェア・モニターに受信されると、主ベクトル X'1044' を含む CP-MSU 内にカプセル化されます。図 112 は、これを示す例です。

LLLL1212nnnn104401038103C4000182F04000zzzz

図 112. X'1044' にカプセル化された RECMS

項目の意味:

LLLL 2 バイトの長さを表します。この長さは、以下のものの合計になっています。

X'D' RECMS の長さです。これより長い場合もあります。

X'2' X'1044' の長さ。

X'2' nnnn の長さ

X'2' 1212 の長さ

X'2' LLLL の長さ。図 112 では、2 バイト長の LLLL は X'0015' です。

1212 CP-MSU を示します。

nnnn 2 バイトの長さを表します。この長さは、以下のものの合計になっています。

X'D' RECMS の長さです。これより長い場合もあります。

X'2' X'1044' の長さ。

X'2' nnnn の長さ。図 112 では、2 バイト長の nnnn は X'0011' です。

X'1044'

RECMS を示す主ベクトル・キー

X'010381'

RECMS のヘッダー

X'82' RECMS 82 を示します (バイト 16 を参照)。RECMS 82 は、オフセット 0 ではなくオフセット 1 で始まります。

F04000zzzz

RECMS 82 の残り

例: X'82' の記録モードを持つ RECMS の検査

```
IF MSUSEG (1044 12) = HEX('82') . THEN  
  COLOR(GRE);
```

この例では、12 バイト目が 82 である (RECMS X'82' を示す) ことを検査し、その場合は、MSU の色を緑にします。2 バイトの長さ、および CP-MSU の X'1212' は含まれません。

注: RECMS では、サブベクトル、サブフィールド、およびサブ・サブフィールドはサポートされていません。また、RECFMS では、限られた数のサブベクトルのみがサポートされています。そのため MSUSEG を使用しても、RECMS および RECFMS のサブベクトル、サブフィールド、サブ・サブフィールドのいずれのキーにもアクセスすることはできません。

MSU アクション

MSU に指定できるアクションには、EXEC アクションとともに使用する、コマンド、コマンド・リスト、コマンド・プロセッサの発行があります。EXEC は任意の MSU で指定することができます。

他のアクションは、ハードウェア・モニターがアラート进行处理する方法を制御します。これらのアクションは、アラート主ベクトルを含みハードウェア・モニターを介して渡される MSU にのみ意味があります。

アクションによっては、アラートの強調表示属性を設定するものもあります。

XHILITE

テキストの明滅、下線表示、反転表示などのフォアグラウンド強調表示オプションを設定します。

COLOR

カラー・モニター用に色を選択する場合に使用します。

HIGHINT

モノクローム・モニターに高輝度 3270 設定を使用するかどうかを指定します。

BEEP 音響警報を鳴らすかどうかを指定します。

以下のアクションは、記録操作を制御します。

SRF 記録フィルターの属性を設定し、MSU を ESREC、AREC、OPER、ROUTE、TECROUTE、および TRAPROUT のフィルターに通すかどうかを決定します。

- AREC フィルターを機能させるためには、ESREC フィルターを通過するように設定する必要があります。以下は、ESREC フィルターの設定方法です。
 - 自動化テーブル
 - SRFILTER コマンド
 - DSIEXI6B インストール・システム出口
- OPER、ROUTE、TECROUTE、および TRAPROUT フィルターを機能させるためには、ESREC フィルターと AREC フィルターを通過するように設定しなければなりません。
 - OPER フィルターは、メッセージの生成を制御します。
 - ROUTE フィルターは、アラートの転送を制御します。
 - TECROUTE フィルターは、指定されたイベント・サーバーへのアラートの転送を制御します。
 - TRAPROUT フィルターは、SNMP マネージャーへのアラートの転送を制御します。

XLO どの記録フィルターも有効にならず、MSU が外部ログに記録されるだけであることを指定します。

強調表示属性と記録操作属性を自動化テーブルに設定すると、ハードウェア・モニターでの指定内容がオーバーライドされます。例えば SRF アクションは、ハードウェア・モニター SRFILTER コマンドを無効にしますが、インストール・システム出口 DSIEX16B は、自動化テーブルでさえもオーバーライドできます。

16 進表記、文字表記、およびビット表記

MSU を使って作業する場合、通常は 16 進表記を使うのが便利です。しかし、ステートメントによっては、文字表記の方が適している場合もあります。MSU に文字データの EBCDIC 表示がある場合は、文字表記が便利です。

サンプル・アラートの、サブフィールド X'10' (階層名リスト) のサブベクトル X'05' (階層/リソース・リスト) には、EBCDIC 文字が含まれます。階層名リストをテストするには、16 進表記または文字表記のいずれかを使用できます。

16 進表記の使用

例えば、サンプル・アラートが、リストの最初のリソースに基づいてアラート・データベースに記録されないようにシャットダウンしたい場合、「SNA Formats」に説明されているように、最初のリソースはサブフィールド X'10' の位置 5 から始まるので、図 113 に示すように、16 進表記で MSUSEG ステートメントをコーディングできます。

```
IF MSUSEG(0000.05.10 5) = HEX('C3D5D4F0F1') . THEN
  SRF(ESREC PASS)
  SRF(AREC BLOCK);
```

図 113. 16 進表記の使用例

文字表記の使用

図 114 のように、同等の文字表記を使用することもできます。

```
IF MSUSEG(0000.05.10 5) = 'CNM01' . THEN
  SRF(ESREC PASS)
  SRF(AREC BLOCK);
```

図 114. 文字表記の使用例

ビット表記の使用

別のオプションは、ビット位置を指定する方法です。ビット位置を使う場合、比較の規則は変わり、式の右辺に指定する項目はビット・ストリングでなければなりません。バイト位置と同様、ビット位置は 0 ではなく 1 で始まります。375 ページの図 115 では、階層名リスト (サブフィールド X'10') をテストするために、ビット位置とビット・ストリングを使用しています。


```
IF MSUSEG (0000.05.10 5 1) = '1000001110010101100101001111000011110001' THEN
    SRF(ESREC PASS)
    SRF(AREC BLOCK);
```

図 115. ビット表記の使用例

ビット・ストリングの比較でテストされるのは、指定したビット数だけなので、図 115 にプレースホルダーを使用することはできません。比較で特定ビットをスキップしたい場合は、ビット・ストリングに X を使用することもできます。

位置の指定が常に 16 進数であるのに対し、バイト位置とビット位置の指定は常に 10 進数です。例えば 図 115 の場合、X'0000'、X'05'、および X'10' は 16 進数ですが、5 および 1 は 10 進数です。

フィールドが 2 度以上出現する場合

MSU に、特定の主ベクトル、サブベクトル、もしくはその他のフィールドが 2 度以上出現する場合があります。その場合、一度目以外のインスタンスを検査するには、オカレンス番号が必要です。

例えば 図 116 のステートメントでは、2 番目のサブベクトル X'10' (プロダクト・セット ID) にストリング IBM LAN マネージャーが入っている場合、アラートが強調表示されます。

```
IF MSUSEG(0000.10(2)) = . 'IBM LAN MANAGER' . THEN
    XHILITE(REV) COLOR(BLU) BEEP(YES);
```

図 116. フィールドの複数オカレンスを検査する例

369 ページの図 104 のサンプル・アラートでは、X'10' サブベクトルが 1 つしかないなので、テストは失敗します。しかし、すべての X'10' サブベクトルを一度に検査すると、サンプル・アラートはテストに成功します。これには、図 117 のようにオカレンス番号にアスタリスク (*) を使用します。

```
IF MSUSEG (0000.10(*)) = . 'IBM LAN MANAGER' . THEN
    XHILITE(REV) COLOR(BLU) BEEP(YES);
```

図 117. フィールドの全オカレンスを検査する例

図 117 では、アスタリスクによって、最初の主ベクトル X'0000' のいずれのサブベクトル X'10' についても、比較の評価は真になります。オカレンス番号およびアスタリスクは、主ベクトル・レベルやサブフィールド・レベルなどの他のレベルでも使用できます。しかし、ハードウェア・モニターを経由して送られる MSU に余分な主ベクトルがある場合は、自動化の前に NetView プログラムによって個々の MSU に分離されるので、有無を検査する必要はありません。

各レベルにおけるデフォルト値では、指定されたフィールドの最初のオカレンスのみを検査するようになっています。376 ページの図 118 のステートメントでは、任意の X'0000' 主ベクトルに X'10' サブベクトルが含まれ、1 番目のサブベクトルには X'11' サブフィールドが、2 番目のサブベクトルには X'00' サブ・サブフィールドが含まれているかどうかを検査されます。この例のステートメントにより、最初の X'00' サブフィールドが検査され、3 番目のバイトの第 4 ビットが 1 で始まり次が 0 であるかどうかを検査されます。

```
IF MSUSEG(0000(*).10.11(2).00 3 4) = '10' THEN
  EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

図 118. MSU フィールドを詳細に検査する例

369 ページの図 104 のサンプル・アラートでは、X'10' サブベクトルに X'11' サブフィールドが 1 つしかないので、図 118 のステートメントの条件は満たされません。

ヘッダー情報の使用

368 ページの図 102 は、MDS-MU に主ベクトル以外のさまざまなヘッダー情報があることを示しています。場合によっては、MDS-MU をそれらのヘッダー情報に基づいて自動化することが出てきます。

MDS-MU をヘッダー情報に基づいて自動化する場合は、MSUSEG の先頭に H を追加してください。H を使用するときも、MSUSEG の構文規則は同じです。しかし、最初に指定するフィールド・レベルは、主ベクトルではなく MDS-MU 内の GDS 変数のレベルです。そのため、主ベクトルの外から情報を得ることができます。

例えば、MDS 経路指定情報 GDS 変数 (X'1311')、宛先位置サブベクトル (X'82')、宛先アプリケーション・サブフィールド (X'03') にあるデータを調べることができます。図 119 のステートメントを指定すると、長さバイトとキー・バイトをスキップし、位置 3 から始まるデータを取得できます。

```
IF MSUSEG(H1311.82.03 3) = 'APPLA' THEN
  EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

図 119. MDS ヘッダーを検査する例

H パラメーターは、MDS-MU にのみ使用できます。MSUSEG(H) を使って処理される NMVT は、ヌルの値を戻します。これは、プログラム間インターフェースから渡される NMVT など、MDS-MU 内にない任意の CP-MSU の場合と同様です。そのため、以下のステートメントを入力すれば、MDS-MU に送られたアラート主ベクトルを検査できます。

```
IF MSUSEG(H1212.0000) ^= '' THEN
  EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

図 120. MDS-MU でアラート主ベクトルを検査する例

図 120 で、H1212 は MDS-MU 内の CP-MSU を選択し、0000 はアラート主ベクトルを検査します。

アラート以外の主ベクトルの使用

アラートは自動化対象として最も一般的な主ベクトルですが、他の主ベクトル (X'0001'、X'0002'、X'0025'、X'1332'、RECMS、RECFMS など) も自動化できます。

レゾリュション主ベクトルの検査

例えば、キー X'0002' を持つレゾリュション主ベクトルは、アラートで識別される問題が解決されたことを通知します。レゾリュション主ベクトルには、ハードウェア・モニター BNJDSERV の XITCI 出口からアクセスすることができ、自動化

処理のために自動化テーブルに転送されます。アラートの場合と同様、ハードウェア・モニターはレゾリュション主ベクトルを表示し、VSAM データベースにログを記録し、ハードウェア・モニター・フィルタでできるように (SRFILTER コマンドで設定) します。

各レゾリュション主ベクトルをトラップし、それを MSU 全体とともに CLISTA に送信したい場合を例にとって説明します。CLISTA は、レゾリュション・データを追跡するためにユーザーが作成したコマンド・リストであり、レゾリュション・データを NetView ブリッジを経由してソフトウェア情報/管理に送信する機能を持っているとします。以下のステートメントを入力することができます。

```
IF MSUSEG(0002) ^= '' THEN  
    EXEC(CMD('CLISTA') ROUTE(ONE AUTO1));
```

図 121. レゾリュション主ベクトルを自動化する例

R&TI GDS 変数の検査

運用管理機能のあるアプリケーションを使用して作業している場合、アプリケーションからハードウェア・モニターに送られた MDS-MU には、経路指定目標設定命令 (R&TI) 汎用データ・ストリーム (GDS) 変数 (X'154D') が含まれている可能性があります。ハードウェア・モニターはこの経路指定目標設定の情報を、まだ同じ CP-MSU にあるアラートまたはレゾリュション主ベクトルのあとに入れます。CP-MSU に、アラート主ベクトル (X'0000') ならびに経路指定目標設定命令 GDS 変数 (X'154D') が、起点アプリケーション名サブフィールド (X'60') とともに含まれていることを検査する場合を例にとって説明します。起点アプリケーション名を抽出して CLISTA に渡すには、以下のステートメントを入力することができます。

```
IF MSUSEG(0000) ^= '' & MSUSEG(154D.60) ^= '' &  
    MSUSEG(154D.60) = ORIGIN THEN  
    EXEC(CMD('CLISTA 'ORIGIN) ROUTE(ONE AUTO1));
```

図 122. 経路指定目標設定命令 GDS を自動化する例

ハードウェア・モニターは、X'0000'、X'0001'、X'0002'、X'0025'、X'1332'、RECMS (X'1044' にカプセル化)、および RECFMS (X'1045' にカプセル化) の各主ベクトルを自動化テーブルに送信します。このとき X'154D' GDS 変数が付加されていれば、これも送信します。しかし、NVAUTO MS アプリケーション、CNMAUTO サービス・ルーチン、または DSIAUTO マクロを使用すれば、任意の主ベクトルを自動化テーブルに送信できます。また、MSUSEG は、送られてきたすべての主ベクトルを処理し、1 つの CP-MSU に複数の主ベクトルを受け入れられます。主ベクトルは、有効な MSU 内に存在する必要があります。

リソース階層の使用

ハードウェア・モニターを経由してアラートが届いた場合、NetView プログラムはそのアラートについてのリソース階層を作成します。この階層は、5 つまでのリソースで構成されます。369 ページの図 104 でわかるように、サンプル・アラートの階層には 3 つのリソースがあります。

DEVLAN3

サービス・ポイント (SP)

LANMGR

トランザクション・プログラム (TP)

DEVLAN3

ローカル・エリア・ネットワーク (LAN)

リソース階層をテストするには、**HIER** キーワードを使用します。図 123 のようにリストの中にリソースの番号を指定すると、**HIER** は 8 文字の名前のあとに 4 文字のリソース・タイプを付けて戻します。

```
IF HIER(2) = 'LANMGR TP ' THEN  
  COLOR(GRE);
```

図 123. リソース階層のリソースを検査する例

図 123 では、スペースが重要な意味を持っています。TP を 9 桁目から開始するには、LANMGR のあとに 2 個のスペースが必要です。図 124 のステートメントでも、サンプル・アラートに一致します。

```
IF HIER(2) = 'LANMGR' . &  
  HIER(3) = 'DEVLAN3' . &  
  HIER(4) = ' ' THEN  
  COLOR(GRE);
```

図 124. リソース階層の複数リソースを検査する例

図 125 のようにリソース番号を省略すると、すべてのリソース名とリソース・タイプを連結したストリングが返されます。

```
IF HIER = 'DEVLAN3' . 'LANMGR' . 'DEVLAN3' . THEN  
  COLOR(GRE);
```

図 125. リソース階層のすべてのリソースを検査する例

サンプル・アラートのリソース階層は、サブベクトル X'05'、つまり階層/リソース・リストにある情報をもとに作成されています。そのため、MSUSEG(0000.05) からリソース階層情報を得ることもできます。しかし、これはどのアラートにも有効な方法とは限りません。ハードウェア・モニター・リソース階層をテストするのに最も確実なのは、**HIER** キーワードを使用する方法です。

ドメイン ID の使用

DOMAINID キーワードを使用すると、MSU を最初に受信した NetView ドメインが通知されます。**DOMAINID** の検査は一般的なテストなので、図 126 のように、他の条件と連結して使用してください。

```
IF MSUSEG(0000) ^= ' ' &  
  HIER(1) = 'DEVLAN3 SP ' &  
  DOMAINID = 'CNM01' THEN  
  COLOR(GRE);
```

図 126. **DOMAINID** キーワードの使用例

メッセージ生成による他のデータの自動化

自動化テーブルが処理するのは、メッセージと MSU ですが、NetView 自動化テーブルで処理されない、他のタイプのデータもあります。これらのタイプのデータに対する応答を自動化したい場合は、まず、タイプをメッセージまたは MSU に変換する必要があります。この処理を表す重要な例として、ハードウェア・モニター・データ・レコードの場合と状況情報の場合について説明します。

ハードウェア・モニター・レコードの自動化

ハードウェア・モニターに送信される問題の通知情報は、それらの通知情報からメッセージを生成して自動化に送ることで自動化できます。

ハードウェア・モニターに送信される問題レコードの多くは、MSU です。これらのレコードについては、自動化対象のメッセージを生成するか、MSU を直接自動化するかオプションがあります。直接自動化する方が効率はよくなりますが、これについては 367 ページの『MSU を自動化する自動化テーブル・ステートメントの作成』に説明があります。しかし、問題レコードには、OBR レコードや MDR レコードなど、他のタイプのものもあり、これらのレコードは自動化テーブルに送られません。これらの問題レコードも、メッセージを生成すれば自動化できます。

ハードウェア・モニターは、アラート・データベースが受け取るレコードごとに、次の 2 つのメッセージを生成します。

BNJ030I

そのデータベースがアラートを受信したことを示す

BNJ146I

アラートに関する情報

BNJ146I メッセージの方が多くの情報を含むため、自動化には通常こちらが使用されます。

OPER フィルターは、どのアラートがメッセージを生成するかを判別しますが、アラートが OPER フィルターを通過してメッセージを生成するには、その前に、ESREC フィルターと AREC フィルターを通過する必要があります。

アラートを自動化する場合、MSGID キーワードを使用してメッセージ BNJ146I を選択してください。BNJ146I のいくつかのフィールドを自動化の基本として使用する方法と、アラートを固有に識別するために十分なフィールドの小サブセットを自動化する方法があります。

NetView 自動化テーブルは、表示のためにメッセージを経路指定する場合だけでなく、BNJ146I メッセージを受信したときに 1 つ以上のコマンド・プロシーチャーを 1 つ以上の NetView タスクの下で実行するようにスケジュールする場合にも使用できます。例えば、あるモニター・オペレーターのタスクの下で実行するように、あるコマンド・プロシーチャーがスケジュールされると、このコマンド・プロシーチャーは BNJ146I メッセージを受信して処理し、それによってより詳しいメッセージがオペレーターのもとに書き込まれるようになります。別のコマンド・プロシーチャーでは、アラートの推奨アクションを自動化できます。

状況変更の自動化

状況モニターまたは NMC によって追跡されている状況に変更が生じた場合、自動化を起動できます。状況モニター初期設定メンバー DSICNM (CNMS5001) に SENDMSG ステートメントをコーディングすると、指定されたタイプのリソースで状況に変更があったときに、NetView プログラムにメッセージ CNM094I を出させることができます。SENDMSG ステートメントの詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

変更内容については、図 127 のように、CNM094I で表示されます。

```
CNM094I STATUS UPDATE FOR RESOURCE = resourcename IN NETWORK = netname  
FROM DOMAIN = domainname STATUS = status
```

図 127. CNM094I メッセージの形式

状況変更は、MSGID キーワードを使用して CNM094I メッセージを選択すると自動化できます。

自動化ステートメントの有効化

自動化ステートメントを使用可能にするには、すべての自動化ステートメントを DSIPARM ライブラリーのメンバーに入れる必要があります。メンバー名は、1 から 8 文字までの長さになります。

実動環境で使用している既存の自動化テーブルを変更する場合、変更前に新規のファイルまたはメンバーに該当テーブルをコピーしてください。新規の自動化テーブルを別ファイルで作成およびテストしている間は、まだ既存の自動化テーブルを実動で使い続けていても構いません。

自動化テーブルを活動化する前にそのステートメントが構文的に正しいことを検証するには、AUTOTBL コマンドに TEST キーワードを指定して出します。さらに LISTING キーワードを使用すれば、詳細なデバッグ情報を得られます。例えば、主に使用する自動化テーブルが ATABLE1 にある場合、図 128 のようにコマンドを出すことができます。

```
AUTOTBL MEMBER=ATABLE1,TEST,LISTING=LIST1
```

図 128. 自動化テーブルを検証する例

構文エラーがある場合は、エラーの発生しているレコードとそのエラーの種類を示すメッセージを受信することになります。このメッセージの情報により、テーブルの構文を訂正できます。

AUTOTEST コマンドを使用して、自動化テーブルの論理をテストすることができます。テストについて詳しくは、531 ページの『第 34 章 自動化テーブルのテスト』を参照してください。

AUTOTBL MEMBER=ATABLE1 を入力すると、テーブルを活動化することができます。自動化テーブルの構文エラーによって、予期しないアクションが発生することのないように、NetView プログラムではすべての構文が正しくない限り、テーブルを活動化しないようになっています。

別の DSIPARM メンバーをアクティブな自動化テーブルのリストに追加するには、AUTOTBL コマンドを使用して、新規のメンバーを挿入するリストの中の位置を指定します。例えば、メンバー DSITBL99 を 2 番目のメンバーとして、アクティブな自動化テーブル・メンバーのリストに挿入するには、次のコマンドを入力します。

```
AUTOTBL MEMBER=DSITBL99 AT=2
```

どの自動化テーブルがまだアクティブであるかを検査するには、STATUS キーワードを指定した AUTOTBL コマンドを次のように使用します。

```
AUTOTBL STATUS
```

AUTOMAN コマンドでは、フルスクリーン・パネル・インターフェースを備えているので、ユーザーは以下の事項が使用可能になります。

- 単一または複数の自動化テーブルを表示および管理する
- 個々の自動化テーブルまたはステートメントを使用可能または使用不可にする
- 既存テーブルとその状況を表示する

詳しくは、287 ページの『複数の自動化テーブルの管理』を参照してください。

関連エンジンを使用したメッセージおよび MSU の関連付け

メッセージおよび MSU のコピーをイベント形式に変換し、それを NetView アドレス・スペースの外側で実行中の自動化関連エンジンに経路指定できます。この関連エンジンを使用すると、異なるメッセージと MSU を、指定した基準に従って関連付けることができます。例えば、異なる ID を持つメッセージ同士またはメッセージと MSU を自動操作のために関連付けられます。

関連エンジンの使用により、タイマーと状態変数の管理が単純化されます。時間の経過に伴って発生する複数のイベントを関連付けることができます。例えば、自動操作を使用して、3 つの異なるメッセージが 10 分間隔の間に発行されたかどうかを検査するとします。状態変数を使用して各メッセージが発行されたことを記録し、タイマーを設定して 10 分が経過した時点で 3 つの状態変数がすべて設定済みになっていることを検証できます。関連エンジンを使用すれば、これを単一の関連ルールを使用して簡単にコーディングできます。

関連プロセスは、自動化テーブルとリンクされます。このプロセスは、自動化テーブル・アクションの実行時にコマンドとして起動されます。関連プロセスからの出力は、再び自動化テーブルによる処理を受けて関連付けられたメッセージまたは MSU の中のいずれか (1 つ以上) です。出力を得たら、自動化テーブル条件を使用できます。これらの条件を使用すると、関連処理済みの出力を識別したり、自動化テーブル・アクションを使用して最終処理を実行したりできます。

関連エンジンのインストールについては、「*IBM Tivoli NetView for z/OS* インストール:追加コンポーネントの構成」を参照してください。

関連の概要

以下に、関連プロセスの概要を示します。

- メッセージまたは MSU は、自動化テーブルによる処理を受けます。自動化テーブル条件 (CORRELATED='0') を使用すると、このメッセージまたは MSU がすでに関連処理を受けたかどうかをテストできます。
- メッセージまたは MSU がまだ関連付けられていない場合は、そのメッセージまたは MSU のコピーが作成され、そのコピーからイベントが作成されます。そのイベントは、COREVENT パイプ・ステージまたは CNMCRMSG コマンド・リストを使用して関連エンジンに送られます。関連エンジンに送られるイベントには、メッセージまたは MSU からの情報の、事前定義によるサブセットが含まれています。PIPE EDIT ステージまたは CNMCRMSG コマンド・リストを使用して、追加情報を含めることもできます。オリジナルのメッセージまたは MSU は、廃棄される場合と、引き続き自動化テーブル処理の対象としてさらなる処理を受ける場合があります。
- 次に、イベントが関連エンジンによって処理されます。イベントは、XML 文書から作成されたルール・ベースに照らして検査されます。イベントがそのルールの基準を満たしている場合は、ルールのタイプと指定したオプションに応じて、1 つ以上のイベントが NetView アドレス・スペースに戻されます。メッセージとイベントを正常に関連付けるためには、ルール・ベースと自動化テーブルとの間で整合性がとれていなければなりません。メッセージまたは MSU からの情報のうち、どの情報がイベントに含められるかは関連ルールによって決まります。
- 関連エンジンからイベントが戻されると、オリジナルのメッセージまたはアラートのコピーが検索されて、その処理が自動化テーブルに対して再実行依頼されます。自動化テーブルは、CORRELATED='1' であるかどうかをテストすることで、この再実行依頼を検出できます。その後、自動化テーブル・アクションを使用して関連関係の検出を処理できます。
- 関連処理済みの出力に対する自動化テーブル処理が終了したら、COREVTDA PIPE ステージを使用して戻されたイベント・データにアクセスできます。イベントの内容を調べて、関連処理中にユーザー作成のコードによって変更が加えられなかったかどうかを確認できます。COREVTDA ステージでは、各スロットとそれぞれが MLWTO の 1 行となる各値を使用して、2 次出力ストリーム上に MLWTO が作成されます。

以下に、無効なコマンド・メッセージを検出する自動化テーブル項目のサンプルを示します。このサンプルでは、オリジナル・メッセージはそのコピーが関連プロセスに送られるときに抑制されます。

```
IF MSGID='DSI002I' THEN
  BEGIN;
  IF CORRELATED='0' THEN
    EXEC(CMD('CNMCRMSG INVALID'))
    DISPLAY(N)
    NETLOG(N)
    SYSLOG(N)
    CONTINUE(N);
  IF CORRELATED='1' THEN
    EXEC(ROUTE(NETOP))
    CONTINUE(N);
  END;
```

CORRFAIL 条件を使用して、関連プロセスに問題があるかどうかを判別できます。以下の例では CORRFAIL 条件を最初にテストすることに注意してください。CORRELATED 条件では、関連プロセスに障害がある場合には、常に '0' が戻されます。

```

IF MSGID='DSI002I' THEN
  BEGIN;
  IF CORRFAIL='1' THEN
    EXEC(CMD('MSG NETOP CORRELATION FAILED'))
    CONTINUE(N);
  IF CORRELATED='0' THEN
    EXEC(CMD('CNMCRMSG INVALID'))
    DISPLAY(N)
    NETLOG(N)
    SYSLOG(N)
    CONTINUE(N);
  IF CORRELATED='1' THEN
    EXEC(ROUTE(NETOP))
    CONTINUE(N);
END;

```

ストレージに関する考慮事項

メッセージおよび MSU は、次のように処理されます。

- 自動化テーブルによって処理されたメッセージおよび MSU を、NetView プログラムが関連エンジン用にイベントに変換します。この変換が発生すると、オリジナルのメッセージまたは MSU は NetView プログラムによって特殊なキューに入れられて、相関ルールが発動されたときに対応するイベントと突き合わされるのを待つことになります。
- NetView プログラムがオリジナルのメッセージまたは MSU をこのキューに保持する期間は、CNMSTYLE メンバーのステートメントによって制御されます。メッセージまたは MSU が制限時間を超えたために廃棄された場合、NetView プログラムはキューに一致するものがなくても相関イベントを受け取れます。この場合、NetView プログラムは新しいメッセージまたは MSU の作成を試行します。

相関イベント形式に変換されるのは、オリジナルのメッセージまたは MSU の内容のサブセットのみです。そのため、NetView プログラムにイベントが送り返されたときに再構成されるメッセージまたは MSU は、オリジナルの正確なコピーではありません。MSU の場合は、その MSU を NetView プログラムに送るときに使用された転送メカニズムに関する情報の一部が失われる可能性があります。メッセージの場合は、**AIFR** およびメッセージ・データ・ブロックの情報が失われます。再構成されたメッセージには、**AIFR** の時刻フィールド、およびジョブ名とジョブ番号のフィールドが含まれています。復元されたメッセージまたは MSU でも十分に事足りる場合は、COREVENT パイプ・ステージでタイマー値 0 を指定して、オリジナルのメッセージまたは MSU は廃棄してもかまわないことを指示できます。このことは、相関のために使用されるストレージの量を削減するのに役立つ可能性があります。

別の理由から、値 0 を指定する場合もあります。例えば、イベント相関ルールを使用している場合でもイベントを変更することがあります。イベント内のメッセージ・テキストを変更し、そのイベントを NetView プログラムに戻した場合、オリジナルのメッセージがキューから検索されてしまうと、行ったメッセージへの変更は使用されないことになります。

関連処理

NetView プログラムは、メッセージまたは MSU からデータを取得して、関連エンジン用にイベント（名前と値の対）を作成します。相関エンジンは、メッセージまたは MSU を直接には処理しません。

相関論理は、XML ファイルに記述された一連のルールとして示されます。メッセージまたは MSU から取得された情報がイベントの属性となり、これは XML ファイル内のルールに従ってテストできます。

COREVENT および CNMCRMSG を使用した相関イベントの作成

メッセージおよびアラートは、COREVENT パイプ・ステージを使用して相関エンジンに送られます。

COREVENT ステージを使用すると、作成したイベントの eventType 属性を設定でき、さらにオプションで、一致する相関イベントが戻されるのを待ってオリジナルのメッセージまたはアラートをキューに入れておく期間を指定できます。

COREVENT ステージの構文については、「*IBM Tivoli NetView for z/OS*プログラミング:パイプ」またはオンライン・ヘルプを参照してください。

NAMEBIND の編集オーダーを使用して、スロットと値の対を出力イベントに追加できます。以下に示す例では、パイプが現行メッセージを取得し、それを USERDATA というユーザー定義のスロットを使用して相関エンジンに送ります。eventType 属性は SAMPEVENT です。

```
'PIPE (NAME CORSAMP)',
'| SAFE *',          /* copy complete message into pipeline */
'| EDIT',            /* begin edit */
'| COPY *',          /* copy complete message to EDIT output*/
'/SAMPLE DATA/' 1,  /* start value one: variable value */
'NAMEBIND /USERDATA/', /* create output line for the new slot */
'| COREVENT SAMPEVENT' /* transfer event to correlation */
```

COREVENT ステージを使用する前に、DSICORSV タスクがアクティブになっていることを確認してください。DSICORSV との通信が失敗すると、エラー・メッセージが 2 次ストリームに書き込まれます。

CNMCRMSG コマンド・リストを使用して COREVENT パイプ・ステージを作成することもできます。CNMCRMSG コマンド・リストを使用してデータを追加できます。最初のパラメーターは、eventType を示します。後続のパラメーターは、対になっていなければなりません。対のうち、最初のパラメーターが名前で、2 番目が値です。以下のコマンドは、上記の例に示されているパイプと等価です。

```
CNMCRMSG SAMPEVENT USERDATA SAMPLE DATA
```

SAMPLE DATA が複数のトークンとして表されていることに注意してください。名前と値の対はコンマで区切る必要があります。

以下の例では、USER1 用の追加スロットをイベントに追加します。

```
CNMCRMSG SAMPEVENT USERDATA SAMPLE DATA,USER1 SOMEDATA
```

以下の例では、タイムアウト値として 0 秒を指定します。

```
CNMCRMSG 'SAMPEVENT 0' USERDATA SAMPLE DATA,USER1 SOMEDATA
```

オプションの時間間隔は、イベント・タイプ・パラメーターと一緒に引用符で囲む必要があることに注意してください。

以下の例では、大/小文字混合の値を使用します。

```
CNMCRMSG SAMPEVENT 'UserData' 'A Mixed Case Value'
```

REXX 関数をパラメーターとして使用できます。例えば、ある MSU 用のエントリー・ポイント・ネットワークを含む EPNET のユーザー・スロットを組み込む場合は、次のように入力します。

```
CNMCRMSG SAMPEVENT EPNET HMEPNET()
```

メッセージおよび MSU からイベントへのマッピング

メッセージおよび MSU は、相関エンジンに送られる前に、相関エンジン・イベントに変換されます。イベント名と値の対のほとんどは、メッセージとアラートを Event Integration Facility (EIF) イベントに変換する際に使用されるデフォルト・マッピングの場合と同じです。イベント・マッピングについては、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。

表 12 に、一連のデフォルトのメッセージ属性をリストします。

表 12. デフォルトのメッセージ属性

属性名	説明
HOSTNAME	相関エンジンを実行中のホストの TCP 名。
ADAPTER_HOST_SNANODE	NetView NETID および NAU 名。
ORIGIN	メッセージの発信元である NetView ドメイン。
DATE	メッセージの日付。
MSGID	メッセージ ID。
MSG	メッセージの最初のテキスト行。
JOBNAME	メッセージのジョブ名 (使用可能な場合)。
SUB_ORIGIN	メッセージのジョブ番号 (使用可能な場合)。
SEVERITY	メッセージを重大度 CRITICAL、FATAL、または WARNING にマップします。メッセージを EIF イベントに変換する際に使用されるのと同じマッピングが使用されます。
TEXT_LINES	メッセージ内のテキスト行の数。MLWTO メッセージの場合は、このフィールドにはメッセージ内の総行数が含まれています。非 MLWTO メッセージの場合は、この属性の値は 1 です。
MLWTO2..MLWTO _n	MLWTO の場合。そのメッセージの 2 番目およびそれ以降の行です。2 番目の行は MLWTO2 に含まれ、3 番目の行は MLWTO3 に含まれます (以下同様)。メッセージの総行数は TEXT_LINES 属性に含まれています。
NV_OBJECT	値「MSG」。
EVENTTYPE	eventType 属性にユーザーが付けた名前。

表 13 に、一連のデフォルトの MSU 属性をリストします。

表 13. デフォルトの MSU 属性

属性名	説明
HOSTNAME	相関エンジンを実行中のホストの TCP 名。

表 13. デフォルトの MSU 属性 (続き)

属性名	説明
ORIGIN	階層名リストまたは階層/リソース・リスト・サブベクトルからの名前/タイプ階層対を含む文字ストリング。この文字ストリングには、次の形式の階層が含まれています。 <i>resnam1/typ1,resnam2/typ2,resnam/typ3, resnam4/typ4,resnam5/typ5</i> サブベクトル内での対番号のみが使用されます。
ADAPTER_HOST_SNANODE	NetView NETID および NAU 名。
DATE	NetView アラート・アダプターがアラートを受け取った日付。形式は、次のとおりです。 <i>MMM HH:MM:SS</i> 例: OCT 10 12:08:30
MSG	長いエラー説明 (問題を記述する長い推定原因メッセージ)。このメッセージは、ハードウェア・モニターの ALERTS-DYNAMIC パネルに表示される ALERT DESCRIPTION:PROBABLE CAUSE メッセージに似ています。
EVENT_TYPE	ハードウェア・モニターの EVENT DETAIL パネルに表示されるイベント・タイプ (例えば、PERMANENT または TEMPORARY)。汎用アラートの場合、これは汎用アラート・データ・サブベクトルのアラート・タイプ・バイトです。
SUB_ORIGIN	階層名リストまたは階層/リソース・リスト・サブベクトルから得られる名前/タイプ階層対リストの中の最後の対を含む文字ストリング。このストリングの形式は、次のとおりです。 <i>resnam_x/typ_x</i> ここで、 <i>x</i> はリスト内の最後の対の番号です。
ARCH_TYPE	GENERIC_ALERT 汎用アラート・データ・サブベクトルを含んでいる NMVT アラート主ベクトル GENERIC_RESOLUTION NMVT レゾリューション主ベクトル NONGENERIC_ALERT 上記以外のすべてのアラート
SEVERITY	汎用アラート・データ・サブベクトルからのアラート・タイプ・フィールド、または重大度の判別に使われるイベント・タイプ (例えば、FATAL、CRITICAL など)。詳細については、「 <i>IBM Tivoli NetView for z/OS</i> カスタマイズ・ガイド」を参照してください。

表 13. デフォルトの MSU 属性 (続き)

属性名	説明
EVENT_CORREL	MSU 相関サブベクトル 47 から抽出される相互係数。これらの相互係数は、アラートを他のアラートに関連付けます。例えば、ユーザーは同一の基本的な問題に関連するアラートを複数持つことができますが、それらのアラートはサブベクトル 47 によって関連付けられます。
INCIDENT_CORREL	インシデント ID サブベクトルから抽出される相互係数。これらの相互係数は、アラートをレゾリューションに関連付けます。
SELF_DEF_MSG	自己定義テキスト・メッセージ・サブベクトル 31 から抽出されるテキスト。
BLOCK_ID	非汎用アラートの場合に、当該アラートと関連した IBM ハードウェアまたはソフトウェアを識別するのに使用されるコード。
ACTION_CODE	非汎用アラートの場合に、事前定義された画面への索引を提供するコード。ブロック ID とアクション・コードの組み合わせで、これによって送信元のプロダクトが一意的に識別されます。
PRODUCT_ID	アラートまたはイベントの送信元のハードウェアまたはソフトウェア・プロダクト・セット ID (PSID)。これは、4 文字、5 文字、7 文字、または 9 文字のいずれかです。これは、すべての汎用アラートと一部の非汎用アラートに付属します。
ALERT_CDPT	汎用アラート・データ・サブベクトルのアラート記述コード・フィールドまたはレゾリューション・データ・サブベクトルのレゾリューション記述コード・フィールドから取得される 2 バイトの 16 進値。
ALERT_ID	非汎用アラート (レゾリューションも含む) の場合に、個々のアラート条件を指定するために送信元によって割り当てられる 8 桁の 16 進値。レゾリューション・アラートの場合は、この値が 00000000 となります。
MSU	文字フォーマットに変換されたアラート。(この属性は、NetView プログラムが内部的に使用するためのものです。)
DOMID	アラートを最初に処理した NetView ドメイン。(この属性は、NetView プログラムが内部的に使用するためのものです。)

状態相関によるフィルター操作

状態相関 は、着信イベントからの情報をモニターし、この情報をユーザー定義のパターンと関連付けます。状態相関は、着信イベントを分析し、ユーザー定義の状態に応じて、重複イベントを抑制し、イベントしきい値を識別し、類似のイベントを収集またはグループ化します。

状態相関を使用すると、類似のイベントを識別してそれらの情報を (可能であれば) サマリー・イベント に統合できるので、イベント・トラフィックを最小化するのに役立ちます。 1 つの要約に包含することができないイベントが単一イベント で

す。イベント・サーバーが受け取るのは、上記のサマリー・イベントと単一イベントです。どの要約の場合も、イベント・サーバーは余分な個々のイベントは受け取りません。

ルールの作成

相関は、状態ベースのルールおよび状態と無関係のルールを使用して実現します。これらのルールは、提供される DTD ファイル rule.dtd によって定義されている XML 構文を使用して指定します。デフォルトの XML ファイルは次の場所にあります。

```
/var/netview/v6r2/rulefiles/znvrules.xml
```

CORRSERV REFRESH を使用すると、このデフォルト・ファイルをオーバーライドできます。

表 14 に、XML ステートメントの要約を示します。

表 14. XML ステートメント

エレメントのタイプ	属性	機能
<rule>... </rule>		ルールの開始と終了を示します。
	id="identifier"	このルールに ID を関連付けます。相関エンジンがアクティブになっているときには、この ID を使用してこのルールを削除、非活動化、または活動化できます。
<eventType>... </eventType>		ルールによって処理されるイベントを制御します。
	event_identifier	イベントのイベント・タイプまたはクラスを指定します。

表 14. XML ステートメント (続き)

エレメントのタイプ	属性	機能
<code><correlation_type>...</code> <code></correlation_type></code>		以下の相関タイプのいずれかを指定します。 コレクター 指定された時間間隔の間に、指定されたルール基準を満たすイベントを収集します。その後、すべてのイベントが処理のために送り出されます。 duplicate ルールに指定されている基準を満たすイベントの中の重複するイベントを抑制します。 match あるイベントがルールに指定されている基準を満たしているかどうかを検査します。 passthrough 複数のイベントを検査して、指定された基準を満たしているイベントを識別します。 ResetOnMatch 複数のイベントを検査して、指定された基準を満たしていないイベントを識別します。 しきい値 ルール基準を満たすイベントの出現回数を検査し、指定された時間間隔の間に、指定された出現回数に達したかどうかを識別します。

表 14. XML ステートメント (続き)

エレメントのタイプ	属性	機能
<correlation_type> (続き)	randomOrder = "true" "false"	相関タイプ passthru および ResetOnMatch の場合に使用し、イベントの順序に意味があるかどうかを指定します。これを偽に設定した場合は、ルール内のイベントが述部によって予想される順序どおりに受信されないと、相関プロセスは開始されません。
	timeInterval = milliseconds	相関タイプ passthru および ResetOnMatch の場合に使用し、時間間隔をミリ秒単位で指定します。
	triggerMode = which_events	アクション・タグの処理対象にするイベントを指定します。 allEvents ルールによって処理されるすべてのイベント。 firstEvent 当該ルール用の相関処理を開始させるための最初のイベント。 lastEvent 当該ルール用の相関処理を開始させるための最後のイベント。 forwardEvents しきい値処理専用。これを指定すると、しきい値ルール処理ではすべてのイベントが戻され、さらに制限時間が満了するまでイベントの転送が継続されます。 注: このオプションをどのようにコーディングしたかが、自動化テーブルのコーディング方法に影響を与える可能性があります。発動されるルールは 1 つでも、結果的に、CORRELATED 属性が '1' に設定された複数のメッセージまたは MSU が処理のために自動化テーブルに送られることもあります。
<cloneable.../>		これを指定すると、相関エンジンは指定されたイベント属性別にイベントを分類し、それらのイベントのために別々の相関プロセスを保守します。このステートメントは、match 相関の場合には使用できません。
	attributeSet = "event_attributes"	イベント属性を指定します。

表 14. XML ステートメント (続き)

エレメントのタイプ	属性	機能
<predicate>... </predicate>		TRUE または FALSE に評価するブール・テストを指定します。 <predicate> タグ内では、イベント属性をテストするための比較演算が使用可能です。 相関タイプが異なれば、必要な述部の数も異なります。 match、duplicate、threshold、および collector 相関の場合は、必要な述部は 1 つです。 passthrough および ResetOnMatch 相関の場合は、複数の述部を必要とします。
<action.../>		ルール内のすべての述部が TRUE であった場合に実行するアクションを指定します。
	function = function_type	相関機能进行处理するために駆動される Java™ クラスの名前を識別します。 NetView プログラムは、デフォルトのアクションである ReturnToNV を提供します。ReturnToNV アクションは、イベントを NetView アドレス・スペースに戻します。このアドレス・スペースでは、オリジナルのメッセージまたは MSU がキューから検索されて自動化テーブルに送られます。オリジナルのメッセージまたは MSU が (タイムアウト条件のために) キュー内で検出されなかった場合は、NetView プログラムは戻されたイベントを使用して新しいメッセージまたは MSU を作成し、それを自動化テーブルに送ります。 NetView プログラムは、スーパークラス FLBCorAction を提供します。これをサブクラス化して使用すれば、このデフォルト・アクションを変更または置換できます。イベント属性を追加したり変更したりする場合は、ユーザー独自のアクションを作成すると便利です。

各ルールはそれぞれ 1 つの状態マシンに定義します。状態マシン は、関連知識のある特定のセットに関する情報を収集して要約します。状態マシンは、状態、遷移、要約、およびその他の特性 (有効期限タイマーや制御フラグなど) から構成されます。

状態ベースのルールは、コレクター・ルール、重複ルール、およびしきい値ルールです。これらはすべて、状態マシンを基にしています。各状態マシンは、それを開始するトリガー・イベントを探します。上記に加え、状態と無関係のルールである突き合わせルールがあります。

状態ベースのルールは、イベントのヒストリーに依存します。一方、状態と無関係のルールは、現行イベント 1 つに対して機能します。以下のもので、ルールを指定します。

- そのルールに関係のあるイベントを突き合わせるための述部
- ルールの起動後に実行されるアクション
- 属性 (しきい値限界など)

述部: 述部ライブラリーに含まれている述部は、ブール演算子とゼロ個以上の引数から構成されています。各引数は、次のタイプを返す述部にできます。

表 15. 述部のタイプと例

述部のタイプ	例
ブール値	同等
値を返す関数	加算
イベント属性	&hostname
定数	ストリング const01

処置: 状態相関の場合のデフォルト・アクションは ReturnToNV アクションです。このアクションでは、共通に使用される singleInstance という名前のオプションのブール属性がサポートされます。この属性が偽である場合は、アクションは異なるルール間で共用されません。アクションのインスタンスは、そのアクションのトリガーとなるそれぞれのルールごとに 1 つずつ作成されます。これがデフォルトの振る舞いです。この属性が真である場合は、アクションのインスタンスは 1 つだけ作成されて、そのアクションのトリガーとなるすべてのルール間で共用されます。

すべてのルールに共通の属性: 以下の属性は、すべてのルールで共通です。

id 各ルールの ID を指定します。ID は、登録先の相関エンジン内で固有でなければなりません。ピリオドはディレクトリーとして扱われます。例えば、id test.threshold を持っている場合は、test.threshold.1 を ID とする別のルールを持つことはできません。

eventType

このルールを適用するイベント・クラスのセットを指定し、パフォーマンスを最適化します。このパラメーターを省略すると、そのルールは状態相関によってすべてのイベント・クラスに適用されます。

突き合わせルール: 突き合わせルールは状態とは無関係です。これらのルールは、着信イベントの属性値に対して受動フィルター操作を実行します。突き合わせルールは、単一の述部から構成されます。その述部が真と評価されると、ルール内に指定されているトリガー・アクションが実行されます。

以下の例では、SAMPLE ルールが自動化テーブルによって送られた INVALID タイプのイベントを処理します (例えば、381 ページの『相関の概要』を参照してください)。SAMPLE ルールは、相関処理を実行依頼済みのイベントのメッセージ ID をチェックします。メッセージ ID が DSI002I であった場合は、そのイベントは NetView プログラムに戻されます。

```
<rule id="SAMPLE">
  <eventType>INVALID</eventType> 1
  <match> 2
```

```

<predicate> 3
  <![CDATA[
    &MSGID == "DSI002I"
  ]]>
</predicate>
</match>
<action function="ReturnToNV"/>
</rule>

```

以下では、上記の XML の例のステートメントについて説明します。

- 1 eventType は、サンプル CNMCRMSG コマンド・リストにパラメーターとして指定されています。
- 2 ルールが示す関連のタイプを定義します。この例では、ルールは指定された基準を満たすイベントを探します。
- 3 &MSGID 属性を検査して、ストリング DSI002I と同等かどうかを調べます。ルール内のすべての述部が TRUE であった場合は、ルール内で定義されているアクションがルールによって実行されます。

重複ルール: 重複ルールは、ある時間間隔の間、重複イベントの転送をブロックします。必要な引数は、次のとおりです。

- 状態相関がその間トリガー・イベントの重複をブロックする時間間隔。この間隔は、timeInterval 属性をミリ秒単位で指定して制御します。トリガー・イベントは、重複ルールによって検出される最初のイベントであり、実際には廃棄されない唯一のイベントです。
- トリガー・イベントの検出に使用される述部。

図 129 に重複ルールの場合の状態遷移を示します。

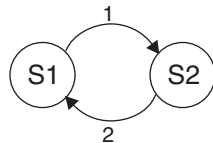


図 129. 重複ルールの場合の状態遷移

図 129 では、状態 1 が初期状態です。着信イベントに一致が見られると、遷移 1 が発生します。その時点で、その一致するイベントが状態相関によって転送され、タイマーが開始されます。時間間隔が満了すると遷移 2 が発生し、状態マシンがリセットされます。以下に、ルールの例を示します。

```

<!-- Show me only the first error number 10
for my hostname that happens each 10
seconds.
-->
<rule id="test.duplicate" >
  <eventType>NV_Error</eventType>
  <duplicate timeInterval="10000">
    <predicate>
      <![CDATA[
        &msg == "internal error " &&
        &hostname == "hostname1" &&
        &errno = 10
      ]]>
    </predicate>
  </duplicate>
</rule>

```

```

        </predicate>
      </duplicate>
      <action function="ReturnToNV"/>
    </rule>

```

しきい値ルール: しきい値ルールは、ある時間間隔の間、あるイベントが n 回出現するまで待ちます。出現回数がしきい値に達すると、イベントは定義済みのアクションに送られます。しきい値ルールは、以下のパラメーターを必要とします。

- triggerMode 属性によって指定される以下の送信モードのいずれか。

firstEvent

最初のイベントを送信します。

lastEvent

最後 (n 番目) のイベントを送信します。

allEvents

これがデフォルトのモードで、1 から n 番目までのすべてのイベントを送信します。

forwardEvents

リセットされるまで、 n 番目より後のすべてのイベントを送信します。

- その間にしきい値に達しなければならない時間間隔。この間隔は、timeInterval 属性をミリ秒単位で指定して制御します。
- その時間間隔が固定かどうかを示す時間間隔モード。このモードは、属性 timeIntervalMode=fixedWindow | slidingWindowinterval によって制御します。デフォルト値は fixedWindow です。
- 一致するイベントの数。thresholdCount 属性によって指定します。
- 1 から n 番目までのイベントの検出に使用されるトリガー述部。

図 130 および 395 ページの図 131 に、timeIntervalMode=fixedWindow を指定されたしきい値ルールの動作を示します。

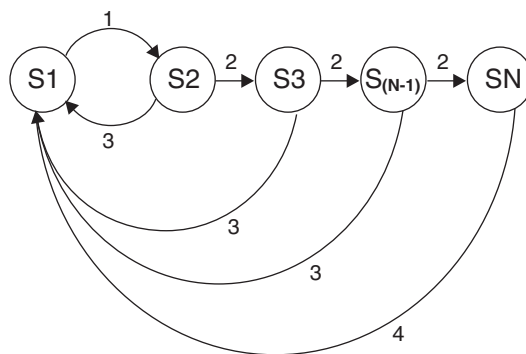


図 130. 基本的なしきい値ルールの場合の状態遷移

図 130 に、モード firstEvent、lastEvent、および allEvents 用の状態マシンを示します。状態相関がトリガー・イベントを検出すると、遷移 1 が発生します (トリガー述部の一致)。遷移 2 は、着信イベントが 2 番目の述部と一致したときに発生します。時間間隔が満了すると遷移 3 が発生し、状態マシンがリセットされます。しき

い値に達した後、遷移 4 によって状態マシンがリセットされます。状態 SN に達すると、リセットの前に、最初のイベント、最後のイベント、または n 個のイベントすべてが送信されます。

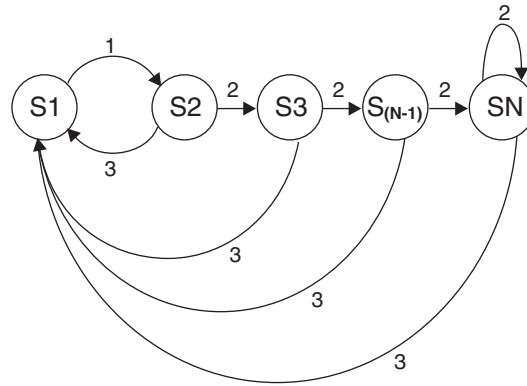


図 131. *forwardEvents* を使用するしきい値ルールの場合の状態遷移

forwardEvents モード (図 131) でのしきい値ルールの動作は、直前のケースでの動作と同様です。ただし、しきい値に達した後、時間間隔が満了するまで、2 番目の述部と一致するすべてのイベントが送信されるという点が異なります。

状態マシンに `timeIntervalMode=slidingWindow` が指定されている場合は、しきい値ルールの動作は `fixedWindow` 時間間隔の場合と同じです。ただし、各ノード K から遷移 1、2、... $K-1$ が発生するという点が異なります。この遷移アカウンタは、スライド式の時間ウィンドウには表示されないイベントから構成されます。以下に、ルールの例を示します。

```

<!--
I'm only interested when at least 5 Node_Down
events for hostnames in my local subnet happen
within 1 minute.
-->
<rule id="test.threshold">
  <eventType>Node_Down</eventType>
  <threshold thresholdCount="5" timeInterval="60000"
timeIntervalMode="slidingWindow" triggerMode="allEvents">
    <predicate>
      <![CDATA[
        (&msg == "node down") &&
        (isMemberOf(&hostname, [ 192.168./16 ]))
      ]]>
    </predicate>
  </threshold>
  <action function="ReturnToNV"/>
</rule>

```

しきい値ルールでは、イベントを単純にカウントする代わりに、複雑な総計値を定義することもできます。このルールは、`aggregate` 構成タグを使用して定義します。述部の定義に似た総計値を作成できます。総計値を使用するしきい値ルールは、その総計値が `thresholdCount` 値以上になった場合にのみ起動されます。以下に、ルールの例を示します。

```

<!--
If I receive a slot value with a relative percentage between
0 and 1, but I want to check my threshold using the normal

```


percentage value of 100%, I can define an aggregate of the slot relative_percentage, by multiplying it by 100 and counting all percentages until it reaches 100%.

```
-->
<rule id="test.aggregate_threshold">
  <eventType>Temperature_Variation</eventType>
  <threshold
    thresholdCount="100"
    timeInterval="2000"
    triggerMode="allEvents"
    timeIntervalMode="fixedWindow" >
    <aggregate>
      <![CDATA[
        &relative_percentage * 100
      ]]>
    </aggregate>
    <predicate>true</predicate>
  </threshold>
  <action function="ReturnToNV"/>
</rule>
```

コレクター・ルール: コレクター・ルールは、指定された期間、特定の述部に一致するイベントを収集します。このルールは、タイマーが満了すると起動され、収集したすべてのイベントを定義済みのアクションに送ります。コレクター・ルールの場合に必要な引数は、次のとおりです。

- 一致するイベントを収集する時間間隔。この間隔は、timeInterval 属性をミリ秒単位で指定して制御します。
- コレクションに追加する関連イベントに対するフィルター操作の一部を構成する述部。

図 132 に、コレクター・ルールの場合の状態遷移を示します。

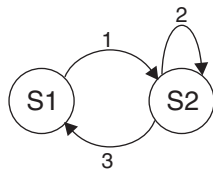


図 132. コレクター・ルールの場合の状態遷移

図 132 では、S1 (状態 1) が初期状態です。着信イベントに一致が見られると、遷移 1 が発生します。初期イベントは、送信されませんが、収集はされます。タイマーが指定された間隔に設定されます。タイマーが満了する前に、すべての着信イベントと一致イベントが収集されます (遷移 2)。時間間隔が満了すると遷移 3 が発生し、状態マシンがリセットされます。この時点で、収集済みのすべてのイベントが送信されます。以下に、ルールの例を示します。

```
<!--
Collects 10 seconds of Server_Down
events for my database.
-->
<rule id="test.collector">
  <eventType>Server_Down</eventType>
  <collector timeInterval="10000" >
    <predicate>
      <![CDATA[
        &servername == "my_database"
      ]]>
    </predicate>
  </collector>
  <action function="ReturnToNV"/>
</rule>
```

```

        </predicate>
    </collector>
    <action function="ReturnToNV"/>
</rule>

```

パススルー・ルール: パススルー・ルールは、指定された時間間隔の間に一連の特定イベントが到着した場合にのみ、トリガー・イベントを転送します。タイマーが満了する前に必要なイベントが到着すると (特定の到着順序の指定はオプション)、トリガー・イベントが転送されます。到着しなかった場合は、タイマーがリセットされ、トリガー・イベントは転送されません。

パススルー・ルールの場合に必要な引数は、次のとおりです。

- 必要なイベントの到着順序はどのような順序でも構わないか、指定した順序でなければならないかを指示するブール値 (randomOrder)。randomOrder が yes に等しい場合は、イベントはどのような順序で到着しても構いません。
- その満了後に状態マシンがリセットされる時間間隔。
- トリガー・イベントを定義するトリガー述部。これが状態マシンを初期化するイベントであり、必要な後続イベントが時間間隔内に到着すると転送されます。
- 必要な後続イベントを指定する 1 つ以上の述部。

図 133 に、パススルー・ルール (必要なイベントは順序どおりに到着しなければならないことを指定した (randomOrder=no) 場合) での状態遷移を示します。

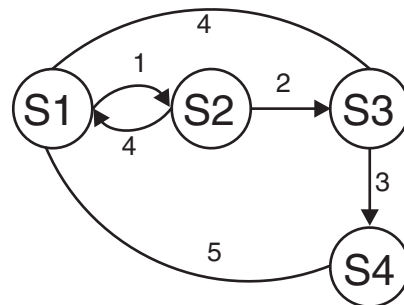


図 133. パススルー・ルール (randomOrder=no) の場合の状態遷移

図 133 では、S1 (状態 1) が初期状態です。トリガー・イベントが検出されると、遷移 1 が発生します。この遷移により、イベントが保管され、タイマーが開始されます。遷移 2 は、着信イベントが必要な順序内で最初の述部と一致したときに発生します。同様に、遷移 3 は着信イベントがその順序内で 2 番目の述部と一致したときに発生します。状態 S4 に達すると、ルールはトリガー・イベントを転送し、初期状態 S1 にリセットされます (遷移 5)。遷移 4 は時間間隔が満了した場合に発生し、ルールが初期状態にリセットされますが、その際トリガー・イベントは転送されません。

398 ページの図 134 に、パススルー・ルール (必要なイベントはどのような順序で到着しても構わないことを指定した (randomOrder=yes) 場合) での状態遷移を示します。

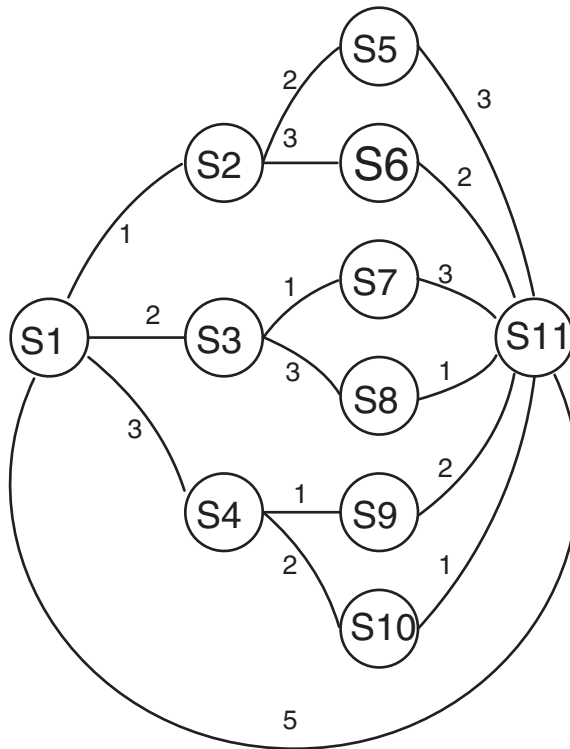


図 134. パススルー・ルール (*randomOrder=yes*) の場合の状態遷移

図 134 では、397 ページの図 133 の場合と同じ遷移が発生します。ただし、このケースでは、最終状態は S5 であり、この状態に達した後に状態マシンがリセットされます。

passthrough 相関機能は、異なるイベントを結合します。以下の例では、2 つの異なるメッセージ (MSG001 および MSG002) が 30 秒の時間フレーム内に受信されると自動化アクションが発生します。

```

<rule id="SAMPLE001">
  <eventType>DUMMYMSG</eventType>
  <passthrough randomOrder="true" 1 timeInterval="30000" 2
                                     triggerMode="allEvents" 3>

    <predicate>
      <![CDATA[&MSGID == "MSG001" ]]>
    </predicate>
    <predicate>
      <![CDATA[&MSGID == "MSG002" ]]>
    </predicate>
  </passthrough>
  <action function="ReturnToNV"/> 4
</rule>

```

以下では、上記の XML の例のステートメントについて説明します。

- 1** ルールは、eventType が DUMMYMSG でメッセージ ID MSG001 および MSG002 を持つものを探します。これらのメッセージのいずれかが受信されると、相関エンジンは 30 秒の制限時間の経過を記録し始めます。randomOrder が false に設定されている場合は、先に MSG002、その後に MSG001 というイベント・シーケンスであったときにはルールは発動されません。

- 2** ミリ秒単位で 30000 ということは、30 秒ということです。
- 3** ルールが起動されると、すべてのイベントが NetView プログラムに戻されます。

triggerMode を使用して、最初のイベントのみ、または最後のイベントのみを NetView プログラムに戻すように指定することもできます。triggerMode を指定する場合は、CORRELATED='1' 条件を検査する自動化テーブル項目も選択してください。

- 4** この例での相関タイプは passthrough ですが、これが ResetOnMatch であったとすると、ReturnToNV アクションは MSG001 または MSG002 のどちらか一方が受信されたときに起動されます (両方とも受信されたときではありません)。

リセット・オン・マッチ・ルール: リセット・オン・マッチ・ルールは、指定された時間間隔の間に一連の特定イベントが到着しなかった場合にのみ、トリガー・イベントを転送します。タイマーが満了する前に必要なイベントが到着すると (特定の到着順序の指定はオプション)、トリガー・イベントは転送されません。到着しなかった場合は、タイマーがリセットされ、トリガー・イベントが転送されます。

リセット・オン・マッチ・ルールの場合に必要な引数は、次のとおりです。

- 必要なイベントの到着順序はどのような順序でも構わないか、指定した順序でなければならないかを指示するブール値 (randomOrder)。randomOrder が yes に等しい場合は、イベントはどのような順序で到着しても構いません。
- その満了後に状態マシンがリセットされる時間間隔。
- トリガー・イベントを定義するトリガー述部。これが状態マシンを初期化するイベントであり、必要な後続イベントが時間間隔内に到着すると転送されます。
- トリガー・イベントが転送されないようにするのに必要な後続イベントを指定する 1 つ以上の述部。

図 135 に、リセット・オン・マッチ・ルール (必要なイベントは順序どおりに到着しなければならないことを指定した (randomOrder=no) 場合) での状態遷移を示します。

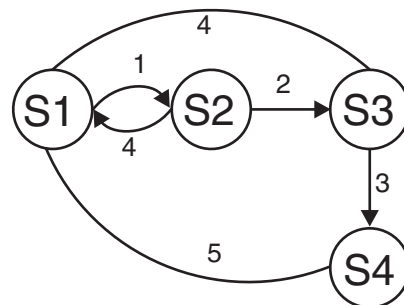


図 135. リセット・オン・マッチ・ルール (randomOrder=no) の場合の状態遷移

図 135 では、S1 (状態 1) が初期状態です。トリガー・イベントが検出されると、遷移 1 が発生します。この遷移により、イベントが保管され、タイマーが開始されます。遷移 2 は、着信イベントが必要な順序内で最初の述部と一致したときに発生します。同様に、遷移 3 は着信イベントがその順序内で 2 番目の述部と一致した

ときに発生します。状態 S4 に達すると、ルールは初期状態 S1 にリセットされます (遷移 5)。時間間隔が満了したときには遷移 4 が発生します。この場合は、ルールはトリガー・イベントを転送した後、初期状態にリセットされます。

図 136 に、リセット・オン・マッチ・ルール (必要なイベントはどのような順序で到着しても構わないことを指定した (randomOrder=yes) 場合) での状態遷移を示します。

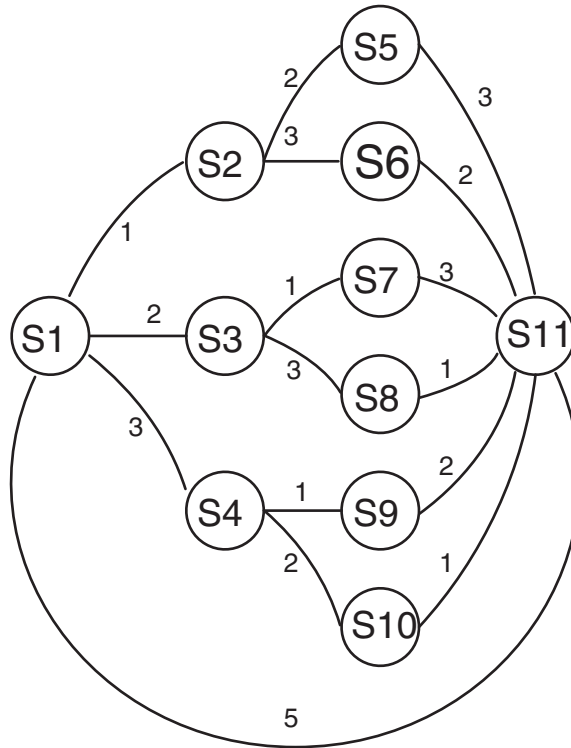


図 136. リセット・オン・マッチ・ルール (randomOrder=yes) の場合の状態遷移

図 136 では、399 ページの図 135 の場合と同じ遷移が発生します。ただし、このケースでは、最終状態は S5 であり、この状態に達した後に状態マシンがリセットされますが、その際にトリガー・イベントは転送されません。

状態マシンの複製

状態ベースのルールは、どれでも **clonable** タグを使用して複製できます。トリガー・イベントの発生時に状態相関がルールを複製する場合、状態相関はそのルールの別のインスタンスを作成します。このルールを使用すれば、多数のルールを作成しなくても複数のイベント・シーケンスを処理できるので便利です。

cloneable タグを使用すると、相関エンジンは指定されたイベント属性別にイベントを分類し、それらのイベントのために別々の相関プロセスを保守します。例えば、NetView タスク初期化メッセージ DSI166I の発行後 10 秒以内に初期化完了メッセージ DSI530I が発行されることを検証するには、メッセージ DSI530I が受信されなかった場合に発動される ResetOnMatch ルールをコーディングします。このルールは、次のようにコーディングできます。

```

<rule id="testcases.taskinitfailure">
  <eventType>INITMSGSGS</eventType>
  <resetOnMatch randomOrder="false" timeInterval="10000" triggerMode="firstEvent">
    <cloneable attributeSet="TASKN"/> 1
    <predicate>
      <![CDATA[&MSGID == "DSI166I" ]]> 2
    </predicate>
    <predicate>
      <![CDATA[&MSGID == "DSI530I" ]]>
    </predicate>
  </resetOnMatch>
  <action function="ReturnToNV"/> 3
</rule>

```

以下では、上記の XML の例のステートメントについて説明します。

- 1** このルールは、eventType INITMSGSGS のすべてのイベントはタスク名を含んでいる TASKN という名前の属性を持っていることを前提としています。
- 2** DSI166I を受け取るたびに、毎回別の関連プロセスが 10 秒タイマーを開始して、同じ TASKN 値と MSGID 値 DSI530I を持つ INITMSGSGS イベントが受信されるのを待ちます。MSGID が DSI530I でも TASKN 値が異なるイベントでは、述部の条件は満たされません。
- 3** 10 秒以内にメッセージが両方ともは受信されなかった場合は、メッセージ DSI166I から取得された最初のイベントが NetView プログラムに戻されます。

自動化テーブルからイベントを作成するには、次のようにコーディングします。

```

IF MSGID='DSI166I' & TEXT= 'DSI166I' TASKNAME 'IS ACTIVATED' . THEN
  BEGIN;
    IF CORRELATED='0' THEN
      EXEC(CMD('CNMCRMSG INITMSGSGS TASKN ' TASKNAME))
      DISPLAY(Y) SYSLOG(Y) NETLOG(Y) CONTINUE(N);
    IF CORRELATED='1' THEN
      EXEC(CMD('PIPE LIT /SAMP003 ' TASKNAME 'FAILED TO COMPLETE ITS
INITIALIZATION WITHIN SPECIFIED TIME PERIOD /|COLOR RED|CONS'))
      DISPLAY(N) SYSLOG(N) NETLOG(N) CONTINUE(N);
    END;

IF MSGID='DSI530I' & TEXT=, '' TASKNAME '' :'. THEN
  BEGIN;
    IF CORRELATED='0' THEN
      EXEC(CMD('CNMCRMSG INITMSGSGS TASKN ' TASKNAME))
      DISPLAY(Y) SYSLOG(Y) NETLOG(Y) CONTINUE(N);
    END;

```

カスタム・アクションの作成

ルールでは、標準のアクションのほかに、Java コードを使用して作成したカスタム・アクションも使用できます。カスタム・アクションを作成すれば、イベント属性の変更も含め、より洗練されたイベント処理を行えます。

アクションは、それぞれが 1 つの Java クラスとして実装されます。状態関連エンジンが開始されると、このエンジンは、ルールによって指定されたあらゆるパラメーターを使用して、ルールが必要とするすべてのアクション・クラスのインスタンスを作成します。アクションが共用アクションとして宣言されている場合は、各アクションのインスタンスは 1 つしか作成されず、この同じインスタンスが、そのアクションを呼び出すすべてのルールによって使用されます。アクションが共用アクションでない場合は、特定のアクション・クラスを使用するそれぞれのルールごと

に別々のインスタンスが作成されます。(ある 1 つのアクションを、複数の異なるルールからそれぞれ異なるパラメーターを使用して呼び出す必要がある場合、そのアクションは共用アクションにはできません。)

イベント・オブジェクト: 状態関連エンジンは、イベントを表す Java オブジェクトを処理します。イベントが到着すると、そのイベントから取得されたすべての属性データを含む Java オブジェクトが作成されます。このオブジェクト (クラス `com.tivoli.zce.engine.Event` のインスタンス) は、状態関連エンジンに送られます。永続性が使用可能にされている場合は、状態関連エンジンはそのイベントのレコードを永続ストアに書き込みます。永続ストアは、状態関連エンジンによるイベントの処理中にゲートウェイがシャットダウンしてもイベントが 1 つも失われないようにするために使用されるリカバリー・メカニズムです。ゲートウェイが再始動すると、永続ストアに記録されたすべての未送信イベントが即時にゲートウェイに送られます。

最初に作成されるときに、Java イベント・オブジェクトには次の 2 種類のイベントのコピーが含まれます。

- 作業用コピー。このコピーには、イベント・オブジェクトのメソッドを使用して直接アクセスできます。アクションは、この作業用コピーを使用して、処理の実行中にイベント属性に変更を加えることができます。
- イベントの状態の内部スナップショット (受け取ったのが状態関連エンジンであるため)。このフィールドの内容は、最初は永続ストアに書き込まれたレコードと同じです。また、作業用コピーに変更が加えられても、このフィールドに対しては、その変更と一致させるための動的更新は行われません。

その後、イベント・オブジェクトは、状態関連ルールおよびそれらのルールから呼び出されるあらゆるアクションによって処理されます。この処理には、イベント属性値の変更や新規イベントの作成が含まれる場合があります。

アクション構造体: アクションは、Java クラスとして実装されます。状態関連エンジンに検出されるためには、`com.tivoli.zce.actions.libs` パッケージに入っている必要があります。このクラスは、状態関連ルール内でのイベント処理をサポートするのに必要なインターフェースを実装していなければなりません。アクション・クラスは、必要なインターフェースとなんらかのデフォルトの振る舞いを提供する `com.tivoli.zce.actions.DefaultActionHandler` クラスを拡張する必要があります。NetView プログラム提供の `FLBCorAction` スーパークラスはこれを行います。`FLBCorAction` はユーザー作成アクションのスーパークラスとして使用できます。アクションは、以下の 3 つのメソッドを実装する必要があります。

- `processEvent()`。このメソッドは、ルールまたはルール内の先行するアクションから呼び出され、単一のイベント・オブジェクトをパラメーターとして取りまします。このメソッドは、アクションが単一イベントを処理している場合に使用されます。
- `processEvents()`。このメソッドは `processEvent()` メソッドに似ていますが、このメソッドの場合はパラメーターとして、複数イベントの配列を含んでいる単一の `EventList` オブジェクトを取りまします。この 2 つのメソッドは、両方とも実装する必要があります。なぜなら、アクションの呼び出しには、単一イベントによるものとイベント・リストによるものとの両方があるからです。 `processEvents()`

メソッドによってリストを構文解析してから、イベントごとに `processEvent()` メソッドを呼び出すといった場合が考えられます。

- `doParse()`。このメソッドは、初期化時に (アクション・クラスがインスタンス化された後で) 状態相関エンジンによって呼び出されます。 `doParse()` メソッドは、アクションを呼び出す XML ルール内に指定されているパラメーターを構文解析します。これらのパラメーターは、アクション・インスタンスの振る舞いがそのインスタンスを使用するすべてのルールに合うように、その振る舞いを制御します。(共用されないアクションの場合は、複数のインスタンスがそれぞれ異なるパラメーターを使用して作成される場合があることに注意してください。)

アクションがイベントを (`processEvent()` メソッドまたは `processEvents()` メソッドを介して) 受け取ったら、アクションはイベント・オブジェクトのメソッドを呼び出して、イベント属性データを検索または変更できます。(詳しくは、404 ページの『イベントの処理』を参照してください。) イベントが廃棄されない場合は、アクションは最後にそのイベントを処理の次のステップに送る必要があります。これについては、イベントが別のアクションに送られる場合と、状態相関エンジンに戻される場合 (現行アクションがルール内における最後のアクションであるとき) とがあります。これを行うためのメソッドは基本的には `forward()` メソッドですが、ある種の状況下では別のメソッドを使用する必要があることがあります。

要約として、図 137 にアクション・クラスの一般的な構造を示します。

```
package com.tivoli.zce.action.libs;

import com.tivoli.zce.IRule;
import com.tivoli.zce.ParserException;
import com.tivoli.zce.CorrelatorException;
import com.tivoli.zce.engine.EventList;
import com.tivoli.zce.engine.Event;
import com.tivoli.zce.actions.DefaultActionHandler;

public class MyAction extends DefaultActionHandler implements ITecEventAttributes
{

    public void processEvents(EventList eventList) throws Exception
    {
        // code to process multiple events
        forward(eventList);
    }

    public void processEvent(Event event) throws Exception
    {
        // code to process single event
        forward(event);
    }

    public Object doParse(IRule rule, String params) throws ParserException
    {
        // method to parse parameters after instantiation
    }
}
```

図 137. アクションの構造

イベントの処理: イベント・オブジェクト (`com.tivoli.zce.engine.Event` のインスタンス) には、イベント属性情報が、ハッシュ・テーブルに保管されている名前と値の対のセットとして含まれています。イベント・クラスは、これらの属性を処理するのに使用できるメソッドを提供します。メソッドには、次のものがあります。

hasAttribute()

`hasAttribute()` メソッドは、パラメーターとして単一のストリングを取り、指定された名前を持つ属性がイベントに含まれているかどうかを示すブール値を返します。例えば、`event.hasAttribute("HOSTNAME")` は、*event* に `HOSTNAME` という属性が含まれている場合に `true` を返します。(属性名では大文字と小文字が区別されることに注意してください。)

getString()

`getString()` メソッドは、パラメーターとして単一のストリングを取り、指定された名前を持つ属性の値が含まれているストリングを返します。例えば、`event.getString("SEVERITY")` は、*event* の `SEVERITY` 属性の現行値を返します。

putItem()

`putItem()` メソッドは、パラメーターとしてストリング・キーと値を取ります。このメソッドは、指定されたキー・ストリングと等しい名前を持つ属性の値を設定します。例えば、`event.putItem("ORIGIN","SCE")` は、`ORIGIN` 属性の値をストリング `"SCE"` に設定します。指定されたキーが既存の属性と一致しない場合は、新しい属性が追加されます。属性値にスペースまたは特殊文字が含まれているときには、その値をネストした単一引用符で囲んで、イベント・サーバーによる構文解析が正しく行われるようにしてください。

第 23 章 整合性のとれた自動操作の確立

コマンドをスケジュールしたりメッセージや MSU に応答したりする以上に複雑な操作でも、自動化できます。自動化できるオペレーター・タスクには、例えば次のようなものがあります。

- 使用しているシステムまたはネットワークに入っているプロダクトの初期設定
- プロダクトのモニター
- 必要に応じたりカバリー・アクションの開始
- プロダクトを非活動化したい場合の正常シャットダウン

拡張自動操作を実現するには、多くのコマンド・プロシージャーとその他の自動化機能との間でアクションを調整する必要があります。例えば自動化テーブルは、メッセージおよび MSU の形式で情報を受け取り、その情報をモニター用のコマンド・プロシージャーに渡しますが、このあとモニター用のプロシージャーは、必要に応じてリカバリー処理を開始できます。さらに、自動化を常に利用できるようにしておくため、自動化アプリケーションを互いにモニターさせるようにします。

拡張自動操作を実現するには自動化アプリケーション間での調整が必要なので、先に自動操作プロジェクトを詳細に設計しておく必要があります。自動化の設計ガイドラインについては、57 ページの『第 4 章 自動操作プロジェクトの設計』を参照してください。

整合性のとれた自動操作を行うには、NetView グローバル変数とリソース・オブジェクト・データ・マネージャー (RODM) の一方または両方を使用します。本章では、NetView グローバル変数を使って整合性のとれた自動操作を確立する方法について説明します。RODM を使用して整合性のとれた自動操作を確立する方法についての説明は、461 ページの『第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作』を参照してください。

NetView グローバル変数により整合性のとれた自動操作を確立する前に、NetView で提供されている拡張自動化サンプル・セットを調べてください。このサンプル・セットを使用すると、MVS のプロダクトおよびコンポーネントの初期設定、モニター、リカバリー、およびシャットダウンが自動化されます。このサンプル・セットでは、内部モニター機能を使用して、自動タスクを活動状態のまま機能を継続します。サンプル・セットを調べると、グローバル変数命名規則とその他の共通規約によって、いかにコマンド・プロシージャー間の通信が効果的に行われるかがわかります。

状態変数を使う方法

整合性のとれた自動操作を実現する方法の 1 つに、状態変数のシステム上に自動化を構築するというものがあります。システムまたはネットワークの操作は、各システムまたはネットワークのリソース状態をモニターするプロセスとして見ることができます。リソースによって状態が変更されるのは、問題が発生した場合、または問題解決のためにアクションを行う場合です。アプリケーション・プログラムのシ

シャットダウン、ネットワーク・リソースの活動化、オペレーターによるログオンのいずれもすべて、状態が変わったことを表します。

システムまたはネットワークをモニターするには、各エレメントの状態を示すメッセージおよび MSU を観察します。エレメントの目標状態を追跡し、目標状態と現行の状態が一致しない場合にはさらに、修正アクションを試行します。

自動化された環境では、自動化アプリケーションによって現行状態と目標状態を追跡することができます。例えば、2 つのグローバル変数を、自動化したい各コンポーネントまたはリソースに割り当て、一方の変数は現行状態の保持に使い他方の変数は目標状態の保持に使うことができます。状態の変更を示すメッセージまたは MSU が自動化テーブルに受信されると、それに応じて現行状態の変数を更新できます。目標状態とは、事前に確立した条件または方針をもとにしたものです。1 例として、VTAM プログラムを午前 6 時から真夜中までの間アクティブにしておくという方針があります。また、目標状態の変数を直接更新できるようにするオペレーター・インターフェースを用意することもできます。

情報の追跡もできます。例えば、各リソースに対して行われる自動化アクションを変数を使って示すことができます。こうして実行中のアクションを追跡していれば、ある問題に対する修正アクションが完了する前に、再度同じ修正アクションを行う事態を回避できます。拡張自動化サンプル・セットでは、実行されるアクションがアクション状態変数に記録されます。例えば、障害発生後にこのサンプル・セットによって TSO が再始動されるとき、TSO の状態が DOWN から STARTING に更新されます。

以下のような、特定のアプリケーションにログオンしているユーザーの数を保管する変数や、方針についての情報を保管する変数を使用することもできます。

- プロダクトの依存関係 (例えば、VTAM がアクティブのとき以外は TSO を始動させない)
- 特定のプロダクトを活動化する時点や再度シャットダウンする時点などのタイミング情報
- 自動化が責任を持つのはリソースを目標状態に保つことか単にリソースをモニターすることか (リソースの操作を手動で行いたい場合は、自動化をオフに切り替えられる)

407 ページの図 138 は、状態変数を使って、自動化を整合性のとれたものにするために考えられる構造を示したものです。

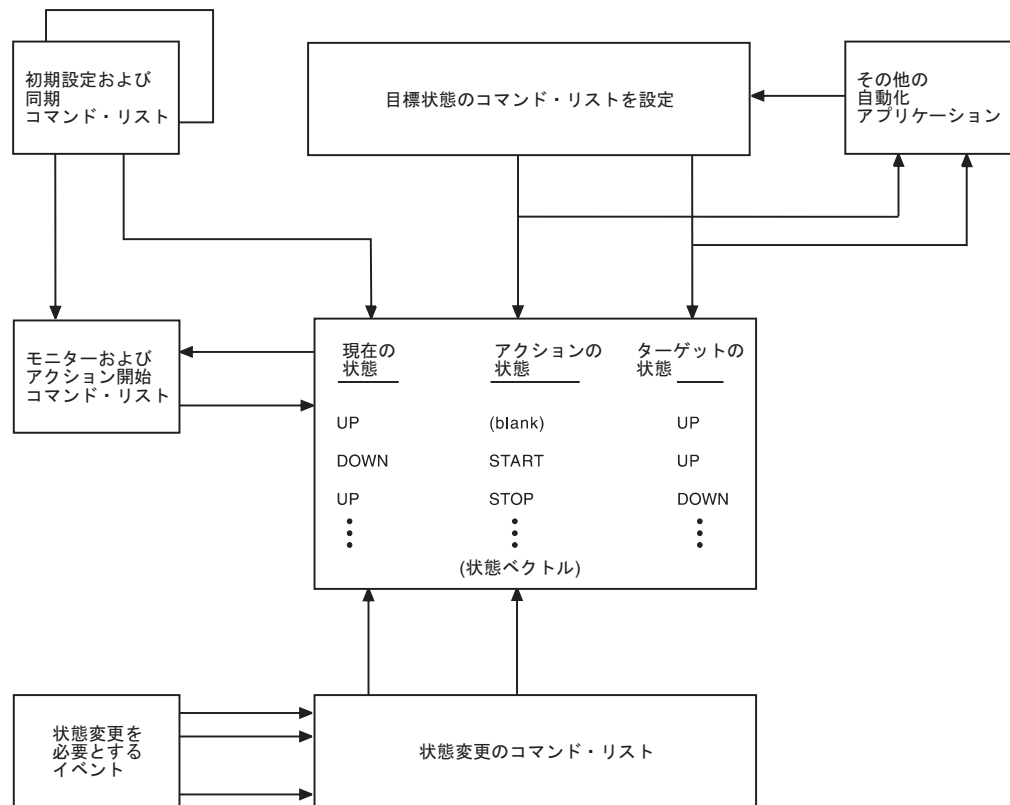


図 138. 状態変数を使って行われる、整合性のとれた自動操作

初期化、モニター、リカバリー、およびシャットダウンの自動化

整合性のとれた自動操作によって実行可能になる重要な操作タスクには、システムおよびネットワークのリソースとコンポーネントについての、初期化、モニター、リカバリー、およびシャットダウンがあります。それぞれ次のことを意味します。

初期設定

プロダクトまたはコンポーネントを開始または活動化すること

モニター

システムをモニターして、各プロダクトまたはコンポーネントの状態を追跡すること

リカバリー

モニターによって、問題が起こっていることや現行状態と目標状態の間に矛盾があることが発見された場合に、修正アクションを行うこと

シャットダウン

プロダクトまたはコンポーネントを通常の方法で停止すること

ここでは、初期化、モニター、シャットダウン、およびリカバリーの処理を自動化する方法を説明します。これらの 4 つのタスクを自動化する方法については、661 ページの『拡張自動操作サンプル・セットの使用』の拡張自動化サンプル・セットについての説明を参照してください。

初期化の自動化

プロダクトまたはコンポーネントの初期化を自動化するには、通常、オペレーターが行う手動のプロセスをモデルにします。オペレーターは、特定のコマンドを出して、初期化の正常終了を示す特定のメッセージを待ちます。一定時間内にメッセージが受信されない場合は、問題が発生したと考えられ、オペレーターはリカバリー・アクションをとります。

システム全体の初期化を自動化するには、まずオペレーティング・システムを手動で活動化します。また、オペレーティング・システムをリモートで活動化することもできます。詳細については、19 ページの『リモート操作の確立』を参照してください。オペレーティング・システムにより自動的に NetView プログラムを活動化でき、また NetView プログラムにより他のプロダクトおよびコンポーネントの初期化を自動化することができます。NetView プログラムを自動的に開始するには、NetView プログラムの開始コマンドを SYS1.PARMLIB の COMMNDxx メンバーに指定します。

NetView 初期コマンド・リストを使用して自動化プロシージャを呼び出し、残りのプロダクトおよびコンポーネントすべての初期化を開始できます。プロダクトの依存関係によっては、あるプロダクトを初期化する前に、別のプロダクトの初期化が正常終了するのを待たなければならないことがあります。例えば、TSO を初期化するには、その前に VTAM を活動化しておかなければなりません。

モニターの自動化

自動化には、以下に示すように、受動モニターと能動モニターの両方を使用できます。

受動モニター

特定のメッセージおよび MSU をモニターし、それらが受信されたときにアクションを起こすこと

事前の対策を講じたモニター

照会コマンドを発行して状況を判別すること

受動モニター

自動化環境では、システムとネットワークのコンポーネント状況を示すすべてのメッセージおよび MSU をオペレーターがモニターする必要はなく、自動化テーブルが受動モニターを行います。メッセージまたは MSU が受信されたときに取られる自動化アクションには、状態変数を更新して NetView プログラムが各コンポーネントの状態についての正確な記録を持つようにするアクションが含まれています。

能動モニター

能動モニターでは、システムまたはネットワークの照会を行って状況についての情報を得るコマンドを出すことが含まれます。照会コマンドは、更新情報を得られるように、一定の時間間隔で出せます。EVERY コマンドのような NetView タイマー・コマンドを使用すると、必要なときに照会コマンドをスケジュールできます。

この時間間隔が短いとより新しい状況情報を得ることができ、それによってより迅速に障害に対応できます。しかし、照会コマンドを頻繁に出しすぎると、システムに不必要な負荷を与えてしまうこともあります。

また、能動モニターを使ってユーザーの自動化アプリケーションの状況をモニターします。例えば、自動タスクをモニターするには、テスト・コマンドまたはテスト・メッセージをモニター対象の自動タスクに一定間隔で送信します。自動化テーブルを利用してテスト・メッセージに対して特定の応答を出すように自動タスクのセットアップが行われている場合、正しい応答の送信に障害が起これば、自動タスクの障害が示されます。拡張自動化のサンプル・セットでは、この手法を示しています。

自動化テーブルをモニターすることもできます。それには、自動タスクが定期的に `AUTOTBL STATUS` コマンドを出してその結果を待つようにします。こうすれば、正しい自動化テーブルが常に実行していることを確認できます。

能動モニターで集めた情報は、受動モニターで集めたものと同様、状態変数に保管できます。この情報を、自動化アプリケーション全体に使用可能にすることができます。

能動モニターと受動モニターの組み合わせ

受動モニターを使うと、自動化は次のスケジュール済み照会コマンドが問題を検出するまで待たないので、速度の点では通常は優れています。しかし、能動モニターを使うと、予定したメッセージを出さないで状態を変更するコンポーネントも正確にモニターできるので、信頼性の点では能動モニターの方が優れています。能動モニターと受動モニターを組み合わせれば、両方の優れている面を利用できます。

リカバリーの自動化

受動モニターまたは能動モニターにおいて、現行状態と目標状態の不一致などの問題が検出された場合、リカバリー処理を開始できます。自動リカバリー処理は、オペレーターがコンソールの異常終了メッセージを受信したあとにプロダクトの再始動を試みる処理と同様のものです。

障害の発生しているコンポーネントのリカバリー処理は、そのコンポーネントの開始処理と同じなので、同じコマンド・プロシーチャーを使用できます。しかし、自動化では、障害メッセージに迅速に応答したり、問題の原因を究明したり、障害のあるコンポーネントで再活動化の準備ができていることを確認したりする作業がコンポーネントの再始動前に必要な場合があります。

シャットダウンの自動化

自動初期化と同じように、自動シャットダウンは通常は手動処理と同様のプロセスで行います。特定のプロダクトをシャットダウンするには、そのプロダクトに対するシャットダウン・コマンドを発行し、正常終了を示すメッセージを待ちます。

システム全体をシャットダウンする場合は、一般的にはプロダクトを、初期化したときとは逆の手順でシャットダウンします。プロダクトをシャットダウンする前には、そのプロダクトに依存している他のすべてのプロダクトのシャットダウンが先に必ず完了しているようにしてください。

第 24 章 オペレーター・インターフェースの拡張

自動化計画を実施する際に重要なのは、環境に適したオペレーター・インターフェースを作成することです。優れたオペレーター・インターフェースが作成されていれば、オペレーターは、環境のモニター、各リソースの状態の調査、およびその自動化が正しく機能しているかどうかの検証に必要な情報を得られます。

この他、自動化ルーチンでまだ自動化されていない問題またはイベントが検出されたときにオペレーターに通知する処理である**例外通知**が必要です。例外通知の機能により、オペレーターの注意を、手動の介入がまだ必要な問題に集中させられます。

自動化環境では、以下の形式の情報がオペレーターに表示されます。

- ・ コマンド機能によって表示されるメッセージ
- ・ 状況モニターおよび NetView マネージメント・コンソール (NMC) による状況情報
- ・ ハードウェア・モニターまたは NMC によるアラート
- ・ VIEW (表示) およびヘルプ・パネルによるフルスクリーン表示
- ・ 電子メールまたは英数字ページ

メッセージの表示

メッセージ形式の情報は、NetView コマンド機能を使用してオペレーターに表示できます。NetView メッセージおよびネットワーク・メッセージの表示は、自動化されていない環境と同様に、コマンド機能で継続させることができます。さらに、コンソールが統合されている場合は、オペレーティング・システム、サブシステム、およびアプリケーションからシステム・メッセージを表示できます。

自動化を行うと、表示されるメッセージの数が削減されます。341 ページの『第 19 章 メッセージの抑止とアラートのフィルター操作』と 361 ページの『第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化』には、メッセージの送信量を減らす方法を説明しています。

しかし、自動化環境でもメッセージは役に立つので、自動化プロシージャを使用してメッセージをコマンド機能で出すこともできます。例えば自動化をテストする場合、自動化プロシージャでメッセージを出して、実行するアクションをオペレーターに通知するようになります。テストが完了して自動化がスムーズに流れれば、この種のメッセージの使用を減らせます。メッセージのもう 1 つの利用方法は、自動化プロシージャにおいて手動の介入が必要な問題が検出された場合にオペレーターに通知するというものです。

コマンド・リストでは、MSG コマンドでメッセージを出すことができます。MSG コマンドの説明については、NetView オンライン・ヘルプを参照してください。

状況情報の表示

NetView には、ネットワークについての状況情報を表示する次の 2 通りの方法があります。

- 状況モニター (テキスト形式)
- NetView マネージメント・コンソール (グラフィック表示)

上記の機能を使用して、ネットワーク・リソースの状態を追跡したり、これらの状態をオペレーターは編成された階層表現を通して見ることができます。また、いずれの機能も、ネットワークに問題があるときは、色を変えて注意を促します。

状況表示を使用すれば、自動化を補足できます。個々のメッセージ、アラート、および MSU が自動化で処理されている間、オペレーターは迅速にネットワーク状態を表示して、自動化によって各リソースが正しい状態に保たれていることを確認できます。

状況モニターによる状況の追跡

状況モニターは状況情報をテキスト形式で表示します。

状況モニターを使うと、情報を表示するだけでなく、アプリケーション・リソースおよびドメイン間リソース以外で障害のあるリソースを自動的に非活動化することもできます。状況モニターは、活動状態でないリソースを示す VTAM プログラムからの状況情報を代行受信し、リソースがアクティブに戻るまで 1 分間隔で再活動化の試行を開始します。この機能を使用可能にするには、DSICNM に O MONIT ステートメントを指定しておく必要があります。MONON コマンドと MONOFF コマンドを使用すると、特定のリソースまたはすべてのリソースに対して自動再活動化のオンとオフの切り替えを行えます。また、VTAMLST メンバーで STATOPT ステートメントを使用すれば、状況モニターが再活動化を試行するリソースを選択できます。

O MONIT および STATOPT ステートメントについては、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

NetView 管理コンソール・ディスプレイによる状況の追跡

NetView マネージメント・コンソールは、ネットワーク図を画面上に描いて、状況情報をグラフィック形式で表示します。この図を、オペレーターに必要な情報を表示するようにカスタマイズできます。複数のオペレーターがそれぞれ別のワークステーションからネットワークのコンポーネントをモニターすることもでき、また、1 人のオペレーターが企業全体をモニターすることもできます。

NetView のグラフィック表示機能は、MVS システムに接続されたワークステーションを使用して、オペレーターに情報を表示します。オペレーティング・システムについての状況情報も、まずその情報を MVS システムに転送すれば、グラフィカルにモニターできます。転送の方法については、423 ページの『第 26 章 集中操作』を参照してください。

グラフィカルな状況表示の詳細については、「*IBM Tivoli NetView for z/OS* NetView 管理コンソール ユーザーズ・ガイド」を参照してください。

ハードウェア・モニターによるアラートのモニター

ハードウェア・モニターは、システムとネットワークをモニターするのに使用できます。自動化環境では、ハードウェアとソフトウェアの両方にハードウェア・モニターを使用できます。ハードウェア・モニターは、例外通知処理により自動化だけでは処理できない問題が発生したときにオペレーターに対してアラートを表示します。

ハードウェア・モニターは、問題が発生したときにメッセージから得られる情報よりも多くの情報を表示できます。表示できる情報には、問題の記述、問題の推奨原因のリスト、および推奨処理のリストがあります。ハードウェア・モニターは、さらに次の情報を提供します。

- 報告された問題のヒストリー
- フィルター操作機能
- ソフトウェア情報/管理プログラムへの問題管理インターフェース
- システム管理機能 (SMF) またはその他の外部ログについての記録機能

そのため、オペレーターの介入が必要なメッセージをアラートに変換して、ハードウェア・モニターで表示することもできます。しかし、そのメッセージを抑止したり自動的に応答する場合は、メッセージをアラートに変換しないでください。自動化の目的は、オペレーターが見なければならないイベント通知の数を減らすことにあります。

以下の機能を使うと、アラートをハードウェア・モニターに送信できます。

- プログラム間インターフェース
- GENALERT コマンド
- MS トランスポート

アラートの発行が頻繁すぎてストレージを使いきってしまうことのないように、アラートを作成する自動化の優先順位が、DSICRTR、BNJDSERV、および LUC (あとに LUC の続くドメイン ID) のタスクより高い優先順位になっていないことを確認してください。

注: プログラム間インターフェースまたは MS トランスポートを介してハードウェア・モニターに送信されたアラートは、GENALERT によって作成されたアラートと同様に、NetView 自動化テーブル処理の対象となります。アラートを使用して自動化を開始し、自動化で別のアラートが作成される可能性がある場合、無限ループにならないよう注意してください。経路指定の詳細については、108 ページの『NetView プログラムのハードウェア・モニター・データ、および MSU の経路指定』を参照してください。

プログラム間インターフェースによるアラートの送信

同じシステムで実行している別のアプリケーション・プログラムから、NetView プログラムにアラートを送信するには、プログラム間インターフェースを使用します。プログラム間インターフェースは、NetView 内からアラートを生成する場合にも使用できます。

プログラム間インターフェースを使用すると、アプリケーション・プログラムは汎用アラートを NMVT または CP-MSU の形式で互いに送信し合ったり、ハードウェア・モニターに送信することができます。アプリケーション・プログラムが問題を

検出して、NetView プログラムにアラートを送信できます。アラートは、NetView サブシステムの CNMCNETV モジュールを呼び出し、アラート情報を NetView プログラムに渡すことによって作成します。NetView プログラムはアラートを非送信請求として処理します。アラートが適切なハードウェア・モニター・フィルターを通ると、このアラートはハードウェア・モニター・アラートとなり、オペレーターに表示できるようになります。

また、REXX コマンドでアラートを作成し、PPI PIPE ステージを使用してそのアラートをハードウェア・モニターに送信することもできます。

プログラム間インターフェースを使用してソフトウェア・アラートを作成する方法については、「*IBM Tivoli NetView for z/OS アプリケーション・プログラマーズ・ガイド*」を参照してください。

GENALERT コマンドによるアラートの送信

NetView プログラム内からアラートを作成したい場合は、GENALERT コマンドを使用します。例えば自動化テーブルで、オペレーターの介入が必要なメッセージが受信された場合に GENALERT コマンドを出すことができます。またコマンド・プロシーチャーで、オペレーターの介入が必要な問題が検出された場合にも GENALERT コマンドを出すことができます。

GENALERT コマンドの詳細については、NetView オンライン・ヘルプを参照してください。GENALERT コマンドで使用できるコード・ポイントおよびコード・ポイント・フォーマットの詳細については、「*IBM Tivoli NetView for z/OS Messages and Codes Volume 2 (DUI-IHS)*」の汎用アラート・コード・ポイントに関する付録を参照してください。

MS トランスポートによるアラートの送信

以下のものからハードウェア・モニターにアラートを送信するには、MS トランスポートを使用します。

- NetView プログラム内
- 使用しているシステム上の別のアプリケーション
- 別のシステム

この場合、MDS-MU にある NetView プログラムの ALERT-NETOP MS アプリケーションにアラートを送信します。個々の MDS-MU には、1 つ以上のアラート主ベクトルを持つ CP-MSU が、必ず 1 つ含まれている必要があります。

NetView 管理コンソールによるアラートのモニター

NetView 管理コンソール使用するオペレーターは、AON によって生成されたアラートを表示するために、アラート・ヒストリーを要求することができます。AON で Automation in Progress (自動化進行中) 状況を設定するので、オペレーターは自動操作により障害リソースの回復を試行していることが確認できます。回復できない障害リソースは、オペレーター介入ビュー (Operator Intervention view (OIV)) に表示されます。

詳細については、「*IBM Tivoli NetView for z/OS NetView 管理コンソール ユーザーズ・ガイド*」を参照してください。

フルスクリーン・パネルの作成

NetView プログラムを使用すると、さまざまなカラーや強調表示オプションを使って好みのフルスクリーン・パネルを作成できます。他のオペレーター・インターフェースを補足したり置き換えたりして、ネットワーク・リソースの状態表示および例外通知を行うためにフルスクリーン・パネルを作成できます。

フルスクリーン・パネルの作成には、ISPF などの標準的なエディターを使用できます。作成したパネルは、コマンド・プロシーチャーから NetView の VIEW コマンドを出して表示できます。

パネルの更新は動的に行えるので、オペレーターは変更情報をモニターできます。オペレーター入力を受け付けるパネル位置を指定することができ、ここから入力された情報は呼び出し元コマンド・プロシーチャーに送り返されます。自動化コマンド・プロシーチャーは、フルスクリーン・インターフェースを通してオペレーターと対話できます。パネルには NetView グローバル変数の値を表示でき、オペレーターはその変数値を変更できるようになります。また、オペレーターが ENTER や PF キーなどの特殊なキーを押した場合も、コマンド・プロシーチャーに通知されます。パネルのチェーンを指定しておくと、オペレーターが特定のキーを押したときに、あるパネルから別のパネルに移動するようにすることもできます。

拡張自動化サンプル・セットは、VIEW コマンドを自動化に使用方法を示しています。サンプル・セットで使用しているフルスクリーン・パネルは、NetView プログラムが自動化している各プログラムまたはコンポーネントの現在の状況を表示するようになっています。それぞれの状況はグローバル変数に保管され、適切な色でオペレーターに表示されます。例えば、CICS 用のパネル行は CICS に障害が起こったときには明るい赤に変わるようになっています。状況が変わると、その状況パネルを見ているすべてのオペレーターの画面を自動的に更新し、最新の状況と、状況が変わった時刻を示します。

オペレーターは、補足パネルにアクセスすれば、自動化を行っている特定のプログラムまたはコンポーネントについての詳細を知ることができます。このサンプル・セットのオペレーター・インターフェースの詳細については、678 ページの『自動操作表示パネル』を参照してください。VIEW コマンドの使用例については、NetView オンライン・ヘルプの NetView コマンド・リストの BROWSE、TUTOR、および DISG にも記載があります。

HELP コマンドも VIEW コマンドを使用するため、独自のヘルプ・パネルの作成や、既存のヘルプ・パネルの変更が可能になります。NetView プログラムでは、オンライン・ヘルプ・パネルおよびオンライン・メッセージ・ヘルプの多様なセットがネットワーク管理用に用意されています。使用している自動化環境用にこれらのパネルを修正すると、問題を解決して標準化されたプロシーチャーを実行するために必要なヘルプをオペレーターに提供できます。また、作成した自動化コマンド・リスト、コマンド・プロセッサー、およびオペレーター・インターフェースを使用するときにオペレーターを支援する新規のヘルプ・パネルを導入することもできます。

ヘルプ・パネルを含むフルスクリーン・パネルの使用に関する詳細については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。

電子メールまたは英数字ページの送信

問題が発生した場合に、連絡すべき要員、時間、方法に関して、ユーザーは INFORM ポリシー・メンバーを使って定義しておくことができます。デフォルトには、INFORM ポリシーがメールと英数字ポケットベル用のサポートを提供します。INFORM ポリシーの詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

第 6 部 複数システムの自動操作

第 25 章 他の NetView システムへの自動操作機能の拡張

全社的なデータ処理を自動化する最初のステップは、まず各 NetView システムで可能な限り多くの、ローカルな自動化を確実に行っていくことです。その結果、1 つのシステムまたは複数のテスト・システムにおいて単一システムの自動化を始めたら、その自動化を NetView システム全体に拡張してください。自動化ルーチンをコピーした後で、新システムに適応するよう調整します。このプロセスでは、できるだけ簡単に保守できるようにシステムすべての自動化を一貫させることが大切です。

システム全体に拡張するには、例外転送とフォーカル・ポイントの使用も準備する必要があります。システムを接続して例外情報の転送を行う場合、1 つのシステムの自動化が他のシステムの自動化に影響することがあります。そのため、自動化を同期化させること、および各システムとのフォーカル・ポイントとの関係を決定することが重要になります。本章では、自動化を効果的に拡張するためのガイドラインを記述します。

ソースに近い自動操作

多重システム環境では、分散システムでできることは分散システム上で自動化し、分散システムで処理できないことだけをフォーカル・ポイントに転送するようにしてください。分散システムでは、自動化テーブルを使用しなくてもオペレーティング・システムの機能 (メッセージ処理機能 - MPF) により行える場合は、該当するオペレーティング・システムの機能を使用してください。そうでない場合に、NetView プログラムを使用してください。

自動化プロシージャの区別

自動化プロシージャは、フォーカル・ポイント制御プロシージャと単一システム自動化プロシージャに分類できます。フォーカル・ポイント制御プロシージャとは、フォーカル・ポイント・システムで実行されるか、フォーカル・ポイントの代わりに分散システムで実行される自動化プロシージャのことで、更新された情報を定期的にフォーカル・ポイントに送る自動化プロシージャなどがあります。単一システム自動化プロシージャとは、システム内自動化プロシージャで、フォーカル・ポイントとして指定されたシステムとの通信を必要としない個々のシステムで使用されるものです。

役割の定義

作業の重複を避けるため、フォーカル・ポイントの役割と分散システムの役割の境界を明確にしてください。例えば、個々の分散システムが、AUTOTBL STATUS を出して自動化テーブルの状況を定期的に検査する自動タスクを持っている場合、分散システムの自動化テーブルをフォーカル・ポイントでモニターする必要はありません。

自動タスク定義の整合性

自動タスク、プロファイル、およびオペレーターのパスワードまたはパスワード・フレーズについては、システム全体にわたって整合性のあるオペレーター定義を使用してください。定義に整合性があると、システム間で変更を行うときに必要な作業を減らすことができます。詳細については、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

単一の MVS システムに複数の NetView プログラムがある場合、またはシスプレックス構成を使用している場合、オペレーター（および自動タスク）のために取得する拡張複数コンソール・サポート (EMCS) の名前は固有のコンソール名でなければなりません。競合を解決するには、オペレーターごとに初期設定コマンド・リストで NetView の GETCONID または SETCONID コマンドを使用してください。

GETCONID または SETCONID コマンドの説明については、NetView オンライン・ヘルプを参照してください。

汎用自動化コマンド・プロシーチャーの開発

複数システムで使う自動化プロシーチャーの開発、保守、および同期化に必要な作業を極力少なくするには、すべての該当システムに対して同等の機能を持つ汎用プロシーチャーを作成します。これらのプロシーチャーは、フォーカル・ポイント機能を実行していないときのフォーカル・ポイント・システムでも機能するようにします。

マイグレーション作業を簡単にするには、システムおよびリソースの名前をコマンド・プロシーチャーに固定的にコーディングせず、それらの名前にグローバル変数を使用します。グローバル変数の定義セットアップが必要なプロシーチャーをできるだけ少数にしておけば、同じ自動化プロシーチャー・セットを複数システムにマイグレーションしたときに、各システムでカスタマイズするプロシーチャーを少数に抑えられます。この方法は、拡張自動化サンプル・セットで使用しています (661 ページの『拡張自動操作サンプル・セットの使用』を参照)。

可搬自動化テーブルの開発

保守作業を軽減するためには、1 つの自動化テーブルを複数のシステムで共通して使用するのが最善ですが、さまざまな要求に合わせて調整しなければならないので、自動化テーブルが大きくなる可能性があります。また、環境が異なると、特定のメッセージが異なる用途に使用される場合もあります。このような場合は、システムごとに個別の自動化テーブルが必要になります。複数の自動化テーブルを使用している場合、システム内の更新が確実に整合性を保つようにしてください。

%INCLUDE ステートメントを使用して、自動化テーブルの一部のセクションを、別のデータ・セット・メンバーに保管しておくことができます。例えば、自動化テーブルの中で、すべてのシステムに共通する部分を 1 つのデータ・セット・メンバーに、各システムに特有の部分を 2 番目のデータ・セット・メンバーに保管することができます。

複数の自動化テーブル・メンバーを使用することもでき、システム単位で、適切なメンバーのみを使用可能にすることができます。これによる別の利点は、1 つのシ

システムから別のシステムへのワークロードの移動を反映するにあたって、自動化論理の一部を使用可能にしたり使用不可にすることが可能になることです。

SYN ステートメントを使用すると、自動化テーブルの中のシステムによって異なっている部分に同義語を定義することができるようになり、保守作業も容易になります。定義した同義語の値は、テーブルを新システムに対応させる場合に変更します。

転送の指定

メッセージ、アラート、およびコマンドの経路指定は、複数のフォーカル・ポイントと分散システムにまたがる自動化の委任を管理するために重要なものです。経路指定は、メッセージおよびアラートの処理を行う場所やコマンドを実行する場所を直接指定します。自動タスク ID やコマンド・プロシーチャーと同様、経路指定を行うときにも整合性が保たれると、自動化の保守作業が軽減されます。詳しくは、94 ページの『NetView プログラム・メッセージ経路指定』および 423 ページの『第 26 章 集中操作』を参照してください。

機能分散前のプロシーチャーのインストールおよびテスト

汎用のシステム間自動化プロシーチャーを開発している場合、すべてのシステムに機能を分散させる前に特定システム上にそのプロシーチャーをインストールしてテストすると、独立した環境で汎用的な問題を解決することができます。そのプロシーチャーが特定の環境で動作することが確認されれば、それを分散させ、グローバル変数または制御ファイルを各システムに対応するようカスタマイズし、企業全体にわたるテストを行います。最初に 1 つの独立したシステムでテストしておくことで、企業全体にわたるテストのときに必要になる修正の数を減らせます。

システム間自動化のログ記録

個々のシステムで発生するシステム間の自動化は、ローカル・ネットワーク・ログに必ず記録するようにしてください。以下の 2 つの理由により、フォーカル・ポイントに転送されるメッセージおよびアラートは、分散システムとフォーカル・ポイントの両方でログに記録するようにします。

- アラートまたはメッセージがフォーカル・ポイントに転送される時、関連する情報がすべて転送されるわけではなく、フォーカル・ポイントのオペレーターは、詳細を知るために分散システムを調べる必要があるかもしれないため。
- メッセージまたはアラートの伝送中に回線の障害が発生した場合には情報が失われるので、フォーカル・ポイントのオペレーターは、分散システムのログをブラウズして情報を収集する必要があるため。

自動化によって実行されるすべてのプロシーチャーは、ログの中でなんらかの方法で特定できるようにしておいてください。自動化サンプル・セットのコマンド・リストでは、メッセージをログに書き込むときに不等号 (<) が先頭に付けられます。ログを一目見るだけで、自動化が NetView プログラム内で発生している活動に積極的な役割を果たしているかどうかわかります。これらのインディケーターが自動化の監査証跡となり、自動化計画の一部としての定量的な目標値に対する達成度を

図る基準となります（47 ページの『第 3 章 自動操作プロジェクトの定義』を参照）。また、自動化の障害イベントにおける問題判別にも役立ちます。

ログ記録の詳細については、549 ページの『第 35 章 ロギング』を参照してください。

第 26 章 集中操作

自動化により、システム、ネットワーク、およびデータ・センターをすべて集中操作にし、1 つのシステムまたは少数の集中システムから管理できるようになります。多くのシステムを不在操作で実行し、操作要員を一箇所にまとめることができる場合もあります。そうすれば、単一システムのコンソールを統合する場合と同じ効果が期待でき、さらに、モニターするコンソールの数を減らすこともできます。

集中操作を実現するには、その前に個々のシステムでローカルに自動化を行い、できるだけ多くの操作を実行するようにしてください。各システムで処理を自動化する方法については、331 ページの『第 5 部 単一システムの自動操作』を参照してください。ローカルで解決できる問題の問題通知は転送しないでください。ただし、以下の情報は、中央システムに転送する必要があります。

- 各ローカル・システムの状態 に関する情報。ネットワークの中のそのシステムに関係する部分も含まれます。これを転送することにより、中央システムのオペレーターおよび自動化は、すべてのリソースとアプリケーションについて正確で最新の情報が得られるようになります。
- 例外 についての通知、もしくはローカルの自動化だけでは解決できない問題。これらの問題は、中央システムの自動化で、あるいは中央システムにログオンしているオペレーターによって解決できる可能性があります。

分散システムから情報を受信する中央システムのことを、フォーカル・ポイント と呼びます。適切なフォーカル・ポイントを選択するための設計指針については、57 ページの『第 4 章 自動操作プロジェクトの設計』を参照してください。

この章では、分散 NetView システムとフォーカル・ポイントの NetView システムの間で情報を伝送する方法について説明します。

データのトランスポート

NetView は、いくつかの異なるデータ・トランスポート方式を提供して、操作の集中化を助けます。トランスポート方式には、LU 6.2、LUC、および OST-NNT があります。このようなトランスポートを使用して、別のノードに常駐する NetView プログラム間でデータを転送します。LU 6.2 を使用すると、NetView と非 NetView プロダクト (例えば AS/400®) の間でデータを転送することもできます。NetView ノード間で操作を集中化するときには、上記の 1 つ以上のトランスポートを使用して、ノード間でデータを移動します。

LU 6.2 トランスポート

NetView プログラムは次の 2 つの LU 6.2 トランスポートをサポートしています。これらのトランスポートでは、異なるバージョンの SNA LU 6.2 プロトコルを使用します。

- 管理サービス (MS) トランスポートは、アラートの送信など、高度の信頼性を必要とする少量の伝送に使用します。

- MS トランスポートの高パフォーマンス・オプションは、高パフォーマンスを必要とする大量の伝送に使用します。

NetView LU 6.2 トランスポートは、「*SNA Management Services Reference*」に記述されている `MULTIPLE_DOMAIN_SUPPORT` 機能セットに基づいています。LU 6.2 トランスポートは、データの送受信のために管理サービス (MS) アプリケーションによって使用されます。

MS アプリケーションは、NetView 提供のアプリケーションやユーザー定義のアプリケーションのように、体系的なアプリケーションである場合があります。アプリケーションは LU 6.2 トランスポートを指定して登録すると、作成されます。登録したあと、アプリケーションはデータを他の登録済みアプリケーションに送信し、それらのアプリケーションからデータを受信します。例えば、NetView ALERT-NETOP アプリケーションにアラートを送信することができるユーザー定義のアプリケーションを作成することができます。NetView プログラムに知らされているアプリケーション (NetView プログラム提供およびユーザー定義の両方) を表示するには、図 139 に示すように、`REGISTER QUERY` コマンドを使用します。

