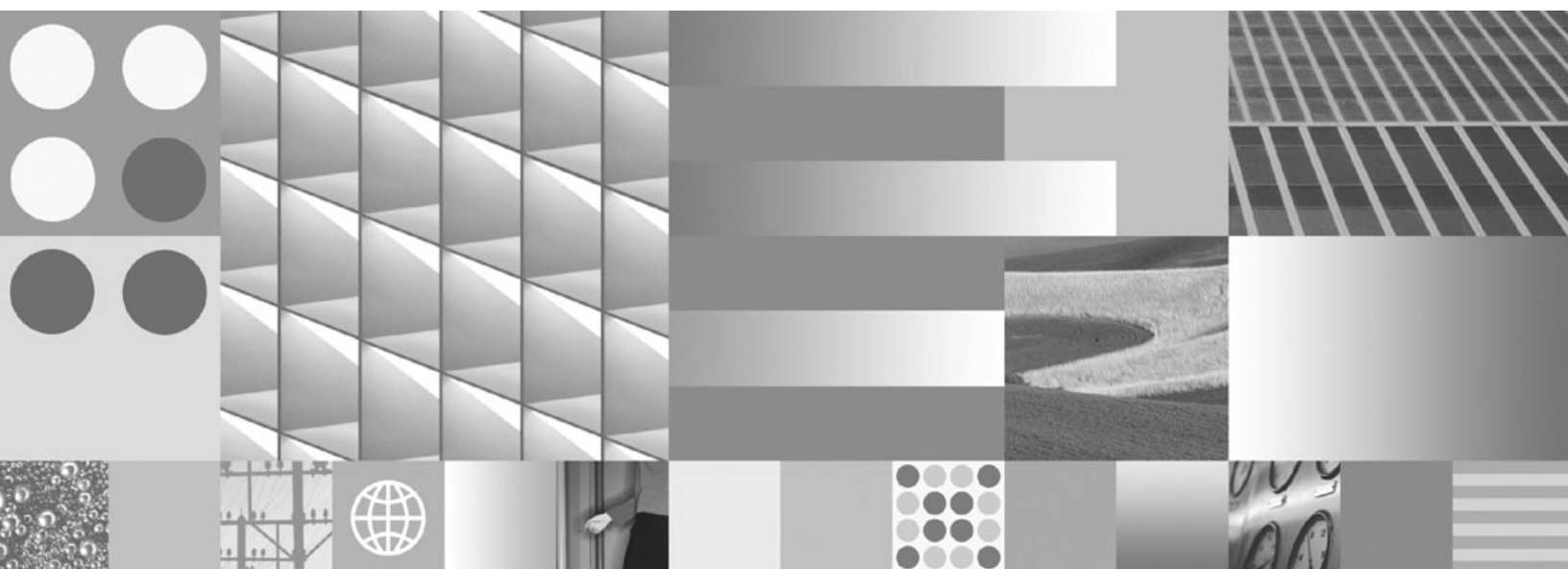
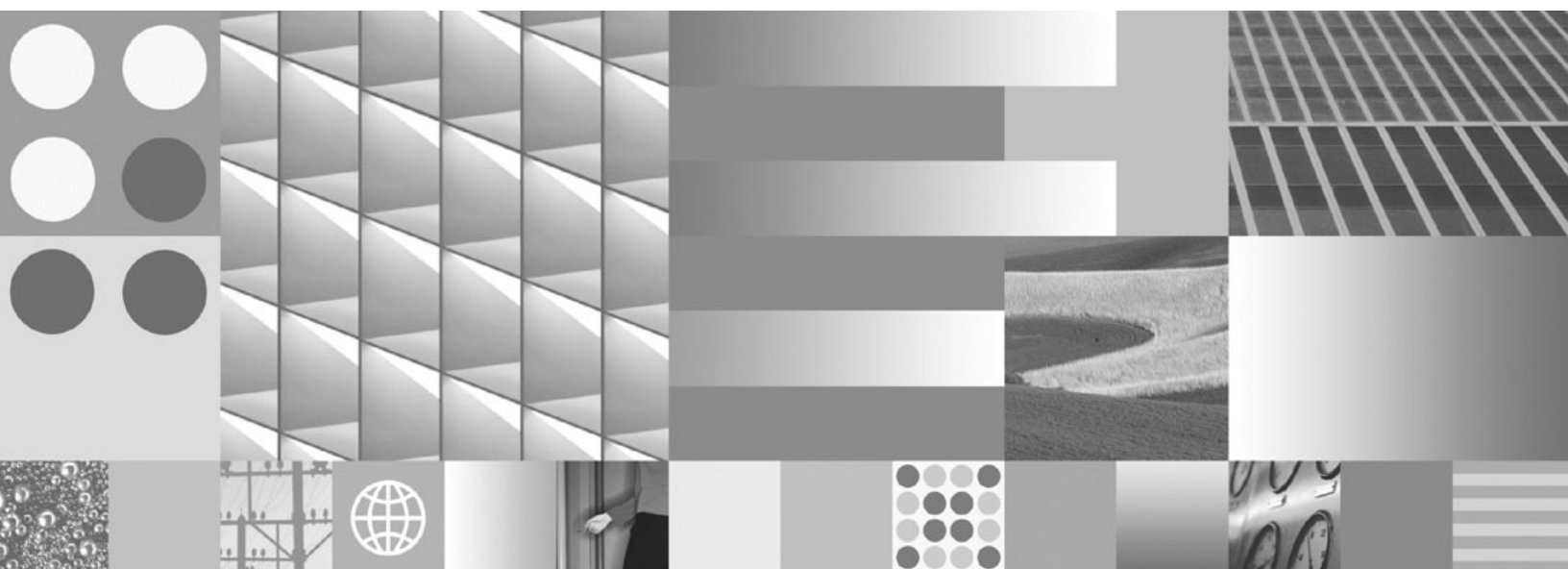




Manual de Programação de Aplicações



Manual de Programação de Aplicações

Nota

Antes de utilizar estas informações e o produto a que se referem, leia as informações incluídas em “Avisos” na página 533.

Esta edição aplica-se à versão 8, edição 4 do IBM DB2 Content Manager Enterprise Edition (número de produto 5724-B19) e à versão 8, edição 4 do IBM DB2 Content Manager for z/OS (número de produto 5697-H60) e a todas as edições e modificações subsequentes, salvo indicação em contrário em novas edições.

© Copyright International Business Machines Corporation 1996, 2007. Todos os direitos reservados.

Índice

ibm.com e recursos relacionados . . . ix

Como enviar comentários . . . ix

Contactar a IBM . . . ix

Acerca deste manual . . . 1

Quem deve utilizar este manual . . . 1

Como iniciar a programação de aplicações de gestão de conteúdos. . . 3

Informações importantes para os utilizadores do

Linux. . . 4

Funções obsoletas e removidas . . . 4

Acesso a dados através de servidores de conteúdos . 5

Conceitos de objectos de dados dinâmicos . . . 6

Objectos de dados dinâmicos (DDO) . . . 6

Objectos de dados expandidos (XDO) . . . 7

Representar conteúdo multimédia . . . 7

Servidores de conteúdos e DDOs . . . 8

Comparar DDO/XDOs com valores de atributos e

partes de artigos . . . 8

Identificadores persistentes (PID) . . . 8

Trabalhar com um servidor de conteúdos federado e pesquisa federada (obsoleto). . . 9

Correlação de esquema federado . . . 10

Utilizar componentes de correlação do servidor de

conteúdos federado. . . 11

Executar consultas federadas . . . 11

Sintaxe de consultas federadas . . . 12

Armazenar resultados da consulta em pastas

federadas (apenas em Java) . . . 15

Trabalhar com a administração do sistema . . . 15

Personalizar o cliente de administração do

sistema do DB2 Information Integrator for

Content. . . 15

Programar com as interfaces de programação de aplicações (APIs). . . 17

Compreender as diferenças entre as APIs de Java e

C++. . . 17

Compreender a arquitectura cliente/servidor

(apenas para Java) . . . 17

Criação de pacotes para o ambiente Java . . . 18

Sugestões de programação . . . 18

Configurar o ambiente Java (apenas para Java) . . 19

Definir as variáveis de ambiente Java para

Windows . . . 19

Definir as variáveis de ambiente Java para AIX . 19

Definir as variáveis de ambiente Java para Solaris

e Linux. . . 20

Definir variáveis de ambiente Java para z/OS

USS . . . 20

Utilizar Remote Method Invocation (RMI) com

servidores de conteúdos . . . 20

Configurar o ambiente C++ (Apenas para C++) . . 21

Definir as variáveis de ambiente C++ para

Windows . . . 22

Definir as variáveis de ambiente C++ para AIX . 23

Construir programas C++ . . . 23

Definir o subsistema da consola para conversão

de página de códigos em Windows . . . 23

Desempenho de aplicações de Java e C++ . . . 24

Colocação da API de Java em memória cache

global . . . 24

Opções de obtenção renovada de APIs nativas. . 25

Escalabilidade e desempenho da API de C++:

colocação em memória cache global e

agrupamento de ligações . . . 25

Expirar o tempo de ligações com a API de C++ . 28

Indicar elementos obtidos ao extrair a lista de

ACLs . . . 29

Compreender as várias opções de pesquisa. . . 30

Tratar excepções. . . 30

Constantes. . . 32

Ligação a servidores de conteúdos. . . 32

Estabelecer uma ligação . . . 33

Estabelecer e anular a ligação a um servidor de

conteúdos num cliente. . . 34

Definir e obter opções do servidor de conteúdos . 34

Enumerar servidores de conteúdos . . . 34

Enumerar as entidades e os atributos de um

servidor de conteúdos. . . 35

Trabalhar com objectos de dados dinâmicos (DDOs) 37

Criar um DKDDO . . . 38

Adicionar propriedades a um DDO . . . 38

Criar um identificador persistente (PID) . . . 39

Trabalhar com artigos e propriedades de dados . 39

Obter o DKDDO e as propriedades de atributo . 41

Apresentar a totalidade do DDO . . . 42

Eliminar um DDO (Apenas para C++) . . . 43

Trabalhar com objectos de dados expandidos

(XDOs). . . 44

Utilizar um identificador persistente (PID) de

XDO. . . 44

Sugestões de programação de Java . . . 44

Programar um XDO como parte de DDO . . . 45

Programar um XDO autónomo . . . 47

Exemplos de trabalho com um XDO . . . 49

Criar documentos e utilizar o atributo DKPARTS. . 51

Criar pastas e utilizar o atributo DKFOLDER . . . 53

Utilizar DKAny (apenas para C++) . . . 56

Utilizar código de tipo. . . 56

Gerir memória em DKAny . . . 56

Utilizar construtores . . . 57

Obter o código de tipo . . . 57

Atribuir um novo valor a DKAny . . . 57

Atribuir um valor de DKAny . . . 57

Apresentar DKAny . . . 58

Destruir o DKAny	58
Sugestões de programação	59
Utilizar recolhas e iteradores.	59
Utilizar métodos de recolha sequencial	59
Utilizar o iterador sequencial	59
Gerir memória em recolhas (apenas para C++)	61
Compreender a recolha e o iterador federados.	62
Consultar um servidor de conteúdos	63
Diferenças entre dkResultSetCursor e DKResults	64
Utilizar consultas paramétricas	64

Trabalhar com o DB2 Content Manager

Versão 8.4 65

Compreender o sistema do DB2 Content Manager	65
Compreender os conceitos do DB2 Content Manager	66
Itens.	66
Atributos	66
Tipos de artigos	67
Componentes raiz e descendentes	67
Objectos	68
Ligações e referências	69
Activar a criação automática de ligações com tipos de itens activados para versões	69
Documentos	71
Pastas	71
Criação de versões	71
Controlo de acessos.	73
Planear uma aplicação do DB2 Content Manager	80
Determinar as funções da aplicação	81
Tratamento de erros	81
Activar o suporte de pilha dupla IPv6	82
Controlador do IBM DB2 para suporte de JDBC e SQLJ no DB2 Content Manager Versão 8.4	83
Trabalhar com exemplos do DB2 Content Manager	85
O exemplo da companhia de seguros.	86
Criar uma aplicação do DB2 Content Manager	87
Compreender os componentes do software	87
Representar itens através de DDOs	87
Estabelecer ligação ao sistema do DB2 Content Manager	88
Trabalhar com artigos	89
Trabalhar com pastas	120
Definir ligações entre itens	127
Trabalhar com controlo de acessos	130
Criar um privilégio	130
Criar um conjunto de privilégios	131
Apresentar propriedades do conjunto de privilégios	132
Definir uma lista de controlo de acessos (ACL)	133
Obter e apresentar informações sobre a ACL	134
Atribuir uma ACL a um tipo de artigo	135
Atribuir uma ACL a um artigo	136
Utilizar ACLs de vistas em operações de eliminação, adição ou remoção de ligações	136
Implementar início de sessão único nas aplicações	138
Saídas de utilizador do servidor de bibliotecas do DB2 Content Manager	140
Saída de utilizador ICMFetchContentExit para alteração de texto	141
Saída de utilizador ICMLogonExit para início de sessão	145