```
* CNM02      REGISTER QUERY
' CNM02
DWO468I TYPE APPL      COMMAND TASK      FPCAT      FOCALPT LOGMODE NOTIFY
DWO469I MS   HMON_DST  BNJ62DST BNJDSEV --NONE--   NO      SNASVCMG NONE
DWO469I MS   HMON_OST  BNJ62OST BNJDSEV --NONE--   NO      SNASVCMG NONE
DWO469I MS   LINKSERV  BNJNETOP BNJDSEV --NONE--   YES     SNASVCMG NONE
DWO469I MS   ALERT     BNJNETOP BNJDSEV ALERT     YES     SNASVCMG NONE
DWO469I MS   OPS_MGMT  DSIOURCP DSI6DST --NONE--   YES     SNASVCMG NONE
DWO469I MS   EP_OPS    DSIOURCP DSI6DST OPS_MGMT NO      SNASVCMG ALL
DWO469I MS   SPCS      DSIYPIF  DSI6DST --NONE--   YES     SNASVCMG NONE
DWO469I MS   MS_CAPS   DSIIFRCV DSI6DST --NONE--   NO      SNASVCMG ERROR
DWO469I HP   RMTCMD_S  DSIUDST  DSIUDST N/A      N/A     PARALLEL NONE
DWO469I HP   RMTCMD_R  DSIUDST  DSIUDST N/A      N/A     PARALLEL NONE
DSI633I REGISTER COMMAND SUCCESSFULLY COMPLETED
```

図 139. `REGISTER QUERY` コマンドからの出力例

図 139 に表示されているアプリケーションの多くは、あとのセクションで説明します。このようなアプリケーションを LU 6.2 トランスポートの上部に位置していると考えることができます。DSI6DST タスクは、LU 6.2 トランスポートを使用するためにアクティブにしなければなりません。

DSI6SCF の制御範囲メンバーの名前と DSI6INIT LU 6.2 トランスポートの初期設定メンバーが、ノード構成に一致することを確認してください。例えば、フォーカル・ポイントの `VTAMCP USE` オプションが `yes` (`VTAMCP USE=YES`) に設定されている場合、フォーカル・ポイントは `DSIDMN` で参照され、エントリー・ポイントの `DSI6INIT` メンバーを使用して `VTAM CPName` で参照する必要があります。フォーカル・ポイントの `USE` オプションが `no` (`USE=NO`) に設定されている場合、フォーカル・ポイントはフォーカル・ポイントを実行している NetView プログラムのドメイン名で参照する必要があります。

VTAMCP ステートメントは、その NetView プログラム上で実行している NetView MS トランスポートが `VTAM CPName` を宛先に指定している `MDS-MU` を受信で

きるかどうか指定します。そのため、DSI6SCF ステートメントと DSI6INIT ステートメントをコーディングするときには、DSIDMN メンバーの VTAMCP ステートメントに注意してください。

MS トランスポートに使用される SNA プロトコルは、NetView に限定されるものではありません。MS トランスポートを使用して、これらのプロトコルをサポートする任意のシステム、もしくは任意の装置 (例えば AS/400) の MS アプリケーションと通信することができます。

LU 6.2 トランスポートを使用するには、まず NetView を LU6.2 アプリケーションとして VTAM に定義します。NetView ドメイン ID が VTAM LU 6.2 アプリケーション名として対応します。

図 140 は、NetView を VTAM LU 6.2 アプリケーションとして定義する方法を示したものです。

```
CNM01    APPL  AUTH=(NVSPACE,ACQ,PASS),PRTCT=CNM01,EAS=6,                X
          MODETAB=AMODETAB,DLOGMOD=DSILGMOD,APPC=YES
```

図 140. VTAM APPL ステートメント

LU 6.2 トランスポートを使用するアプリケーションの作成に関する詳細については、「*IBM Tivoli NetView for z/OS*アプリケーション・プログラマーズ・ガイド」を参照してください。

LUC

LU 6.2 トランスポートとは異なり、LUC トランスポートは NetView プログラム間の通信のみをサポートします。LUC タスク (例えば CNM01LUC) は、LUC トランスポートを使用するためにアクティブにしなければなりません。NV-UNIQ/LUC アラート転送の場合、DSICRTR タスクも活動状態でなければなりません。

OST-NNT

LUC トランスポートと同様に、OST-NNT トランスポートは NetView プログラム間の通信のみをサポートします。OST-NNT セッションを確立するには、中央システムから START DOMAIN コマンドを発行し、ターゲット・システム NNT とのセッションを開始します。OST-NNT セッションについての追加情報は、446 ページの『OST-NNT セッションによる転送』を参照してください。

NetView 設計済みフォーカル・ポイントのサポート

フォーカル・ポイント・アプリケーション は、中央のネットワーク・ノードに常駐し、分散ネットワーク・ノードに常駐するエントリー・ポイント・アプリケーション から、情報を受信します。エントリー・ポイント・アプリケーションからフォーカル・ポイント・アプリケーションに送信される情報は、データの特定期間、例えばアラート・カテゴリーに入ります。このセクションでは、NetView プログラムが、SNA で定義された方法を使用して、運用管理、アラート、およびユーザーの定義した MS データのカテゴリーに関して、フォーカル・ポイントの記録をとる方法を説明します。

NetView プログラムは、「SNA 管理サービス・リファレンス」に記述されている、SNA 管理サービス・フォーカル・ポイント・アーキテクチャーのサブセットを使用します。NetView プログラムによる LU 6.2 フォーカル・ポイント・サポートの扱いは、本書に記載したアーキテクチャーに基づいています。

NetView プログラムに固有である、アラートおよび状況情報を LUC セッションで転送する方法とは対照的にアーキテクチャーに従った方法では、非 NetView プログラムのエントリー・ポイントおよびフォーカル・ポイントまで関連します。また、SNA 管理サービス・フォーカル・ポイント・アーキテクチャーでは、NetView プログラムを、必要な機能サブセットを実施する任意のアプリケーションもしくは装置のアーキテクチャーのフォーカル・ポイントまたはエントリー・ポイントとすることができます。NetView プログラムに、アラート、運用管理、およびユーザーが定義したカテゴリーの情報について、アーキテクチャー・エントリー・ポイントおよび設計済みフォーカル・ポイントとしての機能を持たせることができます。

エントリー・ポイント・アプリケーションがそのフォーカル・ポイントにデータを送信 (転送) すると、エントリー・ポイントはフォーカル・ポイントに NetView LU 6.2 トランスポートの 1 つを介してデータを送信します。LU 6.2 セッションの間のデータ・フローは、MDS-MU の形式になります。

このアーキテクチャーでは、ローカル・フォーカル・ポイントおよびリモート・フォーカル・ポイントが使用されます。

- ローカル・フォーカル・ポイントは、1 つのノードでフォーカル・ポイントとして機能する MS アプリケーションです。

フォーカル・ポイント・アプリケーションが入っているノードの観点からは、フォーカル・ポイントはローカル・フォーカル・ポイント・アプリケーションです (フォーカル・ポイントはそのノードに対してローカル、またはそのノード内に常駐します)。

- リモート・フォーカル・ポイントは別のノードに常駐します。

フォーカル・ポイントが別のノードに常駐しているエントリー・ポイントの観点からは、フォーカル・ポイントはリモート・フォーカル・ポイントです。

例えば、運用管理フォーカル・ポイント・アプリケーション (OPS-MANAGEMENT-NETOP) がノード A に常駐する場合、ノード A は、その運用管理カテゴリーのローカル・フォーカル・ポイントを含んでいると言えます。次に第 2 のノードであるノード B に運用管理エントリー・ポイント (EP-OPS-MANAGEMENT) が含まれている場合、ノード B の運用管理のフォーカル・ポイントがノード A であると、ノード B には運用管理のリモート・フォーカル・ポイントが含まれます。

リモート・フォーカル・ポイントはさらに、1 次フォーカル・ポイントまたはバックアップ・フォーカル・ポイントに分類することができます。あるノードの現行のフォーカル・ポイントは、エントリー・ポイント・アプリケーションがデータを転送するアクティブなリモート・フォーカル・ポイントです。現在のフォーカル・ポイントが 1 次フォーカル・ポイントとバックアップ・フォーカル・ポイントを同時に兼ねることはできません。

- 1 次フォーカル・ポイントが使用可能な場合、それが現在のフォーカル・ポイントになります。

- 1 次フォーカル・ポイントが使用不可の場合、定義済みのバックアップ・フォーカル・ポイントが現在のフォーカル・ポイントとなります。

設計済みフォーカル・ポイントは、そのエントリー・ポイントから受信した情報をフォーカル・ポイントに転送することができます。この処理は、フォーカル・ポイント・ネストと呼ばれます。

次のセクションでは、NetView プログラムが提供するフォーカル・ポイント関連の MS アプリケーションについて説明します。

アーキテクチャーの観点からの説明は、「*SNA Management Services Reference*」の『Focal Point Concepts (フォーカル・ポイントの概念)』を参照してください。

MS-CAPS アプリケーション

MS-CAPS という用語は、設計済みフォーカル・ポイント機能セットの名前である MS 機能 (MS capabilities) の略として使用されることがありますが、本書では、MS-CAPS はフォーカル・ポイントとエントリー・ポイントの関係の確立、終了、および通信を行う MS アプリケーションを指しています。

MS-CAPS アプリケーションでは、他のノードにある MS-CAPS アプリケーションや他のアプリケーションと通信するために、MS トランスポートを通して主ベクトル X'80F0' を送信します。MS-CAPS と対話するアプリケーションを作成したい場合、X'80F0' 主ベクトル内にある以下のサブベクトルについて理解しておく必要があります。

- フォーカル・ポイント許可要求 (X'61')。フォーカル・ポイントはこのサブベクトルをエントリー・ポイントに送信し、そのエントリー・ポイントがフォーカル・ポイントの制御範囲に入るように要求します。
- フォーカル・ポイント許可応答 (X'62')。エントリー・ポイントはこのサブベクトルをフォーカル・ポイントに送り返し、許可要求の受諾または拒否を行います。フォーカル・ポイントが X'62' 受諾サブベクトルをエントリー・ポイントから受信した場合、フォーカル・ポイントの制御範囲にそのエントリー・ポイントが追加されます。

また、エントリー・ポイントは、このサブベクトルをフォーカル・ポイントに送信して、確立済みの関係を取り消すことができます。フォーカル・ポイントに X'62' 取り消しサブベクトルが受信されると、エントリー・ポイントはフォーカル・ポイントの制御範囲から除去されます。

- エントリー・ポイント許可要求 (X'63')。エントリー・ポイントはこのサブベクトルをフォーカル・ポイントに送信し、そのフォーカル・ポイントの制御範囲に入れるように要求します。
- エントリー・ポイント許可応答 (X'64')。フォーカル・ポイントはこのサブベクトルをエントリー・ポイントに送り返し、許可要求の受諾または拒否を行います。フォーカル・ポイントが X'64' 受諾サブベクトルをエントリー・ポイントに送信すると、そのエントリー・ポイントはフォーカル・ポイントの制御範囲に追加されます。
- フォーカル・ポイント通知 (X'E1')。あるカテゴリーのフォーカル・ポイントが変更されると、MS-CAPS は X'E1' サブベクトルを、そのカテゴリーを登録しているすべての MS アプリケーションに送信します。ネットワーク・ノードがそのフ

フォーカル・ポイントを変更する場合、そのネットワーク・ノードにある MS-CAPS は、X'EI' サブベクトルを、すべての実行済みエンド・ノードにある MS-CAPS に送信し、その変更について通知します。例えば、フォーカル・ポイントの変更の原因として、オペレーターの NetView FOCALPT CHANGE コマンドの入力や、セッションの障害が考えられます。

すべてのカテゴリについて現行のフォーカル・ポイントをトラッキングし続けることは MS-CAPS アプリケーションの責任です。NetView MS-CAPS サポートは、「*SNA Management Services Reference*」に記述されている MS_CAPS 機能セット・アーキテクチャーに基づいています。

MS-CAPS は、現行のフォーカル・ポイント情報を VSAM 保管/復元データベースに保管します。MS-CAPS が保管するのは、DEFFOCPT ステートメントで定義されたフォーカル・ポイントの ID と、FOCALPT コマンドまたはセッションの損失によるフォーカル・ポイントの変更情報です。この情報を保管するのであれば、その前に DSISVRT タスクをアクティブにしておく必要があります。NetView プログラムを停止し再始動した場合、MS-CAPS は、保管されたこの情報を使用して、最新の DSI6DST 初期設定の際の 1 次フォーカル・ポイントとバックアップ・フォーカル・ポイントを再度獲得することができます。

DSI6DST 初期設定では、MS-CAPS は DSI6INIT メンバーに入っている DEFFOCPT ステートメントを読み取ります。DEFFOCPT ステートメントでは、1 つの 1 次フォーカル・ポイントと最大 8 つのバックアップ・フォーカル・ポイントを指定することができます。すべての DEFFOCPT ステートメントが読み取られると、MS-CAPS は定義されたフォーカル・ポイントを保管/復元タスクから戻されたフォーカル・ポイント明細と比較します。DSI6DST タスクが最後に初期設定されたあと DEFFOCPT ステートメントに修正がなく、OVERRIDE キーワードが指定されていない場合、MS-CAPS は、保管/復元タスクによって戻されたフォーカル・ポイント名を使用します。つまり、MS-CAPS は各カテゴリ用にこのようなフォーカル・ポイントを獲得しようとします。それ以外のすべての場合、MS-CAPS は DEFFOCPT ステートメントで定義されたフォーカル・ポイント名を使用します。つまり、MS-CAPS は、まず 1 次フォーカル・ポイントを獲得しようとします。獲得できない場合は、バックアップ・フォーカル・ポイントの 1 つを獲得しようとします。バックアップ・フォーカル・ポイントが獲得されるまで、DEFFOCPT ステートメントで指定された順序でバックアップが処理されます。

バックアップが獲得不可能で、さらに MS-CAPS アプリケーションがエンド・ノードで実行中の場合、MS-CAPS はローカル・アプリケーションに、ドメイン・フォーカル・ポイントヘデータを送信するように知らせます (追加情報は、429 ページの『拡張対等通信ネットワーク環境における MS-CAPS』を参照してください)。1 次フォーカル・ポイントが獲得できないときには、MS-CAPS はタイマーを設定し、タイマーが満了すると MS-CAPS は再び 1 次フォーカル・ポイントを獲得しようとします。フォーカル・ポイント (1 次またはバックアップ) が獲得されると、MS-CAPS はローカル・アプリケーションに、獲得されたフォーカル・ポイントヘデータを送信するように知らせます。

あるカテゴリについて定義された複数のフォーカル・ポイント間での競合を MS-CAPS が検出すると、MS-CAPS はタスク・イニシエーターと許可レシーバーにメッセージを出し、競合の結果 MS-CAPS が実行したアクションを示します。

拡張対等通信ネットワーク環境における MS-CAPS

拡張対等通信ネットワーク環境では、ネットワーク・ノードは、エンド・ノード用のサービスを提供します。その 1 つは、マイグレーション・ノード以外のすべてのエンド・ノードに対してドメイン・フォーカル・ポイント の名前および状況を通知するサービスです。

また、エンド・ノードで DEFFOCPT ステートメントか FOCALPT コマンドを使用して、暗黙または明示のフォーカル・ポイントを設定することができます。これを設定した場合、エンド・ノードは、1 次フォーカル・ポイントまたはバックアップ・フォーカル・ポイントに送信できないときにのみ、データをドメイン・フォーカル・ポイントに転送します。エンド・ノードでは、ドメイン・フォーカル・ポイントを制御することも、ドメイン・フォーカル・ポイントを除去することもできません。

障害時の処理

あるエントリー・ポイントの MS-CAPS アプリケーションが、アーキテクチャー・カテゴリー用の 1 次フォーカル・ポイントと通信中にエラー通知を受信した場合、MS-CAPS は以下のことを行います。

- その障害に関する通知を、カテゴリー内にインタレストを登録したすべてのアプリケーションに送信します。
- タイマーが満了したときに、1 次フォーカル・ポイントを再度獲得する処理を試行するように、タイマーを設定します。この処理は、DEFAULTS コマンドの REACQPRI オプションで指定された時間を経過した後に実行されます。DEFAULTS REACQPRI の詳細については、NetView オンライン・ヘルプを参照してください。
- バックアップ・フォーカル・ポイントが存在する場合は、それを獲得するよう試行します。そのバックアップ・フォーカル・ポイントでセッションを確立することができない場合、定義されているバックアップ・フォーカル・ポイントが複数あれば、システムはセッションが確立できるまで、定義されたバックアップ・フォーカル・ポイントとのセッション確立を順に試行します。MS-CAPS がバックアップの獲得に成功すると、そのカテゴリーのローカル・エントリー・ポイント・アプリケーションに通知を送信し、新しいフォーカル・ポイントを知らせます。
- タイマーが満了すると、1 次フォーカル・ポイントを再度獲得できるように試行します。この試行が成功すると、MS-CAPS はローカル・エントリー・ポイント・アプリケーションに通知を送信し、バックアップ・フォーカル・ポイントに取り消し通知を送信します。試行が失敗した場合は、MS-CAPS はタイマーをリセットし、1 次フォーカル・ポイントを再度獲得できるように試行を継続します。

バックアップ・フォーカル・ポイントが現行のフォーカル・ポイントになっていて、MS-CAPS がそのバックアップ・フォーカル・ポイントとの通信中に障害の通知を受信する場合、MS-CAPS はカテゴリーにインタレストを登録したすべてのアプリケーションに通知を送信します。

フォーカル・ポイントのネスト

MS-CAPS はフォーカル・ポイントのネストのサポートを提供しますが、これにより、NetView ノードはローカル・フォーカル・ポイントとリモート・フォーカル・

ポイントとを同時に含むことができます。ローカル・フォーカル・ポイントは、エントリー・ポイントとして機能するローカル・アプリケーションから情報を受信してから、この情報をリモート・フォーカル・ポイントに転送することにより、エントリー・ポイントとして機能することができます。フォーカル・ポイントのネストが誤ってセットアップされている場合、データが無限ループに入ってしまう可能性があります。MS-CAPS はそのようなループを検出し、中断します。ループが検出されると、ループを検出したノードは、指定されたカテゴリーのフォーカル・ポイントを除き、ループを中断します。

FOCALPT コマンドが入力されると、MS-CAPS は FOCALPT コマンドで要求された機能を実行します。例えば、FOCALPT ACQUIRE の場合、MS-CAPS は新規のフォーカル・ポイントを獲得し、直前のフォーカル・ポイントの取り消し (除去) を行います。FOCALPT コマンドで、1 次フォーカル・ポイントとバックアップ・フォーカル・ポイントの変更または除去を行うことができます。

424 ページの図 139 の REGISTER QUERY 出力では、MS-CAPS アプリケーションは、APPL 列で MS_CAPS と識別されます。

設計済みフォーカル・ポイントの制御範囲

設計済みフォーカル・ポイントおよびそのサービスを受けるエントリー・ポイントで、フォーカル・ポイントとエントリー・ポイントの関係を設定したり、終了したり、通信したりできる場合でも、それらの関係を管理する方法が必要です。例えば、フォーカル・ポイントとエントリー・ポイントの関係を制御したり維持したりする場合、集中管理ポイントでこれらの関係を表示する必要があります。フォーカル・ポイントとエントリー・ポイントの関係に障害が起こった場合、集中管理プログラムはその関係を回復する必要があります。

NetView は、アーキテクチャー機能セットがフォーカル・ポイントに用意されています。これは制御範囲マネージャー (SOC-MGR) と呼ばれるもので、フォーカル・ポイントとエントリー・ポイントの関係における集中管理プログラムとしての働きをします。SOC-MGR は、制御範囲にあるすべてのエントリー・ポイントを管理します。制御範囲は、登録済みフォーカル・ポイントとの間に関係が確立されている (もしくは確立される必要がある) すべてのエントリー・ポイントとして定義されます。

フォーカル・ポイントにおける制御範囲の機能

あるアプリケーションがフォーカル・ポイントとしての登録処理を行うとき、フォーカル・ポイントの対象にする管理サービス・データのカテゴリーを指定します。それによって、フォーカル・ポイントとエントリー・ポイントの関係がそのカテゴリーについて確立されます。フォーカル・ポイントおよびエントリー・ポイントでの MS-CAPS アプリケーションには、そのフォーカル・ポイントとエントリー・ポイントの間の関係を確立する責任があります。フォーカル・ポイントにおける MS-CAPS アプリケーションの一部である SOC-MGR により、MS-CAPS は制御範囲の管理を自動化することができ、また、そのフォーカル・ポイントのオペレーターは、そのフォーカル・ポイントの制御範囲のエントリー・ポイントをすべて管理することができます。

制御範囲の MS-CAPS 管理: 制御範囲の管理サービスを自動化するため、MS-CAPS は以下のことを行う必要があります。

- フォーカル・ポイントの制御範囲にあるエントリー・ポイントのリストを維持すること。
- フォーカル・ポイントの制御範囲にある各エントリー・ポイントの状態を保守すること。
- エントリー・ポイントとフォーカル・ポイントの関係が失われたときに、そのエントリー・ポイントとの関係を再度確立しようとする。この処理の方法は、エントリー・ポイントのタイプと状態によって異なります。
- NetView 初期設定中に、制御範囲構成ファイル (DSI6SCF) から情報を読み取り、この情報を使って制御範囲環境のセットアップを行うこと。
- NetView のリカバリー中に、制御範囲環境を復元すること。

制御範囲のオペレーター管理: SOC-MGR を使用すると、オペレーターはフォーカル・ポイントで以下の管理機能を制御範囲環境に対して行うことができます。

- FOCALPT DELETE コマンドを使用し、制御範囲からエントリー・ポイントを削除すること。
- FOCALPT DISPSOC コマンドを使用し、制御範囲にあるエントリー・ポイントの名前および状態を表示すること。
- FOCALPT REFRESH コマンドを使用し、制御範囲環境を初期設定すること。
- SOC-MGR の初期設定が済んでから、制御範囲構成ファイルにエントリー・ポイントを追加し、そのあと FOCALPT REFRESH コマンドを使用して、その変更を SOC-MGR に動的に読み込むこと。

制御範囲タイプ

フォーカル・ポイントでの SOC-MGR の機能の 1 つに、各エントリー・ポイントごとの制御範囲タイプの保守があります。制御範囲タイプとは、エントリー・ポイントを制御範囲に取り込む方法を定義するものです。エントリー・ポイントの制御範囲を表示するには、FOCALPT DISPSOC コマンドを使用してください。制御範囲タイプには、以下の 2 通りがあります。

EXPLICIT

オペレーター・コマンドが出されたか、そのエントリー・ポイントが制御範囲構成ファイルに定義されていたために、フォーカル・ポイントはそのエントリー・ポイントとの関係の確立を開始しました。フォーカル・ポイントは、エントリー・ポイントとの関係確立が成功するまで続けます。関係がいったん確立されると、エントリー・ポイントは関係が失われる場合に、再度その関係を確立する責任を持ちます。

IMPLICIT

そのエントリー・ポイントは、フォーカル・ポイントとの関係の確立を開始しています。エントリー・ポイントは、関係が失われると再度その関係を確立する責任を持ちます。

フォーカル・ポイントとエントリー・ポイントの関係が確立されるとき、制御範囲タイプの EXPLICIT の優先順位は IMPLICIT よりも高くなっています。例えば、フォーカル・ポイントがあるエントリー・ポイントとの関係を開始した場合、このエントリー・ポイントは、そのフォーカル・ポイントの制御範囲に明示的に取り込まれたと考えられます。そのため、このエントリー・ポイントは、SOC-MGR で保守される情報において EXPLICIT の制御範囲タイプを持つと見なされます。

フォーカル・ポイントの制御範囲に明示的に取り込まれたエントリー・ポイントは、同じフォーカル・ポイントとの関係を開始することができます。フォーカル・ポイントとの関係を開始したいというエントリー・ポイントの要求は、正常終了します。しかし、制御範囲タイプの EXPLICIT は IMPLICIT よりも優先順位が高いため、SOC-MGR はこのエントリー・ポイントを EXPLICIT 制御範囲タイプでリストします。

制御範囲の状態

フォーカル・ポイントの SOC-MGR は、制御範囲にあるエントリー・ポイントの状態に関する情報を維持します。エントリー・ポイントの状態は、以下の 3 つによって決まります。

- そのエントリー・ポイントの制御範囲タイプ
- そのエントリー・ポイントの以前の状態
- そのエントリー・ポイントに影響したイベント

エントリー・ポイントの制御範囲状態を表示するには、FOCALPT DISPSOC コマンドを使用してください。エントリー・ポイントの状態は、次のとおりです。

ADD PENDING

フォーカル・ポイントは、そのエントリー・ポイントを制御範囲に取り込もうとしましたが、エントリー・ポイントからの応答をまだ受信していません。要求がエントリー・ポイントに受け入れられたことを示す応答がフォーカル・ポイントに受信されると、エントリー・ポイントは ACTIVE 状態に入ります。

ACTIVE

フォーカル・ポイントは、そのエントリー・ポイントに現在サービスを提供しています。フォーカル・ポイントとエントリー・ポイントの関係は確立されており、そのエントリー・ポイントはフォーカル・ポイントの制御範囲にあると見なされます。

DELETE ADD PENDING

そのエントリー・ポイントが ADD PENDING 状態になっていた間に、そのフォーカル・ポイントのオペレーターが FOCALPT DELETE コマンドを出しました。エントリー・ポイントは、フォーカル・ポイントの制御範囲にまだあり、そのエントリー・ポイント向けのサービスを別のフォーカル・ポイントが代わりに行うまで、もしくは、フォーカル・ポイントとエントリー・ポイントとの間のセッションが失われるまでそのフォーカル・ポイントからのサービスを受け続けます。

DELETE PENDING

エントリー・ポイントが ACTIVE 状態であったときに、そのフォーカル・ポイントのオペレーターが FOCALPT DELETE コマンドを出しました。エントリー・ポイントは、フォーカル・ポイントの制御範囲にまだあり、そのエントリー・ポイント向けのサービスを別のフォーカル・ポイントが代わりに行うまで、もしくは、フォーカル・ポイントとエントリー・ポイントとの間のセッションが失われるまでそのフォーカル・ポイントからのサービスを受け続けます。

INACTIVE

そのエントリー・ポイントが ACTIVE 状態であるときに、フォーカル・ポイントとエントリー・ポイントの関係が失われました。エントリー・ポイントは、その関係を再度確立するか、エントリー・ポイントがそのフォーカル・ポイントの除去要求を出すまで、INACTIVE 状態のままになります。

INACTIVE RETRY

そのエントリー・ポイントが ADD PENDING 状態であったときに、フォーカル・ポイントとエントリー・ポイントの関係が失われました。フォーカル・ポイントは、フォーカル・ポイントとエントリー・ポイントの関係を再度確立しようとしてします。

UNKNOWN

この状態が適用されるのは、IMPLICIT 制御範囲タイプのエントリー・ポイントか、もしくは DELETE PENDING か DELETE ADD PENDING の状態の EXPLICIT エントリー・ポイントに限られます。エントリー・ポイントが UNKNOWN 状態に入るのは、そのフォーカル・ポイントで NetView プログラムがリカバリー操作を実行したあとです。リカバリー操作中にフォーカル・ポイントとエントリー・ポイントの関係を再度確立するのはエントリー・ポイントの責任なので、その関係の消失がエントリー・ポイントに認識されているかどうかは、フォーカル・ポイントにはわかりません。認識されている場合は、別のフォーカル・ポイントとの関係を確立することができます。認識されていない場合、そのエントリー・ポイントは同じフォーカル・ポイントとの関係を維持し続けます。エントリー・ポイントがそのフォーカル・ポイントとの関係を再度確立した場合、エントリー・ポイントの状態は ACTIVE に変わります。

制御範囲環境のセットアップ

制御範囲構成ファイル DSI6SCF は、どのエントリー・ポイントが明示的に特定のフォーカル・ポイントの制御範囲に取り込まれるかを定義するものです。このファイルは、NetView 初期設定中に読み込まれ、フォーカル・ポイントとエントリー・ポイントの関係の制御範囲環境をセットアップするために使用されます。NetView 初期設定のあとには、いつでも制御範囲構成ファイルの更新を行って、フォーカル・ポイントとエントリー・ポイントの関係を最新表示にしたり変更したりすることもできます。制御範囲構成ファイルには、以下の情報が入っています。

- ・ エントリー・ポイントの名前
- ・ 1 次フォーカル・ポイント・カテゴリーの名前
- ・ 1 次フォーカル・ポイントの名前
- ・ バックアップ・フォーカル・ポイントの名前 (任意指定)

制御範囲構成ファイルを定義する方法については、「*IBM Tivoli NetView for z/OS* インストール:概説」を参照してください。

制御範囲環境の更新または変更: NetView プログラムの開始後、フォーカル・ポイントとエントリー・ポイントの関係を動的に最新表示したり変更したりするには、FOCALPT REFRESH コマンドを使用できます。FOCALPT REFRESH コマンドを出すと、フォーカル・ポイントにある MS-CAPS アプリケーションは、制御範囲構成ファイルを読み込み、現行の制御範囲環境を更新します。制御範囲構成ファイルに定義されたフォーカル・ポイントとエントリー・ポイントの関係は、現行の制御範囲環境の関係より優先されます。例えば、制御範囲構成ファイルに EXPLICIT エントリー・ポイントが定義されていれば、FOCALPT REFRESH コマンドが出された場合に、現行の制御範囲環境で IMPLICIT 制御範囲タイプであるエントリー・ポイントは EXPLICIT 制御範囲タイプに変更されます。

また、現行の環境でフォーカル・ポイントの制御範囲に EXPLICIT エントリー・ポイントが存在していても、FOCALPT REFRESH コマンドが出されたときに構成ファイルに定義されていない場合、そのエントリー・ポイントはフォーカル・ポイントの制御範囲から削除されます。

制御範囲環境の復元

フォーカル・ポイントの MS-CAPS アプリケーションは、制御範囲にあるエントリー・ポイントについての情報を VSAM 保管/復元データベースに保管します。エントリー・ポイントが制御範囲から出る場合、そのエントリー・ポイントに関する情報は、VSAM 保管/復元データベースから削除されます。

NetView プログラムまたは DSI6DST タスクが終了した後に回復する場合、MS-CAPS は VSAM 保管/復元データベースを検査します。保管/復元情報が存在すると、MS-CAPS はその情報を使用して、最近のフォーカル・ポイントとエントリー・ポイントの環境を復元します。

エントリー・ポイントの制御範囲タイプが IMPLICIT の場合、もしくは DELETE PENDING か DELETE ADD PENDING の状態である EXPLICIT エントリー・ポイントの場合、そのエントリー・ポイントは終了する前に UNKNOWN 制御範囲状態で復元されます。このあとは、そのフォーカル・ポイントとの関係を再度確立するのはエントリー・ポイントの責任になります。エントリー・ポイントは、そのフォーカル・ポイントとの関係が失われたことを認識している場合は関係を再度確立し、そのエントリー・ポイントの制御範囲の状態は ACTIVE に変わります。エントリー・ポイントが失われたことを認識していない場合は、フォーカル・ポイントとの関係を再度確立することではなく、エントリー・ポイント制御範囲の状態は UNKNOWN のままになります。

設計済みフォーカル・ポイントの定義方法 (DEFFOCPT)

図 141 では、DSI6INIT (CNMS1040) で、アラート、運用管理、およびユーザーが定義したカテゴリーの典型的なフォーカル・ポイントの定義を紹介しています。ここでは、典型的な運用管理エントリー・ポイント定義も紹介しています。

DEFFOCPT および DEFENTPT の詳細については、「*IBM Tivoli NetView for z/OS アドミニストレーション・リファレンス*」を参照してください。DEFENTPT が運用管理カテゴリーにのみ適用されることに注意してください。

```
DSTINIT FUNCT=OTHER,XITDI=DSI6IDM
DEFFOCPT TYPE=ALERT,PRIMARY=NETA.CNM02
DEFFOCPT TYPE=OPS_MGMT,PRIMARY=NETA.CNM02,BACKUP=NETB.CNM99
DEFFOCPT TYPE=OPS_MGMT,BACKUP=CNM03
DEFFOCPT TYPE=USERCAT,BACKUP=NETB.CNM99
DEFFOCPT TYPE=USERCAT,PRIMARY=NETA.CNM02,OVERRIDE
DEFFOCPT TYPE=USERCAT,BACKUP=*.CNM05
DEFFOCPT TYPE=USERCAT,BACKUP=CNM03
DEFENTPT EPONLY=YES
END
```

図 141. DSI6INIT における典型的なフォーカル・ポイントとエントリー・ポイントの定義コメント

DEFFOCPT および DEFENTPT ステートメントは、DSI6DST タスクの初期設定時に MS-CAPS により処理されます。

ALERT-NETOP アプリケーション

ハードウェア・モニターの BNJDSERV タスクの初期設定時に、このタスクはハードウェア・モニターをアーキテクチャー・アラート・フォーカル・ポイントである ALERT-NETOP として登録します。NetView プログラムは自動的に登録を行い、定義は必要ありません。NetView ALERT-NETOP サポートは、ALERT_NETOP 機能セットおよび EP_ALERT 機能セットのアーキテクチャーに基づいています。詳細については、「*SNA Management Services Reference*」を参照してください。

NetView の ALERT-NETOP をインプリメントすると、他のアプリケーションから MS トランスポートを介して MDS-MU 内の CP-MSU の形式でアラートを受信することができます。そのようなアプリケーションは EP-ALERT アプリケーションとして機能し、NetView ノードまたは非 NetView ノードに常駐することがあります。

NetView ALERT-NETOP は、EP-ALERT アプリケーションとして機能すること、また、他の ALERT-NETOP アプリケーション (NetView または非 NetView) に MS トランスポートを介して MDS-MU 内の CP-MSU 形式でアラートを転送することができます。そのため、ALERT-NETOP は MS トランスポートを介して、アラートの受信と転送を行います。これには、MS トランスポートの CP-CP セッションを介して、アラートの送受信ができることも含まれます。

ALERT-NETOP は、NetView プログラム間インターフェースから CP-MSU 形式でアラートを受信することもできます。CP-MSU には、1 つ以上のアラート主ベクトルが含まれていることがあります。ハードウェア・モニターはこのアラート主ベクトルを分割し、それぞれを個別に処理します。ハードウェア・モニターが主ベクトルを処理する方法については、108 ページの『NetView プログラムのハードウェア・モニター・データ、および MSU の経路指定』を参照してください。

ALERT-NETOP は、REGISTER QUERY コマンド出力の APPL 列で ALERT として表示されます。(424 ページの図 139 を参照してください。) ALERT は ALERT-NETOP の省略語で、アラート・フォーカル・ポイント・アプリケーションを示すアーキテクチャー名です。NetView が EP-ALERT アプリケーションを登録しないことに注意してください。ALERT-NETOP は EP-ALERT アプリケーションとして機能するため、NetView が明示的な EP-ALERT アプリケーションを登録する必要はありません。

LU 6.2 で転送されたアラートの表示

ハードウェア・モニターの Alerts Dynamic、Alerts Static、および Alerts History パネルでは、リモート・ノード・アプリケーションから LU 6.2 を介して ALERT-NETOP に転送されたアラートの横に @ インディケーターが表示されます。NetView のノードに常駐するアプリケーションは、ローカル・アプリケーションと考えられますが、ほとんどの例外なしに、ローカル・アプリケーションから送信されたアラートには @ インディケーターが付いていません。追加情報については、「*IBM Tivoli NetView for z/OS ユーザーズ・ガイド: NetView*」を参照してください。

LU 6.2 を使用するアーキテクチャー・アラートの転送の指定

NetView は、ALERT-NETOP アプリケーションを通じて、LU 6.2 トランスポートを介して送信されるアラートの受信をサポートします。NetView ALERT-NETOP

は、アーキテクチャー ALERT-NETOP アプリケーションとして機能し、EP-ALERT として機能するアプリケーションから送信されるアラートを受信します。

NetView ALERT-NETOP アプリケーションを使用して、LU 6.2 を介してアラートを転送するには、CNMSTYLE メンバーの ALERTFWD ステートメント上で SNA-MDS オプションを指定します。ALERTFWD ステートメントにより、NetView がアラートを転送する方法を、次の 2 つから選択することができます。すなわち、SNA-MDS/LU 6.2 (ALERTFWD SNA-MDS の場合) によるか、または NV-UNIQ/LUC (ALERTFWD NV-UNIQ の場合) によるかのいずれかです。ALERTFWD ステートメントの詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

NV-UNIQ を選択する場合、ALERT-NETOP は LU 6.2 を介してアラートを受信することができますが、LU 6.2 を介してアラートを転送することはできません。つまり、444 ページの『LUC を使用したアラート転送』に記述しているように、アラートを転送できるのは、LUC を介する場合だけです。SNA-MDS を選択すると、ALERT-NETOP は、アーキテクチャー ALERT-NETOP および EP-ALERT として機能します。したがって、LU 6.2 を介して、そのフォーカル・ポイントへアラートを転送することができます。次のセクションでは、SNA-MDS アラート転送 (アーキテクチャー・アラート転送、LU 6.2 アラート転送、または LU 6.2 を介するアラート転送 と呼ばれる) について説明します。

NetView プログラムではないフォーカル・ポイントへのアラートの転送

NetView のアラート・フォーカル・ポイントとして、AS/400 のような NetView プログラム以外のプロダクトを選択することができます。そのようなプロダクトの観点から、エントリー・ポイントの NetView プログラムからそのプロダクトが受信するアラートは、以下のカテゴリーに区分されます。

- アーキテクチャーに従うアラート

例えば、サブベクトル X'92' を伴う汎用アラート主ベクトル X'0000'。

- アーキテクチャーに従わないアラート

例えば、X'1330/X'132F' ダブル主ベクトルの OSI のアラーム。

また、このカテゴリーに区分されるアラートの中には、アーキテクチャーに従っているものの、非 NetView プロダクトがサポートせずに受信するアラートもあります。例えば、アーキテクチャーにより解決アラート主ベクトル X'0002' は ALERT-NETOP に転送することができますが、非 NetView プロダクトの中には、アーキテクチャーのサブセットを組み込んでいないために、この主ベクトルの受信をサポートしていないものがあります。

NetView プログラム以外のプログラムのフォーカル・ポイントおよび設計済みアラート: これらのアラートは、他のプロダクトによって適切に処理されます。主ベクトル以外のアラート (RECFMS など) は、UNDETERMINED という推定原因とともに表示されることがあります。詳しい情報については、製品の資料を参照してください。

NetView プログラム以外のプログラムのフォーカル・ポイントおよび設計されていないアラート: フォーカル・ポイントが NetView プログラム以外のプロダクトであるため、そのフォーカル・ポイントで非アーキテクチャー・レコードの処理方法が分からないことがあります。つまり、処理方法はフォーカル・ポイント・プロダクトによって異なります。例えば、非汎用アラートの主ベクトル X'0000' (これにはサブベクトルの X'92' は含まれていない) は、フォーカル・ポイントに送信されるように構築されていませんが、AS/400 プロダクトはこの受信をサポートしています。フォーカル・ポイントが非アーキテクチャー・レコードを受信する場合、プロダクトにより異なりますが、たいてい以下のうち 1 つ以上のアクションを行います。

- エラー・メッセージを出す。
- MDS エラー・メッセージ (MDS-MU 内の X'1532' 主ベクトル) またはアプリケーション・エラー・メッセージ (CP-MSU 内の X'1532' 主ベクトル) をエントリー・ポイント NetView プログラムに送り返します。

エントリー・ポイント NetView が MDS エラー・メッセージまたはアプリケーション・エラー・メッセージを受信すると、このエントリー・ポイントは、CNMSTYLE メンバーの ALERTFWD ステートメントで指定されているオプションに応じて、メッセージ BNH094I または BNH095I を発行します。

ALERTFWD についての詳細は、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照し、CNMSTYLE メンバーについての詳細は、「*IBM Tivoli NetView for z/OS* インストール: 概説」を参照してください。

- 設計されていないアラートを無視する (廃棄する)。

設計されていないアラートは、これらのフォーカル・ポイントで適切に処理されない場合があります。詳細については、その製品の資料を参照してください。

注: エントリー・ポイント NetView プログラムから転送されるすべてのアラートを、フォーカル・ポイントで適切に処理させるには、フォーカル・ポイントは NetView プログラムのバージョン 3 以降である必要があります。

ユーザー定義アプリケーションからのアラートの転送

443 ページの『ユーザー定義のカテゴリおよびユーザー定義のアプリケーション』に説明してあるように、ユーザーはユーザー定義のアプリケーションを作成することができます。ユーザー定義のアプリケーションは、ALERT-NETOP にアラートを送信することができます。これを行うには、ユーザー定義のアプリケーションが MS トランスポートで登録するときに、アプリケーションはカテゴリ ALERT でインタレストを登録しなければなりません。

一度登録すると、MS-CAPS はアプリケーションに通知 (主ベクトル X'80F0' およびサブベクトル X'E1' を持つ MDS-MU) を送信します。この通知には、現行のアラート・フォーカル・ポイントの完全修飾名 (NETID 名、NAU 名、およびアプリケーション名) が入っています。(現行のアラート・フォーカル・ポイントは、通常 NetView ALERT-NETOP です。) ユーザーのアプリケーションがこの通知を受信した後で、アプリケーションは、アラート・フォーカル・ポイントにアラートを送信することができます。そして、送信することにより、アプリケーションは、アーキテクチャー EP-ALERT として機能しています。アラートは、CP-MSU 内でカプセ

ル化しなければなりません。また、CP-MSU は MDS-MU 内でカプセル化しなければなりません。送信されるすべてのアラートは、システム・ネットワーク体系ライブラリーに準拠する必要があります。

LU 6.2 を介してローカル・アプリケーションから送信されたアラートを ALERT-NETOP が受信すると、受信されたアラートは、普通のローカル・アラートとして処理されます。例えば、このようなアラートには、Alerts Dynamic パネルで @ インディケーターが付いていません。これは、アラートがリモート・ノードから転送されたものではないからです。

NetView 中間ノード・フォーカル・ポイント

NetView プログラムにアラート・フォーカル・ポイントがあり、さらに NetView プログラムが LU 6.2 を使用して転送されたアラートを受信する場合には、このアラートは ALERT-NETOP によって NetView のフォーカル・ポイントに再度転送されます。この場合、NetView プログラムは中間ノード・フォーカル・ポイント (ネスト・フォーカル・ポイント としても知られている) になります。これは、エントリ・ポイントがアラートを中間ノード・フォーカル・ポイントに転送し、さらにこのアラートをフォーカル・ポイントに再度転送するためです。ユーザーは、中間ノード・フォーカル・ポイントを 1 つも持たないことも、1 つ以上持つことも可能です。そして、偶然にループを作成した場合には、MS-CAPS アプリケーションはループを検出し、中断します。LU 6.2 を使用して中間ノード・フォーカル・ポイントがアラートを転送する方法を理解するには、図 142 を参照してください。

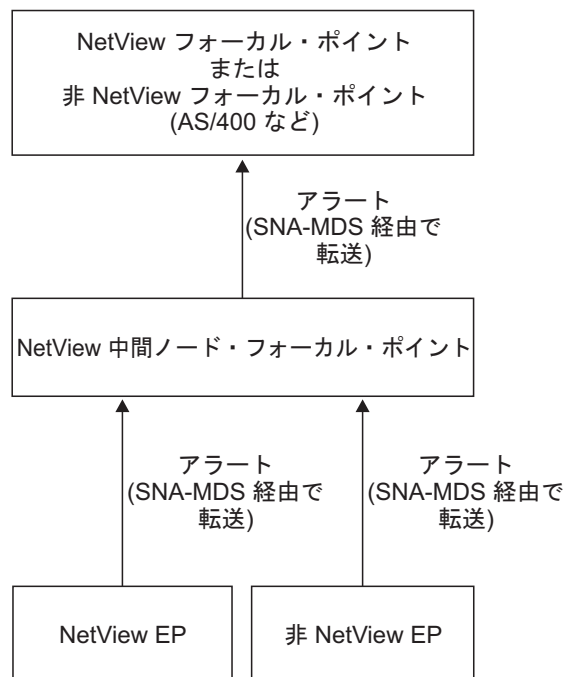


図 142. LU 6.2 を使用した NetView 中間ノード・フォーカル・ポイントのアラート転送

LU 6.2 を介して転送されたアラートだけが、中間ノード・フォーカル・ポイントによって再度転送することができます。このようなアラートを受信する中間ノード・フォーカル・ポイントは、SNA-MDS/LU 6.2 または NV-UNIQ/LUC アラート転送方式を使用して、そのアラートを再度転送することができます。LUC を介して転送

されたアラートに関しては、再度転送されません。LUC を介して転送されたアラートは、エントリー・ポイントからフォーカル・ポイントに一度だけしか転送されません。受信するフォーカル・ポイントは、このアラートを再度転送することができません。LUC アラート転送については、ワン・ホップ・アラート転送方式と考えることができます。

中間ノード NetView に、データをハードウェア・モニター・データベースに記録せずに、単に中間ノードをパススルー するようにしたい場合、BNJMBDST に ALRTINFP NORECORD ステートメントを指定します。ALRTINFP の設定値は、LU 6.2 を使用してリモート・ノードから転送されたアラートにだけ適用されます。つまり、他のすべてのアラートには影響を与えません。詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」の ALRTINFP を参照してください。

最終の (図で一番上の) NetView フォーカル・ポイントにおいて、エントリー・ポイント・アラートをハードウェア・モニター・データベースに記録するドメイン名は、次のように獲得されます。

注: これは、Alerts Dynamic パネル、Alerts Static パネル、および Alerts History パネルの DOMAIN 列に表示されるドメイン名で、いくつかの他のハードウェア・モニター・パネルの最上部の図形階層に表示されます。

- ・ エントリー・ポイントがバージョン 3 以降の NetView プログラムであり、最終のフォーカル・ポイントがバージョン 3 以降の NetView プログラムである場合、最終のフォーカル・ポイントの Alerts Dynamic パネルにアラートを表示させたときに DOMAIN 見出しの下に表示されるドメイン名が、そのエントリー・ポイントのドメイン名です。

アラートは、エントリー・ポイント NetView ドメイン名に対してフォーカル・ポイント・データベースに記録されます。1 つのアラート・レコードだけが、データベースに記録され、データの完全セットがエントリー・ポイント・データベースに存在します。1 つのアラート・レコードをデータベースに記録することにより、データベースのストレージとプロセッサ時間が節約されます。

バージョン 3 以降の最終フォーカル・ポイントのオペレーターは、分散データベース検索機能により Alerts Static パネルから SEL# M を入力し、さらに SDOMAIN コマンドにより、エントリー・ポイント・データベースからハードウェア・モニター・データを検索することができます。追加情報については、「*IBM Tivoli NetView for z/OS* ユーザーズ・ガイド: NetView」および NetView オンライン・ヘルプを参照してください。

LU 6.2 セッションが、最終フォーカル・ポイントとエントリー・ポイントとの間で確立される限り、中間ノードがいくつ存在する (または 1 つも存在しない) かということは重要ではありません。LU 6.2 セッションを最終フォーカル・ポイントとエントリー・ポイントの間で確立できない場合、分散データベース検索機能は失敗します。ただし、SDOMAIN コマンドは LU 6.2 セッションを確立できないと判断すると LUC セッションまたは OST-NNT セッションを確立しようとするため、SDOMAIN コマンドは正常終了することもあります。

- ・ エントリー・ポイントが NetView プログラムではなく、最終フォーカル・ポイントが NetView プログラムである場合、最終フォーカル・ポイントのアラート動的

パネルにアラートが表示され、DOMAIN 見出しの下に表示されるドメイン名が最終フォーカル・ポイントのドメイン名になります。

これは、最終フォーカル・ポイントのバージョンおよびリリース・レベルにかかわらず当てはまります。アラートは、ローカル・アラートであるかのように、フォーカル・ポイントのデータベースに記録されます。

- エントリー・ポイントがバージョン 3 以降の NetView プログラムであり、最終のフォーカル・ポイントが V3R1 より前の NetView プログラムである場合、最終フォーカル・ポイントのアラート動的パネルにアラートが表示され、DOMAIN 見出しの下に表示されるドメイン名が、最終フォーカル・ポイントのドメイン名になります。

これは、V3R1 より前の NetView プログラムは、受信した LU 6.2 転送アラートすべてを、非 NetView プログラムから転送されたかのように扱うためです。アラートは、ローカル・アラートであるかのように、フォーカル・ポイントのデータベースに記録されます。

SNA-MDS/LU 6.2 転送アラート用のフィルターの記録

非 NetView エントリー・ポイントまたはローカル・アプリケーションから LU 6.2 を使用して転送されたアラートには、アラートがローカル・アラートであるかのように提供されたハードウェア・モニター記録フィルターがあります。これらのアラートが記録フィルターを通過すると、データの完全セットがハードウェア・モニター・データベースに記録されます。このデータは、ゼロ、1 つ、または複数のイベント・レコードと統計レコード、およびアラート・レコードなどから構成されます。

リモート・ノードの NetView エントリー・ポイントから LU 6.2 を使用して転送されたアラートにも、アラートがローカル・アラートであるかのように適用されたハードウェア・モニター記録フィルターがあります。ただし、AREC および ESREC 記録フィルターは、常に強制的に PASS されます。これらのアラートはそれぞれ、ハードウェア・モニター・データベースに単一のアラート・レコードとして記録され、データの完全セットは、エントリー・ポイントでのみ使用することができます。このプロセスは、アラートのみ の記録方式として知られていますが、LUC を使用して転送されたアラートもアラートのみとして記録されます。この方法により、フォーカル・ポイントのデータベースがすぐに一杯にならないようになり、プロセッサ時間の使用も減ります。

ハードウェア・モニター ROUTE 記録フィルターを使用して、NetView プログラムによって転送するアラートを指定することができます。しかし、アラートは ROUTE フィルターに達する前に、ESREC フィルターと AREC フィルターを通過する必要があります。また、一度 LUC から転送されたアラートが、再度転送されることはありません。SRFILTER コマンドは、ハードウェア・モニターからフィルター設定を指定するために使用し、SRF アクションは、自動化テーブルからフィルター設定を指定するために使用します。自動化テーブルの SRF アクションは、LUC を使用して転送されたアラートを除くすべてのアラート用の記録フィルターを無効にすることができます。例えば、SRF アクションを使用して非 NetView エントリー・ポイントのアラートをアラートのみとして記録したり、エントリー・ポイント NetView のアラートをデータの完全セットとともに (アラートのみではない) 記録することができます。

SRFILTER コマンドについては、342 ページの『アラートのフィルター操作』を参照し、SRF アクションについては、240 ページの『処置』を参照してください。

フォーカル・ポイントが使用できないときのアラートのキューイング

フォーカル・ポイントが使用できないときに NetView エントリー・ポイントによって受信されたアラートは、アラート・キャッシュ内で保留 (*held*) というマークが付けられます。アラート・キャッシュの定義方法については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」の ALCACHE を参照してください。MS-CAPS が後に正常にフォーカル・ポイントを再度獲得すると、MS-CAPS は ALERT-NETOP に、フォーカル・ポイントが再度獲得されたこと、さらに ALERT-NETOP がアラート・キャッシュを通してループし、保留状態のこれらのアラートを処理することを通知します。この処理は、現在保留状態のアラートに対して ROUTE 記録フィルターを再度適用することを最初に行います。というのは、ROUTE 記録フィルターが最初にアラートに適用されたときには、そのアラートには保留というマークが付けられていないからです。ROUTE 記録フィルターを通過すると、アラートはフォーカル・ポイントに転送されます。

新しいフォーカル・ポイントが獲得される以前は、アラート・キャッシュが定義されていないこともあり、保留アラートがアラート・キャッシュをロールオフすることもあります。そのようなアラートがフォーカル・ポイントに転送されることはありませんが、これらのアラート数のカウントは保持されます。(このカウントは 10,000 で折り返しになります。) フォーカル・ポイントがあとで再度獲得されると、DSI382I メッセージが発行され、メッセージ内にこのカウントを表示します。

1 つ以上の NetView エントリー・ポイントから転送される保留アラートで、フォーカル・ポイントがあふれることがあります。このあふれを起こしたくない場合には、NetView エントリー・ポイントに ROUTE 記録フィルターを設定することができます。これを設定すると、フォーカル・ポイントが再度獲得されるときに保留アラートをブロックし、フォーカル・ポイントに転送しないようにします。ここでは、NPDA SRFILTER ROUTE BLOCK E HELD コマンドを使用します。このコマンドは、CNMSTYLE メンバー内ではコメント化されます (NPDA.PDFILTER ステートメントを参照してください)。このコマンドのコメントを外し、保留アラートがフォーカル・ポイントに転送されないようにすることができます。NetView エントリー・ポイントからの保留アラートは、フォーカル・ポイントではなくエントリー・ポイントでブロックしなければなりません。非 NetView エントリー・ポイントからの保留アラートは、NetView フォーカル・ポイントでブロックしなければなりません。詳細については、440 ページの『SNA-MDS/LU 6.2 転送アラート用のフィルターの記録』、および NetView オンライン・ヘルプの SRFILTER コマンドを参照してください。

フォーカル・ポイント NetView が非 NetView エントリー・ポイントからの保留アラートであふれた場合、このような保留アラートをフィルターにかけるためにフォーカル・ポイントの AREC 記録フィルターを設定することができます。例えば、NPDA SRFILTER AREC BLOCK E HELD コマンドを入力すると、保留ビットがセットされたサブベクトル 92 を含む着信アラートをブロックします。ハードウェア・モニターのデフォルト AREC 記録フィルターは、すべてではないが、多くの保留アラートをブロックします。これは、NPDA DF AREC コマンドを出すことができます。

SNA-MDS/LU 6.2 転送アラート用の分散データベース検索

NetView フォーカル・ポイントでは、分散データベース検索が選択されたアラートに行われるときには、MS トランスポートか LUC トランスポートのいずれかが使用され、エントリー・ポイントのハードウェア・モニター・データベースからデータを検索します。分散データベース検索に使用されるトランスポートは、フォーカル・ポイントがアラートを受信したときのトランスポートと同じです。例えば、フォーカル・ポイントが LU 6.2 を介してアラートを受信した場合、このアラートに対する分散データベース検索がこの後行われるたびに、分散データベース検索も LU 6.2 を使用します。

分散データベース検索の詳細については、「*IBM Tivoli NetView for z/OS* ユーザーズ・ガイド: *NetView*」を参照してください。 中間ノードが 438 ページの

『NetView 中間ノード・フォーカル・ポイント』および 444 ページの『LUC を使用したアラート転送』の説明のように組み込まれているときには、分散データベース検索が失敗することがあります。

SNA-MDS/LU 6.2 転送アラート用の 2 次記録

LUC アラートを転送すると、ハードウェア・モニター 2 次記録は、フォーカル・ポイントで行われません。SNA-MDS/LU 6.2 アラート転送では、2 次記録を使用することができます。2 次記録についての詳細は、「*IBM Tivoli NetView for z/OS* ユーザーズ・ガイド: *NetView*」を参照してください。

XITCI 出口および SNA-MDS/LU 6.2 転送アラート

XITCI 出口、および SNA-MDS/LU 6.2 は転送アラートについての説明は、「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」を参照してください。

MS-CAPS および FOCALPT コマンドで提供されるサービス

ALERT-NETOP は体系的 EP-ALERT として機能するため、MS-CAPS および FOCALPT コマンドで提供されるサービスは、ALERT-NETOP で使用可能です。MS-CAPS によって提供される機能についての説明は、427 ページの『MS-CAPS アプリケーション』を参照してください。また、453 ページの『フォーカル・ポイントの変更、除去、およびリスト』も参照してください。

LINK-SERVICES-NETOP アプリケーション

ハードウェア・モニターの BNJDSERV タスクの初期設定時に、ハードウェア・モニターは、LINK-SERVICES-NETOP、つまり体系的なリンク・イベント (主ベクトル X'0001') フォーカル・ポイントとして登録されます。NetView プログラムは自動的に登録を行い、定義は必要ありません。NetView LINK-SERVICES-NETOP サポートは、「*SNA Management Services Reference*」に記述されている LINK_SERVICES_NETOP 機能セット・アーキテクチャーに基づいています。

NetView LINK-SERVICES-NETOP 機能は、MS トランスポートを介し、MDS-MU 内の CP-MSU の形式で、他のローカル・アプリケーションからリンク・イベントを受信することができます。送信アプリケーションは、NetView プログラムと同じノードに常駐するローカル・アプリケーションでなければなりません。

LINK-SERVICES-NETOP は、424 ページの図 139 の REGISTER QUERY コマンド出力の APPL 列で LINKSERV として表示されます。LINKSERV は

LINK-SERVICES-NETOP の省略語で、リンク・イベント・フォーカル・ポイント・アプリケーションのアーキテクチャー名です。

OPS-MGMT-NETOP および EP-OPS-MGMT アプリケーション

DEFFOCPT ステートメントを追加して他のフォーカル・ポイントを指定しない限り、または DSI6INIT (CNMS1040) の DEFENTPT EPONLY=YES ステートメントを追加しない限り、NetView プログラムも運用管理カテゴリ (OPS-MANAGEMENT-NETOP) 用の体系的フォーカル・ポイントとして登録します。NetView OPS-MANAGEMENT-NETOP および EP-OPS-MANAGEMENT サポートは、「*SNA Management Services Reference*」に記述されている OPERATIONS_MGMT_NETOP 機能セットおよび EP_OPERATIONS_MGMT 機能セットのアーキテクチャーに基づいています。

運用管理フォーカル・ポイントが登録済みであるかどうかにかかわらず、NetView プログラムは自動的に機能の 1 つを体系的な運用管理エントリー・ポイント (EP-OPS-MANAGEMENT) として登録します。ノードがフォーカル・ポイントではなくエントリー・ポイントを持つ (このフォーカル・ポイントはリモート) 予定の場合は、DSI6INIT (CNMS1040) で運用管理の 1 次フォーカル・ポイントおよびバックアップ・フォーカル・ポイントを定義することができます。

NetView OPS-MANAGEMENT-NETOP アプリケーションが登録されると、REGISTER QUERY コマンドではこのアプリケーションを OPS_MGMT として表示します。EP-OPS-MANAGEMENT は EP_OPS として表示されます (424 ページの図 139 を参照)。

ユーザー定義のカテゴリおよびユーザー定義のアプリケーション

NetView MS アプリケーションには、ユーザー定義カテゴリの情報に関して、フォーカル・ポイントとしての機能もエントリー・ポイントとしての機能もあります。アプリケーションの 1 つが MS トランスポートをユーザー定義カテゴリにおけるフォーカル・ポイントかエントリー・ポイントのいずれかとして登録処理を済ませた場合、オペレーターは NetView FOCALPT コマンドを使用して、そのカテゴリのフォーカル・ポイントとして使われているノードを制御することができます。

アプリケーションをフォーカル・ポイントとして登録したい場合、REGISTER コマンド、マクロ、もしくはサービス・ルーチンを使用し、同時に MS アプリケーション名および FOCALPT=YES オペランドを指定すれば、オペレーターまたはコマンド・プロシージャで、そのカテゴリに関するフォーカル・ポイントとエントリー・ポイントの関係を設定することができます。例えば、フォーカル・ポイント・アプリケーションを USERDATA という名前で登録した場合、その USERDATA カテゴリ向けの FOCALPT コマンドを出すことができます。

あるノードをユーザー定義カテゴリ向けのエントリー・ポイントとして設定するには、REGISTER コマンド、マクロ、もしくはサービス・ルーチンを使用し、同時にそのカテゴリを示す FPCAT パラメーターを指定してください。そうすれば、アプリケーションはエントリー・ポイントとしての登録処理を行い、現行のフォーカル・ポイントについて、MS-CAPS から情報を受信します。各ノードのカテゴリには、エントリー・ポイント・アプリケーションを複数指定することができます。

REGISTER コマンド構文については、NetView のオンライン・ヘルプ、DSI6REGS マクロ構文については、「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」、CNMRGS サービス・ルーチンの構文については、「*IBM Tivoli NetView for z/OS* プログラミング: *PL/I* および *C*」を参照してください。

ユーザーは、複数のユーザー定義のエントリー・ポイント・アプリケーションとフォーカル・ポイント・アプリケーション、およびカテゴリを定義することができます。MS トランスポートを使用するユーザー定義のアプリケーションを登録する利点は、このようなアプリケーションが、MS-CAPS アプリケーション (SOC-MGR 機能セットを含む) が提供するサービス、および FOCALPT コマンドを使用することです。例えば、あるユーザー定義のカテゴリ用に 1 つ以上のユーザー定義エントリー・ポイント・アプリケーションがある場合、そのカテゴリの現行フォーカル・ポイントが変わるたびに、MS-CAPS はこれらのアプリケーションすべてに通知します。NetView オペレーターおよび自動化は、FOCALPT コマンドを使用して、各カテゴリでどのシステムがフォーカル・ポイントとして機能するかを制御することができます。MS-CAPS および FOCALPT 機能は、非 NetView アプリケーションを使用する通信にも適用されます。例えば、NetView MS トランスポートとともに登録されたユーザー定義のアプリケーションは、指定されたユーザー定義のカテゴリで非 NetView システム用のフォーカル・ポイントとして機能することができます。また、エントリー・ポイントとしても機能し、非 NetView フォーカル・ポイントを受け入れることもできます。

NetView プログラムに固有のフォーカル・ポイント・サポート

前のセクションで説明したように、NetView の体系的フォーカル・ポイント・サポートにより、NetView プログラムはアラート、運用管理、および LU 6.2 トランスポートを使用するユーザー定義のカテゴリ情報用のエントリー・ポイントおよびフォーカル・ポイントとして機能することができます。このサポートでは、「*SNA Management Services Reference*」アーキテクチャーに基づき、NetView システムと、NetView プログラムが稼働していない NetView システムの相互運用性が可能になります。

NetView プログラムは、NetView プログラムに固有な、アラートおよび状況カテゴリ用のフォーカル・ポイント・サポートも提供します。このフォーカル・ポイント・サポートを使用するときには、エントリー・ポイントとフォーカル・ポイントは NetView プログラムでなければなりません。このフォーカル・ポイント・サポートでは、体系的フォーカル・ポイント・サポートによって提供されるサービスを使用できないため、体系的フォーカル・ポイント・サポートに比べて提供される機能が少なくなります。例えば、NetView プログラムに固有のフォーカル・ポイントは、MS-CAPS アプリケーション (SOC-MGR サポートを含む) が提供するサービスを使用することができません。

LUC を使用したアラート転送

NetView プログラムの LUC セッション (NV-UNIQ/LUC 方式) を使用して、フォーカル・ポイントにアラートを転送することができます。フォーカル・ポイントが V3R1 より前の NetView プログラムである場合を除いて、体系的な SNA-MDS/LU 6.2 転送方式を使用して、アラートを転送することができます。OST-NNT セッションとは異なり、LUC セッションは自動的に確立されます。該当するシステム定義が

すでに設定されていれば、NetView プログラムは LUC セッションをアラートの転送に必要なものとしてオープンします。各 NetView プログラムは、複数のアラートに対し 1 つのフォーカル・ポイントを指定することができます。詳しくは、「Advanced Network and Systems Management」を参照してください。

コマンドとメッセージの転送

分散システムを管理する場合、中央システム上のオペレーターと自動化アプリケーションで、分散システムにコマンドを出さなければならないことがよくあります。コマンドを転送するには、RMTCMD コマンドを使用します。RMTCMD コマンドを使うと、NetView プログラムが通常処理するコマンドのうちフルスクリーン出力を行わないものはすべて転送することができます。分散システムの初期化やシャットダウンなどの特別なタスクのためのコマンドを出すためには、システム自動化支援プログラム (z/OS) ライセンス・プログラムを使用することができます。19 ページの『リモート操作の確立』を参照してください。

メッセージの転送は、コマンドの転送に深く関係します。メッセージの転送には、RMTCMD がセットアップする分散自動タスク、または ROUTE コマンドで使われたものと同じ OST-NNT セッションを使用し、オペレーター端末タスク (OST) と NetView 間タスク (NNT) をリンクします。

RMTCMD コマンドによる転送

RMTCMD コマンドには、転送対象のコマンドとターゲット NetView の LU 名を指定します。ターゲット・システム上の分散自動タスクとのセッションがすでに開始している場合以外、NetView プログラムは、コマンドの転送前に自動的にセッションをセットアップします。そのコマンドが生成するメッセージは、すべてコマンド発行元に戻されます。

ターゲット・システム上には、RMTCMD コマンドが使用可能なオペレーター ID がなければなりません。この ID は、RMTCMD コマンドを出すときに指定することができます。ID を指定しない場合、RMTCMD コマンドは送信元システムにおけるコマンド発行元のものと同じ ID を使用します。ID がまだアクティブになっていないときは、RMTCMD コマンドはその ID を分散自動タスクとして開始し、転送されたコマンドをその自動タスク上で処理します。これ以降は、ターゲット・システム上の分散自動タスクと関係を持つことができます。

ターゲット NetView プログラムは、分散自動タスクにより受信されたすべてのメッセージを、RMTCMD コマンドの発行元システムに送り返します。RMTCMD コマンドを出した OST は、その転送されたメッセージを受信します。転送されたメッセージには、転送されたコマンドに対するすべての応答が含まれています。また、その他の各種メッセージも、ターゲット・システムが分散自動タスクに送信した場合はすべて含まれています。

通信の柔軟性: 分散自動タスクを使用すると、柔軟な通信を行うことができます。分散システムから中央システムに例外通知のメッセージを転送し、ローカルの自動化で検出された問題をオペレーターに知らせたい場合を例にとって説明します。この場合、RMTCMD コマンドを中央システムから出す方法を使うことができますが、分散システムに転送するコマンドは、ダミー・コマンドで構いません。こうし

て分散自動タスクがセットアップされます。このあとは、中央システムに情報を転送したいときにいつでも、自動化でこの分散自動タスク宛てにメッセージを送信することができます。

RMTCMD コマンドをいったん出すと、**ENDTASK** コマンドを出すかログオフするまで、分散自動タスクはアクティブなままになります。そのため、ログオフしない自動タスクから **RMTCMD** コマンドを出せば、メッセージ転送用の持続セッションを確立することができます。

設計によっては、分散自動タスクがまだ存在しないときに分散システム上の自動化メッセージ転送を必要とする場合もあります。この場合、分散システムが **RMTCMD** コマンドを出し、**MSG** コマンドを中央システムに転送してそのメッセージを出すこともあります。この方式では、選択されたテキストを使った新規の **DSIO39I** メッセージが中央システム上に表示されます。しかし、対応する自動化内部機能要求 (**AIFR**) データで補完した既存のメッセージを転送することはできません。

RMTCMD コマンドのネスト: 分散自動タスクがないときに既存のメッセージを転送するには、ネストされた **RMTCMD** コマンドを分散システムで使用し、**RMTCMD** コマンドを中央システムから送り返します。これにより、メッセージの転送に使用可能な分散自動タスクがセットアップされます。図 143 の、**REXX** で作成した自動化プロシーチャーは **NETVDS** でコマンドを出し、**NETVDS** から **NETVCS** へのメッセージ転送を確立できます。

```
'RMTCMD LU=NETVCS,OPERID=AUTO1 EXCMD AUTO2',  
  'RMTCMD LU=NETVDS,OPERID=AUTO3 MSG AUTO3,Dummy Message'
```

図 143. **RMTCMD** の例

このコマンドを出すと、**EXCMD** コマンドが中央システムの **AUTO1** に送信され、そこで 2 番目の **RMTCMD** コマンドが **AUTO2** に経路指定されます。次に **AUTO2** が **RMTCMD** を出して、分散システム上の **AUTO3** とのセッションを確立するので、メッセージの転送を開始することができます。ここでは **AUTO2** がすでにアクティブになっているものとし、活動状態でない場合は、まず **AUTOTASK** コマンドを出して **AUTO2** を開始できます。

各 **RMTCMD** 分散自動タスクは、一度に 1 つのマスター **OST** としか接続することができませんが、マスター **OST** は、必要なだけ分散自動タスクを処理することができます。互いにネスト関係にある複数の **RMTCMD** コマンドを使用すると、中間ノードを経由して、コマンドおよびメッセージを宛先に転送することができます。この場合には、コマンドおよびメッセージを、中間ノードにある自動化テーブルを経由して送るかどうかを判別する **EXP** パラメーターを使用します。**RMTCMD** コマンドの構文については、**NetView** オンライン・ヘルプを参照してください。

OST-NNT セッションによる転送

コマンドとメッセージを転送する第 2 の方法に、**OST-NNT** セッションを使用するものがあります。これらを使用するには、中央システムから **START DOMAIN** コマンドを出して、ターゲット・システム **NNT** とのセッションを開始してください。

NNT は、現在ログオンしていない使用可能オペレーター ID であれば、いずれでもかまいません。タスクの開始方法によっては、オペレーター、分散自動タスク、通常の自動タスク、もしくは NNT で同じオペレーター ID を使用することができます。1 つの OST は、一度に複数のドメインで複数の NNT にログオンすることができますが、1 つのドメインに対して NNT は 1 つだけです。1 つの NNT を一度に複数の OST に接続することはできません。

OST-NNT セッションを確立すると、ROUTE コマンドを使用して、OST から第 2 の NetView 上の NNT にコマンドを送信することができます。NNT に受信されたメッセージはすべて、転送されたコマンドに対する応答も一緒にして、中央 NetView プログラムの OST に送り返されます。このように、NNT の動きは RMTCMD コマンドの分散自動タスクとほとんど同じなので、NNT をメッセージの転送に使用することもできます。

OST-NNT セッションの間で送信されるすべての自動化メッセージは、ターゲット・ドメインで再度作成されます。HOLD、BEEP、および DISPLAY を除くすべての自動化アクションのフラグは、このメッセージ再作成プロセスの間にリセットされます。HOLD、BEEP、および DISPLAY の各アクションは保持されるので、ドメイン間メッセージをターゲット・ドメインで自動化することができます。

重要: 拡張複数コンソール・サポート (EMCS) コンソールを使用している場合は、データの損失を防ぐため、すべてのドメイン間セッションに RMTCMD コマンドおよび LU 6.2 セッションを使用してください。そうしないと、OST と NNT の間でセッションが確立された場合、メッセージ・データ・ブロック (MDB) データ構造が付加されずにメッセージが送信されます。MDB データ構造には、メッセージに使用する強調表示 (色など) を始めとする、メッセージ関連の特殊な情報が含まれています。また、これらのデータ構造には、メッセージに関する DOM 情報が含まれています。そのため、OST-NNT セッションでは、MDB データ構造にあるこのような情報は失われます。

RMTCMD コマンドは、コマンドおよびメッセージの転送のために使用する方式です。RMTCMD コマンドでは、パフォーマンスを向上するために LU 6.2 プロトコルが使われ、オペレーターは、コマンドを転送する前に手動でセッションを開始する必要はありません。RMTCMD、START DOMAIN、および ROUTE コマンドの構文については、NetView オンライン・ヘルプを参照してください。

メッセージ転送用の中間フォーカル・ポイントの使用

中間フォーカル・ポイントを使用すると、複数の分散システムからのデータを収集してそれをフォーカル・ポイントに転送することができます。中間フォーカル・ポイントに割り当てられた分散システムは、中間フォーカル・ポイントをフォーカル・ポイントと見なし、中間システムでのセッションをセットアップしたあと、この中間システムに対し、フォーカル・ポイント・システムに対するのと同様にメッセージを送信します。

このため、中間システムにはオペレーターを置く必要がありません。これは、中間システムは、NetView 自動タスク、コマンド・リスト、および自動化テーブルを使用するので、オペレーターによる介入がなくても機能することができるためです。

中間フォーカル・ポイントを使用すると、セッションを集中させることができます。多くの分散システムは、1 つの中間システムとセッションを持ち 1 つの中間シ

システムは、フォーカル・ポイントと 1 つのセッションを確立します。したがって、フォーカル・ポイントと直接通信する必要があるシステムの数制限します。また、中間システムは分散システムのリカバリーと外部自動化を行い、フォーカル・ポイントの負荷を軽減させることもできます。

中間フォーカル・ポイントは、複数サイト環境において特に有効になります。中間フォーカル・ポイントを戦略的に配置することで、交換回線のオーバーヘッドや専用回線のコストを減らすことができます。

OST-NNT を使用したメッセージ/アラート転送

SNA-MDS/LU 6.2 および NV-UNIQ/LUC のアラート転送方法の他に、アラート転送には 3 番目の方法があります。この方式はより以前の方法であり、また OPER フィルターを使ってアラートを BNJ146I メッセージに変換します。この方式ではメッセージ転送を使用して、別の NetView に BNJ146I メッセージを転送し、GENALERT コマンドを類似のアラートに再構成します。

フルスクリーン機能と端末アクセス機能

中央設置場所から分散システムの管理を行うときには、他の NetView 機能を使用することができます。これらの機能には、フルスクリーン機能と端末アクセス機能 (TAF) があります。

モニター時の SDOMAIN コマンドの使用

集中操作では、他のドメインからデータを表示することができるので、ハードウェア・モニターとセッション・モニターが役に立ちます。例えば、オペレーターがハードウェア・モニター・コマンドの SDOMAIN を出すと、モニターしているドメインが切り替わります。転送されたアラートに特定状況についての十分な情報が入っていないときは、オペレーターが SDOMAIN コマンドを使って、分散システムからハードウェア・モニターの追加情報を得ることができます。同様にセッション・モニターも SDOMAIN コマンドを受け入れるので、オペレーターは分散システムでセッション・データを表示することができます。

ドメイン・シフトのための TAF セッションの使用

モニター対象ドメインをシフトするには、TAF セッションを使用する方法もあります。フォーカル・ポイントのオペレーターは TAF セッションを使用すると、別の NetView ドメインにフルスクリーン・モードかオペレーター制御モードでログオンすることができます。自動化ルーチンでも TAF を使用することができますが、オペレーター制御モードに限られます。自動化に TAF を使用する場合は、486 ページの表 17 を参照してください。

分散システムへの直接のログオン

上記の方法で問題を解決することができない場合は、分散システムの NetView プログラムに直接ログオンすることができます。非常に多くのシステムに同時にログオンする場合は、NetView アクセス・サービスを使用できます。

制限事項

SDOMAIN コマンドを使用するか、フルスクリーンの TAF セッションを使用するか、分散システムに直接ログオンするかのいずれかの場合、複数ドメインからの統合データは 1 画面に表示されません。また、自動化ではフルスクリーン方式が使えないという不利な点があります。そのため、継続的にモニターする場合よりも例外処理の問題を判別する場合に、フルスクリーン方式の使用が適しています。

転送方式の選択

分散システムとフォーカル・ポイントの間では、コマンド転送、メッセージ転送、アラート転送、状況転送、および LU 6.2 トランスポートを使用して情報を伝送することができます。さらに、フルスクリーン方式を使うと、問題判別に関する追加情報を得ることもできます。

以下のガイドラインは、どの情報転送方式が適しているかを判別する場合に利用してください。

- フォーカル・ポイントに MVS システムを使用して、状況転送により、ネットワークの状況情報をオペレーターに効果的に提供することができます。定義の指定とフォーカル・ポイントの選択が必要ですが、転送は自動的に行われ、グラフィック・インターフェースも得られます。
- メッセージを使って処理がしたい場合は、RMTCMD コマンドと分散自動タスクを使用します。この方法により、PIPE コマンドで非同期データを相関させ、アクティブなすべてのリモート・タスクを RMTSESS コマンド・プロセッサで追跡することができます。
- 例外通知を転送する場合は、メッセージ転送、LUC アラート転送、LU 6.2 トランスポートのいずれかを選択することができます。
- アラートを使って処理がしたい場合は、SNA-MDS/LU 6.2 アラート転送を使用します。ただし、ほとんどの場合にアラート・フォーカル・ポイントが V3 より前の NetView プログラムであるときには、NV-UNIQ/LUC アラート転送を使用してください。
- カスタマイズしたい場合には、柔軟に対応する通信オプションが用意されている LU 6.2 トランスポートを使用します。LU 6.2 トランスポートは、例外通知やその他必要なデータ伝送に使用することができます。例外通知など高度な信頼性を必要とする少量伝送には、MS トランスポートを使用してください。処理速度を重視する大量伝送には、MS トランスポートの高パフォーマンス・オプションを使用してください。

LU 6.2 トランスポートの 2 種類のバージョンのいずれを選択するかについての説明は、「*IBM Tivoli NetView for z/OS*アプリケーション・プログラマーズ・ガイド」を参照してください。

- バージョン 2 リリース 2 より前の NetView プログラムと通信する必要がある場合、使用可能なオプションが限られてきます。表 16 は、各転送メカニズムに必要な NetView プログラムのリリースを示したものです。

表 16. リリース別の NetView 転送オプション

オプション

必要なリリース

コマンド転送

ROUTE コマンドの使用	任意
RMTCMD コマンドの使用	V2R2 以降

データ転送

OST-NNT セッション (START DOMAIN) でのメッセージ転送	任意
LU 6.2 セッションでのアラートの転送	V3R1 以降
LUC セッションでのアラートの転送	任意
状況の転送	V2R1 以降
分散自動タスク (RMTCMD) によるメッセージの転送	V2R2 以降
LU 6.2 トランスポート	V2R2 以降

フォーカル・ポイントの変更

CHANGEFP コマンドの使用	任意
FOCALPT CHANGE コマンドの使用	V2R2 以降

構成の選択

集中操作環境の構成を選ぶ場合は、フォーカル・ポイントと分散システムをつなぐ物理接続と、使用する必要があるセッション・タイプの両方を考慮してください。バックアップ・フォーカル・ポイントと中間フォーカル・ポイントも設計に含めることができます。

専用回線と交換回線

フォーカル・ポイントと分散システムの接続には、チャンネルなどの専用回線、または交換回線を使用することができます。専用回線 がシステム間の永久接続であるのに対し、交換回線 はダイヤル操作によって接続するものです。交換回線では、フォーカル・ポイントと分散システム的一方だけで、両者間のセッションの開始と終了を行うことができます。両者の間に通信制御装置が入ることもあります。

ほとんどの場合に、特にフォーカル・ポイントと分散システムの場所が近いときには、専用回線を使用してください。専用回線の場合は、NetView プログラムと VTAM プログラムに必要な CPU 使用率は交換回線より少なくなります。

しかし、交換回線は専用回線よりコストをずっと低く抑えられる可能性があるもので、リモート・サイトに分散システムがある場合や、分散システムとフォーカル・ポイントとの間の通信量がほとんどないと思われる場合は、交換回線の方が適していることがよくあります。また、分散システムとバックアップ・フォーカル・ポイントとを接続する場合にも、交換回線の方が適しています。交換回線を使用すると、相互接続複数システムの自動化の利点を損なわずに、回線コストを最小にすることができます。451 ページの図 144 は、交換回線の構成を示したものです。

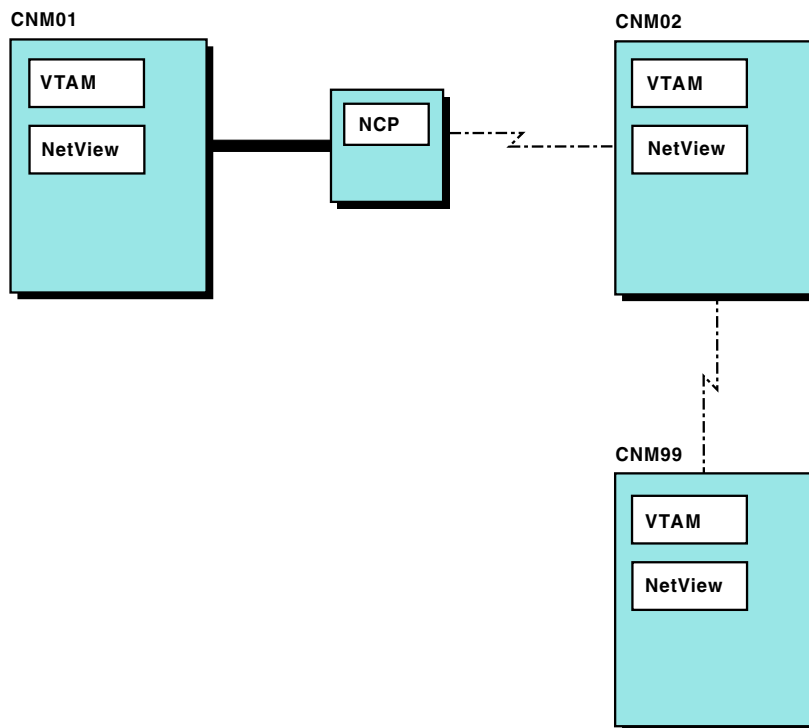


図 144. 交換回線サポート

NetView サンプルによるメッセージ転送か、または LUC アラート転送を行う場合、NetView プログラムは自動的にダイヤル接続を確立します。交換回線の構成については、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。交換回線で LUC アラート転送を実行したい場合、分散データベースの検索にも交換回線が使用されます。RMTCMD および LU 6.2 トランスポート (SNA-MDS/LU 6.2 アラート転送も含む) を使用する場合、ダイヤリングは VTAM プログラムが行うことになります。状況転送は、専用回線でなければなりません。

プログラム・オペレーター・インターフェース (POI) にアクセスするオペレーターまたは自動タスクは、次の VTAM コマンドを出して、交換リンクの活動化または非活動化を明示的に行うことができます。

- V NET,DIAL,ID=linkstation_name で、リンクが活動化されます。
- V NET,HANGUP,ID=linkstation_name で、リンクが非活動化されます。

上記の VTAM コマンドの構文に関する詳細については、z/OS Communications Server ライブラリーを参照してください。

コマンド・プロシーチャーが VTAM コマンドを出してリンクを活動化したり非活動化したりする場合、自動化では、NetView からの交換回線接続が有利です。セッションを開始する前に、NetView は交換回線接続を活動化する必要があるかどうかを判別することができます。必要であれば、セッションを要求する前に回線を活動化します。NetView コマンド・リストの DIALCDRM (CNME7023 および CNME1502) はダイヤル処理を行い、NetView コマンド・プロシーチャー中での VTAM DIAL コマンドと HANGUP コマンドの使い方の例を示します。

持続セッションと非持続セッション

NetView プログラムは、分散システムとフォーカル・ポイントとの接続を自動的に確立することができます。分散システムが、フォーカル・ポイントに送信する必要がある情報があると認識した場合、NetView プログラムはセッションを確立し、データを転送します。定義に従い、NetView プログラムは持続セッションか非持続セッションのいずれかをオープンします。

- 持続 セッションは、データが転送されたあともアクティブなままになります。
- 非持続 セッションは、NetView がデータをもう転送しない場合に、ユーザー指定時間が経過した時点で終了します。

一般的に、持続セッションは、専用回線または専用チャネルが使用され、大量のトラフィックが転送される環境で使用されます。しかし、専用回線が分散システムとフォーカル・ポイント・システムとを接続しているときには、非持続セッションを通常は使用します。

持続セッションと非持続セッションを選択するために必要な定義については、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。LUC セッションを持続にする必要があるか非持続にする必要があるかは、NetView プログラムが通信する各ドメインごとに、それぞれ別の選択をすることができます。LUC セッションは、アラート転送、状況転送、および分散データベース検索に使用できる他、SDOMAIN コマンドによるセッション・モニターまたはハードウェア・モニターでのドメイン間表示にも使用することができます。NetView サンプルを使ってメッセージ転送をセットアップする場合にも、持続セッションにするか非持続セッションにするかを選択することができます。

しかし、RMTCMD コマンドおよび LU 6.2 トランスポート (SNA-MDS/LU 6.2 アラート転送を含む) が使用するセッションを持続にするか非持続にするかを、NetView プログラムは制御することができません。この決定を行うには、VTAM を使う必要があります。

分散システムとそのフォーカル・ポイントの間の回線およびセッションに適用される規則は、分散システムとそのバックアップ・フォーカル・ポイントの間の回線およびセッションにも適用されます。したがって、交換回線でも専用回線でも、さらに持続セッションでも非持続セッションでも使用することができます。

しかし、1 次フォーカル・ポイントへの転送が迅速に再開されようようにしたい場合は、NetView サンプルを使用するメッセージ転送には非持続セッションを使用してください。持続セッションでは、明示的にユーザーがセッションを終了しない限り、1 次フォーカル・ポイントが使用可能になったあとでも、バックアップ・フォーカル・ポイントにデータを送信し続けるからです。

複数のフォーカル・ポイントの使用

分散システムからフォーカル・ポイントに情報を転送する場合、通常はすべてのデータ・タイプに 1 つのフォーカル・ポイントを選択します。このように設計すると、そのフォーカル・ポイントのオペレーターまたは自動化アプリケーションは、特定の分散システムに関連するデータをすべて集めることができるようになります。しかし、複数のフォーカル・ポイントを使うことも可能です。

分散システムは、転送対象データのカテゴリごとに、別々のフォーカル・ポイントを持つことができます。データ・カテゴリには、メッセージ、アラート、状況情報、および運用管理データがあります。また、システムは、MS アプリケーションの個々のユーザー定義カテゴリにつき、1 つのフォーカル・ポイントを設定することもできます。

上記のカテゴリ以外の方法でデータを分割したい場合は、RMTCMD メッセージ転送や LU 6.2 トランスポートなどの、カスタマイズした設計を実現できるメカニズムの 1 つを使います。例えば、優先順位の低い通知を、あるフォーカル・ポイントに送信し、優先順位の高い通知を別のフォーカル・ポイントに送信する場合、各メッセージ受信先と RMTCMD セッションを確立する独自のアプリケーションを作成し、どの受信先が各メッセージを受信するかを決めます。同様に、LU 6.2 トランスポートを使用して、アプリケーションおよび選択したシステムに情報を指示します。

フォーカル・ポイントの変更、除去、およびリスト

FOCALPT CHANGE および FOCALPT ACQUIRE コマンドを使用すると、体系的フォーカル・ポイントおよび NetView 固有のフォーカル・ポイントの両方のフォーカル・ポイントを変更することができます。アラート (SNA-MDS/LU 6.2 および NV-UNIQ/LUC アラート転送の両方のアラート)、状況、運用管理、または MS アプリケーションのユーザー定義カテゴリ、以上のフォーカル・ポイントを変更する場合は、FOCALPT CHANGE または FOCALPT ACQUIRE を使用します。

CHANGEFP コマンドまたは FOCALPT CHANGE コマンドは、新規のフォーカル・ポイント・システムから出し、ターゲット・システムを指定します。コマンド発行元のドメインは、そのターゲット・システムで他の変更コマンドを出すか NetView プログラムを停止および再始動するまで、そのターゲット・システムの 1 次フォーカル・ポイントになります。メッセージ、アラート、運用管理データ、およびユーザー定義 MS カテゴリの場合は、新規のバックアップ・フォーカル・ポイントを指定することもできます。FOCALPT ACQUIRE コマンドは、エントリー・ポイントのシステムから新規のフォーカル・ポイント・システムを指定するために使用します。

バージョン 3 以降のエントリー・ポイント NetView は、CNMSTYLE メンバーに指定されている ALERTFWD ステートメントの内容に応じて (ALERTFWD については「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください)、SNA-MDS/LU 6.2 アラート転送または NV-UNIQ/LUC アラート転送のいずれかでアラートを転送します。したがって、FOCALPT CHANGE および ACQUIRE コマンドは、これらのいずれかのメカニズムにより、バージョン 3 以降のエントリー・ポイント NetView プログラムからそのフォーカル・ポイントへのアラートの転送を確立します。NV-UNIQ/LUC アラート転送が使用されていて、フォーカル・ポイントおよびエントリー・ポイントが両方とも NetView の場合には、フォーカル・ポイントの NetView DSICRTR タスクは、分散システムの通信相手に REQUEST メッセージを送信します。分散システムと変更前のフォーカル・ポイントの間に確立されたアラート転送元の LUC セッションは、そのセッションでキューイングされたすべてのアラートが送信されたあとに終了します。それ以外のアラートは、新しいフォーカル・ポイントに送信されます。

SNA-MDS/LU 6.2 アラート転送が使用されるときには、MS-CAPS アプリケーションは、エントリー・ポイント - フォーカル・ポイント関係を確立します。いったん関係が確立されると、ALERT-NETOP アプリケーションは受信したすべてのアラートを MS トランスポートを介してフォーカル・ポイント (ALERT-NETOP アプリケーションでもある) に転送します。

例えば、SNA-MDS/LU 6.2 または NV-UNIQ/LUC アラート転送の場合、次のコマンドを出すと、CNMDS アラート・フォーカル・ポイントを CNMFP に変更し、バックアップ・フォーカル・ポイントを CNMBA に変更することができます。

```
FOCALPT CHANGE TARGET=CNMDS BACKUP=CNMBA FPCAT=ALERT
```

FPCAT=OPS_MGMT、FPCAT=STATUS、もしくは FPCAT=*user-defined* のオペランドを代わりに指定すれば、運用管理フォーカル・ポイント、状況フォーカル・ポイント、ユーザー定義 MS カテゴリ用のフォーカル・ポイントのいずれかを変更することができます。運用管理カテゴリおよびユーザー定義カテゴリの要求は、NetView プログラム以外のターゲット、あるいは CNMSTYLE メンバーに「ALERTFWD SNA-MDS」がコーディングされているバージョン 3 以降の NetView ターゲットに送信されるアラート・カテゴリの要求と同じように、MS-CAPS アプリケーションで処理されます。カテゴリの詳細については、427 ページの『MS-CAPS アプリケーション』を参照してください。

状況フォーカル・ポイントの変更は、新規のフォーカル・ポイントが初期状況情報の収集から始めなければならないので、処理に時間がかかります。そのため、状況フォーカル・ポイントを変更するのは、1 次フォーカル・ポイントが長時間使用されない場合のみにしてください。BACKUP オペランドは、状況フォーカル・ポイントに使用することはできません。

バックアップ・フォーカル・ポイントの変更および追加を行うには、FOCALPT ACQUIRE コマンドを使用してください。このコマンドは次の場合に使用することができます。

- バックアップ・フォーカル・ポイント名の変更
- データ・カテゴリ用の新規バックアップ・リストの定義
- 既存バックアップ・リストへのバックアップ・フォーカル・ポイントの追加
- フォーカル・ポイントのバックアップ・リストからの除去

FOCALPT ACQUIRE は、フォーカル・ポイントを DSI6INIT に定義されたフォーカル・ポイントに復元する場合に使用することもできます。

システムのフォーカル・ポイントをリストする場合には、FOCALPT QUERY コマンドまたは LIST FOCPT コマンドを使用してください。ある情報カテゴリのフォーカル・ポイントへの転送を停止したいときには、その情報が状況情報またはメッセージである場合以外は、分散システム上で FOCALPT DROP コマンドを出します。FOCALPT DROP コマンドは、バックアップ・リストから 1 つ以上のフォーカル・ポイントを除去する場合にも使用します。ENDTASK コマンドを出すと、分散自動タスクが非活動化されるので、メッセージ転送を終了することができます。OST-NNT セッションでメッセージの転送を停止するには、LOGOFF コマンドを NNT に送信します。

第 7 部 NetView におけるその他の自動操作

第 27 章 その他のシステム、装置、およびネットワークの自動化

ここまでの章では、NetView プログラムを実行できるプロセッサの自動操作について説明してきました。そして、SNA プロトコルに従い VTAM プログラムを通して NetView プログラムに報告する装置およびネットワークの自動化について記載してきました。この章では、他の多くの IBM ならびに IBM 以外のシステム、装置、およびネットワークの自動管理に対応する NetView の自動化機能について説明します。

NetView の自動操作機能が有効か否かは、それらの非 NetView システムや非 SNA 装置の機能によります。このようなシステムまたは装置は、直接または間接に、問題報告書やその他の情報を自動化可能形式 (メッセージまたは MSU) で NetView プログラムに送信し、NetView プログラムからコマンドを受信することができなければなりません。

自動化を通して SNA 以外のネットワークを管理する方法については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

多くの場合、直接自動化することのできないプロダクトも、適切なインターフェース・プロダクトを使用すれば自動化することができます。本章では、サービス・ポイントをインプリメントして NetView 自動化の範囲を拡張できるようにする、以下のようなインターフェース・プロダクトの例をいくつか示します。

- イベント自動化サービス
- サービス・ポイント・アプリケーション (SPA) ルーターおよびリモート操作サービス (ROPS)
- NCP フレーム・リレー交換装置

インターフェース製品の他の例については、25 ページの『第 2 章 自動操作プロダクトの概要』を参照してください。

イベント自動化サービス

イベント自動化サービス は、Event Integration Facility (EIF) イベントを処理する NetView の z/OS 管理環境、マネージャーおよびエージェントと、SNMP マネージャーおよびエージェント間でのイベント・データのゲートウェイとして機能します。このゲートウェイ機能により、任意の管理プラットフォームからすべてのネットワーク・イベントを管理できます。ハードウェア・モニターが受信するアラートは、EIF イベント または SNMP トラップに変換して、対応するイベント・マネージャーに転送できます。NetView プログラムが受信するメッセージは、EIF イベントに変換して、指定されたイベント・サーバーに転送できます。逆に、SNMP トラップまたは EIF イベントはアラートに変換して、ハードウェア・モニターに転送できます。

アラートについては、元のアラート・データの一部だけがイベント自動化サービスへ転送されます。NetView プログラムは、アラートに補足情報を追加して、イベン

ト自動化サービスへ転送します。結合された情報は、EIF イベントまたは SNMP トラップを作成するために、イベント自動化サービスのアラート・アダプター・サービス、確認済みアラート・アダプター・サービス、または ALERT-TO-TRAP サービスが使用します。NetView プログラムから転送される情報をカスタマイズすることによって、出力イベントまたは出力とラップの内容をカスタマイズすることができます。詳しくは、『アラートの転送』を参照してください。

メッセージも同様に処理されます。メッセージ全体が、NetView プログラムにより作成された補足情報と結合され、イベント自動化サービスへ転送されます。結合された情報は、EIF イベントを作成するために、イベント自動化サービスのメッセージ・アダプター・サービスまたは確認済みアラート・アダプター・サービスが使用します。NetView プログラムから転送される情報をカスタマイズすることによって、EIF イベントの内容をカスタマイズすることができます。詳しくは『459 ページの『メッセージの転送』』を参照してください。

アラートの転送

Event Integration Facility (EIF) イベントまたは SNMP トラップの作成方法を変更せずに、ハードウェア・モニター・アラートを転送する場合は、ハードウェア・モニター記録フィルターを使用して、NetView が転送しなければならないアラートを選択します。TECROUTE フィルターは、指定されたイベント・サーバーに転送するアラートを選択し、また TRAPROUT フィルターは、SNMP マネージャーに転送するアラートを選択します。アラートを転送するためには、TECROUTE または TRAPROUT フィルターを PASS に設定しなければなりません。しかし、アラートは、ESREC フィルターおよび AREC フィルターを通過しなければ、TECROUTE フィルター または TRAPROUT フィルターで処理することはできません。SRFILTER コマンドを使用すると、ハードウェア・モニターからフィルターの設定値を指定することができます。また、SRF アクションを使用して、自動化テーブルからのフィルターの設定値を指定することもできます。

アラートが転送されるときに EIF イベント または SNMP トラップを作成する方法をカスタマイズするには、次のように行います。

- ハードウェア・モニター SRFILTER コマンド、または自動化テーブルの SRF アクションのいずれかを使用して、TECROUTE フィルター または TRAPROUT フィルターを PASS に設定する。
- カスタマイズを実行するコマンドを作成する。コマンドの作成方法については、NetView サンプルの CNMEALUS および CNMSIHSA を参照してください。
- NetView 自動化テーブルで、自動化テーブル IF-THEN ステートメントの *cmdstring* パラメーターに、ユーザーが作成したコマンドの名前を指定する。キーワード TECROUTE を接頭部として、*cmdstring* の先頭に追加する。

注:

1. この種の接頭部付きコマンドは、所定のアラートに対して 1 つだけがサポートされます。このコマンドですべての TECROUTE アクションおよび TRAPROUT アクションを処理しなければなりません。このコマンドが実行されるのは、TECROUTE フィルターと TRAPROUT フィルターの両方を通過する場合でも一度のみです。

2. 確認済みアラート・アダプター・サービスを使用している場合は、*cmdstring* パラメーターを *TECROUTE* で始めて、その後に *PIPE* コマンドを記述するか、*TECRTCFM* キーワードを使用して最終的にパイプ *PPI* ステージを呼び出すコマンドの名前を記述する必要があります。
- アラート・アダプター・サービス・クラス定義ステートメント・ファイル (サンプルは、*IHSAACDS*)、確認済みアラート・アダプター・サービス・クラス定義ステートメント・ファイル (サンプルは、*IHSABCDs*)、または *ALERT-TO-TRAP* サービス・クラス定義ステートメント・ファイル (*IHSALCDS*) をカスタマイズする。
 - イベント・サーバーに適用された *BAROC* ファイルをカスタマイズする。詳細については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。この手順は、確認済みアラート・アダプター・サービスの場合は必要ありません。

メッセージの転送

EIF イベントの作成方法を変更せずに、NetView プログラムからのメッセージを転送するには、NetView 自動化テーブルの IF-THEN ステートメントの *cmdstring* に以下の情報を指定します。

```
'PIPE SAFE * | PPI TECROUTE PPI_receiver_ID'
```

このコマンドにおいて、*PPI_receiver_ID* は、イベント自動化サービスに関連付けられた *PPI* レシーバーの名前です。デフォルト値は *IHSATEC* です。デフォルト値を使用する場合でも、*PPI_receiver_ID* に値を指定します。*PPI* ステージが失敗しても、この例ではメッセージは出力されないことにご注意ください。NetView サンプル *CNMEMSUS* には、*PPI* ステージの 2 次出力を使用して、エラー・メッセージを出力する例を示してあります。

注: 確認済みメッセージ・アダプターを使用するには、*cmdstring* 内で以下の情報を使用してください。

```
'PIPE SAFE * | PPI TECRTCFM PPI_receiver_ID'
```

メッセージが指定されたイベント・サーバーへ転送されときの EIF イベントの作成方法あるいはエラー・メッセージの処理方法のカスタマイズは、次のように行います。

- カスタマイズを実行するコマンドを作成する。コマンドの作成方法については、NetView サンプルの *CNMEMSUS* および *CNMSIHSA* を参照してください。
- NetView 自動化テーブルで、自動化テーブル IF-THEN ステートメントの *cmdstring* パラメーターに、ユーザーが作成したコマンドの名前を指定する。
- メッセージ・アダプター・サービス・メッセージ形式ファイル (サンプルは、*IHSAMFMT*) または確認済みアラート・アダプター・サービス・メッセージ形式ファイル (サンプルは、*IHSANFMT*) をカスタマイズする。
- イベント・サーバーに適用された *BAROC* ファイルをカスタマイズする。詳しくは、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」を参照してください。この手順は、確認済みメッセージ・アダプター・サービスの場合は必要ありません。

NCP フレーム・リレー交換装置のサポート

NetView は、NCP フレーム・リレー交換装置用のプロトコル情報を含む、体系化された主ベクトルおよびサブフィールドをサポートしています。

- X'52' サブベクトル内の X'0E' サブフィールドには、フレーム・リレー状況情報が入っています。
- X'52' サブベクトル内の X'0F' サブフィールドには、フレーム・リレー構成情報が入っています。

X'0E' サブフィールドと X'0F' サブフィールドは、X'1332' 主ベクトルにあります。
X' 0E' サブフィールドは、X'0000' 主ベクトルにあります。

フレーム・リレー・ベクトルはハードウェア・モニターには表示されませんが、自動化テーブルに渡されたあと、MSU として処理されます。自動化テーブルのステートメントは、サンプル自動化テーブル DSITBL01 の中に入っていて、コメント化されています。このステートメントを使用すると、コマンド・プロセッサを条件付きで実行し、フレーム・リレー情報を処理することができます。NetView プログラムには、このタイプのサンプル・コマンド・プロセッサは組み込まれていません。

ベクトルとサブフィールドの形式については、NCP ライブラリーを参照してください。

第 28 章 リソース・オブジェクト・データ・マネージャーを使用した自動操作

リソース・オブジェクト・データ・マネージャー (RODM) は、ストレージ内データ・キャッシュで、構成データとリソース状況情報を記憶するものです。RODM は、ネットワークとシステムの両方の自動化に使用することができます。

RODM を使用する自動操作プロジェクトの企画は、RODM の用語と概念をよく理解してから行ってください。NetView プログラムが、RODM およびそのデータ・モデルの記述に使用するオブジェクト指向の用語については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

本章では、RODM による自動化について基本的な概念を説明し、詳しい自動化のシナリオを提供します。基本的な概念は、次のとおりです。

- 1 つ以上の RODM データ・キャッシュの管理
- RODM メソッドからの NetView コマンドの実行
- RODM メソッドから出されたコマンドの検証
- NetView 自動化テーブル、NetView 高水準言語、およびアセンブリ言語プログラムからの RODM へのアクセス

NetView プログラムに用意されている専用 NetView タスクおよび一連のサービスで、RODM を使用することができます。これらの NetView サービスを使うと、基本 RODM API を使用するよりも簡単に、RODM による自動化を行うことができます。この一連の NetView サービスは、**RODM 自動化プラットフォーム** と呼ばれます。

「自動化進行」インディケーターは、NetView により、自動化を実行中のリソース用に RODM 内で保守されます。これにより、NetView グラフィック監視機能を使用するオペレーターは、問題の解決を試行する前に、リソースに対する自動化が完了するまで待つことができます。

RODMView ツールを使うと、RODM 内のデータとオブジェクトを表示し操作することができます。RODMView には、RODM のアプリケーション・プログラミング・インターフェースも含まれています。RODMView についての詳細は、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

複数の RODM データ・キャッシュの管理

RODM は、NetView アプリケーション・アドレス・スペースや NetView サブシステム・アドレス・スペースとは、別のアドレス・スペースに常駐しています。複数の RODM が、単一のシステムに常駐することができます。RODM はそれぞれ、それ自身のアドレス・スペースに常駐します。

RODM データ・キャッシュの管理に含まれる機能には、RODM データ・キャッシュに対する接続、切断、チェックポイント処理が含まれます。**チェックポイント**とは、RODM に含まれる現在のデータをすべて DASD に保管する要求です。RODM データ・キャッシュを管理するには、独自にアプリケーションを作成する方法と、NetView DSIQTSK タスクを使用して NetView アドレス・スペースからその RODM データ・キャッシュを管理する方法があります。

DSIQTSK タスクを使用する RODM の管理

DSIQTSK は、RODM アドレス・スペースとの通信と、指定された RODM データ・キャッシュの管理とに関して、専用に使われるタスクです。NetView アドレス・スペースから管理する各 RODM は、DSIQTSK に定義しておく必要があります。DSIQTSK は、NetView のオプション・タスク (OPT) です。DSIQTSK は CNMSTYLE メンバー内で定義され、START TASK コマンドで開始されます。

DSIQTSK は、RODM データ・キャッシュを管理する他に、以下のことを行います。

- RODM によって送信されたコマンドの受信
- NetView 自動タスク (自動タスクのワークロードをもとにして DSIQTSK に定義されている) への、コマンドのディスパッチ

単一のホストに複数の NetView プログラムがある場合、個々の NetView プログラムにはそれぞれ DSIQTSK タスクがあります。これらの DSIQTSK タスクで使用される受信先名は、それぞれ固有のものでなければなりません。DSIQTSK タスクは、RODM から送られたコマンドを受信先として自動的に登録します。

RODM から送られたコマンドを DSIQTSK で受信したい場合は、DSIQTSKI 初期設定メンバーを使用して、受信先名、ならびにコマンドのディスパッチ先の自動タスクの名前を定義します。

DSIQTSKI 初期設定メンバーを使用する RODM の定義

DSIPARM の DSIQTSKI 初期設定メンバーには、DSIQTSK による RODM データ・キャッシュの管理方法に関する詳細を定義するキーワードが含まれています。キーワードによって定義されているものには、RODM 属性、自動タスク名、およびコマンドを DSIQTSK に送信するときに RODM が使用するコマンド受信先名があります。

各キーワードの定義についてはここで簡単に説明しますが、詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

CMDRCVR

NetView アドレス・スペースにコマンドを送信するために RODM が使用する、プログラム間インターフェース・キューの名前。DSIQTSK タスクが、このキューの専用受信先になります。RODM から NetView プログラムに送信されるコマンドも、EKGSPPI という RODM メソッドを使用してプログラム間インターフェースに入られます。「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」で、NetView プログラム提供のメソッドの説明を参照してください。

REP RODM の名前。管理対象の個々の RODM ごとに、それぞれ 1 つの REP キーワードを使用してください。

REP キーワードには、以下のパラメーターがあります。

CONN DSIQTSK が活動化されているときに、RODM を NetView アドレス・スペースに接続すべきかどうかを指定します。このパラメーターは、CONN=Y もしくは CONN=N のように入力します。

AO この RODM を、ORCONV コマンド、CNMQAPI 高水準言語サービス・ルーチン、および DSINOR アセンブリー言語マクロで使うデフォルトの RODM にするかどうかを指定します。

PASS この RODM 用のパスワードまたはパスワード・フレーズを指定します。このパスワードまたはパスワード・フレーズは、DSIQTSK がこの RODM との接続を試行するときに使われます。

CMD RODM に接続するときに DSIQTSK が使用するコマンドを指定します。

T ORCONV、CNMQAPI、もしくは DSINOR を使って出された要求を、RODM がチェックポイント処理中であるために処理することができない場合に、それらの要求が待機する総時間数を指定します。この待ち時間が満了になると、その要求の処理は失敗し、そのチェックポイントが処理中であることを示す戻りコードが戻されます。

ID DSIQTSK が DSIQTSK そのものを RODM に識別させるために使用するアプリケーション ID を示します。

TASK NetView 自動タスクの名前。

DSIQTSK は、RODM からコマンドを受信すると、そのコマンドを NetView 自動タスクにディスパッチします。DSIQTSK に使用できるようにしたい個々の自動タスクごとに TASK ステートメントを使用してください。

ORCNTL コマンドを使用する RODM の管理

DSIQTSKI 初期設定メンバーに RODM データ・キャッシュを定義して、DSIQTSK タスクを活動化した後で、NetView ORCNTL コマンドを使用して、以下のことを行います。

- 指定した RODM との接続
- 指定した RODM からの切断
- 指定した RODM の接続パスワードまたはパスワード・フレーズの変更
- NetView ORCONV コマンド、CNMQAPI 高水準言語サービス・ルーチン、および DSINOR アセンブリー言語マクロのデフォルト RODM の変更
- 指定された RODM のチェックポイント処理の初期化
- DSIQTSK の制御下にある自動タスクの状況のリスト
- DSIQTSK に管理されるすべての RODM データ・キャッシュの状況のリスト

ORCNTL コマンドの構文と使用法については、NetView オンライン・ヘルプを参照してください。

RODM メソッドからのコマンドの発行

RODM メソッドからコマンドを出すには、EKGSPPI メソッドを使用します。EKGSPPI メソッドは、プログラム間インターフェースを使用して、NetView DSIQTSK タスクにコマンドを送信します。これらのコマンドには、NetView 自動タスクから実行できる任意のコマンドが含まれます。例えば、リソースに障害が起こったときに、メソッドを起動し、VTAM VARY コマンドを使用して自動的にそのリソースの活動化を試行するようにしたい場合を例に考えます。VARY コマンドは、RODM アドレス・スペースから実行することができないので、NetView アドレス・スペースに送られます。NetView アドレス・スペースにある DSIQTSK タスクは、実行するためにそのコマンドを NetView 自動タスクにディスパッチします。

詳細については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」で、NetView プログラム提供のメソッドの説明を参照してください。

RODM を定義するときには、NetView アドレス・スペースにコマンドを送信するために RODM が使用するプログラム間インターフェース・キューの名前を入れてください。DSIQTSK タスクが、このキューの専用受信先になります。

プログラム間インターフェースを通してコマンドを送信できるのは、DSIQTSK に定義された RODM データ・キャッシュの場合に限られます。

RODM メソッドから出されたコマンドの検証

コマンドを DSIQTSK に送信する EKGSPPI メソッドを起動する、RODM メソッドを作成したら、実際にはコマンドを実行しないで、メソッドをテストします。コマンドを自動タスクにディスパッチする代わりに、DSIQTSK はコマンドをメッセージとして表示することができ、オペレーターが、実際に実行するために、コマンドを編集したり、破棄したり、または発行したりすることを可能にします。このプロセスは、**補助モード** でのコマンドの発行と呼ばれます。

以下は、補助モードを使用するのに役立ちます。

- EKGSPPI メソッドの ASSIST パラメーター

使用しているメソッドにおいて、ASSIST パラメーターを EKGSPPI に渡すと、EKGSPPI がコマンドを補助モードで出すように指定することができます。コマンドが補助モードで出される場合、DSIQTSK は、そのコマンドを実行しないでメッセージ (メッセージ DWO670I) に変換します。

EKGSPPI 呼び出しの詳細については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

- SAVECMD コマンド

自動化テーブルで SAVECMD コマンドを使用して、メッセージ DWO670I をオペレーターに経路指定することができます。SAVECMD は ASSISCMD コマンドのコマンド情報およびテキストの情報を保存します。

ASSISCMD は、オペレーターが呼び出すコマンドです。ASSISCMD が、NetView VIEW 機能を使用し、SAVECMD によって保管されたコマンドとテキストのフル

スクリーン・パネルを作成すると、オペレーターはそれらのコマンドを承認したり、変更したり、破棄したりすることができます。

SAVECMD コマンド・リストが実行されるのは、DSIQTSK タスクで自動化メッセージ DWO670I が受信された時点です。SAVECMD コマンドの正確な形式については、オンライン・コマンド・ヘルプを使用してください。

自動化テーブルで SAVECMD コマンド・リストを使用する方法を、図 145 に示します。

```
IF MSGID='DWO670I' THEN  
EXEC(CMD('SAVECMD') ROUTE(ONE NETOP1));
```

図 145. 自動化テーブルでの SAVECMD コマンド・リストの使用

• ASSISCMD コマンド

コマンドがオペレーターに経路指定されたあと、オペレーターにはメッセージ CNM436I が表示されます。それ以後に同一のオペレーターに経路指定されたコマンドは、そのオペレーター用のキューに入れられ、メッセージ CNM436I は表示されません。オペレーターがすべての保存済みコマンドを処理し終わり、キューが空になると、そのオペレーターに経路指定される次のコマンドについて、メッセージ CNM436I が表示されます。

メッセージ CNM436I は、そのオペレーターが ASSISCMD を入力しなければならないことを示します。そのオペレーターが ASSISCMD を使用すれば、SAVECMD によって保存されたコマンドを操作することができます。ASSISCMD を使用すると、オペレーターは以下のことを実行することができます。

- コマンドの削除
- コマンドの編集と再発行
- コマンドをそのまま実行

ASSISCMD コマンドの構文については、NetView オンライン・ヘルプを参照してください。

NetView プログラムから RODM へのアクセス

DSIQTSK タスクの管理する RODM データ・キャッシュにアクセスするには、以下の 3 つのメソッドを使用します。

ORCONV

NetView コマンド

CNMQAPI

高水準言語サービス・ルーチン

DSINOR

アセンブリー言語マクロ

CNMQAPI と DSINOR は、コマンド・プロセッサとインストール・システム出口から出すことができます。ORCONV コマンドおよびアプリケーション・プログラミング・インターフェースを使用することができるのは、DSIQTSK によって管理

される RODM データ・キャッシュの場合に限られます。他の方法で RODM データ・キャッシュを管理している場合には、EKGUAPI アプリケーション・プログラミング・インターフェースを使用して、RODM にアクセスします。

ORCONV コマンド

ORCONV は、フィールドを変更し、RODM にあるメソッドを以下のソースから呼び出すコマンドです。

- NetView 自動化テーブル
- コマンド・リスト
- コマンド機能
- REXX、PL/I、もしくは C で作成されたプロシージャ

ORCONV コマンドにおけるパラメーターの指定とその形式については、オンライン・コマンド・ヘルプを参照してください。

高水準言語およびアセンブリ言語プログラムから RODM へのアクセス

高水準言語 (HLL) サービス・ルーチンの CNMQAPI、およびアセンブリ言語マクロの DSINOR は、それぞれ HLL プログラムおよびアセンブラー・プログラムから呼び出されます。どちらも、固有の RODM アプリケーション・プログラミング・インターフェースの EKGUAPI を使用しています。CNMQAPI と DSINOR は、RODM データ・キャッシュに定義されている DSIQTSK でのみ、使用することができます。

CNMQAPI および DSINOR の詳細については、「*IBM Tivoli NetView for z/OS* プログラミング: PL/I および C」および「*IBM Tivoli NetView for z/OS* プログラミング: アセンブラー」を参照してください。

RODM 自動化シナリオ

この自動化シナリオでは、本章で説明している自動化の概念と機能を実作業例に組み込んでいます。自動化のプラットフォームを使用して、次のことを行う方法がわかります。

- RODM の管理
- RODM のデータの操作
- 障害の発生しているリソースのリカバリーの自動化
- 自動タスクへの作業のディスパッチ
- RODM メソッドから出されたコマンドの検証

このシナリオは、以下の 5 つの部分で構成しています。

- シナリオで発生するイベントの概略
- このシナリオで使用する種々のエンティティ (RODM 名、RODM クラス、リソース、およびオペレーター ID など) の説明
- このシナリオをセットアップする方法
- このシナリオを実行する方法

- このシナリオで使用される変更メソッド (EKGCPPI) のキー・セクションからの引用と解説

シナリオのイベント

このシナリオで発生するイベントには、主として以下のものがあります。

1. DSIQTSK タスクが、自動的に 2 つの RODM データ・キャッシュに接続します。この接続は、初期設定メンバー DSIQTSKI の RODM 定義に基づいて行われます。
2. ORCNTL コマンドを出して、デフォルト RODM を変更します。
3. ネットワーク・リソースの A01A704 に障害が起こります。
4. このリソースに障害が起こったことを示す IST105I が、VTAM によって出されます。
5. IST105I メッセージが、自動化テーブル内でトラップされます。
6. IST105I メッセージの結果、自動化テーブルが ORCONV コマンドを出し、RODM 内のリソース状況を変更します。
7. 状況変更の結果として呼び出された RODM 変更メソッドが AOLEVEL というフィールドをチェックし、このリソースを対象とする自動化コマンドを出すかどうかを決定します。このシナリオの AOLEVEL フィールドは、自動化コマンドが補助モードで出されることを示します。
8. RODM メソッドは DSIQTSK に VTAM コマンドを送信し、リソースを活動化します。このコマンドは、EKGSPPI メソッドを使って補助モードで出されます。EKGSPPI の詳細については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」で、NetView プログラム提供のメソッドの説明を参照してください。
9. コマンドが補助モードで出されたため、DSIQTSK は VTAM コマンドを自動タスクにディスパッチせず、保存します。オペレーターは ASSISCMD コマンドを出して、表示されたコマンドの編集と再発行、破棄、または表示された通りにコマンドを出すことができます。

シナリオのエンティティー

このシナリオでは、以下のエンティティーを参照します。

エンティティー

説明

RODM1

初期設定メンバーの DSIQTSKI に、デフォルト RODM として定義されます。デフォルト RODM は、CNMQAPI、DSINOR、および ORCONV が活動する RODM です。

RODM2

初期設定メンバーの DSIQTSKI に、追加 RODM として定義されます。

TERMINAL

RODM2 に含まれるクラスの名前。

A01A704

RODM2 におけるクラス TERMINAL のオブジェクトの ID。このオブジェクトは、ネットワーク内でローカル接続の論理装置 (LU) です。

STATUS

オブジェクト A01A704 のフィールド名。このフィールドには、リソースの状況 (UP または DOWN) が入ります。RODM 変更メソッドの EKGCPPI は、このフィールドに対応しており、状況が DOWN に変更されると A01A704 を再活動化しようとします。

AOLEVEL

オブジェクト A01A704 のフィールド名。このフィールドには、1、2、3 のいずれかの数値が入ります。このシナリオでは、AOLEVEL フィールドを使用して、リソースに関連する自動化コマンドを出すかどうかを決定します。また、補助モードを使用するかどうかも決定します。

- 1 RODM メソッドは VTAM コマンドを出し、リソースを活動化します。リソースがダウンしていることを NetView プログラムに通知した IST105I メッセージは、いかなる特殊処理も受けません。
- 2 リソースがダウンしたことを NetView プログラムに通知した IST105I メッセージは、ORCONV コマンドの指定に従ってオペレーターに経路指定され、RODM はリソースの活動化を試行しません。
- 3 RODM メソッドは補助モードで VTAM コマンドを出してリソースを活動化します。また、そのリソースがダウンしたことを NetView プログラムに通知した IST105I メッセージは、いかなる特殊処理も受け取りません。

CNM01

DSIQTSK にコマンドを送信するために RODM が使用するコマンド受信先キューの名前。

AUTO1, AUTO2, AUTO3

DSIQTSK がコマンドのディスパッチ先とする 3 つの自動タスク。

シナリオのセットアップ

シナリオをセットアップするには、次のようにしてください。

1. このシナリオで『障害の起こっているリソース』を識別します。

このシナリオでは、障害の起こっているリソースとして、A01A704 を使用します。それにはまず、このシナリオにおいて、A01A704 を VTAM に定義するか、ネットワーク内の別のリソースを使用することを決めて、リソース名を A01A704 に置き換える必要があります。

2. NETOP1 というオペレーター ID がいない場合、NETOP1 を定義するか、このシナリオの全体にわたってオペレーター ID の 1 つを置き換える必要があります。このオペレーター ID を、ORCONV コマンドの MSGPARMS パラメーターに必ず入れるようにしてください。
3. AUTOTASK コマンドを使って、3 つの自動タスクの作成と開始を行います。このシナリオで使用する自動タスク名は、AUTO1、AUTO2、および AUTO3 です。AUTO1、AUTO2、および AUTO3 という名の自動化操作プログラムがない場合は、それらを定義するか、このシナリオに基づき、自動タスク名の 3 つを置き換えます。

4. DSITBL01 という名の自動化テーブルを作成するか、現存する自動化テーブルを DSITBL01 に名前変更するか、このシナリオを通して、自動化テーブルの名前を置き換えることもできます。
5. 使用している RODM データ・キャッシュを DSIQTSK に定義します。

DSIQTSKI 初期設定メンバーは、管理したい RODM データ・キャッシュに関する管理についてのさまざまな詳細を定義することができます。DSIQTSKI メンバーは、DSIPARM に入っています。この DSIQTSKI メンバーのサンプルでは、以下のものが定義されています。

CNM01

コマンド受信先キュー名

RODM1 および RODM2

管理対象になる 2 つの RODM データ・キャッシュ。RODM1 がデフォルト値です。

AUTO1、AUTO2、および AUTO3

DSIQTSK が作業をディスパッチするときの宛先にするのできる 3 つの自動タスク

図 146 には、DSIQTSKI 初期化メンバーを示します。アスタリスクで始まる行は、すべてコメントとして扱われます。1 つの行に定義することのできるキーワードは、1 つだけです。例えば、AUTO1、AUTO2、および AUTO3 についての TASK ステートメントは、行を分けて指定する必要があります。

```
*
* Define the PPI command receiver, and make it APF-authorized.
*
CMDRCVR ID=CNM01
*
* Define two resource object data managers (RODMs) to be managed
* by the DSIQTSK. RODM1 is the default RODM. Both RODMs will
* be connected automatically (using the password) when the DSIQTSK
* task is started.
*
REP  RODM1,CONN=Y,AO=Y,PASS=PASSWORD,T=300,ID=APPL1
REP  RODM2,CONN=Y,AO=N,PASS=PASSWORD,T=300,ID=APPL2
*
* Define three autotasks to which the DSIQTSK can dispatch work.
*
TASK AUTO1
TASK AUTO2
TASK AUTO3
*
```

図 146. DSIQTSK タスクの DSIQTSKI 初期設定メンバーのサンプル

6. 自動化テーブル・ステートメントを DSITBL01 に追加します。このステートメントは、メッセージ IST105I をトラップし、ORCONV コマンドを RODM1 に出してリソース状況を DOWN に変更します。470 ページの図 147 は、これを行う自動化テーブル・ステートメントを示します。

```

IF MSGID = 'IST105I' THEN
  BEGIN;
  IF TOKEN(2) = 'A01A704' THEN
    EXEC(CMD('ORCONV TYPE=CHANGE,
    CLASS=TERMINAL,OBJECT=A01A704,FIELD=STATUS,DATA='DOWN',
    MSGFIELD='AOLEVEL',MSGPARMS='NETOP1,HELD=Y''')
    ROUTE(ONE NETOP1)) DISPLAY(Y);
  END;

```

図 147. IST105I をトラップし、ORCONV コマンドを出す自動化テーブル・ステートメント

このメッセージはオペレーター NETOP1 に表示する必要があるため、DISPLAY は Y に設定します。DISPLAY を N に設定すると、メッセージはどこにも表示されなくなります (ORCONV コマンドが宛先を指定している場合でも)。

7. RODM 変更メソッドを作成し、障害の起こったリソース A01A704 のリカバリーを試行します。

プログラマーが EKGSPPI メソッドを使用するメソッドを作成して VARY NET,ACT コマンドを DSIQTSK に送信すると、DSIQTSK タスクはこのコマンドを受信して、DSIQTSKI に定義された自動タスクの 1 つにこれをディスパッチします。

このシナリオでは、EKGSPPI を呼び出すメソッドは EKGCPPI メソッドです。EKGCPPI は PL/I で作成されます。476 ページの『変更メソッド EKGCPPI のキー・セクション』は、EKGCPPI からの引用を示します。

EKGCPPI メソッドは AOLEVEL フィールドを検査し、VARY NET,ACT コマンドを補助モードで出すかどうかを判別します。このシナリオでは、AOLEVEL は 3 に設定されているので、コマンドの発行は補助モードで行われます。

AOLEVEL フィールドが補助モードを示しているため、EKGSPPI メソッドを呼び出すとき、EKGCPPI は補助モードを要求します。補助モードは、RODM メソッドによって出されたコマンドがいずれも実行されないことを意味します。その代わりに、このコマンドは DSIQTSK によってトラップされ、NetView メッセージの DWO670I として出されます。このメッセージを、自動化テーブルでトラップし、SAVECMD コマンドを使って保存することができます。SAVECMD を使って保存されたコマンドは、ASSISCMD を使用すると、表示したり操作したりすることができます。ASSISCMD を使用する例については、ステップ 9 (473 ページ) を参照してください。

補助モードを使用しないと、DSIQTSK はコマンドを実行するため、自動タスク AUTO1 にディスパッチします。

作成された変更メソッドは、コンパイル後に関係編集して、RODM START プロシージャの STEPLIB データ定義 (DD) ステートメントで指定されたライブラリーの 1 つに入れる必要があります。

8. 自動化テーブル・ステートメントを DSITBL01 に作成します。このステートメントは、メッセージ DWO670I をトラップし、RODM から渡されたコマンドを保存します。

471 ページの図 148 は、これを行う自動化テーブル・ステートメントを示します。このステートメントは、DWO670I メッセージをトラップし、SAVECMD コマンドを使用して、RODM メソッドで出された VTAM VARY コマンド保存を

します。このシナリオでは、この後 SAVECMD は、NetView オペレーター NETOP1 がログオンしていれば NETOP1 に経路指定されます。

SAVECMD コマンドが NETOP1 に経路指定されると、NETOP1 はメッセージ CNM436I を受信し、ASSISCMD を使用して、保存されたコマンドの編集、破棄、もしくは発行を行います。

```
IF MSGID = 'DW0670I' THEN
  EXEC(CMD('SAVECMD'))
    ROUTE(ONE NETOP1))
  DISPLAY(N) NETLOG(Y);
```

図 148. DW0670I をトラップする自動化テーブル・ステートメントのサンプル

9. RODM ローダーへの入力ファイルを作成し、以下のことを行います。

- EKGCPPI メソッドと EKGSPPI メソッドをインストールします。
- このシナリオで使用されるクラスおよびフィールドを作成します。

図 149 は、このシナリオで使われる入力ファイルを示します。『--』の文字で始まる行は、コメント行です。

```
-- Install EKGSPPI and EKGCPPI Method
OP EKG_Method HAS_INSTANCE EKGSPPI;
OP EKG_Method HAS_INSTANCE EKGCPPI;
-- Create the class called TERMINAL under UniversalClass
OP TERMINAL HAS_PARENT UniversalClass;
-- Create fields and subfields for Class TERMINAL
OP TERMINAL HAS_FIELD (INTEGER) AOLEVEL;
OP TERMINAL HAS_FIELD (CHARVAR) STATUS;
OP TERMINAL.STATUS HAS_SUBFIELD CHANGE;
-- Create object instance A01A704 for class TERMINAL
OP TERMINAL HAS_INSTANCE A01A704;
-- Set the value for the fields and subfield for the Object
OP TERMINAL.A01A704.STATUS HAS_VALUE (CHARVAR) 'UP';
OP TERMINAL.A01A704.AOLEVEL HAS_VALUE (INTEGER) 3;
OP TERMINAL.A01A704.STATUS.CHANGE SUBFIELD_HAS_VALUE
  (METHODSPEC) ('EKGCPPI');
```

図 149. RODM ローダーへの入力ファイル

RODM ロード機能に関する詳細については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

シナリオの実行

シナリオを実行するには、次のようにしてください。

1. RODM1 と RODM2 を開始します。複数の RODM データ・キャッシュを開始する方法については、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。
2. RODM2 をロードするために作成した入力ファイルを使って、RODM ローダーを、以下の手順で実行します。
 - EKGCPPI メソッドと EKGSPPI メソッドをインストールします。
 - RODM2 内に、TERMINAL クラス、A01A704 オブジェクト、および STATUS フィールドを作成します。

RODM のロード、およびメソッドのインストールについては、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

3. NetView プログラムに NETOP1 としてログオンします。
4. DSIQTSK タスクを開始します。NetView プログラムが初期設定されたときにこのタスクを開始していない限り、NetView オペレーター・パネルから START TASK=DSIQTSK を実行します。DSIQTSK は、開始されると自動的に RODM1 と RODM2 に接続します。デフォルト RODM は RODM1 になりますが、これは**現行の実行時 RODM**としても参照されます。CNMQAPI サービス・ルーチン、DSINOR マクロ、および ORCONV コマンドは、RODM2 ではなくデフォルト RODM の RODM1 で機能します。

注: DSIQTSK が RODM に接続することができるのは、RODM がアクティブなときに限ります。

5. デフォルト RODM を変更します。このシナリオで活動するクラス、オブジェクト、およびフィールドは RODM2 内のものなので、RODM2 をデフォルト RODM にする必要があります。デフォルト値を変更するには、図 150 のコマンドを使用してください。

```
ORCNTL CHNG,OR=RODM2
```

図 150. デフォルト RODM の変更

これで、自動化テーブルでコーディングした ORCONV コマンドは、RODM2 で機能します。

6. 自動化テーブルを活動化します。自動化テーブルを開始するには、図 151 のコマンドを使用します。

```
AUTOTBL MEMBER=DSITBL01
```

図 151. 自動化テーブルの活動化

A01A704 について IST105I メッセージが受信されると、ORCONV コマンドが出されて、この状況が DOWN に変更されます。状況が DOWN に変更されると、変更メソッドが呼び出され、AOLEVEL が検査されます。AOLEVEL フィールドが 3 に設定されていると、VTAM VARY コマンドが補助モードで出され、実行のために自動タスクへのディスパッチは行われません。

7. DEFAULTS コマンドを使用し、障害の起こったリソースのリカバリーが自動化されていない場合は、そのリソースをオペレーターに通知するようにします。

DEFAULTS コマンドには、SENDMSG というパラメーターがあります。このパラメーターを、ORCONV コマンドの MSGFIELD パラメーターおよび MSGPARMS パラメーターとともに使用すると、ORCONV コマンドが出されるもとなった IST105I メッセージで何を行うべきかを判別することができます。

このシナリオでは、RODM 変更メソッドが AOLEVEL フィールドを検査して、障害の起こったリソースのリカバリーを自動化するかどうかを判別します。リカバリーが自動化されない場合は、リソースに障害が起こったことをオペレーターに通知する必要がありますが、RODM 変更メソッドは、この

IST105I メッセージにアクセスすることができません。 ORCONV コマンドを代わりに使えば、変更メソッドが行ったかのように AOLEVEL フィールドを調べることができます。 ORCONV コマンドはこのフィールドの値に基づいて、ORCONV コマンドで指定したオペレーターに IST105I メッセージを経路指定します。

まず、SENDMSG パラメーターに数値 (もしくは値のリスト) を設定します。このシナリオでは、SENDMSG は 2 に設定されます。図 152 のようなコマンドを、オペレーター・パネルから入力します。

DEFAULTS SENDMSG=2

図 152. DEFAULT SENDMSG パラメーターの設定

MSGFIELD パラメーターは、ORCONV コマンドに SENDMSG で設定された値を AOLEVEL の値と比較するよう要求します (470 ページの図 147 参照)。ORCONV コマンドは、AOLEVEL の値を、SENDMSG によって設定された値と比較します。

AOLEVEL の値が、DEFAULTS SENDMSG コマンドで設定された値の 1 つと一致する場合、ORCONV コマンドを出させたメッセージは、ORCONV の MSGPARMS パラメーターで定義された宛先に経路指定されます。ステップ 6 (469 ページ) では、IST105I メッセージの結果として ORCONV コマンドが出力され、470 ページの図 147 でも、MSGFIELD が AOLEVEL に設定され、MSGPARMS が NETOP1 に設定されることが示されています。そのため、AOLEVEL が 2 に設定された場合は、IST105I メッセージがオペレーター NETOP1 に経路指定されます。

AOLEVEL の値が、DEFAULTS SENDMSG コマンドで設定された値のいずれにも一致しない場合、IST105I メッセージは、MSGPARMS の指定した宛先に経路指定されません。AOLEVEL が 1 に設定された場合も、RODM メソッドは A01A704 の状況を DOWN に変更し、IST105I メッセージは MSGPARMS の指定した宛先に経路指定されません。RODM メソッドは、リソースのリカバリーの自動化を試行します。

8. リソースの A01A704 を非活動化します。

このシナリオを継続するには、リソースを非活動化してください。それには、図 153 のコマンドを NetView プログラムから出します。

V NET,INACT,ID=A01A704,F

図 153. リソース A01A704 を非活動化する例

9. リソースが非活動化されると、メッセージ CNM436I が表示されます。このメッセージは、ASSISCMD コマンドを入力しなければならないことを示します。474 ページの図 154 は、最初の ASSISCMD パネルを示しています。

assispn1 Commands for Operator Assistance

1. V NET,ACT,ID=A01A704
MORE This command was sent by the change method EKGCPPI to act
- 2.
- 3.
- 4.
- 5.
- 6.

Command==>
PF1 = Help
PF6 = Roll

PF2 = Exit

図 154. ASSISCMD コマンドの画面例

補助モードで一度もコマンドが出されなかった場合、CNM436I メッセージが受信されることはなく、VARY NET,ACT コマンドが出されてもオペレーターには表示されません。

ASSISCMD コマンドによって表示される第 1 のパネルは、補助モードで出された最新の 6 つのコマンドを示します。そのコマンドに関連する情報テキストも同時に表示されます。この情報テキストは、480 ページの図 161 に示すような、EKGSPPI メソッドを呼び出すときにコマンドに対応付けられたテキストと同じものです。オペレーターがコマンドを 1 つ処理し終わると、次の保存済みコマンドがパネルに表示されます。表示のキューに入れることのできるコマンドは最高 20 個ですが、この個数は、SAVECMD コマンド・リストを修正すれば変更することができます。

10. 最初の ASSISCMD パネルが表示された後で、オペレーターは、そのパネルのコマンドの隣に、以下の文字を入力することができます。
 - E** そのコマンドを、表示されているとおり実行します。このオプションは、第 1 パネルと第 2 パネルから入力することができます。
 - D** コマンドを削除します。このオプションは、第 1 パネルと第 2 パネルで入力することができます。
 - M** コマンドを修正するか、詳細を表示します。表示する情報がさらにある場合は、MORE が第 1 パネルのコマンド直後の行頭に表示されます。M オプションは、第 2 の ASSISCMD パネルを表示します。オペレーターは、コマンド全体、およびそのコマンドに関連する情報テキストを表示させることができます。

図 154 では、コマンドの下に MORE という語が表示されています。この語は、そのコマンドに関する詳細が利用可能であることを示します。

11. 475 ページの図 155 のように、**M** をコマンドの隣に入力します。

```

assispn1          Commands for Operator Assistance

1. M              V NET,ACT,ID=A01A704
                  MORE This command was sent by the change method EKGCPPI to act
2.
3.
4.
5.
6.

Command==>
PF1 = Help          PF2 = Exit
PF6 = Roll

```

図 155. ASSISCMD コマンドの画面例 -- 詳細を表示する M の入力

12. **M** を入力すると、図 156 のような第 2 の ASSISCMD パネルが表示されます。このパネルには、コマンドが出された理由の説明が表示されます。このテキストは、変更メソッド EKGCPPI で作成され (480 ページの図 161 参照)、パラメーターとしてメソッド EKGSPPI に渡されます。

```

assispn2          Full Detail of Command for Operator Assistance

                  V NET,ACT,ID=A01A704

This command was sent by the change method EKGCPPI to activate a
resource. This command was sent because both of the following
conditions have occurred: (1) The status of the resource has been
set to DOWN in RODM. (2) RODM indicates that recovery of this
resource should be attempted automatically.

Command==>
PF1 = Help          PF2 = Exit          PF3 = Previous Panel
PF6 = Roll

```

図 156. ASSISCMD コマンドの画面例 -- コマンドの詳細

13. このコマンドを処理するには、476 ページの図 157 のように、コマンドの隣に **E** を入力します。コマンドを実行する前に編集しておきたいときは、コマンドに上書きしてから **E** を入力し、そのコマンドを実行します。

```

assispn2          Full Detail of Command for Operator Assistance

E    V NET,ACT,ID=A01A704

This command was sent by the change method EKGCPPI to activate a
resource. This command was sent because both of the following
conditions have occurred: (1) The status of the resource has been
set to DOWN in RODM. (2) RODM indicates that recovery of this
resource should be attempted automatically.

Command==>
PF1 = Help          PF2 = Exit          PF3 = Previous Panel
PF6 = Roll

```

図 157. ASSISCMD コマンドの画面例 -- コマンドを実行する E の入力

変更メソッド EKGCPPI のキー・セクション

ここでは、変更メソッド EKGCPPI から特定部分を選んで説明します。EKGCPPI は、配布テープ上の NetView サンプルです。

以下の EKGCPPI からの引用は、変更メソッドを理解するためのものなので、引用部分だけでなく、メソッド全体を調べてください。それぞれの引用のあとには、フィールドまたはステートメントの一部の説明が続きます。変更メソッドの作成方法については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

プロシージャ・ステートメント

```

:
EKGCPPI: PROCEDURE (IN_FLD_ID,IN_LL,IN_SLP,IN_DATATYPE,
                   IN_CHARLEN, IN_DATAPTR)
          OPTIONS (REENTRANT); 1

```

図 158. 変更メソッド EKGCPPI のプロシージャ・ステートメント

キー 説明

- 1** これは変更メソッドであるため、RODM に入っている STATUS フィールドの VALUE サブフィールドに対して行われる物理的変更に責任があります。この変更を行うには、変更対象のフィールドおよびそのフィールドの新規の値をメソッドに知らせる必要があります。この情報は、RODM によって変更メソッドに渡されるので、そのメソッドのプロシージャ・ステートメントのパラメーターとして定義しておく必要があります。

このシナリオで、ORCONV コマンドは、STATUS フィールドの値の DOWN への変更を試行します。しかし、変更メソッド EKGCPPI は

STATUS フィールドに関連しているため、RODM は EKGCPPI メソッドを起動して、そのフィールドの名前および新規の値を EKGCPPI に渡します。

ローカル変数

```

/*****
/*
/*                                LOCAL VARIABLES                                */
/*
/*
/*****
:
:
:                                /* Selfdefining data for                        */
:                                /* OI method EKGSPPI 1 */
DCL 1 EKGSPPI_BLK UNALIGNED,
3 TOTAL_LEN Smallint,          /* Not including its length */
3 RCVRID_CHARVAR, 2            /* Receiver id CharVar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CHAR_LEN), /* CharVar len */
5 CHAR_DATA CHAR(8) INIT('CNM01'), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 ASSIST_CHARVAR, 3            /* Assist information CharVar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CHAR_LEN), /* CharVar len */
5 CHAR_DATA CHAR(8), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 TASKINFO_CHARVAR, 4         /* Task information CharVar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CHAR_LEN), /* CharVar len */
5 CHAR_DATA CHAR(8) INIT('ONLYANY'), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 TASKNAME_CHARVAR, 5         /* Task name CharVar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CHAR_LEN), /* CharVar len */
5 CHAR_DATA CHAR(8) INIT('AUT01'), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 SENDER_CHARVAR, 6           /* Sender token CharVar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CHAR_LEN), /* CharVar len */
5 CHAR_DATA CHAR(8) INIT('EKGCPPI'), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 CMD_CHARVAR, 7              /* Command Charvar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CMD_LEN), /* CharVar len */
5 CHAR_DATA CHAR(MAX_CMD_LEN), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B), /* Null data */
3 CMD_DESC_CHARVAR, 8         /* Command Description Charvar */
5 DATA_TYPE Smallint INIT(EKG_DT_CharVar), /* Data type */
5 CHAR_LEN Smallint INIT(MAX_CMD_DESC_LEN), /* CharVar len */
5 CHAR_DATA CHAR(MAX_CMD_DESC_LEN), /* CharVar data*/
5 NULL_DATA BIT(8) INIT ('00000000'B); /* Null data */
:
:
:

```

図 159. 変更メソッド EKGCPPI のローカル変数

キー 説明

- 1** EKGSPPI は、コマンドをプログラム間インターフェースに置き、そのコマンドを NetView タスク DSIQTSK に送信するメソッドです。この自己定義ストリングには、オブジェクト独立メソッド EKGSPPI に渡される 7 個のパラメーターが含まれます。入力パラメーターに先行ブランクがある場合は、処理前にすべて削除されます。
- 2** RCVRID_CHARVAR ステートメントは、コマンド受信先名を CNM01 とし

て定義します。これは、プログラム間インターフェースを介して DSIQTSK にコマンドを送信するときに、EKGSPPI が使用する受信先名です。DSIQTSKI 初期設定メンバーに定義された受信先名と同一のものです。

3 ASSIST_CHARVAR ステートメントは、ASSIST か NOASSIST のいずれかを EKGSPPI に渡す変数を定義します。この変数の値が ASSIST の場合、DSIQTSK タスクに送られたすべてのコマンドは、補助モードで出されます。このシナリオでは、この変数の値は、RODM 内にある AOLEVEL フィールドの値に基づいています。

4 TASKINFO_CHARVAR ステートメントは、DSIQTSK がコマンドを、特定の自動タスクまたは次に使用可能な自動タスクに、ディスパッチするかどうかを指定します。CHAR_DATA ステートメントには、以下の属性があります。

属性 意味

ONLYANY

特定の自動タスクが使用可能な場合、その自動タスクが使用されます。ただし、使用可能でない場合は、DSIQTSK に定義された、次の使用可能な自動タスク（最後に使用された自動タスクの次のタスク）がコマンドを出します。自動タスクは、DSIPARM の DSIQTSKI メンバーに定義された順序で使用されます。

ONLY 特定の自動タスクが使用されます。この自動タスクが使用中の場合、コマンドはその自動タスクのキューに入ります。指定された自動タスクが使用可能でない場合、コマンドは出されません。

ANY DSIQTSK に定義された次の自動タスク（最後に使用された自動タスクの次のタスク）が、コマンドを出します。自動タスクは、DSIPARM の DSIQTSKI メンバーに定義された順序で使用されます。

5 TASKNAME_CHARVAR ステートメントは、DSIQTSK が、CMD_CHARVAR に指定されたコマンドを、自動タスク AUTO1 にディスパッチすることを指定します。

6 SENDER_CHARVAR ステートメントは、コマンドを EKGSPPI に送信するメソッドを識別します。このシナリオで、EKGCPPI は ID または送信元トークン として使用されます。

7 CMD_CHARVAR ステートメントは、EKGSPPI から DSIQTSK に渡されるコマンドの名前を含む変数を定義します。

8 CMD_DESC_CHARVAR ステートメントは、EKGSPPI から DSIQTSK に送信されるコマンドを記述するテキストを含む変数を定義します。コマンドが補助モードで出され、オペレーターが ASSISCMD を入力した場合はこのテキストが表示されます。

定数

```
/*
/*
/*          CONSTANTS
/*
/*
/*
*****
:
:
DCL $ASSISTON  FIXED BIN(31) INIT(3); 1 /* Send cmd to DSIQTSK*/
DCL $ASSISTOFF FIXED BIN(31) INIT(1); 2 /* Send cmd to DSIQTSK*/
:
:
DCL FIELD_AO   CHAR(7) INIT('AOLEVEL'); 3 /* Field name for AOLEVEL */
DCL FIELD_MyName CHAR(6) INIT('MyName'); 4 /* MyName Field          */
DCL EKGSPPI_NAME CHAR(7) INIT('EKGSPPI'); 5 /* EKGSPPI OI method name */
/* CMD Value                                */
DCL CMD_VALUE CHAR(MAX_CMD_LEN - RESOURCE_NAME_SIZE) VARYING
INIT('V NET,ACT,ID='); 6
/* CMD description                          */
DCL CMD_DESC_VALUE CHAR(MAX_CMD_DESC_LEN) VARYING; 7
:
:
```

図 160. 変更メソッド EKGCPPI の定数

キー 説明

- 1 \$ASSISTON 定数は、AOLEVEL フィールドが 3 に設定されているかどうかを判別するために使用されます。3 という値は、そのオブジェクト独立メソッド EKGSPPI が補助モードでコマンドを出すことを示します。
- 2 \$ASSISTOFF 定数は、AOLEVEL フィールドが 1 に設定されているかどうかを判別するために使用されます。1 という値は、そのオブジェクト独立メソッド EKGSPPI が補助モード以外で DSIQTSK にコマンドを送信することを示します。つまり、コマンドは DSIQTSK に送信され、自動タスクにディスパッチされたあと、処理されます。
- 3 FIELD_AO 定数には、補助モードでコマンドが出すかどうかを判定するために EKGCPPI が検査するフィールドの名前が入ります。このシナリオでは、このフィールドは AOLEVEL です。
- 4 FIELD_MyName 定数には、オブジェクトの名前を検出するために EKGCPPI が照会するフィールドの名前が入ります。このフィールドは、MyName です。
- 5 EKGSPPI_NAME 定数には、この変更メソッド (EKGCPPI) がプログラム間インターフェースを介してコマンドを DSIQTSK に送信するために起動するオブジェクト独立メソッドの名前が入ります。このシナリオでは、メソッドは EKGSPPI です。
- 6 CMD_VALUE 定数には、プログラム間インターフェースで EKGSPPI が DSIQTSK に送信する実際のコマンドが入ります。このシナリオでは、コマンドは V NET,ACT,ID=opid です。このメソッドのあとの部分で、リソース名が決定されてこのコマンド・ストリングと連結されます。
- 7 CMD_DESC_VALUE 定数には、コマンドに対応する記述テキストが入ります。この記述テキストは、オペレーターが ASSISCMD を入力したときに表示されます。この定数の値は初期設定されず、このメソッドのあとの部分で割り当てられます。

初期設定

```
/* **** */
/*      Initialization      */
/* **** */
/* Set change subfield fid */
F1403_FUNC_BLK.Function_ID = EKG_ChangeSubfield; 1
/* Trigger OI method func id */
F1416_FUNC_BLK.Function_ID = EKG_TriggerOIMethod; 2
/* Set the query subfield fid */
F1502_FUNC_BLK.Function_ID = EKG_QuerySubfield; 3
:
/* Set the cmd desc value */
CMD_DESC_VALUE = 'This command was sent by the change method ' 4
                  'EKGCPPI to activate a resource. '
                  'This command was sent because both of the '
                  'following conditions have occurred: '
                  '(1) The status of the resource has been set '
                  'to DOWN in RODM. '
                  '(2) RODM indicates that recovery of this '
                  'resource should be attempted automatically. ';
```

図 161. 変更メソッド EKGCPPI の初期設定

図 161 はメソッドの一部を示すもので、このメソッド全体で使用される RODM 機能ブロック内の機能 ID フィールドに値を割り当てます。このメソッドでは以下の機能が使用されます。

- サブフィールドの変更
- オブジェクト独立メソッドの起動
- サブフィールドの照会

これらは、標準の RODM 機能です。RODM 機能の詳細については、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

キー 説明

- 1** F1403_FUNC_BLK ステートメントは、サブフィールドを変更するための機能 ID を割り当てます。
- 2** F1416_FUNC_BLK ステートメントは、オブジェクト独立メソッドを起動するための機能 ID を割り当てます。このシナリオで開始されるメソッドは、EKGSPPI です。
- 3** F1502_FUNC_BLK ステートメントは、サブフィールドを照会するための機能 ID を割り当てます。このシナリオでは、EKGCPPI メソッドは以下の 2 つのフィールドを照会します。
 - AOLEVEL フィールドの VALUE サブフィールド
 - MyName フィールドの VALUE サブフィールド
- 4** CMD_DESC_VALUE ステートメントは、479 ページの図 160 の CMD_VALUE 定数で定義されたコマンドに関連してテキストを定義します。

サブフィールドの変更

```
/******  
/* Change the value subfield of the input */  
/******  
:  
:  
/* Set datatype of change fld */  
F1403_FUNC_BLK.Data_type = IN_DATATYPE; 1  
/* Set char data length */  
F1403_FUNC_BLK.New_char_data_length = IN_CHARLEN; 2  
/* Set new data pointer */  
F1403_FUNC_BLK.New_data_ptr = IN_DATAPTR; 3  
:  
:  
/* Change subfield */  
CALL EKGMAPI(TRANS_INFO_BLK,F1403_FUNC_BLK,RESPONSE_BLK); 4  
:  
:
```

図 162. 変更メソッド EKGCPPI のサブフィールドの変更

図 162 は、メソッドの中で STATUS フィールドの値を変更する部分を示します。STATUS フィールドは、メソッドに関連のあるフィールドです。

キー 説明

- 1** IN_DATATYPE は、STATUS フィールドのデータ・タイプを指定します。このパラメーターは、476 ページの図 158 のプロシージャー・ステートメントで指定されています。
- 2** IN_CHARLEN は、新しいデータの長さを指定します。このシナリオの新しい値は、DOWN です。このパラメーターは、476 ページの図 158 のプロシージャー・ステートメントで指定されています。
- 3** IN_DATAPTR は、新しいデータを指すポインターを指定します。このパラメーターは、476 ページの図 158 のプロシージャー・ステートメントで指定されています。
- 4** このステートメントは、RODM API EKGMAPI を呼び出します。EKGMAPI は、状況フィールドを実際に変更します。機能 ID の 1403 は、480 ページの図 161 に EKG_ChangeSubfield として定義されています。

フィールドの照会

```

/*****
/*      Query the AOLEVEL field to see it is 1      */
*****/
:
:
/* Length is set */
FIELD_ACCESS_INFO_BLK.Field_name_length = LENGTH(FIELD_AO); 1
/* Set the field ptr */
FIELD_PTR = ADDR(FIELD_AO); 2
:
:
/* Query a AOLEVEL subfield */
CALL EKGMAPI(TRANS_INFO_BLK,F1502_FUNC_BLK,RESPONSE_BLK); 3
:
:
/* Send the command or not */
IF TEMP_DATA_VALUE = $ASSISTON |
TEMP_DATA_VALUE = $ASSISTOFF THEN
DO; 4
/* Prepare to send command */
/* Is ASSIST ON */
IF TEMP_DATA_VALUE = $ASSISTON THEN
EKGSPPI_BLK.ASSIST_CHARVAR.CHAR_DATA = 'ASSIST'; 5
ELSE
EKGSPPI_BLK.ASSIST_CHARVAR.CHAR_DATA = 'NOASSIST'; 6

```

図 163. 変更メソッド EKGCPPI のフィールドの照会

図 163 は、メソッドの一部で、オブジェクト独立メソッド EKGSPPI が補助モードでコマンドを出すかどうかが判別されます。変更メソッド EKGCPPI は、障害の起こったリソースに対応する AOLEVEL サブフィールドを調べます。

- AOLEVEL の値が 1 (定数 \$ASSISTOFF) の場合、EKGSPPI が ASSIST オプションで呼び出されます。
- AOLEVEL の値が 3 (定数 \$ASSISTON) の場合、EKGSPPI は NOASSIST オプションで呼び出されます。
- AOLEVEL の値がこれ以外の場合、EKGSPPI は呼び出されません。

キー 説明

- 1** このステートメントは、フィールド名の長さを定義します。
- 2** このステートメントは、AOLEVEL フィールド名のアドレスを定義します。FIELD_AO には、479 ページの図 160 の AOLEVEL の値が与えられます。
- 3** このステートメントは EKGMAPI を呼び出し、AOLEVEL フィールドを照会します。機能 ID 1502 は、480 ページの図 161 の EKG_QuerySubfield として定義されています。
- 4** TEMP_DATA_VALUE は、サブフィールド照会の結果に基づいて設定されています。この IF ステートメントは、TEMP_DATA_VALUE の値を検査し、DSIQTSK へのコマンド送信を補助モードで行うかどうかを判別します。
- 5** このステートメントは、TEMP_DATA_VALUE の値が \$ASSISTON であるかどうかを判別します。\$ASSISTON の場合、ASSIST_CHARVAR 変数 (477 ページの図 159 参照) は ASSIST に設定されます。DSIQTSK に送信

されたコマンドは、補助モードで出されます。オペレーターは ASSISCMD コマンドを使って、コマンドの表示、修正、および発行を行うことができます。

- 6** このステートメントは、TEMP_DATA_VALUE の値が \$ASSISTOFF であるかどうかを判別します。\$ASSISTOFF の場合、ASSIST_CHARVAR 変数 (477 ページの図 159 参照) は NOASSIST に設定されます。DSIQTSK に送信されたコマンドは、実行するために自動タスクにディスパッチされますが、補助モードは使用されません。

オブジェクト名の照会

```
/******  
/* Query the Object name for V NET,ACT,ID=objectname */  
/******  
:  
:  
/* Length is set */  
FIELD_ACCESS_INFO_BLK.Field_name length  
= LENGTH(FIELD_MyName); 1  
/* Set the field ptr */  
FIELD_PTR = ADDR(FIELD_MyName); 2  
:  
:  
/* Query a Object name subfld */  
CALL EKGMAPI(TRANS_INFO_BLK,F1502_FUNC_BLK,RESPONSE_BLK); 3  
:  
:
```

図 164. 変更メソッド EKGCPPI のオブジェクト名の照会

メソッドの上記の部分は、障害の起こったりソースのオブジェクト名を判別します。このシナリオでは、オブジェクト名はリソース名です。この名前は、CMD_VALUE の値と連結され、EKGSPPI に送られます。

キー 説明

- 1** このステートメントは、フィールド名の長さを定義します。
- 2** このステートメントは、MyName フィールド名のアドレスを定義します。(FIELD_MyName には、479 ページの図 160 の MyName の値が与えられます。)
- 3** このステートメントは EKGMAPI を呼び出し、MyName フィールドの VALUE サブフィールドを照会します。機能 ID 1502 は、480 ページの図 161 の EKG_QuerySubfield として定義されています。

オブジェクト独立メソッドの起動

```
/* **** */
/* Prepare to trigger OI method EKGSPPI to send the */
/* command V NET,ACT,ID=objectname to DSIQTSK */
/* **** */
:
:
:                               /* Set the cmd with */
:                               /* MyName field value */
EKGSPPI_BLK.CMD_CHARVAR.CHAR_DATA =
  CMD_VALUE || SUBSTR(CHARVAR_RESP_BLK.CHAR_DATA,
    1,CHARVAR_RESP_BLK.CHAR_LEN); 1
:                               /* Set the cmd description */
EKGSPPI_BLK.CMD_DESC_CHARVAR.CHAR_DATA =
  CMD_DESC_VALUE; 2
:                               /* Set OI method name */
F1416_FUNC_BLK.Method_name = EKGSPPI_NAME; 3
:                               /* Set selfdefining parm */
F1416_FUNC_BLK.Method_parms = ADDR(EKGSPPI_BLK); 4
:
:                               /* Message triggered MAPI call*/
CALL EKGMAPI(TRANS_INFO_BLK,F2009_FUNC_BLK,
  RESPONSE_BLK); 5
:
:
```

図 165. 変更メソッド *EKGCPPI* のオブジェクト独立メソッドの起動

メソッドのこの部分は、オブジェクト独立メソッド *EKGSPPI* を起動します。*EKGSPPI* は、*CMD_CHARVAR* に定義されたコマンドをプログラム間インターフェースを介して *DSIQTSK* に送信します。

キー 説明

- 1 このステートメントは、出されるコマンドとリソース名を連結します。そのあと、完全なコマンドが変数 *CMD_CHARVAR* に入れられ、それが *EKGSPPI* に送られる自己定義ストリングの一部になります。
- 2 このステートメントは、コマンドに関連するテキスト（480 ページの図 161 参照）を変数 *CMD_DESC_CHARVAR* に入れ、それが *EKGSPPI* に送られる自己定義ストリングの一部になります。
- 3 このステートメントは、起動されるオブジェクト独立メソッドの名前として *EKGSPPI* を定義します。
- 4 このステートメントは、オブジェクト独立メソッド *EKGSPPI* のパラメーター・リスト（自己定義ストリング）のアドレスを定義します。
- 5 このステートメントは *EKGMAPI* を呼び出して、*EKGSPPI* を開始します。

第 29 章 端末アクセス機能を使用した自動操作

NetView の端末アクセス機能 (TAF) は VTAM 中継機能で、これを使用すると、NetView オペレーターの端末を、LU1 タイプまたは LU2 タイプのプロトコルをサポートしているアプリケーションで、これらのタイプの端末のように扱うことができます。これらのアプリケーションには、顧客情報管理システム (CICS) や情報管理システム (IMS) アプリケーションなどがありますが、それ以外のアプリケーションにも使用可能です。TAF を使うと、NetView オペレーターと自動タスクの両方で、そのアプリケーションのサブシステム・コンソールまたはマスター端末にログオンしているかのように、メッセージを表示したりコマンドを出したりすることができます。NetView をアプリケーションにリンクする TAF LU1 セッションを通じて送られたメッセージは、自動化テーブルで処理することができ、標準 NetView 自動化機能を使って自動化することができます。

TAF は、自動化されたアプリケーションからそのアプリケーションのコンソールまたはマスター端末へのメッセージが、オペレーティング・システム・メッセージ処理機能で使えない場合に特に有効です。例えば MVS システムでは、CICS および IMS は両方とも、コンソールまたはマスター端末にメッセージを生成しますが、それらのメッセージは、サブシステム・インターフェースでブロードキャストされないため、TAF を使用しない限り、NetView は自動化のためにそれらのメッセージを使用することはできません。

概説

TAF の操作には、2 つのモードがあります。1 つの操作モードである、**オペレーター制御** または **OPCTL** セッションは、LU1 セッションで、SNA 3767 端末をエミュレートします。オペレーター制御セッションは、メッセージとコマンドの伝送を、フルスクリーン・モードでなく行単位モードで行います。通常はアプリケーション・サブシステムのコンソールまたはマスター端末でオペレーターに表示されるメッセージは、LU1 セッションを通じて NetView プログラムに送信され、自動化テーブルで処理されます。TAF を含む自動化には、オペレーター制御 (LU1) セッションが使われるため、自動タスクは、アプリケーションに直接接続された 3767 端末から、可能な任意のトランザクション (通常 CICS や IMS のマスター端末から入力される CICS や IMS の制御機能を含む) を入力することができます。

もう 1 つの操作モードである、**フルスクリーン** または **FLSCN** セッションは、LU2 セッションで、SNA 3270 端末をエミュレートします。あるアプリケーションでフルスクリーン・セッションを確立すると、NetView オペレーターは、ローカルに接続されたサブシステム・コンソールまたはマスター端末からそのアプリケーションそのものに直接ログオンしたかのように、そのアプリケーションの画面を NetView 端末で表示することができます。NetView 画面に表示されるデータは、行単位形式ではなくフルスクリーン形式で伝送されるので、NetView 自動化処理機能はそのデータを使うことはできません。LU2 セッションを介して受信され、NetView 画面に表示されたメッセージは、自動化テーブルでは処理されません。自動タスクは、定義上、物理端末を持たないため、フルスクリーン・セッションを見ることはできません。

こうした理由から、フルスクリーン・セッションを自動化に使用することはできません。しかし、自動化でまだアプリケーションの制御機能すべてを処理していない場合には、フルスクリーン・セッションを自動化に使用することができます。この場合、NetView オペレーターは、TAF のフルスクリーン・セッションを使って、必要なアクションを実行することができます。この機能は、複数のサブシステム・コンソールを統合するときに重要な役割を持ち、これによって 1 つの NetView コンソールから、複数のサブシステムおよびアプリケーションを操作することができますようになります。中央システムのオペレーターが、(通常はリモートの) 複数のシステム上のアプリケーションを制御するフォーカル・ポイント操作では、この機能は重要です。

LU1 と LU2 の両セッションは、VTAM セッション・タイプです。オペレーター制御セッションであるか、フルスクリーン・セッションであるかにかかわらず、すべての TAF セッションでは、VTAM プログラムがアクティブでなければなりません。

TAF の動作方法

TAF は、TAF 仮想端末 (ソース LU (SRCLU) と呼ばれる) とアプリケーションの間でセッションが確立されることで動作します。ソース LU は 2 次論理装置で、アプリケーションが 1 次論理装置です。同じソース LU で、同時に複数のアプリケーションとセッションを確立することができます。オペレーターか自動タスクかにかかわらず、オペレーター端末タスク (OST) ごとに、1 つのソース LU を使用することができます。オペレーターは、このソース LU と各アプリケーションの間で TAF セッションを開始させれば、複数のアプリケーションを制御することができます。TAF セッションを介してアプリケーションから戻されたメッセージまたは表示は、すべて、このセッションを開始したソース LU に対応するオペレーター・タスクで受信されます。

表 17 は、TAF を使って NetView プログラムから制御することのできるアプリケーションの一部をリストしたものです。TAF は他の 3270 アプリケーションでも使用できます。TAF を使用すると、フルスクリーン・モードに限り、別の NetView システムにログオンしてコマンドを入力することもできます。

表 17. 端末アクセス機能オプション

サブシステム	オペレーター制御	フルスクリーン
CICS	●	●
IMS	●	●
HCF DPPX	●	●
HCF DPCX		●
TSO		●
リモート NCCF		●
TCAM VER		●
DSX		●
NPM		●
SSP (THRU TSO)		●

TAF のセットアップ

TAF のシステム・セットアップは、定義ステートメントを TAF ソース LU 用の VTAMLST データ・セットに追加することと、端末定義を同じソース LU のために CICS や IMS などのアプリケーションに追加することで構成されています。以下のセクションでは VTAMLST データ・セット、CICS、および IMS に用意されている定義サンプルを説明しますが、これらのサンプルを使用すれば、システムを簡単にセットアップすることができます。

VTAMLST 定義の追加

自動化のために TAF をセットアップするには、TAF SRCLU (仮想端末またはソース LU) 用の定義を、VTAMLST データ・セットに追加する必要があります。NetView サンプル・ネットワークでは、オペレーター制御セッションとフルスクリーン・セッションの両方の定義サンプルが、VTAMLST データ・セットのメンバー A01APPLS (CNMS0013) に入っています。このサンプルでは、TAF01O00、TAF01O01、TAF01O02 のような名前を持つソース LU がオペレーター制御セッションになっており、フルスクリーン・セッションは、TAF01F00、TAF01F01、TAF01F02 のような名前のソース LU になっています。

図 166 は、TAF ソース LU 用の VTAMLST 定義サンプルを、NetView サンプルから引用したものです。

```
TAF01000    APPL    MODETAB=AMODETAB,EAS=9                X
              DLOGMOD=M3767
*           STATOPT='TAFAPPL 000'
```

図 166. TAF ソース LU のための VTAMLST 定義サンプル

図 166 のパラメーターは、以下のとおりです。

パラメーター 意味

MODETAB

各端末タイプのためのログモードを定義する VTAMLST データ・セットのテーブルを示します。

DLOGMOD

セッションを初期化しようとするときに、そのソース LU 用に、ログモード・テーブルのどのログモードを使用するかを指定します。

AMODETAB

NetView サンプル・ネットワークで、VTAMLST データ・セット・メンバー AMODETAB (CNMS0001) に用意されているログモード・テーブル例を表します。

ログモードには、ソース LU セッションを開始した端末用のバインド・パラメーターが指定されており、画面サイズやカラー機能などの情報が含まれています。TAF オペレーター制御セッションの場合、図 166 に示すようにログモードは常に M3767 にする必要があります。オペレーター制御セッションを介して受信された情報は、常に行単位で表示されるので、自動タスクの場合と同様、物理端末が含まれている必要はありません。物理端末が含まれている場合、その物理的特性は関係ありません。

TAF フルスクリーン・セッションの場合、正しいログモードを使用して TAF セッションを確立してください。端末の物理的特性によってフルスクリーン表示の形式設定が決まるため、オペレーターは、フルスクリーン・セッションを開始するとき、NetView プログラムへのログオン時に使用したのと同じログモードを使用する必要があります。

NetView オペレーターがログオンするときに使用するログモードを、NetView プログラムでは追跡しません。そのため、オペレーターは、それぞれの物理端末タイプに関連したログモードを知った上で、フルスクリーン TAF セッションを開始する際にそのログモードを指定する必要があります。フルスクリーン TAF セッションを開始しなければならないすべての NetView オペレーターに必要な各種ログモードについて、十分なソース LU が VTAMLST データ・セットに定義されているかどうかを確認してください。

NetView プログラムとターゲット・アプリケーションの両方を VTAMLST ですべてに定義しているため、TAF セッションを開始するためには、VTAMLST データ・セットと NetView プログラムのいずれにおいても、これ以上の定義は必要ありません。しかし、ソース LU を端末としてアプリケーションに定義するには、アプリケーションに定義を追加する必要があります。

CICS 端末定義の追加

図 167 は、TAF オペレーター制御セッションを CICS に定義する CICS のサンプルを示します。

```
DFHTCT TYPE=TERMINAL,TRMTYPE=3767,NETNAME=TAF01000,TRMIDNT=TOFO,  
        GMSG=YES,RELREQ=(YES,YES),TIOAL=256,VF=YES,  
        LOGMODE=M3767,BUFFER=256,TRMSTAT=TRANSCIVE,RUSIZE=256,  
        BMSFEAT=(NOROUTE,NOROUTEALL),PGESTAT=PAGE,  
        PGESIZE=(12,80)
```

図 167. CICS への TAF の定義

図 167 のキーワードは、以下のとおりです。

TRMTYPE

端末タイプは、オペレーター制御セッションの場合は常に 3767 にする必要があります。そうすれば、TAF が 3767 端末をエミュレートするからです。フルスクリーン・セッションの場合は、これを NetView オペレーターが使用する物理端末にしておく必要があります。

NETNAME

TAF ソース LU (SRCLU) を示します。特定の時点でアプリケーションにログオンの必要があるそれぞれの TAF SRCLU ごとに、別々の CICS 定義をセットアップしておく必要があります。

LOGMODE

オペレーター制御セッションの場合は、ソース LU ログモードと一致させるために、M3767 にする必要があります。フルスクリーン・セッションの場合は、ログモードはソース LU 定義のログモードと一致していなければならないので、NetView オペレーターが使用する物理端末のタイプによって決められます。

TRMSTAT

TAF を使うセッションを介する送受信を使用可能にするためには、
TRANSCIVE にする必要があります。

この他のすべてのパラメーターについては、担当の CICS アプリケーション・プログラマーに問い合わせてください。

これらの端末定義を簡単に作成するために CICS の自動インストール機能があり、これを使うと、LU1 と LU2 の両方のタイプの端末に対する定義を作成することができます。LU1 タイプの端末に対する定義を作成した場合は、結果を検討して、予期していたセキュリティーが達成されていることを確認します。

IMS 端末定義の追加

図 168 は、IMS 定義の例を示しています。

```
TYPE    UNITYPE=SLUTYPE1
TERMINAL NAME=TAF01000
NAME TAF01000
```

図 168. IMS への TAF の定義

IMS とのセッションで使用されると思われる数の SRCLU 用に、IMS 生成の中に定義ステートメントを含めます。これらの定義ステートメントは、追加の論理名が割り当てられているプリンターや端末のように、静的に定義されている端末定義とともに指定されます。

TAF に使用される NetView コマンド

オペレーターまたは自動タスクがアプリケーションで TAF セッションを開始するには、NetView BGNSESS コマンド、または BGNSESS 構文を単純にした NetView コマンド・リストの 1 つを入力します。(BOSESS はオペレーター制御セッションを開始し、BFSESS はフルスクリーン・セッションを開始します。)

自動タスクでパスワードまたはパスワード・フレーズを入力し、アプリケーションにログオンする場合は、セキュリティーを考慮する必要があります。パスワードと抑止文字の発行については、「*IBM Tivoli NetView for z/OS*セキュリティー・リファレンス」を参照してください。

BGNSESS コマンドまたは BOSESS コマンドを出すと、自動タスクでオペレーター制御セッションの確立とログオンを行うことができます。自動タスクには、正しいパスワードを指定する必要があり、システム・ログまたはネットワーク・ログにパスワードまたはパスワード・フレーズを記録せずに、パスワードを入力する必要があります。そのため、NetView コマンド・リスト言語でログオン・プロシーチャーを作成し、パスワードまたはパスワード・フレーズが入っているコマンドの前に NetView 抑止文字を指定し、コマンドを抑止します。この抑止文字のデフォルト値は、疑問符 (?) です。

さらに、コマンド・プロシーチャーをコマンド許可を指定して保護します。LIST および BROWSE コマンドでのキーワード保護についての詳細は、「*IBM Tivoli NetView for z/OS*セキュリティー・リファレンス」を参照してください。

現在確立している TAF セッションを介して、パスワードまたはパスワード・フレーズを入力するかアプリケーションにコマンドを送信する場合は、SENDSSESS コマンドを使ってください。コマンドは、オペレーター制御セッションが確立可能な任意のアプリケーション (CICS/VS、IMS/VS、および HCF など) に送信することができます。SENDSSESS コマンドへの応答としてアプリケーションに生成されたすべてのメッセージは、その SENDSESS コマンドを出したオペレーターまたは自動タスクに受信されます。

アクティブな TAF セッションに関する情報を表示するには、LISTSESS コマンドまたは LSESS コマンド・リストを使用します。TAF セッションを終了するには、ENDSESS コマンドか ESESS コマンド・リストを使用します。

TAF に使用されるコマンドの構文については、NetView オンライン・ヘルプを参照してください。

TAF を使用したアプリケーションの自動化

DB2® などの一部のアプリケーションでは、TAF を使わずにすべての自動化を実行することができます。それ以外の IMS などのアプリケーションでは、行う自動化によって、TAF を使わなくてもよいものと使わなければならないものがあります。

IMS 環境の操作可能メッセージには、以下の 3 種類の送信パスがあります。

- メッセージがコンソールのみに送られる。
- メッセージが、IMS マスター端末オペレーター (MTO) のみに送られる。
- メッセージが、コンソールと MTO に送られる。

自動化の設計は、メッセージ・フローによって異なります。

MVS システムでは、サブシステム・インターフェースでブロードキャストされるメッセージは、TAF を使用しなくても NetView 自動化に使用することができます。例えば、以下のタイプの機能に含まれるメッセージを自動化する場合、TAF は必要ありません。

- システムの始動
- システムのシャットダウン
- リカバリーまたは再始動
- 拡張回復機能 (XRF) の場合、システム間の USERVAR 情報の維持管理

サブシステム・インターフェースでブロードキャストされないメッセージを自動化するには、TAF が必要です。例えば IMS では、エラー・メッセージは IMS マスター・コンソール・オペレーターに送られます。

アプリケーションに対して TAF セッションを確立し、マスター端末またはアプリケーション・コンソールのエミュレーションを行ってアプリケーションにログオンすると、これらのメッセージが TAF セッションを介して NetView プログラムに転送され、自動化に使われるようになります。

アプリケーションへのオペレーター制御セッションが確立している場合、そのアプリケーションまたはサブシステムから送られたメッセージは、他のメッセージと同

様、NetView 自動化処理の対象となります。タイマー・コマンドや自動化テーブルなどの自動化機能では、TAF セッションを介してコマンドをアプリケーションに出すことができます。

第 30 章 Common Base Event に関する自動化

この章では、Common Base Event の概念および Common Base Event を管理する Common Event Infrastructure の概念を紹介します。

注: この章では、Common Base Event のことをイベントと呼ぶこともあります。

Common Base Event の紹介

NetView プログラムは、メッセージと管理サービス単位 (MSU) を選択し、それらを使用して Common Base Event 仕様に準拠したイベントを生成します。これらのイベントは、各種の機能 (例えば、状況の変更の報告、構成の変更、パフォーマンス報告、およびオブジェクトの作成や削除など) のために使用できます。これらのイベントは、XML 文書の形式で表され、その設計は定義済みのスキーマによって制御されます。それぞれのイベントには、そのイベントについて報告したコンポーネントの ID、そのイベントの影響を受けるコンポーネントの ID、およびそのイベントが生成される原因となった状態が含まれています。

Common Event Infrastructure は、これらのイベントを管理するのに使用される IBM コンポーネント・テクノロジーです。これは、Common Base Event 仕様に準拠したイベントをデータベースに保管し、関与しているリスナーにイベントのコピーを配布するサーバーを提供します。

NetView は、イベントを作成し、配布のためにそれらを Common Event Infrastructure に渡します。また、自動化のために、NetView の方が Common Event Infrastructure からイベントを受け取ることもあります。

Common Base Event の形式については、「*IBM Tivoli NetView for z/OS*カスタマイズ・ガイド」で説明されています。

Common Base Event の作成

Common Base Event の作成方法には、次の 2 種類があります。

- メッセージまたは管理サービス単位 (MSU) を自動化して Common Base Event を作成する方法。これについては、『メッセージおよび MSU の自動化による Common Base Event の作成』で説明しています。
- ハードウェア・モニター・フィルタを設定して、ハードウェア・モニターによって MSU が記録されるときに Common Base Event を作成する方法。これについては、494 ページの『ハードウェア・モニター・フィルタの設定による Common Base Event の作成』で説明しています。

メッセージおよび MSU の自動化による Common Base Event の作成

自動化テーブル CBE アクションを使用して、イベントを作成し、それを配布およびデータベースへの記録のために Common Event Infrastructure サーバーに送ることができます。CBE アクションは、EDIT アクションにたいへん似ています。これ

は、イベントの生成に使用できる EDIT パイプ・ステージ指定を受け入れます。この編集指定の鍵となる部分は CBETEMP グローバル・オーダーで、これについては「IBM Tivoli NetView for z/OS プログラミング:パイプ」で説明されています。これは、CNMSTYLE メンバーの CBE.TEMPLATES ステートメントが処理されるときに NetView プログラムによって保管される XML テンプレートをプルします。編集指定内のその他のステージでこのテンプレートを変更して、指定されたイベントを生成できます。

以下に、デフォルトのメッセージ・テンプレート *msgdefault* を使用する自動化テーブル項目のサンプルを示します。

```
IF MSGID=.something. THEN  
  CBE('CBETEMP /msgdefault/ COPY *');
```

この例では、グローバル・オーダー CBETEMP / msgdefault/ が msgdefault テンプレートを読み取ります。これは、出力として、Common Event Infrastructure サーバーに送ることができる完全な XML 文書を生成します。このオーダーがこのテンプレートを EDIT ステージへの 1 次入力とし、後続の COPY オーダーがそれを出力にコピーします。その後、結果の XML 文書がサーバーに送られます。

NetView プログラム提供の CNMSCBET テンプレート・サンプルは、メッセージ用およびアラート用の一連のテンプレートを提供します。テンプレートは、CBETEMP オーダーの処理時にランタイムで、メッセージまたはアラートからの情報がその中にプルされる変数のデータを参照できます。

ユーザー独自のテンプレートを作成することもでき、結果の XML を操作するためのより複雑な EDIT ステージを構成できます。NetView サンプル CNMSCBEA は、編集指定を使用してエレメントを追加または削除したり、テンプレートによって提供される値を変更したりする方法をいくつか示します。

管理サービス単位 (MSU) の場合は、msudefault テンプレートを使用してこれと類似の処理を実行できます。

注: (vsamcorr 拡張データ・エレメントを組み込むことによって) VSAM 関連データが含まれている場合は、ハードウェア・モニターが MSU を記録するまでイベントを完成できません。そのため、イベントは即時には生成されず、ハードウェア・モニターによる記録の実行中に生成されます。上記のような状況下では、フィルターの設定とは無関係に、イベントは 1 つのみ生成されます。VSAM 関連データが含まれていない場合は、Common Base Event が即時に作成されます。また、NPDA がフィルター設定値を使用してもう 1 つ別の Common Base Event を生成することもあります。

ハードウェア・モニター・フィルターの設定による Common Base Event の作成

ハードウェア・モニター・フィルター CBEROUTE (TECROUTE および TRAPROUT と似ています) を使用すると、自動化テーブル・ステートメントをコーディングしなくても、Common Base Event 変換用の MSU を選択できます。これらの Common Base Event は、msugeneric テンプレートまたは msunongeneric テンプレートを使用して作成されます。TECROUTE および TRAPROUT の場合と同様

に、これらのフィルターは自動化テーブルに指定できます。また、SRFILTER コマンドを使用して指定することもできます。

自動操作における Common Base Event の使用

NetView プログラムは、自動操作の目的で Common Base Event 仕様に準拠するイベントを受け取る場合もあります。NetView プログラムがイベントを受け取ると、そのイベントは複数行メッセージ XML 文書として表示されます。この XML 文書の前には、イベントが受信されたことを指示する BNH875I メッセージが表示されます。ACQUIRE 自動化テーブル条件を使用すると、この文書からの情報を比較目的で選択できます。例えば、ある Common Base Event が failure_causes という名前の拡張データ・エレメントを持っているとします。これは、自動化テーブルには以下の形式で示されます。

```
<extendedDataElements name="failure_causes" type="string">
<values>HUB_FAILURE</values>
</extendedDataElements>
```

この値を検査するための ACQUIRE 条件は、さまざまにコーディング可能です。以下に例を示します。

```
IF ACQUIRE(FINDLINE /name="failure_causes"/FINDLINE/<values>/UPTO"</values>
"parse/<word 3.*") = HUB_FAILURE" THEN
```

BNH875I メッセージには、2 つの挿入 type および msg が含まれています。type 挿入が設定されるのは、イベントが別の NetView プログラムによって生成されたものである場合のみです。このような場合、この挿入には値として MSG または MSU のいずれかが含まれています。これは、そのイベントがメッセージによって生成されたのか、MSU によって生成されたのかを示しています。イベントが NetView プログラムによって生成されたものでない場合は、この挿入は N/A に設定されます。msg 挿入には、CommonBaseEvent タグの msg 属性の値が含まれています。通常これは、イベントの高水準テキスト記述です。メッセージの場合、これはメッセージ・テキストの最初の (または唯一の) 行です。アラートの場合、これはアラート記述テキストです。この属性がイベントの一部として提供されていない場合も、この挿入は N/A に設定されます。

Common Base Event の相関処理

イベントがサーバーに送られると、381 ページの『相関エンジンを使用したメッセージおよび MSU の関連付け』で説明されている相関処理に、オプションでパススルーされる可能性があります。デフォルトでは、イベントが相関処理にパススルーされることはありません。イベントが相関処理の対象となるようにするには、値 true を指定した相関拡張データ・エレメントを組み込みます。(詳しくは、CNMSCBET サンプルを参照してください。) イベントが相関処理を受ける場合、XML 文書は相関処理でサポートされるイベント形式に変換されます。フィールド名と属性名がイベントの属性として使用されます。通常は、相関に対して値はストリング値として示されますが、いくつかのケースではその他の値 (Long や stringArray など) が使用されます。以下の表に、イベント・データと相関イベント属性との間のマッピングを示します。相関イベントの型属性の値は常に CBE となります。

Common Base Event に含まれている値のみが関連イベント属性にマップされることに注意してください。Common Base Event のフィールドと属性のほとんどはオプションであるため、これらのフィールドの多くはイベント属性としては提示されません。関連ルールではどの属性が使用可能なのかを理解するためには、関連処理の対象となるイベントの形式をよく知る必要があります。一般に、イベントから取られる関連属性は、Common Base Event のスキーマ内に定義されている大文字の属性名に対応します。ある属性名がイベント・スキーマ内で固有である場合は、その属性名は単独で使用されます。ComponentType エlement の location (reporterComponent または sourceComponent の場合に使用可能) のように固有でない属性名の場合は、REPORTERCOMPONENT.LOCATION のように、それを含んでいるElement の名前が前に付けられます。配列として定義されているElement (拡張データ・Element またはコンテキスト・データ・Element など) の場合は、配列Element の参照には、name= 属性に設定されている値の後ろにピリオド(.) を付けて、さらにその後ろに拡張データ・Element またはコンテキスト・データ・Element 内で定義されている大文字の属性名を付けたものが使用されます。name= 属性から取られる値は大文字でないことに注意してください。AssociatedEvents は name 属性を持たないため、例えば ASSOCIATEDEVENTS0.RESOLVEEVENTS のように、配列の各メンバーには数値が割り当てられます。この数値は 0 から始まり、0 は配列の最初のElement に割り当てられます。

あるイベントがある関連ルールのトリガーとなる場合、実行されるべき最も一般的なアクションは、そのイベントを Common Event Infrastructure か NetView プログラムに送ることです。どちらに送るかは、イベントが受信されたのが NetView プログラムからか NetView WebSphere® クライアントからかによって決まります。NetView プログラムは、関連処理が実行される場合にイベントを経路指定するのに使用できる 2 つのアクション CBETToCEI および CBETToNetView を提供します。

表 18. Common Base Event 関連

Common Base Event 属性	関連属性	関連の値のタイプ
Common Base Event 属性	関連属性	String
CommonBaseEvent.extensionName	EXTENSIONNAME	String
CommonBaseEvent.Version	VERSION	String
CommonBaseEvent.globalInstanceId	GLOBALINSTANCEID	String
CommonBaseEvent.localInstanceId	LOCALINSTANCEID	String
CommonBaseEvent.priority	PRIORITY	String
CommonBaseEvent.creationTime	CREATIONTIME	String
CommonBaseEvent.severity	SEVERITY	Long
CommonBaseEvent.repeatCount	REPEATCOUNT	Long
CommonBaseEvent.elapsedTime	ELAPSEDTIME	Long
CommonBaseEvent.sequenceNumber	SEQUENCENUMBER	Long
reporterComponentId.application	REPORTERCOMPONENTID.APPLICATION	String
reporterComponentId.component	REPORTERCOMPONENTID.COMPONENT	String
reporterComponentId.componentIdType	REPORTERCOMPONENTID.COMPONENTIDTYPE	String
reporterComponentId.componentType	REPORTERCOMPONENTID.COMPONENTTYPE	String
reporterComponentId.executionEnvironment	REPORTERCOMPONENTID.EXECUTIONENVIRONMENT	String
reporterComponentId.instanceId	REPORTERCOMPONENTID.INSTANCEID	String
reporterComponentId.location	REPORTERCOMPONENTID.LOCATION	String
reporterComponentId.locationType	REPORTERCOMPONENTID.LOCATIONTYPE	String
reporterComponentId.processId	REPORTERCOMPONENTID.PROCESSID	String

表 18. Common Base Event 相関 (続き)

Common Base Event 属性	相関属性	相関の値の タイプ
reporterComponentId.subComponent	REPORTERCOMPONENTID.SUBCOMPONENT	String
reporterComponentId.threadId	REPORTERCOMPONENTID.THREADID	String
sourceComponentId.application	SOURCECOMPONENTID.APPLICATION	String
sourceComponentId.component	SOURCECOMPONENTID.COMPONENT	String
sourceComponentId.componentIdType	SOURCECOMPONENTID.COMPONENTIDTYPE	String
sourceComponentId.componentType	SOURCECOMPONENTID.COMPONENTTYPE	String
sourceComponentId.executionEnvironment	SOURCECOMPONENTID.EXECUTIONENVIRONMENT	String
sourceComponentId.instanceId	SOURCECOMPONENTID.INSTANCEID	String
sourceComponentId.location	SOURCECOMPONENTID.LOCATION	String
sourceComponentId.locationType	SOURCECOMPONENTID.LOCATIONTYPE	String
sourceComponentId.processId	SOURCECOMPONENTID.PROCESSID	String
sourceComponentId.subComponent	SOURCECOMPONENTID.SUBCOMPONENT	String
sourceComponentId.threadId	SOURCECOMPONENTID.THREADID	String
AssociatedEvents[n].resolvedEvents	ASSOCIATEDEVENTS n .RESOLVEDEVENTS	String
AssociatedEvents[n].associationEngineInfo.name	ASSOCIATEDEVENTS n .ASSOCIATIONENGINEINFO.NAME	String
AssociatedEvents[n].associationEngineInfo.type	ASSOCIATEDEVENTS n .ASSOCIATIONENGINEINFO.TYPE	String
AssociatedEvents[n].associationEngineInfo.id	ASSOCIATEDEVENTS n .ASSOCIATIONENGINEINFO.ID	String
AssociatedEvents[n].associationEngine	ASSOCIATEDEVENTS n .ASSOCIATIONENGINE	String
contextname.type	contextname.TYPE	String
contextname.contextValue	contextname.CONTEXTVALUE	String
contextname.contextId	contextname.CONTEXTID	String
Any[n]	ANY n	String
MsgDataElement.MsgId	MSGID	String
MsgDataElement.MsgCatalog	MSGCATALOG	String
MsgDataElement.MsgLocale	MSGLOCALE	String
MsgDataElement.MsgCatalogType	MSGCATALOGTYPE	String
MsgDataElement.MsgCatalogId	MSGCATALOGID	String
MsgDataElement.MsgTokens	MSGTOKENS	StringArray
situation.categoryName	CATEGORYNAME	String
situation.reasoningScope	REASONINGSCOPE	String
situation.reportCategory	REPORTCATEGORY	String
situation.operationDisposition	OPERATIONDISPOSITION	String
situation.availabilityDisposition	AVAILABILITYDISPOSITION	String
situation.processingDisposition	PROCESSINGDISPOSITION	String
situation.successDisposition	SUCCESSDISPOSITION	String
situation.situationDisposition	SITUATIONDISPOSITION	String
situation.dependencyDisposition	DEPENDENCYDISPOSITION	String
situation.featureDisposition	FEATUREDISPOSITION	String
situation.situationQualifier	SITUATIONQUALIFIER	String
situation.otherAny	OTHERANY	String
extendedDataElements.type	namehierarchy.TYPE	String
extendedDataElements.values	namehierarchy.VALUES	String or StringArray

表の注

- n は、AssociatedEvents[] 配列内のエレメントを参照するのに使用される数値です。
- *contextname* は、ContextDataElement[] 配列の contextDataElement メンバーの name= 属性の値です。

- *namehierarchy* は、拡張データ・エレメントとその子を参照するのに使用される *name=* 値の階層です。例えば、あるイベントが次のような拡張データ・エレメントを持っているとします。

```
<extendedDataElements
  name=sample
  type=string
  <values>top level value</values>
  <children name=secondLevel type=string>
    <values>secondLevelValues</values>
    <children name=thirdLevel type=string>
      <values>thirdLevelValues</values>
    </children>
  </children>
</extendedDataElements>
```

この拡張データ・エレメント用に作成される相関属性は、次のとおりです。