Saída de utilizador myExit para encaminhamento de documentos	147
Saída de utilizador ICMValidatePassword para validar palavras-passe	149
Rotinas de saída de utilizador de ACL	152
Limitações do servidor de bibliotecas e da base de dados federada	155
Trabalhar com o gestor de recursos	156
Trabalhar com objectos do gestor de recursos	157
Obtenção confidencial de objectos de recurso	158
Remover conteúdo de objectos de recurso	158
Replicação assíncrona num gestor de recursos z/OS	160
Armazenar conteúdo numa recolha especificada pelo utilizador através da API de Java ou C++	163
API para o gestor de recursos na Versão 8.4	164
Gerir documentos em DB2 Content Manager	168
Criar o modelo de dados de gestão de documentos	169
Criar um tipo de artigo de documento	169
Criar um documento	171
Actualizar um documento	173
Obter e eliminar um documento	174
Criação de versões de partes do modelo de dados de gestão de documentos	175
Trabalhar com transacções	176
Pontos a ter em consideração ao conceber transacções na aplicação.	177
Precauções ao utilizar transacções	177
Utilizar operações de dar entrada e saída em transacções	177
Processar transacções.	178
Comportamento das transacções explícitas na Versão 8.4	179
Comportamento de transacção na eliminação de um utilizador que não pertence a um grupo	179

Pesquisar dados 181

Consultar o servidor de DB2 Content Manager	182
Aplicar a linguagem de consulta ao modelo de dados do DB2 Content Manager	183
Compreender a pesquisa paramétrica	185
Compreender a pesquisa de texto	186
Pesquisar conteúdo de objectos	186
Pesquisar documentos	187
Tornar os atributos definidos pelo utilizador pesquisáveis por texto	187
Compreender a sintaxe da pesquisa de texto	187
Criar uma pesquisa combinada de texto e parâmetros	194
Pesquisas exemplo com linguagem de consulta	196
Exemplos de consultas	200
Utilizar sequências de abandono nas suas consultas	207
Utilizar sequências de abandono com o operadores de comparação	207
Utilizar sequências de abandono com o operador LIKE	208
Utilizar sequências de abandono nas pesquisas de texto avançadas	209
Utilizar sequências de abandono com pesquisa de texto básica (funções contains-text-basic e score-basic)	210

Utilizar sequências de abandono em Java e C++	211
Compreender os filtros de vistas com base em linhas nas consultas	212
Considerações importantes para utilização de filtros baseados em linhas	212
Exemplo de situação de utilização	213
Descrição do comportamento de filtros	215
Considerações de desempenho	217
Índices de base de dados em cada atributo filtrado	218
Implicações de Segurança	219
Gramática da linguagem de consulta	219

Encaminhar um documento através de um processo 223

Compreender o processo do encaminhamento de documentos	223
Funções de encaminhamento de documentos	224
Compreender a compatibilidade com versões anteriores à Versão 8.3	226
Alterar a coluna DOCROUTINGFREQ na tabela ICMSTSYSCONTROL para DB2	227
Compreender as classes do encaminhamento de documentos	227
Criar objectos de serviço de encaminhamento de documentos	231
Definir um novo nó de trabalho normal	232
Enumerar nós de trabalho	233
Definir um novo ponto de recolha	234
Definir uma lista de trabalho	236
Enumerar listas de trabalho	238
Definir um novo processo e encaminhamento associado	238
Iniciar um processo de encaminhamento de documentos	240
Terminar um processo	241
Continuar um processo	242
Suspender um processo	242
Retomar um processo	243
Enumerar cadeias de identificador persistente de pacotes de trabalho numa lista de trabalhos	244
Obter informações relativas ao pacote de trabalho	244
Enumerar processos de encaminhamento de documentos	245
Implementar encaminhamento ad hoc	246
Consultas exemplo de encaminhamento de documentos	247
Conceder privilégios para encaminhamento de documentos	248
Trabalhar com listas de controlo de acessos para encaminhamento de documentos	249
Programar saídas de utilizador de encaminhamento de documentos	250
Constantes de encaminhamento de documentos	251

Compreender a busca prévia no DB2 Content Manager for z/OS 253

Busca prévia de objectos	253
Definições de tabela relativas à busca prévia	255

Trabalhar com outros servidores de conteúdos 259

Trabalhar com o OnDemand	260
Representar servidores e documentos do OnDemand	260
Estabelecer e anular ligações ao servidor do OnDemand	261
Enumerar informações no OnDemand	262
Obter um documento OnDemand	264
Activar o modo de pastas OnDemand	269
Pesquisa assíncrona	269
Pastas OnDemand como modelos de pesquisa	270
Pastas OnDemand como entidades nativas	270
Criar e modificar anotações	271
Rastreio	271
Trabalhar com o ImagePlus for OS/390	272
Enumerar entidades e atributos	273
Sintaxe da consulta do ImagePlus for OS/390	277
Trabalhar com o DB2 Content Manager for AS/400	279
Enumerar entidades (classes de índices) e atributos	279
Executar uma consulta	281
Executar uma consulta paramétrica	285

Construir aplicações de fluxo de trabalho do DB2 Information Integrator for Content (obsoleto) . . . 287

Ligação a serviços de fluxo de trabalho	287
Iniciar um fluxo de trabalho	288
Terminar um fluxo de trabalho	289
Enumerar todos os fluxos de trabalho	289
Suspender um fluxo de trabalho	290
Retomar um fluxo de trabalho	291
Enumerar todas as listas de trabalho	291
Aceder a uma lista de trabalhos	292
Aceder a artigos de trabalho	293
Mover artigos no fluxo de trabalho	294
Enumerar todos os modelos de fluxo de trabalho	294
Criar acções próprias (apenas para Java)	295
Trabalhar com os JavaBeans do fluxo de trabalho do DB2 Information Integrator for Content	296
Pré-requisitos	296
Configurar o modelo de dados exemplo	296
Utilizar JavaBeans de fluxo de trabalho na aplicação	304

Construir aplicações com JavaBeans visuais e não visuais 307

Compreender conceitos básicos de beans	307
Utilizar JavaBeans em construtores	308
Utilizar o IBM Websphere Studio Application Developer	309
Invocar os JavaBeans do DB2 Information Integrator for Content	309
Trabalhar com os beans não visuais	310
Configurações de beans não visuais	311
Compreender as funções dos beans não visuais	311
Categorias de beans não visuais	312
Considerações a ter ao utilizar beans não visuais	317
O bean CMBCConnectionPool	318

Nomes de apresentação	322
Rastrear e iniciar sessão nos beans	322
Compreender propriedades e eventos de beans não visuais	323
Construir uma aplicação através de beans não visuais	323
Trabalhar com beans visuais	324
Bean CMBLogonPanel	324
Bean CMBSearchTemplateList	326
Bean CMBSearchTemplateViewer	326
Validar ou editar campos de CMBSearchTemplateViewer	327
Bean CMBSearchPanel	327
Bean CMBSearchResultsViewer	328
Substituir menus emergentes	329
Bean CMBFolderViewer	330
Bean CMBDocumentViewer	331
Especificações do visualizador	332
Visualizadores predefinidos	333
Iniciar visualizadores externos	333
Bean CMBItemAttributesEditor	333
Proibir alterações no CMBItemAttributesEditor	334
Bean CMBVersionsViewer	334
Comportamentos gerais de beans visuais	334
Substituir um bean visual	336
Construir uma aplicação através de beans visuais	336

Trabalhar com serviços XML (apenas para Java). 339

Funcionamento dos serviços XML com outros níveis de programação do DB2 Content Manager	339
Importar e exportar metadados do DB2 Content Manager com serviços XML	342
Importar e exportar objectos de administração como XML	343
Importar e exportar objectos de modelos de dados do DB2 Content Manager como ficheiros de esquema XML (XSD)	345
Importar e exportar objectos de instâncias de dados do DB2 Content Manager como XML	365
Exportar artigos DDO do DB2 Content Manager como artigos XML	366
Importar artigos XML como artigos DDO do DB2 Content Manager	368
Importar e exportar dependências de objectos XML	369
Extrair conteúdo de diferentes origens XML	369
Correlacionar um esquema definido pelo utilizador com um esquema de armazenamento através de uma ferramenta de correlação de esquemas XML	370
Programar operações de tempo de execução através dos JavaBeans XML	373
Enumerar servidores do DB2 Content Manager com ListServerRequest	375
Autenticar pedidos de serviços da Web para segurança	376
Alterar a palavra-passe nos pedidos XML	377
Enumerar entidades do DB2 Content Manager com ListSchemaRequest	378
Criar artigos do DB2 Content Manager com CreateItemRequest	379

Pesquisar artigos do DB2 Content Manager com RunQueryRequest	381
Obter artigos do DB2 Content Manager com RetrieveItemRequest	383
Visualizar os privilégios do utilizador nos pedidos XML	388
Trabalhar com pastas do DB2 Content Manager através de pedido XML	389
Actualizar artigos do DB2 Content Manager com o UpdateItemRequest de XML	392
Eliminar artigos do DB2 Content Manager com DeleteItemRequest	399
Dar saída e entrada de artigos do DB2 Content Manager com CheckoutItemRequest e CheckinItemRequest	400
Ligar artigos do DB2 Content Manager com CreateLinkRequest ou DeleteLinkRequest	401
Mover artigos do DB2 Content Manager entre entidades com o MoveItemRequest	403
Aceder ao encaminhamento de documentos do DB2 Content Manager utilizando pedidos baseados em XML	404
Colocar em lotes vários pedidos de pedidos XML	421

Trabalhar com serviços da Web 425

Descrição geral dos serviços da Web	425
serviços da Web do DB2 Content Manager	426
Implementação dos serviços da Web do DB2 Content Manager	427
Arquitectura dos serviços da Web do DB2 Content Manager	428
Comparação das interfaces de serviços da Web do DB2 Content Manager	429
Funções dos serviços da Web	431
Lista de pedidos de serviços da Web	432
Instalar e configurar os serviços da Web do DB2 Content Manager	435
Plataformas suportadas pelos serviços da Web	435
Instalação e configuração de serviços da Web	436
Topologias do WebSphere Application Server suportadas	437
Verificar, iniciar e parar serviços da Web	438
Configuração pós-instalação	439
Registar e rastrear	441
Desinstalar e remover a configuração	442
Integrar serviços da Web básicos nas aplicações ou processos	442
Integrar o CMWebService	443
Integrar o CMBGenericWebService	451
Autenticar pedidos de serviços da Web para fins de segurança	455
Criar uma nova instância de um artigo através de serviços da Web	456
Configurar início de sessão único	461
Desempenho dos serviços da Web	463
Exemplos de serviços da Web	464
Deteção e correcção de problemas dos serviços da Web	464

Definir a instrução em memória cache no conjunto de ligações do WebSphere Application Server para serviços da Web	465
Utilizar a propriedade forceItemNamespace ao desenvolver aplicações cliente que utilizem serviços da Web	465
Actualizar a configuração do servidor de serviços da Web para consultar as bibliotecas de 32 bis ao utilizar uma instância de 64 bits do DB2	466
Trabalhar com o toolkit do visualizador de documentos de Java . 467	
Descrição geral do toolkit do visualizador de Java	467
Arquitectura do visualizador	469
Os motores de documentos.	470
O motor de anotações	471
Arquitecturas de visualizador exemplo	471
Criar um visualizador de documentos	475
Criar uma aplicação ou applet de visualizador autónomo	475
Trabalhar com documentos e anotações.	477
Personalizar o visualizador	477
Trabalhar com serviços de anotação	483
Utilizar interfaces de serviços de anotação.	484
Compreender o suporte de edição de anotações	485
Construir uma aplicação utilizando os serviços de anotação	486
Adicionar um tipo de anotação personalizado à aplicação do utilizador	487
Tipos e conversões de documentos suportados no toolkit do visualizador de Java	487
Trabalhar com as funções de manipulação de páginas	490
Definições de propriedades de motor para compressão	492
Especificar informações de codificação de caracteres para documentos de texto simples	493
Modificar preferências do visualizador	494
Limitações do visualizador de documentos Java	500
Trabalhar com a biblioteca de identificadores JSP e com a servlet de controlo 503	
Configurar a biblioteca de identificadores e a servlet.	503
Utilizar a biblioteca de identificadores	503
Convenções utilizadas na biblioteca do identificador.	504
Resumo dos identificadores	505
Identificadores relacionados com ligações	505
Identificadores relacionados com esquemas	505
Identificadores relacionados com pesquisa.	506
Identificadores relacionados com artigos	506
Identificadores relacionados com pastas	508
Identificadores relacionados com documentos	508
Servlet de controlo do DB2 Information Integrator for Content	509
O que a servlet pode fazer	509
Referência de servlet	510