```
sample.TYPE=string
sample.VALUES=top level value
sample.secondLevel.TYPE=string
sample.secondLevel.VALUES=secondLevelValues
sample.secondLevel.thirdLevel.TYPE=string
sample.secondLevel.thirdLevel.VALUES=thirdLevelValues
```

第 31 章 自動化オペレーション・ネットワークの使用

自動化オペレーション・ネットワーク (AON) は、ネットワークの自動化を実現するようにカスタマイズし拡張できる、包括的でドロップインのポリシー・ベースのプログラム群を提供します。以下に挙げる、AON のコンポーネントが、複数のネットワーク・プロトコル間での整合性のある自動化を実現します。

- SNA 自動化 (AON/SNA)
- TCP/IP 自動化 (AON/TCP)

これらのコンポーネントのうち 1 つ以上のコンポーネントの実行を選択することができます。

注: NetView Automated Operations Network (AON) コンポーネントの一部として以前実装されていた IP 関連機能の多くは、現在は基本 NetView サービスとして実装されています。これらの機能 (AON IP 機能と呼ばれる) では、AON の使用可能化と構成が不要となりました。代わりに、これらは IPMGT タワーを使用して使用可能にできます。ただし、AON TCP サブタワーを使用して既に使用可能になっている AON IP 機能は、再構成する必要はありません。AON TCP サブタワーおよび IPMGT タワーは同時には使用できません。IPMGT タワーについての追加情報は、「*IBM Tivoli NetView for z/OS*インストール:追加コンポーネントの構成」に記載されています。

AON コンポーネントは、ネットワーク・リソースに関する問題を示しているアラートやメッセージを代行受信します。AON のコンポーネントが、障害を起こしたりリソースを監視し、そのリカバリーを試みます。AON コンポーネントは、また、リソースの障害を記録し、再発するネットワーク障害の追跡を可能にします。AON は、本書に記述している、グローバル変数、自動化テーブル構成、およびタイマーなどの機能の多くを使用しています。AON の自動化をインプリメントすることにより、ユーザーは自動化機能の基本セットを使用できるようになります。

AON の自動化とリカバリーについて

AON により実現される自動化は、メッセージ、MSU、またはタイマーにより起動されます。メッセージと MSU は、自動化テーブルによりトラップされ、定義済みアクションを行う AON ルーチン呼び出します。これらのアクションは、制御ファイルに指定された自動化ポリシーにより決められています。AON を理解するには、自動化テーブルと制御ファイルをよく知る必要があります。AON のコンポーネントが、メッセージと MSU をトラップし、ネットワークに適切なアクションを行う、自動化ポリシーおよび自動化テーブル・ステートメントを提供します。

AON のフルスクリーン・オペレーター・インターフェースにより、AON 機能およびその他の NetView 機能に対するコマンドを発行し、応答を受信することができます。

ネットワークとリソースにはさまざまなタイプがありますが、これらのネットワークを自動化するタスクは、類似しています。AON は、類似しているタスクに対する自動化モジュールを提供します。例えば、あるリソースが、SNA または TCP/IP

ネットワークで障害を起こした場合、自動化リカバリーがリソースに対して進行する前に、リソース・タイプ、接続性、および状況など、そのリソースに関する情報を収集する必要があります。これらネットワーク・タイプのおおのほは、異なるプログラムによりリソース情報を収集しますが、自動化ドライバーは、同一の方法でこれら呼び出し、処理します。

共通自動化ドライバーを持つ利点は、信頼性の向上、習熟時間の短縮、およびネットワークの問題判別の単純化です。

自動化テーブル

自動化テーブルは、メッセージ、アラート、および解決策を検出し、これらのメッセージと MSU に基づくアクションを行います。それから、AON の障害またはリカバリー処理を適宜起動します。

制御ファイル

制御ファイルには、特定のネットワークにおいてどのように自動化が機能するかを規定する、自動化ポリシーがあります。AON のすべてのコンポーネントは、自動化パラメーターをその制御ファイルに保管します。コーディングの柔軟性のために、制御ファイル・データは、キーまたはデータ内容による検索でアクセス可能です。

制御ファイルは、NetView が初期化されるたびに、ストレージにロードされます。コマンド行またはオペレーター・インターフェースから、適当なコマンドを発行して、AON が実行中に制御ファイル・データを変更することができます。これらの制御ファイル・データの変更は、制御ファイルではなく、ストレージに対して実施されます。これらの変更を永続的なものにしたい場合は、システム・プログラマーがエディターを使用して、制御ファイルに変更を加えることができます。

自動化ポリシーは、制御ファイルに含まれていて、次の利点があります。

- サイト依存変数 (例えば、モニターしリカバリーすべきリソースの名前) を、その情報を使用するルーチンから独立して、保存できます。
- 情報は、コーディングが容易な位置パラメーターおよび **KEYWORD=値** パラメーターを使用して、ルーチンに使用される言語から独立しています。
- 設計に柔軟性があります。制御ファイルへの入力、特定リソース名でも、PU や NCP などのリソース・タイプでも可能です。リソース名またはタイプが指定されない場合は、企業全体で共通のデフォルトが使用されます。

注: 制御ファイルの入力項目についての詳細は、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

自動化操作プログラムについて

NetView の自動化タスクの設計では、並行処理を行う個別に定義された自動化操作プログラム間で、AON がそのワークロードを分割することを可能にしています。

自動化操作プログラム は、オペレーター ID を持ったプログラムであって、NetView は、このプログラムを NetView オペレーターであると認識します。自動化操作プログラムが行う多くの自動化作業には、リソースの再活動化または非活動化、追加アクションが必要な場合に他のプログラムへのメッセージと MSU の送信、および追跡目的の自動化ログへの記録の送信などがあります。 動的表示機能

(Dynamic Display Facility (DDF)) と呼ぶオペレーター状況パネルは、リアルタイムにパネルを更新して、これらの変化を反映します。各コンポーネントは、独自の自動化タスクを追加しています。

通知について

自動化の通知は、以下の 1 つ以上の応答で構成することができます。

- メッセージ
- MSU
- Event Integration Facility (EIF) イベント
- 動的表示機能 (DDF) 更新
- Automation In Progress (自動化進行) 状況などの、NetView 管理コンソール更新
- ポケットベルまたは電子メール、INFORM ポリシー定義を使った要求

これらの通知は、AON が検出またはアクションした有効なアクションを記述しています。通知は、リソースがオペレーターの介入が必要なこと、または重要なネットワーク・イベントが発生したことをオペレーターに知らせます。メッセージは、適切なオペレーター (通知オペレーター と呼ばれる) に渡され、任意指定によっては、削除されるまでコマンド機能表示装置に保留されます。

AON メッセージ通知は、メッセージ・クラスの使用法に基づいて発生します。通知オペレーターを、特定のメッセージ・クラスに対する単一の責任を持たせて定義できます。例えば、1 つ以上の通知オペレーターを、TCP/IP リソースにのみ関係するメッセージを受け取るよう定義することがあります。

さらに、通知ポリシー (Inform policy) ステートメントを定義して、ポケットベル (数字または英数字表示の) あるいは電子メールを使用する適切な要員に通知することが可能です。通知ポリシーは、カスタマイズが可能です。例えば、昼間勤務のオペレーターに電子メールを送り、夜間勤務のオペレーターにポケットベルで連絡することができます。

自動化追跡について

問題判別の生産性を向上させるため、AON は、自動化プロセスとネットワーク・リソースの状況の記録用に、次のファイルを提供しています。

状況ファイル

状況ファイルは、リソースに起こった最新の 10 の障害を、時刻を付けて記録します。状況ファイルは、現在の自動化状況を記録し、しきい値例外も記録します。プログラムが最新の障害の通知を受けると、AON は、どのプログラムがリソースに最後に実行されたのかを追跡します。

自動化ログ

自動化アクティビティの記録は、オプションの自動化ログに保管され、自動化情報を一箇所に統合しています。AON は、リソースが使用不可になり、また再び使用可能になったとき、ログに記録します。これらの記録は、アクションを起こした原因が、自動化、ヘルプ・デスク機能、またはオペ

ーターなのかどうかを示します。自動化ログに保管された情報は、自動化とレポートが使用するため、NetView ログにも記録されます。

ハードウェア・モニターでの自動化通知ログについて

AON は、そのリソース関連通知を、特定リソースの問題判別とイベントの関連付けに使用するため、ハードウェア・モニター・データベースに記録します。Alerts Dynamic / Most Recent Events 機能を使用することにより、NetView グラフィック表示装置に AON 自動化アクティビティを表示させることができます。

リソース・リカバリーとしきい値

リカバリーの基準を、個々のリソース、リソースのグループ、あるいは、グローバル・レベルについて設定することが可能です。多くのパラメーターやオプション項目があり、例えば、一定期間にカウントされるエラーの数に対するしきい値の設定、あるいは 1 日の時刻または週の曜日に基づく自動化リカバリーなどです。

リソースが、使用可能から使用不可に変化すると、メッセージまたはアラートが生成されます。AON のコンポーネントが、このメッセージまたはアラートを代行受信し、リソースに対するリカバリー・アクションを開始します。AON は、これらの障害を状況ファイルに記録します。エラーの数が、一定期間のユーザー定義の数を超えると、AON コンポーネントは、リソースのリカバリーを試み、リソースが頻度の高い、または低いエラーに遭遇していることを、適切なプログラムに知らせます。この機能により、可用性の劣化したリソースを予防的に調査する機会を得られます。

あるリソースが、可用性の問題をしばしば提示し、そのためにユーザー指定のクリティカルなしきい値を超えてしまうような場合は、反復する非生産的なリカバリーの試行を防止するために、自動化は停止されます。AON コンポーネントは、リソースの状況のチェックを継続し、制御ファイルに定義された間隔で、リカバリーを継続します。

AON のコンポーネントは、コマンドを発行し、リソースをリカバリーするより以上のことを行います。503 ページの図 169 に、AON リカバリー・ロジックの基本的な例を示します。

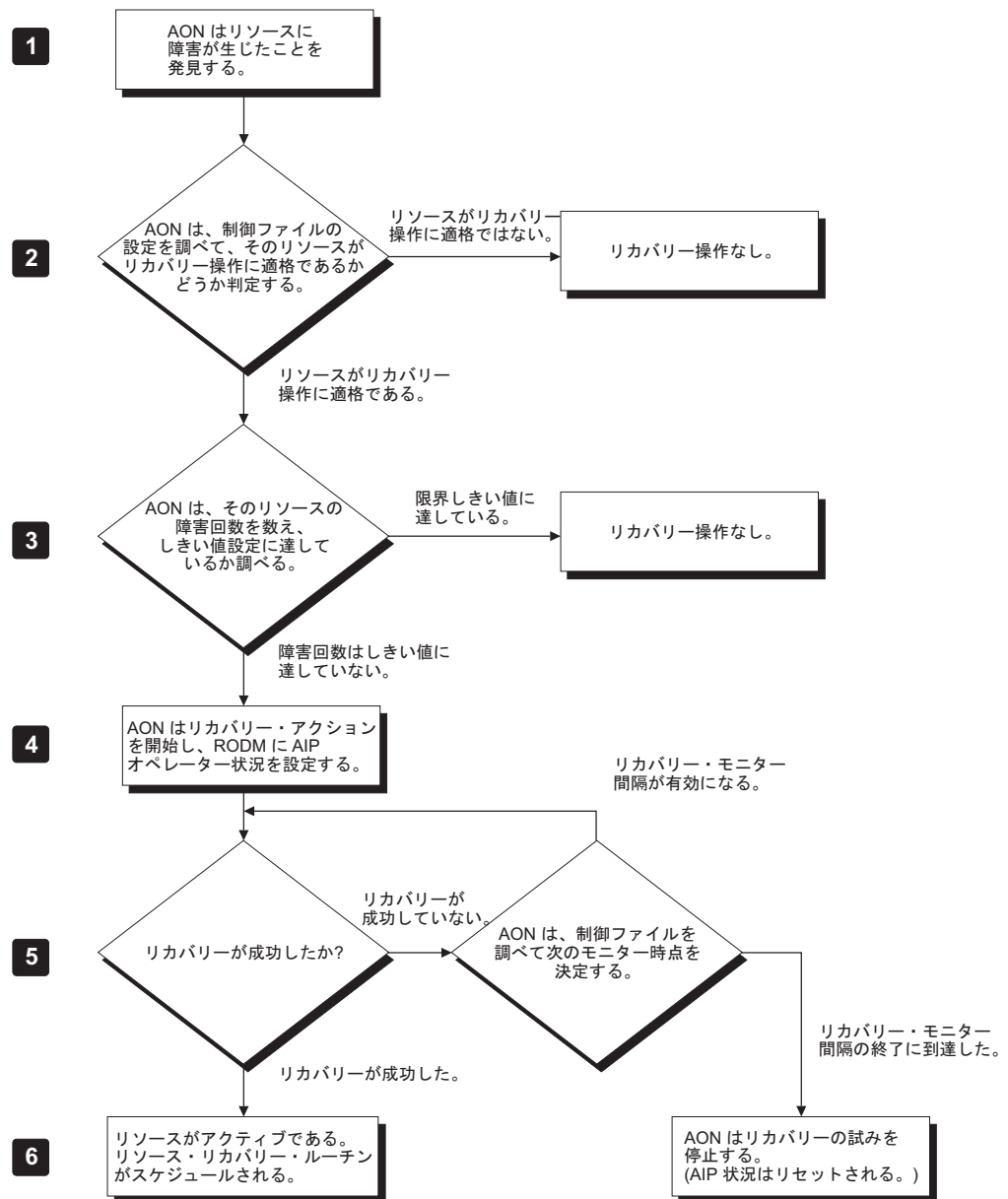


図 169. 自動化障害ロジック

ネットワーク障害が発生すると:

- 1 リソースの状況が、使用可能から使用不可に変化すると、メッセージが生成されます。AON は、このメッセージを代行受信し、リソースに対するリカバリー・アクションを開始します。
- 2 AON は、リカバリー・アクションがリソースに対して、アクティブであるかどうか判定します。リソースに対して自動化を完全に禁止に設定したり、自動化が起動しない時間をスケジュールしたりすることもできます。
- 3 AON は、しきい値により、リソース障害をオペレーターに通知するか、リカバリーの試みを継続するか判断します。3 つのしきい値を設定でき、それらは、「頻度の低い」、「頻度の高い」、「重大な」です。頻度の低いまたは頻度の高いしきい値レベルでは、AON は、リソース障害を予防的に調査する機会を与えるよう、オペレーターに通知します。障害の数が、重大しき

い値を超える場合、AON は、反復する非生産的なリカバリーの試みを防止するため、オペレーターに通知し、それ以上のリカバリーの試みを停止します。

- 4** AON は、再活動化の試みを開始します。障害を起こしたすべてのタイプのリソースに対して、AON は、制御ファイルに指定した間隔で、リソース・リカバリーをモニターします。RODM ユーザーに対しては、AON がリカバリーしているリソースに対する、自動化進行 (AIP) オペレーター状況もセットされます。
- 5** AON は、リカバリーの試みが成功したかどうかを判断します。リカバリーが成功しなかった場合は、AON は、リカバリー・アクションを継続するか停止するか判断するため、制御ファイルをチェックします。
- 6** リカバリーが成功し、リソースがアクティブです。能動モニターが定義されていれば、この時点で開始されます。

ネットワーク・リソース (例えば、SNA リソース用の回線、PU、または CDRM) の各障害ごとに、AON の自動化は、自動化スケジューリング、しきい値分析、再活動化スケジューリング、および通知指示のために、制御ファイルをチェックします。

AON/SNA 自動化

AON/SNA は、システム・ネットワーク体系 (SNA) を自動化し、次のための自動化機能を提供します。

- VTAM/SNA サブエリア・リソース・モニター
- VTAM/SNA 拡張対等通信ネットワーク機能のリソース

505 ページの図 170 に、AON/SNA が制御するネットワーク・リソースのタイプを示します。

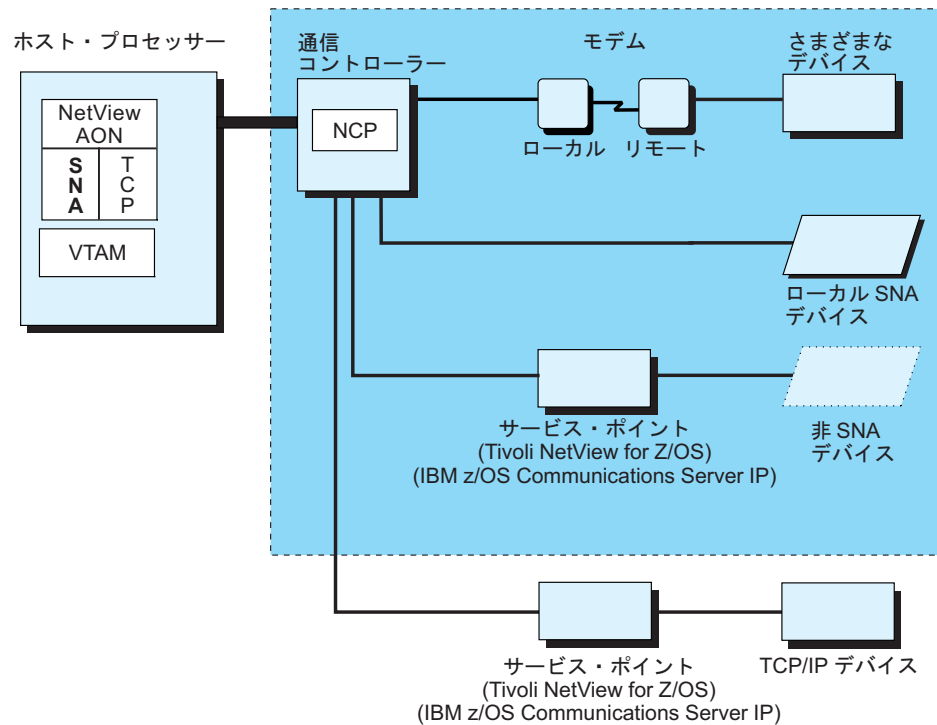


図 170. AON/SNA により自動化されるリソース

AON/SNA は、ネットワーク・リソースの障害を示す VTAM メッセージおよびアラートに基づき、ネットワーク・リカバリーを自動化します。AON/SNA は、ネットワーク・リソースの障害を示すクリティカルな VTAM メッセージおよびアラートを代行受信します。それから、AON/SNA は、障害を起こしたリソースを再活動化するコマンドを発行し、再びアクティブになるまでリソースをモニターします。ネットワークに対する管理と制御の能力により、AON/SNA は、定量化できるほどの節約を生み出します。

AON/SNA の機能のいくつかを記します。

- ・ リアルタイム状況表示
- ・ 簡明オペレーター・インターフェースによる、オペレーター生産性向上機能
- ・ 自動化ヘルプ・デスク
- ・ 状況変化に応じたリソース処理

AON/SNA のタスクを実行するには、オペレーター・インターフェース つまりパネルを使用します。オペレーター・インターフェースをバイパスするには、ユーザーのシステムにインストールされ初期化されている、NetView プログラム、AON、またはいずれの AON コンポーネントにおいても、コマンド行を使用します。また、AON/SNA リカバリーの制御をワークステーション・インターフェースからも管理できます。

AON/SNA オプションについて

AON/SNA を使用して、ユーザーのネットワークの制御を自動化できます。AON/SNA オペレーター・インターフェースを使って、オペレーターとして次のことができます。

- ・ オンライン・チュートリアルを表示

- ネットワーク上の問題の解決
- ドメイン上のリソースのリスト表示
- VTAM 始動オプションの検査と更新
- リソース状況の確認
- VTAM コマンドの発行
- 拡張対等通信ネットワーク機能のリソースのモニター
- 専用回線の管理
- X.25 スイッチド・バーチャル・サーキットのモニター
- NCP リカバリー定義の表示

以下のトピックで、これらオプションを説明します。

AON/SNA チュートリアルの使用

オンライン・チュートリアルは SNA 自動化での各オプションの概要を表示します。メニュー・パネルと、これらのオプションを使用するための具体的な説明があります。

AON/SNA ヘルプ・デスクの使用

AON/SNA には、SNA 固有の自動化ヘルプ・デスク・コンポーネントが用意されていて、経験の浅いヘルプ・デスク・オペレーターにネットワーク障害の解決を可能にしています。SNA ヘルプ・デスクを使用するには、ユーザーのターミナル ID が必要なだけです。適切なパネルにターミナル ID を入力すると、ユーザーのターミナルがどのようにシステムに接続されているかを示すピクチャーを SNA ヘルプ・デスクが表示します。

注: NetView アクセス・サービスがシステムにインストールされている場合は、SNA ヘルプ・デスクが、ユーザー ID を使用してネットワーク障害のある場所を判断します。この場合は、ターミナル ID を知る必要はありません。

SNA ヘルプ・デスクは、問題判別手順を自動化します。ネットワーク障害を解決するには、SNA ヘルプ・デスクの問題判別パネル上のリカバリー手順のリストから、アクションを選択することができます。ヘルプ・デスクの操作性向上のために、SNA ヘルプ・デスクには、以下の特長があります。

- ユーザーの入力量を減らします。
- 問題判別を自動化します。
- ネットワーク構成を知らない場合でも、直ちに生産性を向上させます。
- ネットワーク障害を解決しながら、問題判別のスキルを学べます。

SNA ヘルプ・デスクにより、リソースおよびそれが接続されている上位ノードを表示できます。SNAMAP では、リソースを表示して、それに接続された下位ノードにズームできます。

SNAMAP の使用

AON/TCP は、SNAMAP と呼ぶツールを備えていて、特定のドメイン上のリソースをリストできます。リソースのリストを作成するには、リソース・タイプを選択す

るか、または特定のリソース名を入力してください。カテゴリを選択すると、AON/SNA は、そのカテゴリに合ったリソース名を示すリスト・パネルを表示します。

SNAMAP は、接続されている下位ノードにズームできます。対照的に、SNA ヘルプ・デスクは、リソースとそれが接続された上位ノードの表示を行います。

VTAM オプションの管理

VTAM オプションの管理機能は、ドメインに対する、現行およびデフォルトの VTAM 始動オプションを表示します。このオプション・リストの上をスクロールすることができ、希望する設定の所にカーソルを移動し、表示された情報に上書きして変更することができます。Enter キーを押した後、AON/SNA が、変更の処理を行います。

NetStat の使用

NetStat は、リソース・タイプ、表示したい状況、およびリカバリー・フラグを設定したいかどうかを選定することで作成される、リソース・リストを表示します。NetStat は、この情報を使用して、選定した基準に合うリソースのリストを表示します。このオプションは、表示のみのオプションです。

VTAM コマンドの発行

VTAM コマンド・オプションは、パネルからの VTAM コマンドの発行を可能にし、続くパネルで、コマンドの結果を確認できるようにします。発行したコマンドの結果が、2 つ以上のパネルに表示される場合、パネルをスクロールして、すべての情報を見ることができます。VTAM コマンド・オプションにより、ユーザー・タスク・セッション間でのコマンド発行が節約できます。VTAM コマンド・パネルにコマンドを入力すると、そのコマンドは、同一セッションでまたは後のセッションで、そのパネルに戻ってきたときに、パネルの同一箇所に表示されます。

VTAM コマンド・オプションは、カーソル位置を検知しています。AON/SNA は、カーソルの位置にあるコマンドを発行します。コマンドを発行するには、パネル上に新規のコマンドをタイプするか、カーソルを以前に入力したコマンドの所に移動し、Enter キーを押します。

X.25 スイッチド・バーチャル・サーキットのモニター

AON/SNA は、X.25 スイッチド・バーチャル・サーキット (SVC) を構成する、SNA 物理リソースのモニターを含んでいます。オペレーター・インターフェースにより、AON/SNA がモニターしている SVC リンクの、追加、変更、削除が可能です。

X.25 コマンドは、LUDR プール管理も含んでいます。この機能は、NCP に対して使用可能な論理装置の数を決定するメカニズムを持っています。AON/SNA は、この数をモニターでき、定義済みのしきい値を下回るようになると、オペレーターにアラートを出します。

NCP リカバリー定義の表示

特定の NCP または定義済みのすべての NCP の、NCPRECOV 制御ファイル定義を表示できます。

AON/SNA サブエリア VTAM リソース自動化サポート

AON/SNA は、ネットワークにおける変化を示す VTAM メッセージをトラップして、SNA サブエリア・ネットワークをサポートします。これには、リソースの状況変化、ストレージ障害、およびオペレーターのアクションを必要とする未解決の応答が含まれます。管理されるリソースには、NCP、回線、物理装置、ドメイン間リソース管理プログラム (CDRM)、ドメイン間リソース (CDRSC)、アプリケーション (ACB)、およびセッションが含まれます。

以下のものにより、AON/SNA サブエリア・リソースをモニターできます。

- 受動モニター
- リカバリー・モニター
- 事前の対策を講じたモニター

受動モニターは、VTAM メッセージが示す AON/SNA リソースに障害があるときに、AON/SNA に通知をさせたい場合に生じます。リカバリー・モニターは、MONIT 制御ファイル項目に定義した間隔での、使用不可リソースの検査と、それをリカバリーしようとする試みから成ります。能動モニターは、重要なネットワーク装置に関し、AON/SNA に定期的にモニターし、レポートさせたい場合に使用します。

拡張対等通信ネットワーク機能のリソースのモニター

AON/SNA 拡張対等通信ネットワーク機能は、クライアント/サーバーおよび分散アプリケーションのための、ネットワークでの強力な柔軟なソリューションです。AON/SNA 拡張対等通信ネットワーク機能は、拡張対等通信ネットワーク機能のリソース (制御点および拡張対等通信ネットワーク機能のリソース間のセッションも含む) を対象に、能動モニターを提供します。AON/SNA 拡張対等通信ネットワーク機能は、共通の VTAM 拡張対等通信ネットワーク機能用のオペレーター・インターフェースを提供します。

AON/SNA X.25 モニター・サポート

AON/SNA は、AON/SNA X.25 スイッチド・バーチャル・サーキット (SVC) を構成する、SNA 物理リソースのモニターを含んでいます。オペレーター・インターフェースにより、AON/SNA がモニターしている、スイッチド・バーチャル・サーキット (SVC) リンクの追加、変更、削除が可能です。

交換回線接続は、フルスクリーン・パネルを使用して、モニターされます。モニター回線に関して、接続または切断がある度に、パネルが更新されます。

AON/SNA X.25 コマンドもまた、LUDRPOOL 管理を含んでいます。この機能は、ネットワーク制御プログラム (NCP) に対して、使用可能な論理装置の数を決定するメカニズムを持っています。AON/SNA は、この数をモニターでき、アラートが定義済みのしきい値を下回るようになると、オペレーターにアラートを出します。

X.25 NCP Packet Support Interface (NPSI) のユーザーは、ハードウェア・モニターからネットワーク・イベント情報を入手できます。AON/SNA は、通信ネットワーク管理 (CNM) インターフェースのユーザー出口を提供します。このユーザー出口は、NPSI から入ってくるアラートを抑止し、NPSI により得られる元のデータを含む BNJ146I メッセージを生成します。このメッセージは、アラートでエラー・バイ

トおよびその他の情報をスキャンし、エラー・バイトの意味を探し、GENALERT コマンドによる改良したアラートを送信するプログラムを起動するよう、AON/SNA にプロンプトを出します。ハードウェア・モニターのこのアラートは、オペレーターに、特定のアクションを含む、明快なエラーの解釈を伝えます。AON/SNA は NetView ログにメッセージを挿入します。このメッセージは、VTAM が生成した INOP メッセージと、対応する NPSI アラートを関連付けます。また、ハードウェア・モニターが不正確な XID 障害を検出する度に、アラートがモニターに送られます。

AON/TCP の自動化

AON/TCP は、ユーザーの TCP/IP ネットワークの管理を支援します。NetView プログラムと TCP/IP ネットワークとの間の接続は、z/OS TCP/IP です。

AON/TCP 機能の主なものは以下の通りです。

- AON/TCP は TCP/IP リソース障害を検出し、対応し、NetView オペレーターに通知します。これらのネットワーク障害の検出に、AON/TCP に対し、受動モニター、能動モニター、またはその両方を使用するように指示できます。
- AON/TCP は、リソース障害処理の間、TCP/IP ネットワークでネーム・サーバー障害の検出のため ping リソースを使用します。ネーム・サーバーが故障していると、AON/TCP がオペレーターに通知し、それにより、適切なネーム・サーバーを TCP/IP ネットワークに再接続できます。
- AON/TCP により、インターフェースをルーターと関連付けできます。ルーター・インターフェースが障害を起こすと、AON/TCP はオペレーターに、ルーターおよびルーター・インターフェース (リンク) の両方が十分な動作状態にないことを知らせます。
- 動的表示機能 (DDF) およびオペレーター・インターフェースを使用すると、オペレーターは、TCP/IP ネットワークを例外ベースで、つまりアクティブになっている、正しく動作しているリソースから選別する必要なく、問題のあるリソースのみを調べ処理することで、管理できます。
- 大規模な TCP/IP ネットワークでは、AON/TCP がモニターする障害とリソースのタイプを限定できます。AON 制御ファイルで定義を使用すると、ネットワークの要件が変化するにともない、AON/TCP オペレーター・インターフェースを使って、これらの定義を変更できます。

AON/TCP を使用すると、以下のことが可能になります。

- 複数の TCP/IP スタックにまたがる IP 接続を管理する。
- 停止したセッション (TN3270 および FTP) で問題判別を実施し、適切なアクションをとる。例えば、セッションを取り除くことができます。
- GET および SET などの SNMP 要求を発行し、IP リソースを管理する。
- TCP/IP リソースを IPMAN を使用して管理する。
- MIB ポーリングおよびしきい値分けを選択した IP リソース上で実行する。
- IP インターフェースとホスト・トラップを関連付ける。
- 障害のある IP ソケットをモニターおよびリカバリーする。
- UNIX および TSO 環境の両方をサポートする (並行して)。
- 下記の IP トレースの開始および停止のサポートを提供する。

- CTRACE
- PKTTRACE
- OSATRACE
- SNMP を使用して下記の管理を行う。
 - ユーザーの IP スタック
 - リモート・デバイス

能動モニターは、制御ファイルに定義された z/OS TCP/IP スタックごとに、自動的に始動します。

以下のセクションでは、AON/TCP の機能をより詳細に説明します。

能動モニター

ネットワーク上のデバイスについて連続的にモニターし、レポートすることを能動モニターといいます。

能動的にリソースをモニターするには、AON/TCP 制御ファイルに定義を置きます。AON/TCP は、PING コマンドまたは SNMP GET 要求 (z/OS のみ) を発行し、AON/TCP の初期化時および事前定義された時間間隔で、リソースの使用可能状況をチェックします。

リソースが使用不可である (制御ファイルに定義した状況が、現行の状況と合わない) 場合、AON は、障害を記録し、操作プログラムを更新し、リカバリー・モニターを管理します。障害発生時は、リカバリー・モニターが有効になるので能動モニターは必要ありません。したがって、AON は、あるリソースが使用不可であると分かったときは、常に能動モニターを中断します。リソースが使用可能に戻ったとき、能動モニターのタイマーが復活します。AON/TCP は、障害が、能動モニターかまたは受動モニターかいずれによって検出されているかを知らせるメッセージを発行します。これにより、受動モニターで十分である場合、能動モニターのオーバーヘッドを削減することができます。

リカバリー・モニター

リカバリー・モニターは、AON 制御ファイル項目に定義した間隔での、使用不可リソースの状況の検査から成ります。

AON/TCP は、リソースの障害を検出すると、常にリカバリー・モニターを開始します。このモニターは、AON/TCP 制御ファイルに定義した間隔で続きます。この間隔は、通常不規則な配分になるよう定義する必要があります。というのは、障害の初期、つまりリカバリーが起こる可能性の高いときに、より頻度高くモニターするのが適切だからです。障害がより長く続く時は、そのリソースのリカバリーのモニターは、より低い頻度でもよいでしょう。リカバリーが検出されると、AON 自動リカバリーが開始します。AON は、操作プログラムを更新し、リカバリー・モニターを停止し、制御ファイルに定義された能動モニターを復活します。

MIB ポーリングおよびしきい値分け (z/OS TCP/IP の場合のみ)

AON 能動モニターを使用して、PING コマンドの代わりに、SNMP 要求を処理することができます。SNMP 能動モニターには、MIB ポーリング機能が用意されています。MIB ポーリングでは、モニターされている装置のインターフェース・テーブル

ルを照会します。1 つ以上のインターフェースに誤った状況がある場合、AON/TCP がその通知を送信します。追加の処理のために、ユーザー出口ルーチンをコーディングすることができます。

MIB しきい値分けが可能なのは、ある装置のモニターが能動的に行われている間です。MIB しきい値分けでは、ユーザーが定義している MIB 変数を照会し、それらの予測値を実際の値と比較します。AON/TCP が行う比較の条件は、より小、より小か等しい、等しい、より大か等しい、およびより大です。しきい値を超えていると、AON/TCP は通知を送信します。追加の処理のために、ユーザー出口ルーチンをコーディングすることができます。

例としてユーザーは、ルーターに対して MIB ポーリングとしきい値分け機能を使用できます。MIB ポーリングは、そのルーター上で失敗したインターフェースを検出します。MIB しきい値分けを使用することによって、パフォーマンス関連の MIB 変数とそれらのしきい値を定義することができます。AON/TCP が装置と通信不可能な状態にならない限り、能動モニターは継続することになります。

能動モニター設定の詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

オペレーターの注意

オペレーターは、ネットワーク例外条件に関する更新を受け取るに当たり、複数の AON/TCP インターフェースから選ぶことができます。

- オペレーターを通知プログラムと定義します。これにより、すべての例外条件に対するメッセージを受け取れます。これらのメッセージは、その画面を最新表示するか、DM コマンドを使用して、消去できます。
- DDF (動的表示機能) にアクセスします。そこでは、すべての例外条件は、重大度に基づき色分けされていて、時間別、重大度別、およびリソース・タイプ別に編成されています。注意を必要とする TCP/IP リソースのみを迅速に見いだし、リアルタイムの更新を受け取ることができます。DDF 表示パネルには、各ファンクション・キーがコマンドに設定されていて、ユーザーはカーソルをリソースに移動し、適切なファンクション・キーを押すことによってそのコマンドを発行できます。
- 問題とするリソース (とりわけ、リソースが、現在使用可能である場合) に関するメッセージを、NetView ログまたは AON 自動化ログから表示する。
- 特定のリソースに対するデータが必要な場合、AON AUTOVIEW コマンドを使用して、リソースに関して AON/TCP が認識しているすべてデータの要約を表示します。

これには、制御ファイル・データ、状況ファイル・データ、DDF データ、および PING の結果が含まれます。このパネルには、コマンド付きのポップアップ・ウィンドウがあり、オペレーターはこのコマンドを使用して、自動化の設定値を変更したり、PING を送信したり、このリソースにだけ適用できる NetView ログ・データを表示したり、モニター状態を変更したりすることができます。z/OS TCP/IP では、ユーザー・ネットワークの管理に役立つ IPMAN コマンドが使用できます。

- あるリソースに関するすべてのメッセージを、ハードウェア・モニターに送信します。

これらのメッセージを、ハードウェア・モニターまたは NetView グラフィック表示装置に表示できます。

AON 機能およびカスタマイズ機能に関する詳細は、「*IBM Tivoli NetView for z/OS Automated Operations Network* ユーザーズ・ガイド」を参照してください。

第 32 章 単一システムでの複数 NetView プログラムの実行

単一システムで複数の NetView プログラムを実行させることができます。機能を分離するために、複数の NetView プログラムを単一システムにインストールすると、以下のことができます。

- システム・サポートとネットワーク・サポートとを切り離すことができます。例えば、1 つの NetView プログラムでシステムの操作と自動化を処理し、別の NetView プログラムでネットワークの操作、自動化、および問題判別を処理することができます。
- 自動化と問題判別とを切り離すことができます。この場合、自動化 NetView プログラムには、システム自動化とネットワーク自動化の 2 つがあります。ネットワーク管理アクティビティーの影響により自動化パフォーマンスを低下させたくない場合には、自動化 NetView プログラムが自動化するどのタスクよりも自動化 NetView の実行優先順位を高くし、さらに、問題判別を実行する NetView プログラムについては、サブシステムまたはネットワーク自動化操作と矛盾しないように、スケジュールする必要があります。

単一システムで 2 つの NetView プログラムが稼働していると、それぞれの NetView プログラムは、互いにモニターしあい、リカバリーしあうことができます。1 つの NetView プログラムが、重要なリソース (コマンド・リスト・データ・セットまたはネットワーク・ログなど) の制約により制限される場合には、2 番目の NetView プログラムがあると便利です。ただし、この場合、CPU 使用率は低下しません。

1 つの NetView リリースから他のリリースへマイグレーションするとき、1 つの NetView プログラムを実動システムとして使用しながら、2 番目の NetView プログラムを使用して、新しい NetView リリースをテストすることができます。

複数 NetView プログラムの実行要件:

- ライブラリーおよびログの追加コピーの保守
- NetView プログラム間のメッセージの経路指定
- メッセージが 2 度以上自動化されないことの保証
- 各 NetView サブシステムでの、異なる指定文字の使用

追加の考慮事項:

- システム当たり、または VTAM プログラム当たり、1 つの NetView プログラムでしか使用できない NetView 機能があります。
 - VTAM が 1 つしかない場合、プログラム・オペレーター・インターフェース (POI) を一度に使用できる NetView プログラムは 1 つだけです。
 - 特定の VTAM プログラムから送られてくる非送信請求ネットワーク管理データを受信するために CNMI を使用できる NetView プログラムは 1 つのみです。さらに、VTAM 状況モニター・パフォーマンス強化を使用できる NetView プログラムも 1 つのみです。

- ハードウェア・モニター・ローカル装置を使用できるのは、システム内の 1 つの NetView プログラムのみです。
- LPAR 内でアクティブにすることができるのは、1 つのコマンド改訂テーブル (CRT) のみです。CRT をロードすると、以前のテーブルは必ず非アクティブ化されます。これは、以前のテーブルが NetView プログラムの異なるインスタンスによってロードされた場合でも同じです。
- 各 NetView プログラムでデータベースを別々に管理しているので、例えば、2 つの NetView プログラムのオペレーターは同じネットワーク・ログをブラウズしません。

NetView 機能の制約事項についての詳細は、515 ページの『NetView インターフェースと機能』を参照してください。

複数の NetView プログラムを実行させる場合には、主記憶域が余計に必要になります。システムが小さい場合には、2 つの NetView プログラムが実行するために必要な主記憶容量が不足する場合があります。また、複数の NetView プログラムを使用すると、NetView の CPU 使用率は高くなります。

複数の NetView プログラムのインストール

自動化環境に複数 NetView プログラムをインストールする際には、次の点を考慮してください。

- 1 つの NetView プログラムに、すべての NetView ライブラリーおよびデータ・セットが必要であるとは限りません。しかし、問題を回避したり、リカバリーが必要な状況が発生した場合 (つまり、ある NetView プログラムが別の NetView プログラムの機能を引き継ぐ場合) に使用できるように、NetView 開始プロシージャにはすべてのライブラリーとデータ・セットを保持させておいてください。
- NetView プログラムが正しく初期設定できるように、NetView プロシージャには、オンライン・ヘルプ・パネル、VTAM ファイル、ハードウェア・モニター・パネル、およびハードウェア・モニター・カラー・マップを組み込んでください。
- 複数の NetView プログラムは、NetView データベースを共用することができません。各 NetView プログラムには、それぞれが使用する NetView 機能 (およびリカバリー処理時に受け継ぐ可能性がある NetView 機能) に、固有のデータベースを用意する必要があります。
 - NetView プログラムがアラートを記録するか、ハードウェア・モニターにアラートを表示する場合には、ハードウェア・モニターおよびネットワーク・ログのデータ・セットが必要です。
 - NetView プログラムがアラートの記録も表示もしない場合には、ネットワーク・ログ・データ・セットだけが必要です。
- 1 つのシステムの NetView プログラムは、すべての DSIPARM メンバーを共用することができます。NetView ロード・ライブラリーおよびパネル・データ・セットも、2 つの NetView プログラム間で共用することができます。

NetView インターフェースと機能

1 つのオペレーティング・システム、または 1 つの VTAM プログラムでは、1 つの NetView プログラムにしか使用できない NetView インターフェースや NetView 機能があります。それ以外の NetView インターフェースまたは NetView 機能は、単一システム内の複数の NetView プログラムに使用可能ですが、特殊なセットアップが必要です。以下のセクションでは、複数の NetView 環境での NetView 機能および NetView インターフェースに関する考慮事項について説明します。

プログラム・オペレーター・インターフェース (POI)

NetView プログラムは、POI を介して、非送信請求 VTAM メッセージを受信します。非送信請求 VTAM メッセージは、ネットワーク自動化に使用することができます。VTAM プログラムが 1 つしかない場合、一度に 1 つの NetView プログラムしか POI を使用することができません。POI を使用するためには、VTAM APPL ステートメントで、NetView PPT (*domid*PPT) 用に AUTH=PPO を指定しなければなりません。

1 つの VTAM プログラムで一度に POI を使用できるのは、1 つの NetView プログラムのみですが、他の NetView プログラムを 2 次プログラム・オペレーター (SPO) として定義することができます。SPO として定義される NetView プログラムは、VTAM コマンドを処理し、VTAM プログラムから送信請求メッセージを受信します (例えば、コマンドに対する応答)。これらの NetView プログラムの場合には、VTAM APPL ステートメントで NetView PPT (*domid*PPT) 用に AUTH=SPO を指定してください。SPO NetView プログラムで非送信請求 VTAM メッセージを自動化したい場合には、PPO NetView プログラムからメッセージを送信してください。

アクティブの NetView プログラムが VTAM の 1 次プログラム式オペレーター (PPO) である場合、VTAM はすべての非送信請求メッセージをその NetView プログラムにのみ送信します。サブシステム・インターフェースで非送信請求 VTAM メッセージを使用することはできません。

PPO がアクティブでないか、またはすべての NetView プログラムに AUTH=SPO が指定されている場合、非送信請求 VTAM メッセージはオペレーティング・システム・コンソールに渡されます。この場合、サブシステム・インターフェースを使用しているすべての NetView プログラムは、サブシステム・インターフェースを介してこれらのメッセージを使用することができます。このとき、VTAM メッセージを自動的に 2 回以上発生させないようにしてください。

V3R3 より前のリリースの VTAM プログラムでは、NetView 状況モニターは POI を使用して状況情報を入手しています。この場合、正確な状況情報を取り出すことができるのは、POI を使用している NetView プログラムだけです。

通信ネットワーク管理インターフェース (CNMI)

CNMI を使用して指定した VTAM から非送信請求ネットワーク管理データを受信することができるのは、1 つの NetView プログラムのみです。この制限は、VTAM グローバル経路指定テーブルの指定により制御されます。このテーブルは、以下の各ネットワーク管理データをどの NetView タスクに送信するかを指定しています。

- ネットワーク管理ベクトル・トランスポート (NMVT) (アラート、セッション終了時の非送信請求 RTM データ)
- RECFMS
- その他の CNM データ・レコード

上記のネットワーク管理データのすべてのタイプが CNM ルーターである DSICRTR により受信されるように指定しなければなりません (サンプル VTAM 経路指定テーブル ISTMGC00 に指定されているとおり)。指定すると、DSICRTR は、該当する NetView タスク (セッション・モニターまたはハードウェア・モニター) へデータを配布します。

複数の NetView プログラムが存在する環境では、CNMI を使用する NetView プログラムが使用できる非送信請求ネットワーク管理データは問題判別に使用されるため、この NetView プログラムは問題判別 NetView プログラムと呼ばれることがあります。アラートは CNMI を介して渡されるため、ネットワーク・アラート・メッセージは、CNMI を使用する NetView プログラムでのみ生成されます。これらのアラート・メッセージを自動化するには、CNMI NetView プログラムで自動化するか、または別の NetView プログラムに経路指定して自動化するかのいずれかです。

CNMI を使用する NetView (問題判別 NetView プログラム) は、以下のアクションを実行します。

- CNM ルーター (DSICRTR) および従来のネットワーク管理タスク (セッション・モニター、ハードウェア・モニター、および 4700 サポート機能タスク) を開始します。
- DSICRTR の VTAM APPL を開始させ、ネットワーク管理タスクに VTAM APPL を開始させます。
- ネットワーク管理機能に関連付けられている NetView データベースにアクセスします。
- VTAM CP 名にアドレス指定された MDS-MU を受け取ります。

システム・アラートを処理するために必要であれば、CNMI を使用しない NetView プログラムは DSICRTR を開始し、それ自体のハードウェア・モニター・データベースを使用することができます (518 ページの『GENALERT』を参照)、DSICRTR の VTAM APPL はアクティブにしてはいけません。

注: CNMI をある NetView プログラムから別の NetView プログラムに変更させるためには、従来のネットワーク管理タスクおよびその VTAM APPL を停止させる必要があります。

ハードウェア・モニターのローカル装置インターフェース

オペレーティング・システムからローカル装置レコードを受け取ることができる NetView プログラムは、1 つのシステム内に 1 つだけです。これらのローカル装置レコードには、以下に示すようなシステム形式のアラート・レコードがあります。

- 外部記録 (OBR)
- 各種データ・レコード (MDR)
- 機械チェック・ハンドラー (MCH)
- チャンネル・リカバリー・ワード (CRW) レコード
- 2 次レベル割り込みハンドラー (SLIH) レコード

これらのレコードを自動化する場合には、ハードウェア・モニター・ローカル装置レコードを受け取る NetView プログラムで自動化するか、初めから他の NetView プログラムに経路指定しなければなりません。エラーに関連するメッセージは、MVS サブシステム・インターフェースを使用しているすべての NetView プログラムで使用することができます。

ハードウェア・モニター・ローカル装置レコードを受信する NetView プログラムの場合には、次の操作が必要です。

- 開始プロシージャに PGM=BNJLINTX を使用する。
- BNJMNPDА をアクティブにする。

ハードウェア・モニター・ローカル装置レコードを受信しない NetView プログラムの場合には、次の操作が必要です。

- 開始プロシージャに PGM=DSIMNT を使用する。
- BNJMNPDА をアクティブにしない。

注: ハードウェア・モニター・ローカル装置レコードの受信先を、ある NetView プログラムから別の NetView プログラムに変更するためには、両方の NetView プログラムを停止する必要があります。

MVS サブシステム・インターフェース

サブシステム・インターフェースは、システム・メッセージの受信とシステム・コマンドの入力に使用されます。拡張複数コンソール・サポート (EMCS) コンソールでは、サブシステム・インターフェースは、コマンドの受信に使用され、メッセージの受信には使用されません。単一システム内で、複数の NetView プログラムがサブシステム・インターフェースを使用することができます。サブシステム・インターフェースを使用するかどうかはオプションです。したがって、システム・メッセージを受信する必要がない、あるいは NetView プログラムからシステム・コマンドを入力する必要がないときには、NetView プログラムにサブシステム・インターフェースを使用させる必要はありません。

サブシステム・インターフェースを使用する各 NetView プログラムには、次のものがが必要です。

- NetView サブシステム・アドレス・スペースと NetView アプリケーション・アドレス・スペースの 2 つの NetView アドレス・スペース。アドレス・スペースの各ペアの始動プロシージャの先頭の 4 文字 (サブシステム名) は同一でなければなりません。サブシステム名は、同じシステム内の他の NetView アドレス・スペースのペアのサブシステム名と異ならなければなりません。NetView サブシステム・アドレス・スペースは、MVS システム・メッセージの受信とシステム・コマンドの入力に使用されます。また、NetView プログラム間インターフェースにも使用されます。
- アクティブな CNMCSSIR タスク。
- MVS コマンドを出すオペレーターまたは自動タスクのサブシステム割り振り可能コンソール。MVS システム・メッセージの受信に EMCS コンソールを使用しない場合には、この手順が必要です。
- MVS コンソールから入力する NetView コマンドの先頭に使用される固有な指定文字。指定文字は CNMSTYLE メンバーの MVSPARM.Cmd.Designator ステートメント上で指定します。デフォルトの指定文字は、パーセント記号 (%) です。

単一システム内で複数の NetView プログラムがサブシステム・インターフェースを通して稼働する場合、これらの NetView プログラムは、オペレーター宛メッセージ (WTO) を出すと、サブシステム・インターフェースを介して互いに通信することができます。メッセージを自動的に出せるのは、1 つの NetView プログラムのみであることを確認してください。

注: サブシステム・インターフェースの使用は、NetView サブシステム・アドレス・スペースの開始および停止ではなく、NetView CNMCSSIR タスクの開始および停止で制御してください。

GENALERT

GENALERT を使用してアラートを生成する場合には、VTAM CNMI を必要としません。したがって、単一システムでは、複数の NetView プログラムがアラートを生成することができます。NetView プログラムに CNMI がなくても、アラート操作処理に使用する独自のハードウェア・モニター・データベースを持つことができます。アラートを使用して、まだ自動的に処理されていない問題に関する情報を表示するには、このハードウェア・モニター・データベースが役立ちます。詳しくは、「413 ページの『ハードウェア・モニターによるアラートのモニター』」を参照してください。

GENALERT インターフェースを使用する NetView プログラムには、次の条件が必要です。

- 固有のハードウェア・モニター・データベース
- アクティブ BNJDSERV タスク
- アクティブ DSICRTR タスク
- NetView プログラムが CNMI を使用する場合:
 - BNJMBDST の DSTINIT ステートメントに FUNCT=BOTH を指定してください。
 - DSICRTTD の DSTINIT ステートメントに FUNCT=CNMI を指定してください。
 - DSICRTR および BNJHWMON の VTAM アプリケーションをアクティブにしてください。
- NetView プログラムが CNMI を使用しない場合:
 - BNJMBDST の DSTINIT ステートメントに FUNCT=VSAM を指定してください。
 - DSICRTTD の DSTINIT ステートメントに FUNCT=OTHER を指定してください。これには、プログラム一時修正 (PTF) UY25868 が必要です。
 - DSICRTR および BNJHWMON の VTAM アプリケーションをアクティブにしないでください。

NetView プログラムが GENALERT もハードウェア・モニターも使用しない場合には、BNJDSERV タスクをアクティブにしたり、そのタスクに VTAM APPL 定義も NetView データベースも関連付ける必要はありません。しかし、セッション・モニターが使用するため DSICRTR タスクは必要です。

状況モニターおよびログ・ブラウズ

VTAM 状況モニター・インターフェースを使用できる NetView プログラムは 1 つのみです。

DSICNM に O SECSTAT ステートメントを使用し、2 番目の状況モニターを指定することができます。これによって、NetView プログラムは、VTAM 状況モニター・パフォーマンス強化を使わないようになります。このようにしないと、一番始めに開始した NetView プログラムが VTAM 状況モニター・パフォーマンス強化を使用することになります。

VTAM 状況モニター・パフォーマンス強化に関する詳細については、「*IBM Tivoli NetView for z/OS* インストール: 概説」を参照してください。

インターフェースの使用

1 つのシステムで 2 つまたはそれ以上の NetView プログラムを実行させるには、本セクションで説明する手法を使用してください。

NetView プログラム間では、次の 2 種類のいずれかの方法で機能を分割します。

- ネットワーク機能対システム機能
- 問題判別対自動化

ネットワーク機能とシステム機能の分離

システム機能からネットワーク機能を分離することによって、編成を効率化できます。システム・サポートとネットワーク・サポートを別々のグループに分けると、グループごとに別の NetView プログラムを使うことでグループの独立性が保たれます。

1 つの NetView プログラムでシステムの操作と自動化を処理し、もう 1 つの NetView プログラムでネットワークの操作、自動化、および問題判別を処理します。この場合、システム自動化 NetView プログラムは、サブシステム・インターフェースを使用します。オプションで、システム自動化 NetView プログラムは、GENALERT インターフェースを備えることができます。ネットワーク NetView プログラムは、POI、CNMI、およびハードウェア・モニター・ローカル装置インターフェースを使用します。オプションで、ネットワーク自動化 NetView プログラムもサブシステム・インターフェースと GENALERT を使用することができます。

このアプローチの利点は、

- ネットワーク・サポートとシステム・サポートのそれぞれに自動化テーブルがある。
- ネットワーク・メッセージ、ネットワーク・アラート、およびローカル・アラートを別の NetView プログラムに送信して自動化する必要がある。

このアプローチの不利な点は、

- エラーの状況によっては、ネットワークの自動化とシステムの自動化の調整が必要である。
- ネットワーク自動化操作が、NetView タスク優先順位に従って、問題判別 NetView プログラムの問題判別機能との間でリソースを求めて競争する。

- ログ間でデータを関連付けなければならない場合がある。
- パフォーマンスを向上させるために、ネットワークに障害が発生している最中に、ネットワーク管理タスクをシャットダウンするためのプロシージャが必要な場合がある。

問題判別機能と自動化機能の分離

問題判別などの従来のネットワーク管理機能を自動化機能と分離すると、パフォーマンスを向上させることができます。ネットワーク管理アクティビティーの影響を自動化のパフォーマンスに与えたくないときには、これらの機能を別々の NetView に分離します。これにより、自動化 NetView プログラムは、自動化される他の任意のタスクより高い優先順位で実行され、ネットワーク管理および問題判別 NetView プログラムは、サブシステムおよびアプリケーションの処理に支障をきたさない優先順位で実行することができます。自動化 NetView プログラムは、システムの自動化とネットワークの自動化の両方を処理します。

この場合、問題判別 NetView プログラムは、POI、CNMI、およびハードウェア・モニター・ローカル装置インターフェースを使用し、オプションでサブシステム・インターフェースを使用することができます。また、問題判別 NetView プログラムは、状況モニターを実行させることもできます。問題判別 NetView プログラムは、自動化 NetView プログラムのリカバリーや、非送信請求 VTAM メッセージ、ネットワーク・アラート・メッセージ、およびローカル・アラート・メッセージを自動化 NetView プログラムに経路指定するなどの、自動化を実行することができます。

自動化 NetView プログラムは、大部分の自動化を実行し、完全に自動化することができないシステム・メッセージおよび問題判別についてはオプションでアラートを生成することができます。この場合、自動化 NetView プログラムは、サブシステム・インターフェースを使用し、さらに通常は、GENALERT およびログ・ブラウズ機能も使用します。

マイグレーション

NetView プログラムの以前のリリースから現行のリリースにマイグレーションするときに、NetView プログラムは、NetView テスト プログラムで自動機能を実行しながら、ネットワーク管理機能および問題判別機能を実行させることができます。このとき、実動 NetView プログラムは、POI、CNMI、およびハードウェア・モニター・ローカル装置インターフェースを使用します。テスト NetView プログラムは、サブシステム・インターフェースを使用し、オプションで GENALERT インターフェースを使用します。

2 つの NetView プログラム間の通信

NetView プログラムの機能上の制約から、システム内の 2 つの NetView プログラム間で通信を確立しなければならない場合があります。例えば、問題判別 NetView プログラムが自動化 NetView プログラムに対し、非送信請求 VTAM メッセージ、ローカル装置警報メッセージ、およびネットワーク・アラート・メッセージを送信しなければならない場合があります。また、バックアップやリカバリーの目的で、2 つの NetView プログラム間の通信が必要になる場合もあります。

単一システム内で 2 つの NetView プログラム間の通信を確立するためには、使用できる手段として次のものが挙げられます。

- LUC アラート転送
- コマンドとメッセージの転送
- LU 6.2 トランスポート
- MVS サブシステム・インターフェース

LUC アラート転送

NetView の LUC アラート転送機能を使用して、ある NetView プログラムから別の NetView プログラムに、アラートを経路指定することができます。アラート・データベース内のどのレコードを転送したいか指定するには、ハードウェア・モニターの ROUTE フィルターを使用してください。

アラートが自動化テーブルを介して送信されると、DUIFECMV コマンド・プロセッサが呼び出されます。このコマンド・プロセッサによって、GMFHS に情報が送信され、アラートの GMFHS 処理が開始されます。GMFHS の複数のコピーが実行されている場合は、DUIFECMV パラメーター GMFHSDOM によって、GMFHS のアラート転送先ドメインを変更することができます。詳しくは、「*IBM Tivoli NetView for z/OS Resource Object Data Manager and GMFHS Programmer's Guide*」を参照してください。

コマンドとメッセージの転送

RMTCMD コマンドを使用して、コマンドをある NetView プログラムから別の NetView プログラムに送信することができます。コマンドの実行結果のメッセージは、コマンドが出された NetView プログラムに戻されます。MSG コマンドなどのメッセージを送信するコマンドを出すと、メッセージ転送に RMTCMD コマンドを使用することもできます。

また、OST-NNT セッションを使用して、コマンドおよびメッセージを転送することもできます。ある NetView プログラム上の OST が別の NetView プログラム上の NNT としてログオンすると、ROUTE コマンドを使用してコマンドを転送することができます。OST が NNT にコマンドを出すとその応答としてメッセージが受信され、NNT に到達したメッセージは OST に渡されます。NetView のメッセージ転送サンプルを使用すると、OST-NNT 通信を行うことができますが、転送方式を作成して行うこともできます。

LU 6.2 トランスポート

MS トランスポートまたは高性能トランスポートを使用して、2 つの NetView プログラム間の通信（単一システムでの 2 つの NetView プログラム間の通信も含む）を行うことができます。両方の NetView プログラムに LU 6.2 アプリケーションを作成し、その 2 つの NetView プログラム間で MSU を送信することができます。また、1 つの NetView プログラムにアプリケーションを作成し、他の NetView プログラムにある NetView プログラム提供のアプリケーションと通信を行うことができます。例えば、NetView は NVAUTO アプリケーションを提供していますが、このアプリケーションは MSU を受信し、それを自動化テーブルに直接渡すことができます。

MVS サブシステム・インターフェース

3 つ以上の NetView 間で通信する場合、各 NetView プログラムは、サブシステム・インターフェースを使用しなければなりません。そのため、以下のアクティビティを行う必要があります。

- 各 NetView プログラムをサブシステムとして定義する。
- 各 NetView プログラムのサブシステム・アドレス・スペースを稼働させる。
- 各 NetView プログラムの CNMCSSIR タスクを開始する。

各 NetView プログラムの必要条件について詳しくは、517 ページの『MVS サブシステム・インターフェース』を参照してください。各 NetView プログラムは、サブシステム・インターフェースを通して WTO を出し、他の NetView プログラムは、この WTO を使用します。この方法は、VTAM プログラムに依存しません。

NetView プログラムの自動化リカバリー

2 つの NetView プログラムを実行させる場合、各 NetView プログラムは、他方の NetView をモニターしリカバリーします。各 NetView プログラムは、他の NetView プログラムが異常終了したことを示すシステム・メッセージを検出することができます。また、能動モニターを使うこともできます。例えば、EVERY コマンドを使用して、一方の NetView プログラムからもう一方の NetView プログラムにメッセージを定期的に送信し、応答が戻されるかどうかを確認するといったことが可能です。RMTCMD セッションまたはサブシステム・インターフェースを使用して、メッセージを送信することができます。応答が指定時間内に返されないときには、発信元の NetView プログラムは、他の NetView プログラムでリカバリー処理が必要になっていると考えます。能動モニターを使用している場合、1 つの NetView プログラムを正常終了できるように、自動化操作をオフにする方法を必ず用意してください。

MVS で 2 つの NetView プログラムが互いにモニターし合う場合、相手側の NetView プログラムを再始動させるために MVS コマンドを出さなければならないことがあるため、両方の NetView プログラムにはサブシステム・インターフェースが必要です。EMCS コンソールを使用している場合には、MVS システム・コマンドを出すためにサブシステム・インターフェースを使用する必要はありません。

優先順位

NetView サブシステム・アドレス・スペースのディスパッチング優先順位は、高優先順位にする必要があります。

システム自動化を行う、NetView アプリケーション・アドレス・スペースのディスパッチング優先順位は、サブシステム・アドレス・スペースの優先順位より低く、自動化の対象になるアドレス・スペース (リソース管理機能 (RMF™) と汎用トレース機能 (GTF) のアドレス・スペースを除く) の優先順位より、高くしてください。

ネットワーク自動化およびネットワーク管理機能を行う、NetView アプリケーション・アドレス・スペースのディスパッチング優先順位は、VTAM プログラムの優先順位より低く設定してください。また、アプリケーションの優先順位と NetView アプリケーションの優先順位を同じにしてください。

第 33 章 自動操作のチューニング

自動化機能を調整し、パフォーマンスを最適化するためには、以下のメソッドを使用します。

- ログ分析プログラム
- マルチタスキングおよびタスク優先順位
- 自動化テーブル処理
- ハードウェア・モニター・アラート

その他のメソッドについては、「*IBM Tivoli NetView for z/OSチューニング・ガイド*」を参照してください。

AUTOCNT コマンドを使用すると、自動化テーブル使用報告書を生成することができます。この報告書には、比較項目とシステムで発生する自動化のレベルに関する情報が示されます。詳しくは、「275 ページの『自動化テーブル使用報告書』」を参照してください。

また、TASKMON コマンドおよび TASKUTIL コマンドを使用して、自動化ワークロードを分析することができます。詳しくは、「*IBM Tivoli NetView for z/OSチューニング・ガイド*」を参照してください。

ログ分析プログラム

自動化環境を確立する場合、発信元で不要なメッセージを抑止することで、ネットワーク内で発生するメッセージ通信量を少なくしてください。また、頻繁に出されるメッセージでもオペレーターの介入を必要としない場合は、それらのメッセージを自動化してください。サンプルのログ分析プログラムは、JES2 および JES3 両方のログを分析しますが、このプログラムは抑止および自動化の対象になるメッセージを識別する際に役立ちます。

注: NetView ログ分析プログラムの代わりに、IBM Tivoli for z/OS System Automation 製品用のメッセージ分析プログラム (SYSLOG アナライザー) をダウンロードして使用できます。このツールは、SA z/OS サポートの Web ページからダウンロードできます。

CNMS62J2 NetView サンプルでは、ログ分析プログラムを実行するためのサンプル JCL が用意されています。CNMS6207 NetView サンプルでは、CNMS62J2 サンプルが実行し、SYSANAL 実行可能プログラムとして使用できる、サンプルの PL/I プログラムが用意されています。この JCL は JES2 ログを処理するようにセットアップされています。個々の環境に合わせてこの JCL をカスタマイズします。このカスタマイズでは、ルーチンの分類、ファイルの分類、入力ファイルのボリューム、および PARMS などのオプションでの変更が含まれることがあります。このプログラムは、JES2 および JES3 ログのメッセージ頻度を分析する目的で作成されていますが、任意のログのメッセージ頻度を分析するために修正することができます。別のログを分析するためにプログラムを変更するための指示は、プログラムの先頭のコメントに示されています。

このプログラムのオプションには、処理するログのタイプ、開始時刻と停止時刻、時間間隔、およびフィルター操作などの指定があります。図 171 に示している例では、プログラムに渡されるパラメーターの値は、JES2 のログ・タイプで、開始時刻 01:00:00 (01:00:00 以降のタイム・スタンプが付けられたレコードだけが報告書の対象になります)、停止時刻 03:00:00 (停止時刻が 03:00:00 以前のレコードだけが報告書の対象になります)、および時間間隔として 1 時間です (報告書は毎時間ごとの副報告書に分割されます)。さらに、フィルター操作はオフに設定されています。

```

06/12/10 (000612)                JES2 SYSLOG MESSAGE FREQUENCY ANALYSIS                01:01:17
-----
PARAMETER VALUES:
-----
LOG TYPE:      JES2
START TIME:    01:00:00
STOP TIME:     03:00:00
TIME INTERVAL: 01:00:00
FILTERING:     OFF
06/12/10 (000612)                JES2 SYSLOG MESSAGE FREQUENCY ANALYSIS                01:01:17
-----
+++++ ANALYSIS STATISTICS +++++
CURRENT INTERVAL: 000612 01:01:17 - 000612 02:01:16
NUMBER OF SECONDS ANALYZED IN LOG:      3600
TOTAL NUMBER OF MESSAGE LINES:          27876
TOTAL NUMBER OF UNIQUE MESSAGE IDS:      63
AVERAGE NUMBER OF MESSAGES/SECOND:      7.74
+++++ MESSAGE ID FREQUENCY ANALYSIS +++++
MESSAGE ID      INDIVIDUAL CUMULATIVE
      ID      FREQUENCY  PERCENTAGE  PERCENTAGE  PARTIAL TEXT OF FIRST OCCURRENCE
=====
$HASP250      6300         22.60      22.60      RWJBB13 IS PURGED
$HASP530      6130         21.99      44.59      TJRMB13 ON L9.ST1          2 RECORDS
$HASP540      6002         21.53      66.12      RWJBB13 ON L9.SR1          2 RECORDS
$HASP534      3884         13.93      80.05      L9.ST1 INACTIVE
IKJ574I       2342          8.40      88.46      NO SPACE IN BROADCAST DATA SET FOR MAIL
ACT510I        472          1.69      90.15      #CLR019V.MSG STEP WAS NOT EXECUTED
$HASP100        288          1.03      91.18      U$MASSU ON L37.JR1  PROTO.USING
IEC141I        254          0.91      92.09      COMMSP.AS2.NMSD4W.ACNMCLST
ICH70001        171          0.61      92.71      DATAMGT LAST ACCESS AT 01:00:29 ON TUESDAY, SEPTEMB
$HASP373        171          0.61      93.32      #CLR032V STARTED - INIT 9 - CLASS 0 - SYS C33R
:
=====
TOTAL:      27876

```

図 171. ログ分析プログラムの出力

図 171 の例では、最初の時間間隔でログに記録された 10 番目までのメッセージだけが示されています。上記のように、01:01:17 と 02:01:16 の間には 27876 のメッセージと 63 の異なるメッセージ ID が記録されました。同じ時間内に、拡張複数コンソール・サポート (EMCS) コンソールをモニターしているオペレーターには、コンソールに 1 秒間平均 7.74 個のメッセージが表示されます。

CUMULATIVE PERCENTAGE (累積 %) の欄を調べると、報告書に書き出されている 10 番目までのメッセージがその期間内にログに書き出されたメッセージの 93.32% を占めていますが、これは異常ではありません。通常、最も頻繁に生成される 10 番目までのメッセージが、ログのメッセージの 80% 以上を占めます。

報告書のメッセージを分析すると、オペレーターが環境を管理する際に実際には必要のないメッセージがわかります。その結果、それらのメッセージをオペレーター

に表示しないようにすることもできます。いくつかのメッセージを抑止する場合、フィルターをオンにしてプログラムを再実行することができます。フィルターの対象にするメッセージをフィルター・ファイルに指定すると、それらのメッセージを実際に抑止した場合のシミュレーションを実行することができます。

図 172 は、フィルター・ファイルにフィルターの対象となる 5 つのメッセージを示しています。フィルター・ファイルの形式にはメッセージ処理機能 (MPF) ファイルの形式をそのまま使用できるため、フィルター操作の入力ファイルとして MPF ファイルを使用することができます。

\$HAP250
\$HAP530
\$HAP540
\$HAP100
\$HAP373

図 172. フィルター操作されるメッセージ

この結果生成される報告書の先頭には、524 ページの図 171 に示されているリストと類似したリストが表示されますが、フィルター操作は ON に設定されています。さらに、2 番目のリストでは、フィルター操作の対象となるメッセージが存在していなかったように、ログのメッセージ情報が表示されます。図 173 に、2 番目のリストを示します。

06/12/10 (000612)		JES2 SYSLOG MESSAGE FREQUENCY ANALYSIS			01:01:17

+++++ FILTERING ANALYSIS +++++					
ATTEMPTED FILTERING MESSAGE IDS:					

\$HASP100	\$HASP250	\$HASP373	\$HASP530	\$HASP540	

+++++ ANALYSIS STATISTICS +++++					
CURRENT INTERVAL: 000612 01:01:17 - 000612 02:01:16					
NUMBER OF SECONDS ANALYZED IN LOG: 3600					
TOTAL NUMBER OF MESSAGE LINES: 8985					
TOTAL NUMBER OF UNIQUE MESSAGE IDS: 58					
AVERAGE NUMBER OF MESSAGES/SECOND: 2.50					
+++++ MESSAGE ID FREQUENCY ANALYSIS +++++					
MESSAGE	INDIVIDUAL	CUMULATIVE			
ID	FREQUENCY	PERCENTAGE	PERCENTAGE	PARTIAL TEXT OF FIRST OCCURRENCE	
=====					
\$HASP534	3884	43.23	43.23	L9.ST1 INACTIVE	
IKJ574I	2342	26.07	69.29	NO SPACE IN BROADCAST DATA SET FOR MAIL	
ACT510I	472	5.25	74.55	#CLR019V.MSG STEP WAS NOT EXECUTED	
IEC141I	254	2.83	77.37	COMMS.PAS2.NMSD4W.ACNMCLST	
ICH7000I	171	1.90	79.28	DATAMGT LAST ACCESS AT 01:00:29 ON TUESDAY, SEPTEMB	
\$HASP395	170	1.89	81.17	#CLR019V ENDED	
IEF404I	170	1.89	83.06	#CLR019V - ENDED - TIME=01.02.01	
IEF403I	170	1.89	84.95	#CLR032V - STARTED - TIME=01.02.06	
IKJ144I	147	1.64	86.59	UNDEFINED USERID(S) NCPACF	
\$HASP520	131	1.46	88.05	NOTIFY ON L35.JT2	
:					
:					
=====					
TOTAL:	8985				

図 173. フィルター操作を行ったログ分析プログラムの出力

フィルター操作を実行することで、01:01:17 と 02:01:16 の間で出されたメッセージで 58 個の固有なメッセージ ID を持っているメッセージは、8985 個だけでした。ファイル内の 63 の固有なメッセージ ID が付けられている 5 つのメッセージをフ

フィルター操作の対象にすると、オペレーターに表示されるメッセージ数が 68% 減少し、オペレーターに表示されるメッセージの 1 分間当たり平均が 7.74 から 2.50 に減少しました。

ログに書き込まれているメッセージの分析は進行中のプロセスである必要があります。さらに、さまざまな時間間隔で分析してください。例えば、2 番目のシフトでバッチ・ジョブ・ストリームが実行されると、最初のシフトで生成されたメッセージ・セットと異なったメッセージ・セットが生成される可能性があります。システムおよびネットワークは常に変化しているため、受信されるメッセージも変化します。

リソース制御、タスクの優先順位、およびマルチタスキング

NetView は複数のサブタスクで構成されていますが、各サブタスクはストレージおよび CPU サイクルを獲得するために競合します。さまざまな NetView サブタスクに割り当てられた相対的な優先順位により、実行される作業の順序が変わります。NetView をシステムに定義する方法によって NetView のパフォーマンスは影響を受けます。したがって、NetView サブタスクのパフォーマンスもすべて影響を受けることになります。

リソース制御

NetView には自動化リソースの使用状況を測定する方法が備わっており、処理順序の制御、バックログの削減、およびパフォーマンスの最適化を行うことができます。DEFAULTS コマンドおよび OVERRIDE コマンドを使用することによって、任意のタスクに対して固有の制限を設定することができます。コマンド・パラメーターは相互に作用し合います。したがって、CPU、ストレージ、メッセージ・キューイング、入出力の順にパラメーターを変更することができます。パラメーターを変更するごとに、すべての値について効果をモニターしてください。

重要: 過度に低い値を指定すると、システム・パフォーマンスが低下する場合があります。

CPU 使用率

タスクの CPU 使用率を制限するには、DEFAULTS コマンドおよび OVERRIDE コマンドの MAXCPU パラメーターを使用します。TASKMON コマンドおよび SMF データを使用すると、タスク単位で CPU 使用率を分析することができます。

タスクの CPU 使用率を制限すると、ディスパッチ優先順位の変更よりもパフォーマンスが向上します。これは、使用率を制限すると CPU を使用するタスクではなく、作業に割り込むタスクの数が影響を受けるためです。

ストレージの使用状況

DEFAULTS コマンドには MAXSTG、SLOWSTG、AVLSLOW、および AVLMAX パラメーターがあります。これらのパラメーターを使用すると、特定の限界値以上にストレージを使用している場合にタスクの応答を変更することができます。

OVERRIDE コマンドは、タスクごとに SLOWSTG 値および MAXSTG 値を設定することができます。TASKMON コマンドおよび SMF データを使用すると、タスク単位でストレージの使用状況を分析することができます。

AVLSLOW パラメーターおよび SLOWSTG パラメーターは、大量のストレージを使用している自動化を意図的に遅くするために使用します。タスクのためのストレージを制限するとスループットが低下しますが、自動化の喪失の原因となる突然のストレージの減少を防ぐことができます。

特定のタスクによってストレージが無制限に使用されないように制限するには、MAXSTG パラメーターおよび AVLMAX パラメーターを使用します。これによって、そのタスクについて自動化が中断される場合がありますが、他のタスクが影響を受けないという利点があります。

BNH162I メッセージおよび BNH163I メッセージをモニターしてください。これらのメッセージによって、NetView アドレス・スペースがワークロードに対して小さすぎる場合に通知が行われます。BNH162I メッセージまたは BNH163I メッセージを受け取った場合は、次に NetView プログラムを開始するときに領域サイズを拡大してください。

メッセージ・キューイング

DEFAULTS コマンドおよび OVERRIDE コマンドには、タスク間を流れるメッセージの速度を制御するための MAXMQIN および MAXMQOUT パラメーターがあります。メッセージ通信量が原因であるストレージの過大な使用や待機による遅延を避けるには、フロー速度を調整します。TASKMON コマンドおよび SMF データを使用すると、タスク単位でメッセージ・フロー速度を分析することができます。

MAXMQOUT パラメーターは、NetView メッセージ・キューイング・サービスを使用してタスクがデータを送ることができる速度 (KB/分) を制御します。

MAXMQOUT パラメーターは、主にデータを送信するタスク (CNMCSSIR など) に対して使用します。

MAXMQIN パラメーターは、すべてのタスクについて他のタスクにデータを送信できる速度 (KB/分) を制御します。MAXMQIN の設定値は、主にデータを受け取るタスク (DSILOG など) に対して使用します。

入出力の使用

DEFAULTS コマンドおよび OVERRIDE コマンドには、タスクを実行できる速度 (1 分あたりの入出力回数) を制御するための MAXIO パラメーターがあります。

TASKMON コマンドおよび SMF データを使用すると、入出力速度を分析して、頻繁に入出力を行うタスクを決定することができます。

LOADCL を使用して入出力速度を低下させることもできます。

タスクの優先順位

NetView プログラムのタスク優先順位は、次のようになります。

PPT 基本プログラム・オペレーター・インターフェース・タスク (PPT) は、全タスク (優先順位 1 で初期設定されるデータ・サービス・タスク (DST) を含む) の中の最高優先順位タスクです。

OST および NNT

優先順位は PPT より低く、自動タスクよりも高くなっています。オペレーター端末タスク (OST) と NetView 間タスク (NNT) の相対優先順位は 4 です。

自動タスク

優先順位は PPT および OST/NNT より低くなっています。自動タスクの相対優先順位は 5 です。

DST DST は自動化に直接含まれませんが、優先順位のレベルによっては、自動化のパフォーマンスに影響を与えます。DST の優先順位は、CNMSTYLE メンバーの中で指定するか、START コマンドを使用して指定します。

CNMSTYLE メンバーの詳細については、「*IBM Tivoli NetView for z/OS* インストール:概説」に記載されています。

高優先順位自動化プロシージャに PPT で実行できないコマンドが含まれていない場合は、PPT のもとでその自動化プロシージャを実行することができます。しかし、大部分の自動化プロシージャは自動タスクのもとで実行してください。自動化テーブルからプロシージャを呼び出すときには、実行タスクを指定するために EXEC アクションに ROUTE キーワードを使用することができます。

複数の自動タスク

複数の自動タスクを使用して自動化のワークロードを複数プロセッサ間に分散させ、マルチタスキングを利用してスループットを向上させてください。自動化ワークロードのスループットの制約要因は、システム・プロセッサに対する競合、またはコマンド・リスト・データ制御ブロック (DCB) (これは、コマンド・リスト・データ・セットに対する入出力の同期化を制御します) に対する競合です。

LOADCL コマンドを使用してコマンド・リストが事前ロードされるときには、コマンド・リスト・データ・セットに対する入出力は不要であるため、自動化スループットの制約要因はプロセッサの処理能力だけになります。複数のプロセッサを使用すると、自動タスクの数をプロセッサの数より多くしても、スループットはそれほど向上しません。

複数の NetView プログラム

システムで複数の NetView プログラムを使用する場合 (513 ページの『第 32 章 単一システムでの複数 NetView プログラムの実行』で説明されています) には、自動化 NetView プログラムのタスク優先順位をその NetView プログラムの自動化の対象になる VTAM プログラムおよびアプリケーションより高くしてください。2 番目の NetView プログラムがアプリケーションの応答時間に与える影響を最小限に抑えるため、2 番目の NetView プログラムには重要なアプリケーションの優先順位より低い優先順位を与えることができます。NetView のワークロードをいくつかのアドレス・スペースに分割しても NetView システム全体のプロセッサ使用率は低下せず、しかも NetView 全体のストレージの使用が増加する可能性があります。しかし、ワークロードを分割すると、自動化の即応性および可用性は向上します。

自動化テーブル処理

受信されるメッセージおよび MSU の数が多い場合、自動化テーブルは NetView プログラムにより定期的にチェックされるため、自動化テーブルを慎重に設計すると処理時間の大幅な短縮を図ることができます。266 ページの『自動化テーブルの設計上のガイドライン』では、自動化テーブルの設計を良くするためのいくつかの原則について説明します。本セクションでは、処理効率に直接影響を与えるいくつかの原則について要約します。

- オペレーティング・システムに用意されているメッセージ処理機能を使用して、なるべく多くのメッセージを抑止し、自動化テーブル処理の対象から外してください。
- BEGIN-END セクションを使用してテーブルを構成し、各メッセージまたは MSU に必要な比較回数を減らしてください。
- 頻度に合わせて、BEGIN-END セクションの順序を決定してください。頻繁に受信されるデータを処理するセクションは、自動化テーブルの先頭に指定してください。
- 各 BEGIN-END セクション内では、ステートメントを頻度順に指定してください。頻繁に受信されるデータを処理するステートメントは、セクションの先頭に指定してください。
- 頻繁に受信されるデータでも自動化の対象にしたくない場合には、アクションを指定しないステートメントをテーブルの先頭に指定して、NetView プログラムがテーブル全体を処理しないようにすることができます。例えば、NetView 端末から出されるコマンドを自動化しないときには、テーブルの先頭に次のステートメント (図 174) を追加してください。

```
IF HDRMTYPE = '*' THEN ;
```

図 174. 自動化テーブルのコマンド処理の回避

- 処理に時間がかかる機能が必要以上に呼び出されるのを避けるために、それらの機能は個別に指定してください。処理に時間がかかると思われる機能には、システム・オペレーターが自分で作成する長い自動化テーブル機能 (ATF) 以外に、DSICGLOB および MSUSEG などがあります。これらの機能は、BEGIN-END セクションの中、または論理 AND (&) のあとに指定してください。その機能を必要としているメッセージおよび MSU だけに、機能呼び出ししてください。
- 自動化テーブルだけでメッセージまたは MSU を処理することができる場合には、そのメッセージまたは MSU を処理するためにコマンド・プロシーチャーを呼び出さないでください。

AUTOCNT コマンドを使用して自動化テーブル使用報告書を生成します。この報告書には、各ステートメントがメッセージまたは MSU と比較された回数、およびそのステートメント比較処理が一致した回数が見られます。詳細については、275 ページの『自動化テーブル使用報告書』を参照してください。

AUTOMAN コマンドを使用すると、自動化テーブルを管理することができます。詳細については、287 ページの『複数の自動化テーブルの管理』を参照してください。

ハードウェア・モニター・アラート

ハードウェア・モニターのアラートを使用して自動化環境の情報を表示する場合には、オペレーターに送られるアラートの数を制限してください。

処理対象となるすべてのアラートについて、ハードウェア・モニターはオペレーターが Alerts-Dynamic パネルを表示するかどうかを検査します。ハードウェア・モニターは、このパネルを表示する各オペレーターにどのアラートを送信するか決定し、そのオペレーターの画面を更新します。オペレーター画面の更新の処理費用はイベント、統計、およびアラートのワークロードの処理費用を大幅に上昇させる可能性があります。そのため、アラートを受け取る率が高いときには、オペレーターのコマンド許可検査を使用するか、表示フィルターを使用して、Alerts-Dynamic パネルの使用を制御してください。

第 34 章 自動化テーブルのテスト

自動化テーブルが正しく動作するように、実動環境で使用する前に、自動化テーブルは完全にテストしておいてください。自動化テーブルをテストするには、システム・オペレーターがテーブルの処理対象になるメッセージおよび MSU を生成して、各メッセージおよび MSU に対して自動で正しい応答を出しているかどうか判別します。

自動化テーブルをテストするには、次の方法があります。

- AUTOTEST コマンドおよび AUTOCNT コマンドを使用して、自動化テーブルのメッセージおよび MSU の効果をテストする。
- 実動環境と同じテスト環境で、自動化の手順をセットアップする。
- 実動システムに自動化を段階的に導入し、各段階ごとに自動化の手順が正常に実行されていることを確認してください。

注:

1. AUTOTEST コマンドおよび AUTOCNT コマンドでは、一度にテストできる自動化テーブルは一つです。この自動化テーブルには、%INCLUDE ステートメントを使用することによってメンバーを追加して組み込むことができます。
2. AUTOMAN コマンドを使用して、複数の自動化テーブルを管理することができます。さらに、追加の診断機能が用意されています。詳しくは『287 ページの『複数の自動化テーブルの管理』』を参照してください。

次のトピックでは、上記の方法と自動化テーブルの診断手順について説明します。

自動化テーブルのテスト

AUTOTEST コマンドを使用して、自動化テーブルをテストします。メッセージおよび MSU の記録済み入力ストリーム、またはアクティブの自動化テーブルによって処理されているメッセージおよび MSU のいずれかを使用します。また、将来自動化テーブルをテストするために、メッセージおよび MSU を記録することもできます。

自動化テーブルのテストには、2 つの方法があります。

- アクティブの自動化テーブル処理とテストの自動化テーブル処理を並行して比較する。AUTOCNT コマンドで生成されたレポートが使用できます。
- アクティブの自動化テーブルによって AIFR (メッセージおよび MSU) が処理されるときに、これらの AIFR を前もって記録する。それから、これらの AIFR を使用して、自動化の結果が正常になるまで、必要なだけテストを繰り返します。

並行テストの開始

アクティブの自動化テーブルと並行して自動化テーブルをテストするには、次の手順を行います。

1. 下記のようなコマンドを使用して、並行テストをアクティブにします。

```
AUTOTEST MEMBER=TESTTBL,LISTING=TESTLST,SOURCE=PARALLEL,  
REPORT=TESTRPT
```

このコマンドが、自動化テーブル TESTTBL のテストを開始します。自動化テストの報告書 (TESTRPT) が生成されます。自動化テーブル TESTTBL の統計の新しいセットが保持されます。また、将来のテストのために AIFR を記録する場合は、RECORD のキーワードを AUTOTEST コマンドに追加してください。例えば、次のように指定します。

```
AUTOTEST MEMBER=TESTTBL,LISTING=TESTLST,SOURCE=PARALLEL,  
REPORT=TESTRPT,RECORD=TESTRECS
```

AIFR はメンバー TESTRECS に記録されます。

2. アクティブのテーブル・カウンターを、テスト状態のテーブルと一致するようにリセットします。

```
AUTOCNT RESET
```

3. メッセージと MSU が両方の自動化テーブルによって処理される間、しばらくテストを続行します。
4. アクティブとテストの両方の自動化テーブルの、自動化テーブル使用報告書を作成します。

```
AUTOCNT REPORT=BOTH,FILE=PRODTBL,STATS=DETAIL  
AUTOCNT REPORT=BOTH,FILE=TESTTBL,TEST,STATS=DETAIL
```

5. 任意の時点で結果を検査します。カウンターをリセットして、自動化テーブルのテストを続行するには、次のように入力します。

```
AUTOCNT RESET,STATS=DETAIL  
AUTOCNT RESET,TEST,STATS=DETAIL
```

6. AUTOCNT コマンドおよび AUTOTEST コマンドによって生成された出力を比較して、自動化テーブル処理が正常であるかどうか判断してください。

注: 着信通信量のために、アクティブのテーブル数には入っていないメッセージやアラートのいくつか、テスト・テーブル数に入っている可能性があります。これは、両方のテーブルが全く同時にはクローズされていないからです。

7. 自動化テーブルのテスト報告書を調べて、テーブルの論理が正しいかどうかを検証します。テスト中に届いたキー・メッセージとアラートが、正しく処理されたかどうか (正しい自動化テーブル・ステートメントに合致したか) を、検証してください。テスト報告書例については、534 ページの『AUTOTEST コマンドの報告書例』を参照してください。
8. テストが正常である場合は、テスト自動化テーブルを AUTOTBL コマンドを使用してアクティブにします。テストが正常でない場合は、テスト自動化テーブルを変更して、そのテストを記録済み AIFR で再実行してください。この手順は、次のトピックで説明します。

記録済み AIFR を使用しての、自動化テーブルのテスト

前の AUTOTEST コマンドによって記録されたメッセージと MSU AIFR を使用して、自動化テーブルをテストし、自動化テーブルを変更し、それから一定の入力データのセットを使用してその結果を分析することができます。

AIFR を記録するには、次のことを行ってください。

1. AIFR の記録をアクティブにします。

AUTOTEST RECORD=TESTRECS

2. 処理終了後、AIFR の記録を停止します。

AUTOTEST RECORD=OFF

テストに必要でない AIFR を削除することによって、記録済み AIFR が入っているファイルを変更できます。AIFR をファイルから削除するには、!!----- で始まる行とそれに続く行を、次の !!----- 行まで (この行は含まない) 削除します。

1. 削除する AIFR をファイルの中に探し出します。ファイルのその AIFR の下に、以下のとおり始まる分離線があります。

!!-----

2. AIFR データ行と AIFR に続く分離線を削除します。

記録済み AIFR ファイルを保管した後、記録済みファイルの最初の行のセキュリティ・キーを、S> から <S に変更できます。このように変更することによって、次の AUTOTEST コマンドがファイルを上書きしないようにします。AUTOTEST コマンドでは、コマンドのソース・ファイルとして使用されるファイルの最初のレコードに、S> または S< のセキュリティ・キーが表れることが必要です。

記録済み AIFR を使用して、自動化テーブルのテストを開始するには、次のことを行ってください。

1. テスト自動化テーブル・カウンターをリセットし、統計の収集を始めます。

AUTOCNT RESET,TEST

2. 自動化テーブルのテストを開始します。

AUTOTEST SOURCE=TESTRECS,REPORT=TESTRPT

3. メッセージ BNH382I が表示されると、テスト自動化テーブルに関する統計の収集を停止して、ソース・データ・セットにファイルの終わり (EOF) を示します。

AUTOCNT REPORT=BOTH,FILE=TESTTBL,TEST
AUTOCNT RESET,TEST

4. 報告書と統計を分析して、自動化テーブル処理が正常であるかどうかを判断します。また、自動化テーブルのテスト報告書を調べて、テーブルの論理が正しいかどうかを検証します。正常である場合は、新規自動化テーブルを AUTOTBL コマンドを使用してアクティブにします。正常でない場合は、自動化テーブルを変更して、結果が正常になるまでこの処理を繰り返します。

記録済み AIFR を使用して、2 つの自動化テーブルの間の処理を比較します。例えば、TESTTBL1 と TESTTBL2 の間の処理を比較します。

1. 前の例の記録済み AIFR を使用して、自動化テーブル TESTTBL1 をテストします。

AUTOTEST MEMBER=TESTTBL1,LISTING=LIST1,SOURCE=TESTRECS,
REPORT=TESTRPT1

2. TESTTBL1 のテストが終了したことを示すメッセージ BNH382I が表示されると、自動化テーブル TESTTBL2 のテストが始まります。

AUTOTEST MEMBER=TESTTBL2,LISTING=LIST2,SOURCE=TESTRECS,
REPORT=TESTRPT2

3. TESTTBL2 のテストが終了したことを示すメッセージ BNH382I が表示されると、自動化テーブル TESTRPT1 と TESTRPT2 で検出された報告書を比較します。
4. 自動化テーブルのテスト報告書を調べて、テーブルの論理が正しいかどうかを検証します。

AUTOTEST コマンドの報告書例

AUTOTEST コマンドは、テストしている自動化テーブルによって処理されたメッセージと MSU を示す報告書を作成します。報告書と AUTOTEST リスト・ファイルを使用して、報告書に示された自動化テーブル・ステートメントを理解してください。報告書で注意すべき項目を下記に示します。

- 1** AUTOTEST 報告書には、最初のレコードに 2 文字のセキュリティー・キーがあります。文字 R は、報告書が AUTOTEST コマンドによって作成されたことを、文字 > は、報告書を次の AUTOTEST コマンドによって上書きできることを示します。> を、AUTOTEST コマンドが報告書を上書きできない < に変更できます。同様に、AUTOTEST コマンドによって作成されたリスト・ファイルには、最初の位置に文字 L があり、記録済み AIFR ファイルには、最初の位置に文字 S があります。また、これらのファイルの > または < の文字は、AUTOTEST コマンドがそのファイルを上書きできるかどうかを示します。
- 2** メンバー TESTTBL1 のメッセージ DSI077A には、一致するものが 3 つあります。シーケンス番号 00120020、ステートメント 2、および MYLABEL1 のラベルが付いたステートメントによって一致したことを示します。

形式化された報告書の例を下記に示します。

```
R> 1
>> Automation table test of member DSIPARM.TESTTBL1      Listing: LIST1
>> Time: 04/06/10 08:54:46 Requesting operator: OPER1    Source: TESTRECS

-----> Input number: 1. Type = Message -----
LIST ''

Matches: 0 Comparisons: 1

-----> Input number: 2. Type = Message -----
STATION: OPER1      TERM: NTB4L702

Matches: 0 Comparisons: 1

-----> Input number: 3. Type = Message -----
HCOPY: NOT ACTIVE  PROFILE: DSIPROFA

Matches: 0 Comparisons: 1

-----> Input number: 4. Type = Message -----
STATUS: ACTIVE      IDLE MINUTES: 0

Matches: 0 Comparisons: 1

-----> Input number: 5. Type = Message -----
```

```

ATTENDED: YES          CURRENT COMMAND: LIST
Matches: 0 Comparisons: 1
-----> Input number: 6. Type = Message -----
AUTHRCVR: YES          CONTROL: GLOBAL
Matches: 0 Comparisons: 1
-----> Input number: 7. Type = Message -----
NGMFADMN: NO           DEFAULT MVS CONSOLE NAME: NONE
Matches: 0 Comparisons: 1
-----> Input number: 8. Type = Message -----
NGMFVSPN: NNNN (NO SPAN CHECKING ON NMC VIEWS)
Matches: 0 Comparisons: 1
-----> Input number: 9. Type = Message -----
NGMFCMDS: YES          AUTOTASK: NO
Matches: 0 Comparisons: 1
-----> Input number: 10. Type = Message -----
IP ADDRESS:  N/A
Matches: 0 Comparisons: 1
-----> Input number: 11. Type = Message -----
OP CLASS LIST: NONE
Matches: 0 Comparisons: 1
-----> Input number: 12. Type = Message -----
DOMAIN LIST: NTVB4 (I) CNM02 (I) CNM99 (I) B01NV (I)
Matches: 0 Comparisons: 1
-----> Input number: 13. Type = Message -----
ACTIVE SPAN LIST: NONE
Matches: 0 Comparisons: 1
-----> Input number: 14. Type = Message -----
END OF STATUS DISPLAY
Matches: 0 Comparisons: 1
-----> Input number: 15. Type = Message -----
LIST  KKK
Matches: 0 Comparisons: 1
-----> Input number: 16. Type = Message -----

```

DSI077A 'KKK' STATION NAME UNKNOWN

Matches: 3 Comparisons: 7 **2**

Match	Location	Location Type	Member
01.	00120020	Sequence Number	TESTTBL1
02.	2	Statement Number	TESTTBL1
03.	MYLABEL1	Label	TESTTBL1

-----> Input number: 17. Type = Message -----

LIST ABND

Matches: 0 Comparisons: 1

-----> Input number: 18. Type = Message -----

DSI077A 'ABND' STATION NAME UNKNOWN

Matches: 3 Comparisons: 7

Match	Location	Location Type	Member
01.	00120020	Sequence Number	TESTTBL1
02.	2	Statement Number	TESTTBL1
03.	MYLABEL1	Label	TESTTBL1

-----> Input number: 19. Type = Message -----

MSG ALL HI

Matches: 0 Comparisons: 1

-----> Input number: 20. Type = Message -----

DSI001I MESSAGE SENT TO ALL

Matches: 1 Comparisons: 7

Match	Location	Location Type	Member
01.	00120020	Sequence Number	TESTTBL1

-----> Input number: 21. Type = Message -----

DSI039I MSG FROM OPER1 : HI

Matches: 2 Comparisons: 3

Match	Location	Location Type	Member
01.	00120020	Sequence Number	TESTTBL1
02.	00160020	Sequence Number	TESTTBL1

-----> Input number: 22. Type = Message -----

DSI039I MSG FROM OPER1 : HI

Matches: 2 Comparisons: 3

Match	Location	Location Type	Member
01.	00120020	Sequence Number	TESTTBL1
02.	00160020	Sequence Number	TESTTBL1

-----> Input number: 23. Type = Message -----

DSI039I MSG FROM OPER1 : HI

Matches: 2 Comparisons: 3

Match	Location	Location Type	Member

```

-----
01. 00120020      Sequence Number  TESTTBL1
02. 00160020      Sequence Number  TESTTBL1

-----> Input number: 24. Type = Message -----

DSI039I MSG FROM OPER1      : HI

Matches: 2 Comparisons: 3
Match Location      Location Type  Member
-----
01. 00120020      Sequence Number  TESTTBL1
02. 00160020      Sequence Number  TESTTBL1

-----> Input number: 25. Type = Message -----

DATE

Matches: 0 Comparisons: 1

-----> Input number: 26. Type = Message -----

CNM359I DATE : TIME = 14:59      DATE = 03/31/10

Matches: 0 Comparisons: 1

-----> Input number: 27. Type = Message -----

AUTOTEST OFF

Matches: 0 Comparisons: 1

-----> Input number: 28. Type = Message -----

BNH344I AUTOMATION TABLE TESTING IS NOT ACTIVE

Matches: 0 Comparisons: 1

-----> Input number: 29. Type = Message -----

BNH337I NO TEST AUTOMATION TABLE IS LOADED

Matches: 0 Comparisons: 1

-----> Input number: 30. Type = Message -----

AUTOTEST RECORD=OFF

Matches: 0 Comparisons: 1

>> End of automation table test. Time: 04/06/10 08:54:46

```

テスト環境の使用

テスト環境を使用する場合、自動化されるメッセージおよび MSU を生成するためのアプリケーションをセットアップするか、または自動化されるメッセージをシミュレートするプログラムを作成することができます。

アプリケーションの使用

例えば、給与計算アプリケーションを自動化する場合、給与計算アプリケーションのテスト版をインストールし、自動化されるメッセージおよび MSU を生成するために入力データを用意します。これらの準備をしたあと、アプリケーションを実行

し、メッセージおよび MSU を生成してください。そして、結果を調べ、確認してください。期待する結果でなかった場合は、必要な訂正を行い、テストを繰り返して行ってください。

この方法の利点は、メッセージおよび MSU が実際のアプリケーションから渡され、テスト事例を慎重に選択すれば、実動環境で受信されるメッセージに非常に近くなる点です。ただし、アプリケーションのテスト版をインストールしてテスト事例を作成する場合、そのために必要な手間、マシン・タイム、およびその他のリソースを考慮すると高価になります。この費用は、そのまま使用できるテスト版とテスト事例をアプリケーション開発担当者が用意していれば低く押さえられます。

シミュレーターの使用

自動化が単純であれば、シミュレーター・プログラムの作成は容易になります。シミュレーターは、必要なメッセージで構成されるファイルを読み取り、それらのメッセージを出します。また、シミュレーターはアセンブラの DSIAUTO マクロまたは PL/I および C の CNMAUTO サービス・ルーチンを使用して、MSU を作成しそれらを自動化テーブルに渡すことができます。この方法で実行し、実際の結果と予想した結果とを比較し（例えば、正しいコマンドまたは応答が出されたか？など）、必要に応じて訂正することができます。

メッセージのシミュレーション

メッセージ・シミュレーターは、テストに使用するメッセージを指定したファイルを読み取り、各メッセージを順番に出すことができます。メッセージのテキストだけをチェックする簡単な自動化テーブルのステートメントの場合であれば、複雑なシミュレーターは必要ありません。コマンド・リストは、REXX で SAY を使用するか、または NetView コマンド・リスト言語で &WRITE を使用すると、正しいテキストが組み込まれているメッセージを生成することができます。MSGID、TEXT、および TOKEN 比較項目だけを使用する自動化テーブルのステートメントの場合には、この方法だけで十分です。すべての文字を正しい位置に指定して、各メッセージを正確に出すようにしてください。

ステートメントがテキスト以外の情報をチェックしている場合には、より高度なシミュレーターが必要です。メッセージ・シミュレーターで、正しい経路コード、アクション・コード、およびジョブ名が指定されたメッセージを生成しなければならない場合があります。自動化テーブルでメッセージの MVS ジョブ名をチェックしている場合には、シミュレーターはジョブを生成して実行し、正しい名前のメッセージを生成しなければなりません。

入力メッセージ・ファイルには、自動化テーブルで必要としているすべての情報を組み込むことができます。例えば、MVS では、メッセージ ID、経路コード、記述子コード、メッセージ・テキスト、およびジョブ名を組み込むことができます。このファイルを準備したあとにコマンド・プロシーチャーを作成すると、メッセージ・ファイルを入力として扱い、正しいシステム変数を設定し、メッセージを出すことができます。

メッセージ・シミュレーターを使用する場合の問題点は、メッセージがプロダクトまたはアプリケーションにより生成される実際のメッセージではなく、システム・オペレーターが想像したメッセージをもとに作成される点です。

メッセージのシミュレーションを行う方法としては、メッセージ・ファイルを読む代わりに、既存メッセージ・ログ・ファイルを読み取り、そのログの内容に従ってメッセージを再生成できるように、プログラムを作成する方法もあります。使用されるログは実際のシステム・ログまたはネットワーク・ログでもよく、またはそれらのログを編集して、その内容を変更しテスト用に適合させるか、またはシミュレーターの入力要求を満足させるようにすることもできます。このログを使用することで、テストに使用されるメッセージは、実動環境で実行中にアプリケーションから受け取られる実際のメッセージをもとに作成されたことになります。しかし、シミュレーターを使用すると、テスト用にそれらのメッセージをいつでも作成することができます。

MSU シミュレーション

自動化テーブルにアセンブラー、PL/I、および C のインターフェースを使用すると、MSU をシミュレートすることができます。アセンブラーでは、DSIAUTO マクロが自動化テーブル処理を介して MSU を渡します。CNMAUTO サービス・ルーチンは、PL/I および C で同等の機能を実行します。

自動化テーブルに渡すための MSU を作成する方法には、以前に獲得して保管しておいた MSU を使用する方法があります。自動化テーブルには、MSU を選択しそれらをコマンド・プロシージャーに渡す項目を用意することができます。そのコマンド・プロシージャーをあとで使用するために、バッファーをファイルに保管することができます。別のコマンド・プロシージャーを作成し、保管されたバッファーを取り出し、MSU を再構成し、それを自動化操作に渡すことができます。

また、手操作で MSU データ・フィールドを作成すると、テスト用の MSU を作成することもできます。MSU の形式については、「*System Network Architecture Formats*」またはすでに受信されている MSU をもとにして使用可能です。その後、その MSU データ・フィールドを自動化テーブルに渡します。MSU は、自動化テーブルに指定されている MSUSEG をもとにしたステートメントのテストに使用します。

自動化機能の段階的实施

ユーザーの環境に与える影響を最小限に抑えるために、各ステップにチェックポイントを設けて処理が正しいかどうか確認しながら、自動化をインプリメントします。

自動化の準備をする場合、アクションを取る代わりに、オペレーターまたはログに対しルーチンから通知を送信する方法があります。これらのルーチンでは、自動化機能がアクティブになったときに実行されるアクションを指示します。オペレーターは実際のアクションを手操作で実行しますが、この通知情報により、自動化機能がメッセージまたは MSU を正しく代行受信し、自動化しているかどうかを判別することができます。また、同じ方法を使用して、さらに自動化が可能なアクションを調べることもできます。

初めて自動化機能を導入する場合は、自動化されるすべてのアクションをオペレーターに通知するかあるいはログに記録し、正しいアクションが取られていることを

確認してください。自動化が正しく機能していることに満足したら、通知レベルを引き下げ、最終的には、自動化に関する問題のデバッグ (必要な場合) に必要な情報だけを出力するようにしてください。

自動化を段階的に導入するには、次のステップに従ってください。

- 自動化テーブルの一致条件を検証します。
- 自動化アクション・パラメーターを検証します。
- 時間コマンドを検証します。
- 自動化の効果をチェックします。
- 自動タスクがコマンド・プロシージャーを正しく処理していることを確認します。

自動化テーブルの一致条件の検証

自動化テーブル使用報告書を使用して、メッセージおよび MSU が自動化テーブル・ステートメントと正しく一致しているかどうか検査します。

アクションをコメント化して、自動化テーブルにステートメントを追加することができます。ある特定の期間の詳細使用報告書を作成し、それをチェックして、正確な回数だけステートメントが比較され、一致したことを確認します。図 175 には、コメント化されているアクションを指定した自動化テーブル・ステートメントの例を示します。

```
* Automate message DSI123I
  IF MSGID = 'DSI123I' THEN
*      EXEC(CMD('CLIST1') ROUTE(ONE AUTO1 *))
*      DISPLAY(N)
;
```

図 175. コメント化されているアクションが指定された自動化ステートメント

この方法を正しく行うには、使用統計の対象期間中にこれらのステートメントと一致した自動化メッセージおよび MSU の数を知ることが重要になります。ステートメントが検証されたら、アクションからコメント化を外し、自動化テーブルを再活性化することができます。

自動化アクション・パラメーターの検証

自動化テーブルがコマンドまたはコマンド・プロシージャーを呼び出すと、呼び出されたプロシージャーに情報を渡すことができます。実際にプロシージャーを実行しなくても、テストを実行し、渡される情報を検証することができます。

- 自動化でメッセージがトークンを取り出しそれらを自動化プロシージャーに渡すときには、メッセージにあるすべてのトークンを表示するテスト・プロシージャーを作成することができます。自動化プロシージャーの代わりにテスト・プロシージャーを使用して、渡そうとしている情報が所定のトークンに存在することを検証してください。
- 実際の自動化プロシージャーの代わりに、自動化テーブルから呼び出されるテスト・プロシージャーを作成してください。自動化テーブルから、実際の自動化プロシージャーの名前およびそのプロシージャーが開始されるときのパラメーターを渡してください。

テスト・コマンド・プロシージャーでは、次のような処理を行います。

- ネットワーク・ログまたはファイルに、パラメーターを示すレコードを書き込みます。
- パラメーターが正しいかどうかを分析します。
- 各自動化プロシージャーが受信するさまざまなパラメーターのデータを保存します。

スケジュールされたコマンドの検証

自動化にタイマー・コマンドを追加したら、ネットワーク・ログを定期的に調べ、スケジュールされたコマンドが正しく処理されるかどうかを判別することができます。

AT、EVERY、および AFTER コマンドの場合、メッセージ DSI208I が出され、ネットワーク・ログに書き込まれます。このメッセージには、タイマー・コマンドに関連した ID および開始されるコマンドまたはコマンド・プロシージャーの名前が含まれています。

CHRON コマンドの場合、NOTIFY RUN=*taskname* が CHRON コマンド上で指定されていると、メッセージ BNH549I が出されます。開始されたタイマー ID とコマンド名を判別するには、綿密にメッセージ BNH549 を調査してください。

```
BNH549 CHRON NOTIFY=eventname BY=issueoper ID=timerid  
ROUTE=runoper COMMAND='CHRON text'
```

メッセージ BNH549 においては、以下の値に留意してください。

- *timerid* には、当該タイマー・コマンドに関連したタイマー ID が入っています。
- *text* フィールドには、そのコマンドまたはコマンド・プロシージャーの名前が入っています。

ユーザーは、TIMER コマンドを使用して CHRON コマンドを、(メッセージ BNH549I が出される際に必要な) NOTIFY パラメーターを含めて変更することができます。

コマンド・プロシージャーを作成して、タイマー ID およびスケジュールされたコマンド名をファイルに追加することにより、ネットワーク・ログを使用するよりも検証が容易にできます。そのためには、タイマー ID とスケジュールされたコマンドの名前をメッセージ DSI208I または BNH549I から取り出す自動化テーブル・ステートメントを追加します。ID およびコマンド名を、それらをファイルに保管するコマンド・プロシージャーに渡します。REXX コマンド・プロシージャーは EXECIO を使用してレコードをメンバーに書き込みます。PL/I および C は CNMSMSG を使用してレコードを順次ログ・ファイルに書き込みます。アセンブラー・コマンド・プロセッサは DSIWLS を使用してレコードを順次ログ・ファイルに書き込みます。

自動化効果のチェック

テスト環境で自動化操作を実行中に、予期しないメッセージが出されて自動化に割り込みが発生しないように、ステートメントに環境のテストを指定することができます。例えば、自動オペレーター AUTO1 には TEST と呼ばれる共通グローバル変

数があり、この変数はコマンド・リストがテスト・モードで動作する場合は YES に設定され、コマンド・リストが通常どおり機能する場合は、NO またはヌルに設定されます。

\$HASP098 ENTER TERMINATION OPTION メッセージをテストし、その後、JES2 をダンプリ、再始動したい場合は、次の自動化テーブル項目を使用してください。

```
IF JOBNAME = 'JES2' & MSGID = '$HASP098' THEN
  EXEC(CMD('$HASP098') ROUTE(ONE *)) DISPLAY(Y) NETLOG(Y);
```

REXX コマンド・リスト \$HASP098 を図 176 に示します。テスト・モードの場合、オペレーター (およびシステム・ログ) にメッセージが送信され、そのメッセージには TEST 共通グローバル変数が YES に設定されていないときに実行されるアクションが示されています。テスト・ケース分析ツールを使用して、ログからメッセージを取り出して実際の結果と予測した結果を比較することができます。

```
/* $HASP098 Command list */
/* - Responds to message $HASP098 ENTER TERMINATION OPTION */
/* - If in 'TEST' mode, doesn't send response */
'GLOBALV GETC TEST HASP098 LASTREPLY' /* Get global variables */
IF HASP098_LASTREPLY = '' THEN /* If first reply or */
  HASP098_LASTREPLY = 'PURGE' THEN /* did purge last time */
  REPLY_TEXT = 'DUMP' /* Do a dump this time */
ELSE /* Did dump last time */
  REPLY_TEXT = 'PURGE' /* So do a purge this time */
CMD_TEXT = 'MVS R '||REPLYID()||','||REPLY_TEXT /* Build the command*/
HASP098_LASTREPLY = REPLY_TEXT /* Set LASTREPLY for next */
IF TEST = 'YES' THEN /* Are we in 'TEST' mode? */
  'WTO TEST OF $HASP098 - 'CMD_TEXT /* Write test results */
ELSE /* 'LIVE' mode... */
  CMD_TEXT /* Issue the command */
```

図 176. \$HASP098 コマンド・リスト

自動タスクがコマンド・プロシーチャーを正しく処理することの確認

コマンドおよびコマンド・リストを実行させたり、コマンドをスケジュールするために、自動化では自動タスクは重要です。自動タスクは無人操作であるため、自動化プロシーチャーが正しくしかも迅速に開始できるように、自動タスクに関する問題をなるべく早く識別することが必要です。

次の方法は、自動タスクおよび自動化コマンド・プロシーチャーが予定どおり処理されているかどうかを検証する際に役立ちます。

- 自動化のテスト中には、自動タスクのもとで実行するコマンド・リストのトレース機能をオンにしてください。このトレース機能は、コマンド・リスト・ステートメントが処理されるときにこれらのステートメントをネットワーク・ログに書き込みます。このログを分析し、問題の原因を判別することができます。使用できるトレース・コマンドの例として、REXX の場合は TRACE I、NetView コマンド・リスト言語の場合は &CONTROL ALL があります。自動化コマンド・リストを検証した後、ネットワーク・ログに無意味な情報が記録されるのを防ぐために、トレース機能をオフにすることができます。

- 自動タスクの ID にログオンし、自動タスクのコンソールを観察して、正しいアクションが取られていることを確認してください。自動タスクの ID にログオンしている間は、自動タスクに送信されるメッセージが必ず表示されるように **OVERRIDE DISPLAY=YES** コマンドを出すことができます。
- ネットワーク・ログを調べ、メッセージ **CNM493I** が出されているかどうかを調べてください。このメッセージは、コマンドまたはコマンド・プロシージャーが自動化テーブル・ステートメントにより処理されるたびにログに書き込まれます。**DEFAULTS** コマンド、**CNM493I** 自動化テーブル・アクション、または **OVERRIDE** コマンドを使用すると、このメッセージがログに書き込まれないようにすることができます。このメッセージから、コマンドがスケジュールされた時間とどのタスクでコマンドを実行しようとしたかなどの情報を判別することができます。
- ネットワーク・ログを調べ、メッセージ **DWO032E** が出されているかどうかを調べてください。このメッセージは、コマンドまたはコマンド・プロシージャーが処理されたはずなのに、そのコマンドまたはコマンド・プロシージャーが実行すべきタスクがログオンされていなかったとき、ログに書き込まれます。また、このメッセージは、NetView 自動化テーブルで自動化することができます。自動化をテストする際に、このメッセージを使用すると、正しくスケジュールされなかった自動化アクションを判別することができます。

自動タスクが活動化され、作業できるようにするなんらかの自動タスク・チェック機能を組み込んでください。自動タスクがループで停止するか、永久待ち状態になるか、またはログオフされたために非アクティブになると、問題を識別したり解決したりすることが難しくなります。拡張自動化サンプル・セット (651 ページの『付録 I. 自動操作のサンプル・セット』を参照) は、タスクが所定の時間より長時間にわたり非アクティブであるかどうかを自動的に通知する方法を用意します。この方法では、タイマー・コマンドを使用して、各自動タスクにコマンドを定期的に送信しグローバル変数を設定して、肯定応答が受け取られるはずであることを示します。自動タスクが再びチェックされるときになっても最後のチェックからの通知が受信されないと、問題が発生していることを示すメッセージがシステム・オペレーターに送信されます。サンプル・セットの自動タスクはすべて 1 つのマスター自動タスクによってチェックされますが、このマスター自動タスクは、PPT によってチェックされます。NetView プログラムの実行中には、PPT は常にアクティブです。

デバッグ・ツールの使用

オペレーティング・システムおよび NetView プログラムにはいくつかの監査証跡およびツールが備えられていて、自動化が期待どおりに行われているかどうかを判別したり、予定どおりに行われていないときには援助機能として使用することができます。以下のセクションでは、システム・ログとネットワーク・ログ、自動化していないメッセージと MSU の評価、NetView 自動化テーブルのリスト、および NetView 自動化テーブルのトレースについて説明します。ログについて詳しくは、549 ページの『第 35 章 ロギング』を参照してください。

ログの使用

MVS システム・ログは、SYS1.MACLIB データ・セット中の IHAHCLOG マクロによりマップされます。IHAHCLOG マクロでは、HCLREQFL と呼ばれる 8 バイ

トのフィールドにインストール・システム出口およびメッセージ処理機能 (MPF) 要求フラグがあります。抑止フラグ (バイト 7 および 8) のビット 10 は、MVS がそのメッセージの自動化を要求したかどうかを示しています。通常は、そのメッセージの MPF 項目に AUTO(YES) を指定するかまたはそのメッセージに適用される MPF の .DEFAULT または .NO_ENTRY ステートメントに AUTO(YES) を指定するかのいずれかによって、このビットを設定します。MPF 項目は、SYS1.PARMLIB の MPFLSTxx メンバーに定義します。NetView プログラムがメッセージを処理するのは、自動化要求フラグがオンのときだけです。

メッセージが自動化されていないときには、まずシステム・ログを調べ、自動化要求フラグがオンであることを確認する必要があります。ビット 1 は、MPF がメッセージを表示しないようにしたかどうかを示します。メッセージ抑止の効果を判別するには、このフラグが重要になります。

DEFAULTS コマンド、CNM493I 自動化テーブル・アクション、または OVERRIDE コマンドを使用してメッセージがログに書き込まれないようにしない限り、自動化テーブルがコマンドを生成するたびに、NetView プログラムは CNM493I メッセージをネットワーク・ログに書き込みます。このメッセージの重要なパラメーターはシーケンス番号です。このシーケンス番号は、自動化テーブル・ステートメントの 73 桁から 80 桁目から取り出されます。CNM493I メッセージを利用できるように、自動化テーブル項目にシーケンス番号が割り当てられていることを確認し、テーブル更新時にはこの番号付けのまま更新するようにしてください。

また、CNM493I は、自動化テーブル項目の EXEC 部分のオペランドとして生成されたコマンドを示しています。CNM493I は、処理された内容、およびコマンド・リストまたはコマンド・プロセッサに渡されたパラメーターを通知します。ただし、自動化コマンド・リストおよびコマンド・プロセッサは、自動化テーブル以外の箇所から呼び出すことができます。例えば、別のコマンド・リストおよびコマンド・プロセッサがそれらを呼び出すこともあり、NetView オペレーターあるいはシステム・オペレーターがそれらを開始することもあります。

したがって、各自動化プロシーチャーの先頭に MSG LOG ステートメントを指定して、呼び出されるプロシーチャーの名前およびそのプロシーチャーに渡されたパラメーターをネットワーク・ログに記録することができます。拡張自動化サンプル・セットは、この方法を示しています。コマンド・プロシーチャーがデバッグ・モードであるかどうかをチェックする際に使用する NetView グローバル変数を使用することもできます。この場合、コマンド・プロシーチャーから実行に関する追加情報が得られます。

コマンド・リストに指定される &CONTROL CMD ステートメントもツールとして使用することができます。このツールを使うと、コマンド・リストで出されるすべてのコマンドが表示されます。コマンド・ループ内に &CONTROL ERR を設定することもできます。また、このツールが指定されると、図 177 に示すメッセージがネットワーク・ログに書き込まれます。

```
DSI013I COMMAND LIST cmdlistname COMPLETE
```

図 177. &CONTROL CMD ステートメントによって書き込まれる DSI013I メッセージ

自動化していないメッセージおよび MSU の評価

テストおよびデバッグの一部として、自動化テーブルで自動化されないメッセージおよび MSU のリストを作成することができます。このリストを使用して、自動化を起動しなかったメッセージまたは MSU を検出することができます。また、自動化できるメッセージおよび MSU を判別する際にもこのリストを使用することができます。

例えば、コマンド・プロセッサ LOGSEQ を作成して、そのコマンド・プロセッサに渡される内容を順次ログ・データ・セットに記録するときには、自動化テーブルの最後のステートメントとして図 178 に示されるステートメントを指定することで、すべてのメッセージを LOGSEQ コマンド・プロセッサに渡すことができます。

```
IF MSGID = ANYID THEN  
    EXEC(CMD('LOGSEQ ' ANYID ' NOT AUTOMATED'));
```

図 178. メッセージを LOGSEQ に渡すステートメント

処理されたメッセージのうち、自動化テーブルで一致しなかったメッセージがあるたびに、LOGSEQ は順次ログ・ファイルにレコードを書き込み、メッセージが自動化されなかったことを示します。これにより、その順次ログを分析し、自動化される予定のメッセージが自動化されているかどうかを調べることができます。また、この方法は MSU にも適用することができます。しかし、この方法では、自動化されていないメッセージおよび MSU だけがテーブルの最後に渡されることを想定しています。すべてのテーブルに適しているわけではありません。

また、同じ方法を BEGIN-END セクション内で使用して、メッセージまたは MSU が予定どおりに自動化されているかどうかを判別することができます。これにより、BEGIN-END セクションに、さらにステートメントを追加する必要があるかどうかを判別することができます。

MSGID 以外の条件項目にこの方法を使用して、自動化されていないメッセージおよび MSU に関するその他の情報を得ることができます。

NetView 自動化テーブル・リストの使用

AUTOTEST コマンドおよび AUTOTBL コマンドを使用して、自動化テーブルのリストを作成することができます。構文エラーは、このリストに示されます。また、このリストには組み込まれたすべての自動化メンバーおよび同義語の置き換えが示されるため、自動化ステートメントの順序を決めるのに適しています。また、自動化テーブル全体が 1 箇所に表示されるため、自動化テーブル内の論理エラーの防止およびデバッグに役立ちます。

自動化テーブルとそのリストの例が、273 ページの『自動化テーブル・リストの例』にあります。

NetView 自動化テーブル・トレースの使用

TRACE アクションを使用して、自動化テーブル全体を通してメッセージまたは MSU の処理をトレースできます。TRACE アクションでは、AIFR に対するトレース・タグと同時に、その AIFR がトレースの対象にされているインディケーターを

設定し、自動化テーブルによって AIFR が処理されるようにします。詳細トレース情報は、処理されるそれぞれの自動化テーブル・ステートメントの各部分ごとにメッセージ BNH370I によって表示されます。

以下の例での意図は、コマンドが 8 文字より長く無効であるというメッセージを、オペレーター OPER1 が受信しない理由を判別することです。コマンドが 8 文字より長い場合には、DSI002I 通知メッセージが通常のメッセージとしてではなく即時メッセージとして表示されるということに、留意してください。そのため、HDRMTYPE をチェックし、そのコマンド名が 8 文字より長いものであったかどうかを判別します。

```
SYN %HDRMTYPE_IMMED% = '!!!';

IF (LABEL:LABEL1) MSGID = 'DSI002I' THEN
  TRACE('DSI002_IMMED_TRC');

IF (LABEL:LABEL2) MSGID = 'DSI002I' &
  TOKEN(4) = CMDNAME &
  HDRMTYPE = %HDRMTYPE_IMMED% THEN
  EXEC(CMD('MSG OPER1 COMMAND' CMDNAME ' IS LONGER THAN 8 CHARS'));