Matriz de funções de toolkit da servlet	515
Deteção e correcção de problemas das aplicações de gestão de conteúdos. 519	
Registo de conectores Java, C++ e DB2 Information Integrator for Content	519
Registo de conectores Java	519
Registo de conectores C++	520
Modificar a configuração de registo para os conectores do DB2 Information Integrator for Content	520
Receber um erro ao compilar aplicações de C++ que estejam activadas para Unicode	522
Receber um erro ao utilizar atributos de referência	523
Receber um erro quando o conjunto de ligações do WebSphere Application Server está activado	523
Receber um erro de pesquisa no idioma tailandês quando a configuração da base de dados está incorrecta.	524
Receber um erro ao utilizar a pesquisa de texto do ICM para expressões em tailandês no DB2	524
Não é possível adicionar, armazenar, obter, ou actualizar um artigo	525
Não é possível importar um objecto DKDDO do XML	525
Receber um erro ao actualizar, reorganizar ou utilizar índices de texto para componentes pesquisáveis por texto	526
Impossível descarregar conteúdos do tipo de parte ICMBASESTREAM	526
A pesquisa de texto não devolve resultados para documentos criados em determinados idiomas	527
DKString DKLobICM::getContent () const não recomendada para o conector ICM	527
Não é possível carregar saída de utilizador	528
Não é possível invocar algumas APIs de administração numa transacção explícita	528
A API Connect() não verifica palavras-passe nulas ou vazias no modo de início de sessão fidedigno	529
Por vezes, a pesquisa de caracteres globais do servidor da biblioteca no z/OS pode devolver dados incorrectos	530
Recepção de uma mensagem DKException quando a consulta esgota o tempo de espera.	530
O visualizador de Java não suporta documentos Excel com a protecção de documentos activada	531
Visualizador de Java: os documentos de texto simples UTF-8 e UTF-16 aparecem sempre alinhados à esquerda	532
Visualizador de Java: A translineação provoca o posicionamento incorrecto de anotações gráficas em relação ao texto	532
Avisos 533	
Marcas Comerciais	535
Glossário 537	
Índice Remissivo 549	

ibm.com e recursos relacionados

Estão disponíveis suporte de produtos e documentação em ibm.com.

Suporte e assistência

O suporte de produtos está disponível na Web. Faça clique em Support no sítio da Web do produto, em:

DB2 Content Manager

<http://www.ibm.com/software/data/cm/cmgr/mp/>

DB2 Content Manager for z/OS

<http://www.ibm.com/software/data/cm/cmgr/390/>

Information Center

Pode visualizar a documentação do produto num Information Center baseado em Eclipse que pode ser instalado quando o produto for instalado. Por predefinição, o Information Center é executado num modo de servidor da Web a que outros navegadores da Web podem aceder. Pode igualmente executar o Information Center localmente na estação de trabalho. Consulte o o Information Center em <http://publib.boulder.ibm.com/infocenter/cmgmt/v8r4m0/index.jsp>.

Publicações em PDF

Pode visualizar os ficheiros PDF online através do Adobe Acrobat Reader do seu sistema operativo. Se não tiver o Acrobat Reader instalado, pode descarregá-lo no sítio da Web da Adobe, em <http://www.adobe.com>.

Consulte os seguintes sítios da Web de publicações em Web:

Produto	Sítio da Web
IBM DB2 Content Manager	http://www.ibm.com/support/docview.wss?rs=86&uid=swg27010186
IBM DB2 Content Manager for z/OS	http://www.ibm.com/support/docview.wss?rs=119&uid=swg27010286

Como enviar comentários

Os seus comentários são importantes para nos ajudar a facultar as informações mais precisas e de melhor qualidade.

Envie comentários através do formulário online de comentários de leitor, que se encontra em https://www14.software.ibm.com/webapp/iwm/web/signup.do?lang=en_US&source=swg-rcf.

Contactar a IBM

Para contactar a assistência a clientes da IBM nos Estados Unidos ou Canadá, contacte o número 1-800-IBM-SERV (1-800-426-7378).

Para conhecer as opções de assistência disponíveis, contacte um dos seguintes números:

- Nos Estados Unidos: 1-888-426-4343
- No Canadá: 1-800-465-9600

Para obter mais informações sobre como contactar a IBM, consulte o sítio da Web Contact IBM, em <http://www.ibm.com/contact/us/>.

Acerca deste manual

Este manual descreve como utilizar os Java, JavaBeans, e as interfaces de programação de aplicações (APIs) de C++ fornecidas com o DB2 Information Integrator for Content e o DB2 Content Manager Versão 8.4. As APIs e beans facultam blocos de construção para criar aplicações que têm acesso ao conteúdo armazenado em servidores de conteúdos heterogêneos.

Este manual inclui:

- Uma introdução aos conceitos de programação de aplicações de DB2 Information Integrator for Content e DB2 Content Manager, incluindo conceitos de objectos de dados dinâmicos no contexto de Java e C++
- Uma descrição da função acessível através do conector do DB2 Content Manager Versão 8.4
- Documentação sobre todos os outros conectores de DB2 Information Integrator for Content dos servidores de conteúdos
- Actualizações de JavaBeansvisual e não-visual
- Actualizações para programar informações em fluxo de trabalho

As ilustrações referentes ao DB2 Content Manager implicam quer a pré-Versão 8.1, quer a Versão 8.4 do produto.

Quem deve utilizar este manual

Este manual destina-se a programadores de aplicações com algumas ou todas as seguintes capacidades:

- Experiência com C++, Java, JavaBeans ou HTML
- Familiaridade com conceitos de base de dados relacionais
- Conhecimento do protocolo de DDO/XDO

Como iniciar a programação de aplicações de gestão de conteúdos

O DB2 Content Manager faculta interfaces de programação de aplicações (APIs) Java™, JavaBeans™ e C++ com o DB2 Information Integrator for Content Versão 8 Edição 4. As APIs e beans facultam blocos de construção para criar aplicações que tenham acesso ao conteúdo armazenado em servidores de conteúdos heterogêneos.

A tabela seguinte resume e disponibiliza ligações para as tarefas de alto nível destinadas à programação de aplicações de gestão de conteúdos.

Tarefas de programação	O que pode fazer	Quando executar
"Trabalhar com um servidor de conteúdos federado e pesquisa federada (obsoleto)" na página 9	Utilizar um objecto de DKDatastoreFed para uma pesquisa federada. Executar uma pesquisa federada utilizando APIs, começando por criar uma cadeia de consulta federada.	Ao executar uma pesquisa federada.
"Programar com as interfaces de programação de aplicações (APIs)" na página 17	Utilizar APIs de Java e C++ para conceber as aplicações	Executar ao configurar o ambiente Java e C++, trabalhar com um DDO, com um XDO e com objectos de DKAny.
"Trabalhar com o DB2 Content Manager Versão 8.4" na página 65	Utilizar as APIs de conector ICM para construir e implementar aplicações personalizadas que acedem a um servidor de conteúdos do DB2 Content Manager. Utilizar as APIs para integrar as aplicações existentes num servidor de conteúdos do DB2 Content Manager.	Executar ao planear e criar uma aplicação e ao trabalhar com o servidor de bibliotecas e com o gestor de recursos.
"Pesquisar dados" na página 181	Pesquisar dados utilizando a pesquisa paramétrica e de texto	Executar ao pesquisar dados, utilizar a pesquisa paramétrica e de texto.
"Encaminhar um documento através de um processo" na página 223	Encaminhar documentos através de um processo empresarial, utilizando APIs de encaminhamento de documentos	Executar ao encaminhar um documento através de nós de trabalho, listas de trabalho e processos.
"Compreender a busca prévia no DB2 Content Manager for z/OS" na página 253	Transportar objectos que estejam actualmente localizados em meios mais lentos, como suportes ópticos ou de banda, para meios mais rápidos.	Executar quando necessário.
"Trabalhar com outros servidores de conteúdos" na página 259	Utilizar as classes dkDatastore para definir um servidor de conteúdos adequado aos servidores de conteúdos da sua aplicação.	Executar ao trabalhar com servidores de conteúdos OnDemand, ImagePlus para OS/390, e DB2 Content Manager para AS/400.
"Construir aplicações de fluxo de trabalho do DB2 Information Integrator for Content (obsoleto)" na página 287	Criar ou expandir as suas próprias aplicações para utilizarem o suporte de fluxo de trabalho de DB2 Information Integrator for Content utilizando classes e APIs de DB2 Information Integrator for Content. a.	Executar ao utilizar classes e APIs de DB2 Information Integrator for Content.

Tarefas de programação	O que pode fazer	Quando executar
“Construir aplicações com JavaBeans visuais e não visuais” na página 307	Utilizar os JavaBeans™ visuais e não visuais no DB2 Information Integrator for Content para construir aplicações com base em Java ou de cliente da Web.	Executar ao utilizar beans visuais ou não visuais para criar aplicações.
“Trabalhar com serviços XML (apenas para Java)” na página 339	Importar, armazenar, actualizar, obter e exportar uma grande variedade de objectos ao representar os objectos como XML	Executar ao trabalhar com serviços XML.
“Trabalhar com serviços da Web” na página 425	Integrar as aplicações ou o processo cliente utilizando serviços da Web	Executar ao utilizar serviços da Web
“Trabalhar com o toolkit do visualizador de documentos de Java” na página 467	Utilizar o toolkit do visualizador de documentos para apresentar, manipular e anotar documentos.	Executar ao utilizar o visualizador de Java.
“Trabalhar com a biblioteca de identificadores JSP e com a servlet de controlo” na página 503	Utilizar a biblioteca de identificadores de Java™ Server Pages (JSP) e um servlet para escrever JSPs ou servlets para aplicações da Web.	Executar quando necessário
“Detecção e correcção de problemas das aplicações de gestão de conteúdos” na página 519	Corrigir problemas comuns que ocorrem durante a programação.	Executar quando necessário.

Informações importantes para os utilizadores do Linux

Estas informações são facultadas para efeitos de referência e poderão conter documentação sobre os seguintes componentes que NÃO são suportados em Linux:

- Fluxo de trabalho avançado do DB2 Information Integrator for Content
- Todos os conectores de C++
- Conectores remotos de Java
- Conector do ImagePlus for OS/390
- Leitor VideoCharger

Funções obsoletas e removidas

A tabela de conectores obsoletos e removidos indica os conectores, APIs e serviços que estavam disponíveis no DB2 Content Manager Versão 8.3 e os respectivos estados actuais.

Foram utilizadas as seguintes terminologias:

activo O conector activo indica que continua a ser mantido e, quando apropriado, melhorado, juntamente com os níveis actuais do repositório suporado, ferramentas de desenvolvimento e ambientes operativos.

obsoleto

O conector obsoleto ainda se encontra disponível e é mantido como parte da edição actual para ambientes suportados, mas de uma forma geral não é recomendado para utilização no novo desenvolvimento; é geralmente indicada uma alternativa. Embora as APIs obsoletas ainda sejam mantidas actualmente, estas podem ser removidas em edições posteriores.

removido

Os conectores que suportam sistemas que atingiram o final do suporte são removidos na edição actual.

O DB2 Content Manager Versão 8.4 não suporta o conector de DB2, o conector de ODBC, o conector de JDBC, o conector de DD (Domino Doc), o conector de DL (Content Manager versão 7), o conector de TS (Text Search), o conector de QBIC (Image Search) e a criação de marcas d'água a partir da solução DB2 Information Integrator for Content. Para além disso, o DB2 Content Manager Versão 8.4 não recorre ao conector federado (federação), ao fluxo de trabalho federado avançado e ao conector de OD (DB2 Content Manager OnDemand).

A tabela seguinte identifica o estado e as alterações aos conectores do DB2 Information Integrator for Content no DB2 Content Manager Versão 8.4.

Tabela 1. Conectores obsoletos e removidos

Conector	Descrição	Status
DB2	DB2 UDB	Removido
DD	Domino Doc	Removido
DL	IBM Content Manager V7	Removido
ICM	DB2 Content Manager V8	Activo
IP	Image Plus (IP/390)	Activo
JDBC	Conector relacional	Removido
FED	Federação	Obsoleto, utilize o WebSphere II Content Edition
OD	CM OnDemand	Obsoleto, utilize o OnDemand Web Enablement Kit
ODBC	Conector relacional	Removido
QBIC	Pesquisa de imagens	Removido
TS	Pesquisa de texto	Removido
V4	CM iSeries V5 (VI/400)	Activo

O Advanced Workflow (MQ Workflow) encontra-se igualmente obsoleto no DB2 Content Manager Versão 8.4.

Todos os conectores e funções obsoletos estão documentados em *Application Programming Reference* e não reportam necessariamente avisos de estado obsoleto do compilador. Os conectores e funções obsoletos ainda são suportados actualmente, podendo ser removidos numa edição posterior.

Acesso a dados através de servidores de conteúdos

Um servidor de conteúdos é um depósito de dados que é compatível com o protocolo DDO/XDO. Um servidor de conteúdos suporta sessões, ligações, transacções, cursores e consultas de utilizadores. As aplicações que utilizam as interfaces de programação de aplicações (APIs) e as bibliotecas de classes descritas neste manual, podem executar funções suportadas pelos servidores de conteúdos, tais como adicionar, obter, actualizar e eliminar DDOs. O DB2 Information Integrator for Content suporta os seguintes servidores de conteúdos:

- DB2 Content Manager Versão 8.4
- ImagePlus para OS/390

- OnDemand
- VisualInfo para AS/400

As aplicações que utilizam o DB2 Information Integrator for Content podem criar um servidor de conteúdos associado que funciona como um servidor comum. As classes associadas ao DB2 Information Integrator for Content permitem a pesquisa, obtenção e actualização associadas em vários servidores de conteúdos.

O servidor de conteúdos associado do DB2 Information Integrator for Content e cada um dos servidores de conteúdo possuem esquemas diferentes. A integração de servidores de conteúdos múltiplos e heterogêneos num sistema federado exige conversão e correlação.

As funções de correlação de esquemas fornecem as informações do esquema para cada servidor de conteúdos. As informações fornecidas pela correlação de esquemas são utilizadas durante a pesquisa federada e a recolha federada, bem como a administração do sistema de DB2 Information Integrator for Content. O DB2 Information Integrator for Content mantém o esquema e as correlações, bem como as outras informações da administração na base de dados da administração.

Conceitos de objectos de dados dinâmicos

Em coordenação com o CORBA Persistent Object Service e Object Query Service Specification de Object Management Groups (OMG), o DB2 Information Integrator for Content fornece uma implementação do objecto de dados dinâmicos (DDO) e a sua extensão, o objecto de dados expandidos (XDO), que fazem parte dos protocolos CORBA Persistent Data Service (PDS). Os conceitos de DDO e XDO não são específicos de um único servidor e podem ser utilizados para representar objectos de dados em qualquer sistema de gestão de bases de dados suportados pelo DB2 Information Integrator for Content.

O objecto de dados dinâmico é uma interface utilizada para retirar e introduzir dados num servidor de conteúdos. Os DDOs existem na aplicação como objectos de tempo de execução e por isso não existem após a finalização de uma aplicação.

Objectos de dados dinâmicos (DDO)

Um *DDO* é uma representação neutra a servidores de conteúdos dos dados persistentes de um objecto. O seu objectivo é conter todos os dados para um único objecto persistente. É também uma interface para obter ou carregar dados persistentes num servidor de conteúdos.

Um DDO tem um único ID persistente (PID), um tipo de objecto e um conjunto de artigos de dados cuja cardinalidade se chama contagem de dados. Cada artigo de dados tem um nome, valor, ID, uma ou mais propriedades de dados e contagem da propriedade de dados. Cada propriedade de dados pode ter um ID, um nome e um valor.

Por exemplo, um DDO pode representar uma fila de tabelas de base de dados, cujas colunas são representadas pelos artigos de dados do DDO e pelas suas propriedades. Um DDO pode conter um ou mais objectos de dados expandidos (XDOs) que representam tipos de dados não tradicionais. A Figura 1 na página 7 apresenta objectos de dados dinâmicos e artigos de dados.

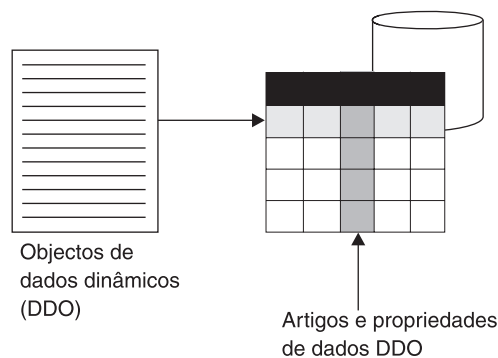


Figura 1. Objectos de dados dinâmicos e artigos de dados

Objectos de dados expandidos (XDO)

Um XDO é uma representação de dados complexos de multimédia como, por exemplo, um artigo de recurso a armazenar uma imagem ou documento no DB2 Content Manager ou um novo tipo de dados introduzido pelos serviços de uma base de dados relacional como, por exemplo, IBM DB2 Extenders.

Os XDOs complementam os DDOs armazenando dados multimédia de tipos complexos e oferecendo funções que implementam os comportamentos dos tipos de dados na aplicação. Os XDOs podem estar contidos ou pertencer a um DDO de forma a representar um objecto de dados multimédia complexo.

Os XDOs têm um conjunto de propriedades para representar essas informações como tipos de dados e IDs. Os XDOs também podem ser objectos de dados dinâmicos autónomos. A Figura 2 demonstra um exemplo de XDOs.

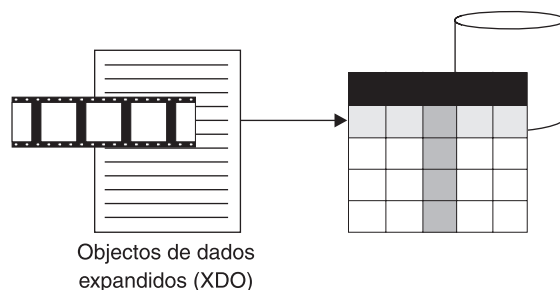


Figura 2. Objectos de dados expandidos (XDOs)

Representar conteúdo multimédia

Os DDOs e os XDOs podem representar objectos de dados de qualquer tipo e estrutura.

Por exemplo, um filme pode ser representado por um DDO. Este DDO contém vários artigos de dados, que representam atributos do filme, como Nome_Realizador ou Título_Filme e XDOs multimédia, que representam os dados multimédia do filme, tais como vídeo clips ou galerias fotográficas.

No DB2 Content Manager Versão 8, um DDO consiste em todos os metadados que descrevem o objecto como, por exemplo, um documento ou imagem. Um XDO expande a funcionalidade de DDO para suportar o conteúdo de recurso. O

conteúdo de recurso é qualquer tipo de conteúdo, desde texto ou dados binários a sequências de vídeo e áudio. Um artigo que implementa (e deve ser) um XDO é também um DDO e suporta toda a funcionalidade facultada por um DDO, bem como a funcionalidade adicional facultada por (e deve ser) um XDO.

Servidores de conteúdos e DDOs

Os DDOs são criados dinamicamente associados a um servidor de conteúdos.

A associação entre um DDO e um servidor de conteúdos é estabelecida com o PID dos DDOs.

Em geral, uma aplicação do DB2 Information Integrator for Content executa os cinco passos enumerados de seguida, de modo a introduzir e retirar dados de um servidor de conteúdos.

1. Criar um servidor de conteúdos.
2. Estabelecer uma ligação ao servidor de conteúdos.
3. Criar os DDOs que serão utilizados e associar o servidor de conteúdos aos DDOs.
4. Adicionar, obter, actualizar e eliminar os DDOs através de métodos apropriados.
5. Fechar a ligação e destruir o servidor de conteúdos.

Comparar DDO/XDOs com valores de atributos e partes de artigos

O tipo de objecto do DDO corresponde ao tipo de artigo associado do artigo. Os artigos de dados de um DDO correspondem aos atributos de um artigo.

Um DDO corresponde a um artigo no DB2 Information Integrator for Content. Por exemplo, no DB2 Content Manager um tipo de artigo é criado utilizando um conjunto de atributos e um artigo é sempre indexado por tipo de artigo.

Um DDO pode manter um ou mais XDOs que correspondam a partes de artigo no DB2 Information Integrator for Content.

Identificadores persistentes (PID)

O identificador persistente (PID) identifica de forma única um objecto persistente em qualquer servidor de conteúdos. O PID de um DDO é formado por um ID de artigo, um nome de servidor de conteúdos e outras informações relacionadas. Quando um DDO é adicionado a um servidor de conteúdos, o sistema atribui um PID único ao DDO.

Devido ao facto de um DDO ser uma interface dinâmica de dados persistentes que são introduzidos e retirados de servidores de conteúdos, diferentes DDOs podem representar as mesmas entidades de dados persistentes e, consequentemente, os DDOs podem ter o mesmo PID. Por exemplo, um DDO pode ser criado para introduzir uma entidade de dados num servidor de conteúdos para armazenar persistentemente dados e outro DDO pode ser criado para manter a mesma entidade de dados, em relação à qual é dada saída do mesmo servidor de conteúdos para ser modificada. Neste caso, estes dois DDOs partilham o mesmo valor de PID.

Trabalhar com um servidor de conteúdos federado e pesquisa federada (obsoleto)

A *Pesquisa federada* é o processo de pesquisar dados num ou mais servidores de conteúdos.

O DB2 Information Integrator for Content está obsoleto no DB2 Content Manager Versão 8.4.

Utilize um objecto de DKDatastoreFed para uma pesquisa federada. A pesquisa federada funciona com classes que são implementações específicas de dkDatastore, dkDatastoreDef e outras classes relacionadas que suportam pesquisas federadas. As classes federadas específicas funcionam junto com outras classes comuns, tais como consultas, recolhas e objectos de dados, fazendo parte da estrutura do DB2 Information Integrator for Content.

As classes federadas funcionam em diferentes servidores de conteúdos, tais como o ImagePlus for OS/390. As classes fornecem um conjunto de funções genéricas para pesquisa federada e acesso através de servidores de conteúdos. Esta vista comum, denominada *modelo de documento federado*, é ilustrada na Figura 3.

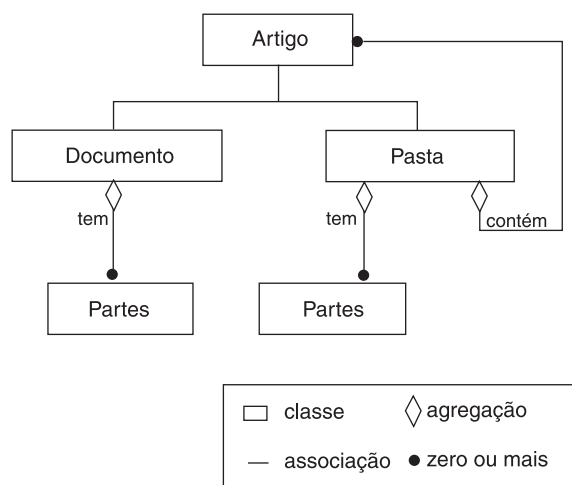


Figura 3. Vista de documento federado

Um artigo pode ser um documento ou uma pasta. Um documento pode conter zero ou mais partes. Uma pasta pode ter zero ou mais artigos que podem ser documentos ou outras pastas.

Nem todos os servidores de conteúdos podem suportar o modelo de documento federado. Por exemplo, uma base de dados DB2 não tem pastas ou partes. Um artigo correlaciona uma linha numa tabela DB2 ou noutra tabela de base de dados relacional e é utilizado caso um servidor de conteúdos não suporte documentos ou pastas.

Em geral, um documento é representado no seu programa através de um objectos de dados dinâmicos (DDO), que é um objecto de dados que se descreve a ele mesmo, para transferir dados para o interior ou para o exterior do servidor. O

próprio DDO tem uma estrutura geral e suporta uma variedade de modelos. Não está limitado ao modelo de documento federado. Esta flexibilidade permite a um DDO representar dados em servidores de conteúdos diferentes, cada um com o seu modelo de dados.

Uma *entidade* é um objecto de servidor de conteúdos que abrange atributos. Um *atributo* é um identificador utilizado para metadados em servidores de conteúdos. Por exemplo, os perfis, campos ou palavras-chave são atributos em servidores de conteúdos.

Cada servidor de conteúdos possui a sua terminologia para explicar o modelo que suporta. A Tabela 2 relaciona a terminologia utilizada para vários servidores de conteúdos no modelo federado:

Tabela 2. Correlacionar terminologia para cada servidor de conteúdos

Servidor de conteúdos	Origem de dados	Entidade	Atributo	Ver
DB2 Content Manager Versão 8.4	servidor de bibliotecas	tipo de artigo	atributo	vista de tipo de artigo ou subconjunto de tipo de artigo
OnDemand	Servidor OnDemand	grupo de aplicações pasta	campo critério	N/A
ImagePlus	Servidor ImagePlus para OS/390	entidade	atributo	N/A
DB2 Content Manager for AS/400	Servidor DB2 Content Manager for AS/400	classe de índices	atributo	vista de classe de índices
Servidor de conteúdos federado	servidor de correlações	entidade federada correlacionada	atributo federado correlacionado	modelo de pesquisa
Servidor de conteúdos federado que suporta pastas federadas	servidor	entidade federada	atributo federado	pasta federada

Correlação de esquema federado

Uma *correlação de esquemas* representa uma correlação entre o esquema no servidor de conteúdos e a estrutura dos artigos que o utilizador pretende processar na aplicação.

Um *esquema federado* constitui o esquema conceptual de um servidor de conteúdos federado de DB2 Information Integrator for Content e informações da correlação entre os conceitos no servidor de conteúdos federado e os conceitos em cada servidor de conteúdos participante. A correlação de esquemas processa a diferença entre o modo de armazenamento físico dos dados e o modo que o utilizador pretende processar os dados numa aplicação.

As informações de correlação estão representadas em memória em classes de correlação de esquemas.