IF (LABEL:LABEL3) MSGID = 'BNH370I' THEN
  COLOR(YEL);
```

メッセージ DSI002I が無効なコマンド (SHORTCMD) に対して出され、先行する自動化テーブル・セグメントによって処理される場合、以下のメッセージが生成されます。

SHORTCMD							1
BNH370I	PASS	TRACE	MAINTABL	INCLTABL	LABEL1	DSI002_IMMED_TRC	2
BNH370I	PASS	MSGID	MAINTABL	INCLTABL	LABEL2	DSI002_IMMED_TRC	3
BNH370I	PASS	TOKEN	MAINTABL	INCLTABL	LABEL2	DSI002_IMMED_TRC	4
BNH370I	PASS	AND	MAINTABL	INCLTABL	LABEL2	DSI002_IMMED_TRC	5
BNH370I	FAIL	HDRMTYPE	MAINTABL	INCLTABL	LABEL2	DSI002_IMMED_TRC	6
BNH370I	FAIL	AND	MAINTABL	INCLTABL	LABEL2	DSI002_IMMED_TRC	7
BNH370I	FAIL	MSGID	MAINTABL	INCLTABL	LABEL3	DSI002_IMMED_TRC	8
DSI002I INVALID COMMAND: 'SHORTCMD'							9

BNH370I メッセージは、先行する自動化テーブル・セグメントにおいて TRACE アクションで指定されていたトレース・タグ DSI002_IMMED_TRC に関するトレース結果を示しています。個々のメッセージの説明は、以下のとおりです。

キー 説明

- 1 コマンド SHORTCMD が入力されています。
- 2 LABEL1 ステートメントでの TRACE アクションは、生成メッセージ DSI002I がこのステートメントの条件と一致したので、正常に実行されました。これは、このメッセージに関するトレースは、自動化テーブル全体を通して処理を継続するので現在有効であることを意味しています。
- 3 MSGID 条件は LABEL2 ステートメント内で一致しています。これは PASS MSGID で示されています。
- 4 TOKEN 条件 (無効なコマンド名を変数に代入する際に使用) は、LABEL2 ステートメント内で一致しています。これは PASS TOKEN で示されています。
- 5 MSGID 条件と TOKEN 条件を結合する論理 AND は、LABEL2 ステートメント内で正常に動作しています。これは PASS AND で示されています。論

理 AND 演算子は、その 2 つのオペランド (MSGID および TOKEN) が正常終了したので正常に動作しています。

- 6** HDRMTYPE 条件は LABEL2 ステートメント内で失敗しています。これは FAIL HDRMTYPE で示されています。HDRMTYPE チェックは、コマンドが 8 文字より長くなかったので失敗しています。メッセージ DSI002I は、即時メッセージではなく通常のメッセージとしての出力です。
- 7** 論理 AND 演算子 (先行する論理 AND 演算子によって HDRMTYPE を結合する) は、LABEL2 ステートメント内で失敗しています。これは FAIL AND で示されています。論理 AND 演算子はその第 2 オペランド (HDRMTYPE) が失敗したので、失敗しています。
- 8** 先行するステートメントは結果が一致しなかったので、自動化テーブル処理は継続しています。MSGID 条件は LABEL3 ステートメント内で失敗しています。これは FAIL MSGID で示されています。
- 9** メッセージ DSI002I (ちょうど自動化テーブルで処理終了) がコンソールに表示されています。

メッセージ BNH370I はコンソールに対して出されます。ユーザーは、自動化ロジックを自動化テーブルに組み込み、望ましい場合はこれを NetView ログに送信することができます。メッセージ ID BNH370I の場合、TRACE アクションはループ状態を引き起こす原因となるので、このアクションを指定することがないようにしてください。BNH370I では、トレース・メッセージを容易に認識できるように、それを黄色に色付けすることが 1 つ前の例の中で自動化されていることに注意してください。

第 35 章 ロギング

本章では、システム、ネットワーク、またはユーザー・レベルで実施できるログ記録について説明します。トピックは、次のとおりです。

- ログに関する考慮事項
- ログの種類
- NetView プログラムで提供されるログ機能
- MVS システム・ログのデータとネットワーク・ログのデータとの相違点

ログ記録に関する考慮事項

通常、データはメッセージ形式で記録されます。しかし、ユーザーの環境で使用されているログの目的によっては、データがコマンド、プログラム・データ、またはその他の情報である可能性もあります。ログの一般的な用途には、次のようなものがあります。

- システム内で発生したイベントの監査証跡を提供します。監査証跡は、自動化プロセスの追跡、または操作上の問題が発生したときに非常に便利です。このような場合、すべての情報は適切で、読みやすく、しかも使いやすい形式で保管しておかなければなりません。監査証跡は、トレースとは見なされません。問題判別に役立つログの使用法については、543 ページの『ログの使用』を参照してください。
- システムの操作特性を報告します。例えば、管理者は、システムの自動化の効率に関する報告書を要求するとします。このデータは、自動化が使用されていない時点で収集されたデータと比較することができます。

ログの価値は、その中に記録されているデータによって決まります。システム・オペレーターは、保持されるデータとそのデータをどのログに書き込むかなどを制御することができます。したがって、個々の環境に最適なログ記録方法を決定してください。重要でないデータは、ログに書き込まないでください。役立つと思われるデータだけを保持すれば、ログは読みやすく、ログ記録によるパフォーマンス上のオーバーヘッドが最小になります。

NetView プログラムは、自動化テーブルで一致条件が発生して、コマンドまたはコマンド・プロシーチャーがスケジュールされるたびに、ネットワーク・ログに CNM493I メッセージを書き込みます。DEFAULTS コマンド、CNM493I 自動化テーブル・アクション、または OVERRIDE コマンドを使用して、このメッセージをログに書き込まないようにすることもできます。

メッセージ CNM493I のログを記録しないようにするための詳細については、「*IBM Tivoli NetView for z/OS*チューニング・ガイド」を参照してください。メッセージ CNM493I の形式は、図 179 のようになります。

CNM493I member : seqnum : commandtext

図 179. メッセージ CNM493I の形式

549 ページの図 179 では、*member* は、コマンドまたはコマンド・プロシージャーをスケジュールしたステートメントの自動化テーブル・メンバーであり、*seqnum* は、ステートメントのシーケンス番号 (ステートメントにシーケンス番号がないときには、(NO SEQ)) であり、*commandtext* は、スケジュールされたコマンドまたはコマンド・プロシージャー (パラメーターも含む) です。

メッセージ CNM493I を使用すると、自動化の使用方法を分析したり、自動化処理が予定どおりに動作しているかどうかを判別する際の監査証跡として使用することができます。ただし、メッセージ CNM493I は、コマンドまたはコマンド・プロシージャーがスケジュールされたことだけを示し、実際に処理されたかどうかについては示しません。コマンドまたはコマンド・プロシージャーの送信先タスクがアクティブでなければ、そのコマンドまたはコマンド・プロシージャーは処理されず、メッセージ DWO032E がログに記録され、自動化テーブルから送信されます。メッセージ DWO032E は、非アクティブのタスクの名前および処理されなかったコマンドまたはコマンド・プロシージャーを示します。

CNM493I メッセージ以外にも、記録するデータについては、考慮が必要です。例えば、次のタイプのデータをログに記録することができます。

- ・ モニター・アプリケーションの一部としてデータを収集する場合も、なんらかの処理が行われると、すべてのメッセージをログに記録します。
- ・ 重要な自動化プロシージャーが実行されるたびに、コマンド・リスト名およびパラメーター・ストリングをログに記録します。メッセージ CNM493I は、自動化テーブルで一致条件が発生すると実行する最初のプロシージャーだけに出されます。メッセージ CNM493I は、引き続きそのプロシージャーによって呼び出される他のコマンド・リストおよびコマンド・プロセッサには出されません。
- ・ 以前にオペレーターが行った操作を直接自動化するアクションを取るたびに、ログにレコードが記録されたことを確認します。具体的には、「自動化アクション」を認識しやすくするようにレコードを記録し、管理統計情報の作成に役立てることができます。

オペレーティング・システムは、システム・メッセージのログを制御します。可能であれば、システム・レベルでシステム・メッセージのログ記録を抑止してください。NetView プログラムによって処理されるメッセージを抑止したり、ハードコピー・ログだけでなくシステム・ログ、ネットワーク・ログ、または各種のユーザー提供ログに送信することができます。

MVS システム・ログ (SYSLOG)

MPFLSTxx 項目で抑止されるメッセージも含めて、すべての MVS オペレーター宛 (WTO) メッセージは、MVS システム・ログ (SYSLOG) に記録することができます。システム・ログへの記録を抑止できるのは、ユーザー作成の MPF インストール・システム出口からです。NetView メッセージは、MVS システム・ログへ送ることができます (551 ページの『ネットワーク・ログ』を参照)。メッセージの抑制は、30 ページの『メッセージ改訂テーブル』で説明しているメッセージ改訂テーブルを使用して行うこともできます。

MVS システム・ログにメッセージを記録するには、JES がアクティブでなければなりません。

注: メッセージ改訂機能によって抑制されたメッセージは、MVS SYSLOG に書き込まれます。メッセージ改訂機能によって削除されたコメントは、MVS SYSLOG には書き込まれません。

ネットワーク・ログ

NetView メッセージは、デフォルトでネットワーク・ログに書き込まれるようになっています。しかし、MVS サブシステム・インターフェースから受信された非送信請求メッセージは、それが ASSIGN によってタスクに割り当てられず、その自動化テーブル項目も存在しないとき、ネットワーク・ログには書き込まれません。

VTAM メッセージをネットワーク・ログに記録するには、VTAM 開始パラメーター PPOLOG=YES を使用してください。これにより、拡張複数コンソール・サポート (EMCS) コンソールから入力されるすべての VTAM コマンド (START および HALT を除く) とすべての VTAM 応答は、ネットワーク・ログに記録されます。このとき、サブシステム・インターフェースのルーター・タスクを開始したタスクでの DEFAULTS および OVERRIDE の指定には影響されません。

メッセージをネットワーク・ログに記録する前に、ネットワーク・ログ・タスク (DSILOG) はアクティブでなければなりません。システム・メッセージを受信するために NetView サブシステム・アドレス・スペースをアクティブにしておかないと、システム・メッセージをネットワーク・ログに記録することはできません。

NetView BROWSE コマンドを使用すると、ネットワーク・ログを表示することができます。NetView BROWSE コマンドを介して使用できるコマンドは、ISPF BROWSE コマンドと類似しています。自動タスクに関連付けられている EMCS コンソールから、あるいは NetView 間タスク (NNT) は、BROWSE を使用することはできません。

ネットワーク・ログ機能は、実際にログにメッセージを書き込む前に、メッセージをバッファに入れます。デフォルトの処理を使って、NetView プログラムは、バッファがいっぱいになった時点でメッセージをネットワーク・ログに書き込みます。初期設定オプションを指定すると、据え置き書き出し (DFR) を使用していますが、ネットワーク・ログ入出力は順次で、挿入処理も削減処理もないため、通常の処理と大きな違いはありません。

ユーザー作成ログ

順次アクセス方式のログ・サポート機能を使用すると、1 つ以上の順次ログ・タスクを定義し、ユーザー定義ログに可変長レコードを書き込むことができます。アセンブラ (DSIWLS を使用)、PL/I、または C (CNMSMSG を使用) で作成されたコマンド・プロセッサから、順次ログ・タスクヘータを送信することができます。詳しくは、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。

高水準言語でコマンド・プロセッサを作成する場合、PL/I および C の入出力サービスも使用することができます。これらのサービスを使用すると、ログ方法を作成することができます。さらに、これらのサービスを ALLOCATE および FREE コマンドとともに使用すると、NetView プログラムの中から簡単にデータ・セットを割り振ったり、解放したりすることができます。

NetView のログ機能

NetView プログラムを使用すると、メッセージを MVS システム・ログ、ネットワーク・ログ、または各種のユーザー作成ログへ渡すかどうかを指定することができます。ユーザー作成ログは、ユーザー作成プログラムからしかアクセスできません。

NetView プログラムは、次の規則に従って、メッセージをログに記録します。

- NetView DEFAULTS コマンドは、他にログ指定項目が指定されていないときのメッセージのログ方法を決定します。DEFAULTS コマンドは任意の NetView ユーザーが入力できるため、許可を受けていないオペレーターが使用するのを避けるために、許可についてチェックする必要があります。CNMSTYLE メンバーの値を指定して、NetView プログラムの初期化時に希望するオプションが確実に有効になるようにしてください。
- 各 NetView オペレーターは OVERRIDE コマンドを使用して、特定のオペレーター端末タスク (OST) へ送られるメッセージのデフォルト値をセットアップすることができます。
- 自動化テーブルに SYSLOG(Y) と NETLOG(Y) を指定し、どのメッセージ (またはメッセージ・セット) をシステム・ログおよびネットワーク・ログへ書き込むかを決めることができます。

MVS メッセージに SYSLOG(Y) を指定したが、そのメッセージが MVS によってすでにシステム・ログ・データ・セットに書き込まれているときには、NetView プログラムは余分なコピーを書き込みません。

SYSLOG(Y) を指定してシステム・ログへ書き出されたメッセージには、そのメッセージ・テキストの先頭に、次の 3 つの NetView フィールドが書き込まれます。そのフィールドとは、ドメイン ID、オペレーター ID、およびメッセージ・タイプ記号 (例えば、ダッシュはコマンド機能メッセージであることを指定) です。実際のメッセージ ID は、メッセージ・テキストの 4 番目のトークンです。

NetView プログラム内で生成されるメッセージも、NetView MSG コマンドを使用してログに記録することができます。LOG に宛先を指定すると、デフォルト (DEFAULTS) 値および指定変更 (OVERRIDE) 設定値に従い、メッセージはネットワーク・ログ (そのログがアクティブの場合) へ書き込まれます。SYSOP を指定すると、メッセージはシステム・ログに書き込まれ、EMCS コンソールに表示されます。

コマンド・プロセッサをアセンブラーで作成するユーザーには、同じような機能 (DSIWLS) が用意されています。ユーザーはコマンド・プロセッサを高水準言語 (特に、PL/I または C) で作成することができます。高水準言語のサービス・ルーチン CNMSMSG を使用して、ユーザーはメッセージをネットワーク・ログ、ユーザー作成順次ログ、または SMF などの外部ログへ書き込むことができます。SYSOP を指定すると、メッセージはシステム・ログに書き込まれ、EMCS コンソールに表示されます。

MVS システム・ログと NetView ネットワーク・ログのレコード

MVS システム・ログに記録されるデータとネットワーク・ログに記録されるデータとでは、いくつかの相違点があります。メッセージ・テキストは両方のログに書き込まれますが、MVS WTO メッセージの場合、システム・ログにはメッセージの発信元 (ジョブ番号、コンソール) および後処理 (経路コード、MPF アクションなど) に関する情報も書き込まれます。

また、システム・ログは記録されたメッセージ・タイプ (複数行、コマンド応答など) も示します。システム・ログは、メッセージが出されたシステムを示します。これに対し、ネットワーク・ログはメッセージが最初に受信されたドメインを示します。通常、ドメインの名前とシステムの名前は簡単に関連付けられるように選択されます。

JES3 グローバル・プロセッサの場合、複合体内の異なったプロセッサから送られてくるメッセージは、すべてグローバル・プロセッサへ送られます。したがって、グローバル・プロセッサからだけサブシステム・インターフェース・メッセージを取り出すときには、すべてのメッセージのドメイン・ネームが同じでシステム名が記録されていないために、ネットワーク・ログでメッセージの発信元を識別することは困難です。

MVS メッセージの場合、システム・ログは、メッセージが出された時間を示しています。これに対し、ネットワーク・ログは、そのメッセージに対する自動化テーブル処理が実行された時間 (大幅に遅れる可能性もある) を示しています。例えば、自動化テーブルを処理している NetView タスクが、AUTOWRAP NO コマンドが指定されているために現在オペレーターの入力待ち状態である場合、通常のタスク処理が再開されるまで、そのタスクはメッセージを処理しません。NetView タスクがセッション・モニターまたはヘルプ・パネルなどのフルスクリーン・アプリケーションにある場合も、この状態が発生します (ただし、NetView タスクが状況モニターとのセッション中にある場合には、発生しません)。

ネットワーク・ログへのメッセージの書き込みは、あとになります。自動化テーブル処理の結果、NetView プログラムがシステム・ログに書き込むメッセージについても、同じような遅延状態が発生します。したがって、メッセージはその発生原因の順序とは異なる順序でネットワーク・ログに書き込まれることがあるため、問題判別が困難になることがあります。NetView プログラムにおけるメッセージの流れを慎重に設計し、自動化テーブル内で ASSIGN コマンドおよび ROUTE オペランドを指定すれば、これらの問題を回避することができます。

第 36 章 ジョブ入力サブシステム 3 (JES3) の自動化

ジョブ入力サブシステム 3 (JES3) 複合体は、いくつかのチャンネル間 (CTC) 接続プロセッサで構成されることもあります。オペレーターは、これらのプロセッサを単一システムとして扱うことができます。JES3 複合体の操作制御は、1 つのプロセッサ (グローバル・プロセッサ) により実行されます。グローバル・プロセッサは、ジョブ入力、ジョブに必要なリソースの制御、および複合体に属するプロセッサへの作業の配分などを行う際の中心点になります。複合体が複数のプロセッサで構成されている場合、グローバル・プロセッサ以外のプロセッサは、ローカル・プロセッサと呼ばれます。

複合体に含まれるプロセッサ数により、操作制御は過酷な作業になる場合があります。複合体はグローバル・プロセッサに接続されているコンソールを通して操作され、すべてのプロセッサから送られてくるメッセージは、これらのコンソールに表示されます。システム・ログの処理も、グローバル・プロセッサで行われます。複合体のプロセッサを制御するオペレーター・コマンドは、グローバル・プロセッサに接続されているコンソールから出されます。

グローバル・プロセッサを自動化のフォーカル・ポイントにしてください。NetView プログラムは、そのグローバル・プロセッサで実行する必要があります。JES3 複合体に含まれるすべてのプロセッサから送られてくるメッセージは、グローバル・プロセッサのサブシステム・インターフェースに表われます。コマンドは、グローバル・プロセッサからすべてのプロセッサへ送信することができます。したがって、NetView プログラムをグローバル・プロセッサにインストールすれば、JES3 複合体を自動化することができます。しかし、リカバリー手順を自動化したいのであれば、NetView プログラムをローカル・プロセッサにインストールすることも検討してください。

操作タスクを自動化する場合、コンソールの使用について構成を同じに保つ必要はありません。MPF および自動化アクションによりメッセージの通信量が減少するときには、いくつかの機能を同じコンソールに統合することができます。

JES3 複合体でのメッセージ・フロー

以下のセクションでは、グローバル・プロセッサで生成されるメッセージ・フローとローカル・プロセッサで生成されるメッセージ・フローについて説明します。

グローバル・プロセッサで生成されるメッセージ

グローバル・プロセッサから出される WTO メッセージはすべて、グローバル・プロセッサで MPF 処理の対象になり、抑止、保存、および自動化のインディケータは作業キュー・エレメント (WQE) 内で設定されます。

- メッセージ ID が MPFLSTxx に記載されている場合、
 - 特定の AUTO、SUP、RET、および USEREXIT 指定値が存在する場合、それらの値が使用されます。

- メッセージ ID 項目に特定の定義が行われていない場合 (つまり、 AUTO および SUP の場合) には、.DEFAULT 項目が使用されます。
- .DEFAULT 項目が存在しないときは、デフォルト値 AUTO (NO) および SUP (YES) が使用されます。
- メッセージ ID の MPFLSTxx に項目が存在しない場合、
 - .NO_ENTRY ステートメントの定義が使用されます。
 - .NO_ENTRY ステートメントが存在しないときには、システム・デフォルト値 AUTO(YES) および SUP(NO) が使用されます。
- 次に、メッセージは MPF 出口へ渡されます。この出口では、抑止指定および自動化指定が変更される場合があります。
 - メッセージ ID に MPF 出口が指定されていると、MPF 出口ルーチンが処理されます。
 - MPF 出口は指定されていないが、IEAVMXIT 出口ルーチンが存在するときには、IEAVMXIT 出口ルーチンが処理されます。

MPF 処理が完了したら、MVS はメッセージをグローバル・プロセッサのサブシステム・インターフェースに書き込みます。各サブシステムでは、サブシステム名テーブルによって決められた順序に従ってメッセージを検査します。JES3 が主基本サブシステムであるため、メッセージを最初に検査します。

JES3 は、メッセージに対して MSGROUTE 処理を実行し、出口 57 (IATUX57) および出口 69 に渡します。

注: JES3 は、複数コンソール・サポートにおけるメッセージの表示に影響を与えることがあります。つまり、メッセージは MPF では抑止されなくても、JES3 により抑止されることがあります。

NetView プログラムによるメッセージ改訂処理の実行後には、次のようになります。その処理で削除アクションが指定された場合は、メッセージがサブシステム・インターフェースに書き込まれた後、そのメッセージに対してはそれ以上の処理は行われません。NETVONLY アクションが指定された場合は、そのメッセージは自動化のために直接 CNMCSSIR に送られますが、それ以外の場合は削除メッセージと同様に処理されます。

自動化要求が MPF またはメッセージ改訂のいずれかから出された場合、NetView プログラムは自動化のためにそのメッセージをサブシステム・インターフェースからコピーします。

最後に、メッセージはサブシステム・インターフェースから複数コンソール・サポートへと流れ、宛先コード、MPF 指定値、メッセージ改訂の処理、およびサブシステム・インターフェースからの戻りコードに応じて、表示されたり、保存されたり、ハードコピー・ログに記録されたりします。

ローカル・プロセッサで生成されるメッセージ

ローカル・プロセッサから出される WTO メッセージはすべて、ローカル・プロセッサで MPF 処理の対象になり、抑止、保存、および自動化のインディケータは作業キュー要素 (WQE) 内で設定されます。

- メッセージ ID の MPFLSTxx 項目が存在する場合、

- その AUTO、SUP、RET、および USEREXIT 定義が使用されます。
- 項目が存在していても定義されていないときは、.DEFAULT 項目が使用されます。
- .DEFAULT 項目が存在しないときは、デフォルト値 AUTO (NO) および SUP (YES) が使用されます。
- メッセージ ID の MPFLSTxx に項目が存在しない場合、
 - .NO_ENTRY ステートメントの定義が使用されます。
 - .NO_ENTRY ステートメントが存在しないときには、システム・デフォルト値 AUTO(YES) および SUP(NO) が使用されます。
- 次に、メッセージは MPF 出口へ渡されます。
 - このメッセージ ID に MPF 出口が指定されていれば、その出口が処理されます。
 - MPF 出口は指定されていないが、IEAVMXIT 出口ルーチンが存在すれば、IEAVMXIT 出口ルーチンが処理されます。

このあと、メッセージはローカル・プロセッサの MVS サブシステム・インターフェースでブロードキャストされます。各サブシステムでは、サブシステム名テーブルによって決められた順序に従ってメッセージを検査します。JES3 が主基本サブシステムであるため、メッセージを最初に検査します。

JES3 は、MSGROUTE 処理を実行し、メッセージを出口 57 (IATUX57) および出口 69 (IATUX69) に渡します。

NetView プログラムは、すべてのメッセージ改訂処理を実行します。

NetView サブシステムがローカル・プロセッサで実行していて、メッセージに AUTO(YES) が指定されているか、あるいは MPF のデフォルト値として AUTO(YES) が採用されているときには、NetView プログラムは、自動化を実行する目的で、そのメッセージをコピーします。

サブシステム・インターフェースの処理後、経路コード、抑止インディケータと保存インディケータ、サブシステム・インターフェースの戻りコード、および JES3 MSGROUTE 指定値に従い、メッセージの表示、保存、およびハードコピー・ログのために、メッセージはローカル・プロセッサの複数コンソール・サポートへ渡されます。

ローカル・プロセッサのコンソール・サポートは、メッセージをコピーし、コピーされたメッセージを XCF 接続を介してグローバル・プロセッサへ渡します。JES3 は、ローカル・プロセッサの元のメッセージの抑止、自動化、または保存インディケータを変更しません。

グローバル・プロセッサ上のコンソール・サポートは、ローカル・プロセッサ上の XCF 接続からメッセージを受け取り、それを次のように処理します。

- メッセージを MPF 出口に渡します (JES3 GLOBMPF パラメータによって要求された場合)。
 - メッセージ ID に MPF 出口が指定されていると、MPF 出口ルーチンが処理されます。

- MPF 出口は指定されていないが、IEAVMXIT 出口ルーチンが存在するときには、IEAVMXIT 出口ルーチンが処理されます。

サブシステム・インターフェース処理後、経路コード、MPF 指定値、およびサブシステム・インターフェースの戻りコードに従って、グローバル・プロセッサでのメッセージの表示、保存、およびハードコピー・ログのために、メッセージは複数コンソール・サポートに渡されます。

JES3 環境におけるコマンド

このトピックでは、JES3 環境でコマンドを出すいくつかの方法について説明します。

NetView プログラムからの JES3 コマンドの発行

JES3 では、グローバル・プロセッサの NetView プログラムからコマンドを出すことができます。NetView から JES3 コマンドを入力する際には、JES3 ID (例えば、*) を指定する必要があります。JES3 コマンドを NetView プログラムから入力するとき、接頭部 MVS を使用することができます。また、NetView サンプル・メンバー CNMS6403 の項目を DSIPARM の CNMCMD メンバーに追加すれば、MVS 接頭部を付けずに NetView プログラムから JES3 コマンドを入力することができます。その方法を使用すれば、NetView オペレーターは MVS コマンドを出す許可が与えられていなくても、特定の JES3 コマンドを出すことができます。MVS 接頭部を付けずに NetView から JES3 コマンドを出すとき、コマンドのうしろに、コンマではなくスペースを指定してください。

JES3 コマンド verb はセキュリティの対象となりますが、コマンドのキーワードと値は、セキュリティの対象ではありません。例えば、NetView コマンド許可テーブルまたはシステム許可機能 (SAF) プロダクト (RACF (リソース・アクセス管理機能) など) を使用すると、*SEND コマンドをコマンド許可で保護することができます。ただし、このコマンドを使用して送信されるものは、保護の対象になりません。コマンド許可については、「*IBM Tivoli NetView for z/OS*セキュリティ・リファレンス」を参照してください。

グローバル・プロセッサの NetView から出される JES3 コマンドは、サブシステム・インターフェースに渡されたあと、JES3 に渡されます。NetView から出された JES3 コマンドの結果として出されるメッセージは、サブシステム・インターフェースに渡されますが、NetView は、このインターフェースでそれらのメッセージにアクセスすることができます。これらのメッセージは、コマンドの応答として扱われず、NetView では非送信請求メッセージとして扱われます。JES3 は複数行の WTO (MLWTO) を出しませんので、コマンドの応答として表示されるメッセージ・グループのように見えるメッセージは、実際にはいくつかの別個の非送信請求メッセージで構成され、NetView 自動化機能では、それぞれ別個の非送信請求メッセージとして処理しなければなりません。

大部分の JES3 コマンドは、ローカル・プロセッサから出すことはできません。ローカル・プロセッサの NetView から出すことができる JES3 コマンドは、次のコマンドだけです。

- *CALL,DSI
- *CALL,VARYL

- *START,DSI
- *START,VARYL
- *CANCEL,DSI
- *CANCEL,VARYL
- *DUMP
- *RETURN

ローカル・プロセッサ上で稼働中の NetView は、JES3 コマンドをグローバル・プロセッサに送信し、それに対する応答を MVS ROUTE コマンドを使用して受信できます。

JES3 複合体の NetView からの MVS コマンドの発行

JES3 環境では、ローカル・プロセッサまたはグローバル・プロセッサの NetView が、そのプロセッサで実行される MVS コマンドを出すことができます。

グローバル・プロセッサ上の NetView は、JES3 *SEND コマンドを使用してシステム・コマンド (つまり、MVS コマンド) をローカル・プロセッサに送信し、そのコマンドに対する応答を受信することができます。ローカル・プロセッサへの MVS コマンドの送信とそれに対する応答の受信は、MVS ROUTE コマンドを使用して行うこともできます。

JES3 複合体におけるオペレーティング・システム・コンソールからの NetView コマンドの発行

NetView コマンドは、次の場合に MVS コンソールから入力することができます。すなわち、NetView サブシステム指定文字が接頭部として NetView コマンドに付けられ、自動タスクがすでに MVS コンソールと関連付けられている場合 (つまり、CONS オペランドが指定された NetView AUTOTASK コマンドが出され、そのオペランドに、NetView コマンドを入力するコンソールの番号を指定した場合) です。

JES3 環境における NetView プログラム

NetView 自動化コマンド・プロシーチャーを JES3 環境で使用する際には、次の考慮する必要がある項目がいくつかあります。

- サブシステム・インターフェース上のメッセージについては、NetView メッセージ改訂は以下のものを変更できます。
 - JES3 ログ
 - 複数コンソール・サポート・コンソールのメッセージの経路指定/表示
 - 複数コンソール・サポートのハードコピー・ログ

MPFLSTxx メンバーに AUTO(YES) が指定されると、NetView プログラムは自動化を行うためにメッセージのコピーを作成します。

- 各プロセッサで不要なメッセージを抑止し、自動化の対象になるすべてのメッセージを自動化するように、MPFLSTxx を設定してください。メッセージは、自動化のためにグローバル・プロセッサへ送られます。自動化 NetView プログラムは、複合体にある任意のプロセッサにコマンドを出すことができます。

- 自動化 NetView プログラムはいくつかのプロセッサからメッセージを受信するため、コマンド・リストでは、SYSID() 関数 (REXX) または &SYSID 制御変数 (NetView コマンド・リスト言語) をチェックして、どこからメッセージが送られてきたかを検査したあと、*SEND または MVS ROUTE コマンドを使って特定のプロセッサにコマンドを送り返す必要があります。

JES3 がアクティブでなければ、SYSID() または &SYSID の値は、IEASYSxx メンバー (GRS 名) の SYSNAME パラメーターにより定義されている名前です。JES3 がアクティブであれば、SYSID() または &SYSID は、MAINPROC ステートメントに指定されている JES3 プロセッサの名前です。したがって、それらの名前を同じ名前に設定するか、または容易に認識できるように関連付けてください。

- JES3 は複数行メッセージ (MLWTO) を使用しません。JES3 は、コマンドの応答として同じメッセージを複数回出すことがあります。コマンド・リストは応答としてこれらのメッセージをすべて受信し、通常は、すべてのメッセージを待ちます。しかし、コマンド・リストが受信するメッセージの数を、あらかじめ調べることは困難です。それを調べる方法としては、次の方法が考えられます。
 - まず、エレメントの数 (例えば、特定のキュー上のジョブの数) を調べるために、JES3 コマンドを出します。次に、さらに詳しいコマンドを出すと、送られてくるメッセージの数を予測することができます。
 - 受信したいメッセージの最大数を指定するために、*INQUIRY コマンドで N オペランドを指定します。
- コマンド・プロシージャが、JES3 コマンドに応答して返される送信請求メッセージを待つようにするには、常にコマンドを次のように指定してください。
 - NetView コマンド・リスト言語で作成されたコマンド・リストでは、&WAIT 制御ステートメントにコマンドを指定します。
 - REXX で作成されたコマンド・リストでは、TRAP 命令のあとにコマンドを指定します。
 - 高水準言語 (HLL) で作成されたコマンド・プロセッサでは、TRAP コマンドのあとにコマンドを指定します。

上記のように指定すれば、送信請求メッセージは必ず戻されます。JES3 コマンドを &WAIT または TRAP の前に指定すると、早く戻される応答があるため、&WAIT または TRAP がそれらの応答を受信できない場合があります。

- JES3 メッセージが繰り返し出されると、自動化コマンド・リストが 2 回起動されることがあります。最初のメッセージは、自動化テーブルの REPLYID が正しく設定されたアクション・メッセージです。2 番目のメッセージはアクション・メッセージではありませんが、同じメッセージ ID が割り当てられています。この場合、REPLYID 条件項目は設定されません。自動化コマンド・プロシージャは、2 番目のメッセージが実際のアクション・メッセージかどうかをテストする必要があります。正しい応答 ID を得るには、次のようにしてください。
 - NetView コマンド・リスト言語で作成されたコマンド・リストでは、&REPLYID 制御変数を使用します。
 - NetView REXX コマンド・リストでは、REPLYID 機能を使用します。
 - HLL コマンド・プロセッサでは、CNMGETA サービス・ルーチンを使用します。

- ネットワーク・ログでは、どのプロセッサがメッセージを出したかを示しません。システム・オペレーターが出口ルーチンをカスタマイズするとき、ネットワーク・ログに書き込むメッセージのシステム ID を組み込むようにすることができます。

第 37 章 SNMP トラップの自動化

この章では、SNMP トラップを調整して、NetView 自動化に渡される SNMP トラップの自動化 CP-MSU にする方式を説明します。

TCP プロトコルでもユーザー・データグラム・プロトコル (UDP) でも、完全な SNMP トラップを受け取った場合には、SNMP トラップの自動化タスクによって常に CP-MSU が作成され、NetView 自動操作に渡されます。CP-MSU 内には、トラップからのデータが入った GDS 変数があります。

SNMP トラップの自動化タスク

SNMP トラップの自動化タスクは、SNMP トラップの受信と自動化に使用される NetView データ・サービス・タスク (DST) です。SNMP トラップの自動化タスクは、並行サーバー (TCP クライアントの SNMP エージェントの場合)、データグラム受信側 (UDP でトラップを送信する SNMP エージェントの場合)、またはその両方として、セットアップすることができます。SNMP トラップの自動化タスクを使用して、IPv4 ネットワークと IPv6 ネットワークの両方の SNMPv1、SNMPv2c、および SNMPv3 トラップの受信および自動化を行うことができます。

SNMP トラップの自動化の作業負荷を分散するために、複数の SNMP トラップの自動化タスクを使用することができます。NetView プログラムのインスタンス内の各 SNMP トラップの自動化タスクには、固有のタスク名が必要ですが、そのすべてに同じ DST 初期化メンバー (サンプル CNMTRAPI) が使用されます。

SNMP トラップの自動化タスクは、トラップを CP-MSU に変換してから NetView 自動化に渡すことによって、SNMP トラップを自動化します。MSU の自動化は、361 ページの『第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化』で説明します。SNMP トラップの自動化タスクが作成する CP-MSU の内容は、この章で説明します。

SNMP トラップの自動化タスクの構成

SNMP トラップの自動化タスクは、特定のタスクに固有の共通グローバル変数を設定することによって構成します。これらの変数は、SNMP 自動化タスクによって読み取られます。これらの変数の詳細については、「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」に記載されています。SNMP トラップの自動化タスクによって読み取られる共通グローバル変数は次のとおりです。

CNMTRAP.taskname.CONFIGFILE

この変数は、SNMPv3 トラップ処理情報が入った構成ファイルの名前を提供します。

CNMTRAP.taskname.MAXTCPCONN

この変数は、SNMP トラップの自動化タスクによってサポートされている最大 TCP 接続数を提供します。

CNMTRAP.taskname.STACKNAME

この変数は、SNMP 自動化タスクがアフィニティーを得る TCP/IP スタックの名前を提供します。

CNMTRAP.taskname.TCPPORT

この変数は、クライアントが接続して SNMP トラップを送信できる並行サーバー (TCP) のポート番号を提供します。

CNMTRAP.taskname.TRACE

この変数は、SNMP 自動化タスクのトレースを使用可能または使用不可に設定します。

CNMTRAP.taskname.UDPPORT

この変数は、エンティティーが SNMP トラップ・データグラム (UDP) を送信できるポート番号を提供します。

例として、次の CNMTRAPD という名前の SNMP トラップの自動化タスクを構成する定義を検討します。

```
•  
  
COMMON.CNMTRAP.CNMTRAPD.STACKNAME = &CNMTCPN  
•  
  
COMMON.CNMTRAP.CNMTRAPD.TCPPORT = 162  
•  
  
COMMON.CNMTRAP.CNMTRAPD.UDPPORT = 162  
•  
  
COMMON.CNMTRAP.CNMTRAPD.CONFIGFILE =  
    /usr/lpp/netview/v5r3/cnmtrapd.conf  
•  
  
COMMON.CNMTRAP.CNMTRAPD.MAXTCPCONN = 50
```

CNMTRAPD では、名前が &CNMTCPN シンボル (通常、TCPNAME キーワードの値) によって提供された TCP/IP スタックへのアフィニティーを得ることになります。CNMTRAPD では、トラップがないか、TCP ポート 162 および UDP ポート 162 を “listen” することになります。UNIX システム・サービス・ファイル /usr/lpp/netview/v6r2/cnmtrapd.conf (『SNMP トラップの自動化タスクの構成ファイル』で説明されています) は、CNMTRAPD タスクの SNMPv3 トラップ処理情報を含むことになります。この例では CNMTRAP.CNMTRAPD.TRACE 共通グローバル変数が定義されなかったため、CNMTRAPD の SNMP トラップの自動化トレースは使用不可になります。

SNMP トラップの自動化タスクの構成ファイル

SNMP トラップの自動化タスクが暗号化された SNMPv3 トラップを受信する場合には、SNMP トラップの自動化タスクがトラップ・データの暗号化を解除して認証できるようにするために、SNMP トラップの自動化タスクの構成ファイルに該当するパスフレーズまたは鍵情報を用意する必要があります。

SNMP トラップの自動化タスクは、SNMPv3 メッセージ・ヘッダーのメッセージ・フラグ・フィールド (msgFlags) に暗号化 (privFlag = 1) と認証 (authFlag = 1) の両方が指定されている場合に、SNMPv3 トラップは暗号化されていると想定します。それ以外の場合には、暗号化解除と認証の手順は実行されません。

SNMPv3 の暗号化解除と認証の情報は、SNMP トラップの自動化タスクの構成ファイルの単一ステートメントによって提供されます。例の `cnmtrapd.conf` には、サポートされたファイル特性、ステートメント構文、および例が説明されています。ファイル特性および行構文は次のとおりです。

- ファイルは、固定または可変レコード・フォーマットを使用でき、論理レコード長は最大 2048 バイトである。
- 1 桁目がアスタリスク (*) または番号記号 (#) の場合には、ファイル行はコメントと見なされる。
- コメント行以外の行は、次のフォーマットを取る (この例は実際には単一行ですが、スペースの制約の関係上 2 行で表示されています)

```
ipAddress port protocol snmpVer userName passPhrase
                        authAlg authKey privAlg privKey
```

項目の意味

ipAddress

SNMPv3 トラップのソースの IP アドレスを指定します。

port 1 から 65535 の範囲で、有効なポート番号を指定します。ポート番号は現在、暗号化解除および認証プロセスで使用されていません。

protocol

有効なプロトコル (トランスポート方式。TCP か UDP) を指定します。プロトコルは現在、暗号化解除および認証プロセスで使用されていません。

snmpVer

SNMP のバージョンを指定します。このパラメーターに **snmpv3** の値を指定する必要があります。

userName

セキュリティ名 (サポートされている SNMPv3 ユーザー・ベースのセキュリティ・モデルのユーザー名)。この名前には、1 から 32 文字の長さのものを指定できます。

passPhrase

上記ユーザー名の認証鍵および暗号化解除 (プライバシー) 鍵を生成するために使用されるパスワード。鍵が後続のパラメーターで提供される場合には、ハイフン (-) を指定できます。それ以外の場合には、8 から 64 文字の長さのパスワードを指定できます。パスワードが指定された場合には、後続のパラメーターで指定されたすべての鍵は無視されます。

authAlg

SNMP トラップ PDU の認証用のメッセージ・ダイジェストを生成するために使用するアルゴリズム。有効な値は、HMAC-SHA (Simple Hashing Algorithm 1, SHA-1) と HMAC-MD5 (Message Digest 5) です。これは、ステートメントで提供されたすべての鍵を生成するために使用されたアルゴリズムの記述で仮定されます。

passPhrase が提供されると、このアルゴリズムを使用して、認証鍵 (SNMP トラップ PDU の認証用) またはプライバシー鍵 (SNMP トラップ PDU の暗号化解除用)、またはその両方を生成します。

authKey

恐らく authAlg で指定されたアルゴリズムを使用して生成された、認証鍵を表す 16 進数字の文字表現。authAlg が HMAC-SHA の場合には、このパラメーターは 20 バイトの鍵になります。authAlg が HMAC-MD5 の場合には、このパラメーターは 16 バイトの鍵になります。

privAlg

暗号化/暗号化解除 (プライバシー) アルゴリズム。有効な値は、DES (暗号化ブロック・チェーン付きデータ暗号化規格) です。

privKey

恐らく authAlg で指定されたアルゴリズムを使用して生成された、プライバシー鍵を表す 16 進数字の文字表現。authAlg が HMAC-SHA の場合には、このパラメーターは 20 バイトの鍵になります。authAlg が HMAC-MD5 の場合には、このパラメーターは 16 バイトの鍵になります。

privKey に値を指定する場合には、authKey にも値を指定する必要があります。

注:

- 暗号化解除および認証情報が入った該当する SNMP トラップの自動化の構成ファイル・レコードを検索するために、2 つの項目 (発信元 IP アドレスとユーザー名) が使用されます。TCP クライアントからトラップを受信した場合には、発信元 IP アドレスは接続ピアになります。そうでない場合には、データグラム送信側になりますが、これは必ずしもトラップを送信した SNMP エージェントであるとは限りません (特にエンティティーがエージェントの代わりにトラップをフォワードした場合)。ユーザー名は、メッセージ・セキュリティ・パラメーターから抽出されます (ユーザー・ベースのセキュリティ・モデルが仮定されます)。
- ステートメントの継続はありません。ステートメントは、1 つの論理レコードに収める必要があります。
- ステートメント内でシンボルを使用できます。ステートメント内のシンボルは、ステートメントの処理前に置換されます。

以下に例を示します。

例 1:

```
10.42.44.25 162 TCP snmpv3 adam adampassword HMAC-SHA - DES -
```

これは、IP アドレス 10.42.44.25 から (現在 TCP と UDP の両方、およびすべての発信元ポートに適用可) 発信され、メッセージ・セキュリティ・パラメーターにユーザー名 **adam** が入っている、SNMPv3 トラップのエントリーです。フレーズ **adampassword** は、トラップの認証および暗号化解除に使用されるローカライズされていない鍵を生成するために SNMP トラップの自動化タスクが使用するパスフレーズです。

- 例 2:

注: 本文のスペースの制約のため、この例は複数行になっていますが、実際には 1 行で作成する必要があります。

```
10.42.44.25 162 UDP snmpv3 usermd5 - HMAC-MD5 67ef017ccb81111ba63b92e429338906
DES 67ef017ccb81111ba63b92e429338906
```

これは、IP アドレス 10.42.44.25 から (現在 TCP と UDP の両方、およびすべての発信元ポートに適用可) 発信され、メッセージ・セキュリティ・パラメーターにユーザー名 **usermd5** が入っている、SNMPv3 トラップのエントリーです。この例では、パズフレーズの代わりに、認証および暗号化解除に必要な鍵が提供されています。HMAC-MD5 が認証アルゴリズムに選択されているため、鍵が 16 バイトの長さの 16 進データの文字表現になっていることに注意してください。

SNMP トラップの自動化 CP-MSU

NetView プログラム内で既存の CP-MSU 自動化と重複する可能性を減らすために、CP-MSU には、GDS 変数が含まれており、そのキーは文脈依存型データの範囲内にあります。文脈依存型データについては、「*Systems Network Architecture Management Services Formats (GC31-8302)*」、「*Systems Network Architecture: Formats (GA27-3136)*」、または「*Systems Network Architecture: Network Product Formats (LY43-0081)*」を参照してください。

CP-MSU SNMP トラップの自動化内部にある各 GDS 変数は、2 バイト長で始まり、その後に 2 バイトのキーが続きます。長さの値には、長さとキーのフィールドの長さが含まれます。

CP-MSU 内の最初の GDS 変数には、SNMP トラップの発信元のエンティティに関する情報が入っています。この GDS 変数は常に存在します。

表 19. CP-MSU 内の GDS 変数

GDS 変数キー	説明
FFF0	SNMP トラップの発信元

この GDS 変数の内部には、次のように発信元を説明する追加の GDS 変数があります。

表 20. 発信元を説明する GDS 変数

GDS 変数キー	説明
FFF1	IPv4 または IPv6 の当てはまるほうの発信元 IP アドレス、標準テキスト表示形式。
FFF2	10 進数のポート番号の文字表現で表される、発信元ポート番号。
FFF3	トラップが受信されたプロトコル、IP トランスポートには、文字 TCP または UDP が含まれていることがあります。

CP-MSU の内部にある次の GDS 変数には、トラップから抽出されたデータを持つ、その他の GDS 変数がすべて含まれます。GDS 変数キーは、以下に示すように SNMP トラップのタイプによって異なります。

表 21. トラップのタイプを説明する GDS 変数

GDS 変数キー	説明
FFA4	SNMPv1 トラップ
FFA7	SNMPv2c または SNMPv3 トラップ

GDS 変数内の GDS 変数には、SNMP トラップで生成されたときのデータが含まれています (タグおよび長さなし)。これには例外があり、CP-MSU 自動化をサポートして変数バインディングで表示されることがあるさまざまなデータを許可するように設計されています。

エージェント IP アドレス (SNMPv1 トラップのみ)

エージェント IP アドレスは、CP-MSU 内で、IP (IPv4) アドレスの標準テキスト表示形式で表されます。

タイプ・オブジェクト ID のデータ

テストを容易にして、自動化内の BER エンコードのオブジェクト ID の比較的複雑な変換を実行しなくてもよいようにするために、これは、ASN.1 オブジェクト ID (ピリオドで区切られた 10 進数のサブ ID) の文字表現で表されます。

数量を表すデータ

integer、*counter*、*gauge*、*timeticks*、および *counter64* などのデータ型の場合、CP-MSU GDS 変数内に配置された値は常に、データ型に関連した最大値を表すのに十分な長さになります。つまり、*integer*、*counter*、*gauge*、または *timeticks* データ型は常に、4 バイト値 (データ型でサポートされた最大値を保持するのに必要な 32 ビット) で表されます。一方、*counter64* データ型は、8 バイト値で表されます。必要に応じて、左側をゼロで埋めることができます。

SNMP トラップの自動化ではコンストラクター・データ型 (例: SEQUENCE) で追加の処理を行わないことに注意してください。タグおよび長さがない値が、GDS 変数にそのまま配置されるだけです。

以下のテーブルでは、SNMP トラップの自動化 CP-MSU 内で作成できるすべての GDS 変数が示されています。また、GDS 変数が表示される可能性がある SNMP トラップの GDS 変数も記されています。特に記載がない場合には、GDS 変数は 1 回だけ表示されます。

表 22. SNMP トラップの自動化 CP-MSU 内で作成できる GDS 変数

キー	説明	該当する SNMP トラップの GDS 変数
FF00	SNMP バージョン、integer データ。常に表示	すべて
FF01	コミュニティ名、octet string データ	SNMPv2c トラップの場合にのみ FFA4、FFA7
FF02	エンタープライズ・オブジェクト ID、オブジェクト ID データ	FFA4
FF03	エージェント IP アドレス	FFA4
FF04	汎用トラップ、integer データ	FFA4

表 22. SNMP トラップの自動化 CP-MSU 内で作成できる GDS 変数 (続き)

キー	説明	該当する SNMP トラップの GDS 変数
FF05	特定のトラップ、integer データ	FFA4
FF06	タイム・スタンプ、timeticks データ	FFA4
FF07	要求 ID、integer データ	FFA7
FF08	エラー状況、integer データ	FFA7
FF09	エラー・インデックス、integer データ	FFA7
FF0A	メッセージ ID、integer データ	SNMPv3 トラップの場合にのみ FFA7
FF0B	メッセージ最大サイズ、integer データ	SNMPv3 トラップの場合にのみ FFA7
FF0C	メッセージ・フラグ、octet string データ	SNMPv3 トラップの場合にのみ FFA7
FF0D	メッセージ・セキュリティ・モデル、integer データ	SNMPv3 トラップの場合にのみ FFA7
FF0E	メッセージ・セキュリティ・パラメーター 注: GDS 変数の「注」のキー FF0E の追加情報を参照してください。	SNMPv3 トラップの場合にのみ FFA7
FF0F	コンテキスト・エンジン ID、octet string データ	SNMPv3 トラップの場合にのみ FFA7
FF10	コンテキスト名、octet string データ	SNMPv3 トラップの場合にのみ FFA7
FF11	変数バインディング・コンテナ。1 つの <i>name</i> GDS 変数 X'FF12' に続いて、対応する <i>value</i> GDS 変数 X'FF13' が格納されている。	すべて。(存在する場合は) 発信元 SNMP トラップの各変数バインディングに 1 本。
FF12	変数バインディングの変数 <i>name</i> 、オブジェクト ID データ。トラップの各変数バインディングで 1 回発生。	すべて。(存在する場合は) 発信元 SNMP トラップの各変数バインディングに 1 本。
FF13	変数バインディングの変数 <i>value</i> 。フォーマットに関しては下記の注を参照。トラップの各変数バインディングで 1 回発生。関連変数 <i>name</i> GDS 変数の直後に配置。 注: GDS 変数の「注」のキー FF13 の追加情報を参照してください。	すべて。(存在する場合は) 発信元 SNMP トラップの各変数バインディングに 1 本。

GDS 変数:

1. メッセージ・セキュリティ・パラメーターの GDS 変数、キー X'FF0E' には、次のとおり他の 4 本の GDS 変数が含まれます。

FF1E 権限サーバー・エンジン ID、octet string データ

FF2E 権限サーバー・エンジン・ブート、integer データ

FF3E 権限サーバー・エンジン時刻、integer データ

FF4E ユーザー名、octet string データ

2. *value* GDS 変数、キー 'X'FF13' には、その中に含まれるデータの型を通信する必要があるフォーマットがあります。そのフォーマットは次のとおりです。

....nnnnFF13tttttttthhhhhhhhhhhh...

ここで、*tttttt* は、続くデータの型を通信する GDS 変数のデータの最初の 4 バイト・フィールドです。このフィールドの値は、本質的には BER エンコードのトラップ内の値から取られた *tag* データです。データは、次の 568 ページの表 21 で説明するガイドラインに従った GDS 変数に配置されます。

注:

1. SNMP トラップの自動化タスクが受け入れる個々のトラップの最大サイズは、32500 バイトです。
2. 使用されるフォーマット規則と追加される発信元情報が原因で、32500 バイト未満の SNMP トラップにより、CP-MSU の最大サポート対象サイズ (32600 バイト) を超える CP-MSU が生成される可能性があります。このサイズを超えた場合、SNMP トラップは自動化できません。
3. SNMPv2c または SNMPv3 トラップが SNMP アーキテクチャーに従う場合は、*sysUpTime.0* と *snmpTrapOID.0* に 1 つずつ、2 つの変数バインディングが存在します。そのようなトラップに SNMP トラップの自動化タスクが作成した CP-MSU は、1 本の 'X'FFA7' GDS 変数と 2 本の 'FF11' GDS 変数を持ち、それぞれに、1 本の 'X'FF12' GDS 変数とそれに続く 1 本の 'X'FF13' FF13 GDS 変数が含まれています。

異なるバージョンの SNMP トラップのそれぞれに、作成された SNMP トラップの自動化 CP-MSU の標準的な開始のされ方があります。SNMP トラップの自動化 CP-MSU 内の各 GDS 変数は、2 バイトの長さで開始し、その後に 2 バイトのキーが続くことに注意してください。これらの例では、2 バイトの長さフィールドは、GDS 変数の開始としてポイントされます。

これは、SNMPv1 トラップの例です。GDS 変数 値 'X'FFA4' および *SNMP Version* 値 (X'00') の最終バイトは、SNMP バージョン 1 を示すことに注意してください。

CP-MSU	Origin	Origin IP	Origin	Protocol
	MV	Addr GDS Variable	Port	GDS Variable
			GDS Variable	
LLLL12120022FFF0000FFFF1F1F04BF1F04BF1F6F34BF70008FFF2F1F0F2F40007FFF3E3C3D7				

SNMPv1	SNMP	Community
Trap	Version	Name GDS Variable
MV		
nnnnFFA40008FF000000000000mmmmFF01hhhhhhhh...		

これは、SNMPv2c トラップの例です。SNMP バージョン SNMPv2c と SNMP バージョン SNMPv3 は両方とも GDS 変数 値 'X'FFA7' を使用します。この例では、*SNMP Version* 値の最終バイト (X'01') は、SNMP バージョン SNMPv2c を示します。

CP-MSU	Origin	Origin IP	Origin	Protocol
	MV	Addr GDS Variable	Port	GDS Variable
			GDS Variable	
LLLL12120022FFF0000FFFF1F1F04BF1F04BF1F6F34BF70008FFF2F1F0F2F40007FFF3E4C4D7				

SNMPv2c	SNMP	Community

Trap MV	Version	Name GDS Variable
nnnnFFA7	0008FF	0000000001mmmmFF01hhhhhhh...

これは、SNMPv3 トラップの例です。SNMP バージョン SNMPv2c と SNMP バージョン SNMPv3 は両方とも GDS 変数 値 X'FFA7' を使用します。この例では、SNMP Version 値の最終バイト (X'03') は、SNMP バージョン SNMPv3 を示します。

CP-MSU	Origin MV	Origin IP Addr GDS Variable	Origin Port GDS Variable	Protocol GDS Variable
LLLL12120022FF	0000FFFF	1F1F04BF1F04BF1F6F34BF70008FF	2F1F0F2F40007FF	3E3C3D7

SNMPv3 Trap MV	SNMP Version	Message ID GDS Variable
nnnnFFA7	0008FF	0000000003mmmmFF0Ahh...

X'FFF0' GDS 変数内に、トラップの発信元情報 (文字形式での IP アドレスおよびポート)、および SNMP トラップが使用したトランスポート・プロトコル名 (TCP または UDP) が表示されます。

SNMP バージョンの GDS 変数には、各 SNMP バージョンで表示される値が強調表示されています。SNMP トラップの主 GDS 変数キー (X'FFA4' および X'FFA7') は、キー内に該当する SNMP トラップ・コンストラクター・タグ (SNMPv1 の場合は X'A4'、SNMPv2c および SNMPv3 の場合は X'A7') を含んでいることに注意してください。

すべてのタイプの SNMP トラップで、X'FF11' GDS 変数は、変数バインディングがトラップ内に存在しない場合には、SNMP トラップの自動化 CP-MSU 内に作成されません。発生する各変数バインディングで、X'FF11' GDS 変数は次のように作成されます。

VarBind	Variable	Variable
Container	"Name" Variable GDS Variable	"Value" GDS Variable
...LLLLFF11	nnnnFF12objectID	m...FF13ttttttthhhhhh...

nnnn は、(name の文字表現、つまり、オブジェクト ID を含む) name GDS 変数の長さ、mmmm は、(value のデータ型を表す 4 バイト値に続いて value 自体を含む) value GDS 変数の長さ、LLLL は、変数 container の合計長、LLLL は、nnnn + mmmm + 4。

SNMP トラップの自動化の例

完全な SNMP トラップの自動化 CP-MSU を表示するには、「IBM Tivoli NetView for z/OSカスタマイズ・ガイド」の詳細なトラップとアラートの変換例で使用されている SNMP トラップを検討してください。

```
*
* Outermost constructor for the trap (tag and length)
*
30820127
* SNMP version (00 = SNMPv1)
020100
```

```

* Community name (public)
04067075626C6963
* Trap PDU
A4820118
* Enterprise object ID (1.3.6.1.4.1.12270)
06072B06010401DF6E
* Agent address (10.71.225.20)
40040A47E114
* Generic trap code (6 = enterprise specific)
020106
* Specific trap code (32 in decimal)
020120
* Timeticks
430402A2D49D
* Variable bindings "container"
308200F9
* Variable binding 1
3015
* Variable 1 (1.3.6.1.4.1.12270.200.2.1.1.1)
060D2B06010401DF6E814802010101
* Value 1 (octet string "1493")
040431343933
* Variable binding 2
3019
* Variable 2 (1.3.6.1.4.1.12270.200.2.1.1.2)
060D2B06010401DF6E814802010102
* Value 2 (octet string "/L20/050")
04082F4C32302F4F3530
* Variable binding 3
3024
* Variable 3 (1.3.6.1.4.1.12270.200.2.1.1.3)
060D2B06010401DF6E814802010103
* Value 3 (octet string "2005-01-10T16:13:00")
0413323030352D30312D31305431363A31333A3030
* Variable binding 4
3014
* Variable 4 (1.3.6.1.4.1.12270.200.2.1.1.4)
060D2B06010401DF6E814802010104
* Value 4 (octet string "I14")
0403493134
* Variable binding 5
3025
* Variable 5 (1.3.6.1.4.1.12270.200.2.1.1.5)
060D2B06010401DF6E814802010105
* Value 5 (octet string "DIGIN ON OCCURRED")
0414444947494E204F4E202020204F43435552524544
* Variable binding 6
3015
* Variable 6 (1.3.6.1.4.1.12270.200.2.1.1.6)
060D2B06010401DF6E814802010106
* Value 6 (octet string "DI=1")
040444493D31
* Variable binding 7
3025
* Variable 7 (1.3.6.1.4.1.12270.200.2.1.1.7)
060D2B06010401DF6E814802010107
* Value 7 (octet string "RC2 Gas Status Man. ")
04145243322047617320537461747573204D616E2E20
* Variable binding 8
3011
* Variable 8 (1.3.6.1.4.1.12270.200.2.1.1.8)
060D2B06010401DF6E814802010108
* Value 8 (NULL)
0500
* Variable binding 9
3011

```

```
*      Variable 9 (1.3.6.1.4.1.12270.200.2.1.1.9)
060D2B06010401DF6E814802010109
*      Value 9 (NULL)
0500
```

このトラップが IP アドレス 3A20:ABCD:70:1AB:10:10:173:7 とポート 1038 で、TCP クライアントから、エンティティの代わりに送信されたとします。このトラップを受信した SNMP トラップの自動化タスクは、次のような SNMP トラップの自動化 CP-MSU を生成します (GDS 変数は、GDS 変数が対応する発信元のトラップからのデータを示すように、別々かつ注釈を付けて表示されています)。

```
*
* Outermost CP-MSU container
*
026D1212
* Trap origin GDS variable
0033FFF0
* Origin IP address GDS variable ("3A20:ABCD:70:1AB:10:10:173:7")
0020FFF1F3C1F2F07AC1C2C3C47AF7F07AF1C1C27AF1F07AF1F07AF1F7F37AF7
* Origin port ("1038", character representation of the port number in decimal)
0008FFF2F1F0F3F8
* Transport ("TCP")
0007FFF3E3C3D7
* SNMP trap GDS variable (for the SNMPv1 trap)
0236FFA4
* SNMP version GDS variable (00 = SNMPv1)
0008FF0000000000
* Community GDS variable ("public")
000AFF017075626C6963
* Enterprise object ID GDS variable ("1.3.6.1.4.1.12270",
* notice EBCDIC character representation)
0015FF02F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF1
* Agent address GDS variable ("10.71.225.20")
0010FF03F1F04BF7F14BF2F2F54BF2F0
* Generic trap GDS variable (enterprise specific trap)
0008FF04000000006
* Specific trap GDS variable (32 in decimal)
0008FF050000000020
* Timestamp GDS variable
0008FF0602A2D49D
* Variable binding GDS variable 1
0031FF11
* Variable name 1 (1.3.6.1.4.1.12270.200.2.1.1.1)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF1
* Variable value 1 (00000004 = octet string, data = "1493",
* notice unchanged ASCII data)
000CFF1300000000431343933
* Variable binding GDS variable 2
0035FF11
* Variable name 2 (1.3.6.1.4.1.12270.200.2.1.1.2)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF2
* Variable value 2 (00000004 = octet string, data = "/L20/O50")
0010FF130000000042F4C32302F4F3530
* Variable binding GDS variable 3
0040FF11
* Variable name 3 (1.3.6.1.4.1.12270.200.2.1.1.3)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF3
* Variable value 3 (00000004 = octet string, data = "2005-01-10T16:13:00")
001BFF13000000004323030352D30312D31305431363A31333A3030
* Variable binding GDS variable 4
0030FF11
* Variable name 4 (1.3.6.1.4.1.12270.200.2.1.1.4)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF4
* Variable value 4 (00000004 = octet string, data = "I14")
000BFF13000000004493134
* Variable binding GDS variable 5
```

```

0041FF11
* Variable name 5 (1.3.6.1.4.1.12270.200.2.1.1.5)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF5
* Variable value 5 (00000004 = octet string, data = "DIGIN ON OCCURRED")
001CFF1300000004444947494E204F4E202020204F43435552524544
* Variable binding GDS variable 6
0031FF11
* Variable name 6 (1.3.6.1.4.1.12270.200.2.1.1.6)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF6
* Variable value 6 (00000004 = octet string, data = "DI=1")
000CFF130000000444493D31
* Variable binding GDS variable 7
0041FF11
* Variable name 7 (1.3.6.1.4.1.12270.200.2.1.1.7)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF7
* Variable value 7 (00000004 = octet string, data = "RC2 Gas Status Man. ")
001CFF13000000045243322047617320537461747573204D616E2E20
* Variable binding GDS variable 8
002DFF11
* Variable name 8 (1.3.6.1.4.1.12270.200.2.1.1.8)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF8
* Variable value 8 (00000005 = NULL, no data following the data type)
0008FF13000000005
* Variable binding GDS variable 9
002DFF11
* Variable name 9 (1.3.6.1.4.1.12270.200.2.1.1.9)
0021FF12F14BF34BF64BF14BF44BF14BF1F2F2F7F04BF2F0F04BF24BF14BF14BF9
* Variable value 9 (00000005 = NULL, no data following the data type)
0008FF13000000005

```

オブジェクト ID が ASN.1 フォーマットの EBCDIC 文字表現 (ピリオドで区切られた 10 進数のサブ ID) で表示されていることに注意してください。これによって、自動化のテストがかなり容易になります。

第 8 部 付録

付録 A. NetView プログラムの新しい自動操作機能へのマイグレーションの計画

この付録では、NetView プログラムが提供する新規の自動化機能を計画する際に、あるいはこれらの機能へマイグレーションする際に使用できる一般情報を、早見表の形で提供します。これらの新機能の中には、NetView のみで実現する自動化の改善もあれば、他のプロダクトにおける NetView 自動化の改善との関連で実現するものもあります。

この付録では、IBM Tivoli NetView for z/OS プロダクトのさまざまなバージョンにおける自動化の改善に焦点を当てています。

NetView for z/OS V6R2 プログラム

表 23 に、Tivoli NetView for z/OS V6R2 プログラムにおける自動化の機能拡張および改善に貢献している項目（コンポーネント、機能、サービス、サポート）を示します。

表 23. NetView for z/OS V6R2 プログラムにおける自動化の拡張

機能拡張	説明	参照情報
分析に使用可能な初期 IPL メッセージ	システムの初期化 (IPL) を行うと、すべてのメッセージが記録され、Canzlog の確認および自動化に使用できるようになります。	IBM Tivoli NetView for z/OS アドミニストレーション・リファレンス
	サブシステム初期化の前に記録されたメッセージには、CHKEY 値 M が付きます。また、NetView プログラムの開始前に出されたメッセージには、MVSPARM.Msg.Automation.Oldtag CNMSTYLE ステートメントで指定される autotoken 標識が付きます。	
	特定の NetView インスタンスは、同じサブシステム名を持つ NetView インスタンスにより以前は自動化されていたメッセージを自動化することはありません。これは、NetView プログラムおよびサブシステム・ルーター・タスク (CNMCSSIR) を再開しても同じです。	

NetView for z/OS V5R4 プログラム

表 24 に、Tivoli NetView for z/OS V5R4 プログラムにおける自動化の機能拡張および改善に貢献している項目（コンポーネント、機能、サービス、サポート）を示します。

表 24. NetView for z/OS V5R4 プログラムにおける自動化の拡張

機能拡張	説明	参照情報
NetView MVS コマンド改訂	MVS コマンド改訂機能を使用して、MVS コマンドの検査、変更、または削除（取り消し）を実行できます。 WTOR に対する応答の取得、他の MVS コマンドに対する応答の取得、ファイルの読み取りなど、NetView アドレス・スペースへの転送が必要な複雑な変更を行うことができます。 既存の MVS コマンド管理機能は、マイグレーション目的でのみ引き続きサポートされますが、非推奨扱いとなります。	149 ページの『第 14 章 コマンド改訂テーブル』 - IBM Tivoli NetView for z/OSインストール:マイグレーション・ガイド

NetView for OS/390 V1R4 プログラム

表 25 に、Tivoli NetView for OS/390 V1R4 プログラムにおける自動化の機能拡張および改善に貢献している項目（コンポーネント、機能、サービス、サポート）を示します。

表 25. NetView for OS/390 V1R4 プログラムにおける自動化の拡張

機能拡張	説明	参照情報
ポリシー・サービスの概要	NetView ポリシー・サービスは、動的なポリシー・ペー スの管理およびユーザーのリソースの自動化を可能にする 一連の機能です。ポリシー・サービスには、以下がありま す。 - ポリシー・エンジン API - 自動化ポリシー・エンジン - タイマー API	301 ページの『第 16 章 ポ リシー・サービスの概要』
MVS コマンド管理	MVS コマンド管理機能を使用すると、ほとんどの MVS コマンドを検査、変更、または拒否できるようになりま す。コマンドごと、またはコンソール名ごとに、コマンド を個別に処理に組み込んだり、除外することができます。 MVS コマンド管理機能は、マイグレーション目的でのみ 引き続きサポートされますが、非推奨扱いとなります。 MVS コマンド改訂機能が MVS コマンド管理機能の置き 換えとなります。	175 ページの『条件項 目』 - IBM Tivoli NetView for z/OSインストール:マイグ レーション・ガイド

付録 B. サンプル・プロジェクト計画

この付録では、自動操作プロジェクト計画のサンプルを提供します。それぞれの計画を立てるときのモデルとして利用してください。ここに挙げた計画は 1 つの例にすぎません。したがって、どの環境にも適用されるものではありません。

この自動化計画は、本書の他の場所でも使用されている 4 つのフェーズに従っています。表 26 に全フェーズをリストし、各フェーズに含まれるアクティビティーを要約します。各フェーズは通常、順番に行いますが、前のフェーズをすべて完了する前に別のフェーズを開始しなければならない場合もあるでしょう。つまり、あるフェーズのタスクについて、別のフェーズのタスクの完了が影響してくる場合があります。例えば、定義フェーズのコスト評価タスク (タスク 1.13) を実行するには、事前に設計フェーズのリソース定義タスク (タスク 2.03) が完了していなければなりません。また、テスト・システムにおいてパフォーマンスの追跡および比較を行うためには (タスク 2.05)、そのシステムにあらかじめ自動化をインストールします (タスク 3.04)。

表 26. プロジェクトのフェーズ (プロジェクト計画の例)

フェーズ	フェーズ名	アクティビティーの要約
1	定義	アクティビティーを管理および実行するための計画チームを編成する。 目標と目的を決定する。 自動操作プロジェクトの短期目標と長期目標を定義する。 現在の操作環境と操作手順を文書化する。 自動化のためのプロジェクト計画を作成する。
2	設計	アクティビティーを管理および実行するための設計チームを編成する。 自動化の実施計画を設計し、計画が短期および長期の目標と目的にかなったものであることを確認する。
3	実施	アクティビティーを管理および実行するための実施チームを編成する。 設計フェーズでの計画に従って、識別された領域内での操作を自動化するプロシージャを開発する。 これらのプロシージャーを、テスト・システム上でコーディングしてテストする。
4	実動	アクティビティーを管理および実行するための実動チームを編成する。 実動システム上に自動化プロシージャーをインストールして、テストを行う。 システム・パフォーマンスを追跡し、必要に応じてプロシージャーを変更する。 次の自動化のステージを計画するためのデータを収集する。

すべてを一度に行うのではなく、段階を追って自動化を実施すると、自動化処理全体にわたって、自動化計画内の作業を繰り返し行うことができます。このため、1つの段階で得られた経験とデータを、次の段階のための計画を改善するのに役立てることができます。

操作環境は刻々と変化するものです。例えば、自動化計画に携わっているときに、組織が、そのシステムとネットワークに新しいソフトウェアやハードウェアを追加するような場合もあります。したがって、こういった変更に対処できるようにしておいてください。スケジュールには、変更内容の分析、新規プロダクトに関する自動化の評価、および自動化処理への新規プロダクトの組み込みを行うための時間的余裕をもたせてください。

組織に、操作プロシージャ、方針、および報告書が完備している場合は、既存の資料を収集するだけで、情報収集作業の多くを達成することができます。

プロジェクトの定義

表 27 に、自動操作プロジェクトの定義に使用するタスクおよびサブタスクのいくつかをリストします。プロジェクト定義フェーズに関する一般情報については、47 ページの『第 3 章 自動操作プロジェクトの定義』を参照してください。

表 27. 定義フェーズ

タスク	タスクのアクション	サブタスクのアクティビティー
1.01	最初の提案会議の開催	プロジェクトのための最初の提案会議を行う。 計画処理のための経営幹部の確約を確認する。 管理者の中から自動操作プロジェクト定義の責任者を選ぶ。 管理者の中から自動化の設計、実施、および実動の責任者を選ぶ。 プロジェクトの管理者を選ぶ。 プロジェクト定義チームのメンバーを選ぶ。
1.02	方針決定会議の開催	プロジェクト定義チームのメンバーと Tivoli 営業所のシステム・エンジニアの間で方針決定会議を開催する。 自動操作に関してチームのメンバーを教育する。 プロジェクトの状況に対する報告手順を確立する。 購入可能な既存の自動化アプリケーション・プログラムを調査する。
1.03	組織の業務環境の分析	事業目的を検討する。 データ処理の目的を検討する。 運用構成を検討する。 運用管理方針を検討する。 システム管理の各規律に関する作業と目的を決定する。 運用分野における問題解決プロセスを決定する。

表 27. 定義フェーズ (続き)

タスク	タスクのアクション	サブタスクのアクティビティー
1.04	操作上の問題の識別	オペレーターから操作状況に関する情報を収集する。 メッセージと MSU を分析する。 コマンドを分析する。 サービス水準に関する合意を検討する。 オペレーター手順書を分析する。 問題管理報告書および手順を分析する。 ヘルプ・デスクのログ記録を分析する。 ユーザーから情報を収集する。 経営幹部から情報を収集する。 自動化の結果を追跡するための測定基準を設ける。
1.05	自動化の目標の設定	測定可能な長期および短期の目標を定義する。 組織に対する自動化による利益を定量化する。 目的を設定する。
1.06	操作リソースの目録作成	ハードウェアに関するデータを収集する。 ソフトウェアに関するデータを収集する。 システムとネットワークの構成について理解し、資料を作成する。 インストールすべきハードウェアとソフトウェアを決定する。
1.07	ユーザーから情報を収集する。	ユーザーの数と操作上の要件を定義する。 ユーザーが自分で行える操作の作業を定義する。 ユーザーに配布できるメッセージを定義する。
1.08	変更管理グループの確立	組織内および操作環境における変更を監視するためのグループを確立する。 このグループで使用する報告手順を確立する。
1.09	設計チームの編成	設計フェーズを実行する人々を決定する。 セミナーを実施し、自動化の概要および設計チームの目的について教育する。
1.10	自動化環境における管理者の役割決定	自動化環境における経営層の役割を決定する。 自動化環境における問題解決手順を定義する。
1.11	自動化環境における社員の役割の定義	自動化環境における社員の役割を決定する。 各役割に対する仕事の記述と主な作業を決定する。
1.12	社員に対する教育要件の確立	自動化環境のために必要な社員教育の概略を作成する。 社員トレーニングのための要件を設定する。
1.13	自動化の正当性の説明	計画チームと設計チームの両方によって収集された情報を使用して、自動化のためのコスト面からの正当性の説明を行う。

表 27. 定義フェーズ (続き)

タスク	タスクのアクション	サブタスクのアクティビティー
1.14	提案書の開発	自動化のための提案書を開発する。 設計チームのアクティビティー、発見事項、推薦事項を要約した最終報告書を作成する。 経営層にこの提案書と報告書を提示する。

設計

表 28 に、自動操作プロジェクトの設計にかかわるタスクおよびサブタスクのいくつかをリストします。プロジェクト設計フェーズに関する一般情報については、 57 ページの『第 4 章 自動操作プロジェクトの設計』を参照してください。

表 28. 設計フェーズ

タスク	タスクのアクション	サブタスクのアクティビティー
2.01	最初の設計会議の開催	計画チームおよび設計チームとともに会議を行う。 設計チーム・メンバーの役割を決定する。 設計フェーズの報告手順を設定する。 自動化設計のための承認手順を設定する。 自動操作に関する設計担当者を教育する。
2.02	自動化のための高水準設計の設定	プロジェクト定義チームによる調査結果を検討する。 今後のシステム構成を決定する。 自動化のための運用手順を決定する。 システムとネットワークを自動化する際の順序を決定する。 さまざまなプログラムを自動化する際の順序を決定する。
2.03	自動化のためのリソースの定義	現在および今後必要なリソースを定義する。 人材 ハードウェア・プロダクトとソフトウェア・プロダクト 設備サポート
2.04	テスト・システムへの導入に先立って、プロセスの承認	自動化プロシーチャーをテスト・システムに導入する前に、それらを検討して承認する。
2.05	テスト・システム上の自動化のパフォーマンスの追跡	テスト・システム上の自動化の測定結果を分析する。 結果をタスク 1.04 の測定結果と比較する。
2.06	実動システムにおける自動化のパフォーマンスの追跡	実動システム上の自動化の測定結果を分析する。 それらの結果と、自動化以前の測定結果とを比較する。

実施

表 29 に、自動操作プロジェクトの実施に伴うタスクおよびサブタスクのいくつかをリストします。実施フェーズに関する一般情報については、69 ページの『第 5 章 自動操作プロジェクトの実施』を参照してください。

表 29. 実施フェーズ

タスク	タスクのアクション	サブタスクのアクティビティー
3.01	最初の実施会議の開催	他のチームのメンバーとともに会議を行う。 実施チームのメンバーに自動操作の説明を行う。 プロジェクトの状況に対する報告手順を確立する。
3.02	トレーニング・セミナーへの参加	自動操作に関するトレーニング・セミナーに参加する。 確立されたプロシーチャーを学んで、自動操作プロシーチャーの資料を作成する。 コマンド・リストの書き方など、自動操作プロシーチャー開発に必要な新しい技能があれば、それを習得する。
3.03	必要なプロダクトのインストール	自動化の開発およびテストに必要なすべてのプロダクトをインストールする。
3.04	実施計画の実行	設計チームが作成した実施計画を実施させる。 設計チームによって確立した指針に従って、自動化機能とプロシーチャーを作成する。
3.05	テスト・システムへの自動化のインストール	テスト・システムに自動化をインストールする。
3.06	プロシーチャーのテスト	テスト・システム上で自動化をテストする。
3.07	パフォーマンスの測定と追跡	自動化環境におけるテスト・システムのパフォーマンスを測定および追跡する。 AUTOCNT コマンドを使用して自動化テーブル使用報告書を生成する。 TASKUTIL コマンドを使用してタスクのパフォーマンスおよび CPU 使用率を監視する。
3.08	プロシーチャーの検討と調整	必要に応じてプロシーチャーを調整する。 そのプロシーチャーを再度テストする。 テスト・システムのパフォーマンスを再度測定および追跡する。
3.09	プロシーチャーの承認	テスト結果とシステム・パフォーマンスに基づいて、自動操作プロシーチャーを正式に検討および承認する。プロシーチャーの承認グループには、全自動化チームからのメンバーが含まれていなければならない。
3.10	実動チームのサポート	新規プロダクトのインストールの援助を行う。 実動システムに対する自動操作プロシーチャーの調整の援助を行う。 実動システムの結果を監視する。 実動システムに合わせて調整されたプロシーチャーがある場合、そのプロシーチャーのテストを援助する。

実動

表 30 に、自動操作プロジェクトの実動にともなうタスクおよびサブタスクのいくつかをリストします。実動フェーズに関する一般情報については、69 ページの『第 5 章 自動操作プロジェクトの実施』を参照してください。

表 30. 実動フェーズ

タスク	タスクのアクション	サブタスクのアクティビティー
4.01	最初の実動会議の開催	他のチームのメンバーとともに会議を行う。 実動チーム・メンバーの役割を決定する。 インストール・チームのメンバーに、自動操作に関する教育を行う。
4.02	必要なプロダクトのインストール	適切なシステムに自動操作プロダクトをインストールする。 そのシステム上でテストを行う。
4.03	自動化プロシージャのマイグレーション	コマンド・リストおよび自動化テーブルのような自動化機能とプロシージャを、実動システムにマイグレーションする。
4.04	プロシージャの調整	すべてのプロシージャを調整およびテストして、各システムの要件に適合させる。AUTOCNT コマンドおよび AUTOTEST コマンドを使用して自動化テーブル使用報告書を生成する。TASKMON コマンドおよび TASKUTIL コマンドを使用してタスクのパフォーマンスおよび CPU 使用率を監視する。
4.05	プログラムの更新および機能拡張のインストール	自動操作プロダクトが更新および機能拡張された場合は、それをインストールしてテストする。
4.06	機能拡張の調整	ソフトウェアまたはハードウェアの拡張機能すべてを調整、およびテストして、システムの要件に適合させる。
4.07	プロシージャの検討とオペレーターの訓練	自動化を定期的に検討して、自動化環境に関するオペレーターの教育のための計画を実施する。 自動化以前の測定結果と、測定結果を比較する。

計画表

計画の各フェーズのタスクを完了するために要した時間を計算するには、次の表を利用してください。

表 31. プロジェクト定義のための計画表 (段階 1)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
1.01				
1.02				
1.03				
1.04				
1.05				
1.06				
1.07				

表 31. プロジェクト定義のための計画表 (段階 1) (続き)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
1.08				
1.09				
1.10				
1.11				
1.12				
1.13				
1.14				
合計				

表 32. 設計のための計画表 (段階 2)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
2.01				
2.02				
2.03				
2.04				
2.05				
2.06				
合計				

表 33. 実施のための計画表 (段階 3)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
3.01				
3.02				
3.03				
3.04				
3.05				
3.06				
3.07				
3.08				
3.09				
3.10				
合計				

表 34. 実動のための計画表 (段階 4)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
4.01				
4.02				

表 34. 実動のための計画表 (段階 4) (続き)

タスク	完了までの見積 日時	完了までの実績 日時	責任者	コメント
4.03				
4.04				
4.05				
4.06				
4.07				
合計				

付録 C. 進捗状況測定サンプル

この付録では、測定可能な目的および尺度の例を紹介します。これらの目的および尺度を使用することによって、自動化の価値と自動化のコストとの比較を見積もることができます。これらの目的および指標は、プロジェクトの進捗具合を測定する上でも役に立ちます。

表 35 に、『オペレーター生産性の向上』および『オペレーター・タスクの簡素化』といった一般的な目的について、いくつかの尺度および測定方法を示します。表には見積コスト欄が設定されていますが、尺度や測定方法の中にはコストの割り当てが不可能なものもあります。また、尺度および測定方法の中には重複しているものもあります。したがって、記入するコストの全額を合計する必要はありません。各組織の目標に関連する目的と尺度だけを選択してください。

リストされている項目すべてが、使用中のシステムに適用するとは限りません。

表 35. 自動化の価値を評価するための尺度および測定方法

目標に関連する尺度または測定方法	自動化以前	コスト	自動化後	コスト
オペレーターの生産性の向上 必要なオペレーターの数 システムおよびサブシステムごとのオペレーターの数 リカバリー時間 システムを監視するコンソール・オペレーターの数 オペレーターによって実行されるシステム操作の数				
オペレーター・タスクの簡素化 人材育成に必要な時間 必要なコンソールの数 表示されたメッセージの数 表示されたアラートの数				
人的エラーの削減 プロシージャ・エラーの数 コンソール・エラーの数 障害停止の数				
コンソール・メッセージ・トラフィックの削減 抑止されたメッセージの数 自動化されたメッセージの数 配布されたメッセージの数 表示されたメッセージの数				
ハードウェア・モニター・アラート・トラフィックの削減 抑止されたアラートの数 (ESREC または AREC フィルター)				

表 35. 自動化の価値を評価するための尺度および測定方法 (続き)

目標に関連する尺度または測定方法	自動化以前	コスト	自動化後	コスト
自動化されたアラートの数 ハードウェア・モニターにより表示されたアラートの数 表示されたアラートの数				
問題応答時間の削減 通知から訂正アクションまでの時間				
システム使用可能性の向上 障害停止の数 平均障害停止時間 コンポーネント別の平均障害停止時間 リカバリー時間の分数 ネットワーク再始動の数				
集中オペレーター制御 必要なコンソールの数 問題通知に対する応答時間 各システムに必要なオペレーターの数				
集中報告処理 ログの数 問題管理担当者の数 ログ内の不整合の数 ログ内の重複の数				
管理の改善 プロシージャ・エラーの数 不正応答の数 リカバリーにおける障害停止時間 サービス・レベル承認での変更 操作監査証跡中のエラーの数				
拡張のための制約の削減 ワークロードに対するコンソール・オペレーターの数 必要なコンソールの数 人材育成に必要な時間 システムごとに必要なオペレーターの数 新しいシステムをインストールするための時間 新しいシステムのための新しいプロシージャの数 新しいシステムを現在の環境に接続するための時間				

付録 D. MVS メッセージおよびコマンド処理

この付録では、NetView プログラムにより提供される診断、修正、およびチューニング情報について説明します。

重要: この診断、修正、チューニング情報を、プログラム・インターフェースとして使用しないでください。

大部分の自動化操作では、システムおよびそのアプリケーションとサブシステム (IMS、CICS、JES2、JES3 など) により出されるメッセージを処理します。コマンドが MVS、NetView プログラム、またはその他のアプリケーションのいずれから出されても、コマンドの出し方には、同じような考慮事項が適用されます。この付録では、NetView プログラムが MVS を使用してどのようにメッセージやコマンドを処理するかについて説明します。

MVS におけるメッセージ・フロー

MVS では、オペレーターへ送られる通常のメッセージは、プログラムが以下の要求のいずれかを出すと作成されます。

WTO オペレーター宛メッセージ

WTOR

要応答オペレーター宛メッセージ

これらの要求は MVS 監視プログラムの一部である WTO プロセッサにより処理されます。MVS は、各メッセージごとに WQE (作業キュー・エレメント) と呼ばれる制御ブロックを作成します。WQE には、メッセージ・テキスト、および WTO パラメーター・リストから取り出されたそのメッセージのすべての関連情報と WTO が出された時点のシステムに関する関連情報とが含まれています。

メッセージ処理機能

インストール先では MPF を使用すると、メッセージの処理方法を決定することができます。MPF の重要な機能には、以下の機能があります。

- メッセージの抑止
- 出口
- アクション・メッセージの処理
- 自動化

MPF は、アクティブな MPF リスト (SYS1.PARMLIB のメンバー MPFLSTxx) に指定されている、メッセージ処理指定を取り扱います。そのメンバーに含まれているステートメントでは、次の 2 種類の規則を指定しています。

- MVS コンソール・サポートがメッセージを処理する際に守らなければならない規則
 - 表示の抑止
 - アクション・メッセージの保存
 - メッセージ記述子コードの表示属性

- メッセージが自動化機能を行う際に守らなければならない規則
 - MPF インストール・システム出口の選択
 - 自動化サブシステムの選択

MPF インストール・システム出口で行われる処理は、WTO プロセッサと WTO を出したプログラムに同調して、同期的に起こります。NetView プログラムの自動化サブシステムで行われる処理は、WTO プロセッサとは非同期に実行され、WTO を出したプログラムと並行して処理されます。

MPF では、メッセージ ID の SUP キーワードをチェックしてそのメッセージはハードコピー・ログ専用かどうかのマークを付けます。メッセージにハードコピー・ログ専用のマークが付けられていると、そのメッセージは複数コンソール・サポート・コンソールでは表示されません。メッセージ ID の AUTO および RETAIN[®] キーワードをチェックしてマーク付けします。

最後に、メッセージを検査して、インストール・システム出口ルーチンにメッセージを渡すかどうかを判別します。渡すときには、ルーチンがロードされ、制御権がそのルーチンに渡されます。WQE には、MPF 処理の結果を示すフィールドがあります。インストール・システム出口ルーチンは、いくつかのフィールドを更新することもできます。

メッセージ処理におけるサブシステム

MPF 処理後、メッセージはアクティブのすべてのサブシステムにブロードキャストされます。メッセージは、各サブシステムへ次々と渡されます。各サブシステムではそのメッセージを調べ、該当するサブシステム処理を行うことができます。サブシステムは、メッセージ・テキストまたはその他の特性 (WQE フィールド) を変更することができます。WQE には、各メッセージを処理する際にジョブ入力システムにより与えられる特殊ジョブ関連情報用のフィールドがあります。代表的な MVS サブシステムには、JES2、JES3、NetView プログラム、OPC/ESA、CICS、および IMS があります。

1 次ジョブ入力サブシステム (JES2 または JES3) がアクティブのときには、SYS1.PARMLIB の IEFSSNxx メンバー (サブシステム名テーブル (SSNT) として参照される) におけるそのサブシステムの位置にかかわらず、そのサブシステムが常に最初にメッセージを処理します。その後、メッセージは SSNT に指定された順序に従い、各サブシステムに渡されます。ただし、IMS はこの規則の例外になります。IMS が MVS サブシステムとして実行されているときには、IMS がサブシステム・インターフェースで最後のサブシステムであることを確認するための特別なコードを用意しています。

注: JES2 の場合、2 次ジョブ入力サブシステムが存在するときには、IEFSSNxx 内のサブシステム・リストで、そのサブシステムの定義を NetView プログラムの定義より前に指定してください。

メッセージ用にサブシステム・インターフェースを使用する場合に、アクティブの NetView サブシステム・アドレス・スペースが存在するときには、そのアドレス・スペースがサブシステム・インターフェースで WQE を受信します。NetView サブシステムは NetView アプリケーションへ渡すメッセージを選択し、関連データを WQE からそのサブシステムの『メッセージ』へコピーします。WQE は NetView

により修正されず、次のサブシステム (存在する場合) へ渡されます。NetView サブシステムは、新しいメッセージが NetView アプリケーションによりキューから削除される前に、それらのメッセージをそのサブシステムのアドレス・スペースのキューに入れます。メッセージが NetView アプリケーションに渡されるのは、NetView アプリケーションが実行中で、サブシステム・インターフェースのルーター・タスクがそのアプリケーションでアクティブのときだけです。

NetView が拡張複数コンソール・サポート (EMCS) コンソールを使用しているとき、MVS 機能は、仮想記憶域間転送を使用して、EMCS コンソールとして定義された NetView タスクに直接メッセージを送信します。

NetView サブシステムは、MPF の処理中に、メッセージが自動化処理用にマークされているかどうかをチェックします。マークが付けられた場合、WQE からのメッセージ・テキストおよび選択された属性はコピーされ、NetView サブシステム・アドレス・スペースのキューに入れられます。

NetView 自動操作にメッセージを送信するには、そのメッセージの MPF テーブルの AUTO キーワードを AUTO(YES)、または AUTO(token) に設定してください。メッセージがオペレーターに表示されないように抑止するためには、メッセージ処理機能 (MPF) テーブルの SUP キーワードを SUP(YES) に設定します。ただし、メッセージを NetView 自動操作へは送信せず、しかもそのメッセージを抑止する場合は、そのメッセージは MPF テーブルで停止し、実質的に消失します。したがって、MPF を出て行くメッセージを完全に停止するのでない限り、同一メッセージで AUTO(NO) と SUP(YES) を同時に使用することは避けてください。

NetView で獲得した EMCS コンソールは、MPF テーブルで SUP(YES) のマークが付いたメッセージについては、経路コードで送信請求を行うことはできません。

MPF テーブルで AUTO(YES) または AUTO(token) とマークされているメッセージ、あるいは NETVONLY または REVISE("I" AUTOMATE) 改訂テーブル・アクションの対象となるメッセージ、またはそれらと類似のメッセージは、EMCS コンソールで受信できます。デフォルトの解釈では、CNMCSSIR タスクがこれらのメッセージを受信します。

注: JES2、JES3、および NetView では、サブシステムの複数コピーをアクティブにすることができます。各コピーには SSNT 内で固有の名前を付ける必要があり、メッセージ処理のために各コピーはそのサブシステム名で呼び出されます。このため、各 NetView サブシステムはメッセージを選択して、別の NetView アプリケーションに処理してもらうためにそのメッセージをキューに入れることができます。

複数コンソール・サポート

メッセージはすべてのアクティブなサブシステムにブロードキャストされたあと、複数コンソール・サポートへ渡されます。メッセージにハードコピー・ログ専用のマークが付けられていないときには、そのメッセージは、該当する経路コードを持つすべての複数コンソール・サポート・コンソールで表示されます。また、メッセージが MVS によってアクション・メッセージまたは要応答オペレーター宛 (WTOR) メッセージであると見なされる場合は、RETAIN インディケーターがチェックされ、画面表示カラーに関する MPF 指定が使われます。

その後、メッセージはハードコピー・ログに書き込まれますが、通常、このログはシステム・ログ・データ・セットです。メッセージがシステム・ログ・データ・セットへ書き込まれないように、MPF インストール・システム出口から指定することはできますが、この指定は通常は行われません。

シスプレックス環境で、MVS メッセージの受け渡しにサブシステム・インターフェースを使用すると、NetView プログラムは、シスプレックス内の別のシステムから送られてくるメッセージの処理を行いません。これにより、シスプレックス内の複数システムで NetView プログラムが実行しているとき、自動化の重複を防ぐことができます。

シスプレックス環境で、MVS メッセージの受け渡しに EMCS コンソールを使用すると、EMCS コンソールに MSCOPE の値を設定し、特定の EMCS コンソールだけが MVS システム・メッセージを受け取れるようにすることができます。これらのシステム・メッセージは、コンソールのシステム、シスプレックス内のすべてのシステム、またはシスプレックス内の特定のシステムのリストから送信されます。デフォルトの解釈としては、NetView プログラムが使用している EMCS コンソールがシスプレックス内のすべてのシステムからメッセージを受け取ります。CNMCSSIR タスクは、このデフォルトの解釈の例外となります。このタスクは、そのタスクが実行中であるシステムからのメッセージのみを受け取ります。

コマンドの流れ

複数コンソール・サポート・コンソールから出されるコマンドも、サブシステム・インターフェースでブロードキャストされます。コマンドの処理方法は、メッセージの処理方法と類似しています。コマンドはアクティブの各サブシステムにより検査されますが、その順序は、最初にジョブ入力サブシステムにより検査され、次に SSNT に指定されている順序に従って他のサブシステムにより検査されます。

処理の判別

各サブシステムはコマンドを調べ、その処理を判別します。例えば、JES2 は、標準的な JES2 を想定して、先頭文字がドル記号 (\$) かどうかを検査します。ドル記号の場合には、JES2 はそのアドレス・スペースで処理するコマンドとしてそのコマンドを受け付け、JES2 のコマンド・プロセッサへそのコマンドを渡します。また、JES2 はサブシステムにより処理されたことを示すマークをそのコマンドに付けるので、識別不能のコマンドに関するエラー・メッセージは出されません。

同様に、JES3 はアスタリスク (*) または 8 を検索し、NetView プログラムはデフォルトの解釈としてパーセント記号 (%) を検索します。NetView プログラムは、他のサブシステム同様、先頭文字を使用して NetView コマンドを区別します。2 つの JES サブシステムのそれぞれに異なった文字を指定すると、コマンドを別々のサブシステムに送ることができます。各サブシステムに同じ文字を指定すると、両方の JES サブシステムがそのコマンドを受け付けて処理します。これは、NetView プログラムの場合も該当します。また、NetView プログラムは、自動化タスク (自動タスク) がコマンド発行元のコンソールと関連のある NetView アプリケーション・アドレス・スペースで動作しているかどうかチェックします。該当するアプリケーションが存在するときには、コマンドは NetView サブシステム・アドレス・スペースでコピーされ、キューに入れられます。そのコマンドが後に NetView アプリケーションによりキューから削除されると、関連自動タスクへ経路指定されます。

最後に、どのサブシステムもコマンドに処理済みのマークを付けなかったときには、そのコマンドは MVS コマンドと見なされ、該当するコマンド・プロセッサが処理します。該当するコマンド・プロセッサが存在しなければ、エラー・メッセージが出されます。

コンソールから出されるコマンド

コマンドがコンソールから出されると、そのコマンドに関連する制御ブロック (CSCB) は、どのコンソールから出されたかを表示します。したがって、そのデータはコマンド・プロセッサから利用できますが、通常は、コマンドを出したコンソールへ応答メッセージを出す場合 (WTO を使用して) に使用されます。複数コンソール・サポート・コンソールから発行される NetView コマンドは、そのコンソール名から、その複数コンソール・サポート・コンソールと関連付けられている自動タスクによって処理されます。

NetView 自動タスクまたはオペレーターは、NetView MVS コマンドを使用して MVS へコマンドを入力することができますが、その場合には SVC 34 が出されます。応答がコマンドの発行元だけに送られるようにするために、NetView MVS コマンド・プロセッサは、MVS コンソールが MVS コマンドを出すタスク用に取得されていることを確認します。

サブシステム割り振り可能コンソールは、EMCS コンソールであり、物理コンソールではありません。サブシステム割り振り可能コンソールは、プログラム (JES2 および NetView プログラムなど) が使用する仮想コンソールであり、特定のコマンドだけを入力し出力を検索します。NetView タスクが MVS コマンドを入力すると、タスクにまだ MVS コンソールが割り当てられていなければ、MVS コンソールが割り当てられます。他の NetView コマンドを使用してコンソールを解放しない限り、タスクがアクティブの間、同じタスクがそのコンソールをそのまま使用します。

注: MVS またはサブシステム・コマンドのうち、コンソール ID を使用した応答を出さないコマンドもありますが、その場合、送信請求メッセージの代わりに非送信請求メッセージが出されます。コマンドが複数コンソール・サポートの設計に違反しているため、これらのコマンド応答は、コマンドの発行元には戻されません。

MVS とのインターフェースとしての NetView

NetView プログラムには、メッセージの作成元およびコマンドの送信先として使用されるすべてのシステム、サブシステム、およびネットワーク・コンポーネントとのインターフェースが備わっているため、NetView は自動化のフォーカル・ポイントとして最適です。また、NetView プログラムには、自動化テーブル、コマンド・リスト処理、およびインストール・システム出口によりメッセージを処理するためのプログラミング機能も備わっています。

NetView プログラムは、2 つのアドレス・スペースにより構成されます。その 1 つである NetView サブシステム・アドレス・スペースには、MVS サブシステム・インターフェースとのインターフェースを取るプログラムが含まれています。このアドレス・スペースは、NetView アプリケーション・アドレス・スペースのサービス・アドレス・スペースとして動作します。NetView アプリケーション・アドレ

ス・スペースには、オペレーターおよびコマンド・リストを処理するプログラムなど、さらに一般的な NetView プログラムが含まれています。

NetView プログラムで表示または処理される WTO として出されるメッセージ

MVS WTO として出されるメッセージは、サブシステム・インターフェースまたは EMCS コンソールを使用して処理することができます。

サブシステム・インターフェースを使用する WTO 処理

1 つの MVS システムでは、任意の数の NetView サブシステムを含むことができますが、各サブシステムは NetView アプリケーションと関連付けられます。NetView サブシステムは MVS WTO 処理からのメッセージを受信します。メッセージは MPF テーブルに AUTO(YES) および AUTO(token) のマークが付けられるか、あるいはメッセージがコマンド応答であれば、NetView サブシステムはそのメッセージを NetView アプリケーションに渡します。それ以外の場合には、メッセージ処理は通常どおり行われます。

EMCS コンソールを使用する WTO 処理

EMCS コンソールを使用すると、MVS はシステム・メッセージを直接これらのコンソールに送信します。サブシステム・インターフェースを使用しません。EMCS コンソールを通した送信請求メッセージ (コマンド応答) および CNMCSSIR タスクを通した非送信請求メッセージに注意してください。

RACF OPERPARM セグメントまたは MVS VARY コマンドを使用して、EMCS コンソールの属性を変更することができます。例えば、特定の EMCS コンソールが特定の経路コードを持つメッセージだけを受け取るように指定することができます。ただし、EMCS コンソールの属性を変更すると、システム・メッセージの自動化が繰り返し実行される場合もあります。

NetView プログラムによって発行される MVS コマンド

NetView コマンドは、CNMCMD というパラメーター・ライブラリー・メンバー内の CMDDEF 定義ステートメントを使用して定義します。プロダクト・サンプル CNMCMD に定義されているコマンドの中には、**MVS** と呼ばれるコマンドがあります。NetView オペレーター ID により **MVS** コマンドが出されると、NetView MVS コマンド・プロセッサによりそのオペレーター ID に割り当てられた MVS コンソールから、コマンド・オペランドのストリング全体がオペレーター・コマンドのテキストとして MVS へ送信されます。

サブシステム・インターフェースを使用すると、NetView サブシステムは、MVS からコマンド発信元のコンソールへ送られたコマンド応答メッセージを選択し、コマンドを出したオペレーター ID へ送り返します。EMCS コンソールを使用すると、コマンド応答メッセージは直接 NetView オペレーター・タスクへ送られます。

MVS コンソールからサブシステム・コマンドとして出される NetView コマンド

MVS コンソールから NetView コマンドを出すには、次の 3 つの条件が満たされていなければなりません。

- NetView コマンドのコマンド接頭部文字を認識するために NetView サブシステムはアクティブでなければなりませんが、この接頭部文字はコマンド・テキストの残りを NetView アプリケーションに渡して処理することを指示しています。デフォルトのコマンド接頭部文字は、パーセント記号 (%) です。
- NetView アプリケーションで CNMCSSIR タスクが実行中でなければなりません。
- NetView オペレーター ID および関連 MVS コンソールの両方を指定して開始された自動タスクが実行中でなければなりません。

MODIFY (F) コマンドを使用した MVS コンソールからの NetView コマンドの発行

MVS コンソールから MODIFY コマンドを使用して NetView コマンドを発行するには、NetView オペレーター ID および関連 MVS コンソール名の両方を指定して開始された自動タスクが実行中でなければなりません。

VTAM インターフェースを使用したメッセージおよびコマンド

以下のセクションでは、さまざまな VTAM インターフェースを使用したメッセージおよびコマンドの処理方法について説明します。

端末アクセス機能 (TAF)

端末アクセス機能 (TAF) オペレーター制御セッションは、CICS および IMS などの LU1 セッションをサポートする VTAM アプリケーションへ NetView オペレーター ID を関連付けるために NetView プログラムにより使用されます。アプリケーション・プログラムから送られるメッセージは、TAF により NetView メッセージ・バッファーとしてまとめられ、TAF オペレーター制御セッションを所有するオペレーター ID のキューに入れられます。同様に、NetView SENDSESS コマンドにより、そのオペレーター ID からアプリケーション・プログラムへコマンド・テキストが送られます。

インターフェース

NetView 基本プログラム・オペレーター・インターフェース・タスク (PPT) がアクティブであり、AUTH=PPO が指定された VTAM アプリケーションとして定義されていると、非送信請求 VTAM メッセージは VTAM プログラム・オペレーター・インターフェース (POI) を介して NetView プログラムへ送信されます。POI がアクティブでないとき、非送信請求 VTAM メッセージは WTO を介して出され、MVS サブシステム・インターフェースを介して NetView プログラムから使用することができます。NetView が POI を介して VTAM プログラムへコマンドを送ると、応答はそのコマンドを出した NetView オペレーターの ID と関連付けられます。

通信ネットワーク管理インターフェース

アラートは、VTAM の通信ネットワーク管理インターフェース (CNMI) を通して非送信請求データとして VTAM から NetView プログラムへ送ることができます。また、MVS はハードウェア・モニターのローカル装置レコードを使用してアラート・データを NetView へ送信することができます (VTAM がアクティブの場合もそうでない場合も)。最後に、NetView 内の GENALERT コマンド・プロセッサが出さ

れ、ハードウェア・モニターが処理できるようにアラートを生成します。(この場合も、VTAM がアクティブかどうかは関係ありません。)

フィルター

NetView プログラムがアラート・データを受信すると、そのデータの処理方法は、NetView SRF コマンドにより設定されているハードウェア・モニター・フィルターにより決定されます。フィルター・オプションを使用して、選択したアラートのデータを NetView BNJ146I メッセージのテキストに変換することができます。変換されたメッセージは、通常の NetView 経路指定および自動化処理の対象になります。

通信ネットワーク管理

VTAM 通信ネットワーク管理 (CNM) データは、NetView の DSICRTR タスクがアクティブであり VTAM アプリケーション (APPL 名は DSICRTR および AUTH=CNM) として定義されているときに限り、NetView プログラムにより受信されます。VTAM のグローバル経路指定テーブルは、指定されたタイプの非送信請求データ・バッファーを受信するために、1 つの AUTH=CNM アプリケーションを決定します。そのテーブルでは、すべてのアラート・データは 1 つの APPL (名前は DSICRTR) へ経路指定されます。1 つのシステムにつき 1 つの NetView プログラムだけがオペレーティング・システムからハードウェア・モニターのローカル装置レコードを受信することができます。システムで複数の NetView プログラムが実行している場合には、ハードウェア・モニターのローカル装置レコードを受信できるのは 1 つの NetView プログラムだけであり、DSICRTR という名前の CNM APPL をオープンできるのも 1 つの NetView プログラムだけです。

同一システムでの 2 つの NetView プログラムの実行について詳しくは、513 ページの『第 32 章 単一システムでの複数 NetView プログラムの実行』を参照してください。

VTAM CNMI を備えていない NetView アプリケーションで GENALERT からのアラートを処理するには、次の操作を実行してください。

1. NetView プログラムで生成されるアラートを処理するには、DSICRTR タスクの DSICRTTD 初期設定メンバーに DSTINIT FUNCT=OTHER を指定してください。
2. 図 180 のようなステートメントを指定して、NetView プログラムの AUTH=CNM アプリケーションを開始させてください。

```
// EXEC PGM=BNJLINTB
```

図 180. AUTH=CNM アプリケーションを開始させるステートメント

3. 図 181 に示すステートメントを指定して、他の NetView アプリケーション・プログラムを開始させてください。

```
// EXEC PGM=DSIMNT
```

図 181. 他の NetView アプリケーション・プログラムを開始させるステートメント

コンソールの操作

MVS のオペレーター・コンソールはローカルに接続されたシステム割り振り可能装置で、システム COMMTASK へ直接割り振られます。これらのコンソールは通常 3270 表示装置であり、オペレーティング・システム・コンポーネント、ジョブ入力サブシステム、アプリケーション・サブシステム、およびシステムまたはシステムのローカル複合体で実行するアプリケーション・プログラムから送られてくる WTO メッセージを表示するために使用されます。

コンソール・オペレーターは、これらのコンソールからシステム・コマンド、サブシステム・コマンド、およびプログラム応答を入力します。VTAM プログラムが MVS のもとで実行しているときには、システム・オペレーターはシステムの MODIFY、REPLY、DISPLAY、VARY、および HALT コマンドとのインターフェースを使用して、これらのコマンドをシステム・コンソールから VTAM へ出すことができます。VTAM メッセージを受信する NetView PPT が使用できないときには、VTAM メッセージは WTO メッセージとして出されます。

NetView コンソールは、VTAM ログオン処理により NetView に接続される 3270 表示装置です。NetView コンソールで、ネットワーク・オペレーターは割り当てられた ID およびパスワードまたはパスワード・フレーズを使用して、ログオンすることができます。NetView プログラムで採用されている VTAM POI を通じ、VTAM メッセージがネットワーク・オペレーターに表示され、VTAM コマンドがネットワーク・オペレーターから受信されます。NetView プログラムから出されたメッセージは表示され、NetView コマンドは受信されます。

コマンドおよびコマンド・リストをサブシステム・コマンドとして発行するための MVS オペレーター・コンソールの使用

システム・コンソールからは、NetView が MVS サブシステムと同じように見えます。NetView コマンドまたはコマンド・リストは、以下の条件がすべて満たされていれば、NetView サブシステムにより受け入れられ、関連 NetView アプリケーションへ渡されます。

- サブシステム名が MVS に定義されている。

SYS1.PARMLIB では、システム・パラメーター SSN=(xx) で参照するサブシステム名テーブル・メンバー IEFSSNxx に、有効なサブシステム名の定義が含まれています (MVS ライブラリーを参照)。

- NetView サブシステムがアクティブである。

NetView インストール時に、NetView サブシステムをアクティブにする START コマンドが SYS1.PARMLIB メンバー COMMNDxx に追加されます。

- NetView 始動プロシージャの PARM フィールドに定義されているコマンド接頭部文字が、MVS コンソールから入力されるコマンドの先頭文字として使用されている。

NetView が提供するサンプル始動プロシージャで指定されているデフォルトのコマンド接頭部文字は、パーセント記号 (%) です。接頭部文字を変更するには、「*IBM Tivoli NetView for z/OS*インストール:概説」の指示に従ってください。

- 4 文字のサブシステム名で始まるジョブ名 (START コマンドで使用される) を持つ NetView アプリケーションが完全に開始済みの CNMCSSIR タスクと一緒に現在アクティブになっている。

CNMCSSIR タスクを定義する TASK ステートメントは、CNMSTYLE メンバーに指定されていて、そのまま使用します。

NetView コマンドの STARTCNM ALL は、CNMSTYLE メンバー内の指定で起動することができます。CNMSTYLE メンバーの詳細については、「*IBM Tivoli NetView for z/OS*インストール:概説」に記載されています。

- NetView プログラムと複数コンソール・サポート・コンソールを関連付けるために、AUTOTASK コマンドが入力されています。

自動タスクとが関連付けられていないコンソールから NetView コマンドを出すと、NetView サブシステムからエラー・メッセージが返され、そのコンソールには NetView サブシステムを使用する許可がないことが示されます。CNMSTYLE メンバーの AUTOTASK ステートメントは、AUTO2 および MVS コンソール用の自動タスクを開始します。NetView コマンドを発行するそれぞれの MVS コンソールを有効な OPID と関連付けるには、CNMSTYLE メンバーに類似のステートメントを追加します。

AUTOTASK コマンドについて、詳しくは、NetView オンライン・ヘルプを参照してください。

コマンドおよびコマンド・リストを MODIFY (F) コマンドとして発行するための MVS オペレーター・コンソールの使用

システム・コンソールからは、NetView は MVS サブシステムと見なされます。複数コンソール・サポート・コンソールと NetView プログラムを関連付けるための AUTOTASK コマンドが入力済みである場合は、あらゆる NetView コマンドまたはコマンド・リストは NetView サブシステムに受け入れられ、関連付けられている NetView アプリケーションに渡されることがあります。

自動タスクとが関連付けられていないコンソールから NetView コマンドを出すと、NetView サブシステムからエラー・メッセージが返され、そのコンソールには NetView サブシステムを使用する許可がないことが示されます。CNMSTYLE メンバーの AUTOTASK ステートメントは、AUTO2 および MVS コンソール用の自動タスクを開始します。NetView コマンドを発行するそれぞれの MVS コンソールを有効な OPID と関連付けるには、CNMSTYLE メンバーに類似のステートメントを追加します。

AUTOTASK コマンドについて、詳しくは、NetView オンライン・ヘルプを参照してください。CNMSTYLE メンバーについての詳細は、「*IBM Tivoli NetView for z/OS*インストール:概説」を参照してください。

複数コンソール・サポート・オペレーターによるコマンド・リストの使用

NetView 自動化の簡単な適用例は、オペレーターが複数コンソール・サポート・コンソールから実行することができるコマンド・リストの作成です。そのためには、

コンソールに関連付けられる自動タスクと処理されるコマンド・リストのセットが必要です。例えば、特定のマウント属性を指定しなければならない DASD セットをオンラインにする場合、オペレーターは図 182 のような一連のコマンドを入力する必要があります。