Utilizar componentes de correlação do servidor de conteúdos federado

Os componentes de correlação de esquemas do servidor de conteúdos federado incluem o ID de utilizador e informações de registo do servidor de conteúdos.

Para além das informações relativas à correlação de esquemas destinadas à correlação de entidades e atributos, um servidor de conteúdos federado deve igualmente possuir acesso às seguintes informações:

Correlação do ID de utilizador e palavra-passe

Para suportar uma única função de início de sessão, cada ID de utilizador no DB2 Information Integrator for Content pode ter a correlação definida para o ID de utilizador correspondente em cada servidor de conteúdos.

Registo do servidor de conteúdos

Cada servidor de conteúdos terá de ser registado de forma a poder ser localizado e ter a sessão iniciada pelo DB2 Information Integrator for Content.

O ID de utilizador e as informações do servidor de conteúdos são mantidas na base de dados da administração do DB2 Information Integrator for Content.

Executar consultas federadas

A execução de consultas federadas inclui a criação de uma cadeia de consulta federada e, em seguida, a execução da consulta.

Para executar uma pesquisa federada utilizando APIs, comece por criar uma cadeia de consulta federada. Pode criar e executar uma consulta, transmitindo a cadeia de consulta ao método `execute` ou ao método `evaluate` do servidor de conteúdos federado de modo a processar directamente a consulta.

A cadeia de consulta é interpretada num formulário de consulta federada que constitui, essencialmente, uma representação neutra do servidor de conteúdos da consulta.

Se a consulta tiver origem numa aplicação baseada numa interface gráfica de utilizador (GUI), a consulta não tem de ser analisada e o formulário da consulta federada correspondente pode ser construído directamente.

À medida que é processada uma pesquisa federada, o DB2 Information Integrator for Content executa os passos seguintes:

- Traduz o formulário canónico da consulta para várias consultas nativas que se executam em cada servidor de conteúdos. As informações de tradução são obtidas a partir da correlação de esquemas.
- Converte entidades e atributos federados em entidades e atributos nativos para cada um dos servidores de conteúdos. Este processo utiliza os mecanismos de correlação e conversão descritos na correlação de esquemas.
- Filtra apenas os dados relevantes durante a construção de consultas nativas.
- Formula consultas nativas e submete-as aos servidores de conteúdos individuais.

- Converte entidades e atributos nativos em entidades e atributos federados segundo as informações de correlação.
- Filtra os resultados para incluir apenas os dados solicitados.
- Intercala os resultados de vários servidores de conteúdos numa recolha federada.

A recolha federada fornece a possibilidade de separar os resultados da consulta segundo o servidor de conteúdos. Crie um iterador sequencial invocando o método `createMemberIterator` na recolha federada. Utilizando este iterador sequencial, pode aceder a cada recolha de membro, que consiste num objecto `DKResults`, e processá-la separadamente.

```
graph TD
    UI[Interface gráfica do utilizador] --> FC[Formulário canônico da consulta]
    UI --> IC[Interpretador da consulta federada]
    FC --> IC
    FC --> PPF[Processador de consultas federadas]
    FC --> PRFQ[Processador de resultados FQ]
    IC --> PPF
    PPF --> PRFQ
    PRFQ --> RF[Recolha federada]
    RF --> R[Resultados da consulta]
    PRFQ --> BDEIP[Base de dados do Enterprise Information Portal]
    BDEIP --> PPF
    BDEIP --> PRFQ
    PRFQ --> CE[Correlação de esquemas]
    CE --> FC
    PRFQ --> CN1([Consulta nativa-1])
    PRFQ --> CN2([Consulta nativa-2])
    PRFQ --> CNn([Consulta nativa-n])
    CN1 --> ADN1[Arquivo de dados nativo-1]
    CN2 --> ADN2[Arquivo de dados nativo-2]
    CNn --> ADNn[Arquivo de dados nativo-n]
    ADN1 --> PRFQ
    ADN2 --> PRFQ
    ADNn --> PRFQ
```

Sintaxe de consultas federadas

12 Manual de Programação de Aplicações

Quando criar uma consulta federada, terá de estar na sintaxe apropriada, como é demonstrado mais abaixo. O servidor de conteúdos federado não suporta consulta de imagens.

```
PARAMETRIC_SEARCH=([ENTITY=entity_name,]
                    [MAX_RESULTS=maximum_results,]
                    [COND=(conditional_expression)]
                    [; ...]
                    );
[OPTION=( [CONTENT=yes_no_attronly]
          )]

[and

TEXT_SEARCH=(COND=(text_search_expression)
              );
[OPTION=( [ASSOCIATED_ENTITY={associated_entity_name}];]
          [MAX_RESULTS=maximum_results;]
          [TIME_LIMIT=time_limit]
          )]
]
```

O operador NOT não é suportado em pesquisas federadas.

Consulta paramétrica federada com o operador LIKE

```
"PARAMETRIC_SEARCH = ( ENTITY = F_DGSAMP71, MAX_RESULTS = 5,
COND = ( fName LIKE '%' ) )"
```

Consulta paramétrica federada com os operadores LIKE e >

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 5,
COND = ( fJTitle LIKE 'Java%' AND fJNumPages > 20 ) )"
```

Consulta paramétrica federada com os operadores LIKE e <

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal,
MAX_RESULTS = 5, COND = ( fJTitle LIKE 'Java%' AND fJNumPages < 20 ) )"
```

Consulta paramétrica federada com o operador BETWEEN

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,
COND = ( fJNumPages BETWEEN 5 200 ) )"
```

MAX_RESULTS devolve todos os resultados quando definido como zero.

Consulta paramétrica federada com o operador NOTBETWEEN

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,
COND = ( fJNumPages NOTBETWEEN 5 100 ) )"
```

Consulta paramétrica federada com o operador IN

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,
COND = ( fJArticleTitle IN ('Java', 'Multi-Disk B-trees.',
'On Beyond Data.', 'IBM') ) )"
```

Consulta paramétrica federada com o operador NOTIN

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,
COND = ( fJArticleTitle NOTIN ('Java', 'Multi-Disk B-trees.',
'On Beyond Data.', 'IBM') ) )"
```

Consulta paramétrica federada com o operador ==

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,
COND = ( fJEditorName == 'Harth' ) )"
```

Consulta paramétrica federada com o operador <>

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,  
COND = (fJSectionTitle <> 'not available') )"
```

Consulta paramétrica federada com os operadores AND e OR

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,  
COND = ((fJTitle LIKE '%Java%') OR ((fJEditorName <> NULL) AND  
(fJArticleTitle LIKE 'Computer%')) ) ); OPTION = (CONTENT = YES)"
```

Consulta paramétrica federada com o operador CONTAINS_TEXT_IN_CONTENT

Este exemplo pesquisa texto no conteúdo. O conteúdo pode ser uma palavra ou uma expressão. Esta só é válida quando a entidade federada pesquisada por texto (FedTextResource) é correlacionada a um tipo de artigo pesquisável por texto do DB2 Content Manager Versão 8 ou uma entidade pesquisável por texto do Extended Search.

```
"PARAMETRIC_SEARCH = ( ENTITY = FedTextResource, MAX_RESULTS = 6,  
COND = ( CONTAINS_TEXT_IN_CONTENT 'XML' ) ); OPTION =  
( CONTENT = YES )"
```

Consulta paramétrica federada com o operador CONTAINS_TEXT

Pesquisa texto em valores de atributo. O texto pode ser uma palavra ou expressão. Esta só é válida quando o atributo federado pesquisável por texto (fJTitle) é correlacionado a um atributo pesquisável por texto do DB2 Content Manager Versão 8 ou um atributo pesquisável por texto do Extended Search.

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal, MAX_RESULTS = 0,  
COND = (fJTitle CONTAINS_TEXT 'Java') )"
```

Consulta de texto federada

Pesquisa texto no conteúdo. O texto pode ser uma palavra ou expressão. A palavra-chave ASSOCIATED_ENTITY só se aplica quando uma entidade federada é pesquisável por texto. Esta só é válida quando a entidade federada pesquisável por texto (FedEntity) é correlacionada a um tipo de artigo pesquisável por texto do DB2 Content Manager Versão 8 ou uma entidade pesquisável por texto do Extended Search.

```
"TEXT_SEARCH = ( COND = ('XML') ); OPTION = ( ASSOCIATED_ENTITY=FedEntity )"
```

Consulta paramétrica federada com o operador OR

```
"PARAMETRIC_SEARCH = ( ENTITY = FedTextResource,  
MAX_RESULTS = 0, COND = (FedTextResourceJTitle LIKE '%test%'  
) ) OR TEXT_SEARCH = ( COND = ('UNIX') );  
OPTION = ( ASSOCIATED_ENTITY =  
FedTextResource; MAX_RESULTS = 4 )"
```

Consulta paramétrica federada com o operador AND

```
"PARAMETRIC_SEARCH = ( ENTITY = FedTextResource, COND =  
(FedTextResourceJTitle LIKE '%test%') ) AND TEXT_SEARCH =  
( COND = ('UNIX') ); OPTION = ( ASSOCIATED_ENTITY =  
FedTextResource)"
```

Consulta paramétrica e consulta de texto federadas em atributos com o operador OR

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal,  
COND = (fJTitle LIKE 'Java%' OR fJArticleTitle  
CONTAINS_TEXT 'Database') );OPTION = ( CONTENT = ATTRONLY )"
```

Consulta paramétrica e consulta de texto federadas em atributos com o operador OR

```
"PARAMETRIC_SEARCH = ( ENTITY = F_ICMNLSDb_Journal,  
COND = (fJTitle LIKE 'Java%' OR fJArticleTitle  
CONTAINS_TEXT 'Database') );OPTION = ( CONTENT = YES )"
```

Armazenar resultados da consulta em pastas federadas (apenas em Java)

O DB2 Information Integrator for Content faculta actualmente entidades federadas especiais que suportam pastas federadas.

Estas pastas federadas podem armazenar os resultados combinados de uma consulta federada como, por exemplo, um documento do Content Manager e um documento relacionado do OnDemand. O utilizador pode enviar os resultados directamente para um fluxo de trabalho.

O DB2 Information Integrator for Content armazena as pastas como DDOs, que poderá adicionar a uma recolha DKFolder. Pode também armazenar as pastas como XDOs numa recolha DKParts.

A entidade federada especial utiliza tabelas adicionais para conter as pastas. Qualquer outra funcionalidade (como, por exemplo, consultas, APIs e atributos) tem comportamentos idênticos a uma entidade federada normal. A entidade especial só armazena PIDs de DDO nas pastas.

Se optar por não correlacionar uma entidade federada especial, a consulta federada pesquisa apenas as entidades federadas especiais. Se optar por correlacionar uma entidade federada especial a outras entidades nativas, a consulta federada pesquisa também essas entidades.

Para obter exemplos de código, consulte o directório samples.

Trabalhar com a administração do sistema

O DB2 Information Integrator for Content faculta as classes e APIs para que o utilizador possa aceder às funções de administração do sistema.

Consulte Application Programming Reference para obter informações relativas às classes específicas.

Personalizar o cliente de administração do sistema do DB2 Information Integrator for Content

Personalize o cliente de administração do sistema do DB2 Information Integrator for Content de acordo com os seus requisitos.

O cliente de administração do sistema do DB2 Information Integrator for Content suporta a expansão da aplicação de administração do sistema para incluir as funções de personalização:

- O utilizador pode substituir os diálogos de utilizador e de grupo de utilizadores no cliente de administração do sistema de DB2 Information Integrator for Content por diálogos personalizados.
- Pode adicionar novos nós à hierarquia no cliente de administração do sistema de DB2 Information Integrator for Content.
- Pode adicionar novos itens de menu ao menu **Tools (Ferramentas)** no cliente de administração do sistema.

Pode invocar saídas de utilizador antes e depois de iniciar uma sessão na administração do sistema do DB2 Information Integrator for Content.

Programar com as interfaces de programação de aplicações (APIs)

As interfaces de programação de aplicações (APIs) são um conjunto de classes que acedem e manipulam dados locais ou remotos. Esta secção descreve as APIs, a implementação de várias funções de pesquisa e a conectividade da Internet.

As APIs suportam:

- Um modelo de objectos comum para o acesso a dados.
- Pesquisas e actualizações múltiplas numa combinação heterogénea de servidores de conteúdos .
- Um mecanismo flexível para utilizar uma combinação de motores de pesquisa; por exemplo, a função de pesquisa de texto do Content Manager.
- Capacidade de fluxo de trabalho.
- Funções de administração.
- Implementação cliente/servidor para aplicações de Java.

O suporte de sequência múltipla para as APIs de Java está totalmente activado só para servidores de Windows. Os servidores e clientes de AIX e os clientes de Windows não suportam sequência múltipla.

Tenha em atenção que o DB2 Content Manager usa a Interface de Nível de Chamada (CLI - Call Level Interface) para estabelecer ligação ao DB2 Universal Database, por isso, deverá utilizar todas as CLI do DB2 ou todas as SQL incorporadas, não misturadas, para gerir as ligações ou transacções nas aplicações.

Compreender as diferenças entre as APIs de Java e C++

Existem diferenças fundamentais entre as APIs de Java e de C++.

A lista que se segue descreve as diferenças entre os conjuntos de APIs de IBM DB2 Information Integrator for Content de Java e de C++:

- Os operadores definidos nas APIs de C++ não estão definidos nas APIs de Java. São suportados como funções de Java.
- O objecto de classe de Java (`java.lang.Object`) é utilizado em substituição da classe de `DKAny C++` para representar um objecto genérico.
- As constantes comuns e globais são definidas na interface `DKConstant` nas APIs de Java. Em C++, são definidas no ficheiro `DKConstant.h`.
- As APIs de Java usam o garbage collector de Java.
- As funções `DKDDO.toXML()` e `DKDDO.fromXML()` de Java não estão disponíveis em C++.
- As classes XML só estão disponíveis em Java.

Compreender a arquitectura cliente/servidor (apenas para Java)

As APIs facultam uma interface de programação conveniente para programadores de aplicações. As APIs podem residir no servidor ou no cliente de DB2 Information Integrator for Content (ambos facultam a mesma interface). A API cliente comunica com o servidor para aceder a dados através da rede, por intermédio de RMI

(Remote Method Invocation) de Java. A comunicação entre o cliente e o servidor é executada pelas classes; não é necessário adicionar quaisquer programas adicionais.

As classes de API são formadas pelos seguintes pacotes: servidor, cliente, cs e comum. As classes de servidor e de cliente fornecem as mesmas APIs, mas têm implementações diferentes.

- O pacote do servidor é `com.ibm.mm.sdk.server`. As classes do pacote do servidor comunicam directamente com o servidor de conteúdos federado ou de fundo.
- O pacote do cliente é `com.ibm.mm.sdk.client`. As classes do pacote do cliente comunicam com as classes do pacote do servidor por intermédio de RMI.
- As classes comuns são partilhadas pelo cliente e pelo servidor. Por vezes uma aplicação não sabe onde reside o conteúdo. Por exemplo, uma aplicação pode ter conteúdo que reside no cliente numa dada altura e no servidor noutra altura. O pacote `cs` liga dinamicamente o cliente e o servidor.

A aplicação cliente deve importar o pacote cliente, a aplicação dinâmica deve importar o pacote `cs` e a aplicação do servidor deve importar o pacote do servidor.

Apesar de se fornecer a mesma API para cliente e servidor, o pacote cliente tem um artigo de excepção adicional, visto que comunica com o pacote do servidor. No entanto, tenha em atenção que a interface cliente/servidor não é suportada por todos os conectores. Por exemplo, esta não é suportada pelo CM V8.

Criação de pacotes para o ambiente Java

As APIs do DB2 Information Integrator for Content encontram-se em quatro pacotes.

Os quatro pacotes que contêm APIs do DB2 Information Integrator for Content como parte de `com.ibm.mm.sdk`: `common`, `server`, `client` e `cs`.

server (`com.ibm.mm.sdk.server`)

Acede e manipula informações do servidor de conteúdos

client (`com.ibm.mm.sdk.client`)

Comunica com o pacote do servidor através do Remote Method Invocation (RMI)

common (`com.ibm.mm.sdk.common`)

Classes comuns para o pacote de servidor, pacote de cliente e pacote de `cs`.

cs (`com.ibm.mm.sdk.cs`)

Liga cliente ou servidor de forma dinâmica

A sua aplicação tem de utilizar o pacote `common` com o pacote `server` para aplicações locais, o pacote `client` para aplicações que acedam ao servidor remoto ou o pacote `cs`.

Sugestões de programação

Não importe pacotes de cliente e de servidor no mesmo programa.

Se estiver a desenvolver uma aplicação cliente, importe o pacote cliente. Caso contrário, importe o pacote do servidor. Se não souber onde reside o conteúdo, utilize o pacote `cs` (com os pacotes do servidor ou do cliente). A importação de vários pacotes pode provocar erros de compilação.

Alguns conectores contêm chamadas ao código C da implementação. No caso destes conectores, utilize o pacote cliente para aplicações da Web que necessitam de interfaces puras de Java. O pacote cliente é criado com programas puros de Java; o pacote do servidor pode incluir chamadas de JNI.

Visto que um cliente exige a exceção, `java.rmi.RemoteException`, anexe sempre esta exceção à aplicação, quer seja executada num servidor ou num cliente.

Configurar o ambiente Java (apenas para Java)

Para configurar o ambiente Java, é necessário importar vários pacotes e configurar as variáveis de Java.

Ao configurar o seu ambiente de Windows, AIX, Solaris ou Linux, deverá ter importado os seguintes pacotes:

pacote do servidor

Importe quando um servidor de conteúdos e uma aplicação estão do lado do servidor

- `com.ibm.mm.sdk.common`
- `com.ibm.mm.sdk.server`

pacote do cliente

Importe quando um servidor de conteúdos e uma aplicação estão do lado do cliente.

- `com.ibm.mm.sdk.common`
- `com.ibm.mm.sdk.client`

pacote cs

Importe quando a localização de um servidor de conteúdos é diferente da localização da aplicação.

- `com.ibm.mm.sdk.common`
- `com.ibm.mm.sdk.cs`

Definir as variáveis de ambiente Java para Windows

Pode executar `cmbenv81.bat` numa linha de comandos de Windows para configurar o ambiente.

Caso pretenda modificar as variáveis de ambiente, altere o seguinte:

PATH Certifique-se de que o PATH contém `%IBMCMROOT%`.

CLASSPATH

Certifique-se de que o CLASSPATH contém `%IBMCMROOT%`.

Definir as variáveis de ambiente Java para AIX

No ambiente de AIX, pode utilizar um procedimento base, `cmbenv81.sh`, para configurar o ambiente de desenvolvimento de aplicações de DB2 Information Integrator for Content.

Se não utilizar o script, deve definir as seguintes variáveis de ambiente:

PATH Certifique-se de que o PATH contém `/opt/IBM/db2cmv8`.

LIBPATH

Certifique-se de que o LIBPATH contém `/opt/IBM/db2cmv8`.

LD_LIBRARY_PATH

Certifique-se de que o LD_LIBRARY_PATH contém /opt/IBM/db2cmv8/lib.

CLASSPATH

Certifique-se de que CLASSPATH contém /opt/IBM/db2cmv8/lib/xxx, em que xxx corresponde aos ficheiros .jar, (por exemplo, cmbfed81.jar)

Definir as variáveis de ambiente Java para Solaris e Linux

Para configurar o ambiente para Solaris, utilize um procedimento base ou defina as variáveis de ambiente.

No ambiente Solaris, pode utilizar um procedimento base, cmbenv81.sh, para configurar o ambiente de desenvolvimento de aplicações de DB2 Information Integrator for Content.

Se não utilizar o script, deve definir as seguintes variáveis de ambiente:

PATH Certifique-se de que o PATH contém /opt/IBM/db2cmv8/lib.

LIBPATH

Certifique-se de que o LIBPATH contém /opt/IBM/db2cmv8/lib.

LD_LIBRARY_PATH

Certifique-se de que o LD_LIBRARY_PATH contém /opt/IBM/db2cmv8/lib.

CLASSPATH

Certifique-se de que CLASSPATH contém /opt/IBM/db2cmv8/lib/xxx, em que xxx corresponde aos ficheiros .jar, (por exemplo, cmbfed81.jar)

Definir variáveis de ambiente Java para z/OS USS

No ambiente z/OS USS, pode utilizar um procedimento base (cmbenv81.sh) para configurar o ambiente de desenvolvimento de aplicações de DB2 Information Integrator for Content.

Se não utilizar o script, deve definir as seguintes variáveis de ambiente:

PATH Certifique-se de que PATH contém \$IBMCMROOT/cmgmt, \$IBMCMROOT/cmgmt/connectors, \$IBMCMROOT/lib

LIBPATH

Certifique-se de que LIBPATH contém \$IBMCMROOT/lib

LD_LIBRARY_PATH

Certifique-se de que LD_LIBRARY_PATH contém \$IBMCMROOT/lib

CLASSPATH Certifique-se de que CLASSPATH contém

\$IBMCMROOT/lib/xxx, em que xxx correspondem aos ficheiros .jar, por exemplo, cmbcm81.jar.

Utilizar Remote Method Invocation (RMI) com servidores de conteúdos

Visto que as classes de clientes nas APIs de Java têm de comunicar com as classes de servidor para aceder a dados através da rede, tanto o servidor como o cliente têm de estar preparados para a execução cliente/servidor. Na máquina do servidor, o servidor de RMI tem de estar em execução para receber o pedido do cliente que utiliza um número de porta especificado. O programa cliente exige o nome do

servidor e o número de porta. Para realizar comunicações entre cliente e servidor, o cliente deve saber qual o número de porta do servidor ao qual necessita estabelecer uma ligação.

Um servidor de RMI pode estabelecer ligação a um número infinito (apenas limitado pelos recursos do sistema) de servidores de conteúdos, mas cada servidor deve estar ligado a, pelo menos, um servidor de conteúdos. Um servidor de RMI principal pode consultar outros servidores de RMI no conjunto de servidores. Quando um cliente de RMI pesquisa pela primeira vez um servidor de conteúdos, este inicia um servidor de RMI. Se o servidor de conteúdos não for encontrado, os servidores do conjunto RMI são pesquisados de seguida.

Se o mesmo cliente de RMI pesquisar novamente o servidor de conteúdos, o cliente pesquisa o servidor de RMI no qual encontrou pela primeira vez o servidor de conteúdos.

Para iniciar o servidor de RMI, utilize o `cmbregist81.bat` em Windows ou `cmbregist81.sh` em AIX ou Solaris. Antes de iniciar o servidor de RMI, defina o número de porta correcto e o tipo de servidor. Para obter informações relativas à configuração e administração de servidores de RMI, consulte as informações de planeamento e instalação e as informações administrativas.

Importante: O conector de ICM não suporta a configuração de RMI.

Configurar o ambiente C++ (Apenas para C++)

Para configurar o ambiente C++, é necessário definir as variáveis de ambiente e o sinalizador de compilação.

Ao configurar o ambiente do Windows ou o ambiente de AIX, tem de estabelecer as definições descritas nesta secção. A Tabela 3 na página 22 enumera os requisitos de Biblioteca, AIX partilhado e DLL.

Não se esqueça: Para utilizar C++, é necessário instalar o suporte de cliente e o Assistente de configuração de cliente de DB2 em todos os servidores remotos que acedem à base de dados do DB2 Information Integrator for Content. O ID de utilizador e palavra-passe utilizados para estabelecer uma ligação à base de dados devem ser iguais ao ID de utilizador e palavra-passe que utiliza na base de dados do DB2 Information Integrator for Content. Para obter mais detalhes, consulte as informações administrativas.

Importante: O `cmbcm81x.lib` destina-se à construção de edições e o `cmbcm81xd.lib` destina-se à construção de depuração, sendo que *x* representa o compilador Microsoft Visual C++ .NET Versão 2003 (71) ou Microsoft Visual C++ Versão 2005 (8).

Sugestão:

O compilador Microsoft Visual Studio 2005 pode ser instalado na mesma estação de trabalho que o Microsoft Visual Studio .NET 2003.

O compilador Microsoft Visual C++ 2005 pode ser também utilizado para compilar saídas de utilizadores de servidores de bibliotecas para o DB2 Content Manager instalado em Windows.

Para compilar aplicações do DB2 Content Manager utilizando o compilador .NET 2003, é necessário adicionar o sinalizador de compilação MSVC71 aos ficheiros .mak ou aos ficheiros de projectos. Por exemplo, nas definições do sinalizador de compilação, adicione /D MSVC71. Se não adicionar o sinalizador de compilação, a aplicação não vai compilar. Para obter mais informações, verifique os ficheiros de procedimentos exemplo de C++.

Para compilar aplicações do DB2 Content Manager utilizando o compilador Microsoft Visual C++ 2005, é necessário adicionar o sinalizador de compilação MSVC80 e MSVC71 nos ficheiros .mak ou nos ficheiros de projectos. Por exemplo, nas definições do sinalizador de compilação, adicione /D MSVC80 e /D MSVC71. Se não adicionar o sinalizador de compilação, a aplicação não vai compilar. Para obter mais informações, verifique os ficheiros de procedimentos exemplo de C++.

Tabela 3. Informações relativas a objectos partilhados e ambiente de DLL

Conector	Biblioteca	DLLs do Windows	Objectos partilhados em AIX
Nota Comum: Isto não é um conector. É um conjunto comum de APIs utilizado por todos os conectores.	cmbcm81x.lib cmbcm81xd.lib	cmbcm81x.dll cmbcm81xd.dll	libcmbcm817.a
DB2 Content Manager Versão 8	cmbicm81x.lib cmbicm81xd.lib	cmbicm81x.dll cmbicm81xd.dll cmbicmfac81x.dll cmbicmfac81xd.dll	libcmbicm817.a libcmbicmfac817.so
Federado	cmbfed81x.lib cmbfed81xd.lib	cmbfed81x.dll cmbfed81xd.dll cmbfedfac81x.dll cmbfedfac81xd.dll	libcmbfed817.a libcmbfedfac817.so
OnDemand	cmbod81x.lib cmbod81xd.lib	cmbod81x.dll cmbod816xd.dll cmbodfac81x.dll cmbodfac81xd.dll	libcmbod817.a libcmbodfac817.so
ImagePlus for OS/390	cmbip81x.lib cmbip81xd.lib	cmbip81x.dll cmbip81xd.dll cmbipfac81x.dll cmbipfac81xd.dll	Não suportado
VisualInfo	cmbv481x.lib cmbv481xd.lib	cmbv481x.dll cmbv481xd.dll cmbv4fac81x.dll cmbv4fac81xd.dll	Não suportado

Definir as variáveis de ambiente C++ para Windows

Pode executar cmbenv81.bat numa linha de comandos DOS para configurar o ambiente.

Caso pretenda modificar as variáveis de ambiente, altere o seguinte:

PATH Defina `PATH=%PATH%;%IBMCMROOT%\DLL` se pretender incluir o directório DLL em PATH.

INCLUDE

`INCLUDE=%INCLUDE%;%IBMCMROOT%`

Definir as variáveis de ambiente C++ para AIX

Para configurar as variáveis de ambiente C++ para AIX, pode utilizar o ficheiro de lotes cmbenv81.sh.

Para configurar as variáveis de ambiente C++:

Defina as seguintes variáveis de ambiente:

1. No ambiente AIX, pode utilizar o ficheiro de lotes cmbenv81.sh para configurar o ambiente de desenvolvimento.
2. Se não utilizar o script, defina as seguintes variáveis de ambiente:

caminho de NLS