```
M 350,VOL=(SL,VOL001),USE=STORAGE
M 351,VOL=(SL,VOL002)
⋮
```

図 182. DASD をオンラインにするために使用するコマンド

マウントしなければならないボリュームが多いと、このシーケンスは非常に長くなります。しかし、必要な MVS コマンドすべてを NetView プログラムから実行するコマンド・リストを作成することができます。オペレーターはこれを使用して、NetView 指定文字およびコマンド・リスト名を入力すれば、DASD をオンラインにすることができます。

NetView オペレーター ID からの MVS コマンドの発行

NetView オペレーター ID から MVS オペレーター・コマンドを出すには、コマンド MVS を入力し、その次にスペースを入力し、スペースの次に MVS オペレーター・コンソールから出す場合と同じように MVS オペレーター・コンソール・コマンドを入力します。NetView がコマンド許可を使用するときには、オペレーター ID がコマンドを出せるようにしなければならないことに注意してください。「*IBM Tivoli NetView for z/OS* アドミニストレーション・リファレンス」を参照してください。

EMCS コンソールの使用

EMCS コンソールを使用すると、オペレーター名ごとに NetView コンソールを定義し、MVS、NetView プログラム、および RACF (リソース・アクセス管理機能) のようなシステム許可機能 (SAF) プロダクトの定義を使用して、さまざまなセキュリティ・クラスと許可クラスを指定することができます。

EMCS コンソールが割り当てられていないオペレーターが MVS コマンドを出すと、NetView プログラムが、コンソール名としてオペレーター ID を使用して EMCS コンソールを獲得しようとします。

そのコンソール名に問題があれば、GETCONID コマンドを使用して、EMCS コンソールを獲得することもできます。EMCS コンソールの数には制限がありません。

SETCONID コマンドは、コンソール名を、実際には割り振らずに、割り当てののに使用します。

付録 E. VTAM メッセージおよびコマンド処理

この付録では、NetView プログラムにより提供される診断、修正、およびチューニング情報について説明します。

重要: この診断、修正、チューニング情報を、プログラム・インターフェースとして使用しないでください。

VTAM オペレーターは、VTAM メッセージから情報を得ます。しかし、メッセージ量が多いため、システム・パフォーマンスが低下し、オペレーターが重要な情報を見落とす危険もあります。メッセージ表示頻度を制御するために、VTAM ではいくつかのメッセージ抑止のメカニズムを提供しています。この付録では、VTAM メッセージおよびコマンドの流れ、およびメッセージ抑止のメカニズムについて説明します。

VTAM のメッセージおよびコマンドの流れ

VTAM メッセージには、送信請求メッセージと非送信請求メッセージがあります。送信請求メッセージは DISPLAY などのコマンドの応答として出され、通常はコマンドを入力したオペレーターへ返されます。非送信請求メッセージは、システム・コンポーネントの状況情報を知らせるために、通常の操作時に VTAM により出されます。

NetView オペレーターがプログラム・オペレーター・インターフェース (POI) を介して VTAM にコマンドを実行依頼すると、送信請求応答が、そのコマンドを出した NetView オペレーター ID と関連付けられます。NetView の基本 POI タスク (PPT) が VTAM アプリケーション (AUTH=PPO) として定義されている場合、非送信請求 VTAM メッセージは VTAM POI を介して NetView へ送られます。VTAM アプリケーション (AUTH=PPO) として定義されていない場合、非送信請求 VTAM メッセージは、MVS サブシステム・インターフェースを介して NetView により受信されます。

VTAM メッセージはオペレーティング・システムを介して NetView へ常に渡されるところとは限らないため、メッセージ処理機能 (MPF) は、常に VTAM メッセージにより駆動されるところとは限りません。つまり、抑止されるメッセージとして MPF 内に指定されている VTAM メッセージは、NetView プログラムと VTAM との間の POI インターフェースが使用できないときに限り、抑止されます。そのため、VTAM メッセージは、VTAM メッセージあふれ防止テーブル (602 ページの『メッセージあふれ防止テーブル』を参照) または自動化テーブルにより抑止してください。

POI およびサービス・ポイント操作 (SPO) の定義時に、1 つのシステムに 2 つの NetView プログラムが存在するときには、特別な注意が必要です。詳しくは、「513 ページの『第 32 章 単一システムでの複数 NetView プログラムの実行』」を参照してください。

メッセージあふれ防止テーブル

VTAM メッセージあふれ防止テーブルは、大量のメッセージが、基本的なイベントまたは状態のあとに出されるか、または頻繁に繰り返される場合に使います。このテーブルは、メッセージ過剰の潜在的な原因として識別されるメッセージのリストです。このようなメッセージは、規定時間 (デフォルト値は 30 秒) 内に可変フィールドが変わらずに繰り返されると抑止されます。メッセージあふれ防止のための以下の抑止規則は、VTAM MODIFY SUPP コマンドと同じものです。

- メッセージは、VTAM 内部トレース・テーブルに記録されます。
- メッセージは作成されますが、オペレーターへ送信されません。他の領域 (例えば、ネットワーク・ログ) へ経路指定される場合があります。
- 複数行のオペレーター宛 (MLWTO) メッセージ・グループの最初の行が抑止されると、そのグループのすべての行が抑止されます。
- 不定様式システム・サービス (USS) メッセージは抑止されません。

オペレーター・コマンドを出した結果生成されるメッセージは、メッセージがメッセージあふれ防止テーブルのメンバーであり、オペレーターが同じコマンドを規定時間内に出すと、抑止されます。

MLWTO メッセージ・グループのヘッダーが抑止されると、そのグループ内のすべてのメッセージも抑止されます。その情報が直前の情報と異なっている場合、抑止されます。詳しくは、z/OS Communications Server ライブラリーを参照してください。

VTAM メッセージを抑止すると、AON/SNA 自動化に影響を及ぼす可能性があります。自動化テーブル DSITBL01 においてもトラップされているメッセージ抑止テーブルには、どのような VTAM メッセージもコーディングしないでください。

VTAM メッセージの抑止基準

どの抑止方法を実施するかを決定する際の作業の 1 つは、非送信請求 VTAM メッセージの頻度を測定し、その頻度が高いかどうかを判別することです。一定の時間に出されたメッセージ数をカウントし、そのカウントと、決められている限界値とを比較することができます。システムが異なっていれば属性も異なるため、その環境に合わせた頻度を検討してください。頻度が高すぎるときには、自動メッセージ抑止を検討してください。自動化テーブルおよびコマンド・リストによって、メッセージ通信量をモニターし、メッセージの選択的抑止を自動的に行うことができます。

自動化テーブルによるイベントの認識

イベントを認識するためには、主に NetView 自動化テーブルが使用されますが、このテーブルはメッセージが到着するたびに検索されます。検索引数 (メッセージの任意の箇所に指定可能) が見つからない場合、メッセージは通常の処理のために NetView プログラムにより戻されます。つまり、そのメッセージはコンソールに表示されます。検索が正常に終了すると、メッセージを保持するか、あるいはコンソールから削除することができます。また、メッセージによりコマンド・プロセッサまたはコマンド・リストにさらにアクションを実行させることもできます。

大部分の VTAM メッセージは USS テーブル (主要なテーブルは ISTINCNO) 内にあり、USSMSG マクロにより定義されます。メッセージはそのマクロにより修正可能ですが、通常は IBM 提供のテーブルを修正するのではなく、新しいテーブルを作成することをお勧めします。

抑止レベルについて

USSMSG マクロの SUPP パラメーターはメッセージ・クラスの割り当てに使用されますが、このクラスとメッセージ抑止レベルは、メッセージを表示するかどうかを判別する際に使用されます。これらのクラスを重大度の低いレベルから高いレベルへ並べると、次のようになります。

SUPP=INFO

通知

SUPP=WARN

警告

SUPP=NORM

通常

SUPP=SER

重大

上記の 4 つのクラス以外にも、メッセージに SUPP=ALWAYS または SUPP=NEVER を定義できますが、これらはその時点で有効な抑止レベルとは無関係に指定することができます。すべての標準的な VTAM オペレーター・メッセージおよび抑止クラスの説明については、z/OS Communications Server ライブラリーを参照してください。

メッセージ抑止レベルは、VTAM START コマンドまたは MODIFY SUPP コマンドの SUPP パラメーターにより設定されます。どちらのコマンドの場合も、SUPP パラメーターには、上記 4 つのクラスのいずれかを指定します。これら 4 つの抑止クラスのいずれかがメッセージに指定されると、そのレベルとその上位レベルで抑止されます。

最後に、SUPP=NOSUP を指定すると抑止機能を無効にすることができます。その場合、SUPP=ALWAYS が定義されたメッセージだけが抑止されます。

抑止できないメッセージの識別

メッセージに SUPP=NEVER を定義すると、そのメッセージに設定されている抑止レベルにかかわらず、メッセージは抑止されません。これらのメッセージ (抑止不能メッセージと呼ばれる) には、以下のメッセージがあります。

- タスクの異常終了により生成されたエラー・メッセージ
- オペレーターの応答が必要なメッセージ (接頭部が A のアクション・メッセージ)
- DISPLAY または START コマンドの結果生成されたメッセージ

複数行 WTO メッセージの個別行を抑止することはできません。つまり、ヘッダー行が抑止されたら、そのグループに属するすべての行が抑止されます。

メッセージの処理を自動化するには、まず、メッセージが特定のオペレーターに経路指定して処理されるように、NetView プログラムを初期設定する必要があります。NetView ASSIGN コマンドを使用すると、非送信請求メッセージは、そのメッセージを処理する特定のオペレーター端末タスク (OST) へ直接経路指定されます。その後、自動化テーブルはそのタスクのもとで実行されます。メッセージが ASSIGN により割り当てられないときには、それらのメッセージは自動化テーブルを介して経路指定されなければならないので、すべてのメッセージが 1 つのタスクのキューに入れられ、処理が遅れる可能性があります。

付録 F. NetView メッセージおよびコマンドの詳細な流れ

この付録では、NetView プログラムにより提供される診断、修正、およびチューニング情報について説明します。

重要: この診断、修正、チューニング情報を、プログラム・インターフェースとして使用しないでください。

この付録には、NetView 内のメッセージおよびコマンドの流れを示す図とその説明があります。説明では、番号が付けられている各出口ルーチン (DSIEX01、DSIEX02A など) をどこで指定するか、およびその出口ルーチンがどのようにコマンドと関連して処理されるかを示しています。メッセージ処理の順序および文脈に関する説明は、自動化テーブル、ASSIGN コマンド、およびその他のメッセージ処理機能を使用してメッセージを自動化する際に特に役立ちます。

フローチャート

本セクションの図では、NetView プロダクト内におけるメッセージとコマンドの流れを表しています。図の中の番号が付けられているタグは、613 ページの『流れの説明』での番号付きのトピックに対応しています。

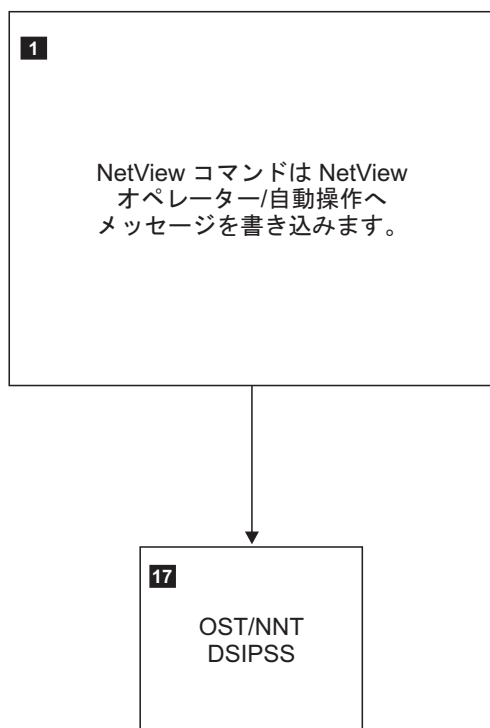


図 183. NetView コマンド入力のフローチャート (VTAM 端末)

NetView オペレーターはクロスドメイン・コマンドを入力します。

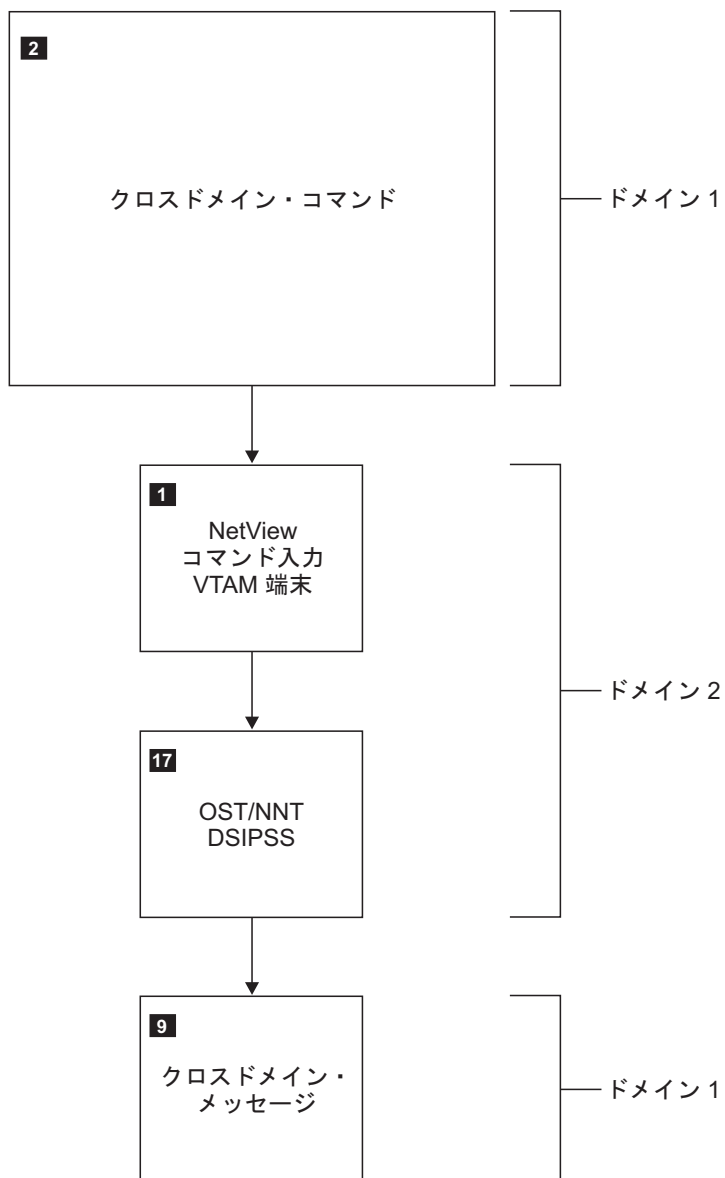


図 184. ドメイン間コマンドのフローチャート

NetView オペレーターは VTAM コマンドを入力します。

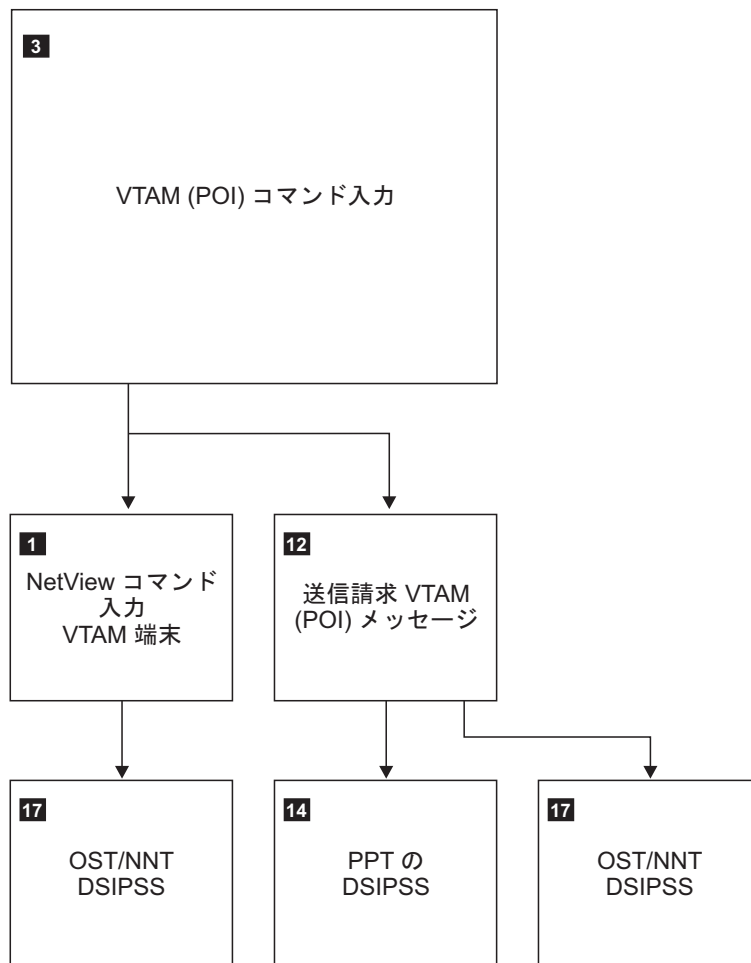


図 185. VTAM (POI) コマンド入力のフローチャート

NetView オペレーターは MVS システム・コマンドを入力します。

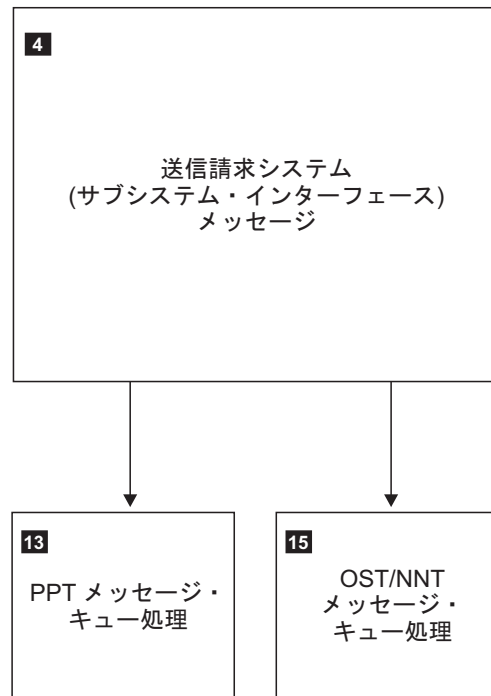


図 186. 送信請求システム (サブシステム・インターフェース) メッセージのフローチャート

システム・オペレーターは NetView コマンドを入力します。

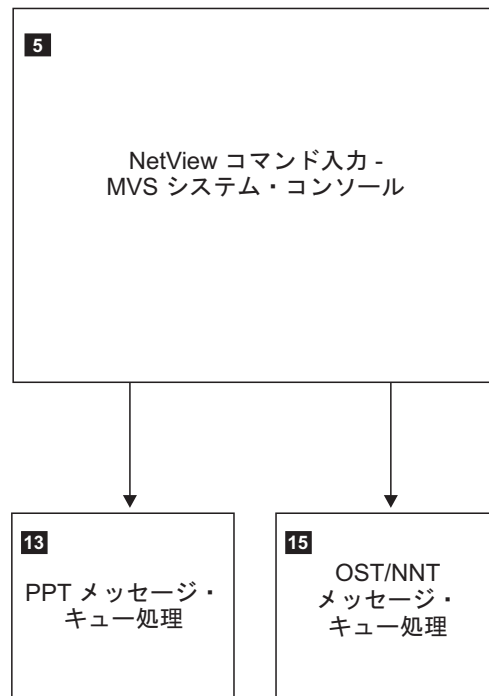


図 187. NetView コマンド入力のフローチャート (MVS)

システム・オペレーターは、NetView WTOR に応答します。



図 188. NetView WTOR に対する応答のフローチャート

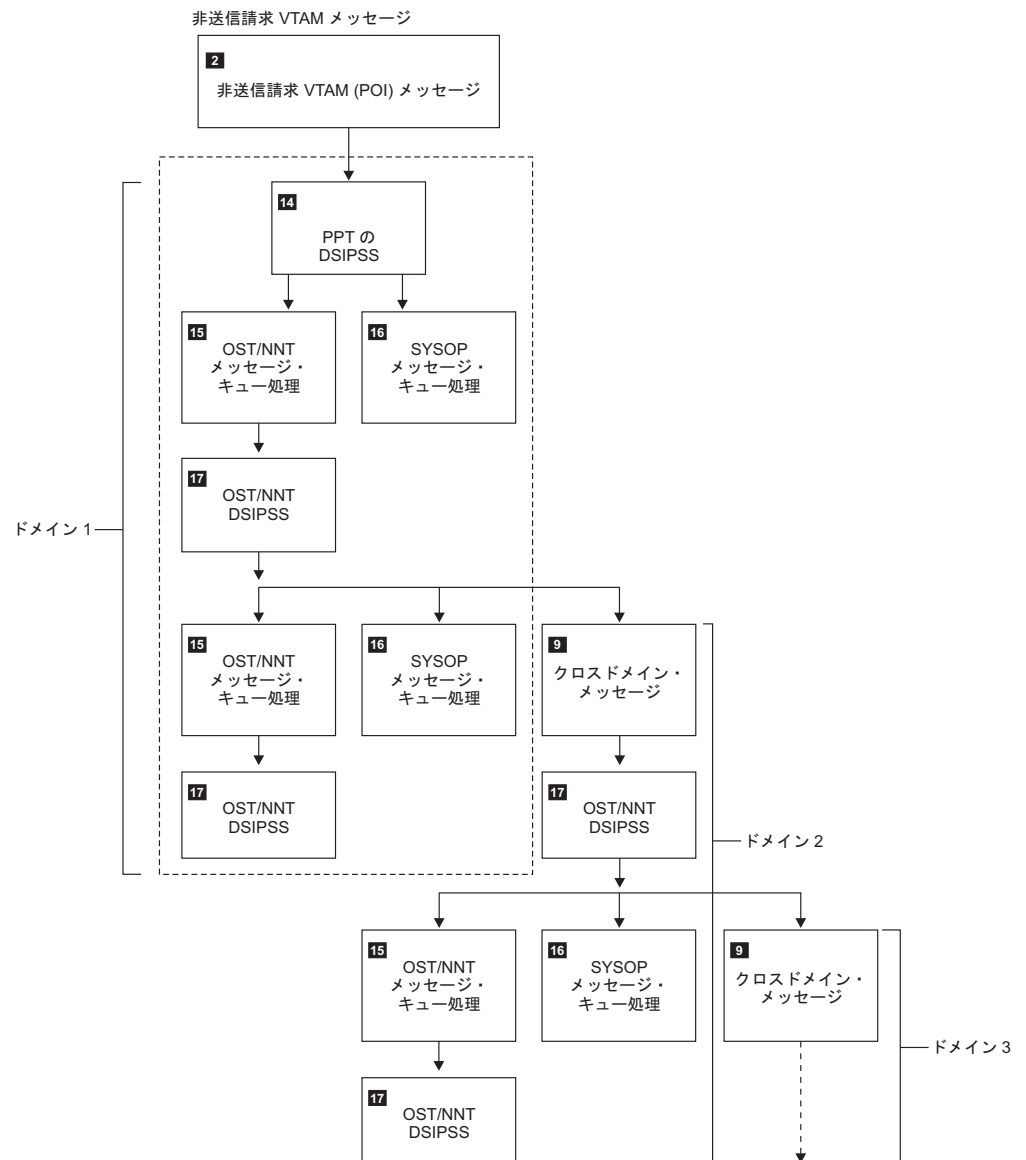


図 189. 非送信請求 VTAM (POI) メッセージのフローチャート

非送信請求メッセージ

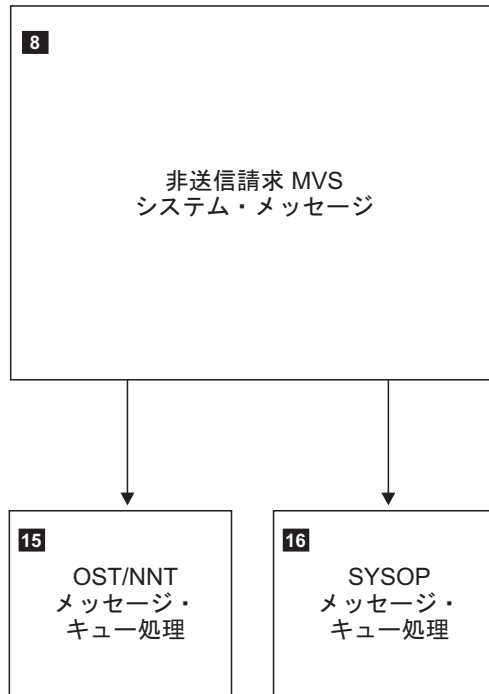


図 190. 非送信請求システム (SSI または MVS 拡張コンソール) メッセージのフローチャート (CNMCSSIR)

NNT は OST へメッセージを送信します。

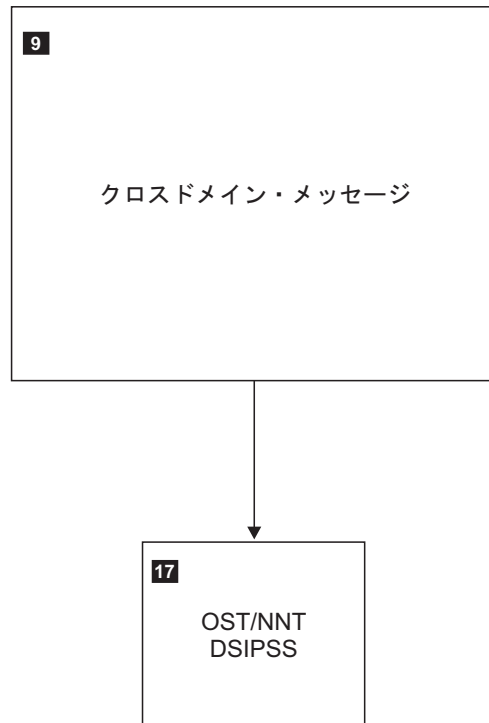


図 191. ドメイン間メッセージ (NNT から OST へ) のフローチャート

オペレーターは PPT です。

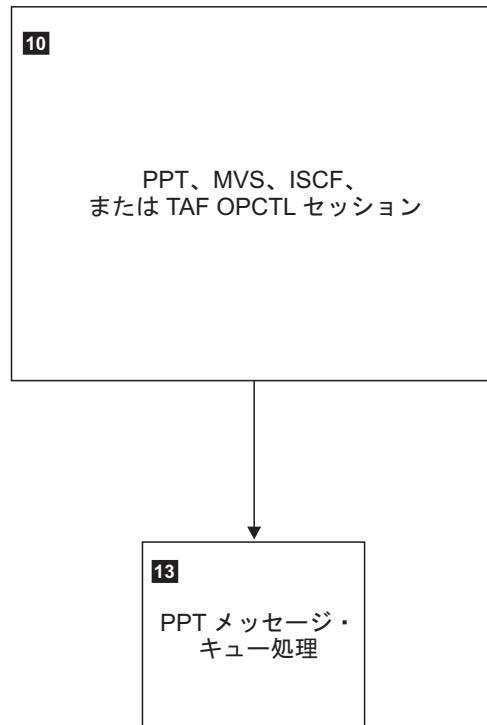


図 192. メッセージのフローチャート (オペレーターは PPT)

オペレーターは OST/NNT です。

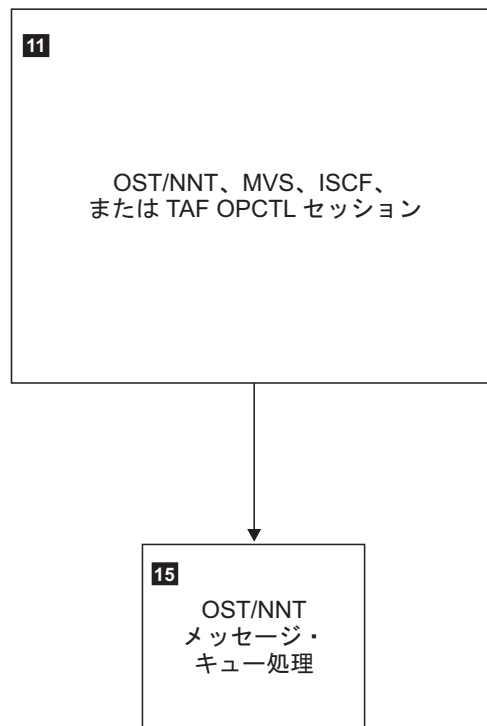


図 193. メッセージのフローチャート (オペレーターは OST/NNT)

MVS は、拡張 MCS コンソールへメッセージを送信します。

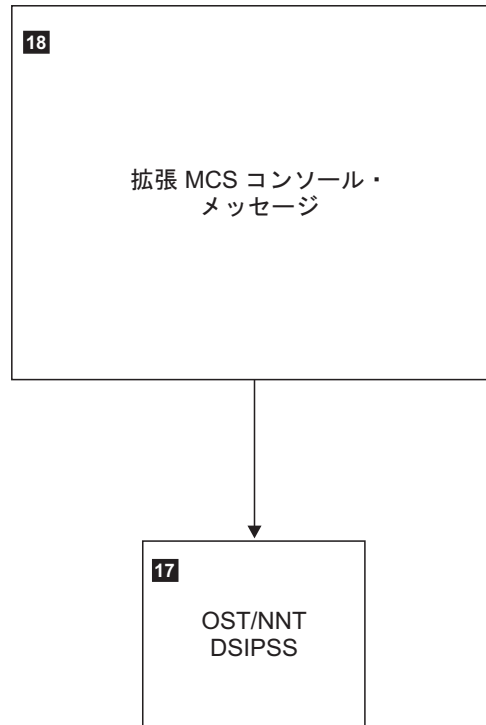


図 194. OST、NNT、または自動タスク用の送信請求および非送信請求システム MVS 拡張コンソール・メッセージのフローチャート

MVS は、拡張 MCS コンソールへメッセージを送信します。

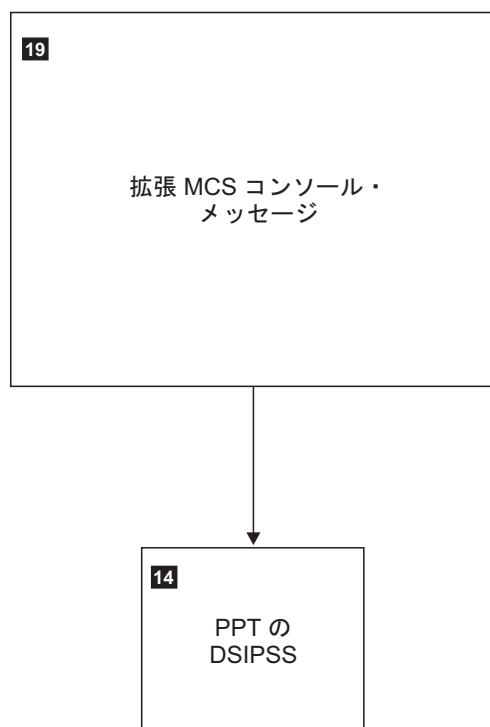


図 195. PPT 用の送信請求および非送信請求システム MVS 拡張コンソール・メッセージのフローチャート

流れの説明

このセクションでは、NetView プログラム内のメッセージとコマンドの流れについて説明します。それぞれの流れの説明には、以下の情報が含まれています。

原因 それぞれの流れを開始させる条件またはイベント

元タスク

条件またはイベントが発生したタスク

処理の流れ

メッセージ処理の順序

1. NetView コマンド入力 (VTAM 端末)

原因: NetView オペレーターが行モード・コマンドを入力するか、または NetView 間タスク (NNT) が OST からコマンドを受け取る。コマンドは、MVS、または TAF によって経路指定されます。

元タスク: OST または NNT

処理の流れ:

1. DSIEX01 が呼び出されます (TVBINXIT=ON)。

出口でコマンドを削除すると、コマンドはこれ以上処理されません。

2. コマンド・プロセッサが呼び出され、処理は次のように継続されます。
 - 即時コマンドが非同期入力出口から実行され、TVBINXIT=ON になります。
 - 通常のコマンドが通常のタスク処理から実行され、TVBINXIT=OFF になります。
 - コマンドは DSIPSS を出すことができます。624 ページの『17. OST または NNT DSIPSS』を参照してください。

2. ドメイン間コマンド (OST から NNT へ)

原因:

- NetView ROUTE コマンド (DSIRTP)
- 暗黙または明示のドメイン間経路指定を指定した NetView VTAM コマンド (DSIVTP)

元タスク: OST または NNT

処理の流れ:

1. DSIEX07 が呼び出されます。

出口でコマンドを削除すると、コマンドはこれ以上処理されません。

2. コマンドは NNT へ送信されます。

613 ページの『1. NetView コマンド入力 (VTAM 端末)』を参照してください。

3. VTAM (POI) コマンド入力

原因: オペレーターまたはコマンド・リストが、関連 CMDMDL 定義が DSIVTP を指定している NetView コマンド・プロセッサを使用し、VTAM コマンドを入力する。

元タスク: OST、NNT、または基本プログラム・オペレーター・インターフェース (POI) タスク (PPT)

処理の流れ:

1. タスクが OST または NNT であり、コマンドが暗黙または明示のドメイン間 VTAM コマンドの場合。
 - a. DSIEX07 が呼び出されます。出口でコマンドを削除すると、コマンドはこれ以上処理されません。
 - b. コマンドは NNT へ送信され、他のドメイン内の NNT のもとで実行します。必要に応じて、明示的な経路指定が 1 つ削除されます。613 ページの『1. NetView コマンド入力 (VTAM 端末)』を参照してください。
2. 上記以外の場合。
 - a. DSIEX05 が呼び出されます。出口でコマンドを削除すると、コマンドはこれ以上処理されません。
 - b. コマンドは、VTAM へ送信されます。
 - c. メッセージは、非同期的に戻されます。619 ページの『12. 送信請求 VTAM (POI) メッセージ』を参照してください。

4. 送信請求システム・メッセージ

原因: MVS コマンドが NetView プログラムから入力されたり、MVS がコンソール名を使用して WTO を出す。コンソール名が使用されないときには、メッセージは非送信請求され、NetView に非送信請求メッセージと見なされます。

注: MVS コマンドが NetView プログラムに所有されているコンソールから出され、さらにコマンドの応答に AUTO(YES) と SUP(YES) が指定されている場合、そのメッセージは CNMCSSIR タスクのもとで自動化されます。そのメッセージは、非送信請求の MVS システム・メッセージとして処理されます。

元タスク: OST、NNT、または PPT。メッセージが実際に自動化されるかどうか自動化テーブルと比較チェックされると、NetView プログラムは、NetView プログラムから送られてくるサブシステム・インターフェース・メッセージを無視します。

処理の流れ: メッセージが自動化内部機能要求 (AIFR) に変換されたあと、DSIEX17 が呼び出されます。

DSIEX17 は、メッセージの場合にもオペレーター・メッセージ削除 (DOM) コマンドの場合にも呼び出されます。メッセージは、関連タスクに送信されます。620 ページの『13. PPT メッセージ・キュー処理』および 622 ページの『15. OST または NNT メッセージ・キューの処理』を参照してください。

5. NetView コマンド入力 (MVS システム・コンソール)

原因: システム・オペレーターが NetView サブシステム指定機能を付けてコマンドを入力し、また、以下の条件下にある。

- NetView サブシステムがアクティブである。
- CNMCSSIR タスクがアクティブである。
- 図 196 に示されているコマンドが NetView プログラムで出され、NetView 自動タスクとこのシステム・コンソールが関連付けられている。

```
AUTOTASK OPID=operid,CONSOLE=name
```

図 196. 自動タスクをシステム・コンソールと関連付けるコマンド

元タスク: CNMCSSIR

処理の流れ: HDRMTYPE=HDRTYPET を持つコマンドが OST、NNT、または PPT へ送信されます。

620 ページの『13. PPT メッセージ・キュー処理』および 622 ページの『15. OST または NNT メッセージ・キューの処理』を参照してください。

6. NetView WTOR に対する応答

原因: オペレーターが、NetView WTOR のシステム応答コマンドを入力する (メッセージ番号 DSI802A および DSI803A)。

元タスク: DSIWTOMT

処理の流れ:

1. オペレーターは WTOR に応答します。
2. NetView メインタスクは DSIEX10 を呼び出します。

出口で入力を削除すると、入力はいくつか以上処理されません。

3. メインタスクは、CLOSE、REPLY、または MSG コマンドを処理します。
4. REPLY コマンド (IFRCODCR がオンのコマンド) は PPT タスクへ渡され、処理されます。

NetView PPT は DSIEX03 を呼び出します。出口でコマンドを削除すると、コマンドはいくつか以上処理されません。削除しない場合、コマンドは PPT のもとで実行します。

7. 非送信請求 VTAM (POI) メッセージ

原因: VTAM が NetView PPT (AUTH=(PPO) が指定されているアプリケーション) へ非送信請求メッセージを送信する。

元タスク: PPT

処理の流れ:

1. VTAM MSGMOD オプションがアクティブの場合
 - PPT は MSGMOD によりメッセージをログに記録します (診断用)。

DSIEX04 出口ルーチンは、ログ処理時に呼び出されます。 DSIEX04 を使用して、ハードコピー・ログ、ネットワーク・ログ、またはシステム・ログへ送信するか、あるいはメッセージを削除するか、置き換えるかどうかを指定します。
 - PPT は MSGMOD ID を削除し、メッセージを自動化で処理できるようにします。
 - PPT は、MSGMOD がアクティブでない場合と同様に処理を継続します。
2. メッセージがネットワーク状況を更新するために状況モニターで使用される場合、そのメッセージは状況モニターにより処理されます。
3. PPT は DSIEX11 を呼び出します。出口でメッセージを削除すると、メッセージはいくつか以上処理されません。
4. メッセージが PPOLOG メッセージの場合、PPT はメッセージをログに記録します。

DSIEX04 は、ログ処理時に呼び出されます。戻りコードをレジスター 15 に書き込むと、DSIEX04 はメッセージがハードコピー、ネットワーク、またはシステム・ログへ送信するかどうか判別します。

PPOLOG メッセージに関しては、これ以上の処理は行われません。

5. それ以外の場合、許可レシーバーが PPT メッセージの宛先です。PPT は DSIPSS TYPE=OUTPUT を出して、許可レシーバーへメッセージを送信します。620 ページの『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』を参照してください。

8. 非送信請求 MVS システム・メッセージ

原因: MVS WTO が NetView サブシステムを介して CNMCSSIR タスクへ経路指定される。

元タスク: NetView オペレーターに割り当てられていないコンソールに WTO を出す、任意のアドレス・スペース内のタスク。

処理の流れ: メッセージが自動化内部機能要求 (AIFR) に変換されたあと、DSIEX17 が呼び出されます。DSIEX17 は、メッセージの場合にもオペレーター・メッセージ削除 (DOM) コマンドの場合にも呼び出されます。この処理は OST または NNT DSIPSS 処理と PPT DSIPSS 処理と並行して行われます。620 ページの『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』との相違点は、CNMCSSIR が許可レシーバーを検索しないことです。検索した場合、すべてのシステム・メッセージが無分別に経路指定されます。

1. メッセージに ASSIGN PRI が指定されていると、メッセージは指定されたオペレーターに送信されます。

622 ページの『15. OST または NNT メッセージ・キューの処理』および 623 ページの『16. NetView コンソール出力または SYSOP メッセージ・キューの処理』を参照してください。

ASSIGN SEC が指定されたメッセージは、この NetView ドメインの自動化テーブルによって処理されることはありません。別の NetView ドメインへ送信されると、これらのメッセージは DSIEX02A、自動化テーブル、および DSIEX16 の処理の対象になります。

2. その他のメッセージの場合

- a. DSIEX02A が呼び出されます。

出口でメッセージを削除すると、これ以上の処理は行われません。DSIEX02A は、NetView ドメイン内の固有なメッセージごとに一度のみ呼び出されます。

呼び出されたあと、ASSIGN コマンドまたは自動化テーブルによりメッセージのコピーが作成されても、この NetView ドメインでは DSIEX02A を呼び出しません。メッセージのコピーが別の NetView ドメインへ送信されると、そのドメイン内の DSIEX02A が呼び出される可能性があります。

- b. 自動化テーブルがチェックされます。テーブルでコマンド・アクションおよび表示アクションを選択することがあります。

自動化テーブルは、NetView ドメイン内の固有なメッセージごとに一度のみ呼び出されます。呼び出されたあと、ASSIGN コマンドまたは自動化テーブルによりメッセージのコピーが作成されても、この NetView ドメインでは自動化テーブルは呼び出されません。

メッセージのコピーが別の NetView ドメインへ送信されると、そのドメインでは自動化テーブルが呼び出される可能性があります。

- c. この時点で、DSIEX16 が呼び出されます。
- d. DSIEX16 により示されるアクションが実行されます。バッファー構造により、取るべきアクションが決定されます。

DSIEX16 は、NetView ドメイン内の固有なメッセージごとに一度のみ呼び出されます。呼び出されたあと、ASSIGN コマンドまたは自動化テーブルによりメッセージのコピーが作成されても、この NetView ドメインでは DSIEX16 は呼び出されません。メッセージのコピーが別の NetView ドメインへ送信されると、そのドメイン内で DSIEX16 が呼び出されることがあります。

注: 620 ページの『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』との相違点は、この時点までにアクションが何も指定されていない場合、CNMCSSIR がメッセージを廃棄する点です。この場合、メッセージは表示されず、その Canzlog エントリーは適切であるとしてマークを付けられません。

自動化テーブルが経路指定するメッセージについては、622 ページの『15. OST または NNT メッセージ・キューの処理』も参照してください。

ROUTE キーワードを指定しなくても、自動化テーブル項目を指定することができます。この場合、CNMCSSIR タスクは Canzlog エントリーを適切であるとしてマークを付けますが、メッセージは表示されません。コマンドが指定されていると、そのコマンドは 1 次自動タスクに経路指定されます。

自動タスクに CNMCSSIR を開始させると、自動タスクが CNMCSSIR の状況をモニターする際に便利です。これにより、自動化処理のデフォルトの宛先も指定されます (自動化ステートメントに ROUTE オプションが指定されていない場合)。

9. ドメイン間メッセージおよびコマンド (NNT から OST へ)

原因: NNT が、受信したすべてのメッセージを、OST-NNT セッションを開始させた (START DOMAIN コマンドを使用) OST へ送信する。コマンドは、NNT からその関連 OST へ HDRTYPEX バッファの DSIPSS TYPE=OUTPUT により送信されることがあります。詳しくは、「*IBM Tivoli NetView for z/OS プログラミング: アセンブラー*」の DSIPSS マクロを参照してください。

元タスク: NNT

元タスクの NNT では、EMCS コンソールに渡される情報は、OST へは送信されません。

処理の流れ:

OST を受信する場合、

1. 受け取られたバッファがコマンドであり、HDRMTYPE が HDRTYPEI で、IFRCODE (AIFR) が IFRCODAI である場合、また、IFRAUTBA が示すバッファの HDRMTYPE が HDRTYPEX である場合、ビット IFRAUPHI および IFRAUPLO を使用して優先順位を設定します。その優先順位によって、受け取られたコマンドは、再び OST メッセージのキューに入れられます。

IFRAUPHI も IFRAUPLO もオンでないときには、HIGH (TYPE=IMMED および TYPE=BOTH コマンドの場合) または LOW (その他のコマンドの場合) のデフォルト値が使用されます。IFRAUTBA フィールドの値が IFRAUCMB フィー

ルドに移され、IFRAUCMD ビット (IFRAUCMB が指すコマンドを示す) がオンになります。受け取られたバッファーにあるすべての自動化フラグ (BEEP、DISPLAY、および HOLD 以外) が再設定されます。

2. バッファーがメッセージの場合 (つまり、IFRAUTBA が HDRTYPEEX バッファー以外の場合) には、HIGH に優先順位が設定されます。
3. あらかじめ設定された優先順位に応じて、AIFR は OST メッセージのキューに入れます。ただし、ドメイン間ログオンに関連した TYPE=IMMED メッセージは、この優先順位の例外になります。

これらのメッセージは、キューには入れられません。これらのメッセージを受け取ると、DSIPSS マクロを介してただちに表示します。

これ以上の処理については、622 ページの『15. OST または NNT メッセージ・キューの処理』を参照してください。

10. MVS または TAF OPCTL オペレーターとしての PPT

原因:

- PPT が MVS コマンドを出した。
- BGNSSESS OPCTL コマンドを PPT で実行した。

元タスク: PPT

処理の流れ: すべての送信請求および非送信請求のメッセージが、PPT のメッセージ・キューの MVS、または TAF OPCTL セッションから受信されます。620 ページの『13. PPT メッセージ・キュー処理』を参照してください。

11. MVS または TAF OPCTL オペレーターとしての OST または NNT

原因:

- OST/NNT が MVS コマンドを出した。
- BGNSSESS OPCTL コマンドを OST/NNT で実行した。

元タスク: OST または NNT

処理の流れ: すべての送信請求および非送信請求のメッセージが、OST または NNT のメッセージ・キューの MVS、または TAF OPCTL セッションから受信されます。622 ページの『15. OST または NNT メッセージ・キューの処理』を参照してください。

12. 送信請求 VTAM (POI) メッセージ

原因: VTAM コマンドの CMDDEF 定義がモジュール DSIVTP を指定する。

元タスク: OST、NNT、または PPT

処理の流れ:

1. VTAM MSGMOD オプションがアクティブの場合

- メッセージは、MSGMOD とともにログに記録されます (診断用)。DSIEX04 は、ログ処理時に呼び出されます。戻りコードをレジスター 15 に書き込むと、DSIEX04 はメッセージがハードコピー、ネットワーク、またはシステム・ログへ送信するかどうか判別します。
 - MSGMOD ID が削除され、メッセージを自動化で処理できるようにします。
 - MSGMOD がアクティブでない場合と同様に処理が継続されます。
2. メッセージがネットワーク状況を更新するために状況モニターで使用される場合、そのメッセージは状況モニターにより処理されます。
 3. DSIEX06 が呼び出されます。メッセージを削除すると、これ以上の処理は行われません。
 4. DSIPSS TYPE=OUTPUT を出してメッセージを送信します。

『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』および 624 ページの『17. OST または NNT DSIPSS』を参照してください。

13. PPT メッセージ・キュー処理

原因:

- MSG コマンドから出されるメッセージ
- 自動化テーブルが送信するメッセージおよびコマンド
- DSIMQS マクロにより出される汎用タスク間メッセージ
- 端末アクセス機能 (TAF) オペレーター制御

元タスク: 任意

処理の流れ:

1. HDRMTYPE=HDRTYPEI または HDRMTYPE=HDRTYPEP (メッセージが内部機能要求) の場合、要求された機能が実行されます。
 - HDRMTYPE=HDRTYPEI および IFRCODE=IFRCODUS (ユーザー内部機能要求) の場合、PPT はメッセージ・バッファーを処理するために DSIEX13 を呼び出し、制御権が戻された時点でバッファーを解放します (DSIFRE を使用)。
 - それ以外の HDRMTYPE=HDRTYPEP または HDRMTYPE=HDRTYPEI および IFRCODE=IFRCODCR の場合、PPT はコマンド入力処理のために DSIEX03 を呼び出します。DSIEX03 がコマンドを削除すると、コマンドの処理は終了します。削除しないときには、コマンド・プロセッサが実行します。
2. それ以外の場合には、PPT がメッセージを処理します。
 - メッセージは、DSIPSS TYPE=OUTPUT により処理されます。『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』を参照してください。

14. PPT または NetView 許可レシーバー・メッセージの DSIPSS

原因:

- PPT で出された DSIPSS

- PPT に送信されるメッセージ
- 許可レシーバーへ送られる DSIMQS
- 非送信請求 VTAM (POI) メッセージ
- BGNSESS コマンドを使用して PPT が開始する、TAF OPCTL セッションから受信されるメッセージ

元タスク: 任意

処理の流れ:

1. メッセージに ASSIGN PRI が指定されていると、メッセージは ASSIGN コマンドにより指定されているオペレーターへ送信されます。

622 ページの『15. OST または NNT メッセージ・キューの処理』および 623 ページの『16. NetView コンソール出力または SYSOP メッセージ・キューの処理』を参照してください。

このステップでは、次のタイプのメッセージを割り当てることはできません。

- ASSIGN コマンドを使用して以前に経路指定されたメッセージ
- このシステムの NetView アドレス・スペースからの WTO

ASSIGN SEC が指定されたメッセージは、この NetView ドメインの自動化テーブルによって処理されることはありません。別の NetView ドメインへ送信されると、これらのメッセージは DSIEX02A、自動化テーブル、および DSIEX16 の処理の対象になります。

2. それ以外の場合、メッセージは、以下の許可レシーバー (アクティブの場合) のいずれかに送信されます。

- POS 端末にログオンしているオペレーター

複数の POS 端末が定義されている場合は、最初に定義された端末に最高優先順位が割り当てられます。

- POS として定義されていないオペレーター

複数のこのようなオペレーターが定義されている場合は、最初に定義されたオペレーターに最高優先順位が割り当てられます。

- ドメイン間オペレーター

複数のドメイン間オペレーターが定義されている場合は、最初に定義されたオペレーターに最高優先順位が割り当てられます。

- 自動タスク・オペレーター

複数の自動タスクが定義されている場合は、最初に定義されたタスクに最高優先順位が割り当てられます。自動タスクを非送信請求メッセージの受信側にするときには、ASSIGN コマンドを使用してください。

3. 上記の 1 にも 2 にも当てはまらない場合。

- a. DSIEX02A が呼び出されます。出口でメッセージを削除すると、これ以上の処理は行われません。
- b. 自動化テーブルがチェックされます。テーブルでコマンド・アクションおよび表示アクションを選択することがあります。

- c. テーブルがメッセージを削除しても、この時点までには DSIEX16 が呼び出されます。
- d. DSIEX02、自動化テーブル、および DSIEX16 は、NetView ドメイン内の固有なメッセージごとに一度だけ呼び出されます。

呼び出されたあと、ASSIGN コマンドまたは自動化テーブルによりメッセージのコピーが作成されても、この NetView ドメインでは DSIEX02、自動化テーブル、または DSIEX16 は呼び出されません。メッセージが別の NetView ドメインへ送信されると、そのドメイン内に呼び出しが発生することもあります。

- e. ログ記録を抑止しなければ、PPT はメッセージをログに記録します。

ネットワーク状況を更新するために状況モニターが使用するメッセージは、状況モニターにより処理されます。

- f. 他のアクションが指示されていなかったり、DSIEX16 によりアクションが指示された場合、メッセージはシステム・コンソール (SYSOP) へ送信されます。

NetView のサブシステム・インターフェースから生成されるメッセージは、システム・コンソールへは書き込まれません。自動化テーブルが経路指定するメッセージについては、『15. OST または NNT メッセージ・キューの処理』も参照してください。

15. OST または NNT メッセージ・キューの処理

原因:

- MSG コマンドから出されるメッセージ
- 許可レシーバーが経路指定するメッセージ
- ASSIGN PRI、SEC、COPY が指定されたメッセージ
- 自動化テーブルが送信するメッセージおよびコマンド
- DSIMQS マクロにより出される汎用タスク間メッセージ
- このオペレーターが MVS オペレーターの場合は、MVS メッセージ
- 端末アクセス機能 (TAF) オペレーター制御 (コンソール) から送られるメッセージ

元タスク: 任意

処理の流れ:

1. HDRMTYPE=HDRTYPEI または HDRMTYPE=HDRTYPET (メッセージが内部機能要求) の場合、要求された機能が実行されます。
 - HDRMTYPE=HDRTYPEI および IFRCODE=IFRCODUS (ユーザー内部機能要求) の場合、DSIEX13 がメッセージ・バッファーを処理するために呼び出されます。
 - HDRMTYPE=HDRTYPET または HDRMTYPE=HDRTYPEI および IFRCODE=IFRCODCR の場合には、OST/NNT がコマンド入力进行处理するため

に DSIEX03 を呼び出します。DSIEX03 がコマンドを削除すると、コマンドの処理は終了します。削除しないときには、コマンド・プロセッサが実行します。

2. 上記以外の場合、OST/NNT は次のいずれかの操作を実行します。

- HDRMTYPE=HDRTYPEM の場合、OST/NNT はメッセージ・バッファーを処理するために DSIEX13 を呼び出し、制御権が戻された時点でバッファーを解放します (DSIFRE を使用)。
- HDRMTYPE=HDRTYPEM 以外の場合、OST/NNT は DSIPSS TYPE=OUTPUT によりメッセージを処理します。624 ページの『17. OST または NNT DSIPSS』を参照してください。

16. NetView コンソール出力または SYSOP メッセージ・キューの処理

原因:

- NetView が DSIWCS を出して、メッセージをシステム・コンソールに送信する。
- 他に宛先が指定されていないかまたは使用できない場合、NetView 許可レシーバーがメッセージをシステム・コンソールに経路指定する。
- 他に宛先が指定されていないかまたは使用できない場合、NetView PPT がメッセージをシステム・コンソールに経路指定する。
- メッセージが SYSOP のキューに入れられる。
- ASSIGN PRI、SEC、COPY が指定されたメッセージが SYSOP へ経路指定されます。

NetView によって処理される、サブシステム・インターフェースから生成されるメッセージは、システム・コンソールに再び表示されます。記述コード 13 は、これらのメッセージが潜在的なループ状態を起こさないようにオンにされます。

元タスク: 任意

処理の流れ:

1. DSIWCS がバッファーをログに記録したあと、次の操作を両方とも実行します。
 - DSIEX02A が呼び出されなかった場合には、DSIWLS マクロ処理が DSIEX04 を呼び出します。戻りコードをレジスター 4 に書き込むと、DSIEX04 はメッセージをハードコピー・ログ、CanzLog ログ、ネットワーク・ログ、またはシステム・ログへ送信するかどうか判別します。
 - DEFAULTS および OVERRIDE コマンドに応じて、DSIWLS がログにメッセージを書き込みます。

注: この NetView プログラムのサブシステム・インターフェースから生成されるメッセージは、システム・ログへは書き込まれません。

2. DSIEX02A が呼び出されなかった場合には、DSIWCS マクロ処理が DSIEX09 を呼び出します。DSIEX09 がメッセージを削除すると、メッセージはコンソールへは書き込まれません。
3. DSIWCS がシステム・コンソールへメッセージを書き込みます。

注: ASSIGN PRI=SYSOP が指定されていると、DSIPSS 処理をバイパスするため、これらのメッセージは自動化の対象にはなりません。

17. OST または NNT DSIPSS

原因: DSIPSS

元タスク: OST、NNT

処理の流れ:

1. DSIEX02A が呼び出されます。この出口でメッセージを削除すると、処理はここで終了します。
2. &WAIT および WAIT 検索が呼び出されます。

メッセージを &WAIT および WAIT 処理で抑止する場合、表示またはログ記録の対象にならないように強制フラグが立てられます。しかし、出口ルーチン DSIEX16 がこれらのメッセージを操作できるように、処理は継続されます。

3. 自動化テーブル処理が開始されます。

テーブルのアクションは、DSIEX16 へ渡されるバッファ構造に示されています。

4. DSIEX16 が呼び出されます。
5. DSIEX16、DSIEX02A、および自動化テーブルは、NetView ドメイン内の固有なメッセージごとに一度のみ呼び出されます。

呼び出されたあと、ASSIGN コマンドまたは自動化テーブルによりメッセージのコピーが作成されても、この NetView ドメインでは DSIEX16 は呼び出されません。メッセージが別の NetView ドメインへ送信されると、そのドメイン内で DSIEX16 が呼び出されることもあります。

6. ログ記録、表示、経路指定、およびコマンド・アクションは、自動化内部機能要求 (AIFR) バッファと、現行の DEFAULTS および OVERRIDE コマンドの設定にしたがって処理されます。

バッファ構造により、取るべきアクションが決定されます。

自動化テーブルが経路指定するメッセージについては、622 ページの『15. OST または NNT メッセージ・キューの処理』も参照してください。

7. メッセージが表示されると、ASSIGN COPY サービスが実行されます。

ASSIGN COPY が指定されたメッセージは、この NetView ドメインの自動化テーブルにより処理されることはありません。別の NetView ドメインへ送信されると、これらのメッセージは DSIEX02A、自動化テーブル、および DSIEX16 の処理の対象になります。

メッセージは、OST/NNT メッセージのキューに入れられます。622 ページの『15. OST または NNT メッセージ・キューの処理』および 623 ページの『16. NetView コンソール出力または SYSOP メッセージ・キューの処理』を参照してください。

8. OST の場合、メッセージは、NetView 端末 (VTAM) か、あるいはシステム・コンソール (AUTOTASK OPID=name) に表示されます。NetView のサブシステム・インターフェースから生成されるメッセージは、システム・コンソールへは書き込まれません。
9. NNT の場合、メッセージは OST ヘドメイン間送信されます。618 ページの『9. ドメイン間メッセージおよびコマンド (NNT から OST へ)』を参照してください。次のドメインでメッセージ (ASSIGN SEC および ASSIGN COPY が指定されたメッセージの場合でも) が受信されると、DSIEX02A、自動化テーブル、および DSIEX16 処理が許可されます。

18 OST、NNT、または自動タスクの送信請求および非送信請求システム MVS 拡張コンソール・メッセージ

原因:NetView プログラムが、MVS/ESA システム・メッセージ用に EMCS コンソールを使用するように構成されています。OST、NNT、または自動タスクは、EMCS コンソール・メッセージを受信するように構成されています。

元タスク: OST、NNT、または自動タスク

処理の流れ:

1. DSIEX17 が、メッセージと DOM の両方のために呼び出されます。メッセージまたは DOM を削除すると、これ以上の処理は行われません。
2. DSIPSS が、メッセージまたは DOM のために出されます。624 ページの『17. OST または NNT DSIPSS』を参照してください。

NetView タスク (CNMCSSIR タスクを除く) によって使用されている EMCS コンソールで受信される MVS システム・メッセージは、NetView プログラムによって送信請求メッセージと見なされます。したがって、これらのメッセージは ASSIGN コマンドの PRI および SEC 処理の対象にはなりません。

MVS/ESA 第 4 版 リリース 3 およびそれ以降のリリースは、コマンド応答の抑止をサポートします。MVS コマンドが NetView コンソールから出され、さらにコマンドの応答に AUTO(YES) と SUP(YES) が指定されている場合、そのメッセージは CNMCSSIR タスクのもとで自動化されます。そのメッセージは、非送信請求の MVS システム・メッセージとして処理されます。

19 PPT の送信請求および非送信請求システム MVS 拡張コンソール・メッセージ

原因:NetView プログラムが、MVS/ESA システム・メッセージ用に EMCS コンソールを使用するように構成されています。PPT は、EMCS コンソール・メッセージを受信するように構成されています。

元タスク: PPT

処理の流れ:

1. DSIEX17 が、メッセージと DOM の両方のために呼び出されます。
メッセージまたは DOM を削除すると、これ以上の処理は行われません。
2. DSIPSS が、メッセージまたは DOM のために出されます。

620 ページの『14. PPT または NetView 許可レシーバー・メッセージの DSIPSS』を参照してください。

PPT によって使用中の EMCS コンソールで受信される MVS システム・メッセージは、NetView によって送信請求と見なされます。しかし、送信請求メッセージを受信する他の NetView タスクと違って、PPT は、ASSIGN コマンド PRI および SEC 処理を可能にします。

付録 G. NetView メッセージ・タイプ (HDRMTYPE) の説明

この付録では、NetView プログラムにより提供される診断、修正、およびチューニング情報について説明します。

重要: この診断、修正、チューニング情報を、プログラム・インターフェースとして使用しないでください。

この付録では、NetView メッセージ・タイプをアルファベット順にリストします。メッセージ・タイプは、コマンド、メッセージ、および MSU に適用されます。自動化テーブルのメッセージ・タイプを調べる際には、HDRMTYPE キーワードを使用してください。メッセージ・タイプは、BUFHDR 制御ブロックの HDRMTYPE フィールドに保管されています。

HDRTYPAC (A)

MVS V1R2 およびそれ以降のリリースの NetView プログラムでは使用されていません。このメッセージ・タイプは、自動化 IFR (HDRMTYPE=HDRTYPEI, IFRCODE=IFRCODAI) に置き換えられています。NetView プログラムの古いリリースとのドメイン間セッション時に、このメッセージ・タイプを受信することがあります。

HDRTYPDT (D)

非メッセージ・データ・タイプを示しています。

HDRTYPEA (T)

MVS V1R2 およびそれ以降のリリースの NetView プログラムでは使用されていません。ネットワーク通信管理機能 (NCCF) の TCAM からの送信請求メッセージを示しています。TCAM NCCF とのドメイン・セッションでこのメッセージ・タイプを受信する可能性があります。

HDRTYPEB (?)

表示およびログ記録が抑止されているコマンドまたはコマンド・リスト・バッファーを示しています。CNMSTYLE メンバーに定義されている抑止文字を指定して入力されたコマンドの、表示およびログ記録の抑止に使用されます。CNMSTYLE メンバーの詳細については、「*IBM Tivoli NetView for z/OS* インストール:概説」に記載されています。HDRTYPEB は、同じ抑止文字が先頭に指定されているコマンド・リスト・ステートメントの表示およびログ記録の防止にも使用できます。

HDRTYPEC (C)

コマンド・リストからのコマンドまたはメッセージを示します。抑止されたコマンド・リスト・ステートメントの場合には、HDRTYPEB になります。

HDRTYPED (!)

即時コマンド・プロセッサからのメッセージを示します。通常は、DSIPSS TYPE=IMMED を使用して画面に送信されます。このタイプのメッセージが画面の即時メッセージ域に表示されるときには、HDRMTYPE および DOMAIN 名は表示されません。ドメイン間モードで受信されると、このタイプのメッセージ (ドメイン・ネームとタイプ接頭部を含む) は、通常の

出力域に表示されます。DSIPSS TYPE=IMMED によって、HDRTYPED が強制されたり、設定されたりすることはありません。

HDRTYPEE (E)

オペレーティング・システムからのメッセージを示します。このタイプは、タイトル行モードの複数行オペレーター宛メッセージ (MLWTO)、システム・アクション、および WTOR メッセージには使用されません。オペレーティング・システム・メッセージの他の形式については、HDRTYPEK および HDRTYPEY も参照してください。

HDRTYPEF (F)

VSAM レコードを示しています。オペレーターの画面には表示されません。データ・サービス・タスク (DST) 中で使用されます。

HDRTYPEG (G)

CNMI レコードを示しています。オペレーターの画面には表示されません。DST 中で使用されます。

HDRTYPEI (I)

内部機能要求を示しています。このバッファは、タスク内およびタスク間の定様式インターフェースです。IFR には、バッファの形式と機能を決定する機能番号 (IFRCODE) が含まれます。詳細については、「*IBM Tivoli NetView for z/OS プログラミング: アセンブラー*」を参照してください。

HDRTYPEJ (J)

NetView プログラム自身から生成されるタイトル行モード複数行オペレーター宛 (MLWTO) メッセージを示しています。これらのバッファは、順序どおりでなければならず、制御の説明、ラベル、データおよび終了指定機能を含んでいなければなりません。MVS V1R2 およびそれ以降のリリースの NetView プログラムでは、これらの一連のバッファを表示および自動化用の 1 つのメッセージとして扱います。

HDRTYPEK (K)

HDRTYPEJ と同じ意味ですが、NetView プログラムが提供しない IBM ルーチンで生成されるメッセージ用です。

HDRTYPEL (L)

HDRTYPEJ と同じ意味ですが、Tivoli 以外のルーチンで生成されるメッセージ用です。

HDRTYPEM (M)

NetView メッセージ・コマンド・プロセッサからのメッセージを示しています。

HDRTYPEN (N)

NetView からの通常の単一バッファ・メッセージを示しています。

HDRTYPEP (P)

PPI からのメッセージを示しています。

HDRTYPEQ (Q)

単一バッファの非送信請求メッセージである VTAM POI からのメッセージを示しています。他の VTAM POI メッセージについては、HDRTYPEV、HDRTYPEY、および HDRTYPEK も参照してください。この

メッセージ・タイプは、オペレーティング・システム・インターフェースで受信される VTAM からのメッセージには設定されません。

HDRTYPER (R)

オペレーターが、NetView WTOR 番号 DSI802A に応答して VTAM REPLY コマンドを入力したことを示しています。このメッセージ・タイプはログには記録されますが、NetView コンソールには表示されません。

HDRTYPES (S)

バッファが交換されたことを示すために、特定のインストール・システム 出口インターフェースで使用されます。

HDRTYPET (*)

NetView 端末から NetView に出されたコマンドを示します。このメッセージ・タイプは、そのバッファがメッセージではなくコマンドであることを示しています。

IFRCODE=IFRCODCR とともに HDRMTYPE=HDRTYPEI が指定されると、同様にバッファは処理するコマンドを示します。ただし、IFRCODCR は、通常、オペレーター端末タスク (OST) と DST タスク間などの内部定 様式コマンドを意味します。HDRTYPET は、通常、オペレーターがコマ ンドを入力した場合と同様にコマンド・バッファを意味します。 IFRCODCR バッファには、印刷不能データを入れることができます。 HDRTYPET バッファには、印刷不能テキストを入れることはできません。

HDRTYPEU (U)

Tivoli 以外のユーザー用に予約済みです。アクション・メッセージ、WTOR またはタイトル行 (MLWTO) メッセージには使用することができません。

HDRTYPEV (HEX('40'))

単一バッファ送信請求メッセージである VTAM POI からのメッセージを示しています。他の VTAM POI メッセージについては、 HDRTYPEQ、HDRTYPEY、および HDRTYPEK も参照してください。この メッセージ・タイプは、オペレーティング・システム・インターフェースで 受信される VTAM からのメッセージには設定されません。

注: このメッセージ・タイプは、値 X'40' (スペース文字) です。

HDRTYPEW (+)

NetView メッセージではない、Tivoli 単一行メッセージを示しています。こ のメッセージ・タイプは、HDRTYPEN および HDRTYPEU と類似していま す。

HDRTYPEX (X)

ドメイン間 (NNT から OST へ) コマンドを示しています。逆方向のコマ ンドを送信できるようにします。通常、コマンドは ROUTE コマンドなど に より OST から NNT へ経路指定されます。

NNT で実行するプログラムは HDRTYPEX バッファへ DSIPSS TYPE=OUTPUT を出すことができますが、対応するコマンドはその NNT でセッションを開始した OST で処理されます。これは、フルスクリーンま

たはそれ以外の形式設定で、不定様式 (16 進) データを NNT から OST へ送信する際に便利です。データは最大 256 バイトであり、オペレーターの画面には表示されません。

HDRTYPEY (>)

単一バッファ・アクションまたは WTOR を示しています。MVS V1R2 およびそれ以降のリリースの NetView プログラムでは、オペレーティング・システム・インターフェースおよび VTAM POI からのメッセージを示す場合もあります。MVS V1R2 およびそれ以降のリリースの NetView プログラムでは、アクションが取られるまで、または応答が入力されるまで、これらのメッセージが NetView コマンド機能の画面に表示されたままになります。オペレーターは、より大 (>) 文字を重ねて入力し ENTER を押して、これらのメッセージを削除することができます。メッセージは、次に画面上でテキストが折り返されると消えます。インストール・システム出口では、メッセージを強制的に保留するためにこのメッセージ・タイプを設定することができます。

HDRTYPEY フラグは設定されているが、IFRAUWQE フラグが設定されていないとき、NetView プログラムは、メッセージ・テキスト内のメッセージ番号の直前にある 3 桁の応答 ID を探します。応答 ID が存在すれば、メッセージは VTAM WTOR です。応答 ID が存在しなければ、メッセージは保留メッセージ (IFRAUWQE がゼロの場合) として扱われます。

IFRAUWQE が 1 に設定されている場合、作業キュー・エレメント (WQE) データが WTOR またはアクション・メッセージのどちらを示しているのかを調べるために IFRAUWQD データがチェックされます。WTOR を示している場合には、メッセージ ID の直前に応答 ID (2 から 4 文字から成る) が存在します。応答 ID が存在するときには、メッセージ ID と応答 ID はスペース 1 つで区切られています。

HDRTYPEZ (Z)

HDRTYPEN と類似していますが、特にデータ・サービス・タスク (DST) からのメッセージを示しています。

HDRTYPE\$ (\$)

印刷不可のデータを含むメッセージ示しています。

HDRTYPE1 (V)

HDRTYPEV と類似していますが、PPOLOG メッセージを示しています。

HDRTYPE2 (Y)

HDRTYPEY と類似していますが、PPOLOG メッセージを示しています。

HDRTYFEB (HEX'FB')

バッファをフラッシュするメッセージを示しています。

HDRTYPLS (s)

ユーザー置換コマンドを示しています。

HDRTYPLT (L)

メッセージ・バッファではなく、トレース・レコードを示しています。このメッセージ・タイプは NetView 内部トレース専用であり、メッセージ・バッファ内で使用されてはならないものです。

HDRTYPOR (HEX '4F')

パイプライン生成のメッセージを示しています。

HDRTYPQC (HEX'50')

二重抑止コマンドで指定されたコマンドを示しています。オペレーター・コンソールでのコマンド・エコーとコマンドの実行により生成される同期メッセージ出力の両方が抑止されます。非同期メッセージは抑止されません。

HDRTYPWT (W)

WAIT 条件に一致したため表示されたメッセージを示しています。画面上およびログ内のメッセージ・タイプ・フィールドには W が表示されますが、バッファ内の HDRMTYPE フィールドには表示されません。バッファ内の HDRMTYPE フィールドには、元のメッセージ・タイプが入れられます。

HDRTYP10 (HEX'10')

管理サービス単位 (MSU) バッファを示しています。MSU バッファには、関連 MSU HIER バッファが含まれていることがあります。

HDRTYP11 (HEX'0B')

リモート・データ転送メッセージを示しています。

付録 H. MVS コマンド管理 (非推奨)

MVS コマンド管理機能は非推奨となり、MVS コマンド改訂機能に置き換えられました。詳しくは、149 ページの『第 14 章 コマンド改訂テーブル』を参照してください。MVS コマンド管理機能は、マイグレーション目的でのみサポートされています。MVS コマンド改訂機能へのマイグレーションについては、「*IBM Tivoli NetView for z/OS*インストール:マイグレーション・ガイド」を参照してください。

MVS コマンド管理を使用すると、ほとんどの MVS コマンドを検査、変更、またはリジェクトできます。コマンドごと、またはコンソール名ごとに、コマンドを個別に処理に組み込んだり、除外することができます。

MVS コマンド管理が活動化された後で、すべての MVS コマンドが NetView MVS コマンド出口に渡されます。大半の MVS コマンドは、それらが組み込まれていないか、特に除外されていない限り、処理のために NetView プログラムに送られます。NetView プログラムでは、DSIMCAOP 自動タスクでの MVS コマンドによって、CNMEMCXY REXX コマンド・リストが呼び出されます。このコマンド・リストに論理を追加して、MVS コマンドを検査、変更、またはリジェクトできます。MVS コマンドは、リジェクトされない場合、MVS に戻されて処理されます。コマンドが NetView MVS コマンド出口によって処理されると、RACF 検査が実行されます。

634 ページの図 197 に、MVS コマンド管理の論理フローを示します。このコマンド管理を使用可能にするには、MVS 環境および NetView 環境に対する変更が必要です。

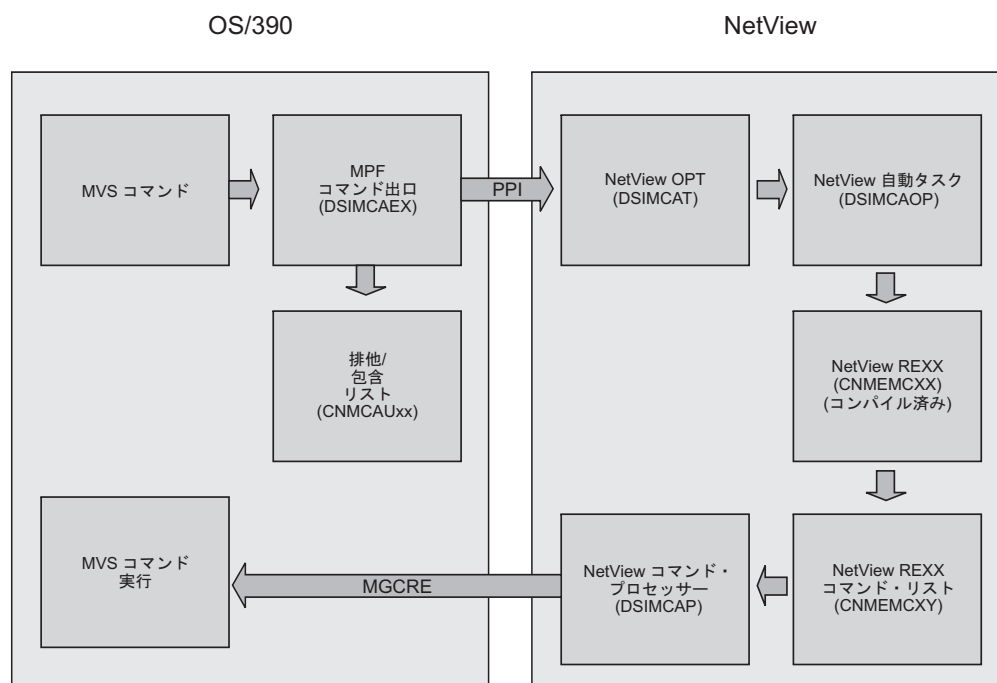


図 197. MVS コマンド管理の流れ

NetView 環境での MVS コマンド管理の使用可能化

MVS コマンド管理を NetView 環境で使用可能にするには、以下の手順を実行します。

1. 新規の NetView オペレーター DSIMCAOP を DSIOPF または SAF プロダクト内で定義します。DSIMCAOP 以外のオペレーター名を使用する場合は、CNMSTUSR または CxxSTGEN メンバーの以下のステートメントを使用して、使用するオペレーター ID を追加します。CNMSTYLE ステートメントの変更については、「IBM Tivoli NetView for z/OSインストール:概説」を参照してください。

```
function.autotask.mvsCmdMgt=operid
```

このステートメントが CNMSTYLE メンバーに含まれていない場合は、DSIMCAOP がデフォルトのオペレーター ID になります。

2. DSIMCAP、CNMEMCXX、および CNMEMCXY を無許可での使用から保護します。

注: オペレーター定義およびコマンド許可を目的として、RACFなどの SAF プロダクトを使用している場合は、これらの定義に対して同等の更新を行ってください。

3. CNMSTYLE メンバーのタワー・ステートメントが MVScmdMgt タワーの前にアスタリスク (*) を指定していないことを確認します。

MVS 上の MVS コマンド出口の使用可能化

MVS コマンド出口では、NetView プログラム間インターフェース (PPI) が使用されます。出口を使用可能にする前に、必ず NetView サブシステムのアドレス・スペース・プログラム (SSI) を開始してください。

MVS での処理のために、MVS コマンド出口を使用可能にするには、以下の手順を実行します。

1. ロード・モジュール DSIMCAEX が、MVS LINKLST 連結のロード・ライブラリー内にあることを確認します。必要に応じて以下のコマンドを発行し、それを使用可能にします。

F LLA,REFRESH

2. 以下のステートメントを追加して、PARMLIB 内の MPFLSTxx メンバーを更新します。

```
.CMD USEREXIT(DSIMCAEX)
```

変更を活動化するには、以下のコマンドを発行します。

SET MPF=xx

この場合、xx は MPFLST メンバーの接尾部です。

3. コマンドの包含/除外リストが提供されていない限り、ほとんどのコマンドは NetView プログラムに送られます。NetView プログラムに送られないようにコマンドを制限するには、コマンドの包含/除外リストを使用します。NetView プログラムでは、サンプル・リスト CNMCAU00 が提供されています。このサンプルを使用するか、独自のリストを作成して、論理 PARMLIB 内に置くことができます。

変更を活動化するには、以下のコマンドを発行します。

SET CNMCAUT=yy

ここで、yy は PARMLIB の CNMCAU メンバーの接尾部です。この場合も、包含/除外リストが通常モードで使用可能になります。包含/除外リストを使用しない場合は、yy に **ON** という値を指定します。

その後、以下のコマンドを発行すると、包含/除外リストをテスト・モードに設定できます。

```
SET CNMCAUT=TEST
```

テストが正常に実行されたら、以下のいずれかのコマンドを発行して、テスト・モードをリセットします。

SET CNMCAUT=yy

SET CNMCAUT=ON

4. テスト後、MPFLSTxx メンバーに項目を追加すると、コマンドが変更されるたびに発行されるメッセージ IEE295I を抑止できます。そうしなかった場合は、各コマンドが出口によって処理されるたびに、以下のメッセージが表示されます。

```
IEE295I COMMAND CHANGED BY EXIT 043
ORIGINAL: command ' '
MODIFIED: command
```

追加のコマンド・エコーおよび IEE295I メッセージの抑止

以下の条件がすべて整うと、システム・ログにコマンドのエコー出力が複数回記録されます。

- NetView MVS コマンド管理機能がアクティブになっている
- コマンドが処理用に NetView プログラムに送信されます。
- コマンドが CNMEMCXY によって実行を拒否されない

システム・ログに IEE295I メッセージが記録されます。論理 PARMLIB メンバー CNMCAUaa 内で、TRACKING.ECHO ステートメントを指定すると、追加のコマンド・エコーを抑止でき、ISSUE.IEE295I ステートメントを指定すると、IEE295I メッセージを抑止できます。

TRACKING.ECHO ステートメントの構文は、以下のとおりです。

TRACKING.ECHO = Y | N

TRACKING ECHO コマンドのデフォルトは、以下のとおりです。

TRACKING.ECHO = Y

ISSUE.IEE295I ステートメントの構文は、以下のとおりです。

ISSUE.IEE295I = Y | N

ISSUE.IEE295I のデフォルトは、以下のとおりです。

ISSUE.IEE295I = Y

これらのステートメントは、論理 PARMLIB メンバー CNMCAUaa 内の任意の場所にコーディングできます。

- TRACKING.ECHO = Y という指定は、追加のコマンド・エコーがシステム・ログに記録されることを示しています。
- TRACKING.ECHO = N という指定は、追加のコマンド・エコーがシステム・ログに記録されないことを示しています。TRACKING.ECHO = N をコーディングした場合、CNMEMCXY によるコマンドの変更はシステム・ログにも NetView ログにも示されないので注意してください。コマンド応答は、当初入力したコマンドとはまったく異なるコマンドに対するものである場合があります。CNMEMCXY によってコマンドが変更された時刻を知るには、TRACKING.ECHO = Y を指定するか、CNMEMCXY にコードを追加して、コマンドが変更された前後にコマンドをログに記録します。
- ISSUE.IEE295I = Y という指定は、IEE295I メッセージがシステム・ログに記録されることを示しています。
- TRACKING.ECHO = N という指定は、IEE295I メッセージがシステム・ログに記録されないことを示しています。

CNMCAUaa メンバー内で複数の TRACKING.ECHO ステートメントまたは ISSUE.IEE295I ステートメントを指定すると、最後の有効な値が使用されます。ステートメントをコーディングしなかった場合、またはステートメントが無効である場合は、デフォルトが使用されます。

例: MVS コマンド D T を入力したとします。さらに、コマンドは除外されず、TRACKING.ECHO と ISSUE.IEE295I のいずれもコーディングしていない (または、こ

これらのステートメントを Y という値で明確にコーディングしていた) とします。これらのステートメントは、システム・ログに以下のように記録されます。

```
1. D T
2. D T ' '
3. IEE295I COMMAND CHANGED BY EXIT 314
4. ORIGINAL: D T ' '
5. MODIFIED: D T
6. D T
7. IEE136I LOCAL: TIME=14.34.45 DATE=2009.103 UTC: TIME=18.34.45 DATE=2009.103
```

TRACKING.ECHO = N を指定すると、行 2 および 6 はログに記録されなくなります。ISSUE.IEE295I = N を指定すると、行 3、4 および 6 はログに記録されなくなります。

ステートメントが CNMCAUaa に

```
TRACKING.ECHO = N
ISSUE.IEE295I = Y
```

と指定されると、これらのステートメントはシステム・ログに以下のように記録されます。

```
1. D T
2. IEE295I COMMAND CHANGED BY EXIT 423
3. ORIGINAL: D T ' '
4. MODIFIED: D T
5. IEE136I LOCAL: TIME=14.39.44 DATE=2009.193 UTC: TIME = 18.34.45 DATE=2009.103
```

CNMCAUaa 内のステートメントが

```
TRACKING.ECHO = Y
ISSUE.IEE295I = N
```

とコーディングされると、これらのステートメントはシステム・ログに以下のように記録されます。

```
1. D T
2. IEE295I COMMAND CHANGED BY EXIT 423
3. ORIGINAL: D T ' '
4. MODIFIED: D T
5. IEE136I LOCAL: TIME=14.39.44 DATE=2009.193 UTC: TIME = 18.34.45 DATE=2009.103
```

CNMCAUaa 内のステートメントが

```
TRACKING.ECHO = N
ISSUE.IEE295I = N
```

とコーディングされると、これらのステートメントはシステム・ログに以下のように記録されます。

```
1. D T
2. IEE136I LOCAL: TIME=14.39.44 DATE=2009.193 UTC: TIME = 18.34.45 DATE=2009.103
```

以下のコマンドを実行すると、TRACKING.ECHO または ISSUE.IEE295I の値を変更できます。

```
SET CNMCAUT=aa
```

TRACKING.ECHO ステートメントおよび ISSUE.IEE295I ステートメントをコーディングしている場合、CNMCAUaa がロードされ、正常に処理されると、指定した値が有効になります。これらのステートメントをコーディングしていない場合は、デフォルト値が使用されます。CNMCAUaa がロードされず、正常に処理されなかつ

た場合は、これらのステートメントに指定した値は変更されません。これらのステートメントのデフォルト値は、以下のとおりです。

```
TRACKING.ECHO = Y  
ISSUE.IEE295I = Y
```

以下のコマンドを実行しても、TRACKING.ECHO および ISSUE.IEE295I の値は変更されません。

```
SET CNMCAUT=OFF  
SET CNMCAUT=TEST
```

排他または包含リスト

MVC コマンド管理を使用すると、以下を制御できます。

- CONSOLE EXCLUSION LIST または CONSOLE INCLUSION LIST (両方とも 639 ページの『コンソール排他リストおよびコンソール包含リスト』で説明) を指定することにより、特定のコンソールからの入力を組み込むか除外する。
- COMMAND EXCLUSION LIST または COMMAND INCLUSION LIST (両方とも 639 ページの『コマンド排他リストおよびコマンド包含リスト』で説明) を指定することにより、特定のコマンドを組み込むか除外する。
- CMDTEXT EXCLUSION LIST または CMDTEXT INCLUSION LIST (両方とも 641 ページの『CMDTEXT 排他リストおよび CMDTEXT 包含リスト』で説明) を指定することにより、標準形式に適合しない他の特定のタイプのコマンドを組み込むか除外する。

これらの各リストは、論理 PARMLIB メンバーである CNMCAU_{aa} 内で定義します (_{aa} は英数字にする必要があります)。論理 PARMLIB には複数の CNMCAU_{aa} メンバーを置くことができますが、アクティブにすることができるのは常に 1 つだけです。

論理 PARMLIB メンバー - CNMCAU_{aa}

コンソールおよびコマンドの排他リストおよび包含リストは、PARMLIB メンバー CNMCAU_{aa} に定義します。

以下のコマンドを自動化処理から除外します。

```
COMMAND EXCLUSION LIST  
The following are internally-issued DB2 commands:  
S DSNAMSTR  
S DSNAILRM  
S DSNADBM1  
S DSNADIST  
S DSNASPAS
```

```
The following are internally-issued MQ Series commands:  
S MQMIMSTR  
S MQM1CHIN
```

CNMCAU_{aa} ステートメントの構文

以下の構文規則は、CNMCAU_{aa} ステートメントに適用されます。

- 1 桁目にスラッシュ (/) があり、2 桁目にアスタリスク (*) がある場合は、コメントを示します。
- 73 から 80 桁目までは無視されます。
- 認識されるのは、コメント、コンソール排除リスト、コンソール包含リスト、コマンド排除リスト、コマンド包含リスト、コンソール名、およびコマンドのみです。
- コメント・ステートメント以外の場合、1 桁目はブランクまたはアスタリスク (*) にする必要があります。
- コンソール名とコマンド・テキストでは、ワイルドカード文字 (*) がサポートされます。
- 1 桁目のアスタリスク (*) は、コンソール名またはコマンド・テキストの先頭にワイルドカードの一致 (0 から n 文字の一致) があることを示すインディケータです。

コンソール排除リストおよびコンソール包含リスト

以下のルールは、コンソール排除リストまたはコンソール包含リストに適用されます。

- CONSOLE EXCLUSION LIST (または CONSOLE INCLUSION LIST) は、2 桁目から始める必要があります。

注: ワードとワードの間に余分なブランクを入れないでください。

- 除外するすべてのコンソール名を CONSOLE EXCLUSION LIST (または CONSOLE INCLUSION LIST) の行の後に入力してください。
- 1 桁目はワイルドカード文字 (*) 用に予約されます。

ワイルドカード文字 (*) を指定しない場合、1 桁目はブランクにする必要があります。

- 1 つの行にはコンソール名を 1 つだけ入力し、2 桁目から始めるようにしてください。
- 各コンソール名の長さは、8 文字 (ブランクを含める) にすることが前提です。
- ワイルドカード文字 (*) は、コンソール名の先頭か末尾に指定できますが、両方に指定することはできません。
 - 1 桁目にアスタリスク (*) を付けると、コンソール名の先頭でワイルドカードを突き合わせるようにすることを示します。
 - コンソール名の末尾にアスタリスク (*) を付けると、コンソール名の末尾でワイルドカードを突き合わせるようにすることを示します。
 - アスタリスク (*) がコンソール名に入力された唯一の文字である場合は、ワイルドカード文字ではなく、通常の文字として扱われます。

コマンド排除リストおよびコマンド包含リスト

以下のコマンド排除リスト規則が適用されます。

- COMMAND EXCLUSION LIST (または COMMAND INCLUSION LIST) は、2 桁目で始まります。

注: ワードとワードの間に余分なブランクを入れないでください。

- 除外するすべてのコマンドを COMMAND EXCLUSION LIST (または COMMAND INCLUSION LIST) の行の後に入力してください。
- 各コマンドの長さは、1 から 122 文字です。
- 1 桁目はワイルドカード文字 (*) 用に予約されます。

ワイルドカード文字 (*) を指定しない場合、1 桁目はブランクにする必要があります。

- コマンドは 2 桁目で始まり、71 桁目 (コマンドの先頭でワイルドカードを突き合わせない場合) まで実行できます。
- 72 桁目は継続表示桁です。

継続しない場合は、72 桁目をブランクにする必要があります。継続文字はストリングには含まれません。

- 73 から 80 桁目までは無視されます。
- コマンドに余分なブランクを入力することはできません。
- 継続行は、2 桁目から始める必要があります。
- 72 桁目がブランクの場合、末尾ブランクは削除されます。
- ワイルドカード文字 (*) は、コマンド・テキストの先頭か末尾に指定できますが、両方に指定することはできません。
 - 1 桁目にアスタリスク (*) を付けると、コマンド・テキストの先頭でワイルドカードを突き合わせるようにすることを示します。
 - コマンド・テキストの末尾にアスタリスク (*) を付けると、コマンド・テキストの末尾でワイルドカードを突き合わせるようにすることを示します。
 - アスタリスク (*) がコマンド・テキストに入力された唯一の文字である場合は、ワイルドカード文字ではなく、通常の文字として扱われます。

COMMAND EXCLUSION LIST の使用上の注意:

- コマンド排除リストを使用している場合は、内部で実行されたすべての START DB2 コマンドをこのリストに追加してください。

これらのコマンドがユーザーの環境でどのように表示されるかを見るには、START DB2 コマンドの実行直後に SYSLOG を参照してください。DB2 は、以後、内部で多くの START コマンドを実行して、その従属アドレス・スペースを開始します。コマンド除外リストに追加する必要があるのは、これらのコマンドです。

現時点で、DB2 サブシステムは、少なくとも 5 つの以下のような名前のアドレス・スペースから構成されています。

```
DSNAMSTR
DSNAIRLM
DSNADBM1
DSNADIST
DSNASPAS
```

- コマンド排除リストを使用している場合は、内部で実行されたすべての MQ START コマンドをこのリストに追加してください。これらのコマンドが自分の環境でどのように表示されるかを見るには、MQM1 START OMGF コマンドの実行直後に SYSLOG を参照してください。MQ は、以後、内部で多くの START コマンドを実行して、その従属アドレス・スペースを開始します。コマンド除外リストに追加する必要があるのは、これらのコマンドです。現時点で、MQ サブシステムは、少なくとも 2 つの以下のような名前のアドレス・スペースから構成されています。

```
MQM1MSTR
MQM1CHIN
```

- 追加情報については、645 ページの『コンソールとコマンドの包含および排除リストの一般的な処理』を参照してください。

COMMAND INCLUSION LIST の使用上の注意:

- コマンド包含リストを使用している場合は、内部で実行された DB2 START コマンドをこのリストに追加しないでください。
- コマンド包含リストを使用している場合は、内部で実行された MQ START コマンドをこのリストに追加しないでください。
- 追加情報については、645 ページの『コンソールとコマンドの包含および排除リストの一般的な処理』を参照してください。

CMDTEXT 排除リストおよび CMDTEXT 包含リスト

一部のコマンドは、639 ページの『コマンド排除リストおよびコマンド包含リスト』で説明したのと同じコマンド形式でない場合があります。例えば、一部の NetView コマンドでは、キーワードや値の区切り文字として、コンマの代わりにブランクを使用できます。前述のコマンド形式に合致しないこうしたコマンドでは、CMDTEXT EXCLUSION LIST および CMDTEXT INCLUSION LIST が同様の機能を提供します。CMDTEXT EXCLUSION LIST および CMDTEXT INCLUSION LIST には、以下のルールが適用されます。

- CMDTEXT EXCLUSION LIST および CMDTEXT INCLUSION LIST は、両方とも 2 桁目から始める必要があります。
- 除外対象のコマンド・ストリングはすべて、CMDTEXT EXCLUSION LIST 行の後に記述し、包含対象のコマンド・ストリングはすべて、CMDTEXT INCLUSION LIST 行の後に記述する必要があります。
- COMMAND EXCLUSION LIST および COMMAND INCLUSION LIST の下にリストされた、ワイルドカード文字および継続文字に関するルールはすべて、CMDTEXT EXCLUSION LIST および CMDTEXT INCLUSION LIST に適用されます。
- CMDTEXT EXCLUSION LIST および CMDTEXT INCLUSION LIST は、互いに排他的です。つまり、定義できるのはどちらか一方のみです。

COMMAND INCLUSION (または EXCLUSION) LIST と CMDTEXT INCLUSION (または EXCLUSION) LIST とは区別する必要があります。COMMAND INCLUSION (または EXCLUSION) LIST にコマンドを含める場合、コマンドとその他のキーワードまたは値とはブランク文字で区切ります。第 2 のブランク文字がある場合、第 2 のブランク文字の後にあるすべての情報は破棄されます。ただし、

CMDTEXT INCLUSION (または EXCLUSION) LIST の場合は、入力したコマンド内のすべての情報が CMDTEXT INCLUSION (または EXCLUSION) LIST 内の詳細と比較されます。以下のいずれかのリスト (CONSOLE INCLUSION LIST、CONSOLE EXCLUSION LIST、COMMAND INCLUSION LIST、COMMAND EXCLUSION LIST、CMDTEXT INCLUSION LIST、CMDTEXT EXCLUSION LIST) を作成する場合は、コンソールまたはコマンドを記述するときに、必要に応じて具体的に指定することも、ワイルドカード文字を使用することもできます。例えば、リストには以下のような特定のコマンドを含めることができます。

```
START MYPROC,INT1=USER,PARM='ANY PARM
```

この場合、コマンド全体と、そのキーワードおよび値を包含 (または除外) の対象リストに含める必要があります。または、以下のようにコマンドのみをリストに含めることもできます。

```
START *
```

この場合、他のキーワードや値に関係なく、START コマンドが包含 (または除外) されます。

突き合わせの順序

前述のリストのいずれかに対するコンソールおよびコマンドの突き合わせは、以下の順序で行われます。以下の順次手順のいずれかで一致するものが見つかり、突き合わせ処理は完了し、その先の突き合わせ手順は行われません。

1. CONSOLE EXCLUSION LIST (または CONSOLE INCLUSION LIST)
2. COMMAND EXCLUSION LIST (または COMMAND INCLUSION LIST)
3. CMDTEXT EXCLUSION LIST (または CMDTEXT INCLUSION LIST)

ただし、それぞれの論理グループ内では、以下の指針に従います。

- CONSOLE EXCLUSION LIST と CONSOLE INCLUSION LIST の両方が指定されることはありません。
- COMMAND EXCLUSION LIST と COMMAND INCLUSION LIST の両方が指定されることはありません。
- CMDTEXT EXCLUSION LIST と CMDTEXT INCLUSION LIST の両方が指定されることはありません。

MVS コマンド管理の開始

NetView コマンド出口を MPF メンバー内に定義し、NetView 自動タスクとオプション・タスクを NetView プログラムに定義した後で、以下を実行することによって MVS コマンド処理を開始できます。

1. MVS コマンド出口の活動化
2. MVS コマンド処理の開始

MVS コマンド管理は、SET MPF=xx が入力されて処理されると、MPFLSTxx メンバーが DSIMCAEX を MVS コマンド出口として定義している場合でも、常に非活動化状態にされます。MVS コマンド管理が SET MPF=xx コマンドによって非活

動状態にされた場合には、NetView MVS コマンド出口 DSIMCAEX が再度活動状態にされた後に SET CNMCAUT=ON または SET CNMCAUT=xx を入力して再始動させる必要があります。

MVS コマンド出口の活動化

MVS コマンド出口を使用可能に設定するには、次の MVS コマンドを発行します。

```
SET MPF=xx
```

この場合、xx は MPFLSTxx の接尾部です。MPFLSTxx には、次のステートメントが必要です。

```
.CMD USEREXIT(DSIMCAEX)
```

MVS コマンド処理の開始

MVS コマンド処理を開始するには、次のコマンドを実行します。

```
SET CNMCAUT=ON or SET CNMCAUT=aa
```

この場合、aa は CNMCAUaa メンバー (ON を除く) の接尾部です。

MVS コマンド管理設定の表示

アクティブな CNMCAUaa メンバーの名前、TRACKING ECHO ステートメント、および ISSUE.IEE295I と CNMCAUT の設定を検索するには、次のコマンドを実行します。

```
DISPLAY CNMCAUT または  
D CNMCAUT
```

CNMCAUaa の内容を検索するには、次のコマンドを実行します。

```
D CNMCAUT=TABLE または  
DISPLAY CNMCAUT=TABLE
```

MVS コマンド管理の停止

MVS コマンド管理の停止および CNMCAUaa メンバーの保持

MVS コマンド管理を停止し、アクティブな CNMCAUxx PARMLIB メンバーをストレージ内に保持するには、任意の MVS コンソールから次のコマンドを実行します。

```
SET CNMCAUT=OFF
```

これによって TRACKING.ECHO の設定および ISSUE.IEE295I の設定が変更されることはありません。CNMCAUT=ON コマンドが実行されると、CNMCAUxx PARMLIB メンバーは再びアクティブになります。

MVS コマンド管理の停止および CNMCAUaa メンバーの保持

CNMCAUaa PARMLIB メンバーを削除し、MVS コマンド管理を停止するには、任意の MVS コンソールから次のコマンドを実行します。

```
SET CNMCAUT=DELETE
```

これにより、MVS コマンドの NetView プログラムへの送信が停止します。
NetView MVS コマンド出口は、引き続き各 MVS コマンドを制御します。
CNMCAUaa メンバーを削除すると、TRACKING.ECHO の設定も Y に変更され、
ISSUE.IEE295I の設定も Y に変更されます。これらはデフォルト値です。

MVS コマンド出口の呼び出しの停止

NetView MVS コマンド出口の呼び出しを停止するには、任意の MVS コンソールから次のコマンドを実行します。

```
SET MPF=yy
```

yy は MPFLSTyy メンバーですが、これには .CMD USEREXIT(DSIMCAEX) ステートメントが含まれていません。あるいは、SET MPF=NO コマンドを入力して MPF 処理を停止することができます。

注: SET MPF=NO を実行すると、すべての MPF 処理が停止されてしまうので、最後の手段としてのみ使用してください。

MVS コマンド出口の非活動化

MVS コマンド出口を非活動化するには、MVS コンソールから以下のいずれかのコマンドを実行します。

MVS コマンド管理は、SET MPF=xx が入力されると、MPFLSTxx メンバーが DSIMCAEX を MVS コマンド出口として定義している場合でも、非活動化状態にされます。MVS コマンド処理は、SET CNMCAUT=ON または SET CNMCAUT=xx を入力することによって、再始動できます。

注: SET MPF=NO を実行すると、すべての MPF 処理が停止されてしまうので、最後の手段としてのみ使用してください。

または

```
SET MPF=NO
```

または

```
SET MPF=yy
```

ここで yy は、MPFLSTxx と同じステートメント (.CMD USEREXIT (DSIMCAEX) ステートメントを除く) が含まれている MPFLSTyy メンバーの接尾部です。

MVS コマンド管理は、SET MPF=xx が入力されると、MPFLSTxx メンバーが DSIMCAEX を MVS コマンド出口として定義している場合でも、常に使用不可能になります。MVS コマンド管理を再始動するには、SET=CNMCAUT=ON または SET CNMCAUT=xx を入力してください。

必要な情報

MPFLSTnn

参照先

z/OS ライブラリー

MVS コマンド管理のテスト

MVS コマンド管理をテストするには、任意の MVS コンソールから次のコマンドを実行します。

SET CNMCAUT=TEST

テスト・モードでは、各コマンドは CNMCAUT=ON であるかのように処理されます。MVS コマンドは、コマンド出口処理の終了直後に MVS によって処理され、その後で NetView プログラムによって処理されます。NetView プログラムはコマンドを MVS に戻しません。

テスト・モードをオフにするには、次のように入力します。

SET CNMCAUT=OFF

テスト・モードは、SET CNMCAUT=ON または SET CNMCAUT=xx コマンドが正常に終了した場合にオフになります。SET CNMCAUT=TEST および SET CNMCAUT=OFF を発行すると、TRACKING.ECHO ステートメントおよび ISSUE.IEE295I ステートメントの設定は変更されません。

排他または包含リストの開始

排他リストまたは包含リストを開始するには、任意の MVS コンソールから次のコマンドを実行します。

SET CNMCAUT=aa

この場合、aa は PARMLIB メンバー CNMCAUaa の接尾部です。

排他または包含リストの変更

排他リストまたは包含リストを変更するには、新規 CNMCAUaa メンバーを開始するか、既存のメンバーを変更して SET CNMCAUT=aa コマンドを再入します。

コンソールとコマンドの包含および排他リストの一般的な処理

コンソールとコマンドの包含および排他リストは、どのように組み合わせても使用できます。次のコマンド・リストの図表は、使用される論理を示しています。

		COMMAND LIST			
		EXCLUSION		INCLUSION	
		match	no match	match	no match
Console List	exclusion match	IGNORE	IGNORE	IGNORE	IGNORE
	exclusion no match	IGNORE	-->NetView	-->NetView	IGNORE
	inclusion match	IGNORE	-->NetView	-->NetView	IGNORE
	inclusion no match	IGNORE	IGNORE	IGNORE	IGNORE

- コンソール排他リスト内に一致するものがあると、コマンドは無視されます。

リスト内に一致するものがないと、コマンド・リストに対してコマンドのテストが行われます。

- コマンド排除リスト内に一致するものがあると、コマンドは無視されます。

リスト内に一致するものがないと、コマンドが NetView プログラムに送信されて、処理されます。

- コマンド排除リスト内に一致するものがあると、コマンドは無視されます。

リスト内に一致するものがないと、コマンドが NetView プログラムに送信されて、処理されます。

- コマンド包含リスト内に一致するものがないと、コマンドは無視されます。

リスト内に一致するものがあると、コマンドが NetView プログラムに送信されて、処理されます。

NetView コマンド出口によって除外されるコマンド

以下のコマンドは、たとえ CNMCAU_{aa} メンバーに組み込まれていても、MVS コマンド出口 (DSIMCAEX) によって NetView に送信されません。

コマンドの最初の文字が単一引用符であるすべてのコマンド

CONTROL E,SEG

CONTROL E,PFK

CONTROL E,N

CONTROL E,F

CONTROL E

CONTROL V

CONTROL T

CONTROL S

CONTROL D

CONTROL C

CONTROL A

CONTROL E,*nn*

CONTROL

CONTROL N

K E,SEG

K E,PFK

K E,N

K E,F

K E

K V

K T

K S

K D

K C

K A

I

K

K N

MOUNT

LOGON

RO T=nnn,dest,mvscmd または ROUTE T=nnn,dest,mvscmd (*dest* が *ALL または *OTHER のようにアスタリスク (*) で始まる場合、および *dest* が (sysnamelist) のように左括弧で始まる場合を除く)

S INIT.INIT

SET MPF=NO

START INIT.INIT

T MPF=NO

制限事項

- MVS コマンド管理を使用して START コマンドを処理するために NetView プログラムに渡すと、START コマンドを使用する際に問題が発生することがあります。これは、この関数を使用した結果 MGCRC から戻される戻りコード (R15) および ASID (R0) が不正であるためです。

注: これは、DB2 および MQ Series が内部で実行する START コマンドの問題の発生原因として知られています。

内部的に発行された DB2 および MQ START コマンドは、あらゆるコマンド自動化処理から除外してください。これを行うには、これらのコマンドをコマンド排他リストに追加するか、特別にコマンド包含リストに入れないでおきます。

- TSO エンド・ユーザーがログオンするたびに、MVS コマンド管理は内部で実行される LOGON コマンドを特別に除外します。

LOGON コマンドは、ASID と戻りコードも戻します (START コマンドと同様)。

- MVS コマンド管理は、テープのマウントを手動で要求するためにオペレーターが実行する MOUNT コマンドを特別に除外します。

MOUNT コマンドは、ASID と戻りコードも戻します (START コマンドと同様)。

- MVS コマンド管理は、実際の拡張複数コンソール・サポート (EMCS) コンソールを制御するためにオペレーターが実行する各種 K (CONTROL) コマンドを明確に除外します。

K コマンド・プロセッサは、コマンド自動化コードとの互換性がないことを前提とします。

- REXX 内の等号 (=) の右側のストリングは、250 文字を超えてはなりません。

コマンド・テキストは、印刷可能な 16 進数形式で REXX exec CNMEMCXY 内に渡されます (REXX がコマンドを構文解析しないようにするため)。処理できるのは、123 文字以内のコマンドだけです (コマンド長を印刷可能な 16 進数形式で伝達するために 4 を使用)。

注: 待機処理を CNMEMCXY でコーディングしないで ください。待機が終了するまでキューに入れられたままの MVS コマンドの処理が遅れる可能性があります。

この場合、待機処理には、REXX 待機、PIPE 待機、WTOR、およびコマンドの構文解析プル・タイプが含まれます。

- コマンドが一度のみ処理されるように、現行メカニズムを使ってそれらのコマンドを「タグ化」しているため、処理可能なコマンドの最大長は、さらに 122 文字に削減されます。

唯一これらの制限にアプローチすることが知られているコマンドは、内部で実行される SEND コマンドです。これらのコマンドは、ジョブが完了したときや、NJE ファイル伝送が発生したときに TSO エンド・ユーザーに通知する際に使用します。これらのコマンドは、INTERNAL および INSTREAM というコンソール名を指定するコンソール排他リストを使用することによって、現在、処理対象から除外されています。

NetView プログラムでの MVS コマンド管理の処理

CONSOLE EXCLUSION LIST の行は、2 桁目で始まります。

注: ワードとワードの間に余分なブランクを入れないで ください。

MVS コマンド管理の処理が活動化された後で、除外されない MVS コマンドが、それぞれ NetView プログラムに送信され、さらに処理されます。NetView 側では、オプション・タスク DSIMCAT が PPI から MVS コマンドを受け取り、REXX CLIST CNMEMCXY を呼び出します。CNMEMCXY は、実行されると以下のパラメーターを受け取ります。

注: 待機処理を CNMEMCXY でコーディングしないで ください。待機が終了するまでキューに入れられたままの MVS コマンドの処理が遅れる可能性があります。

この場合、待機処理には、REXX 待機、PIPE 待機、WTOR、およびコマンドの構文解析プル・タイプが含まれます。

MODE=mode

mode は T (テスト) または O (オン) です。

ISYN=isyn

isyn は実行システム名で、16 進数です。 *isyn* は 16 桁の 16 進数です。

CNNM=consname

consname は実行コンソール名で、16 進数です。 *consname* は 16 桁の 16 進数です。

C4ID=consid

consid は実行コンソール ID で、16 進数です。 *consid* は 8 桁の 16 進数です。

TOKN=token

token はユーザーのコマンド・トークンで、16 進数です。トークンは、8 桁の 16 進数です。

AUTH=auth

auth はユーザーのコマンド許可で、16 進数です。 *auth* は 4 桁の 16 進数です。

ASID=asid

asid はユーザーの ASID で、16 進数です。 *asid* は 4 桁の 16 進数です。

TRNM=termname

termname はユーザーの端末名で、16 進数です。 *termname* は 16 桁の 16 進数です。

CLNM=conclass

conclass はコンソール・クラス名で、16 進数です。 *conclass* は 16 桁の 16 進数です。

CART=cart

cart はコマンドおよび応答トークンで、16 進数です。 *cart* は 16 桁の 16 進数です。

OCID=ocid

ocid は発信コンソール ID で、16 進数です。 *ocid* は 16 桁の 16 進数です。

UTKN=utoken

utoken はユーザー・トークンで、16 進数です。 *utoken* は 160 桁の 16 進数で、変更することはできません。 *utoken* は、未変更のまま MVS に戻さなければなりません。

CTXT=cmdtext

cmdtext はコマンド・テキストで、16 進数です。 *cmdtext* は 250 桁までの 16 進数です。最初の 4 桁は、コマンドの長さです。

注: CNMEMCXY への入力、MODE を除きすべて 16 進数です。入力を検査するには、REXX 関数 X2C を使ってその入力を文字フォーマットに変換します。

コマンドを検査した後で、呼び出し側の REXX CLIST に戻りコードを送信し、z/OS にコマンドを戻して、さらに処理を続けたいことを示します。

以下は、戻ることができる戻りコードとその意味です。

- 0 処理を継続します。コマンドは未変更です。 MVS コマンドは MVS に戻されます。
- 4 コマンド・テキストが変更されました。変更済みのコマンドは MVS に送信する必要があります。
戻りコード 4 が戻された場合は、変更された MVS コマンドを MVSCMD という名前のセーフに保存する必要があります。
- 8 コマンドは MVS に戻してはいけません。

MODE=T が指定されると、MVS コマンドは、戻りコードに関係なく MVS には戻されません。

コマンドの長さを変更する場合は、それに従って長さフィールド (コマンド・テキストの最初の 4 バイト) も変更してください。 (コマンド・テキストの最大長は 122 文字です。)

注: 待機処理を CNMEMCXY でコーディングしないでください。待機が終了するまでキューに入れられたままの MVS コマンドの処理が遅れる可能性があります。

この場合、待機処理には、REXX 待機、PIPE 待機、WTOR、およびコマンドの構文解析プル・タイプが含まれます。

MVS コマンド管理処理の保護

オペレーターが無許可の MVS コマンドを実行できないようにするには、NetView コマンドの DSIMCAP をすべての NetView オペレーターから保護する必要があります。ただし、DSIMCAOP、または以下の CNMSTYLE ステートメントで定義されている NetView オペレーターは例外です。