```
export NLSPATH=${NLSPATH}:/opt/IBM/db2cmv8/msg/En_US/%N
```

PATH

```
PATH=${PATH}:/opt/IBM/db2cmv8/lib
```

LIBPATH

```
export LIBPATH=${LIBPATH}:/opt/IBM/db2cmv8/lib
```

Consulte os ficheiros MAK exemplo no directório samples para obter mais informações.

Construir programas C++

Siga os procedimentos relativos ao compilador e ambiente de desenvolvimento para criar os ficheiros MAK e construir a aplicação.

Não existe qualquer depuração construída no AIX.

Restrição: As APIs C++ do Content Manager V8 não suportam Unicode.

Definir o subsistema da consola para conversão de página de códigos em Windows

O exemplo mostra como configurar o subsistema da consola para conversão de página de códigos em Windows.

Exemplo: C++

Para configurar o subsistema da consola para C++

```
#include <DKConstant.h>
#include <DKEnvironment.hpp>

void main(int argc, char *argv[])
{
    // set sub system to console at the beginning of program this
    // will cause the code page that the error messages are returned
    // in by DKExceptions to be converted from the Windows Graphical
    // User Interface (ANSI format) to the Console (OEM format)
    // If this is not specified the default is DK_SS_WINDOWS
    DKEnvironment::setSubSystem(DK_SS_CONSOLE);
    ...
}
```

Desempenho de aplicações de Java e C++

Para o DB2 Content Manager Versão 8.4, as APIs de Java e C++ fornecem várias funções que melhoram o desempenho, a capacidade de utilização e a escalabilidade de aplicações de API.

Colocação da API de Java em memória cache global

Pode utilizar a cache global para melhorar o desempenho e a escalabilidade de aplicações de API de Java.

Por predefinição, cada instância de DKDatastoreICM armazena determinados objectos (instâncias de DKDatastoreICM) na memória numa cache local (o que significa que as outras instâncias de DKDatastoreICM não podem aceder à mesma). Estes objectos podem incluir tipos de artigo, vistas de tipos de artigo, tipos de componente, vistas de tipos de componente, etc. A colocação em memória cache melhora os tempos de resposta evitando que a API de Java comunique repetidamente com o servidor de bibliotecas para diagnosticar estes objectos.

Ao activar a cache global, pode consolidar as caches de várias instâncias de DKDatastoreICM numa área global. A colocação em cache global impede a duplicação de objectos idênticos e maximiza o desempenho e a escalabilidade de um ambiente intermédio. Se activar a cache global, os seguintes tipos de objectos são armazenados na área global e não nas áreas locais:

- Atributo
- Grupo de atributos
- Tipo de evento
- Tipo de ligação
- Classificação de XDO
- Tipo Mime
- Palavras-chave de NLS
- Tipo semântico
- Privilégio
- Grupo de privilégios

Para activar a cache global, edite o ficheiro *IBMCMROOT/cmgmt/connectors/cmbicmcache.ini* e defina o seguinte: `JAVAGlobalCacheEnabled=TRUE`.

É necessário limpar a cache global sempre que forem efectuadas modificações (tais como criação, actualização ou eliminação) a qualquer um dos objectos de definição, excepto a palavras-chave de NLS. Para obter exemplos de limpeza de caches, consulte os exemplos que se seguem.

Limpar as caches global e local de todos os arquivos de dados para uma aplicação que utilize implementação própria de conjunto de arquivos de dados

```
dsPool.clearCacheOfConnections(DK_CM_CLEAR_CACHE_ALL);
```

Este exemplo garante que a cache global é limpa uma vez e que a cache local de todos os arquivos de dados são limpas.

Limpar a cache local e a cache global

```
dsICM.clearCache(DK_CM_CLEAR_CACHE_ALL);
```

Limpar apenas a cache local

```
dsICM.clearCache(DK_CM_CLEAR_CACHE_LOCAL);
```

Limpar apenas a cache global

```
dsICM.clearCache(DK_CM_CLEAR_CACHE_GLOBAL);
```

Se a aplicação não definir a opção através da utilização do método `setOption()` mas invocar `clearCache()`, é utilizado o método `DK_CM_CLEAR_CACHE_ALL`.

Opções de obtenção renovada de APIs nativas

Para melhorar a capacidade de utilização, o desempenho, a documentação e o suporte de APIs nativas, o DB2 Content Manager renova a interface de opções de obtenção e a implementação subjacente. As opções renovadas proporcionam maior controlo granular à aplicação para um melhor controlo do desempenho e permitem um grande número de optimizações incorporadas.

Cada definição de opções de obtenção inclui uma explicação detalhada, considerações de desempenho, considerações de ordem de precedência, estruturas de dados (como localizar os dados solicitados nos DKDDOs), o valor predefinido, notas sobre o comportamento, documentação detalhada de excepções, etc. Além disso, inclui funções como, por exemplo, uma projecção geral única e simples em todos os tipos de artigo ao construir filtros de atributos durante a utilização das novas listas de projecção firmemente integradas.

O DB2 Content Manager fornece uma classe de `DKRetrieveOptionsICM` e uma classe integrada de `DKProjectionListICM`. As opções de obtenção granular foram concebidas para selecção dos dados requeridos com base em passos incrementais com impacto no desempenho. Além das optimizações incorporadas activadas automaticamente através das opções, pode conseguir o melhor desempenho de obtenção possível obtendo apenas o subconjunto de dados que pretende utilizar e as considerações de desempenho. Adicionalmente, são possível várias combinações de selecções de obtenção, tais como a obtenção de um nível de elementos descendentes, a inclusão de atributos descendente e a obtenção de listas de partes com atributos de partes.

Para obter mais informações, consulte o Javadoc detalhado para a classe de `DKRetrieveOptionsICM`.

Importante: As variantes de opções de obtenção números inteiros relativos a bits e sem opções de métodos de obtenção que assumem uma opção 'int' relativa a bits como comportamento predefinido estão obsoletas. Não é necessário efectuar alterações nas aplicações existentes. As opções de obtenção de números inteiros relativos a bits anteriores ainda são suportadas, não existindo qualquer alteração no comportamento. Contudo, para todos os novos desenvolvimentos ou ao modificar aplicações existentes, utilize as opções renovadas para um melhor desempenho.

Escalabilidade e desempenho da API de C++: colocação em memória cache global e agrupamento de ligações

A cache global, o conjunto de arquivos de dados e o conjunto de ligações da base de dados ajudam a melhorar o desempenho e a escalabilidade de aplicações de API de C++.

Para maximizar o desempenho e a escalabilidade, utilize a cache global, o conjunto de arquivos de dados e os conjuntos de ligações da base de dados em simultâneo para os seguintes tipos de cenários:

- Ambientes com várias instâncias de DKDatastoreICM, especialmente os que utilizem servidores intermédios como o DB2 Document Manager. Estes ambientes incluem aplicações personalizadas, tais como aplicações Microsoft .NET, COM e DCOM, nas quais a API de C++ API é executada num ambiente de servidor intermédio.
- Aplicações que utilizem múltiplos módulos, especialmente módulos de processamento intermédio que repitam o seguinte padrão:
 1. Ligação ao DB2 Content Manager.
 2. Execução de uma operação (como, por exemplo, operações de criação, obtenção, actualização, eliminação e consulta de tempo de execução ou tempo de administração/definição).
 3. Anulação da ligação ao DB2 Content Manager.

As três funções da API de C++ funcionam da seguinte forma:

Cache global

Por predefinição, cada instância de DKDatastoreICM armazena determinados objectos (instâncias de DKDatastoreICM) na memória numa cache local (o que significa que as outras instâncias de DKDatastoreICM não podem aceder à mesma). Estes objectos podem incluir tipos de artigo, vistas de tipos de artigo, tipos de componente, vistas de tipos de componente, entre outros. A colocação em memória cache melhora os tempos de resposta evitando que a API de C++ API comunique repetidamente com o servidor de bibliotecas para diagnosticar estes objectos.

Ao activar a cache global, pode consolidar as caches de várias instâncias de DKDatastoreICM numa área global. A colocação em memória cache global impede a duplicação de objectos idênticos. Se activar a cache global, os seguintes tipos de objectos são armazenados na área global e não nas áreas locais:

- Atributo
- Grupo de atributos
- Tipo de evento
- Tipo de ligação
- Classificação de XDO
- Tipo MIME
- Palavras-chave de NLS (National Language Support)
- Tipo semântico

Para activar a cache global, crie manualmente um novo ficheiro de configuração, *IBMCMROOT/db2cmv8/cmgt/connectors/cmbicmcache.ini*, e inclua a seguinte linha no ficheiro .ini:
`CPPGlobalCacheEnabled=TRUE.`

É necessário limpar a cache global sempre forem efectuadas modificações (tais como criação, actualização ou eliminação) a qualquer um dos objectos de definição, excepto a palavras-chave de suporte de idioma nacional.

Conjunto de arquivos de dados

Num processo, os módulos que tenham de aceder ao servidor do DB2 Content Manager podem obter objectos de DKDatastore ligados e com

sessão iniciada a partir de um conjunto de arquivos de dados e, em seguida, devolvê-los ao conjunto quando o processo estiver concluído. Em particular, os conjuntos de arquivos de dados podem diminuir os tempos de resposta no servidor intermédio. Em combinação com a cache global, o conjunto de arquivos de dados reduz o número de ligações e chamadas ao servidor de bibliotecas.

Um arquivo de dados não requer configuração.

Conjunto de ligações da base de dados

O conjunto de ligações da base de dados é utilizado internamente pela API do DB2 Content Manager Versão 8 para agrupar ligações da base de dados na base de dados do servidor de bibliotecas. Este procedimento é semelhante ao modo como o conjunto de arquivos de dados é utilizado para agrupar arquivos de dados. O conjunto mantém ligações activas para a base de dados do servidor de bibliotecas que são utilizadas quando necessárias. O conjunto torna o acesso ao servidor de bibliotecas mais rápido, uma vez que não é necessária ligação de base de dados. Além disso, minimiza o número de ligações da base de dados.

Para activar o conjunto de ligações da base de dados, modifique o ficheiro de configuração: *IBMCMROOT/db2cmv8/cmgmt/connectors/cmbpool.ini*, inserindo a seguinte linha no ficheiro .ini:

```
CPPDBConnectionPoolMaxSize = N
```

em que *N* representa o maior número de ligações da base de dados que o administrador pretende permitir para qualquer processo de aplicações de C++. A definição da propriedade *CPPDBConnectionPoolMaxSize* como 0 desactiva o conjunto de ligações da base de dados.

Requisito: A propriedade *CPPDBConnectionPoolMaxSize* não deverá exceder 70 caso o nível de registo no ficheiro *cmblogconfig.properties* esteja definido como um dos seguintes valores: INFO, TRACE_NATIVE_API, TRACE_ENTRY_EXIT, TRACE ou DEBUG.

Recomendação: Se o conjunto de ligações da base de dados estiver activado, certifique-se de que a aplicação encerra o conjunto de ligações ao terminar. A activação garante um encerramento sem problemas com uma libertação atempada de recursos. O seguinte método estático na instância de *DKDatastoreICM* deverá ser utilizado para encerrar todas as ligações no conjunto de ligações da base de dados:

```
static DKEXPORT void closeDBConnectionPool();
```

Limpar a cache local e a cache global

Para limpar as caches local e global:

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();  
dsICM->setOption(DK_CM_OPT_CLEAR_CACHE, DK_CM_CLEAR_CACHE_ALL);  
dsDefICM->clearCache();
```

Limpar apenas a cache local

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();  
dsICM->setOption(DK_CM_OPT_CLEAR_CACHE, DK_CM_CLEAR_CACHE_LOCAL);  
dsDefICM->clearCache();
```


Limpar apenas a cache global

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();  
dsICM->setOption(DK_CM_OPT_CLEAR_CACHE, DK_CM_CLEAR_CACHE_GLOBAL);  
dsDefICM->clearCache();
```

Se a aplicação não definir a opção através da utilização do método `setOption()` mas invocar `clearCache()`, é utilizado o método `DK_CM_CLEAR_CACHE_ALL`.