```
Function.autotask.mvsAuto=opid
```

これは DSIMCAP、CNMEMCXX、または CNMEMCXY の発行を許可されている唯一のタスクであることを確認してください。

AUTHCHK=TARGETID を使用している場合、他の NetView オペレーターが自動タスクにコマンドを送って実行することができないように、DSIMCAOP、または MVS コマンド管理処理によって使用される定義済みの自動タスクを保護する必要があります。

付録 I. 自動操作のサンプル・セット

NetView プログラムには、自動操作を開始するのに使用できるサンプル・セットが含まれています。これらのサンプル・セットは、**自動操作サンプル・セット**と呼ばれています。自動操作サンプル・セットは、自動操作の手法の例を紹介する目的で作成されています。このサンプル・セットは、そのまま自動操作に使用できるものではありません。

自動操作サンプル・セットは、次のサンプル・セットで構成されています。

メッセージの抑止

このセットには、2 つの MVS メッセージ抑止リスト (消極的リストと積極的リスト) が含まれています。

基本自動操作

このセットには、自動化テーブル項目、コマンド・リスト、および始動プロシージャが含まれています。このセットでは、日常のオペレーター・アクションの自動化方法について紹介しています。

拡張自動操作

このセットには、自動化テーブル項目とコマンド・リストが含まれています。このセットでは、本書で説明されている自動操作方法を使用して、サブシステムとアプリケーションの初期設定、リカバリー、およびシャットダウンの方法を紹介しています。

ログ分析プログラム

このセットには、JES2 および JES3 のログ分析プログラムが含まれています。このプログラムは、他のログを分析するために修正することができます。分析プログラムは、抑止または自動化の対象になる頻出メッセージを識別する際に役立ちます。

セットアップ・サンプル

このセットには、自動操作サンプル・セットの他のコンポーネントのサンプル名を変更する際に使用できるサンプルが組み込まれています。

自動操作サンプル・セットに含まれるサンプルは、それぞれの名前で識別することができます。CNMS62、CNMS64、CNME62、または CNME64 で始まる名前のサンプルは、自動操作サンプル・セットに属しています。自動操作サンプル・セットに含まれているすべてのサンプルについては、691 ページの『コマンド・リストおよびサンプルの相互参照リスト』で説明されています。

自動操作サンプル・セットの使用

自動操作サンプル・セットを自動化操作の例として使用するには、次のステップに従ってください。

1. 自動操作サンプル・セットのサンプルの名前をすべて変更してください。
2. まず、メッセージ抑止の例に、メッセージ抑止サンプル・セットのコンポーネントを使用してください。メッセージ抑止については、341 ページの『第 19 章 メッセージの抑止とアラートのフィルター操作』で説明しています。

3. ログ分析プログラムを使用すると、メッセージの抑止と NetView プログラムの自動化に役立ちます。ログ分析プログラムは、自動化の対象になる共通メッセージを示します。
4. NetView プログラムと MVS の間の通信を準備してください。このステップは、基本サンプル・セットおよび拡張サンプル・セットを活動化するために必要です。
5. メッセージ自動操作 (361 ページの『第 22 章 メッセージおよび管理サービス単位 (MSU) の自動化』で説明されている方法) の例として、基本自動操作サンプル・セットを使用してください。それぞれの環境に合ったこのサンプル・セットの一部をアクティブにしてください。
6. 整合性のとれた自動操作 (405 ページの『第 23 章 整合性のとれた自動操作の確立』で説明されている方法) の例として、拡張自動操作サンプル・セットを使用してください。それぞれの環境に合った拡張自動操作サンプル・セットの一部をアクティブにしてください。

自動操作サンプル・セットの位置および名前変更

以下のセクションでは、MVS システムの自動操作サンプル・セットの位置と改名方法について説明します。

次の表では、NetView インストール処理の一部としてサンプルをインストールした直後の、サンプルの名前および位置を示しています。

表 36. MVS システムにおける自動操作サンプル・セットの位置

サンプル・セット	サンプル・タイプ	ライブラリー	メンバー名
メッセージの抑止	サンプル MPFLSTxx メンバー	SYS1.CNMSAMP	CNMS6201-CNMS6202
基本自動操作	各種 NetView サンプル	SYS1.CNMSAMP	CNMS6205-CNMS6206
	サンプル・プロシージャ	SYS1.CNMSAMP	CNMS6211-CNMS6214
	サンプル・パラメータ	SYS1.CNMSAMP	CNMS6221-CNMS6224
	サンプル・コマンド・リスト	SYS1.CNMCLST	CNME6201-CNME6205
拡張自動操作	各種 NetView サンプル	SYS1.CNMSAMP	CNMS6401-CNMS6410
	サンプル・パネル	SYS1.CNMSAMP	CNMS64P0-CNMS64P5
	サンプル・コマンド・リスト	SYS1.CNMCLST	CNME6400-CNME6440
ログ分析	サンプル分析プログラム	SYS1.CNMSAMP	CNMS6207
	サンプル JCL	SYS1.CNMSAMP	CNMS62J2
設定	改名用 JCL	SYS1.CNMSAMP	CNMS62J1

サンプル CNMS62J1 を使用して、自動操作サンプル・セットに提供されたサンプルの名前を変更してください。これらのサンプルは、新しい名前で参照するため、実行前に名前を変更してください。サンプルの名前を変更するには、次のステップを実行してください。

- CNMS62J1 を調べ、PROC ステートメントに設定されているシンボルがそれぞれの環境に適切であり、各ステップの OUTDD DD ステートメントにサンプルを保管させたいデータ・セットが指定されていることを確認してください。
CNMS62J1 に指定されている JCL は、名前が変更されたサンプルを新しいデータ・セットにコピーします。既存のデータ・セットを使用したい場合には、それに合わせて、CNMS62J1 に含まれているサンプル JCL を更新してください。
- PARMLIB メンバーが保管されているライブラリーを変更するには、サンプル CLRSMF (CNMS6212) および LGPRNT (CNMS6213) が新しいサンプルを指すように更新してください。
- PROCLIB サンプルが含まれているデータ・セットは有効な PROCLIB データ・セットでなければならないため、そのデータ・セットを JES プロシーチャーの PROC00 DD の DD ステートメントの 1 つとして指定してください。
- 名前が変更された DSIPARM および DSIPRF サンプルが含まれているデータ・セットを、NetView 始動プロシーチャーの該当する DD ステートメントに指定してください。この変更を NetView 始動プロシーチャーで有効にするために、NetView をいったん停止してからもう一度始動してください。NetView プログラムで提供されているサンプルを使用している場合には、その NetView 始動プロシーチャーは CNMPROC (CNMSJ009) です。
- TSO SUBMIT コマンドまたは NetView SUBMIT コマンドのいずれかを使用して、CNMS62J1 (バッチ・ジョブとして実行される) をシステム入力ストリームに組み込み、実行依頼してください。
- 必要であれば、名前を変更したサンプルの 2 番目のコピーを作成してください。そうすると、サンプルの一部を変更すれば、すでに名前を変更済みのコピーのバックアップになります。

メッセージ抑止サンプル・セットの使用

メッセージの抑止を開始するために、メッセージ抑止サンプル・セットを使用するか、あるいはサンプル・セットのコンポーネントを既存のメッセージ抑止機能に組み込むことができます。詳しくは、341 ページの『第 19 章 メッセージの抑止とアラートのフィルター操作』を参照してください。

ログ分析プログラムの使用

ログ分析プログラムは自動操作を実行しませんが、独自の自動操作を作成するためのツールとして使用することができます。詳しくは、523 ページの『ログ分析プログラム』を参照してください。

NetView プログラムと MVS 間の通信のセットアップ

MVS システムで基本または拡張サンプル・セットをアクティブにする前に、NetView プログラムは、オペレーティング・システムへコマンドを送信し、MVS システム・メッセージを受信できなければなりません。NetView プログラムをサブシステム (IEFSSNxx を持つ) として MVS へ定義し、メッセージが NetView プログ

ラムへ転送されるようにします。MPFLSTxx PARMLIB メンバーをチェックし、サンプル・セットが要求しているメッセージを抑止しないようにしてください。

333 ページの『システム自動化のための z/OS システムの準備』は、サブシステムとしての NetView プログラムの定義、および MVS から NetView プログラムへのメッセージの転送について説明しています。

基本自動操作サンプル・セットの使用

以下のセクションでは、基本自動操作サンプル・セットにより提供されている機能、サンプル自動化テーブルの機能提供方法、および基本自動操作サンプル・セットの活動化方法について説明します。

基本自動操作サンプル・セットにより実行される機能

基本自動操作サンプル・セットにあるサンプルは、117 ページの『第 4 部 NetView プログラム自動操作機能』で説明されている自動操作機能を使用して、日常のオペレーターのアクションをどのように自動化するかを示しています。

サンプルには、自動化テーブル項目、コマンド・リスト、および始動プロシージャが含まれています。コマンド・リストは NetView コマンド・リスト言語と REXX の両方で作成されています。NetView REXX 機能を備えているシステムで実行する場合には、コマンド・リストは REXX で実行します。NetView REXX 機能を備えていないシステムで実行する場合には、コマンド・リストは NetView コマンド・リスト言語で実行します。

基本自動操作サンプル・セットのサンプルには、実行できる自動化アクションの小規模なサブセットが含まれています。サンプルでは、NetView 自動操作機能を使用して、環境内で発生するイベントに対する簡単に日常的な応答をどのように自動化するかを示しています。基本自動操作サンプル・セットに含まれているサンプルは、次の自動化アクションを提供します。

- SYS1.LOGREC データ・セットをクリアする
- SYS1.LOGREC データ・セットを印刷する
- チャンネルをオンラインに変更する
- GTFTRACE パラメーター要求に応答する
- SMF メモリー・ダンプ・タスクに MVS START コマンドを出す
- DSILOG タスクが基本タスクから 2 次タスク、またはその逆へ切り替えられたあとに、ネットワーク・ログを印刷する
- JES2 スプール使用率をモニターし、使用率が 70% を超えていて、24 時間以上保留されているファイルを除去する

基本自動操作サンプル・セットで使用されている自動化テーブル

本セクションでは、基本自動操作サンプル・セットで使用されている自動化テーブルと、その機能の提供方法について説明します。655 ページの図 198 は、自動化テーブルを示しています。

```

*****
* (C) COPYRIGHT IBM CORP. 2010 *
* IEBCOPY SELECT MEMBER=((CNMS6205,ACOTABLE,R)) *
* LAST CHANGE: 08/25/10 *
* *
* DESCRIPTION: *
* DESCRIPTION: SAMPLE DSIPARM - MSG AUTOMATION DEFS FOR BASIC *
* AUTOMATION SAMPLE SET *
* *
* CNMS6205 CHANGED ACTIVITY: *
* CHANGE CODE DATE DESCRIPTION *
* ----- *
*****
*
* IF MSGID = 'IFB040I' 1
* THEN EXEC(CMD('MVS S CLRLOG') ROUTE(ONE AUTO1));
*
* IF MSGID = 'IFB060E' 2
* THEN EXEC(CMD('MVS S LGPRNT') ROUTE(ONE AUTO1));
*
* IF MSGID = 'IOS150I' & TOKEN(3) = DEVICE 3
* THEN EXEC(CMD('MVS VARY ' DEVICE ',ONLINE') ROUTE(ONE AUTO1));
*
* IF MSGID = 'AHL125A' & TEXT(1) = REPLYID . 4
* THEN EXEC(CMD('MVS REPLY ' REPLYID ',U')
* ROUTE(ONE AUTO1));
*
* IF MSGID='DSI556I' & TOKEN(2)='DSILOG' & TOKEN(6) = 'CLOSE' & 5
* TEXT = . 'DDNAME = '' LOGID '' RETURN CODE =' .
* THEN EXEC(CMD('MVS S DSIPRT,NAME='LOGID) ROUTE(ONE AUTO1));
*
* IF (MSGID = 'IEE362A' | MSGID = 'IEE362I') & TEXT = STRNG 6
* THEN EXEC(CMD('IEE362A ' STRNG) ROUTE(ONE AUTO1));
*
* IF MSGID = '$HASP646' & TEXT = STRNG 7
* THEN EXEC(CMD('$HASP646 ' STRNG) ROUTE(ONE AUTO1));

```

図 198. 基本自動操作サンプル・セット自動化テーブル項目

図 199 では、基本自動操作サンプル・セットの自動化テーブルにより自動化されるメッセージをリストしています。

```

IFB040I SYS1.LOGREC AREA IS FULL,hh.mm.ss A
IFB060E SYS1.LOGREC NEAR FULL B
IOS150I DEVICE ddd NOW AVAILABLE FOR USE C
xx AHL125A RESPECIFY TRACE OPTIONS OR REPLY U D
DSI556I DSILOG : VSAM DATASET 'CLOSE' COMPLETED,DDNAME='ddname' E
RETURN CODE=X'code',ACB ERROR FIELD=X'code'.
IEE362n SMF ENTER DUMP FOR SYS1.MANx ON ser F
$HASP646 xx PERCENT SPOOL UTILIZATION G

```

図 199. 基本自動操作サンプル・セットの自動化テーブルにより自動化されるメッセージ

コマンドの発行: 図 198 のステートメント 1 は、自動化テーブルからシステム・コマンドを直接出します。このステートメントは、図 199 のメッセージ A を自動化します。

NetView プログラムがメッセージ A を受信すると、サンプル・プロシージャー CLRLOG 用の MVS START コマンドを出します。このプロシージャーは SYS1.LOGREC データ・セットをクリアします。CLRLOG は、自動タスク AUTO1 へ渡され処理されます。MVS は NetView コマンド・プロセッサであり、NetView プログラムから MVS システム、サブシステム、およびアプリケーション

ン・コマンドの入力を可能にするものです。NetView コマンドも VTAM コマンドも出すことができます。この場合、コマンド・プロセッサをコマンドの前に指定しなくても自動化テーブルから直接出すことができます。

ステートメント **1** では、処理されるコマンドは 1 人のオペレーターだけに経路指定され、自動タスク AUTO1 へ渡されます。AUTO1 がアクティブでなければ、アクションは取られないので、図 200 に示しているステートメントを指定することもできます。

```
ROUTE(ONE AUTO1 opid1 opid2 opid3)
```

図 200. ROUTE コマンドでの複数の自動タスクとオペレーターの指定

図 200 に示されている ROUTE コマンドでは、opid1、opid2、および opid3 は他の自動タスクかオペレーター ID です。その場合、コマンドは、アクティブであるかまたはログオンされているリスト上の最初のタスクへ経路指定されます。

655 ページの図 198 に示している **2** では、同じ方法を使用して、メッセージ ID IFB060E (655 ページの図 199 のメッセージ **B**) を受信すると、SYS1.LOGREC データ・セットを印刷します。

変数への値の割り当て: 655 ページの図 198 に示している **3** では、メッセージ ID IOS150I (655 ページの図 199 のメッセージ **C**) のメッセージを受信すると、チャンネルをオンラインにします。

ステートメント **3** は、トークン 3 (ddd、装置番号) の情報を取り出し、その値を DEVICE という名前の変数としてコマンドに渡します。DEVICE が単一引用符で囲まれていない理由は、DEVICE が比較項目ではなく変数であるためです。次に、変数 DEVICE はアクションで使用されて処理されます。ここでも、MVS は、NetView コマンド・プロセッサ (VARY コマンドを NetView プログラムから出せるようにする) であり、自動タスク AUTO1 に経路指定されて処理されます。

655 ページの図 198 に示しているステートメント **4** では、コマンドまたはコマンド・プロシージャーに変数を渡し、WTOR に応答します。ステートメント **4** は、メッセージ AHL125A (655 ページの図 199 のメッセージ **D**) に応答します。メッセージ AHL125A は、汎用トレース機能 (GTF) のトレース・オプションを再指定することを要求するか、あるいは初期設定を継続するために U に応答することを要求します。メッセージ AHL125A では、xx は応答 ID です。ステートメント **4** は、応答 ID を変数 REPLYID として取り出します。TEXT(1) は、1 桁目から始まるテキストを示しています。ピリオド (.) は、次のブランクまでに指定されているテキストのみを REPLYID として使用することを意味するプレースホルダーです。したがって、変数 REPLYID には 1 桁目から最初のブランクまでのテキストが割り当てられます。次に、自動化テーブルは、EXEC ステートメントに指定されているコマンドへ REPLYID 変数を渡します。ピリオドを指定しないと、メッセージの最後まですべてのテキストが REPLYID 変数に割り当てられます。

ステートメント **4** の残りの部分では、GTFTRACE パラメーターの要求に対し U を応答しています。このステートメントにより、自動タスク AUTO1 のもとで MVS コマンド REPLY が該当する応答 ID に出されます。

655 ページの図 198 に示しているステートメント **5** は、メッセージ ID DSI556I (655 ページの図 199 のメッセージ **E**) に関するもので、メッセージ・テキストの一部を解析し、その一部を自動化テーブルから出される MVS START コマンドで使います。そのコマンドは、DSI556I メッセージを受信したときに、1 次ネットワーク・ログまたは 2 次ネットワーク・ログを印刷します。

ステートメント **5** は、3 つの比較項目と変数 LOGID の割り当てで構成されます。自動化テーブルがアクションを開始するのは、次のすべての比較条件が満たされたときだけです。

- メッセージ ID が DSI556I でなければならない。
- 2 番目のトークンが DSILOG でなければならない。
- 6 番目のトークンが 'CLOSE' でなければならない。

これらの条件が満たされると、次のアクションが取られます。

- メッセージ・テキストが解析され、テキストを LOGID という名前の変数へ割り当てます。この変数はストリング DDNAME = ' とストリング ' RETURN CODE = の間にあります。
- MVS START コマンドがサンプル・プロシージャ DSIPRT へ出されます。このプロシージャは 1 次ログまたは 2 次ログ (どちらか直前でクローズされた方) のログを印刷します。

メッセージ・テキストの中に、単一引用符 (') があり、それを比較条件の中に入れないとときには、単一引用符を 2 つの連続した単一引用符 (") で表します。自動化テーブル・ステートメントの CLOSE の前後に指定されている 3 つの単一引用符のうち、外側の引用符は比較条件を満たすために、テキストが TOKEN(6) に含まれていなければならないことを示しています。単一引用符がないときには、テキストは変数に割り当てられます。内側の 2 つの単一引用符は、メッセージ・テキストである CLOSE を囲む単一引用符です。

コマンド・リストおよびコマンド・プロセッサの呼び出し: 655 ページの図 198 に示している **1** から **5** まではすべて、自動化テーブルから直接コマンドを出しています。コマンドはすべて MVS システム・コマンドまたはサブシステム・コマンドでしたが、これらのコマンドを、アプリケーション・コマンド、あるいは NetView または VTAM コマンドにすることもできます。655 ページの図 198 に示しているステートメント **6** は、メッセージ ID IEE362A または IEE362I (655 ページの図 199 のメッセージ **F**) に関連するものですが、単一コマンドより複雑なアクションが必要な場合の、NetView 自動操作機能の使用法を示しています。これらの場合は、自動化テーブル・ステートメントは、NetView コマンド・リスト (NetView コマンド・リスト言語または REXX で作成されている)、またはコマンド・プロセッサ (PL/I、C またはアセンブラで作成されている) を呼び出すことができます。

ステートメント **6** は、メッセージ ID IEE362A または IEE362I (両方ともメッセージ **F** に示されているテキスト) を探しています。

ステートメント **6** は、どちらかのメッセージ ID が受信されると、メッセージ・テキスト全体が STRNG という名前の変数として取り出され、コマンド・リスト IEE362A に渡されて自動操作処理されることを指定しています。コマンド・リスト

はメッセージ・ストリングを解析してメッセージに含まれている x の値を判別し、MVS START コマンドを出すときにその値を使用してサンプル・プロシーチャー CLRSMF を始動します。

通常、自動化するアクションに複数のコマンドを出す必要があるか、またはメッセージから情報を取り出す処理が複雑すぎて自動化テーブルでは処理しきれない場合には、コマンド・リストまたはコマンド・プロセッサを使用します。また、複数行のオペレーター宛 (MLWTO) メッセージでは、コマンド・リストまたはコマンド・プロセッサで実行しなければならない特殊メッセージ処理が必要です。

655 ページの図 198 に示しているステートメント **7** は、メッセージ ID \$HASP646 (655 ページの図 199 のメッセージ **G**) に関連するものですが、実際には、JES2 スプール使用率をモニターするサンプルの一部です。スプールの使用率をモニターするには、NetView タイマーを設定して一定間隔ごとに \$D SPOOL コマンドを出します。提供されている例では、自動タスク AUTO1 が初期設定されるたびに、NetView コマンド EVERY が初期コマンド・リストで出され、\$D SPOOL コマンドが 24 時間ごとに出力されるようにスケジュールされています。

\$D SPOOL コマンドが実行されると、メッセージ \$HASP646 が出力されます。自動化テーブルがメッセージ \$HASP646 を受信すると、そのメッセージ・テキストを NetView コマンド・リスト \$HASP646 へ渡します。そのコマンド・リストでは、スプールの使用率 (%) をチェックします。スプールの使用率が 70% を超えると、コマンド・リストは 24 時間以上経過した保留ジョブすべてを取り消します。さらに、コマンド・リストは 1 時間ごとに \$DSPOOL2 コマンド・リストを駆動する 2 番目のタイマーを設定します。そのコマンド・リストは、スプールの使用率が 20% 未満になるまでスプール・スペースをモニターします。

一定の間隔で照会コマンドを出し、状況に応じてアクションを取る方法は、**能動モニター** と呼ばれます。基本自動操作サンプル・セットでは、非常に基本的な能動モニターの例を使用しています。663 ページの『能動モニター』を参照してください。

基本自動操作サンプル・セットの活動化

自動操作サンプル・セットを利用して独自の自動操作を実現しようとする場合、基本自動操作サンプル・セットをアクティブにすることができます。基本自動操作サンプル・セットからサンプルをアクティブにするには、次の操作が必要です。

- 必要なメッセージが NetView プログラムへ転送されることを確認する。
- コマンド・リストにコマンド同義語を定義する。
- サンプル自動化テーブルを作成し活動化する。
- AUTO1 自動タスクを活動化する。
- 基本自動操作サンプル・セットを活動化する。

以下のセクションでは、基本自動操作サンプル・セットを活動化するためのステップを説明します。

コマンド・リスト同義語の定義: 基本自動操作サンプル・セットを実行するには、CNMS6206 に含まれているコマンド定義と既存の CNMCMD とを組み合わせ、必要なコマンド同義語を定義しなければなりません。

CNMS6206 には、基本自動操作サンプル・セット・コマンド・リストごとに項目が 1 つあります。コマンド同義語は、基本自動操作サンプル・セットで使用されます。いずれかの同義語がシステムですでに定義されている同義語と矛盾する場合、基本自動操作サンプル・セットのコマンド同義語を変更しなければならないことがあります。その場合、そのコマンド同義語を参照しているすべてのサンプルをその同義語で変更するようにしてください。

サンプル自動化テーブルの作成および活動化: 基本自動操作サンプル・セットのサンプル自動化テーブルは、ACOTABLE (CNMS6205) です。ACOTABLE を実行するには、次のことを行います。

- NetView 自動化テーブルを使用できるように準備する。
- テーブルの構文をテストする。
- テーブルを活動化する。

サンプル自動化テーブルの準備: 基本自動操作サンプル・セットの自動化テーブルである ACOTABLE (CNMS6205) の内容をよく調べて、どのサンプルを使用するかを決定しなければなりません。基本自動操作サンプル・セットのサンプルのうちアクティブにならないサンプルがあるときには、使用したくないサンプルを駆動する ACOTABLE 項目を削除してください。

基本自動操作サンプル・セットとともに提供されている自動化テーブル ACOTABLE (CNMS6205) は、独立型 NetView 自動化テーブルとして使用することもできるし、既存テーブル (通常は、実動環境ですでに使用されている) と組み合わせて使用することもできます。すでに自動化テーブルを使用しているときには、使用したいサンプルを駆動している項目を ACOTABLE から既存の自動化テーブルへコピーすることもできます。または、それらの項目を別のファイルに入れておき、%INCLUDE ステートメントを使用して独自のテーブルへ組み込むこともできます。

使用する ACOTABLE 項目が、自動化テーブルの既存の項目と矛盾していないことを確認してください。例えば、CONTINUE(Y) を指定していない場合にメッセージ ID が重複していると、自動化テーブルは順次処理されるので、メッセージ受信時にテーブル内の最初のステートメントだけが駆動されます。矛盾があれば、テーブルを編集して問題を解決します。ACOTABLE 項目の組み込み後、処理効率を向上させるためにテーブルを再編成します。266 ページの『自動化テーブルの設計上のガイドライン』には、従わなければならない方針がリストされています。

サンプル自動化テーブルの構文のテスト: 基本自動操作サンプル・セットからの自動化テーブルと既存の自動化テーブルとを組み合わせる自動化テーブルを作成したら、図 201 のコマンド **1** を出して構文をテストしてください。ここで、automem とは、自動化テーブルが入っているメンバーの名前です。構文が正しいときには、メッセージ CNM501I (図 201 のメッセージ **2**) が表示されます。

```
AUTOTBL MEMBER=automem,TEST 1
CNM501I  TEST OF MESSAGE AUTOMATION FILE "automem" WAS SUCCESSFUL. 2
```

図 201. 自動化テーブルのテスト

構文が正しくないときには、構文を訂正し、もう一度テストしてください。

サンプル自動化テーブルの活動化: 自動化テーブルのテストが正常に終了したら、その自動化テーブルを活動化することができます。アクティブの自動化テーブルを変更するのに、NetView を停止し、再始動する必要はありません。図 202 に示している AUTOTBL コマンドを使用して、自動化テーブルを活動化してください。

```
AUTOTBL MEMBER=automem
```

図 202. 自動化テーブルの活動化

指定したテーブルは、NetView プログラムを再始動するか、自動化テーブルを非活動化するか、あるいは別の自動化テーブルを活動化するまで、アクティブの自動化テーブルとして扱われます。

NetView プログラムが起動されるたびにテーブルを自動的に活動化するには、CNMSTYLE メンバーでこのテーブルを指定してください。

自動タスク AUTO1 の活動化: 自動タスク AUTO1 (基本自動操作サンプル・セットにより使用される) が初期コマンド・リストにより開始されることを確認してください。CNME1034 (NetView プログラムに組み込まれている初期コマンド・リスト) を使用している場合には、AUTO1 自動タスクは既に開始されています。別の初期コマンド・リストを使用するときには、図 203 に示されているコマンド **1** を組み込むようにしてください。

```
AUTOTASK OPID=AUTO1 1  
EXCMD AUTO1 AUTO1IC 2
```

図 203. 自動タスク AUTO1 の活動化

JES2 スプール使用率のモニターおよびリカバリー・サンプルを使用したいときには、\$D SPOOL コマンドをスケジュールして一定間隔で処理しなければなりません。そのためには、AUTO1 のもとで AUTO1IC コマンド・リストを実行してください。許可オペレーターのコンソールから、図 203 に示しているコマンド **2** を入力すれば、AUTO1 のもとで AUTO1IC コマンド・リストを実行することができます。

AUTO1 が初期コマンド・リストにログオンしそのリストを実行するたびに \$D SPOOL コマンドを自動的にスケジュールして繰り返し実行するには、AUTO1 の初期コマンド・リストから AUTO1IC コマンド・リストを呼び出してください。AUTO1 の初期コマンド・リストは、NetView サンプルの DSIPRF のメンバー DSIPROFC (CNMS1026) に LOGPROF2 (CNME1032) として指定されます。したがって、NetView サンプルを使用していれば、LOGPROF2 に AUTO1IC コマンド・リストを呼び出す 1 行を追加してください。

基本自動操作サンプル・セットのテスト: 基本自動操作サンプル・セットを実動で使用する前に、NetView 自動化テーブル項目が期待通りのアクションを行っていることを確認してください。自動化のテストについては、531 ページの『第 34 章 自動化テーブルのテスト』を参照してください。

拡張自動操作サンプル・セットの使用

拡張自動操作サンプル・セットには、サブシステムおよびアプリケーションの初期設定、モニター、リカバリー、およびシャットダウンなどの機能を自動化する際の、NetView プログラムの使用方法を示すサンプルが入っています。このサンプルは、自動操作にそのまま使用するものではなく、405 ページの『第 23 章 整合性のとれた自動操作の確立』で説明している整合性のとれた自動操作の例です。

拡張自動操作サンプル・セットには、コマンド・リスト、自動操作の状況を表示するフルスクリーン・パネル、および自動化テーブル項目などのネットワーク定義サンプルが入っています。コマンド・リストは NetView コマンド・リスト言語と REXX の両方で作成されています。NetView REXX 機能を備えているシステムで実行する場合には、コマンド・リストは REXX で実行します。NetView REXX 機能を備えていないシステムで実行する場合には、コマンド・リストは NetView コマンド・リスト言語で実行します。

注: 本セクションでは、**プロダクト** という用語は、拡張自動操作サンプル・セットを使用して自動化されるサブシステムまたはアプリケーションのことを意味します。691 ページの『コマンド・リストおよびサンプルの相互参照リスト』には、拡張自動操作サンプル・セットに含まれているすべてのサンプルの相互参照リストがあります。

拡張自動操作サンプル・セットにより実行される機能

拡張自動操作サンプル・セットは、405 ページの『第 23 章 整合性のとれた自動操作の確立』で説明した方法を使用してシステム上の特定のプロダクトの初期設定、モニター、リカバリー、およびシャットダウンを自動化する方法を示しています。また、このサンプル・セットは、415 ページの『フルスクリーン・パネルの作成』で説明した VIEW パネルを使用して、自動化環境でできるように調整されたオペレーター・インターフェースの使用法についても示しています。拡張自動操作サンプル・セットには、表 37 にリストされているプロダクトの操作を自動化するコマンド・リストが含まれています。

表 37. 各プロダクトの拡張自動操作サンプル・セットにより自動化される処理

プロダクト	初期設定	モニター	リカバリー	シャットダウン
JES2	●	●	●	●
JES3	●	●	●	●
VTAM	●	●		●
TSO	●	●		●
IMS	●	●	●	●
CICS	●	●	●	●

初期設定: 拡張自動操作サンプル・セットは、システム全体および特定のプロダクトの初期設定を自動化する方法を示しています。拡張自動操作サンプル・セットは、最初にコマンド・リスト AOPIVARS (CNME6400) (拡張自動操作サンプル・セットの主要なコマンド・リスト) を処理して、システムの初期設定を正常に行います。AOPIVARS は、自動化グローバル変数を初期設定し、使用される自動化テーブルをロードしてから、自動タスク AUTOMGR (拡張自動操作サンプル・セットの中心となる自動タスク) にログオンします。

AUTOMGR の初期コマンド・リスト AOPIMGIC (CNME6402) は、自動化プロダクトごとに自動タスクにログオンし、能動モニター・コマンド・リスト AOPMACT を定期的に起動するタイマー・コマンドを処理して、AOPIVARS で初期設定された自動化プロダクトをすべて正常に始動させます。AUTOMGR が始動する各自動タスクの初期コマンド・リスト AOPIGNIC (CNME6403) は、その自動タスクに対応するプロダクトを始動させ、そのプロダクトを稼働するために必要な他のすべてのプロダクトがアクティブであることを確認します。

各プロダクトの開始コマンド・リストは、プロダクトを開始するコマンドを出し、プロダクトが正常に初期設定されたことを示すメッセージを待ちます。各自動タスクの初期コマンド・リストもタイマー・コマンドを出して、コマンド・リスト AOPMCHEK を定期的に実行し、自動タスクの状況をチェックします。初期設定したいプロダクトの開始コマンド・リストを処理すれば、特定のプロダクトを初期設定することができます。

拡張自動操作サンプル・セットの初期設定部分では、受動モニターを使用しています。各プロダクトの初期設定完了メッセージは、自動化テーブルからプロダクトの自動タスクへ経路指定されます。

モニター: 拡張自動操作サンプル・セットは、プロダクトの初期設定、シャットダウン、およびリカバリーについて、メッセージの受動モニターを使用します。拡張自動操作サンプル・セットは能動モニターを使用して、自動化プロダクトおよび自動タスクの状況を定期的にチェックします。

受動モニター: 基本自動操作サンプル・セットには、システム関連操作の受動モニターの例が含まれています。オペレーターによる簡単で日常的な応答を必要とするメッセージの項目が、自動化テーブルに含まれています。メッセージとそれに対応する自動操作プロシージャがテーブルに追加されると、メッセージに対する応答は自動的に行われるようになり、オペレーターがメッセージに応答する必要はなくなります。

拡張自動操作サンプル・セットは、さらに複雑な状況におけるシステム操作関連の受動モニターの自動操作を実行します。サンプルでは、単一のイベントに対する応答ではなく、単一のシステムにおけるサブシステムおよびアプリケーションの初期設定、リカバリー、またはシャットダウンなどの実行に必要な処理を中心に行います。

受動モニターの例: 自動操作がインストールされていないシステムのオペレーターを想定してください。システムでは CICS が実行していて、この CICS を常に行行させたいとします。なんらかの理由で CICS が異常終了すると、図 204 に示しているメッセージが表示されます。

```
+DFH0606 ABEND xxxx HAS BEEN DETECTED
$HASP395 CICS      ENDED
```

図 204. CICS 異常終了メッセージ

このメッセージが表示された段階でシステム・オペレーターがメッセージに気付かなければ、ユーザーからの苦情により気付くかまたはログをブラウズし障害メッセージを検出して初めてこの障害に気付くことになります。

受動モニターを導入していれば、問題は自動的に訂正されるので、意識する必要はありません。自動化テーブルには、DFH0606 終了メッセージを監視する IF ステートメントがあり、この終了メッセージを受信するとトラップを実行し、コマンド・リスト DFH0606 を呼び出してアプリケーションを再始動させます。例えば、拡張自動操作サンプル・セットとともに提供されている自動化テーブルでは、図 205 に示されているステートメントで DFH0606 アクションを自動化しています。

```
IF MSGID = '+DFH0606' THEN
  EXEC(CMD('DFH0606') ROUTE (ONE AUTOMGR))
  DISPLAY(Y) NETLOG(Y);
```

図 205. 拡張自動操作サンプル・セットにおける受動モニター

障害メッセージが受信されると、このステートメントはそのメッセージを検出し、アプリケーションを再始動させます。システム・オペレーターがアクションを取る必要はありません。

能動モニター: 能動モニターでは、環境の状況を監視するためにシステムおよびネットワークに対する照会処理が実行されます。能動モニターは、タイマー・コマンドを使用して定期的にコマンド・リストを処理するために、拡張自動操作サンプル・セットに組み込まれています。能動モニターは自動タスク AUTOMGR により制御されます。この自動タスクは拡張自動操作サンプル・セットで使用される集中自動操作の自動タスクです。拡張自動操作サンプル・セットとともに提供される能動モニターのサンプルには、次の 2 つのコンポーネントがあります。

- 能動モニター・コマンド・リスト AOPMACT (CNME6439)。このリストは、初期設定後に自動化プロダクトをアクティブのままにしておきます。
- 自動タスク・モニター・コマンド・リスト AOPMCHEK (CNME6440)。このリストは、拡張自動操作サンプル・セットとともに提供されるすべての自動タスクが、アクティブのままであることを確認するために、それらを定期的にチェックします。

AOPMACT は、自動操作として定義した自動化プロダクトに照会コマンドを出し、これらのコンポーネントの現在の状況に関する情報を積極的に送信請求します。コンポーネントにより返される状況情報は、グローバル変数に定義されているコンポーネントの望ましい状態と比較されます。望ましい状態と実際の現行状態とが一致しなければ、担当者にこのことを知らせるメッセージが送られます。

AOPMCHEK は、すべての拡張自動操作サンプル・セットの自動タスクに定期的にメッセージを送信し、応答を待ちます。自動タスクの状況を管理するためには、グローバル変数が使用されます。PPT を使用して自動タスク AUTOMGR をチェックし、AUTOMGR を使用してその他のすべての拡張自動操作サンプル・セットの自動タスクをチェックします。自動タスクが応答しないときには、システム・オペレーターにこのことを知らせるメッセージが送信されます。

自動操作では、自動タスクは強力なツールです。しかし、自動タスクは、通常、関連コンソールを使用せずに無人操作で実行するため、自動タスクに障害が発生したり、自動タスクから応答が戻されないときに見逃される可能性があります。その結果、自動タスクに関する新しい作業はすべてキューに入れられ処理されないか、あるいは自動タスクがログオフされれば、受信されなくなります。

拡張自動操作サンプル・セットは次の方法を使用して、自動タスクの障害を早期に検出するようにしています。

1. 自動タスクの初期コマンド・リストにタイマー・コマンドを指定し、各自動タスクに対し自動タスク・モニター・コマンド・リスト AOPMCHEK (CNME6440) の処理をスケジュールします。拡張自動操作サンプル・セットでは、AOPMCHEK の処理が定期的にスケジュールされます。AOPMCHEK を呼び出す時間間隔は、AOPIVARS に設定されています。他の自動タスクを制御する自動タスク AUTOMGR をモニターするために、AOPMCHEK が PPT のもとで処理します。拡張自動操作サンプル・セットに含まれている他の自動タスクの場合には、AOPMCHEK は自動タスク AUTOMGR のもとで処理されます。自動タスク・オペレーター ID および CHECK の状況が、AOPMCHEK へのパラメーターとして渡されます。
2. AOPMCHEK にストリング CHECK が渡されて処理されると、その自動タスクの状況がすでに CHECK に設定されているかどうかチェックされます。設定されているときには、最後に実行された自動タスクのチェック以後 AOPMCHEK へ肯定応答が送信されていないことを意味しているため、エラー・メッセージがシステム・オペレーターへ送信され、自動タスクが応答していないことを示します。自動タスクの状況がまだ CHECK に設定されていなかったときには、状況は CHECK に設定され、応答を要求するメッセージが自動タスクへ送信されます。
3. 応答を要求するメッセージが自動タスクへ送信されると、自動化テーブルがそのメッセージを代行受信してコマンド・リスト AOPMCHEK を処理するコマンドに変換し、肯定応答を生成します。
4. 自動タスクが応答すれば、AOPMCHEK が処理され、肯定応答が生成されます。肯定応答が生成されたら、AOPMCHEK は自動タスクの状況を CHECK から ACTIVE へ変更します。

能動モニターの例: 本セクションでは、能動モニターが AUTOJES 自動タスクにどのように働くかについて説明します。AOPIGNIC は、AUTOJES の初期コマンド・リストです。AOPIGNIC は、図 206 に示しているコマンド **1** を出します。ここでは、自動タスクのチェック間隔を AOPIVARS に 5 分と設定しています。

```
EXCMD AUTOMGR,EVERY 5,ID=JESCHK,AOPMCHEK AUTOJES CHECK 1  
MSG AUTOJES CHECKING AUTOTASK - AUTOJES 2  
DSI039I MESSAGE FROM AUTOMGR : CHECKING AUTOTASK - AUTOJES 3
```

図 206. AUTOJES 自動タスクの能動モニター

コマンド **1** により、AUTOMGR が 5 分おきに AUTOMGR の状況をチェックします。コマンド・リスト AOPMCHEK が呼び出されると、自動タスクは AUTOJES として認識され、行う処理タイプを CHECK として認識します。

チェックが要求されて AOPMCHEK が処理され、AUTOJES の現行状況が ACTIVE であったときには、状況は CHECK に設定され、図 206 のコマンド **2** が出されます。

その結果、メッセージ DSI039I (図 206 のメッセージ **3**) が生成され、AUTOJES へ送信されます。

このメッセージは、次のように、図 207 の自動化テーブル・ステートメントにより代行受信されます。

```
IF MSGID = 'DSI039I' & TOKEN(9) = 'AUTOJES' THEN  
  EXEC(CMD('EXCMD AUTOMGR,AOPMCHEK AUTOJES ACKNOWLEDGEMENT'))  
  ROUTE (ONE AUTOJES)) SYSLOG(N) NETLOG(N) DISPLAY(N);
```

図 207. メッセージ *DSI039I* の能動モニター

メッセージ *DSI039I* が受信され、その 9 番目のトークンが自動タスク名のときには、図 208 に示されているコマンドが自動タスク *AUTOJES* へ送信されます。

```
EXCMD AUTOMGR,AOPMCHEK AUTOJES ACKNOWLEDGEMENT
```

図 208. *DSI039I* メッセージに応答する自動化テーブルの *EXCMD* コマンド

図 208 のコマンドは、自動タスク *AUTOMGR* のもとでコマンド・リスト *AOPMCHEK* を呼び出します (*AUTOJES* がアクティブであることを示す肯定応答として)。

- *AUTOJES* が応答すれば、コマンドが処理され、コマンド・リスト *AOPMCHEK* により *AUTOJES* の状況が *CHECK* から *ACTIVE* に変更されます。
- *AUTOJES* が応答しなければ、コマンドは処理されず、*AUTOJES* の状況は *CHECK* のままです。

チェックが要求されて *AOPMCHEK* が処理されたが、*AUTOJES* の現行状況が *CHECK* であったときには、自動タスクは最後に送信されたチェックにまだ応答していないことになります。この場合、*AUTOJES* が応答していないことを示すエラー・メッセージがシステム・オペレーターへ送られます。

リカバリー: 拡張自動操作サンプル・セットは、プロダクトの異常終了を示すメッセージを受信した場合の自動化プロダクトのリカバリー方法を示しています。この機能は、異常終了を示すコンソール・メッセージを受信したあとにオペレーターがプロダクトを再始動する操作と同等です。

自動化テーブルが特定のメッセージを受信すると、コマンド・リストの処理によって、障害が発生しているプロダクトの再始動が試みられます。コマンド・リストは、一定時間経過したあとに処理されると、リカバリーの試みが失敗したことを示すタイマー・コマンドを設定します。失敗したことが示されると、リカバリー・コマンド・リストは障害が発生しているプロダクトの開始コマンド・リストを処理します。リカバリーが正常に行われると、リカバリーが失敗したメッセージを出すタイマー・コマンドは除去されます。

拡張自動操作サンプル・セットのリカバリー部分では、受動モニターを使用しています。異常な障害が発生したか、あるいは再始動を試みたが失敗したことを示すメッセージを自動化テーブルが受信すると、リカバリー処理が開始されます。リカバリー機能を向上させるには、以下の操作が必要です。

- 自動化したいこれらのメッセージ用に自動化テーブル・ステートメントを追加して、再始動を要求します。
- 必要に応じてリカバリー・コマンド・リストを追加します。

シャットダウン: 拡張自動操作サンプル・セットでは、システムで自動化プロダクトをシャットダウンする方法を示しています。システムまたは特定のプロダクトの自動シャットダウンは、オペレーターがシステムまたはプロダクトをシャットダウンするのと同じ処理を行います。特定のプロダクトをシャットダウンするには、シャットダウン・コマンドを出します。シャットダウン操作は、プロダクトが正常にシャットダウンされたことを示すメッセージが受信されると完了します。すべての自動化プロダクトをシャットダウンしようとする場合にプロダクト間に依存関係が存在すれば、プロダクトによっては、そのプロダクトが依存しているプロダクトのシャットダウン処理が完了するまでシャットダウンされないものもあります。したがって、シャットダウンを行うには順序があります。

すべての自動化プロダクトをシャットダウンする主コマンド・リストは、AOPSMAN (CNME6412) です。AOPSMAN は、プロダクトが初期設定された順序と逆の順序で停止させていき、プロダクトを正常にシャットダウンします。また、AOPSMAN は従属プロダクトのシャットダウンが完了する前にプロダクトをシャットダウンしないようにします。あるプロダクトにアクティブの従属プロダクトが存在しないことを AOPSMAN が判別すると、そのプロダクトの停止コマンド・リストが呼び出されます。停止コマンド・リストはそのプロダクトに停止コマンドを出し、シャットダウンが完了したことを示すメッセージが出されるのを待ちます。特定のプロダクトをシャットダウンするには、そのプロダクトの停止コマンド・リストを処理してください。

拡張自動操作サンプル・セットのシャットダウン部分では、受動モニターを使用しています。プロダクトのシャットダウン処理が完了したことを示すメッセージが受信されると、そのメッセージは自動化テーブルによりトラップされ、そのプロダクトのシャットダウン自動タスクへ経路指定されます。

オペレーター・インターフェースの拡張: コマンド・リスト AOPUSTAT (CNME6438) は、自動化プロダクトの自動操作情報を表示する VIEW パネルの使用方法を示しています。主パネル CNMS64P0 は、自動操作による状況の変更または状況の検出に合わせて、最新の状況を知らせます。自動化プロダクトの状況に変更があると、CNMS64P0 は動的に最新表示されます。拡張自動操作サンプル・セットに含まれているオペレーター・インターフェースについては、677 ページの『オペレーター・インターフェース・コマンド・リストおよびパネル』で説明します。

拡張自動操作サンプル・セットで使用されるコマンド・リスト

次のセクションでは、システムの初期設定、リカバリー、能動モニター、およびシャットダウンを実行するために、拡張自動操作サンプル・セットでコマンド・リストと自動化テーブルがどのように使用されているかを説明します。

注: 自動操作が行われている最中にプロダクトを手操作で始動または停止するには、該当する初期設定またはシャットダウン・コマンド・リストを使用してください。これにより、自動操作は、プロダクトの望ましい状況と最後に状況が変更された時刻に関する正しい情報を管理することができます。

拡張自動操作サンプル・セットの機能

本セクションでは、拡張自動操作サンプル・セットに組み込まれている各種機能を実行するコマンド・リストとサンプルをリストします。

- 自動化プロダクトごとに、初期自動化グローバル変数の設定値を設定するためのコマンド・リストがあります。自動化グローバル変数には、プロダクトの現行状況およびそのプロダクトを開始する MVS コマンドなどに関する情報が含まれています。

関連コマンド・リスト: AOPIVARS、AOPIGUPD

- 不要なメッセージを抑止し、メッセージに応答し、正しいオペレーターにメッセージを経路指定し、メッセージに응答してコマンド・リストを処理する自動化テーブル項目があります。

関連サンプル: DSITBL11 (CNMS6405)

- 1 つの制御自動タスクが、各自動化プロダクトの自動タスクを活動化します。各プロダクトの専用自動タスクが、そのプロダクトに関連する自動タスクを実行します。

関連コマンド・リスト: AOPIMGIC、AOPIGNIC、AOPIVARS

関連サンプル: CNMS6408、CNMS6409、CNMS6410 (DSIOPF および自動化操作プログラム・プロファイル)

- 初期設定ルーチンは、自動化プロダクトを開始し、初期設定に必要なメッセージに응答し、さらに決められた時間内にプロダクトをアクティブにする MVS コマンドを出します。
 - プロダクトを初期設定し、決められた時間内にプロダクトをアクティブにするコマンド・リストがあります。このコマンド・リストは、プロダクトの望ましい状況を ACTIVE に設定し、プロダクトの現行状況が変更される時刻を設定します。

関連コマンド・リスト: AOPIJES3、AOPIJES2、AOPIVTAM、AOPITSO、AOPIMS、AOPICICS

- プロダクトを初期設定するために自動化テーブルから処理されるコマンド・リストがあります。

関連コマンド・リスト: \$HASP426

- シャットダウン・ルーチンは、自動化プロダクトをシャットダウンし、シャットダウンに必要なメッセージに응答し、さらに決められた時間内にプロダクトを非アクティブにする MVS コマンドを出します。
 - プロダクトをシャットダウンし、決められた時間内にプロダクトを非アクティブにするコマンド・リストがあります。このコマンド・リストは、プロダクトの望ましい状況を INACTIVE に設定し、プロダクトの現行状況が変更される時刻を設定します。

関連コマンド・リスト: AOPSMAN、AOPSJES3、AOPSJES2、AOPSVTAM、AOPSTSO、AOPSTSO2、AOPSIMS、AOPSIMS2、AOPSCICS、AOPSPURG

- プロダクトをシャットダウンするために、コマンド・リストはタイマー・コマンドを使用して処理されます。

関連コマンド・リスト: VTAMTMRZ

- プロダクトをシャットダウンするために、コマンド・リストは自動化テーブルから処理されます。

関連コマンド・リスト: AOPTJRC3、DFS996I、DFS000IB

- リカバリー・ルーチンは、プロダクトの望ましくない状況を示しているメッセージをトラップし、そのプロダクトの再初期設定を試みます。
- プロダクトをリカバリーするために、コマンド・リストは自動化テーブルから処理されます。

関連コマンド・リスト: IAT3714、IAT3708、\$HASP095、\$HASP098、\$HASP085、DFS629I、DFH0606、IKT002I

- プロダクトをリカバリーするために、コマンド・リストはタイマー・コマンドを使用して処理されます。

関連コマンド・リスト: JESTMRA、IMSTMR、CICSTMRA

- 能動モニター・ルーチンは、一定の間隔で自動操作の状況をチェックし、プロダクトおよび自動化操作プログラムの状況をモニターします。
- 自動化プロダクトの現行状況と望ましい状況とが一致しているかどうかを確認するために、定期的にチェックを行う能動モニター・コマンド・リストがあります。そのコマンド・リストは、AUTOMGR 自動タスク初期コマンド・リスト (AOPIMGIC) です。

関連コマンド・リスト: AOPMACT

- 自動化テーブルを使用して拡張自動操作サンプル・セットの自動タスクがアクティブのままであることを確認するために、それらの自動タスクを定期的に監視する自動タスク・モニター・コマンド・リストがあります。そのコマンド・リストは、自動タスク AUTOMGR (AOPIMGIC) の初期コマンド・リスト、およびその他の拡張自動操作サンプル・セットの自動タスク (AOPIGNIC) の初期コマンド・リストにより開始されます。

関連コマンド・リスト: AOPMCHEK

- すべての自動化プロダクトの現行状況、および各自動化プロダクトに関する個別情報を表示するオペレーターが開始するコマンド・リストと関連表示パネルがあります。

関連コマンド・リスト: AOPUSTAT

関連パネル: CNMS64P0 - CNMS64P5

拡張自動操作サンプル・セットでは、以下の標準通知メッセージを用意しています。

- 自動操作では解決できない障害や望ましくない状態が発生すると、メッセージがシステム・オペレーターへ送信されます。
- コマンド・リストへ制御権が渡されると、メッセージがネットワーク・ログへ送信されます。このメッセージをネットワーク・ログに書き込みたくないときには、そのメッセージを送信するステートメントを削除することができます。

拡張自動操作サンプル・セットのコマンド・リストの命名規則

拡張自動操作サンプル・セットのコマンド・リストは、次の汎用規則に従って命名されます。

- 特定のオペレーター・タスクを行うコマンド・リストのうち、メッセージによって起動されないものには、最初の 3 文字に AOP を使用します。4 番目の文字は、コマンド・リストにより実行されるアクションのタイプを表します。

AOPI...

初期設定

AOPM...

事前の対策を講じたモニター

AOPS...

シャットダウン

AOPU...

ユーティリティー

- 自動化テーブルから呼び出されるコマンド・リストには、可能であれば、自動化テーブル内でそれらのコマンド・リストを起動するメッセージと同じ名前を割り当てます。例えば、\$HASP426 は、JES2 メッセージ \$HASP426 を処理するために呼び出されるコマンド・リストです。
- いずれかの拡張自動操作サンプル・セットのコマンド・リストから、NetView のタイマー・コマンドにより呼び出されるコマンド・リストには、通常、そのコマンド・リスト名に TMR という文字が組み込まれています。

初期設定および能動モニター・コマンド・リスト

初期設定および能動モニター・コマンド・リストを下記に示します。

AOPIVARS

主初期設定コマンド・リスト

DSITBL11

自動化テーブルの活動化

AOPIGUPD

共通グローバル変数の設定 (複数回呼び出される)

AUTOMGR

AUTOMGR 自動タスクの活動化

AOPIMGIC

AUTOMGR 初期コマンド・リスト

AUTOJES

JES 自動タスクの活動化

AOPIGNIC

汎用初期コマンド・リスト

AOPIJES2

JES2 の開始とその確認

\$HASP426

JES2 のオプションの指定

AOPIJES3

JES3 の開始とその確認

AOPMCHEK

AUTOJES 自動タスクの定期的チェック

AUTOVTAM

VTAM 自動タスクの活動化

AOPIGNIC

汎用初期コマンド・リスト

AOPIVTAM

VTAM の開始とその確認

AOPMCHEK

AUTOVTAM 自動タスクの定期的チェック

AUTOTSO

TSO 自動タスクの活動化

AOPIGNIC

汎用初期コマンド・リスト

AOPITSO

TSO の開始とその確認

AOPMCHEK

AUTOTSO 自動タスクの定期的チェック

AUTOIMS

IMS 自動タスクの活動化

AOPIGNIC

汎用初期コマンド・リスト

AOPIIMS

IMS の開始とその確認

AOPMCHEK

AUTOIMS 自動タスクの定期的チェック

AUTOCICS

CICS 自動タスクの活動化

AOPIGNIC

汎用初期コマンド・リスト

AOPICICS

CICS の開始とその確認

AOPMCHEK

AUTOCICS 自動タスクの定期的チェック

AOPMACT

プロダクトをアクティブのままにするための能動モニター

AOPMCHEK

AUTOMGR 自動タスクの定期的チェック

AOPIVARS を実行するように、CNMSTYLE メンバーをカスタマイズする必要があります。CNMSTYLE メンバーの詳細については、「*IBM Tivoli NetView for z/OS* インストール:概説」に記載されています。

AOPIVARS:

1. 自動化テーブルを活動化します。
2. 拡張自動操作サンプル・セットで使用するグローバル変数 (例えば、自動化される各プロダクトの開始、シャットダウン、および表示コマンドなど) を設定します。
3. AUTOMGR 自動タスク ID をログオンします。

AOPIMGIC は、AUTOMGR がログオンされるときに初期コマンド・リストです。

1. AOPIMGIC は、各自動化プロダクトの自動タスクを開始します。各自動タスクの名前は、プロダクトの名前 (AOPIVARS に設定されている) に AUTO という頭文字を付けたものです。AOPIVARS に設定されるプロダクト名は、4 文字以下でなければなりません。拡張自動操作サンプル・セットとともに提供される自動タスクには、AUTOJES、AUTOVTAM、AUTOTSO、AUTOIMS、および AUTOCICS があります。各自動タスクでは、初期コマンド・リストに AOPIGNIC を使用します。
2. AOPIMGIC は、タイマー・コマンドを出し、一定間隔で能動モニターのコマンド・リスト AOPMACT を呼び出します。
3. また、AOPIMGIC は、タイマー・コマンドを出して、コマンド・リスト AOPMCHEK を定期的に呼び出し、AUTOMGR をアクティブのままにします。

AOPIGNIC は、汎用初期コマンド・リストであり、呼び出されるプロダクトの開始コマンド・リスト (AOPIJES2、AOPIJES3、AOPIVTAM、AOPITSO、AOPIIMS、または AOPICICS) を呼び出して、自動化プロダクトを開始させます。また、AOPIGNIC は、タイマー・コマンドを出してコマンド・リスト AOPMCHEK を定期的に呼び出し、自動化プロダクトをアクティブのままにします。プロダクトによっては、そのプロダクトを活動化する前に別のプロダクトをアクティブにしなければならない場合があります。AOPIGNIC には、現在活動化しようとしているプロダクトの活動化を試みる前に、その先行プロダクト (存在する場合) が活動化されるのを待つという特別な論理が組み込まれています。

AOPIJES2 は、JES2 環境で操作している場合には、AOPIGNIC により呼び出されます。AOPIJES2 は、JES2 の状況をチェックします。

- JES2 がすでにアクティブのときには、AOPIJES2 は望ましい状況を ACTIVE に設定し、処理を終了します。
- JES2 がアクティブでないときには、AOPIJES2 は JES2 開始コマンドを出して、JES2 が活動化されるまで指定された時間待ちます。その時間内に JES2 がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、JES2 がまだ初期設定されていないことを通知します。

JES2 を開始すると、メッセージ \$HASP426 (オプション指定) が生成され、このメッセージにより、同じ名前が割り当てられているコマンド・リストが起動されます。\$HASP426 コマンド・リストは、未処理の要求に対し、初期設定時に AOPIVARS に設定されたユーザー指定の応答で答えます。

AOPIJES3 は、JES3 環境で操作している場合には、AOPIGNIC により呼び出されます。AOPIJES3 は、JES3 の状況をチェックします。

- JES3 がすでにアクティブのときには、AOPIJES3 は望ましい状況を ACTIVE に設定し、処理を終了します。
- JES3 がアクティブでないときには、AOPIJES3 は JES3 開始コマンドを出して、JES3 が活動化されるまで指定された時間待ちます。その時間内に JES3 がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、JES3 がまだ初期設定されていないことを通知します。

AOPIVTAM は、AOPIGNIC により呼び出され、VTAM の状況をチェックします。

- VTAM がすでにアクティブのときには、AOPIVTAM は望ましい状況を ACTIVE に設定し、処理を終了します。
- VTAM がアクティブでないときには、AOPIVTAM は VTAM 開始コマンドを出し、VTAM が活動化されるまで指定された時間待ちます。その時間内に VTAM がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、VTAM がまだ初期設定されていないことを通知します。

AOPITSO は、AOPIGNIC により呼び出され、TSO の状況をチェックします。

- TSO がすでにアクティブのときには、AOPITSO は望ましい状況を ACTIVE に設定し、処理を終了します。
- TSO がアクティブでないときには、AOPITSO は TSO 開始コマンドを出し、TSO が活動化されるまで指定された時間待ちます。その時間内に TSO がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、TSO がまだ初期設定されていないことを通知します。

AOPIIMS は、AOPIGNIC により呼び出され、IMS の状況をチェックします。

- IMS がすでにアクティブのときには、AOPIIMS は望ましい状況を ACTIVE に設定し、処理を終了します。
- IMS がアクティブでないときには、AOPIIMS は IMS 開始コマンドを出し、IMS が活動化されるまで指定された時間待ちます。その時間内に IMS がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、IMS がまだ初期設定されていないことを通知します。

AOPIIMS は、IMS 用の WTOR に返答して、データ通信を開始することができます。また、コマンド・リスト内の該当する行がコメント解除されていれば、IMS 領域を開始することもできます。

AOPICICS は、AOPIGNIC により呼び出され、CICS の状況をチェックします。

- CICS がすでにアクティブのときには、AOPICICS は望ましい状況を ACTIVE に設定し、処理を終了します。
- CICS がアクティブでないときには、AOPICICS は CICS 開始コマンドを出し、CICS がアクティブになるまで指定された時間待ちます。その時間内に CICS がアクティブにならないときには、システム・オペレーターに警告メッセージが出され、CICS がまだ初期設定されていないことを通知します。

AOPMCHEK は、タイマー・コマンドを介して AOPIMGIC および AOPIGNIC により呼び出され、拡張自動操作サンプル・セットの自動タスクを定期的にチェックし、それらの自動タスクがログオンされていてアクティブであることを確認しま

す。AOPMCHEK は、NetView MSG コマンドを使用して、自動タスクへメッセージを送信します。メッセージは自動化テーブルにより代行受信されます。自動化テーブルでは自動タスクに対するコマンドを処理し、AOPMCHEK を再び呼び出してまだアクティブであることを示す肯定応答を出します。自動タスクが応答しないとき、AOPMCHEK はシステム・オペレーターにメッセージを送信します。

AOPMACT は、AOPMGNIC により呼び出され、自動化プロダクトをモニターして、プロダクトがアクティブのままであることを確認します。プロダクトが非アクティブになると、システム・オペレーターにメッセージが送信されます。

リカバリー・コマンド・リスト

リカバリー用のコマンド・リストは、以下のとおりです。

JES2:

\$HASP095

JES2 異常終了コードの保管

\$HASP098

JES2 異常終了コードに基づいたメッセージの応答

\$HASP085

JES2 の再始動

JESTMRA

JES2 異常終了カウンターのリセット

JES3:

IAT3714

JES3 メモリー・ダンプ要求の応答と、JES3 の再始動

IAT3708

JES3 状況を ACTIVE にリセット、JES3 リカバリーのチェック・タイマーの除去

AOPTJRC3

JES3 異常終了カウンターのリセット

IMS:

DFS629I

IMS 異常終了時の再始動

IMSTMR

15 分後に IMS タイマーをブランクに更新

CICS:

DFH0606

CICS 異常終了時の再始動

CICSTMRA

5 分後に CICS タイマーをブランクに更新

JES2 リカバリーは、メッセージ \$HASP095 により起動されます。

1. 自動化テーブルがメッセージ \$HASP095 を受信すると、コマンド・リスト \$HASP095 が呼び出されます。コマンド・リスト \$HASP095 は、メッセージに含まれている異常終了コードを保管し、異常終了カウンタを増加させます。
2. 次に、メッセージ \$HASP098 に応答して、コマンド・リスト \$HASP098 が自動化テーブルから呼び出されます。このコマンド・リストは、コマンド・リスト \$HASP095 により管理された異常終了コードに従ってメッセージ \$HASP098 に応答します。
 - 異常終了コードが \$PJ2 の場合、オペレーターは異常終了コマンドを出し、JES2 の状況は STOPPING に設定されます。
 - 他の異常終了コードの場合、タイマー・コマンドはコマンド・リスト JESTMRA が 5 分後に呼び出されるようにスケジュールし、JES2 の状況は ABEND に変更されます。JESTMRA が呼び出されると、JES2 の状況終了カウンタが 0 に設定されます。メッセージ \$HASP098 への望ましい応答が、メッセージの応答変数として AOPIVARS に設定されています。
3. コマンド・リスト \$HASP085 が自動化テーブルにより呼び出され、JES2 を再始動します。

JES3 リカバリーは、メッセージ IAT3714 により起動されます。自動化テーブルがメッセージ IAT3714 を受信すると、コマンド・リスト IAT3714 が呼び出されます。IAT3714 は、JES3 の状況を ABEND に更新します。

- JES3 が初めて異常終了した場合、メッセージに対する応答が送信され、異常終了カウンタが設定され、JES3 異常終了が発生したことを示し、メッセージ JES3 RECOVERY FAILED が 5 分後に表示されるようにタイマー・コマンドが出されます。
- 2 回目の JES3 異常終了の場合、または JES3 の応答が無効であった場合は、オペレーターに手操作によるリカバリー処理が必要なことを示すメッセージが送られ、異常終了カウンタをリセットするためにコマンド・リスト AOPTJRC3 が呼び出されます。

メッセージ IAT3708 (JES3 の活動化が完了したことを示す) が自動化テーブルにより代行受信されると、JES3 状況を ACTIVE にリセットし JES3 のメッセージ・タイマーを除去するために、コマンド・リスト IAT3708 が開始され、IAT3714 応答カウンタを 0 にリセットするために AOPTJRC3 が呼び出されます。

CICS リカバリーは、メッセージ DFH0606 により起動されます。自動化テーブルがこのメッセージを受信すると、コマンド・リスト DFH0606 が呼び出されます。DFH0606 は、CICS の状況を ABEND に更新します。

- CICS がこの 5 分間以内に異常終了している場合は、そのことを示すメッセージがオペレーターに送信されます。
- それ以外の場合には、5 分後にコマンド・リスト CICSTMRA を駆動するタイマー・コマンドが出され、次に、CICS 開始コマンドが出されます。CICSTMRA は、異常終了が最新の 5 分間以内に発生したことを示すインディケータをオフに設定します。

IMS リカバリーは、メッセージ DFS629I により起動されます。自動化テーブルがこのメッセージを受信すると、コマンド・リスト DFS629I が呼び出されます。DFS629I は、IMS の状況を ABEND に更新します。

- IMS がこの 15 分間以内に異常終了している場合は、そのことを示すメッセージがオペレーターに送信されます。
- それ以外の場合には、15 分後にコマンド・リスト IMSTMR を呼び出すタイマー・コマンドが出され、次に、IMS 開始コマンドが出されます。IMSTMR は、異常終了が最新の 15 分間以内に発生したことを示すインディケーターをオフに設定します。

シャットダウン・コマンド・リスト

シャットダウン用コマンド・リストは、以下のとおりです。

AOPSMMAIN

すべての自動化プロダクトをシャットダウンするコマンド・リスト

AOPSCICS

CICS のシャットダウンの開始

AOPSIMS

ユーザーに対しログオフするメッセージを出し、AOPSIMS2 を呼び出すタイマーを設定して、IMS のシャットダウンを開始

- DFS000IB - シャットダウンに必要な IMS 領域番号の保管
- AOPSIMS2 - IMS のシャットダウンの完了
- DFS996I - あとで使用するための WTOR の取り出しと保管

AOPSTSO

ユーザーに対しログオフするメッセージを出し、AOPSTSO2 を呼び出すタイマーを設定して、TSO のシャットダウンを開始

- AOPSTSO2 - TSO のシャットダウンの完了

AOPSVTAM

VTAM のシャットダウンを開始し、必要ならば、VTAMTMRZ を呼び出すタイマーを設定

- VTAMTMRZ - 必要な場合、VTAM をシャットダウンするキャンセル・コマンドの発行

AOPSJES2

AOPSPURG を呼び出し、JES2 をシャットダウンするコマンドを出して、JES2 のシャットダウンを開始

- AOPSPURG - すべての装置 (プリンター、穿孔、開始プロダクトなど) のキュー・アクション (このコマンド・リストは、ユーザーの環境に合わせてカスタマイズすることが必要。)

AOPSJES3

JES3 のシャットダウンの開始

- AOPTJRC3 - メッセージ IAT3714 の応答カウンタを 0 にリセット

AOPSMMAIN は、**AOPIVARS** に値が設定されているすべてのプロダクトをシャットダウンします。別のプロダクトをシャットダウンしてからでないとシャットダウンできないプロダクトの場合には、後続のシャットダウン・プロシーチャーを開始する前に、**AOPSMMAIN** が先にシャットダウンしなければならないプロダクトのシャットダウンが完了したかどうかを定期的にチェックします。プロダクトは、

AOPIVARS にリストされている順序と逆の順序でシャットダウンされていきます。
AOPSMAN は、次のコマンド・リストを呼び出します。

- CICS を停止させる AOPSCICS
- IMS を停止させる AOPSIMS
- TSO を停止させる AOPSTSO
- VTAM を停止させる AOPSVTAM
- JES2 を停止させる AOPSJES2
- JES3 を停止させる AOPSJES3

AOPSCICS は CICS の状況をチェックします。

- CICS がすでに終了している場合、AOPSCICS は望ましい状況を DOWN に設定し、処理を終了します。
- CICS が終了していない場合、CICS を停止するコマンドが出されます。シャットダウンが決められた時間内に完了しなければ、システム・オペレーターにメッセージが出されます。

AOPSIMS が呼び出されると、IMS は 2 分以内にシャットダウンされるので、ユーザーに対しログオフするメッセージをブロードキャストし、コマンド・リスト AOPSIMS2 を 2 分後に呼び出すためにタイマー・コマンドを出します。

- AOPSIMS は、IMS の望ましい状況を DOWN に更新します。また、現行状況を示す変数をチェックして、その変数がすでに DOWN に設定されていれば、処理を終了するか、あるいはその変数を STOPPING に設定します。
- AOPSIMS はメッセージをブロードキャストし、ユーザーに対し 2 分以内にログオフするように警告します。
- AOPSIMS はアクティブの領域を表示し、自動化テーブルから出されるコマンド・リスト DFS000IB は IMS 領域番号を保管します。
- 次に、AOPSIMS は 2 分後に AOPSIMS2 を呼び出すためにタイマー・コマンドを出します。AOPSIMS2 は、IMS 領域、データ通信、および IMS を停止するコマンドを出します。
 - シャットダウンが正常に完了したことを示すメッセージが決められた時間内に受信されると、AOPSIMS2 は現行状況を示す変数を DOWN に更新し、自動化テーブルから出されるコマンド・リスト DSF996I は、IMS WTOR をあとで使用するために取り出します。
 - 決められた時間内にメッセージが受信されないと、AOPSIMS2 はシステム・オペレーターにメッセージを出し問題を報告します。

AOPSTSO は、ユーザーに TSO がシャットダウンされることを示すメッセージを出します。次に、AOPSTSO は、コマンド・リスト AOPSTSO2 を呼び出すためにタイマー・コマンドを出します。AOPIVARS は、AOPSTSO2 を呼び出す前に AOPSTSI の待ち時間を指定します。その時間が経過すると、AOPSTSO2 は処理を開始し、TSO の状況をチェックします。

- TSO が終了している場合、AOPSTSO2 は望ましい状況を DOWN に設定して処理を終了します。

- TSO がまだ終了していない場合、TSO を停止させるコマンドを出します。シャットダウンが決められた時間内に完了しなければ、システム・オペレーターにメッセージが出されます。

AOPSVTAM は、VTAM の状況をチェックします。

- VTAM が終了している場合、AOPSVTAM は望ましい状況を DOWN に設定し、処理を終了させます。
- VTAM が終了していない場合、AOPSVTAM は、3 分後にコマンド・リスト VTAMTMRZ を呼び出すタイマー・コマンドを出し、VTAM を停止するコマンドを出します。VTAM が正常にシャットダウンされると、VTAMTMRZ を呼び出すタイマー・コマンドは除去されます。

VTAMTMRZ は VTAM の状況をチェックし、終了していれば、処理を終了します。VTAM が終了していなければ、VTAMTMRZ は VTAM へ取り消し要求を出します。シャットダウンが決められた時間内に完了しなければ、システム・オペレーターにメッセージが出されます。

AOPSJES2 は、JES2 の状況をチェックします。

- JES2 がすでに終了している場合、AOPSJES2 は望ましい状況を DOWN に設定し、処理を終了します。
- JES2 が終了していない場合、コマンド・リスト AOPSPURG を出します。AOPSPURG は、JES2 に対しすべての装置のキュー・アクションを要求します。

すべての装置のキュー・アクションが終わると、JES2 を停止するコマンドが出されます。シャットダウンが決められた時間内に完了しなければ、システム・オペレーターにメッセージが出されます。

AOPSJES3 は、JES3 の状況をチェックします。

- JES3 がすでに終了している場合、AOPSJES3 は望ましい状況を DOWN に設定し、処理を終了します。
- JES3 が終了していない場合、AOPSJES3 は JES3 をシャットダウンするコマンドを出します。シャットダウンが決められた時間内に完了しなければ、システム・オペレーターにメッセージが出されます。

オペレーター・インターフェース・コマンド・リストおよびパネル

拡張自動操作サンプル・セットには、VIEW パネルとともに使用できるユーティリティ・コマンド・リストがあり、自動化システムの操作およびモニターに使用することができます。操作インターフェース・コマンド・リストおよびパネルは、AOPIVARS に定義されているプロダクトだけを表示します。

注: 自動化プロダクトが 10 個以上になったら、コマンド・リスト AOPUSTAT およびパネル CNMS64P0 を変更しなければなりません。

自動操作表示コマンド・リスト: AOPUSTAT はパネルを表示し、パネルには、自動化プロダクトの現行状況と最後に更新された時刻が示されます。

- 表示されたパネル (CNMS64P0) 上で特定のプロダクトを選択すると、そのプロダクトの状況に関する詳細な情報 (そのプロダクトの開始および停止に使用されるコマンドおよびコマンド・リストなど) が表示されます。

- 特定のプロダクト情報パネル (CNMS64P1) から、AOPIVARS に設定されているそのプロダクトのメッセージ応答変数を表示することができます。各パネルにはヘルプ・パネルが対応していて、関連パネルにより提供される機能について説明しています。

自動操作表示パネル: すべての自動操作表示パネルは、コマンド・リスト AOPUSTAT から呼び出されます。このコマンド・リストは、同義語 AOSTAT を使用して呼び出すことができます。

CNMS64P0 は、主表示パネルです。CNMS64P0 には、すべての自動化プロダクトとその現行状況、開始または停止の試行ヒストリーをもとにしたプロダクトの状況 (DESIRED STATE)、能動モニターにより状況が最後にチェックされた時刻、および各プロダクトの状況が最後に変更された時刻が表示されます。また、CNMS64P0 上で特定のプロダクトの番号を入力し、そのプロダクトに関する詳細な情報をパネル CNMS64P1 に表示することができます。CNMS64P3 は、CNMS64P0 のヘルプ・パネルです。図 209 は、パネル CNMS64P0 の例です。

注: パネル CNMS64P0 は、現行状況、望ましい状況、または状況が変更された時刻の変数が増えるたびに自動的に更新されるので、常に最新の状況情報が表示されています。

CNMS64P0		OPER2		12/21/10	15:53
	PRODUCT / SUBSYSTEM	CURRENT STATUS	DESIRED STATE	CHECKED AT	STATUS CHANGED AT
	=====	=====	=====	=====	=====
1	JES	ACTIVE	ACTIVE	15:51	15:28
2	VTAM	ACTIVE	ACTIVE	15:51	15:29
3	TSO	ACTIVE	ACTIVE	15:51	15:30
4	CICS	ACTIVE	ACTIVE	15:51	15:32
5	IMS	ACTIVE	ACTIVE	15:51	15:32
Select a number and press ENTER for product specific information					
OR press ENTER to refresh this panel.					
Action====>					
PF1= Help PF2= End PF3= Return					
PF6= Roll					

図 209. CNMS64P0 表示の例

CNMS64P1 は、汎用パネルです。このパネルは、パネル CNMS64P0 上で、詳細な情報が必要なプロダクトが選択されると呼び出されます。CNMS64P1 には、パネル CNMS64P0 に表示されていた情報、初期設定、シャットダウン、およびプロダクトの状況表示に使用されるコマンドの変数値、およびプロダクトの初期設定とシャットダウンに使用されるコマンド・リストが表示されます。このパネル上で **R** を指定すると、このプロダクトに関連するメッセージ応答変数の値が表示されます。その情報はパネル CNMS64P2 に表示されます。CNMS64P4 は、CNMS64P1 のヘルプ・パネルです。679 ページの図 210 は、TSO が開始された直後に表示されたパネル CNMS64P1 です。TSO がアクティブになると、STATUS フィールドが ACTIVE から STARTING へ変更され、プロダクトがアクティブになった時刻を示すフィールドにその時刻が表示されます。

```
CNMS64P1                OPER2                12/21/10    15:30
                        TSO
BECAME ACTIVE AT:      WENT INACTIVE AT:
                        TSO      AUTOMATION VALUES
                        -----
CURRENT STATUS      =  STARTING
DESIRED STATUS      =  ACTIVE
START      =  S TSO
STOP      =  P TSO
DISPLAY      =  D J,TSO
START COMMAND LIST -  AOPITSO
STOP COMMAND LIST  -  AOPSTSO
Press R and ENTER to display message responses for TSO
OR press ENTER to refresh this panel.
Action====>
      PF1= Help   PF2= End   PF3= Return
      PF6= Roll
```

図 210. CNMS64P1 表示の例

CNMS64P2 には、パネル **CNMS64P1** で現在調査の対象になっているプロダクトのメッセージ応答変数の値が表示されます。メッセージ応答変数は、コマンド・リスト **AOPIVARS** で設定されます。**CNMS64P5** は、**CNMS64P2** のヘルプ・パネルです。**CNMS64P2** は、情報の表示専用パネルです。したがって、変数の値を変更するためにパネルの数字を変更することはできません。

各種サンプル

上記で説明した 6 つの **VIEW** パネル以外にも、拡張自動操作サンプル・セットには次の 10 種類のサンプルが用意されています。

CNMS6401

MVS コマンド **verb** のコマンド・モデル・ステートメント。これらのステートメントは、**NetView** オペレーター端末から出すことができます。拡張自動操作サンプル・セットを正常に機能させるために、これらのステートメントが必須というわけではありません。

CNMS6402

JES2 コマンド **verb** のコマンド・モデル・ステートメント。これらのステートメントは、**NetView** オペレーター端末から出すことができます。拡張自動操作サンプル・セットを正常に機能させるために、これらのステートメントが必須というわけではありません。

CNMS6403

JES3 コマンド **verb** のコマンド・モデル・ステートメント。これらのステートメントは、**NetView** オペレーター端末から出すことができます。拡張自動操作サンプル・セットを正常に機能させるために、これらのステートメントが必須というわけではありません。

CNMS6404

拡張自動操作サンプル・セットのコマンド・リストのコマンド定義ステートメント。これらのステートメントを既存の **CNMCMD** に追加してください。

CNMS6405

拡張自動操作サンプル・セットの自動化テーブル項目。サンプルを活動化する前に、これらの項目を既存の自動化テーブルへ追加してください。コマンド・リスト **AOPIVARS** (**CNME6400**) は、これらの項目をテーブル

DSITBL11 でアクティブにします。これらの項目を異なった名前のテーブルに書き込むときには、AOPIVARS を修正して使用する名前を反映させてください。

CNMS6406

NetView コマンド・リスト言語で作成されたコマンド・リストから、コメントを取り除いたコピーを作成するための TSO コマンド・リスト。

CNMS6406 は、コメントを示すために大部分の Tivoli サンプルで使用されている形式に一致する行を削除します。つまり、EXEC は、1 桁目にアスタリスクが指定されていて 2 桁目にスペースが指定されている行、または 20 個のアスタリスクが連続して現れる行を削除します。CNMS6406 を実行する前に、コピーしようとしているファイルが、この形式に一致していることを確認してください。また、コメント以外の行が削除基準に一致していないことを確認してください。

CNMS6408

拡張自動操作サンプル・セットに必要な自動化操作プログラムの定義。既存の DSIOPF 定義にこれらの定義を追加してください。

CNMS6409

拡張自動操作サンプル・セットの自動タスク AUTOMGR の自動タスク・プロファイル。

CNMS6410

AUTOMGR を除くすべての拡張自動操作サンプル・セットの自動タスクに対する、汎用自動タスク・プロファイル。

拡張自動操作サンプル・セットの使用準備

拡張自動操作サンプル・セットを使用せずにすでに自動操作を行っている場合には、拡張自動操作サンプル・セットを既存の自動操作に組み込むことができます。しかし、拡張自動操作サンプル・セットにより提供されている自動操作機能を十分に使いこなすには、まず、拡張自動操作サンプル・セットを開始して、サンプル・セットに必要な修正を加える必要があります。拡張自動操作サンプル・セットを使用して NetView プログラムを起動する前に、以下のセクションで説明している NetView 定義を変更する必要があります。

NetView 初期設定の準備

NetView プログラムは、システム内のその他のサブシステムや適用業務 (JES と VTAM を含む) より先に開始することができます。拡張自動操作サンプル・セットでは、NetView プログラムがオペレーティング・システムにより初期設定され、他のすべてのアプリケーションおよびサブシステムはそれ以降の NetView プログラムにより初期設定されることを想定します。しかし、これは自動操作の必須条件ではありません。

JES より先に NetView プログラムを開始する: JES は、NetView プログラムと同時に、または NetView 自動操作機能がアクティブになってから開始することができます。後者の方法が有利な点は、初期設定時に出される JES メッセージを自動化テーブルで処理することができるため、JES 始動中の障害発生時にシステムで実行されている処理が分かるという点です。この情報により、初期設定処理を通常通り継

続するか、あるいはなんらかのリカバリー処理が必要かどうかを決定することができます。また、この情報は、初期設定時のリカバリー処理で重要な要素となる可能性があります。

JES より先に NetView プログラムを開始する場合には、特別なセットアップが必要です。このセットアップには、次のことが含まれます。

- NetView プロシーチャーは、SUB=MSTR オペランドを指定した START コマンドで開始しなければなりません。

例えば、NetView プログラムで提供されるサンプル・プロシーチャーを使用している場合には、SYS1.PARMLIB の COMMNDxx メンバーの最後のステートメントとして次のステートメントを追加します。

```
S CNMPSSI,SUB=MSTR  
S CNMPROC,SUB=MSTR
```

最初のステートメントは、NetView サブシステム・アドレス・スペースを開始します。次のステートメントは、NetView アプリケーションを開始します。どちらを先に指定しても構いません。

- NetView プロシーチャーは、JES がサポートするユーザー・ライブラリーではなく、SYS1.PROCLIB に保管しなければなりません。
- NetView プロシーチャーには、1 つのジョブ・ステップしか指定できません。

注：以下の操作を実行すれば、複数のジョブ・ステップを指定することもできます。

- MVS LINK マクロ・インターフェースを介して各ステップからプログラムを呼び出すユーザー作成ドライバを作成してください。
- 各ステップの DD ステートメントを 1 つのグループにまとめてください。
- ジョブの EXEC ステートメントにプログラムを指定してください。
- すべてのデータ・セットは VOL=SER で指定するか、またはマスター・カタログに保管しなければなりません。
- SYSIN、SYSOUT または VIO データ・セットを参照することはできません。
- NetView プログラムがアクティブになるまで JES ステートメントを開始しないようにするために、NOSTART パラメーターを指定した JES ステートメントを SYS1.PARMLIB の IEFSSNxx メンバーに指定しなければなりません。
- SYS1.PARMLIB の IEFSSNxx メンバーに NETVIEW ステートメントの前に、JES ステートメントをコーディングしなければなりません。
- SUB=MSTR を指定して NetView プログラムを開始すると、DSIRQJOB タスクが NetView ジョブのジョブ ID を要求するときに、デフォルトの解釈として JES ジョブ・ログが割り当てられます。JES ジョブ・ログが必要ないときには、DSICTMOD の JES ジョブ・ログ定数を変更してください。
- DSIRQJOB がジョブ ID を JES から受け取った後に、JES が異常終了、または DSIRQJOB にジョブ ID の解放を通知せずに終了した場合、DSIRQJOB および NetView プログラムは、JES がもう一度アクティブになるまで停止できません。JES が異常終了したか、コマンド行からユーザーによって終了した場合、ユーザーは NetView MVS コマンド改訂を使用してこの問題を回避できます。この回避策について詳しくは、「IBM Tivoli NetView for z/OSインストール:概説」で JES より先に NetView プログラムを開始する方法を参照してください。

VTAM より先に NetView プログラムを開始する: NetView プログラムを使用して、特定の基準に従って VTAM またはその他のサブタスクを開始します。例えば、VTAM リソースを開始させる順序および開始させるリソース数は、システムの IPL 実行時点によります。指定されたリソースを一日の時間による順序でアクティブにするために、NetView プログラムを使用して (IPL 時点に従って) さまざまなコマンド・リストを実行することができます。

VTAM がアクティブになる前に NetView を起動する場合には、VTAM が最初是非アクティブでなければならない NetView タスクを定義してください。(VTAM 従属タスクは、NetView アプリケーションを定義するために使用される VTAM サンプル A01APPLS (CNMS0013) に示されています。) これを行うには、CNMSTYLE メンバーの TASK ステートメントに INIT=N とコーディングします。これにより、タスクが VTAM への接続待ちの間に不要なメッセージを生成することが防止されます。これらのタスクは、自動化テーブルが IST020I (VTAM 初期設定完了) メッセージを代行受信すると呼び出される NetView プログラムのコマンド・リストにより、開始されます。

システム許可機能製品より先に NetView プログラムを開始する: システム許可機能 (SAF) プロダクト (例えば、RACF (リソース・アクセス管理機能)) を NetView プロダクトと一緒に使用する場合に特有の問題があります。任意のタイプのセキュリティの SAF プロダクトを使用するときには、NetView プログラムを開始する前に、SAF プロダクトを開始する必要があります。そのため NetView プログラムを開始する前に、あらかじめ SAF プロダクトと必要な SAF クラスを開始してください。

拡張自動操作サンプル・セットの修正

拡張自動操作サンプル・セットを機能させるために、AOPIVARS (CNME6400) を実行する必要があります。これは、自動化テーブルをロードし、拡張自動操作サンプル・セットで使用されるグローバル変数に初期値を設定し、自動タスク AUTOMGR をアクティブにします。AOPIVARS は、操作環境に合わせて次のようにカスタマイズしてください。

- 自動化しないプログラムのすべてのグローバル変数の値の更新値を削除してください。例えば、AOPIVARS には JES2 および JES3 の両方の定義が含まれていますが、そのうちの少なくとも 1 つを削除してください。
- 開始、停止、およびプロダクトの状況の表示に使用されるコマンドに割り当てられているすべての変数の値を、オペレーターがインストール先で使用するコマンドおよび名前と一致させてください。
- さまざまな自動操作処理のために AOPIVARS に設定されている時間は、操作環境に合わせた最適値に変更することができます。例えば、AOPIVARS に設定されている変数は、自動操作のための自動タスクの状況をモニターする頻度を自動操作に通知します。

JES2 を自動化する場合には、すべての JES2 装置のキュー・アクションを実行できるように、AOPSPURG コマンド・リストをカスタマイズする必要があります。

自動タスクの定義

拡張自動操作サンプル・セットは自動タスクを使用して、自動化環境でコマンド・リストを実行します。拡張自動操作サンプル・セットには、オペレーター・プロフ

ファイル (CNMS6408) が提供されています。この定義は、DSIOPF に入っている既存のオペレーター・プロファイル定義に追加してください。拡張自動操作サンプル・セットで使用される各自動タスク (AUTOMGR、AUTOJES、AUTOVTAM、AUTOTSO、AUTOIMS、および AUTOCICS) ごとにオペレーター定義項目が用意されています。拡張自動操作サンプル・セットによりサポートされているプロダクトを 1 つも自動化しない場合には、オペレーター定義を DSIOPF に追加する必要はありません。

各 DSIOPF 項目は、自動タスクのオペレーター・プロファイル (DSIPRF に保管されている) を指定することができます。このプロファイルは、自動タスクの初期コマンド・リストを順番に指定することができます。自動タスクには、2 つの初期コマンド・リストが拡張自動操作サンプル・セットとともに提供されています。

- AOPIMGIC (CNME6402) は、自動タスク AUTOMGR の初期コマンド・リストです。
- AOPIGNIC (CNME6403) は、その他の自動タスクに使用される汎用初期コマンド・リストです。

AUTOMGR は、拡張自動操作サンプル・セットとともにサンプルとして提供され定義されている自動タスクです。AUTOMGR に指定されている定義を、自動タスク定義時のモデルとして使用できます。図 211 には、AUTOMGR 用の DSIOPF 項目を示します。この項目は、拡張自動操作サンプル・セット・サンプル CNMS6408 の一部分です。

```
*****
*      (C) COPYRIGHT IBM CORP. 2010                      *
*      LAST CHANGE:      07/12/10                        *
*      DESCRIPTION: NETVIEW OPERATOR DEFINITIONS/PASSWORDS *
*                  FOR AUTOMATED OPERATORS                *
*      CNMS6408 CHANGED ACTIVITY:                          *
*      CHANGE CODE  DATE      DESCRIPTION                 *
*      -----      -
*****
* THESE OPERATOR DEFINITIONS SHOULD BE ADDED TO YOUR EXISTING *
*      DSIOPF OPERATOR DEFINITIONS.                          *
*****
:
:
AUTOMGR  OPERATOR  PASSWORD=AUTOMGR
        PROFILE   DSIPROFM
:
:
```

図 211. CNMS6408 の抜粋 (AUTOMGR オペレーター定義)

684 ページの図 212 には、拡張自動操作サンプル・セットの一部分である AUTOMGR 自動タスクのオペレーター・プロファイルを示します。

```

*****
* (C) COPYRIGHT IBM CORP. 2010 *
* ALL RIGHTS RESERVED. *
* US GOVERNMENT USERS RESTRICTED RIGHTS - USE, DUPLICATION *
* OR DISCLOSURE RESTRICTED BY GSA ADP SCHEDULE CONTRACT *
* WITH IBM CORPORATION. *
* IEBCOPY SELECT MEMBER=((CNMS6409,DSIPROFM,R)) *
* LAST CHANGE: 07/12/10 *
* DESCRIPTION: AUTOMATED OPERATOR (AUTOMGR) PROFILE DEFINITION *
* CNMS6409 CHANGED ACTIVITY: *
* CHANGE CODE DATE DESCRIPTION *
* -----*
*****
* MINIMAL AUTOMATED OPERATOR PROFILE STATEMENTS FOR AUTOMGR *
* STARTED WITH AUTOTASK COMMAND TO RUN AS AN UNATTENDED OPERATOR. *
* AUTOMGR IS THE AUTOMATED OPERATOR THAT MANAGES THE OTHER *
* AUTOMATED OPERATORS USED IN THE CONSOLE AUTOMATION SAMPLE SET. *
*****
DSIPROFM PROFILE IC=AOPIMGIC
AUTH MSGRECV=NO,CTL=GLOBAL
END

```

図 212. CNMS6409 の抜粋 (DSIPROFM オペレーター・プロファイル)

AUTOMGR が DSIOPF に定義されている限り、AUTOMGR は NetView オペレーター ID として有効です。AUTOMGR が開始されると、AUTOMGR に関連のあるオペレーター・プロファイル上で定義されている初期コマンド・リストが AOPIMGIC であるため、AOPIMGIC コマンド・リストが実行されます。

コマンド定義ステートメントの定義

CNMCMD では、コマンド・リストのコマンド同義語を定義します。

拡張自動操作サンプル・セットには、拡張自動操作サンプル・セットを実行させるために既存の CNMCMD に追加しなければならない項目が含まれています。これらの項目は、拡張自動操作サンプル・セットの CNMS6404 に含まれています。項目は、拡張自動操作サンプル・セットのコマンド・リストごとに用意されています。これらの項目に定義されている同義語は拡張自動操作サンプル・セット全体で使用されるため、CNMCMD メンバーにこれらの項目が組み込まれるまで、コマンド・リストは実行しません。

注: コマンドの同義語がユーザーのシステムですでに定義されているコマンドの同義語と矛盾する場合には、拡張自動操作サンプル・セットおよびその同義語を参照している拡張自動操作サンプル・セットのすべてのコマンド・リストの中で、その同義語を変更する必要があります。

上記の必須項目以外にも、次の 3 種類のコマンド同義語のセットが拡張自動操作サンプル・セットとともに提供されています。

- CNMS6401 には、MVS コマンド verb のコマンド定義サンプルが含まれています。
- CNMS6402 には、JES2 コマンド verb のコマンド定義サンプルが含まれています。
- CNMS6403 には、JES3 コマンド verb のコマンド定義サンプルが含まれています。

これらのコマンドは操作上の便宜を考慮して提供されていますが、拡張自動操作サンプル・セットを使用する際に必須というわけではありません。

自動化テーブルの修正

拡張自動操作サンプル・セットには、拡張自動操作サンプル・セットを使用する際に必要な自動化テーブル項目が含まれています。自動化したい分野に関する自動化テーブル項目と、プロダクトとともに提供されている自動化テーブル項目がともに、実動で使用される自動化テーブルに含まれていることを確認してください。このためには、次の操作を実行します。

- 現在、実稼働環境で使用している自動化テーブルがないときには、提供されているテーブル項目を DSITBL11 (CNMS6405) から NetView プログラムで提供されている自動化テーブル・サンプル (DSITBL01) にコピーするか、%INCLUDE ステートメントを使用して DSITBL01 に DSITBL11 を組み込むことができます。
- 現在、実動環境の自動化テーブルを使用しているときには、その中に DSITBL11 (CNMS6405) をコピーするか、または %INCLUDE を使用して DSITBL11 を組み込むことができます。これによりテーブル内でメッセージ番号が重複した場合には、これらのメッセージの NetView 自動化テーブル項目を 1 つのステートメントにしてください。ステートメントを 1 つにせず、ステートメントに CONTINUE アクションも指定されていないと、重複ステートメントの最初のステートメントしか実行されません。
- 自動操作に使用する自動化テーブルの名前が DSITBL11 以外の場合には、コマンド・リスト AOPIVARS (CNME6400) を更新し、DSITBL11 の代わりに新しいテーブル名をロードしてください。
- 特定のプロダクト (例えば、TSO) を自動化しない場合には、そのプロダクトに関する DSITBL11 (CNMS6405) 項目を使用中の実動自動化テーブルにコピーしないでください。コピーすると、オペレーターがそのタスクを手操作で開始しようとするときに、そのコマンドの関連メッセージが端末に表示されません。
- NetView プログラムの実行中に自動化テーブルを変更するときには、AUTOTBL MEMBER=*automem* コマンドを使用してテーブルをリサイクルしなければなりません。ここで、*automem* は自動化テーブルが含まれているメンバー名です。

自動化テーブルは複数定義できますので、拡張自動操作サンプル・セットを実行する場合と実行しない場合、それぞれに別の自動化テーブルを用意することもできます。コマンド・リスト AOPIVARS からロードされる自動化テーブルの名前は、拡張自動操作サンプル・セットの項目が含まれているテーブルの名前と一致していなければなりません。

拡張自動操作サンプル・セットのカスタマイズ

独自の自動操作を確立するために拡張自動操作サンプル・セットを使用する場合、拡張自動操作サンプル・セットのサンプルは、操作環境に合わせて簡単にカスタマイズすることができます。機能を追加して、プロダクト、オペレーティング・システム、コマンド・リスト、または自動化テーブル・ステートメントをサポートすることもできます。拡張自動操作サンプル・セットの基本的な設計方法を理解すると、導入先で必要な任意の変更を行うことができます。以下のセクションでは、拡張自動操作サンプル・セットへの追加方法を紹介します。

グローバル変数のカスタマイズ

サンプル・セットに機能を追加する場合、その機能では、サンプル・セットが使用しているグローバル変数の規則と同じ規則を使用してください。自動化される各プロダクトの情報は、タスクおよび共通グローバル変数を使用して、コマンド・リスト間に共通して使用されます。各プロダクトには、プロダクトの関連変数と、プロダクトの自動化情報が保管される変数（プロダクトの開始コマンドおよび停止コマンドなど）があります。これらの変数は自動操作初期設定時に初期設定され（コマンド・リスト AOPIVARS）、機能（初期設定、リカバリー、モニター、ユーティリティ、シャットダウン）を実行するコマンド・リストは、そのプロダクトに必要な変数にアクセスすることができます。

本セクションでは、複合グローバル変数が共通グローバル変数からどのように作成されるかについて説明し、また、複合グローバル変数の使用例を示します。

複合グローバル変数の作成と命名： 使用されるグローバル変数の中には、他の値をいくつか組み合わせて作成されるものがあります。例えば、プロダクト開始コマンドが含まれる共通グローバル変数には、そのコマンドの自動化対象システムを指定するリソース共通接頭部（RCP）、変数タイプ（START）を指定する値、および変数が参照するプロダクトのインディケータが必要です。カスタマイズ作業を容易にするために、複合グローバル変数の作成時にはパターンに従います。

自動化されるシステムは、リソース共通接頭部（RCP）グローバル変数に保存されます。RCP（REXX）または &RCP（NetView コマンド・リスト言語）は、NetView プログラムが実行しているシステムを示す変数です。RCP または &RCP 変数の値は最大 5 文字で、AOPIVARS が実行しているシステムのドメイン ID に AOPIVARS で自動的に設定されます。

機能情報は、3 文字の変数で示されます。例えば、CST は、プロダクトの START 情報を示す変数です。拡張自動操作サンプル・セットで使用されている機能タイプは、次のとおりです。

CST	プロダクトの開始 (START) コマンド
CSP	プロダクトの停止 (STOP) コマンド
CSA	プロダクトの現行の状況 (STATUS)
CDS	プロダクトの望ましい状態 (STATE)
CDP	プロダクトの状況を表示 (DISPLAY) するコマンド
CT1	プロダクトが最後にアクティブになった時間
CT2	プロダクトが最後に非アクティブになった時間
STC	開始 (START) コマンド・リスト
SPC	停止 (STOP) コマンド・リスト
PDD	プロダクトの従属関係
WTI	初期設定完了までの待ち秒数
WTS	シャットダウン完了までの待ち秒数
CTI	能動モニターより最後に状況がチェックされた時間
CNA	プロダクトの名前

CO n メッセージ応答 (n は任意の数値)

プロダクト ID は 3 文字の変数で、複合グローバル変数の作成時にも使用されます。拡張自動操作サンプル・セットで使用されるプロダクト ID を、表 38 にリストします。

表 38. 拡張自動操作サンプル・セットのプロダクト ID

プロダクト	REXX 変数	NetView コマンド・リスト言語の変数	値
JES2 または JES3	JES	&JES	JES
VTAM	VTAM	&VTAM	VTM
TSO	TSO	&TSO	TSO
CICS	CICS	&CICS	CCS
IMS	IMS	&IMS	IMS

主要な自動操作初期設定コマンド・リストは、AOPIVARS です。AOPIVARS は、各プロダクトの情報が含まれるグローバル変数を作成します。AOPIVARS は、コマンド・リスト AOPIGUPD を呼び出して共通グローバル変数を作成します。この AOPIGUPD は、グローバル変数を作成しコマンド・リストへの入力に従ってその変数の値を設定します。AOPIVARS では、自動化されるシステムのリソース共通接頭部および自動化されるすべてのプロダクトのプロダクト ID は、共通グローバル変数に設定されます。

複合グローバル変数の使用例: 本セクションでは、複合グローバル変数が作成され、自動操作で使用される例を、ステップごとに説明します。この例では、NetView コマンド・リスト言語の変数を使用していますが、作成処理は REXX でも同じです。また、この例では、TSO 開始コマンドを保管する変数が使用されています。システム ID (&RCP) には CNM01 を想定してください。システム ID は、システムごとに異なります。開始変数を作成しているので、機能 ID は CST です。プロダクト変数 (3 文字のプロダクト ID) は &TSO で、その値は TSO です。

&RCP および &TSO を定義するには、図 213 の情報を使用することもできます。

```
&CGLOBAL RCP TSO
&RCP = CNM01
&TSO = TSO
```

図 213. TSO 開始変数の変数の定義

これで、図 214 に示すように、開始 TSO 変数を作成して値を設定することができます。

```
&START = &CONCAT &RCP CST&TSO
```

図 214. TSO 開始変数の作成

NetView コマンド・リスト言語の &CONCAT 機能は、&RCP および CST&TSO の値を連結します。変数 &TSO の値が代わりに使用され、CST と組み合わせられます。この例では、&START には、下記の方法で派生した値 CNM01CSTTSO が割り当てられます。

```
&START = リソース共通接頭部 + 機能 + プロダクト
&START = &RCP + CST + &TSO
&START = CNM01 + CST + TSO
&START = CNM01CSTTSO
```

このコマンド・リストが実行されると、変数 `&START` の値は `CNM01CSTTSO` になります。 `&START` は図 215 に示しているステートメントにより、共通グローバル変数として定義することができます。

```
&CGLOBAL &START
```

図 215. 共通グローバル変数として `&START` を定義するステートメント

`&START` を共通グローバル変数として定義すると、実際には変数が共通グローバル変数として定義される前に、`&START` 変数の置き換えが行われます。これは、`&CGLOBAL` 機能では定義される変数にアンパーサンドが不要だからです。`RCP` を共通グローバル変数として定義するステートメントには、`RCP` の前にアンパーサンドが付いていません。`&CGLOBAL RCP` は、変数 `&RCP` を共通グローバル変数として定義しています。したがって、上記の例では、`CNM01CSTTSO` が `&CGLOBAL` ステートメントの `&START` の代わりに使用され、その結果 `CNM01CSTTSO` は共通グローバル変数として定義されます。

`&CNM01CSTTSO = 'S TSO'` のように、値を直接更新することで変数に値を割り当てることができますが、変数ごとにシステム ID、機能、およびプロダクトを覚えていなければならないため、この方法は面倒です。この方法を使用する代わりに、変数を間接的に更新することができます。 `&START` は、すでに `CNM01CSTTSO` として定義されています。これを使用して、図 216 に示しているように変数の値を変更します。

```
&&START = 'S TSO'
```

図 216. 共通グローバル変数の間接的な更新

このコマンド・リスト処理時には、`&START` の値がステートメントに置き換えられ、実際の割り当ては図 217 に示されているようになります。

```
&CNM01CSTTSO = 'S TSO'
```

図 217. 割り当てでの共通グローバル変数の置き換え

拡張自動操作サンプル・セットの共通グローバル変数を作成する際には、`AOPIGUPD` が呼び出されます。 `AOPIGUPD` の処理は、上記で説明した方法と類似しています。

共通変数の値をこのように共有することで、自動操作コマンド・リスト間で簡単に情報を共有できるようになります。拡張自動操作サンプル・セットに新しいシステム ID、機能、またはプロダクトを追加する際には、同じグローバル変数の規則を使用して新しい項目を追加してください。

拡張自動操作サンプル・セットの微調整

自動操作を実現するための指針として拡張自動操作サンプル・セットを使用する場合には、変更を加えて、実動環境における不要なオーバーヘッドを排除することが

できます。ユーザーのシステム環境で出すことができないメッセージは、自動化テーブルから削除してください。拡張自動操作サンプル・セットでサポートされているプロダクトが導入先で使用できない場合には、関係するコマンド・リストおよびサンプルの実動コピーを変更し、そのプロダクトを処理しないようにすることができます。変更されるコマンド・リスト、サンプル、およびパネルのコピーはそのまま保存して、必要に応じあとで参照できるようにしてください。自動化テーブル項目を削除したり、拡張自動操作サンプル・セットのコマンド・リストを変更したために使用されなくなったコマンド・リストおよびパネルは、DASD を節約し、混乱を防ぐため、実動自動操作システムから削除することができます。この場合も、将来必要になる可能性を考慮し、削除されるコマンド・リストのコピーをバックアップとして保存してください。

プロダクトの追加: 自動操作により処理されるプロダクトを新たに追加する前に、まず、そのプロダクトのどの範囲を処理するのかを見極める必要があります。新しいプロダクトの初期設定だけを処理するための自動操作か、または新しいプロダクトの処理（初期設定、能動モニター、リカバリー、シャットダウン）を可能な限り完全に行うための自動操作かを見極めなければなりません。

この範囲が決まると、実際にどのメッセージを処理するのかを見極める必要があります。

- このメッセージ数はなるべく少なくしてください。

例えば、プロダクトに複数の障害メッセージが存在するときに、各メッセージのあとにプロダクトを継続できないことを示す同じメッセージが常に表示される場合には、そのメッセージだけを自動化してください。

- 追加されるメッセージごとに、自動化テーブルに変更を加えなければならない場合があります。
- メッセージがシステム・メッセージの場合、そのメッセージがオペレーティング・システムのメッセージ処理機能を介して NetView プログラムに渡されることを確認してください。
- 新しい項目の状況が変化するたびに呼び出されなければならないコマンド・リストがある場合には、そのコマンド・リストを作成しなければならないか、あるいは既存のコマンド・リストを変更してメッセージの受信を正しく処理できるように変更しなければなりません。

範囲が決まったら、既存のサンプルを変更するか、または新しいコマンド・リストを作成してください。

- 自動化プロダクトを追加すると、自動操作処理の対象範囲によっては、自動化テーブルを変更したり、コマンド・リストを変更しなければならない場合があります。
- いずれの場合も、追加項目は、拡張自動操作サンプル・セットで使用されているパターンと同じパターンを使用すれば、簡単に定義することができます。

以下に、追加プロダクトにすべての自動操作機能を追加するために、更新しなければならないコマンド・リストとサンプルの概要を示します。

AOPIVARS

新しいプロダクトのグローバル変数を設定します。

DSIOPF

プロダクトの新しい自動化操作プログラムを追加します。

DSITBL11

自動化テーブル項目を追加します。

AOPInew

プロダクトの新しい初期設定コマンド・リストを作成します。

AOPSnew

プロダクトの新しいシャットダウン・コマンド・リストを作成します。

new clists

必要に応じ、自動化テーブルから呼び出される新しいコマンド・リストを追加します。

MPF システム・メッセージが NetView プログラムに転送されることを確認します。

汎用オペレーター・プロファイル DSIPROFG (CNMS6410)、汎用自動タスク初期コマンド・リスト AOPIGNIC (CNME6403)、主シャットダウン・コマンド・リスト AOPSMAN (CNME6412)、および自動操作状況表示コマンド・リスト AOPUSTAT (CNME6438) と関連パネルは、すべて、追加された新しい自動化プロダクトに使用することができます。新しい項目の始動およびシャットダウンを処理するために、新しいコマンド・リストを作成しなければならない場合があります。メッセージが自動化テーブル処理の対象になるとときには、新しい項目のリカバリーを処理するために、新しいコマンド・リストを作成しなければならない場合もあります。

自動操作による新しいメッセージの処理: 自動操作により新しいメッセージを処理するためには、まず、新しいメッセージが受信されたときに自動操作によりどのような処理を実行するかを正確に理解していなければなりません。メッセージを抑止したい場合には、オペレーティング・システムのメッセージ処理機能を使用して抑止してください。自動操作によりメッセージを経路指定するか、またはコマンド・リストを開始する場合には、自動化テーブルに変更を加える必要があります。さらに、システム・メッセージを NetView プログラムに転送するためには、オペレーティング・システムのメッセージ処理機能を更新しなければなりません。新しいメッセージのステートメントで行おうとしている処理が、別のメッセージの自動化テーブル・ステートメントですすでに行われている場合には、既存の自動化テーブル・ステートメントに新しいメッセージを追加するだけです。新しい自動化テーブル・ステートメントが必要な場合には、処理効率を考慮して、そのステートメントをテーブルの該当する箇所に挿入してください。例えば、BEGIN-END セクションを使用している場合には、新しいステートメントを該当するセクションに指定してください。

タイマー・コマンドの時間間隔の変更: 拡張自動操作サンプル・セットでは、タイマー・コマンドは、一定の時間間隔にしたがって、コマンド・リストを呼び出すか、あるいはコマンドを出します。AOPIVARS を編集すると、それぞれの環境に合わせてこの時間間隔を変更することができます。例えば、自動タスク・モニターはアクティブかどうかを確認するために各自動タスクを 2 分ごとにチェックします。この自動タスクのチェック頻度は、多くも少なくもできます。また、能動モニ

ター・コマンド・リストは、すべての自動化プロダクトの状況を 3 分ごとにチェックしています。能動モニターのチェック頻度は、操作頻度に合わせて変更することができます。

コマンド・リストの事前ロード: LOADCL および DROPCL コマンドは、共通に処理されるコマンド・リストを主記憶域にロードおよび除去するために使用することができます。コマンド・リストをストレージにロードしておく、コマンド・リストを実行するたびにロードする手間を省き、処理時間を大幅に短縮することができます。AOPIVARS がコマンド・リスト AOPIGUPD を呼び出す回数が多いため、拡張自動操作サンプル・セットのコマンド・リスト AOPIGUPD は、デフォルトの解釈ではコマンド・リスト AOPIVARS の中で事前ロードされています。

追加または変更された自動操作のテスト

拡張自動操作サンプル・セットを変更するときには、自動操作環境でアクションが取られるため、実動自動操作として使用する前に、制御されたテスト環境で自動操作をテストするのが最適です。手操作で呼び出されたコマンド・リストの処理をトレースすれば、自動操作を手操作でテストすることができます。例えば、NetView コマンド・リスト言語で作成されたコマンド・リストの部分を実行している場合には &CONTROL ALL を使用し、REXX を使用している場合には TRACE を使用してテストすることができます。コマンド・リストのテストが完了したら、トレース処理の部分削除を削除することができます。ネットワーク・ログまたはオペレーター・コンソールに対するメッセージをコマンド・リストに追加しておいて、正しい動作が確認されたらそれらを削除することができます。自動操作は、自動化テーブルから呼び出された場合と同様に、コマンド・リストを直接開始してテストすることができます。

コマンド・リストおよびサンプルの相互参照リスト

以下に、自動操作のためのサンプル・セットのコンポーネントとして提供されるサンプルおよびコマンド・リストをリストします。テーブルでサンプル名が指定されていない場合には、名前変更用 JCL (CNMS62J1) でそのサンプルに対する新しい名前を提供せず、出荷時の名前がそのまま使用されます。

基本自動操作サンプル・セット

サンプル

表 39. 基本自動操作サンプル・セットに含まれるサンプル

出荷時の名前	サンプル名	説明
CNMS6205	ACOTABLE	自動化テーブル項目
CNMS6206	CNMCMD	CNMCMD CMDDEF エントリー
CNMS6211	CLRLOG	SYS1.LOGREC をクリアし、記録できるようにする。
CNMS6212	CLRSMF	SYS1.MANX または SYS1.MANY をクリアし、記録できるようにする。
CNMS6213	LGPRNT	SYS1.LOGREC を印刷する。
CNMS6214	DSIPRT	1 次または 2 次ネットワーク・ログ、あるいはその両方を印刷する。
CNMS6221	\$CLRSMF	CLRSMF プロシージャへの入力

表 39. 基本自動操作サンプル・セットに含まれるサンプル (続き)

出荷時の名前	サンプル名	説明
CNMS6222	\$SOFT	LGPRNT プロシージャへの入力 (ステップ名 SOFT)
CNMS6223	\$SYSEXN	LGPRNT プロシージャへの入力 (ステップ名 SYSEXN)
CNMS6224	\$SYSUM	LGPRNT プロシージャへの入力 (ステップ名 SYSUM)

コマンド・リスト

表 40. 基本自動操作サンプル・セットに含まれるコマンド・リスト

出荷時の名前	コマンド同義語名	説明
CNME6201	AUTO1IC	AUTO1 自動タスクの初期コマンド・リスト。 \$DSPPOOL コマンド・リストを 24 時間ごとに開始する。
CNME6202	\$DSPPOOL	グローバル・パラメーター Z\$DSPPOOL および Z\$DSPLHRS を通常にリセットする。
CNME6203	\$DSPPOOL2	必要に応じてスプール・スペースのキュー・アクションをするために、グローバル・パラメーター Z\$DSPPOOL および Z\$DSPLHRS を設定する。
CNME6204	\$HASP646	スプール・スペースの使用率に従って、\$DSPPOOL2 の呼び出しを制御する。
CNME6205	IEE362A	SMF ファイルのクローズ後に、そのファイルを印刷する CLRSMF プロシージャの呼び出しを制御する。

拡張自動操作サンプル・セット

サンプル

表 41. 拡張自動操作サンプル・セットに含まれるサンプル

出荷時の名前	サンプル名	説明
CNMS64P0		すべてのプロダクトの自動操作状況を表示するパネル
CNMS64P1		特定プロダクトの自動操作情報を表示するパネル
CNMS64P2		メッセージ応答変数の値を表示するパネル
CNMS64P3		パネル CNMS64P0 のヘルプ・パネル
CNMS64P4		パネル CNMS64P1 のヘルプ・パネル
CNMS64P5		パネル CNMS64P2 のヘルプ・パネル
CNMS6401	CNMCMD	MVS コマンドの CMDDEF ステートメント
CNMS6402	CNMCMD	JES2 コマンドの CMDDEF ステートメント
CNMS6403	CNMCMD	JES3 コマンドの CMDDEF ステートメント
CNMS6404	CNMCMD	拡張自動操作サンプル・セットのコマンド・リストの CMDDEF ステートメント

表 41. 拡張自動操作サンプル・セットに含まれるサンプル (続き)

出荷時の名前	サンプル名	説明
CNMS6405	DSITBL11	拡張自動操作サンプル・セットに必要な NetView 自動化テーブル項目
CNMS6406	AOPUMCMT	NetView コマンド・リスト言語で作成されているコマンド・リストのコメントを削除してコピーする TSO コマンド・リスト
CNMS6408	DSIOPF	自動化操作プログラムの定義
CNMS6409	DSIPROFM	AUTOMGR 自動タスクのオペレーター・プロファイル
CNMS6410	DSIPROFG	汎用自動タスクのオペレーター・プロファイル

出荷時の名前で分類したコマンド・リスト

表 42. 出荷時の名前で分類した拡張自動操作サンプル・セットに含まれるコマンド・リスト

出荷時の名前	コマンド同義語名	説明
CNME6400	AOPIVARS	自動操作初期設定の主コマンド・リスト
CNME6401	AOPIGUPD	共通グローバル変数の値を設定する。
CNME6402	AOPIMGIC	AUTOMGR 初期コマンド・リスト
CNME6403	AOPIGNIC	自動タスクの汎用初期コマンド・リスト
CNME6404	AOPIJES3	JES3 を開始する。
CNME6405	AOPIJES2	JES2 を開始する。
CNME6406	AOPIVTAM	VTAM を開始する。
CNME6407	AOPIIMS	IMS を開始する。
CNME6408	AOPICICS	CICS を開始する。
CNME6409	AOPITSO	TSO を開始する。
CNME6410	\$HASP426	\$HASP426 SPECIFY OPTIONS に応答する。
CNME6412	AOPSMMAIN	主シャットダウン・コマンド・リスト
CNME6413	AOPSJES3	JES3 をシャットダウンする。
CNME6414	AOPSJES2	JES2 をシャットダウンする。
CNME6415	AOPSVTAM	VTAM を停止する。
CNME6416	AOPSIMS	IMS シャットダウンを開始する。
CNME6417	AOPSIMS2	IMS コンポーネントを停止する。
CNME6418	AOPSCICS	CICS を停止する。
CNME6419	AOPSTSO	TSO シャットダウンを開始する。
CNME6420	AOPSTSO2	TSO を停止する。
CNME6421	AOPTJRC3	IAT3714 の応答カウントをリセットする。
CNME6422	\$HASP098	メッセージ \$HASP098 に応答する。
CNME6423	\$HASP095	JES2 異常終了コードを保管する。
CNME6424	AOPSPURG	すべての装置のキュー・アクションをする。
CNME6425	VTAMTMRZ	VTAM を取り消し、シャットダウンする。
CNME6426	DFS996I	IMS WTOR を保管する。
CNME6428	DFS000IB	シャットダウンに必要な IMS 領域番号を保管する。

表 42. 出荷時の名前で分類した拡張自動操作サンプル・セットに含まれるコマンド・リスト
(続き)

出荷時の名前	コマンド同義語名	説明
CNME6429	IAT3714	JES3 開始タイプの応答を出す。
CNME6430	IAT3708	JES3 状況をアクティブに更新する。
CNME6431	\$HASP085	異常終了の場合には、JES2 の再始動を試みる。
CNME6432	JESTMRA	JES2 異常終了カウンターのリセット
CNME6433	DFS629I	IMS を再始動する。
CNME6434	IMSTMR	IMS タイマーをブランクに更新する。
CNME6435	DFH0606	CICS を再始動する。
CNME6436	CICSTMRA	CICS タイマーをブランクに更新する。
CNME6437	IKT002I	TSO の状況を異常終了に更新する。
CNME6438	AOPUSTAT	自動操作状況情報を含むパネルを表示する。
CNME6439	AOPMACT	自動化プロダクトの能動モニターを行う。
CNME6440	AOPMCHEK	拡張自動操作サンプル・セットの自動タスクをモニターする

コマンド同義語名で分類したコマンド・リスト

表 43. 同義語名で分類した拡張自動操作サンプル・セットに含まれるコマンド・リスト

出荷時の名前	コマンド同義語名	説明
CNME6431	\$HASP085	異常終了の場合には、JES2 の再始動を試みる。
CNME6423	\$HASP095	JES2 異常終了コードを保管する。
CNME6422	\$HASP098	メッセージ \$HASP098 に応答する。
CNME6410	\$HASP426	\$HASP426 SPECIFY OPTIONS に応答する。
CNME6408	AOPICICS	CICS を開始する。
CNME6403	AOPIGNIC	自動タスクの汎用初期コマンド・リスト
CNME6401	AOPIGUPD	共通グローバル変数の値を設定する。
CNME6407	AOPIIMS	IMS を開始する。
CNME6405	AOPIJES2	JES2 を開始する。
CNME6404	AOPIJES3	JES3 を開始する。
CNME6402	AOPIMGIC	AUTOMGR 初期コマンド・リスト
CNME6409	AOPITSO	TSO を開始する。
CNME6400	AOPIVARS	自動操作初期設定の主コマンド・リスト
CNME6406	AOPIVTAM	VTAM を開始する。
CNME6439	AOPMACT	自動化プロダクトの能動モニターを行う。
CNME6440	AOPMCHEK	拡張自動操作サンプル・セットの自動タスクをモニターする
CNME6418	AOPSCICS	CICS を停止する。
CNME6416	AOPSIMS	IMS シャットダウンを開始する。
CNME6417	AOPSIMS2	IMS コンポーネントを停止する。
CNME6414	AOPSJES2	JES2 をシャットダウンする。
CNME6413	AOPSJES3	JES3 をシャットダウンする。

表 43. 同義語名で分類した拡張自動操作サンプル・セットに含まれるコマンド・リスト (続き)

出荷時の名前	コマンド同義語名	説明
CNME6412	AOPSMMAIN	主シャットダウン・コマンド・リスト
CNME6424	AOPSPURG	すべての装置のキュー・アクションをする。
CNME6419	AOPSTSO	TSO シャットダウンを開始する。
CNME6420	AOPSTSO2	TSO を停止する。
CNME6415	AOPSVTAM	VTAM を停止する。
CNME6421	AOPTJRC3	IAT3714 の応答カウントをリセットする。
CNME6438	AOPUSTAT	自動操作状況情報を含むパネルを表示する。
CNME6436	CICSTMRA	CICS タイマーをブランクに更新する。
CNME6435	DFH0606	CICS を再始動する。
CNME6428	DFS000IB	シャットダウンに必要な IMS 領域番号を保管する。
CNME6433	DFS629I	IMS を再始動する。
CNME6426	DFS996I	IMS WTOR を保管する。
CNME6430	IAT3708	JES3 状況をアクティブに更新する。
CNME6429	IAT3714	JES3 開始タイプの応答を出す。
CNME6437	IKT002I	TSO の状況を異常終了に更新する。
CNME6434	IMSTMR	IMS タイマーをブランクに更新する。
CNME6432	JESTMRA	JES2 異常終了カウンターのリセット
CNME6425	VTAMTMRZ	VTAM を取り消し、シャットダウンする。

メッセージ抑止サンプル

表 44. メッセージ抑止サンプル

出荷時の名前	サンプル名	説明
CNMS6201	MPFLSTAC	消極的 MVS MPF メッセージ抑止
CNMS6202	MPFLSTAA	積極的 MVS MPF メッセージ抑止

ログ分析サンプル

表 45. ログ分析サンプル

出荷時の名前	サンプル名	説明
CNMS6207	SYSANAL	JES2 および JES3 ログ分析プログラム
CNMS62J2		ログ分析プログラムの実行

セットアップ・サンプル

表 46. セットアップ・サンプル

出荷時の名前	サンプル名	説明
CNMS62J1		改名用 JCL

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。

国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用するのですが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

プログラミング・インターフェース

本書には、主にプログラムを作成するユーザーが Tivoli NetView for z/OS のサービスを使用するためのプログラミング・インターフェースが記述されています。本書の情報は、Tivoli NetView for z/OS のプログラミング・インターフェースとして使用されることも意図して記述されたものではありません。この情報は、章またはセクションの始まりの文によって、あるいは下記の表示によって、その出現箇所を識別できます。

NOT Programming Interface information
End of NOT Programming Interface information

商標

IBM、IBMロゴおよびibm.comは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれIBM または各社の商標である場合があります。現時点でのIBMの商標リストについては、<http://www.ibm.com/legal/copytrade.shtml>をご覧ください。

Adobe は、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

Java およびすべての Java 関連の商標およびロゴは Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における登録商標です。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品（「ソフトウェア・オファリング」）では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie を含む様々なテクノロジーの使用の詳細については、IBM の『IBM オンラインでのプライバシー・ステートメント』（<http://www.ibm.com/privacy/details/jp/ja/>）の『クッキー、ウェブ・ビーコン、その他のテクノロジー』および『IBM Software Products and Software-as-a-Service Privacy Statement』（<http://www.ibm.com/software/info/product-privacy>）を参照してください。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

アクション

自動化テーブル 240

アクション・メッセージ 363

アクセシビリティ xxv

アクティブな自動化テーブル

識別 381

アセンブラ言語 26

アドレス・スペース 335, 593

アプリケーション・アドレス・スペース 335, 593

アプリケーション・プログラム・インターフェース (API) 30

アラート 458

作成 413, 518

情報の表示 14

チューニングに関する考慮事項 530

転送 343

ハードウェア・モニターの表示 413

表示, NetView 管理コンソール 414

フィルター操作 9, 342

フォーカル・ポイントへの転送 343, 444

ブロック 9

例外通知 413, 414

アラート主ベクトル 370, 376

アラート率 4

アラート・アダプター 458

アラーム 29

位置

メッセージの検索に使用する 364

位置決め, 自動操作のサンプル・セットの 652

一貫した操作手順 5

イベント

オーファン 387

子 387

トリガー 391

要約 387

EIF 459

イベント自動化サービス 40

フォーカル・ポイントへの転送 457

色

フルスクリーン表示 16

メッセージ 30, 37

MSU 30

インストール・システム出口 31

概要 329

DSIEX02A 329

インストール・システム出口 (続き)

DSIEX16 108, 330

DSIEX16B 330

DSIEX17 330

XITCI 329

インターフェース 515, 595

オペレーター 411

サービス・ポイント 457

使用 519

その他の NetView プログラムの 93

通信ネットワーク管理 595

ネットワーク例外条件に関する更新を受け取る場合の 511

プログラム間

アラートの送信 413

CNM

非送信請求ネットワーク管理データ 515

LU 6.2 トランスポート 414

MVS サブシステム 517

NetView

オペレーティング・システム 92, 108

NetView プログラム 91

POI (プログラム・オペレーター・インターフェース)

93

TAF 485

インターフェース, オペレーターの 13

インターフェース, NetView

メッセージ経路指定機能 96

インターフェース, NetView プログラム

自動化テーブル

経路指定メッセージ 106

メッセージ属性の設定 108

メッセージの処理 106

メッセージの廃棄または表示 108

ASSIGN COPY 処理 108

DSIEX16 108

待機処理 106

ハードウェア・モニター・データおよび MSU 93

メッセージ経路指定機能

ASSIGN コマンドによる経路指定 96

EMCS コンソールへの経路指定 102

MSGROUTE コマンドによる経路指定 102

MVS からの非送信請求メッセージ 95

インディケーター, 状況モニター重要メッセージ 253

インディケーター, 進行度の 587

引用符 167

同義語の値の区切り文字として 266

迂回, フィルターの 346

受け取る, 更新を

AON/TCP インターフェースの選択 511

運用グループおよび自動化チーム 48

エレメント, コマンド改訂テーブル・ステートメントの 149

- エレメント、自動化テーブル・ステートメントの 163
- エレメント、メッセージ改訂テーブル・ステートメントの 139
- エントリ・ポイント、制御範囲の 66
- 応答、メッセージおよび MSU への 12
- オカレンス番号
 - MSUSEG の検査に使用する 375
 - MSUSEG 戻り値 223
- オペレーター
 - インターフェース 411
 - オプション 13
 - 設計 62
 - 生産性 5
 - 定義 683
 - 定義ファイル (DSIOPF) 60
 - プロファイル 684
 - 役割 63
- オペレーター宛メッセージと要応答オペレーター宛メッセージ 33
- オペレーター制御
 - 動的 100
- オペレーター制御 TAF 485
- オペレーターの注意 511
- オペレーター・インターフェース
 - 拡張
 - サンプル・セット、拡張自動操作の 666
 - コマンド・リストおよびパネル
 - サンプル・セット 677
 - 自動操作表示パネル
 - サンプル・セット 678
- オペレーター・インターフェースの拡張
 - 拡張自動操作
 - サンプル・セット 666
- オペレーター・メッセージ削除 (DOM) コマンド 39
- オペレーティング・システム
 - NetView との通信の確立 333
 - NetView プログラム・インターフェース 92
 - z/OS
 - NetView プログラムとの通信の確立 340
 - z/OS オペレーティング・システム
 - NetView との通信の確立 333
- オペレーティング・システム自動操作機能
 - 概要 33
 - ソースに近い自動化のための使用 61
 - メッセージ・タイプ 6
- オンライン資料
 - アクセス xxv
- オンライン・ヘルプ・パネル 415

[力行]

- カートリッジ自動挿入機構、3480 用 42
- 会計上の分析 54
- 開始
 - MVS コマンド処理 643
- 開始プロシージャ
 - NetView 339

- 解析テンプレート
 - 複数行メッセージでの使用 367
 - ブレースホルダーを使用する 366
 - 変数の使用 367
- 概説 485
- 回線、専用および変換 450
- 改訂、コマンドの
 - CRT の使用 149
- 改訂、メッセージの 9
 - 概要 30
 - 使用、MRT の 139
- ガイドライン
 - 自動化テーブルの設計 266
- 概念視点
 - CP-MSU (制御ポイント管理サービス単位) 368
 - NMVT 368
- 概要
 - タイマー・コマンド 127
- 各種
 - 拡張自動操作
 - サンプル・セット 679
- 拡張 MCS コンソール
 - 動的定義
 - GETCONID 337
 - RELCONID 339
 - SETCONID 339
- 拡張 MCS コンソールの動的定義
 - コマンド
 - GETCONID 337
 - RELCONID 339
 - SETCONID 339
- 拡張可能な自動化 58
- 拡張コンソール
 - 動的定義 337
- 拡張自動操作 407
 - オペレーター・インターフェースの拡張
 - サンプル・セット 666
- 各種
 - サンプル・セット 679
- サンプル・セット 661
 - オペレーター・インターフェースの拡張 666
 - オペレーター・インターフェース・コマンド・リストおよびパネル 677
- 機能 666
 - コマンド・リスト 669
 - コマンド・リストの命名規則 669
 - 実行される機能 661
 - 自動操作表示パネル 678
 - シャットダウン 666
 - シャットダウン・コマンド・リスト 675
 - 受動モニター 662
 - 準備、使用の 680
 - 初期設定 661
 - で使用されるコマンド・リスト 666
 - 能動モニター 663
 - リカバリー 665, 679

拡張自動操作 (続き)

サンプル・セット (続き)

リカバリー・コマンド・リスト 673

JES より先に NetView を開始する 680

NetView 初期設定の準備 680

VTAM より先に NetView を開始する 682

シャットダウン

サンプル・セット 666

受動モニター 662

能動モニター 663

リカバリー

サンプル・セット 665

拡張自動操作サンプル・セット 692

拡張対等通信ネットワーク機能の制御範囲 66

拡張対等通信ネットワーク機能のリソース

モニター 508

拡張の制約 4

拡張複数コンソール・サポート・コンソール

考慮点 73

紹介 73

使用のための計画

経路コード 76

コンソールの獲得 75

コンソール命名規則 74

コンソールを使用可能にする 74

セキュリティ・アクセス 78

属性値 76

デフォルト値 76

メッセージ損失 79

メッセージ・キューの制限 79

メッセージ・ストレージ 78, 79

MPF を使用したメッセージ送信 76

MRT を使用したメッセージ送信 76

マイグレーション

クロスドメイン通信 80

コンソール名 80

AUTO 属性 81

MVS VARY コマンド 80

利点 73

確認、z/OS から NetView プログラムへのシステム・メッセージの転送の 336

EMCS コンソールの使用 336

確認、z/OS システムから NetView へのシステム・メッセージの転送の

サブシステム・インターフェースの使用 336

確認済みアラート・アダプター 458

活動化

基本自動操作サンプル・セット 658

コマンド・リスト同義語の定義 658

コマンド改訂テーブル 150

サンプル自動化テーブル 660

自動化テーブル 164, 287

自動タスク 340

メッセージ改訂テーブル 140

MVS コマンド出口 643

活動化、自動化テーブルの

セキュリティ・アプリケーション 682

テスト 290, 380

カプセル化された

RECMS (レコード保守統計) 372

カプセル化された RECMS

選択 372

可用性

会計上の値 54

設計 60

利点 4

カレンダー API 313

環境変数の表記 xxviii

監査可能な自動化 59

管理

自動化テーブル 287

管理サービス (MS) および高性能トランスポート

アラートの送信 414

管理サービス機能

(MS-CAPS) アプリケーション 66

管理サービス主ベクトル 368

管理サービス単位 (MSU)

アクション 373

強調表示 30

システム 6

自動応答 12

自動化する自動化テーブル・ステートメントの作成 367

シミュレート 539

処理、自動化テーブルの 29

近い処理、ソースに 61

デフォルト 261

ネットワーク 7

フィルター操作 9

MSU タイプの自動化テーブル・ステートメント 165

MVS システム 37

管理サービス・トランスポート 15

管理報告 43

関連プロダクト、自動操作の 41

キーワード

ACQUIRE、IF-THEN ステートメントの 179

ALWAYS 168

ALWAYS ステートメントの ALWAYS 263

AUTOMAN 287, 381

AUTOMATED

IF-THEN ステートメント 185

IF-THEN または ALWAYS ステートメント 241

AUTOMATED、IF-THEN ステートメント 178

AUTOTASK

IF-THEN ステートメント 186

AUTOTASK、IF-THEN ステートメント 178

AUTOTBL 287, 380

BEEP

IF-THEN または ALWAYS ステートメント 241

BEGIN、IF-THEN または ALWAYS ステートメント 168, 172, 263

キーワード (続き)

BIT
 ATF 条件項目、IF-THEN ステートメント 181
 BLU
 COLOR アクション、IF-THEN または ALWAYS ステートメント 242
 CBE 241
 COLOR
 IF-THEN または ALWAYS ステートメント 242
 CONTINUE
 IF-THEN または ALWAYS ステートメント 243
 CORRELATED
 IF-THEN ステートメント 187
 CORRFAIL
 IF-THEN ステートメント 187
 CURRDATE、IF-THEN ステートメントの 178, 187
 CURRTIME、IF-THEN ステートメントの 178, 187
 DISPLAY
 IF-THEN または ALWAYS ステートメント 243
 DOMACTION
 IF-THEN または ALWAYS ステートメント 243
 EDIT
 IF-THEN または ALWAYS ステートメント 245
 END ステートメント 168
 GRE
 COLOR アクション、IF-THEN または ALWAYS ステートメント 242
 HMCPLINK、IF-THEN ステートメントの 177, 192
 HMSPECAU、IF-THEN ステートメントの 177, 202
 IF-THEN ステートメントの ACTIONDL 179
 IF-THEN ステートメントの ACTIONMG 180
 IF-THEN ステートメントの AREAID 181
 IF-THEN ステートメントの ATF 181
 IF-THEN ステートメントの ATTENDED 178, 184
 IF-THEN ステートメントの AUTOTOKE 186
 IF-THEN ステートメントの CART 186
 IF-THEN ステートメントの CURSYS 178, 187
 IF-THEN ステートメントの DESC 188
 IF-THEN ステートメントの DISTAUTO 178, 188
 IF-THEN ステートメントの DOMAIN 178, 188
 IF-THEN ステートメントの DOMAINID 178, 189
 IF-THEN ステートメントの HDRMTYPE 178, 189
 IF-THEN ステートメントの HIER 189
 IF-THEN ステートメントの HMASPRID 177, 190
 IF-THEN ステートメントの HMBLKACT 177, 191
 IF-THEN ステートメントの HMEVTYPE 177, 195
 IF-THEN ステートメントの HMFWDDED 177, 196
 IF-THEN ステートメントの HMGENCAU 177, 198
 IF-THEN ステートメントの HMONMSU 177, 200
 IF-THEN ステートメントの HMORIGIN 177, 200
 IF-THEN ステートメントの HMSECREC 177, 201
 IF-THEN ステートメントの HMUSRDAT 177, 203
 IF-THEN ステートメントの IF 168, 170
 IF-THEN ステートメントの IFRAUI3X 178
 IF-THEN ステートメントの IFRAUIN3 178, 205
 IF-THEN ステートメントの IFRAUIND 178, 204

キーワード (続き)

IF-THEN ステートメントの IFRAUSB2 178, 206
 IF-THEN ステートメントの IFRAUSC2 178, 206
 IF-THEN ステートメントの IFRAUSDR 178, 207
 IF-THEN ステートメントの IFRAUSRB 178, 207
 IF-THEN ステートメントの IFRAUSRC 178, 207
 IF-THEN ステートメントの IFRAUTA1 178, 208
 IF-THEN ステートメントの IFRAUWF1 209
 IF-THEN ステートメントの INTERVAL 179, 209
 IF-THEN ステートメントの JOBNAME 210
 IF-THEN ステートメントの KEY 211
 IF-THEN ステートメントの LINEPRES 211
 IF-THEN ステートメントの LINETFLG 214
 IF-THEN ステートメントの MCSFLAG 214
 IF-THEN ステートメントの MSGAUTH 215
 IF-THEN ステートメントの MSGCATTR 216
 IF-THEN ステートメントの MSGCMISC 216
 IF-THEN ステートメントの MSGCMLVL 217
 IF-THEN ステートメントの MSGCMSGT 217
 IF-THEN ステートメントの MSGCOJBN 218
 IF-THEN ステートメントの MSGCPROD 218
 IF-THEN ステートメントの MSGDOMFL 219
 IF-THEN ステートメントの MSGGBGPA 219
 IF-THEN ステートメントの MSGGDATE 220
 IF-THEN ステートメントの MSGGFPGA 220
 IF-THEN ステートメントの MSGGMFLG 221
 IF-THEN ステートメントの MSGGMID 221
 IF-THEN ステートメントの MSGGTIME 221
 IF-THEN ステートメントの MSGID 222
 IF-THEN ステートメントの MSGSRCNM 222
 IF-THEN ステートメントの MSUSEG 223
 IF-THEN ステートメントの MSUSEG 条件項目の H 376
 IF-THEN ステートメントの MVSLEVEL 224
 IF-THEN ステートメントの NETID 225
 IF-THEN ステートメントの NETVIEW プログラム 179, 225
 IF-THEN ステートメントの NETVIEW、プログラム 179
 IF-THEN ステートメントの OPSYSTEM 179, 227
 IF-THEN ステートメントの ROUTCDE 227
 IF-THEN ステートメントの SESSID 227
 IF-THEN ステートメントの SYSCONID 228
 IF-THEN ステートメントの SYSID 228
 IF-THEN ステートメントの TASK 179, 229
 IF-THEN ステートメントの TEXT 229
 IF-THEN ステートメントの THEN 168, 172
 IF-THEN ステートメントの THRESHOLD 179, 230
 IF-THEN ステートメントの TOKEN 232
 IF-THEN ステートメントの VTAM 179, 233
 IF-THEN ステートメントの VTCOMPID 233
 IF-THEN ステートメントの WEEKDAYN 233
 IF-THEN ステートメントのヌル (") 235, 239
 IF-THEN または ALWAYS ステートメントの COLOR アクションの WHI 243
 IF-THEN または ALWAYS ステートメントの EXEC 245
 IF-THEN または ALWAYS ステートメントの EXEC アクションの ALL 247

キーワード (続き)

IF-THEN または ALWAYS ステートメントの EXEC アクションの CMD 245
IF-THEN または ALWAYS ステートメントの EXEC アクションの ONE 247
IF-THEN または ALWAYS ステートメントの HCYLOG 252
IF-THEN または ALWAYS ステートメントの HIGHINT 252
IF-THEN または ALWAYS ステートメントの HOLD 253
IF-THEN または ALWAYS ステートメントの NETLOG 253
IF-THEN または ALWAYS ステートメントの SRF 255
IF-THEN または ALWAYS ステートメントの SRF アクションの AREC 255
IF-THEN または ALWAYS ステートメントの SRF アクションの BLOCK 256
IF-THEN または ALWAYS ステートメントの SRF アクションの BOTH 256
IF-THEN または ALWAYS ステートメントの SRF アクションの ESREC 255
IF-THEN または ALWAYS ステートメントの SRF アクションの OPER 255
IF-THEN または ALWAYS ステートメントの SRF アクションの PASS 256
IF-THEN または ALWAYS ステートメントの SRF アクションの PRI 256
IF-THEN または ALWAYS ステートメントの SRF アクションの SEC 256
IF-THEN または ALWAYS ステートメントの SYSLOG 258
IF-THEN または ALWAYS ステートメントの XHILITE 258
IF-THEN または ALWAYS ステートメントの XHILITE アクションの BLI 258
IF-THEN または ALWAYS ステートメントの XHILITE アクションの NONE 258
IF-THEN または ALWAYS ステートメントの XHILITE アクションの REV 258
IF-THEN または ALWAYS ステートメントの XHILITE アクションの UND 258
IF-THEN または ALWAYS ステートメントの XLO 258, 346
LISTING 例
AUTOTBL 275
MSUSEG 条件項目の H
IF-THEN ステートメント 223
NUMERIC、IF-THEN ステートメントの 225
NVCLOSE 179, 225
OPID
IF-THEN ステートメント 179
OPID、IF-THEN ステートメント 226
PIN
COLOR アクション、IF-THEN または ALWAYS ステートメント 242

キーワード (続き)

PPT
IF-THEN または ALWAYS ステートメントの EXEC アクション 247
RED
COLOR アクション、IF-THEN または ALWAYS ステートメント 242
ROUTE
IF-THEN または ALWAYS ステートメントの EXEC アクション 246
IF-THEN または ALWAYS ステートメントの SRF アクション 255
SYN ステートメントの SYN 265
TECROUTE
IF-THEN または ALWAYS ステートメントの SRF アクション 255
TRACE、IF-THEN または ALWAYS ステートメントの 258
TRAPROUT
IF-THEN または ALWAYS ステートメントの SRF アクション 256
TUR
COLOR アクション、IF-THEN または ALWAYS ステートメント 242
VALUE、IF-THEN ステートメントの 179, 232
YEL
COLOR アクション、IF-THEN または ALWAYS ステートメント 243
%INCLUDE 264
記憶管理 43
記憶管理サブシステム (SMS) 43
記述子コード 3 メッセージ 363
規則
書体 xxvii
機能
拡張自動操作サンプル・セット 666
コマンドの経路指定 114
自動化テーブルの ROUTE キーワード 114
CNMSMSG サービス・ルーチン 114
タスクへの経路指定
EXCMD コマンド 114
RMTCMD コマンド 114
機能、拡張自動操作サンプル・セットによって実行される 661
機能、基本自動操作のサンプル・セットによって実行される 654
機能拡張、自動化のための
NetView for OS/390 V1R4 578
NetView for z/OS V5R4 578
NetView for z/OS V6R2 577
基本アセンブラ言語 26
基本自動化テーブル
サンプル・セット
コマンドの発行 655
コマンド・リストおよびコマンド・プロセッサの呼び出し 657
変数への値の割り当て 656

基本自動操作
 サンプル・セット
 実行される機能 654
基本自動操作サンプル・セット 654, 691
 活動化 658
 コマンド・リスト同義語の定義 658
 サンプル・セットで使用する自動化テーブル 655
 テスト 660
キューに入れられたコマンド
 優先順位 115
強調表示
 メッセージ 30
 MSU 30
共通グローバル変数 121
許可レシーバー
 非送信請求メッセージ 95
 メッセージの経路指定の流れ 105
 DST からの非送信請求メッセージ 95
 MVS システムからの非送信請求メッセージ 95
記録 AIFR 532
記録フィルター 342
記録フィルター属性、設定 373
組み込み、メンバーおよびファイルの (%INCLUDE) 264
クラス、自動操作の 5
グラフィック・モニター機能、NetView の 15
グラフィック・モニター機能ホスト・サブシステム (GMFHS)
 データ・モデル 87
グループ
 自動化テーブル 287
グループ化
 自動化テーブル・ステートメント
 BEGIN-END セクションによる 267
グループ化の順序
 条件
 IF-THEN ステートメント内で 173
グローバル変数 405
 共通 121
 タスク 121
グローバル変数およびコマンド・プロシージャ 27
計画、プロジェクトの
 計画表
 実施 585
 実動 585
 設計 585
 プロジェクトの定義 584
 作成 49
 定義 579
 フェーズ、プロジェクトの
 概要 579
 実施フェーズ 583
 実動フェーズ 584
 設計フェーズ 582
 定義フェーズ 580
 例 579
計画およびプロジェクト定義フェーズ 47

経路コード
 メッセージの EMCS コンソールへの経路指定のための使用
 102
経路コード、拡張複数コンソール・サポート・コンソールの
 76
経路コードの指定 102
経路指定
 コマンド
 自動化テーブルの ROUTE キーワード 114
 CNMSMSG サービス・ルーチン 114
 DSIMQS マクロ 114
 送信請求メッセージ 99
 タスクへの
 EXCMD コマンド 114
 RMTCMD コマンド 114
 非送信請求メッセージ
 ASSIGN コマンドの使用 97
 割り当てられた宛先の検証 100
 ASSIGN による自動タスクへのメッセージ 100
 NetView プログラム・コマンド 113
経路指定、プレースホルダーを使用するメッセージの、例
 プレースホルダーを使用する 366
経路指定、論理 OR 論理を使用するメッセージの、例
 論理 OR 論理の使用 365
経路指定機能
 コマンド 114
 メッセージの 96
経路指定の流れ
 メッセージ 103
 許可レシーバーの処理 105
 ASSIGN PRI/SEC 処理 104
 DSIEX02A 処理 105
 DSIEX17 処理 104
 PIPE CORRWAIT 104
経路指定メッセージ 106
経路指定目標設定命令 GDS 変数 377
決定、自動化するメッセージおよび MSU の 361
言語
 アセンブラー 26
 選択 26
 C 26
 CLIST (コマンド・リスト) 26
 PL/I 26
 REXX 26
言語選択 119
検査
 カプセル化された RECMS 372
 サブベクトル 370
 全オカレンス、フィールドの 375
 フィールドの存在 369
 メッセージ ID による 362
 Alert Major Vectors MDS-MU に 376
 MDS-MU におけるアラート主ベクトル 377
 MSU におけるフィールドの複数オカレンス 375
 RECMS と RECFMS 371
 X'82' の記録モードを持つ RECMS 372

検査、フィールドの内容 371
検索
 コマンド改訂テーブル 150
 自動化テーブル 164
 全オカレンス、フィールドの 375
 メッセージ改訂テーブル 140
 MSU におけるフィールドの複数オカレンス 375
検索、メッセージのグループの
 プレースホルダーの使用による 366
研修 62
 Tivoli 技術研修を参照 xxvi
研修、Tivoli 技術 xxvi
検証
 自動化ステートメントの構文 380
 自動化テーブル 380
検討、NetView 開始プロシージャの 339
コーディング、コマンド改訂テーブルの 151
コーディング、自動化テーブルの 166
コーディング、メッセージ改訂テーブルの 141
交換回線 450
更新
 受け取る場合の AON/TCP インターフェースの選択 511
高水準言語 (PL/I および C) 26
構成、集中操作の 450
高性能および MS トランスポート
 アラートの送信 414
構文
 自動化テーブル・ステートメント
 BEGIN-END セクション 168
互換性
 コマンドとタスクとの 113
コスト分析 54
コピー、自動化ルーチンの
 設計 58
 単一システムの自動操作 16
コマンド
 改訂 149
 機能、情報の表示の 14
 キューに入れられたものの優先順位 115
 経路指定機能 114
 自動化テーブルの ROUTE キーワード 114
 CNMSMSG サービス・ルーチン 114
 DSIMQS マクロ 114
 コマンドのスケジューリング 11
 システム 6
 MVS 594
 処理プログラム 26, 87
 タイマー
 検証 541
 説明 127
 保管および復元 130
 タイマー・コマンド
 概要 28
 タスクとの互換性 113
 タスクへの経路指定
 EXCMD コマンド 114

コマンド (続き)
 タスクへの経路指定 (続き)
 RMTCMD コマンド 114
出すための z/OS プロセッサの使用
 NetView プログラムからの 339
転送 445
ネットワーク 7
プロシージャ
 概要 25
 自動化 26
 統合、コマンドの 11
リスト 25
リスト言語 25
AFTER 28, 127
ALLOCATE 125, 551
AON TIMER 28
AT 28, 128
AUTOCNT 275
AUTOMAN
 活動化、自動化テーブルの 287, 381
AUTOTASK 134, 660
AUTOTBL
 活動化、自動化テーブルの 287, 380
 セキュリティ・アプリケーション 682
AUTOTEST 531
CHRON 28, 129
 検証 541
DBCS ストリング 245
DEFAULTS
 メッセージ属性 260
 ロギング 552
DELAY 128
DFILTER 346
DOM 125
DROPCL 126, 691
EVERY 28, 128
FOCALPT 453
FREE 551
GENALERT 15
 アラートの送信 414
 複数の NetView プログラム 518
GETCONID 75, 337
HELP 16
LIST TIMER 28, 131
LOADCL 126, 691
MAPCL 126
OVERRIDE
 メッセージ属性 260
 ロギング 552
PURGE TIMER 28, 131
RELCONID 339
RESTORE 130
RMTCMD
 コマンドの転送 445
ROUTE 446
SETCONID 75, 339

コマンド (続き)

SRFILTER 9, 343
SUBMIT 125
SVFILTER 346
TIMER 129
VIEW 16
WTO 125, 589
WTOR 125, 589

コマンド応答の自動化の制限 272

コマンド改訂テーブル

検索 150
コーディング 151
コーディング、ステートメントの
END 155
ISSUE.IEE295I 152
NETVONLY 156
OTHERWISE 155
REVISE 155
SELECT 154
TRACKING.ECHO 152
UPON 153
WHEN 154
WTO 157

コメント 151

使用 149

処理 150

制限 514

テスト 161

コマンド改訂テーブル・ステートメント

エレメント
END ステートメント 150
EXIT ステートメント 150
INCLUDE ステートメント 150
NETVONLY ステートメント 150
OTHERWISE ステートメント 150
REVISE ステートメント 150
SELECT ステートメント 149
UPON ステートメント 149
WHEN ステートメント 150

コマンド処理

MVS

開始 643

コマンドの流れ、NetView システムと z/OS システム 36

コマンドの発行

基本自動操作サンプル・セット 655

コマンド・プロシージャ

説明 119
統合、コマンドの 355
文書化 357
例 359

コマンド・プロセッサ

説明 119
統合、コマンドの 355

コマンド・ラベル接頭部 115

コマンド・リスト

拡張自動操作サンプル・セットの 666

コマンド・リスト (続き)

シャットダウン

拡張自動操作サンプル・セット 675

初期設定

拡張自動操作サンプル・セット 669

説明 119

リカバリー

拡張自動操作サンプル・セット 673

REXX 119

コマンド・リストおよびコマンド・プロセッサの呼び出し

基本自動操作サンプル・セット 657

コマンド・リストおよびパネル

オペレーター・インターフェース

拡張自動操作サンプル・セット 677

自動操作表示パネル

拡張自動操作サンプル・セット 678

コマンド・リスト言語 25

コミュニケーション管理構成 (CMC) システム、フォーカル・
ポイントとしての 65

コメント

コマンド改訂テーブル 151

自動化テーブル 167

メッセージ改訂テーブル 141

コレクター・ルール 396

コンソール

コマンド機能 73

統合 10

MVS 38, 73

NetView 10

コンソール、EMCS

動的定義 337

コンソールの統合 349

[サ行]

サービス水準報告 (SLR) プログラム 43

サービス・ポイント 457

サービス・レベル・アグリーメント

環境について 53

要件の識別 50

再構造化拡張実行プログラム (REXX) 言語 26

作成、アラートの 413

作成、自動化テーブル・ステートメントの (メッセージを自動
化する) 362

作成、Common Base Event の 493

サブシステム

アドレス・スペース 335, 593

インターフェース 517

名前テーブル (SSNT) 590

サブシステム・インターフェース

メッセージの送信 35

サブフィールド

選択 370

サブフィールドの検査 370

サブフィールドの選択 370

- サブベクトル
 - 選択 370
- サポート xxvi
- サンプル
 - 自動化 13
 - 進行の測定方法 587
 - プロジェクト計画 579
 - IHSAACDS 459
 - IHSABCDs 459
 - IHSALCDS 459
 - IHSAMFMT 459
 - IHSANFMT 459
- サンプル自動化テーブル
 - 活動化 660
 - 準備 659
 - テスト、構文の 659
- サンプル自動化テーブルの準備 659
- サンプル・セット
 - 各種
 - 拡張自動操作 679
 - 拡張自動操作 661
 - オペレーター・インターフェースの拡張 666
 - オペレーター・インターフェース・コマンド・リストおよびパネル 677
 - 機能 666
 - コマンド・リストの命名規則 669
 - 実行される機能 661
 - 自動操作表示パネル 678
 - シャットダウン 666, 675
 - 受動モニター 662
 - 初期設定 661
 - 初期設定および能動モニター・コマンド・リスト 669
 - で使用されるコマンド・リスト 666
 - 能動モニター 663
 - リカバリー 665, 673
- 基本自動操作 654
 - 活動化 658
 - 実行される機能 654
- 基本自動操作の活動化
 - 活動化 658
- 自動化された処理 661
- 自動化テーブル
 - 基本自動操作サンプル・セットを使用して 655
 - コマンドの発行 655
 - コマンド・リストおよびコマンド・プロセッサの呼び出し 657
 - 変数への値の割り当て 656
- 自動操作の 651
- 準備、使用の
 - 拡張自動操作 680
- メッセージの抑止 653
- ログ分析プログラム 653
- JES より先に NetView を開始する
 - 拡張自動操作 680
- MVS サンプル・ロケーション 652

- サンプル・セット (続き)
 - NetView 初期設定の準備
 - 拡張自動操作 680
 - VTAM より先に NetView を開始する
 - 拡張自動操作 682
- サンプル・セット、自動操作の
 - 拡張コマンド・リスト 693, 694
 - 拡張サンプル 692
 - 基本コマンド・リスト 692
 - 基本サンプル 691
 - セットアップ・サンプル 695
 - プロダクト ID 687
 - メッセージ抑止サンプル 695
 - ログ分析サンプル 695
- シーケンス、自動化における各段階の
 - 概要 8
 - 例 21
- シーケンス番号
 - 自動化テーブル 287
- しきい値 502
- しきい値ルール 394
- しきい値分けおよび MIB ポーリング
 - z/OS TCP/IP の場合のみ 510
- 事業目標と自動化 49
- システム
 - 可用性
 - 会計上の値 55
 - 設計 60
 - 利点 4
 - コマンド 6
 - 自動化 5
 - メッセージ 6
 - ログ 51, 550
 - MSU 6
- システム間カップリング・ファシリティ (XCF)、シスプレックスの 83
- システム指向の要件 50
- システム・コマンド
 - 出すための z/O プロセッサの使用
 - NetView プログラムからの 339
- システム・シンボルの置換 167
- システム・メッセージ
 - NetView プログラムにより処理される数の限度 266
 - z/OS から NetView への転送の確認 336
 - EMCS コンソールの使用 336
 - z/OS システムから NetView プログラムへの転送の確認
 - サブシステム・インターフェースの使用 336
- システム・メッセージの抑制 341
- シスプレックス 420
- シスプレックス、MVS の
 - システム間カップリング・ファシリティ (XCF) 83
- 自動操作の計画
 - 集中化、NetView プログラム自動操作の 85
 - メッセージ経路指定 37, 84
 - MPF アクション 85
- 紹介 83

シスプレックス、MVS の (続き)

利点、自動化の 84

事前ロード、コマンド・リストの 691

持続セッション 452

実施フェーズ 69, 583

実動フェーズ 69, 584

指定文字 38

自動化

アクション 363

オペレーター・インターフェースの拡張

サンプル・セット、拡張自動操作の 666

各種サンプル

拡張自動操作 679

機能 119

サンプル・セット 651

シャットダウン

サンプル・セット、拡張自動操作の 666

準備、使用の

拡張自動操作サンプル・セット 680

初期設定 661

受動モニター 662

能動モニター 663

整合性をとる

高度の 405

タイプ 5

チューニング 523

テスト 531

伝搬

同期化 419

統計 275

非 SNA 457

プラットフォーム 461, 466

リカバリー

サンプル・セット、拡張自動操作の 665

ロギング 549

JES より先に NetView を開始する

拡張自動操作サンプル・セット 680

JES3 555

NetView 初期設定の準備

拡張自動操作サンプル・セット 680

SNMP トラップ 563

TAF (端末アクセス機能) 485

VTAM より先に NetView を開始する

拡張自動操作サンプル・セット 682

自動化サポート

VTAM リソース、SNA サブエリア 508

自動化障害ロジック 502

自動化処理

メッセージおよび MSU 361

自動化ステートメント

検証 380

使用可能化 380

自動化セットアップ 333

システム・メッセージの転送、z/OS から NetView プログラムへの 336

NetView と z/OS オペレーティング・システム 333

自動化セットアップ (続き)

自動タスクの定義と活動化 340

NetView とオペレーティング・システム 333

NetView を z/OS オペレーティング・システムにサブシステムとして定義 336

自動化操作プログラム

理解 500

自動化追跡

自動化ログ 501

理解 501

自動化通知ログ

ハードウェア・モニターでの 502

自動化テーブル 500

アクション 240

活動化 380

管理 287

基本自動操作サンプル・セットの

コマンドの発行 655

コマンド・リストおよびコマンド・プロセッサの呼び出し 657

変数への値の割り当て 656

グループ 287

経路指定メッセージ 106

検索 164

検証 540

コーディング 166

コメント 167

シーケンス番号 287

システム・シンボルの置換 167

終了ラベル 287

主要メンバーの例 274, 278

使用可能化および使用不可 287

使用不可ステートメント 287

使用報告書 275

処理 112, 164

設計のガイドライン 266

説明 163

チューニング 529

テスト 531

トレース

デバッグ 545

ブロック 287

メッセージ

DSIEX16 108

メッセージ属性の設定 108

メッセージの処理 106

メッセージの廃棄または表示 108

読みやすく作成 267

ラベル 287

リスト

説明 271

デバッグ 545

例 275, 279

ALWAYS ステートメント 263

ASSIGN COPY 処理 108

DBCS ストリング 167

自動化テーブル (続き)

- IF-THEN ステートメント 169
- in basic automation サンプル・セット 655
- SYN ステートメント 265
- %INCLUDE ステートメント 264
 - 一連のロード 270
- 自動化テーブル、アクティブ識別 381
- 自動化テーブル機能 (ATF) 182
 - DSICGLOB 183
- 自動化テーブル経路指定と ASSIGN コマンド 101
- 自動化テーブル・ステートメント 362
 - エレメント
 - Always ステートメント 164
 - BEGIN-END セクション 164
 - IF-THEN ステートメント 163
 - SYN ステートメント 163
 - %INCLUDE ステートメント 164
 - グループ化 例 268
 - 構文
 - BEGIN-END セクション 168
 - メンバー DSIPARM へのステートメントの保管 164
- 自動化とリカバリーの理解 499
- 自動化内部機能要求 174
- 自動化の機能拡張
 - NetView for OS/390 V1R4 578
 - NetView for z/OS V5R4 578
 - NetView for z/OS V6R2 577
- 自動化の伝搬 419
- 自動化メッセージ
 - トークンの使用 363
- 自動化ログ 501
- 自動化ロジック
 - 宛先への経路指定検証のために使用 100
- 自動コンソール操作 10
- 自動操作
 - アプローチの選択 48
 - 機能 25
 - クラス 5
 - コマンド・プロシージャ機能 26
 - コミットメント 56
 - サンプル 13
 - 事業目標 49
 - システム 5
 - 実施 69
 - 実動システムへの移行 69
 - 紹介 3
 - 使用報告書 30
 - 新機能へのマイグレーション 577
 - 整合性のとれた 12
 - 設計 57
 - タスク 28
 - 単一システム
 - 設計、伝搬の 58
 - 段階 8

自動操作 (続き)

- 単一システム (続き)
 - 定義 7
 - 伝搬 16
- 段階
 - 概要 8
 - 例 21
- チームの編成部門 48
- 近い、ソースに 61
- テーブル 29
 - 定義 47
- ネットワーク 6
- 複数システム
 - 段階 8
 - 定義 8
- プロダクト 25, 41
- 見積もりの値 587
- メッセージと MSU の応答 12
- 利点 3
- MVS システム 33
- MVS のシスプレックス 37, 83
- MVS を使用した EMCS コンソール 73
- NetView プログラムの終了 273
- NetView プログラム用の定義 3
- RODM
 - 機能 89
 - 自動化 88
 - 自動操作の統合 13
 - 紹介 30
 - 利点と考慮点 88
- RODM のイベント (シナリオ) の概要 89
- 自動操作サンプル・セット
 - 使用 651
- 自動操作表示パネル
 - オペレーター・インターフェース
 - 拡張自動操作サンプル・セット 678
- 自動タスク
 - 概要 28
 - 活動化 134
 - 関連付け、複数コンソール・サポート・コンソールの 353
 - 経路指定のための ASSIGN コマンドの使用 100
 - 自動化 135
 - 説明 133
 - タイマー・コマンド 129
 - 定義 133
 - 定義と活動化 340
 - パスワード 133
 - 非活動化 135
 - 複数 528
 - 複数コンソール・サポート (MCS) コンソールの関連付け 134
- 自動電源制御 (APC)、9370 の 20
- シナリオ (イベントの概要)、RODM を使用した自動化の 89
- シミュレート
 - メッセージ 538
 - MSU 539

- シャットダウン 409
 - 拡張自動操作
 - サンプル・セット 666
 - コマンド・リスト
 - 拡張自動操作サンプル・セット 675
- 修正、コマンド・プロシーチャーの 357
- 集中操作 8, 18, 423
- 終了
 - NetView プログラム、自動操作の 273
- 終了ラベル
 - 自動化テーブル 287
- 受動モニター 12, 408, 662
- 主ベクトル
 - アラート 261, 370, 373
 - 管理サービス 367, 368
 - 経路指定目標設定命令 GDS 変数 377
 - 自動化可能 376
 - レゾリューション 376
 - 1044 371
 - 1045 371
- 準備、使用の
 - 拡張自動操作
 - サンプル・セット 680
- 使用上の注意、ACTIONDL の 180
- 使用上の注意、ATF の 182
- 紹介、自動操作の 3
- 使用可能化
 - 自動化ステートメント 380
 - 自動化テーブル 287
- 使用可能ステートメント 174
- 状況情報
 - 表示 412
- 状況情報の転送 423
- 状況のモニター 12
- 状況変更
 - 自動化 380
- 状況モニター 15, 412, 519
- 条件
 - グループ化の順序
 - IF-THEN ステートメント内で 173
 - 論理 AND 演算子により結合される
 - IF-THEN ステートメント内で 172, 173
 - 論理 OR および論理 AND 演算子
 - IF-THEN ステートメント内で 173
- 条件項目
 - 定義 181
 - メッセージおよび MSU に関して 175
 - MSU に関して 175
- 条件項目、IF-THEN ステートメントの
 - オカレンス削除例 268
 - 構文 170
 - 複数リンク 172
- 状態相関
 - 概要 387
 - コレクター・ルール 396
 - サンプル XML ファイル 388
- 状態相関 (続き)
 - しきい値ルール 394
 - 状態マシン 391
 - 遷移 393, 394, 395, 396
 - 総計値 395
 - 属性、共通の 392
 - 重複ルール 393
 - 突き合わせルール 392
 - パススルー・ルール 397
 - 複製 400
 - リセット・オン・マッチ・ルール 399
 - ルール 392
 - allEvents モード 394
 - firstEvent モード 394
 - forwardEvents モード 394
 - id 属性 392
 - lastEvent モード 394
 - thresholdCount 属性 394
 - timeInterval 属性 394
 - timeIntervalMode 属性 394
 - triggerMode 属性 394
 - XML 388
- 状態変数 405
- 状態マシン
 - 定義 391
- 使用不可
 - 自動化テーブル 287
- 使用不可ステートメント 174
 - 自動化テーブル 287
- 情報
 - メッセージおよび MSU の抽出 366
- 情報管理システム (IMS) 490
- 使用報告書
 - 自動化テーブル 275
- 使用報告書、自動化テーブルの 30
- 初期設定 408
 - 拡張自動操作 661
 - コマンド・リスト
 - 拡張自動操作サンプル・セット 669
- 初期設定、分散システムの 19
- 除去
 - 非送信請求メッセージ 99
- 書体の規則 xxvii
- 処理
 - コマンド改訂テーブル 150
 - 自動化テーブル 164
 - メッセージ改訂テーブル 140
- 資料
 - アクセス、オンライン xxv
 - 注文 xxv
 - NetView for z/OS xxi
- 人事
 - 教育 62
 - 編成 48
 - 役割 63

スケジューリング
 コマンド実行
 タイマー・コマンドの使用 (概要) 28
 自動化の段階 11
 プロジェクト 57
 スケジュール済みコマンド、検証 541
 スケジュールの定義
 NMC ビューのリソースのための
 NMCSTATUS ポリシー定義に基づいて 301
 ステートメント、コマンド改訂テーブル
 END 155
 ISSUE.IEE295I 152
 NETVONLY 156
 OTHERWISE 155
 REVISE 155
 SELECT 154
 TRACKING.ECHO 152
 UPON 153
 WHEN 154
 WTO 157
 ステートメント、自動化テーブル
 ALWAYS 263
 SYN 265
 %INCLUDE 264
 ステートメント、自動化テーブルの
 BEGIN-END セクション 168
 IF-THEN 169
 ステートメント、メッセージ改訂テーブル
 END 142
 EXIT 142
 NETVONLY 142
 OTHERWISE 143
 REVISE 143
 SELECT 144
 UPON 144
 WHEN 145
 制御行、MLWTO メッセージの 173
 制御点管理サービス単位 (CP-MSU)
 概念視点 368
 制御範囲
 オペレーター管理 431
 概要 66
 環境 433
 機能 430
 状態 432
 タイプ 431
 マネージャー 66
 例 65, 66
 MS-CAPS 管理 430
 SOC-MGR 430
 制御ファイル
 AON 500
 制限
 NetView プログラムにより処理されるシステム・メッセー
 ジ数 266
 制限、コマンド応答の自動化の 272
 整合性のとれた自動操作 12, 407
 生産性、オペレーターの 5
 正符号
 MVS メッセージの 174
 制約、拡張に対する 4
 制約事項
 コマンド改訂テーブル 514
 セキュリティー 682
 拡張複数コンソール・サポート・コンソール 78
 設計 60
 設計、自動操作の 57, 582
 設計上の、自動化テーブルの 266
 設計作業、自動操作プロジェクトのための 57
 実施のためのスケジュール段階 57
 自動化すべきプロシージャーと機能の確認 57
 標準の確立 58
 プロシージャーおよび機能の優先順位付け 57
 設計のガイドライン
 オペレーター・インターフェースの提供 62
 概要 58
 スタッフの教育 62
 スタッフの役割変更の予測 63
 設計、拡張と伝搬 58
 設計、可用性の 60
 設計、監査能力の 59
 設計、セキュリティの 60
 選択、複数の NetView プログラムの 61
 ソースに近い自動操作 61
 提供、問題管理、変更管理の 64
 テストの提供 63
 フォーカル・ポイントの選択 64
 フォーカル・ポイント・バックアップの使用 66
 セッション
 持続 452
 非持続 452
 フルスクリーン TAF 485
 LU 6.2、アラートの送信 414
 LUC、アラート転送 444
 OST-NNT 446
 PPI と TCP/IP、アラートとメッセージの転送 457
 設定、自動化テーブルの
 処理 164
 セットアップ、NetView と MVS 間の通信の 653
 接頭部、コマンド・ラベル 115
 セミコロン
 同義語にはなし
 名前 266
 変数 266
 遷移 391, 393, 394, 395, 396
 全オカレンス、フィールドの
 検索 for 375
 選択
 カプセル化された RECMS 372
 サブベクトル 370
 タスク 129
 フィールドの存在 369

選択 (続き)

- メッセージ ID 362
- AON/TCP インターフェース
 - 更新を受け取る場合の 511
- MDS-MU におけるアラート主ベクトル 376
- RECMS と RECFMS 371
- X'82' の記録モードを持つ RECMS 372

選択、自動化チームによるアプローチの 48

選択、フィールドの内容 371

専用回線 450

ソース LU (SRCLU) 486

関連

- イベント 383
- イベント・マッピング 385
- 概要 381
- 自動化テーブル項目 382
- ストレージに関する考慮事項 383
- メッセージ 383
- メッセージ処理 383
- CNMCRMSG 384
- COREVENT 384
- MSU 383
- MSU 処理 383

関連、状態

「状態関連」を参照 387

総計値 395

操作手順

- 一貫性 5
- 操作の集中化 17

送信

メッセージ、MVS オペレーター・コンソールへの 125

送信請求メッセージ

- 経路指定のための ASSIGN コマンド 99, 100
- メッセージ経路指定 94

送信元トークン 478

属性

- 状態関連 392
- ALERTPCT 79
- QLIMIT 79
- thresholdCount 394
- timeInterval 394
- timeIntervalMode 394
- triggerMode 394

測定、進行度の 54, 587

[タ行]

ターゲット・システム

定義 18

ターゲット・システム制御機能 (TSCF) 19

待機処理 106, 123

タイプ

- コマンド改訂テーブル・ステートメント 149
- 自動化テーブル・ステートメント 163
- メッセージ改訂テーブル・ステートメント 139
- ALWAYS ステートメント 165

タイプ (続き)

IF-THEN ステートメント 165

タイプ、自動操作の 5

タイマー

状態関連 393, 394, 396

タイマー API 313

タイマー・コマンド

概要 28, 127

検証 541

コマンドのスケジューリング 11

保管および復元 130

多重仮想記憶域オペレーティング・システム

自動化での役割 33

タスク

コマンドとの互換性 113

タイマー・コマンドの 129

優先順位 527

タスク・グローバル変数 121

単一システムの自動操作

設計、伝搬の 58

段階 8

定義 7

伝搬 16

端末アクセス機能 (TAF)

オプション 486

集中操作 448

説明 485

VTAM インターフェース 595

チーム

教育 62

編成 48

役割 63

チームの編成部門 48

チャンネル間 (CTC) 555

チュートリアル

AON/SNA 506

チューニング勧告 523

中間フォーカル・ポイント 447

超過、キュー制限の 79

重複ルール 393

直接アクセス記憶装置 (DASD) 管理 43

通信

NetView と z/OS オペレーティング・システムの 336

NetView とオペレーティング・システムの 333

z/OS からのシステム・メッセージの転送の確認 336

EMCS コンソールの使用 336

z/OS システムからのシステム・メッセージの転送の確認

サブシステム・インターフェースの使用 336

通信ネットワーク管理 596

インターフェース 595

通信ネットワーク管理インターフェース (CNMI)

非送信請求ネットワーク管理データ 515

通知

理解 501

通知、オペレーターへの問題の 13

突き合わせルール 392

データ機能記憶管理サブシステム (DFSMS) プロダクト 43
データ処理
 自動化の追加計画 53
 自動化の要件 50
データ・モデル、RODM の 87
テープ管理 42
定義
 コマンド改訂テーブル 149
 自動化テーブル 163
 自動タスク 340
 条件項目 181
 比較項目 181
 メッセージ改訂テーブル 139
 リテラル、比較項目 235
 Always ステートメント 164
 BEGIN-END セクション 164
 DoForeignFrom ステートメント 139
 END ステートメント 139, 150
 EXIT ステートメント 140, 150
 IF-THEN ステートメント 163
 NetView 自動操作 3
 NetView を z/OS システムにサブシステムとして 336
 NETVONLY ステートメント 140, 150
 OTHERWISE ステートメント 140, 150
 REVISE ステートメント 140, 150
 SELECT ステートメント 140, 149
 SYN ステートメント 163
 UPON ステートメント 140, 149
 WHEN ステートメント 140, 150
 WTO ステートメント 150
 %INCLUDE ステートメント 164
定義 (動的)
 EMCS コンソール 337
定義、自動操作プロジェクトの 47, 580
停止
 MVS コマンド
 NetView への送信 643
 MVS コマンド出口
 呼び出し 644
ディスパッチング優先順位 522
ディレクトリー名の表記 xxviii
テキストの位置
 検索 364
テキストの比較
 構文解析テンプレートを使用する 366
出口 31
出口 16 はメッセージに対する制御権を受け取り、出口 16B
 は制御権を受け取る 31
テスト 63
 基本自動操作サンプル・セット 660
 自動化ステートメント 380
 リソース階層 378
テスト、構文の
 サンプル自動化テーブル 659
テスト、コマンド改訂 161
テスト、自動化の 531

テスト、メッセージ改訂 146
デバッグ補助 543
デフォルトのコマンド接頭部文字、パーセント記号 (%) の
 594
デフォルトの論理演算子
 メッセージ 260
 ロギング 552
 ALWAYS および CONTINUE の設定 272
 MSU 261
転送
 アラート 343, 444, 457
 オプション 449
 コマンド 445
 状態情報 423
 方式の選択 449
 メッセージ
 イベント自動化サービス 457
 概要 445
例外 423
2 つの NetView プログラム間での 520
z/OS から NetView プログラムへのシステム・メッセージ
 336
 サブシステム・インターフェースの使用 336
z/OS システムから NetView プログラムへのシステム・メ
 ッセージ
 EMCS コンソールの使用 336
 z/OS システムから NetView へのシステム・メッセージ
 サブシステム・インターフェースの使用 336
転送、例外の 17
伝搬
 設計 58
 単一システムの自動操作 16
トークン
 メッセージの検索に使用する 363
同期、システム間での自動化の 419
同義語
 名前
 パーセント記号なし 266
 パーセントなし 266
 変数
 ; なし 266
統合
 コマンド 11
 コンソール 10
 RODM による自動化 13
動的オペレーター制御 100
動的定義、EMCS コンソールの 337
ドメイン ID
 検索 363
トランスポート、MS および高性能
 アラートの送信 414
トリガー・イベント 391
トレース
 TRACE キーワード、IF-THEN または ALWAYS ステート
 メントの 258
トレース、自動化テーブルの 545

[ナ行]

流れ

メッセージ

MVS 589

VTAM 601

メッセージ経路指定 103

許可レシーバーの処理 105

ASSIGN PRI/SEC 処理 104

DSIEX02A 処理 105

DSIEX17 処理 104

PIPE CORRWAIT 104

名前変更、自動操作のサンプル・セットの 652

入手、環境情報の 27

入出力管理 42

ヌル ("")

解析テンプレート

IF-THEN 比較項目 239

ビット・ストリング IF-THEN 比較項目 235

ネットワーク

可用性

会計上の値 55

設計 60

利点 4

管理ベクトル・トランスポート (NMVT) 37

コマンド 7

自動化 6

パフォーマンス管理 41

メッセージ 7

ログ 51

MSU 7

ネットワーク管理コンソール・ビュー

スケジュールの定義

NMCSTATUS ポリシー定義に基づいて 301

ネットワーク・メッセージの抑制 341

ネットワーク・ログ 519, 551

能動モニター 12, 663

拡張自動操作サンプル・セット 663

拡張対等通信ネットワーク機能のリソース 508

基本自動操作サンプル・セット 658

コマンド・リスト

拡張自動操作サンプル・セット 669

説明 408

Tivoli NetView (AIX) 用の 510

[ハ行]

ハードウェア・モニター 14

オペレーター・インターフェース 413

継続処理 112

自動化通知ログの理解 502

初期処理 111

チューニングに関する考慮事項 530

データ レコード 369

フィルター 342

ハードウェア・モニター・データおよび MSU

インターフェース 93

廃棄または表示

メッセージ 108

排他リスト、MVS

開始 645

変更 645

パススルー・ルール 397

バス名の表記 xxviii

バックアップ・フォーカル・ポイント 66

パネル、フルスクリーンまたはヘルプの 415

パネルおよびコマンド・リスト

オペレーター・インターフェース

拡張自動操作サンプル・セット 677

自動操作表示パネル

拡張自動操作サンプル・セット 678

パフォーマンス管理、ネットワークの 41

パラメーターおよびコマンド・プロシージャ 27

非 NetView システム

インターフェース 41

比較、構文解析テンプレートを使用するテキストの 366

比較項目

定義 181

比較項目、IF-THEN ステートメントの

解析テンプレート 235

ビット・ストリング 234

非活動化

MVS コマンド出口 644

非持続セッション 452

非自動化メッセージおよび MSU の評価 545

非送信請求メッセージ

許可レシーバー 95

経路指定のための ASSIGN コマンド 97

除去のための ASSIGN コマンド 99

メッセージ経路指定 95

ビット表記

MSU アクション 374

ビット・ストリング比較項目

IF-THEN ステートメント 374

ビット・ストリング比較項目、IF-THEN ステートメントの
234

表記

環境変数 xxviii

書体 xxviii

バス名 xxviii

表示

NCP リカバリー定義 507

表示、情報の 16, 411

表示コマンド

MVS コマンド管理設定 643

標準の確立 58

ファイル

tecsce.dtd 388

フィールドが 2 度以上出現する

MSU に 375

- フィールドの存在
 - 検査 369
- フィールドの内容
 - 選択 371
- フィルター 596
 - 迂回 346
 - 記録 342
 - 説明 342
 - ビュー 346
- フィルター操作、アラートの 9
- フィルター操作、イベントの
 - 状態相関 387
- フォーカル・ポイント
 - 概要 423
 - 集中操作 18
 - 選択 64
 - 中間 447
 - バックアップ 66
 - 複数の使用 452
 - 変更、除去、およびリスト 453
 - 例外の転送 17
- 複数
 - 自動タスク、NetView の 528
 - NetView システム 528
 - NetView、システム 513
- 複数 NetView プログラムの設計 61
- 複数オカレンス、フィールドの
 - MSU での検索 375
- 複数行メッセージ
 - 構文解析テンプレートの使用 367
- 複数コンソール・サポート
 - 自動タスクとの関連付け 353
 - 発行する、NetView コマンドおよびコマンド・リストを
 - MODIFY コマンドとして 598
 - 発行する、NetView コマンドおよびコマンド・リストをサ
 - ブシステム・コマンドとして 597
 - NetView コンソールとの相違点 350
- 複数コンソール・サポート (MCS) コンソール
 - 自動タスクとの関連付け 134
- 複数サイト構成の例 65
- 複数システムの自動操作
 - 設計のガイドライン 61
 - 定義 8
- 複数ドメイン・サポート・メッセージ単位 (MDS-MU)
 - NMVT (ネットワーク管理ベクトル・トランスポート) 368
- 複数リソース
 - リソース階層 378
- 複製、ルールの 400
- ブック
 - 資料を参照 xxi
- 部門、自動化
 - 教育 62
 - 編成 48
 - 役割 63
- フラッシュ・メッセージ 362
- ブランク行
 - MLWTO メッセージの先頭に 173
- プリロード・コマンド・リスト 125
- プリンター管理 43
- フルスクリーン
 - 集中操作 448
 - 表示パネル 415
 - TAF セッション 448, 485
- フルスクリーン表示 16
- ブレースホルダー
 - 構文解析テンプレートで 366
 - ピリオド (ブレースホルダー) 365
 - IF-THEN ステートメントの解析テンプレート比較項目 238
- プログラム、自動操作 25
- プログラム間インターフェース
 - アラートの送信 413
 - MVS システム 37
- プログラム言語/I (PL/I) 26
- プログラム・オペレーター・インターフェース (POI) 93
- プロシージャ・ブック 52
- プロジェクト、実動の 69
- プロジェクト定義フェーズ 47, 580
- プロダクト、自動操作 25
- ブロック
 - 自動化テーブル 287
- ブロック、アラートの 9
- 分散システム、定義 18
- 分散ネットワーク 40
- 文書化、コマンド・プロシージャの 357
- 分析、ログの 51
- ページ 416
- ヘッダー
 - MDS の検査 376
 - MDS-MU 376
 - RECMS 371
- ヘルプ・デスク
 - AON/SNA 506
- ヘルプ・デスク・ログ 53
- ヘルプ・パネル 415
- 変更、フォーカル・ポイントの 453
- 変更管理 64
- 変数
 - グローバル 121
 - 構文解析テンプレートで 367
 - 状態 405
 - 比較項目、IF-THEN ステートメントの
 - 文字 237
 - 16 進数 237
 - メッセージおよび MSU からの情報の抽出 366
 - R&TI GDS の検査 377
 - %INCLUDE ステートメントの 264
- 変数値
 - 比較項目、IF-THEN ステートメントの
 - 説明 237
- 変数の表記 xxviii

変数への値の割り当て

基本自動操作サンプル・セット 656

変数名

比較項目、IF-THEN ステートメントの

説明 236

ポータブル自動化テーブル 420

包含リスト、MVS

開始 645

変更 645

報告書管理配布システム (RMDS) プログラム 43

防止

MVS コマンド出口

呼び出し 644

保管

自動化テーブル・ステートメント 164

情報 121

タイマー・コマンド 130

補助モード 464

ポリシー定義

グループの照会 310

削除 312

照会 310

追加 312

変更 311

ポリシー・サービス 301

管理 305

構文 303

使用 302

定義 302

ポリシー・ファイル

構文テスト 308

ロード 309

ロードされる 308

ポリシー・リポジトリ 303

[マ行]

マイグレーション、新規自動化機能への

NetView for OS/390 V1R4 578

NetView for z/OS V5R4 578

NetView for z/OS V6R2 577

マニュアル

資料を参照 xxi

無許可プログラム 174

無人

操作 423

命名規則

コマンド・リスト

拡張自動操作サンプル・セット 669

メソッド、RODM の 30, 87

メッセージ

アクション・メッセージ 363

位置による検索 364

オペレーターへの割り当て 96

改訂 139

拡張複数コンソール・サポート・コンソールの損失 78

メッセージ (続き)

強調表示 30

経路指定の流れ 103

許可レシーバーの処理 105

ASSIGN PRI/SEC 処理 104

DSIEX02A 処理 105

DSIEX17 処理 104

PIPE CORRWAIT 104

検索、ドメイン ID による 363

システム 6

自動応答 12

自動化 362

自動化、応答の 33

自動化テーブル 29

自動化を指定するためにトークンを使用する 363

シミュレート 538

情報の提示 14

相関 381, 383

タイプ 6

近い処理、ソースに 61

デフォルト 260

転送 459

概要 445

統合 10

流れ

MVS 589

VTAM 601

ネットワーク 7

廃棄または表示 108

複数行 173

フラッシュ 362

メッセージ・タイプ (HDRMTYPE) 627

メッセージ・タイプの自動化テーブル・ステートメント

165

要約報告書 285

抑止

サンプル・セット 653

MVS メッセージ 341

VTAM メッセージ 602

抑制または改訂 9

例外通知 411

ロギング 552

ASSIGN コマンドによる経路指定 96

BNJ146I 345, 379

EMCS コンソールへの経路指定 102

IF-THEN ステートメントの条件項目 175

MPF テーブル 76

MSGROUTE コマンドによる経路指定 102

MVS オペレーター・コンソールへの送信 125

MVS システム 33, 79

メッセージ ID

検索 362

メッセージあふれ防止テーブル 602

メッセージおよび MSU

自動化処理 361

IF-THEN ステートメントの条件項目 175

メッセージ改訂テーブル

概要 30

検索 140

コーディング 141

コーディング、ステートメントの

DoForeignFrom 142

END 142

EXIT 142

NETVONLY 142

OTHERWISE 143

REVISE 143

SELECT 144

UPON 144

WHEN 145

コメント 141

使用 139

処理 140

テスト 146

メッセージの識別、自動化の 33

メッセージ改訂テーブル・ステートメント

エレメント

DoForeignFrom ステートメント 139

END ステートメント 139

EXIT ステートメント 140

INCLUDE ステートメント 140

NETVONLY ステートメント 140

OTHERWISE ステートメント 140

REVISE ステートメント 140

SELECT ステートメント 140

UPON ステートメント 140

WHEN ステートメント 140

メッセージ経路指定 94

自動タスクのメッセージのための ASSIGN コマンド 100

送信請求メッセージ 94

送信請求メッセージのための ASSIGN コマンド 99

非送信請求メッセージ 95

許可レシーバー 95

非送信請求メッセージのための ASSIGN コマンド 97

非送信請求メッセージを除去する ASSIGN コマンド 99

割り当てられた宛先の検証

自動化ロジックによる ASSIGN コマンドの使用 100

DST からの非送信請求メッセージ 95

MVS システムからの非送信請求メッセージ 95

メッセージ経路指定機能 96

メッセージ自動化 34

メッセージ処理機能 (MPF)

オプションの設定、メッセージ処理の 34

拡張複数コンソール・サポート・コンソール 76

メッセージ属性

設定 108

メッセージ属性の設定 108

メッセージのグループ

検索 365

検索、論理 OR 論理による 365

プレースホルダーの使用による 366

論理 AND 論理による検索 365

メッセージのグループの検索 365

論理 AND 論理 365

論理 OR 論理による 365

メッセージの検索

位置による 364

トークンによる 363

ドメイン ID による 363

プレースホルダーによる 365

メッセージ ID による 362

メッセージの流れ、NetView プログラム

MVS

サブシステム・インターフェース 35

メッセージ抑止サンプル・セット

サンプル・セット 653, 695

メッセージ率 4

メッセージ・タイプの自動化テーブル・ステートメント 165

メッセージ・テーブル 29

メンバーを組み込む (%INCLUDE)

保守 420

目視フィルター 346

目的

事業 49

自動化 54

進行度の測定 54

測定方法の例 587

データ処理 50

目標

事業 49

自動化 54

進行度の測定 54

測定方法の例 587

データ処理 50

文字

変数 237

リテラル 235, 374

文字表記

MSU アクション 374

モニター

コンポーネント

グラフィック 15

状況 15

ハードウェア 14

受動 408

拡張自動操作 662

タイプ

受動 12

能動 12

能動 408

拡張自動操作 663

能動と受動の結合 409

リソース状況 12

Tivoli NetView (AIX) 用の能動 510

Tivoli NetView (AIX) 用のリカバリー 510

モニター、拡張対等通信ネットワーキング機能のリソースの
508

問題

管理 64
通知 14
転送 17
報告書 52

[ヤ行]

役割

オペレーターの 63
自動操作プロダクトの 25
ユーザー作成ログ 551
ユーザー指向の要件 50
ユーザー定義の MS フォーカル・ポイント・カテゴリ 443
ユーザー入力 53
ユーザー・グループ

NetView、Yahoo での xxvii
Tivoli xxvi

有効な指定、テーブルの

THRESHOLD 230

優先順位

キューに入れられたコマンド 115
タスク 527
ディスパッチング 522

要件

事業 49
システム指向の 50
自動化 54
進行度の測定 54
測定方法の例 587
データ処理 50
ユーザー指向の 50

要約アクション 391

抑止、メッセージの 9, 341, 653

読みやすく作成

自動化テーブル 267

[ラ行]

ラック・マウントの ES/9000 の初期設定 21

ラベル

自動化テーブル 287

ラベル、コマンド接頭部 115

利益算出用ワークシート 55

理解

自動化追跡 501
自動化通知ログ
ハードウェア・モニターでの 502
通知 501
AON の自動化操作プログラム 500
AON の自動化とリカバリー 499
AON/SNA オプション 505

リカバリー

概要 409

リカバリー (続き)

拡張自動操作

サンプル・セット 665

コマンド・リスト

拡張自動操作サンプル・セット 673

NetView 522

リカバリー・モニター

Tivoli NetView (AIX) 用の 510

リスト、自動化テーブルの

デバッグ 545

例 275

AUTOTBL コマンド 271

リセット・オン・マッチ・ルール 399

リソース

モニター、拡張対等通信ネットワーク機能の 508

リソース NMC ビュー

スケジュールの定義

NMCSTATUS ポリシー定義に基づいて 301

リソース階層 377

テスト 378

DEVLAN3 377

LANMGR 378

リソース・オブジェクト・データ・マネージャー (RODM)

イベント、自動化の 89

自動化 87

自動操作の計画 89

自動操作の統合 13

紹介 30, 87

対話 87, 88

データ・モデル 87

メソッド・プロシージャ (メソッド) 30, 87

利点 88, 89

リソース・リカバリー 502

リソース・リスト

NetStat 507

リテラル、比較項目

IF-THEN ステートメント

説明 235

文字 235

16 進数 236

IF-THEN ステートメント、文字表記 374

IF-THEN ステートメント、16 進表記 374

利点、自動化の

概要 3

分析 54

リモート

確立 19

初期設定 19

操作 8, 423

リモート・オペレーター機能 (ROF)、9370 の 20

理由、自動化の 3

両方のタイプの自動化テーブル・ステートメント 165

ルール

状態相関

記述された 392

コレクター 396

ルール (続き)

状態相関 (続き)

しきい値 394

重複 393

突き合わせ 392

複製 400

例

IF-THEN ステートメント、結合される条件

論理 OR および論理 AND 演算子 173

例、自動化テーブル機能の

OPERID 182

例外

オペレーターへの通知 411

通知 13

転送 17, 423

レプリケーション主ベクトル 376

ロード、コマンド・リストのストレージへの 125, 691

ロギング

概要 549

NETLOG キーワード、IF-THEN または ALWAYS ステートメントの 253

SYSLOG キーワード、IF-THEN または ALWAYS ステートメントの 258

ログ

使用、自動化テーブルの 29

情報の入手 51

設計、自動操作の 59

デバッグ 543

ネットワーク

説明 551

ログ・ブラウズ 519

MVS システム・ログ 553

分析、プログラムの 51

ヘルプ・デスク 53

ユーザー提供 551

MVS システム 550, 553

ログ分析プログラム

サンプル・セット 653, 695

出力 524

チューニング 523

フィルター操作 525

[ワ行]

ワークロード管理 41

割合、情報の 4

割り当て、オペレーターのグループへの 97

割り当て、メッセージのオペレーターへの 96

[数字]

1044 主ベクトル 371

1045 主ベクトル 371

16 進数

変数名 237

16 進数 (続き)

リテラル 236, 374

16 進表記

MSU アクション 374

2 つの NetView プログラムの設計 61

3480 カートリッジ 42

9370 システム 20

9370 プロセッサ、リモートからの初期設定 20

A

ACQUIRE、IF-THEN ステートメントの 179

AFTER コマンド 28, 127

AIFR 174

AIFR、記録 532

AIP 414

ALERTPCT 338

ALERTPCT 属性 79

ALERT-NETOP 111

ALL キーワード、IF-THEN または ALWAYS ステートメントの EXEC アクションの 247

allEvents モード 394

ALLOCATE コマンド 125, 551

ALWAYS ステートメント

タイプ 165

Always ステートメント

定義 164

ALWAYS ステートメント、自動化テーブル

構文 263

設計のガイドライン 271, 272

ALWAYS ステートメント、自動化テーブルの
テーブルのアクション 240

AMRF (アクション・メッセージ保持機能) 590

AND 演算子

結合される条件

論理 AND 演算子 172, 173

AON TIMER コマンド 28

AON 制御ファイル 500

AON、機能 499

AON/SNA

NCP リカバリー定義 507

NetStat 507

SNAMAP 506

VTAM オプションの管理 507

VTAM コマンドの発行 507

X.25 スイッチド・パーチャル・サーキット 507

AON/SNA X.25 モニター・サポート 508

AON/SNA オプション 505

AON/SNA 自動化

概要 504

AON/SNA チュートリアル 506

AON/SNA ヘルプ・デスク 506

AON/SNA リソースのリスト

NetStat 507

AON/TCP 509

AON/TCP (続き)
z/OS TCP/IP の場合のみの MIB ポーリングおよびしきい値
分け 510
AON/TCP インターフェース
更新を受け取る場合の選択 511
APC (自動電源制御)、9370 の 20
API (アプリケーション・プログラム・インターフェース) 30
AREC
フィルター 342
IF-THEN または ALWAYS ステートメントの SRF アクシ
ョンのキーワード 255
ASID 180
ASSIGN COPY 処理 108
ASSIGN PRI/SEC
メッセージの経路指定の流れ 104
ASSIGN コマンド
宛先への経路指定検証のために使用 100
送信請求メッセージの経路指定の 99
動的オペレーター制御のために使用 100
と自動化テーブル経路指定 101
非送信請求メッセージの経路指定の 97
非送信請求メッセージの除去の 99
メッセージの経路指定のための使用 96
メッセージの自動タスクへの経路指定の 100
ASSISCMD コマンド 464, 470
AT コマンド 28, 128
ATF (自動化テーブル機能) 182
自動化テーブル機能 (ATF)
DSITGLOB 183
説明 181
DSICGLOB 183
AUTOCNT コマンド 30, 275, 531
AUTOMAN
活動化、自動化テーブルの 287
自動化テーブルに対するアクションで使用 164
INCLUDE 構造の表示に使用 164
AUTOMAN コマンド
活動化、自動化テーブルの 381
AUTOMATED
IF-THEN または ALWAYS ステートメント
説明 241
AUTOMATED、IF-THEN ステートメント 185
AUTOTASK コマンド 134, 660
AUTOTASK、IF-THEN ステートメント 186
AUTOTBL
活動化、自動化テーブルの 287, 380
自動化テーブルに対するアクションで使用 164
AUTOTBL コマンド
セキュリティ・アプリケーション 682
AUTOTBLE 380
AUTOTEST コマンド 531

B

BEEP 373
IF-THEN または ALWAYS ステートメント 241

722 自動操作ガイド

BEGIN キーワード、IF-THEN または ALWAYS ステートメン
トの
ALWAYS ステートメント 263
BEGIN、IF-THEN または ALWAYS ステートメント
説明 168
IF-THEN ステートメント 172
BEGIN-END セクション
構文
自動化テーブル・ステートメント 168
設計のガイドライン 267
タイプ 166
定義 164
BIT
ATF 条件項目、IF-THEN ステートメント 181
BLI キーワード、IF-THEN または ALWAYS ステートメント
の XHILITE アクションの 258
BLOCK キーワード、IF-THEN または ALWAYS ステートメン
トの SRF アクションの 256
BLU
COLOR アクション
IF-THEN または ALWAYS ステートメント 242
BNJ146I メッセージ 379
BOTH キーワード、IF-THEN または ALWAYS ステートメン
トの SRF アクションの 256

C

C 言語 26
CBE 241
CHRON コマンド 28, 129
検証 541
CLOSE コマンド 34
CMC (コミュニケーション管理構成) システム、フォーカル・
ポイントとしての 65
CMD キーワード、IF-THEN または ALWAYS ステートメント
の EXEC アクションの 245
CMDDEF ステートメント
コマンド・プロセッサ 356
NetView プログラムからシステム・コマンドを出すための
追加 339
CNMAUTO サービス・ルーチン 539
CNMCAUaa PARMLIB メンバーの削除
MVS コマンド管理の停止 643
CNMCMD
CMDDEF ステートメントのセットアップ 339
CNMCRMSG 384
CNME7023 451
CNMI (通信ネットワーク管理インターフェース)
非送信請求ネットワーク管理データ 515
CNMPROC プロシージャ
CNMSJ009 339
CNMPSSI プロシージャ
CNMSJ010 339
CNMS6207 サンプル 652
CNMSCRT1 サンプル 159

CNMSMSG サービス・ルーチン
 経路指定コマンド 114
CNMSRVMC サンプル 157
CNMSTYLE 303
COLOR 373
 IF-THEN または ALWAYS ステートメント 242
Common Base Event
 作成 493
 自動化 493
 紹介 493
CONTINUE
 アクション 272
 IF-THEN または ALWAYS ステートメント 243
 設計のガイドライン 272
CONTINUE(Y) 164
COREVENT 384
CP-MSU (制御ポイント管理サービス単位)
 概念視点 368
CRT (コマンド改訂テーブル)
 使用 149
 制限 514
CTC 555
CURRDATE、IF-THEN ステートメントの 178, 187
CURRTIME、IF-THEN ステートメントの 178, 187

D

DASD 管理 42
DB2 490
DBCS ストリング
 コマンド 245
 自動化テーブル 167
DEFAULTS コマンド
 メッセージ属性 260
 ロギング 543, 552
DELAY コマンド 128
DELMSG 244
DEVLAN3
 リソース階層 377
DFILTER コマンド 346
DFSMS (データ機能記憶管理サブシステム) プロダクト 43
DIALCDRM コマンド・リスト 451
DISPLAY
 IF-THEN または ALWAYS ステートメント 243
DoForeignFrom ステートメント
 コーディング、メッセージ改訂テーブルの 142
 定義 139
DOM コマンド 39, 125
DOMACTION
 IF-THEN または ALWAYS ステートメント 243
DOMAINID キーワード、IF-THEN ステートメントの
 例 378
DROPCL コマンド 126, 691
DSI6DST 434
DSIAUTO マクロ 539

DSICGLOB
 自動化テーブル機能
 タスク・グローバル変数 121
DSICGLOB 自動化テーブル機能 183
DSIEX02A インストール・システム出口 31, 329
DSIEX02A 処理
 メッセージの経路指定の流れ 105
DSIEX16 108
DSIEX16 インストール・システム出口 330
DSIEX16B インストール・システム出口 330
DSIEX16B インストール・システム出口 MSU 31
DSIEX16b 処理 112
DSIEX17 31, 330
 メッセージの経路指定の流れ 104
DSIMQS マクロ
 経路指定コマンド 114
DSINOR 463, 472
DSIOPF メンバー 60
DSIPARM メンバー
 自動化テーブル・ステートメントの保管 164
DSIQTSK タスク 462
DSIQTSKI メンバー 462
DSITGLOB 自動化テーブル機能 183
DSIWLS 552
DST からの非送信請求メッセージ 95
DUIFECMV コマンド・プロセッサ 521

E

E メール 416
EDIT
 IF-THEN または ALWAYS ステートメント 245
EIF イベント 459
EKGSPPI
 オブジェクト独立メソッドの起動 484
 オブジェクトの照会 483
 サブフィールド変更 481
 初期設定 480
 定数 479
 フィールドの照会 482
 変更メソッド 462, 477
 ローカル変数 477
EMCS (拡張複数コンソール・サポート) コンソール
 発行する、NetView コマンドおよびコマンド・リストを
 MODIFY コマンドとして 598
 発行する、NetView コマンドおよびコマンド・リストをサ
 ブシステム・コマンドとして 597
EMCS コンソール
 使用のための計画
 メッセージ損失 78, 79
 メッセージ・キューの制限 79
 動的定義 337
 NetView での使用 73
END ステートメント
 コーディング、コマンド改訂テーブルの 155
 コーディング、メッセージ改訂テーブルの 142

END ステートメント (続き)
定義 139, 150
ENDLABEL 170
ESREC
フィルター 342
IF-THEN または ALWAYS ステートメントの SRF アクシ
ョンのキーワード 255
ES/9000 の初期設定、ラック・マウントの 21
EVERY コマンド 28, 128
EXCMD
タスクへの経路指定 114
ラベル 115
EXEC アクション、IF-THEN または ALWAYS ステートメン
ト 245
EXIT ステートメント
コーディング、メッセージ改訂テーブルの 142
定義 140, 150
EZLEQAPI 322
EZLEQCAL 326
EZLETAPI 313
E/AS 40

F

firstEvent モード 394
FLSCN (フルスクリーン) TAF セッション 448, 485
FOCALPT コマンド 453
forwardEvents モード 394
FREE コマンド 551

G

GDS 変数
R&TI 377
GENALERT コマンド 15
アラートの送信 414
複数の NetView プログラム 518
GETCONID コマンド 75, 337
GETCONID パラメーター
ALERTPCT 338
QLIMIT 338
QRESUME 338
STORAGE 338
GMFHS (グラフィック・モニター機能ホスト・サブシステム)
データ・モデル 87
GMFHSDOM パラメーター、DUIFECMV の 521
GRE
COLOR アクション
IF-THEN または ALWAYS ステートメント 242
GROUP 170
GROUP オプション 97

H

H キーワード
MSUSEG 条件項目
IF-THEN ステートメント 223
H キーワード、MSUSEG 条件項目、IF-THEN ステートメント
の 376
HCYLOG キーワード
経路指定コマンド 250
IF-THEN または ALWAYS ステートメント 252
HDRMTYPE
値の説明 627
HELP コマンド 16
HIER キーワード
IF-THEN ステートメント 377
HIGHINT 373
HIGHINT キーワード、IF-THEN または ALWAYS ステートメン
トの 252
HLL (高水準言語 - PL/I および C) 26
HMCPLINK、IF-THEN ステートメントの 177, 192
HMSPECAU、IF-THEN ステートメントの 177, 202
HOLD キーワード、IF-THEN または ALWAYS ステートメン
トの 250

I

IBM 以外のネットワーク 41
id 属性 392
IF-THEN ステートメント
グループ化の順序
条件 173
結合される条件
論理 AND 演算子 172, 173
論理 OR および論理 AND 演算子、例 173
構文 169
条件項目
定義 181
メッセージおよび MSU に関して 175
メッセージの 175
MSU に関して 175
タイプ 165
テーブルのアクション 240
定義 163
比較項目
定義 181
IF-THEN ステートメントの ACTIONDL 179
IF-THEN ステートメントの ACTIONMG 180
IF-THEN ステートメントの AREAID 181
IF-THEN ステートメントの ATTENDED 184
IF-THEN ステートメントの AUTOTOKE 186
IF-THEN ステートメントの CART 186
IF-THEN ステートメントの CORRELATED 187
IF-THEN ステートメントの CORRFAIL 187
IF-THEN ステートメントの CURSYS 178, 187
IF-THEN ステートメントの DESC 188
IF-THEN ステートメントの DISTAUTO 178, 188

IF-THEN ステートメントの DOMAIN 178, 188
 IF-THEN ステートメントの DOMAINID
 使用法 189
 要約 178
 IF-THEN ステートメントの HDRMTYPE 178, 189
 IF-THEN ステートメントの HIER 189
 IF-THEN ステートメントの HMASPRID 177, 190
 IF-THEN ステートメントの HMBLKACT 177, 191
 IF-THEN ステートメントの HMEVTYPE 177, 195
 IF-THEN ステートメントの HMFWDDED 177, 196
 IF-THEN ステートメントの HMGENCAU 177, 198
 IF-THEN ステートメントの HMONMSU 177, 200
 IF-THEN ステートメントの HMORIGIN 177, 200
 IF-THEN ステートメントの HMSECREC 177, 201
 IF-THEN ステートメントの HMUSRDAT 177, 203
 IF-THEN ステートメントの IF 168, 170
 IF-THEN ステートメントの IFRAUI3X 178
 IF-THEN ステートメントの IFRAUIN3 178, 205
 IF-THEN ステートメントの IFRAUIND 178, 204
 IF-THEN ステートメントの IFRAUSB2 178, 206
 IF-THEN ステートメントの IFRAUSC2 178, 206
 IF-THEN ステートメントの IFRAUSDR 178, 207
 IF-THEN ステートメントの IFRAUSRB 178, 207
 IF-THEN ステートメントの IFRAUSRC 178, 207
 IF-THEN ステートメントの IFRAUTA1 178, 208
 IF-THEN ステートメントの IFRAUWF1 209
 IF-THEN ステートメントの INTERVAL 179, 209
 IF-THEN ステートメントの JOBNAME 210
 IF-THEN ステートメントの KEY 211
 IF-THEN ステートメントの LINEPRES 211
 IF-THEN ステートメントの LINETFLG 214
 IF-THEN ステートメントの MCSFLAG 214
 IF-THEN ステートメントの MSGAUTH 215
 IF-THEN ステートメントの MSGCATTR 216
 IF-THEN ステートメントの MSGCMISC 216
 IF-THEN ステートメントの MSGCMLVL 217
 IF-THEN ステートメントの MSGCMSGT 217
 IF-THEN ステートメントの MSGCOJBN 218
 IF-THEN ステートメントの MSGCPROD 218
 IF-THEN ステートメントの MSGDOMFL 219
 IF-THEN ステートメントの MSGGBGPA 219
 IF-THEN ステートメントの MSGGDATE 220
 IF-THEN ステートメントの MSGGFGPA 220
 IF-THEN ステートメントの MSGGMFLG 221
 IF-THEN ステートメントの MSGGMID 221
 IF-THEN ステートメントの MSGGTIME 221
 IF-THEN ステートメントの MSGID 222
 IF-THEN ステートメントの MSGSRCNM 222
 IF-THEN ステートメントの MSUSEG 223, 225
 IF-THEN ステートメントの MVSLEVEL 224
 IF-THEN ステートメントの NETID 225
 IF-THEN ステートメントの NETVIEW 179
 IF-THEN ステートメントの NETVIEW プログラム 225
 IF-THEN ステートメントの NETVIEW、プログラム 179
 IF-THEN ステートメントの OPSYSTEM 179, 227
 IF-THEN ステートメントの ROUTCDE 227

IF-THEN ステートメントの SESSID 227
 IF-THEN ステートメントの SYSCONID 228
 IF-THEN ステートメントの SYSID 228
 IF-THEN ステートメントの TASK 179, 229
 IF-THEN ステートメントの TEXT 229
 IF-THEN ステートメントの THEN 168, 172
 IF-THEN ステートメントの THRESHOLD 179, 230
 IF-THEN ステートメントの TOKEN 232
 IF-THEN ステートメントの VTAM 179, 233
 IF-THEN ステートメントの VTCOMPID 233
 IF-THEN ステートメントの WEEKDAYN 233
 IF-THEN ステートメントの解析テンプレート比較項目 235
 IHSAAACDS 459
 IHSABACDS 459
 IHSALCDS 459
 IHSAMFMT 459
 IHSAMFNT 459
 IMS (情報管理システム) 490
 INCLUDE 構造 164
 INCLUDE ステートメント
 コマンド改訂テーブルの 150
 メッセージ改訂テーブルの 140
 IP ネットワーク 40
 ISSUE.IEE295I ステートメント
 コーディング、コマンド改訂テーブルの 152

J

JES より先に NetView を開始する
 拡張自動操作
 サンプル・セット 680
 JES3 の自動化 555
 JES、NetView のあとに開始する 680

L

LABEL 170
 LANMGR
 リソース階層 378
 lastEvent モード 394
 LIST TIMER コマンド 28, 131
 LOADCL コマンド 126, 691
 LOG 宛先、メッセージの 552
 LU 6.2 セッションへのマイグレーション 80
 LU 6.2 トランスポート 15
 アラートの送信 414
 LUC セッション
 アラート転送 444

M

MAPCL コマンド 126
 MCS コンソール、拡張
 動的定義
 GETCONID 337

MCS コンソール、拡張 (続き)

動的定義 (続き)

RELCONID 339

SETCONID 339

MCSFLAG

プログラム言語の比較 215

MDS-MU (複数ドメイン・サポート・メッセージ単位)

自動化 368

MDS-MU、自動化 376

MLWTO

先頭のブランク行 173

MPF (メッセージ処理機能) 589

オプションの設定、メッセージ処理の 34

拡張複数コンソール・サポート・コンソール 76

MRT (メッセージ改訂テーブル)

概要 30

使用 139

MS (管理サービス) および高性能トランスポート

アラートの送信 414

MS トランスポート 15

MSG 詳細報告書 283

MSGID キーワード、IF-THEN ステートメントの 362

MSGROUTE コマンド

メッセージの経路指定のための使用 102

MSU

自動化処理 361

選択 370

関連 381, 383

IF-THEN ステートメントの条件項目 175

MSU アクション

ビット表記 374

文字表記 374

16 進表記 374

BEEP 373

COLOR 373

HIGHINT 373

SRF 373

XHILITE 373

XLO 374

MSU (管理サービス単位)

アクション 373

アラート

アクション 373

MSU 以外の問題レコードの自動化 379

管理 services unit (MSU)

詳細報告書 283

要約報告書 285

自動化 367

シミュレート 539

デフォルト 261

MSU タイプの自動化テーブル・ステートメント 165

MSU サブベクトル

選択、フィールドの内容 371

MSU の経路指定 108

MSU の選択

例 370

MSUSEG

例 369

MS-CAPS アプリケーション 66, 427

MVS

オペレーター・コンソール 125

コマンド接頭部文字 594

コマンドの流れ 592

システム・ログ 550, 553

排他または包含リスト

開始 645

変更 645

メッセージ ID 362

メッセージ・フロー 589

MVS EMCS コンソール

使用のための計画

メッセージ損失 78

メッセージ・ストレージ 78

MVS 拡張複数コンソール・サポート・コンソール

考慮点 73

紹介 73

使用のための計画

経路コード 76

コンソールの獲得 75

コンソール命名規則 74

コンソールを使用可能にする 74

セキュリティ・アクセス 78

属性値 76

デフォルト値 76

メッセージ損失 79

メッセージ・キューの制限 79

メッセージ・ストレージ 79

MPF を使用したメッセージ送信 76

MRT を使用したメッセージ送信 76

マイグレーション

クロスドメイン通信 80

コンソール名 80

AUTO 属性 81

MVS VARY コマンド 80

利点 73

MVS からの非送信請求メッセージ 95

MVS コマンド 39

NetView への送信の停止 643

MVS コマンド管理 633

停止

CNMCAUaa PARMLIB メンバーの削除 643

MVS コマンド管理、停止 643

MVS コマンド管理、テスト 645

MVS コマンド管理設定

表示コマンド 643

MVS コマンド管理用の MPF 出口 633

MVS コマンド処理

開始 643

MVS コマンド出口

活動化 643

非活動化 644

呼び出しの停止 644

MVS コマンドの自動化 39
MVS サンプル・ロケーション 652
MVS のシスプレックス
システム間カップリング・ファミリティー (XCF) 83
自動操作の計画
集中化、NetView プログラム自動操作の 85
メッセージ経路指定 37, 84
MPF アクション 85
紹介 83
利点、自動化の 84
MVS メッセージ
無許可プログラムから出された 174
MVSCmdRevision ステートメント 157

N

NCP リカバリー定義
表示 507
NETLOG キーワード、IF-THEN または ALWAYS ステートメントの 253
NetStat 507
NetView
アプリケーション・アドレス・スペース 335, 593
アラート 413, 414
インストール・システム出口 329
開始プロシージャ 339
拡張 MCS コンソールの動的定義
GETCONID 337
RELCONID 339
SETCONID 339
グラフィック・モニター機能 15
コマンド改訂テーブル
使用 149
サブシステム・アドレス・スペース 335, 593
システム・コマンドを出すための CMDDEF ステートメントの追加 339
自動化の伝搬 419
複数コンソール・サポート・コンソールの動的定義 337
複数の NetView プログラムの実行、システム 513
メッセージ・タイプ 627
リカバリー 522
MVS コマンド管理 634
z/OS オペレーティング・システムにサブシステムとして定義 336
z/OS からのシステム・メッセージの転送の確認 336
サブシステム・インターフェースの使用 336
NetView (UNIX) サービス・ポイント・プログラム 40
NetView インターフェース
オペレーティング・システム 108
NetView および MVS
通信のセットアップ 653
NetView 管理コンソール 412
NetView 初期設定の準備
拡張自動操作
サンプル・セット 680
NetView とオペレーティング・システムの通信の確立 333
NetView パフォーマンス・モニター (NPM) プログラム 41
NetView プログラム
コマンド・プロシージャ 87
コマンド・リスト言語 26
自動化テーブル
使用 163
自動操作機能 25, 119
自動操作の定義 3
メッセージ改訂テーブル
使用 139
z/OS からのシステム・メッセージの転送の確認
EMCS コンソールの使用 336
NetView プログラムが稼働していないシステム
自動化 21
NetView プログラムからシステム・コマンドを出すための
CMDDEF ステートメントの追加 339
NetView プログラムと z/OS オペレーティング・システムの通信の確立 333
NetView プログラム・インターフェース 91
オペレーティング・システム 92
自動化テーブル
経路指定メッセージ 106
メッセージ属性の設定 108
メッセージの処理 106
メッセージの廃棄または表示 108
ASSIGN COPY 処理 108
DSIEX16 108
その他の NetView プログラム 93
ハードウェア・モニター・データおよび MSU 93
MVS からの非送信請求メッセージ 95
POI (プログラム・オペレーター・インターフェース) 93
待機処理 106
メッセージ経路指定機能 96, 102
ASSIGN コマンドによる経路指定 96
MSGROUTE コマンドによる経路指定 102
NetView プログラム・コマンド
経路指定 113
NetView プログラム・メッセージ経路指定 94
許可レシーバー 95
DST からの非送信請求メッセージ 95
非送信請求メッセージ 95
NetView メッセージ経路指定
送信請求メッセージ 94
NETVONLY ステートメント
コーディング、コマンド改訂テーブルの 156
コーディング、メッセージ改訂テーブルの 142
定義 140, 150
NMC
オペレーター・インターフェース 414
スケジュールの定義
NMCSTATUS ポリシー定義に基づいて 301
NMC ビュー
使用するアプリケーション 301

NMCSTATUS ポリシー定義
スケジュールの定義
NMC ビューのリソースのための 301

NMVT
概念視点 368
構造 368

NMVT (ネットワーク管理ベクトル・トランスポート) 37

NODELMSG 244

NONE キーワード、IF-THEN または ALWAYS ステートメントの XHILITE アクションの 258

NPM (NetView パフォーマンス・モニター) プログラム 41

NUMERIC
IF-THEN ステートメントの MSUSEG 225

NVCLOSE、IF-THEN ステートメント 179, 225

NVDELID、IF-THEN ステートメント 226

O

OEM システムおよび装置
インターフェース 41
自動化 21

OIV 414

ONE キーワード、IF-THEN または ALWAYS ステートメントの EXEC アクションの 247

OPCTL (オペレーター制御) TAF セッション 485

OPC/ESA (運用計画管理/ESA) プログラム
ワークロード管理での使用 41

OPER
フィルター 342
IF-THEN または ALWAYS ステートメントの SRF アクションのキーワード 255

OPERID
例、自動化テーブル機能の 182

OPID、IF-THEN ステートメント 179

OPI、IF-THEN ステートメント 226

ORCNTL コマンド 463

ORCONV
コマンド 466

OST 定義ステートメント
NetView DSIOPF での (自動タスクの場合) 340

OST-NNT セッション 446

OTHERWISE ステートメント
コーディング、コマンド改訂テーブルの 155
コーディング、メッセージ改訂テーブルの 143
定義 140, 150

OVERRIDE コマンド
メッセージ属性 260
ロギング 543, 552

P

PASS キーワード、IF-THEN または ALWAYS ステートメントの SRF アクションの 256

PIN
COLOR アクション
IF-THEN または ALWAYS ステートメント 242

PIPE CORRWAIT
メッセージの経路指定の流れ 104

PL/I (プログラム言語/I) 26

POI 515

POI (プログラム・オペレーター・インターフェース) 93, 515

PPT
タイマー・コマンド・オプション 129
IF-THEN または ALWAYS ステートメントの EXEC アクション 247

PRI
IF-THEN または ALWAYS ステートメントの SRF アクションのキーワード 256

PURGE TIMER コマンド 28, 131

Q

QLIMIT 338

QLIMIT 属性 79

QRESUME 338

R

RECFMS (保守用統計様式化レコード) 371

RECMS
選択 372

RECMS 82 371

RECMS と RECFMS
選択 371

RECMS (レコード保守統計) 371
カプセル化された 372

RED
COLOR アクション
IF-THEN または ALWAYS ステートメント 242

REFRESH コマンド
動的オペレーター制御のために使用 100

RELCONID コマンド 339

RESTORE コマンド 130

REV キーワード、IF-THEN または ALWAYS ステートメントの XHILITE アクションの 258

REVISE ステートメント
コーディング、コマンド改訂テーブルの 155
コーディング、メッセージ改訂テーブルの 143
定義 140, 150

REXX
コマンド・リスト・プロシーチャーの 119
統合、コマンドの 355

REXX API 313

REXX (再構造化拡張実行プログラム) 言語 26

RMDS (報告書管理配布システム) プログラム 43

RMTCMD
タスクへの経路指定 114
ラベル 115

RMTCMD コマンド
 コマンドの転送 445
RMTCMD コマンドへのマイグレーション 80
RODM 461
ROF (リモート・オペレーター機能)、9370 の 20
ROUTE
 コマンド 446
 フィルター 343, 345
 IF-THEN または ALWAYS ステートメントの EXEC アク
 ションのキーワード 246
 IF-THEN または ALWAYS ステートメントの SRF アクシ
 ョン 255
ROUTE キーワード
 自動化テーブルの
 コマンドの経路指定の 114
RTNDEF.BASE.AGENT ステートメント 85
R&TI GDS 変数 377

S

SAF プロダクトより先に NetView を開始する
 拡張自動操作
 サンプル・セット 682
SAVECMD コマンド 464
SEC キーワード、IF-THEN または ALWAYS ステートメント
 の SRF アクションの 256
SELECT ステートメント
 コーディング、コマンド改訂テーブルの 154
 コーディング、メッセージ改訂テーブルの 144
 定義 140, 149
service xxvi
Service Management Connect xxvi
SETCONID コマンド 75, 339
SLR (サービス水準報告) プログラム 43
SMC xxvi
SMS (記憶管理サブシステム) 43
MSGID 180
SNA サブエリア VTAM リソース自動化サポート 508
SNAMAP 506
SNMP
 トラップの自動化 563
SOC-MGR 66, 430
SRCLU (ソース LU) 486
SRF 373
 IF-THEN または ALWAYS ステートメントのキーワード
 255, 343
SRFILTER コマンド 9, 343
SSNT (サブシステム名テーブル) 590
STORAGE 338
SUBMIT コマンド 125
SVFILTER コマンド 346
SYN
 同義語
 statements, 例 273
SYN ステートメント 265
 設計のガイドライン 270
SYN ステートメント (続き)
 定義 163
synonym 値の区切り文字
 引用符 266
SYSLOG
 MVS システム 550
SYSLOG キーワード、IF-THEN または ALWAYS ステートメ
 ントの 258
SYSOP 宛先、メッセージの 262, 552

T

TAF セッション 485
TAF (端末アクセス機能)
 オプション 486
 集中操作 448
 説明 485
 VTAM インターフェース 595
TCB 180
TCP/IP
 自動化 509
 z/OS TCP/IP の場合のみの MIB ポーリングおよびしい値
 分け 510
TECROUTE
 フィルター 343, 345
 IF-THEN または ALWAYS ステートメントの SRF アクシ
 ョン 255
tecsce.dtd ファイル 388
THRESHOLD
 テーブルの有効な指定 230
thresholdCount 属性 394
timeInterval 属性 394
timeIntervalMode 属性 394
TIMER コマンド 129
Tivoli
 研修、技術 xxvi
 ユーザー・グループ xxvi
Tivoli NetView (AIX)
 能動モニター
 AON での 510
 リカバリー・モニター
 AON での 510
Tivoli ソフトウェア・インフォメーション・センター xxv
TOKEN キーワード、IF-THEN ステートメントの 363
TRACE キーワード、IF-THEN または ALWAYS ステートメン
 トの 258
TRACKING.ECHO ステートメント
 コーディング、コマンド改訂テーブルの 152
TRAPROUT
 フィルター 343, 346
 IF-THEN または ALWAYS ステートメントの SRF アクシ
 ョンのキーワード 256
triggerMode 属性 394
TUR
 COLOR アクション
 IF-THEN または ALWAYS ステートメント 242

U

UND キーワード、IF-THEN または ALWAYS ステートメント
の XHILITE アクションの 258

UPON ステートメント

コーディング、コマンド改訂テーブルの 153

コーディング、メッセージ改訂テーブルの 144

定義 140, 149

V

VALUE、IF-THEN ステートメントの 179, 232

VIEW コマンド 16

VTAM

オプションの管理 507

コマンドの発行 507

メッセージあふれ防止テーブル 602

メッセージおよびコマンド処理 601

メッセージの抑止 602

リソース自動化サポート、SNA サブエリア 508

VTAM ブロック 174

VTAM より先に NetView を開始する

拡張自動操作

サンプル・セット 682

VTAM、NetView のあとに開始する 682

W

WHEN ステートメント

コーディング、コマンド改訂テーブルの 154

コーディング、メッセージ改訂テーブルの 145

定義 140, 150

WHI キーワード、IF-THEN または ALWAYS ステートメント
の COLOR アクションの 243

WTO コマンド 125, 589

WTO ステートメント

コーディング、コマンド改訂テーブルの 157

定義 150

WTO と WTOR 33

WTOR コマンド 125, 589

X

XCF (システム間カップリング・ファシリティ)、シスプレッ
クスの 83

XHILITE 373

XHILITE キーワード、IF-THEN または ALWAYS ステートメ
ントの 258

XITCI インストール・システム出口 32, 329

XITCI 処理 111

XLO 374

XLO キーワード、IF-THEN または ALWAYS ステートメント
の 258, 346

XML ファイル 388

X.25 スイッチド・バーチャル・サーキット

モニター 507

X.25 スイッチド・バーチャル・サーキットのモニター 507

Y

Yahoo のユーザー・グループ、NetView xxvii

YEL

COLOR アクション

IF-THEN または ALWAYS ステートメント 243

Z

z/OS

コマンド・プロセッサ 339

z/OS からのシステム・メッセージの転送の確認 336

サブシステム・インターフェースの使用 336

EMCS コンソールの使用 336

z/OS TCP/IP の場合のみの MIB ポーリングおよびしきい値分
け 510

z/OS オペレーティング・システム

システム自動化の準備 333

NetView のサブシステムとしての定義 336

[特殊文字]

;

同義語にはなし

変数 266

同義語の名前にはなし 266

% INCLUDE ステートメントで 264

%

同義語にはなし

名前 266

% INCLUDE ステートメントで

終わりには ; なし 264

% (デフォルト値コマンド接頭部文字) 594

%INCLUDE ステートメント

組み込み、メンバーおよびファイルの (%INCLUDE) 270

構文 264

定義 164

保守 420



Printed in Japan

SA88-4387-01



日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21