Limpar as caches global e local de todos os arquivos de dados para uma aplicação que utilize implementação própria de conjunto de arquivos de dados

Para a primeira instância de `DKDatastoreICM` no conjunto, adicione este código à aplicação:

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();  
dsICM->setOption(DK_CM_OPT_CLEAR_CACHE, DK_CM_CLEAR_CACHE_ALL);  
dsDefICM->clearCache();
```

Para as restantes instâncias de `DKDatastoreICM`:

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();  
dsICM->setOption(DK_CM_OPT_CLEAR_CACHE, DK_CM_CLEAR_CACHE_LOCAL);  
dsDefICM->clearCache();
```

Este exemplo garante que a cache global é limpa uma vez e que a cache local de todos os arquivos de dados são limpas.

Limpar as caches global e local de todos os arquivos de dados para uma aplicação que utilize uma implementação de API de C++ de conjunto de arquivo de dados

Para limpar as caches global e local:

```
dsPool->clearCacheOfConnections();
```

Expirar o tempo de ligações com a API de C++

O DB2 Content Manager permite a expiração do tempo de ligações no conjunto de bases de dados.

A API de C++ suportava um mecanismo para agrupar ligações da base de dados do DB2 Content Manager Enterprise Edition Versão 8 Edição 3 Pacote de correcções 2 e posteriores. O conjunto de ligações da base de dados é utilizado pela API de C++ do DB2 Content Manager Versão 8 para agrupar ligações de base de dados na base de dados do servidor de bibliotecas. O conjunto de ligações mantém as ligações activas para a base de dados do servidor de bibliotecas que são utilizadas quando necessárias. O servidor obtém uma ligação do conjunto quando é iniciada uma transacção (implícita ou explícita) e devolve a ligação ao conjunto quando a transacção é concluída. Esta acção permite a reutilização máxima das ligações no conjunto e minimiza o número de ligações de base de dados necessárias.

Para activar o conjunto de ligações da base de dados, modifique o ficheiro de configuração em `%ibmcmroot%/db2cmv8/cmgt/connectors/cmbpool.ini` e introduza a seguinte linha:

```
CPPDBConnectionPoolMaxSize = N
```

em que *N* representa o maior número de ligações da base de dados que o administrador pretende permitir para qualquer processo de aplicações de C++. A

definição de `CPPDBConnectionPoolMaxSize` como 0 desactiva o conjunto de ligações da base de dados. Consulte a secção "Escalabilidade e desempenho da API de C++: colocação em memória cache global e agrupamento de ligações" no *Ficheiro Readme do DB2 Content Manager Enterprise Edition Versão 8 edição 3 Pacote de correcções 2* para obter mais informações sobre o tipo de cenários nos quais o conjunto de ligações da base de dados da API de C++ deve ser activado.

O DB2 Content Manager Enterprise Edition activa um mecanismo de tempo limite para remover as ligações não utilizadas no conjunto de ligações para a API de C++. A expiração do tempo limite de uma ligação resulta na anulação dessa ligação e consequente remoção do conjunto. O valor de tempo limite, em minutos, pode ser definido no ficheiro `cmbpool.ini` (localizado em `%ibmcmroot%/cmgmt/connectors`) e é efectivo apenas se o conjunto de ligações da base de dados de C++ estiver activado. Está disponível uma nova propriedade denominada `CPPDBConnectionPoolTimeOut` para expiração do tempo limite de ligações não utilizadas no conjunto de ligações da base de dados.

Importante: Os valores para `CPPDBConnectionPoolMaxSize` e `CPPDBConnectionPoolTimeOut` estão definidos no ficheiro `cmbpool.ini`.

As ligações não utilizadas são ligações que não são obtidas (ou utilizadas) do conjunto durante o período indicado pela propriedade `CPPDBConnectionPoolTimeOut`. Cada ligação possui uma hora de última ligação, que é utilizada para determinar se a ligação não foi utilizada durante o período de tempo indicado pelo valor de `CPPDBConnectionPoolTimeOut`. Se a ligação não tiver sido utilizada durante o período de tempo indicado pela propriedade `CPPDBConnectionPoolTimeOut`, a ligação é removida do conjunto. A remoção de ligações não utilizadas anula também a atribuição dos recursos no servidor de bibliotecas e na API. O mecanismo de tempo limite da base de dados é desactivado através da definição do valor da propriedade `CPPDBConnectionPoolTimeOut` como 0. Se este valor não for definido no ficheiro `cmbpool.ini` e conjunto de ligações da base de dados for activado, é utilizado um valor predefinido de 5 minutos.

Importante: É necessário actualizar manualmente o ficheiro `cmbpool.ini` para definir os valores de tempo limite apropriados para o ambiente.

Indicar elementos obtidos ao extrair a lista de ACLs

O DB2 Content Manager fornece APIs para a classe de `DKAuthorizationMgmtICM` que permite às aplicações indicar os elementos que necessitam de ser obtidos durante a extracção da lista ACLs no sistema.

As APIs `listAccessControlLists(*, retrievalOption)` e `listUserAccessControlLists(*, retrievalOption)` proporcionam maior flexibilidade às aplicações na determinação dos elementos que têm de ser obtidos ao enumerar ACLs (* significa 0 ou mais parâmetros para `listAccessControlLists` e `listUserAccessControlLists`).

O DB2 Content Manager faculta ainda outra API para a classe de `DKAuthorizationMgmtICM`, `listPrivilegeSets(*, options)` que permite às aplicações ter maior flexibilidade na determinação dos elementos que têm de ser obtidos ao enumerar conjuntos de privilégios. As opções podem corresponder a uma das seguintes:

DK_ICM_PRIVILEGE_SET_RETRIEVAL_OPTION_NAME_ONLY

Devolve uma recolha de objectos de conjunto de privilégios (instância de DKPrivilegeSetICM). Cada conjunto de privilégios contém apenas o nome do conjunto de privilégios.

DK_ICM_PRIVILEGESET_RETRIEVAL_OPTION_ATTRIBUTES_ONLY

Devolve uma recolha de objectos de conjunto de privilégios (instância de DKPrivilegeSetICM). Cada conjunto de privilégios contém as seguintes propriedades:

- Nome do conjunto de privilégios
- Descrição do conjunto de privilégios
- Recolha de domínios aos quais o conjunto de privilégios pertence

DK_ICM_PRIVILEGESET_RETRIEVAL_OPTION_ALL

Devolve uma recolha de objectos de conjunto de privilégios (instância de DKPrivilegeSetICM). Cada privilégio contém o seguinte:

- Nome do conjunto de privilégios
- Descrição do conjunto de privilégios
- Recolha de domínios aos quais o conjunto de privilégios pertence
- Recolha de privilégios que constituem o conjunto de privilégios (cada privilégio é uma instância da classe de DKPrivilegeICM)

Importante: Os objectos de DKPrivilegeSetICM devolvidos por DK_ICM_PRIVILEGE_SET_RETRIEVAL_OPTION_NAME_ONLY e DK_ICM_PRIVILEGESET_RETRIEVAL_OPTION_ATTRIBUTES_ONLY estão incompletos, uma vez que todas as propriedades não se encontram preenchidas. Não utilize o objecto incompleto para executar quaisquer actualizações. Em alternativa, utilize a API `retrievePrivilegeSet(long id)` para obter o objecto totalmente formatado, actualize o objecto totalmente formatado e utilize-o para actualizar as informações do conjunto de privilégios no arquivo persistente.

Ambas as APIs `listAccessControlLists(*, retrievalOption)` e `listPrivilegeSets(*, options)` estão disponíveis para Java e C++. Para obter mais informações, consulte o manual *Application Programming Reference*

Compreender as várias opções de pesquisa

As pesquisas podem ser executadas com base em, virtualmente, qualquer parte de um artigo ou componente, texto num artigo ou componente ou texto num conteúdo de recurso.

O Content Manager possui um motor de pesquisa e três opções relativas ao modo e ao momento em que as pesquisas devem ser executadas (e os resultados devolvidos): `Execute`, `Evaluate` e `Execute with Callback`. Consulte o exemplo `SSearchICM` para obter a tabela e as explicações.

Tratar excepções

Para resolver problemas que surgem nas aplicações API, pode utilizar o tratamento de excepções.

Quando as APIs detectam um problema, estas devolvem uma excepção. O envio de uma excepção cria um objecto de excepções da classe de DKException ou uma das respectivas subclasses.

Quando é criada uma DKException, o nível do conector regista informações de diagnóstico num ficheiro de registo, assumindo que é utilizada a configuração de registo.

Quando é capturada uma DKException, esta permite-lhe visualizar mensagens de erro, códigos de erro e estados de erro ocorridos durante a execução. Quando um erro é capturado, é emitida uma mensagem de erro juntamente com a localização do surgimento da excepção. Também são facultadas informações adicionais como, por exemplo, de ID de erro.

O código seguinte apresenta um exemplo do processo de libertação e obtenção em DB2 Information Integrator for Content e CM:

Exemplo: Java

```
try{
    ... EIP API Operations ...
}
catch (DKException exc){
    // NOTA: Função de Impressão fornecida no exemplo de API
    // SConnectDisconnectICM.
    System.out.println("");
    System.out.println("XXXXXXXXXXXXXXXXXXXXXXXXXXXX");
    System.out.println("X    !!! Exception !!!    X");
    System.out.println("XXXXXXXXXXXXXXXXXXXXXXXXXXXX");
    System.out.println("        Name: " + exc.name());
    System.out.println(" Message: " + exc.getMessage());
    System.out.println(" Message ID: " + exc.getErrorId());
    System.out.println("Error State: " + exc.errorState());
    System.out.println(" Error Code: " + exc.errorCode());
    exc.printStackTrace();
    System.out.println("-----");
} catch (Exception exc) {
    // NOTA: Função de Impressão fornecida no exemplo de API
    // SConnectDisconnectICM.
    System.out.println("");
    System.out.println("XXXXXXXXXXXXXXXXXXXXXXXXXXXX");
    System.out.println("X    !!! Exception !!!    X");
    System.out.println("XXXXXXXXXXXXXXXXXXXXXXXXXXXX");
    System.out.println("        Name: " + exc.getClass().getName());
    System.out.println(" Message: " + exc.getMessage());
    exc.printStackTrace();
    System.out.println("-----");
}
```

Exemplo: C++

```
try{
    ... EIP API Operations ...
}
catch(DKException &exc) {
    // NOTA: Função de Impressão fornecida no exemplo de API
    // SConnectDisconnectICM.
    cout << endl;
    cout << "XXXXXXXXXXXXXXXXXXXXXXXXXXXX" << endl;
    cout << "X    !!! Exception !!!    X" << endl;
    cout << "XXXXXXXXXXXXXXXXXXXXXXXXXXXX" << endl;
    cout << "        Name: " << exc.name() << endl;
    cout << " Message ID: " << exc.errorId() << endl;
    cout << "Error State: " << exc.errorState() << endl;
```

```

        cout << " Error Code: " << exc.errorCode() << endl;
        // Print API Location(s) Detecting Error
        for(unsigned int j=0; j < exc.locationCount(); j++){ //Print all locations
            const DKExceptionLocation* p = exc.locationAtIndex(j);
            cout << " Location " << j << ": " << p->fileName() << " :: "
                << p->functionName() << '[' << p->lineNumber() << ']' << endl;
        }
        if(exc.textCount()<=0) // Escrever instrução se não houver localizações.
            cout << " Locations: <none> " << endl;
        // Error Message(s)
        for(unsigned int i=0; i < exc.textCount(); i++)
            // Imprimir todas as mensagens
            cout << " Message " << i << ": " << exc.text(i) << endl;
        if(exc.textCount()<=0) // Notify user if no messages.
            cout << " Messages: <none> " << endl;
        cout << "-----" << endl;
    }
}

```

Para obter mais informações relativas à detecção e tratamento de erros, consulte o Exemplo Formativo da API SConnectDisconnectICM.

Constantes

As constantes especificadas encontram-se na forma de DK_CM_ (Constantes comuns) ou DK_XX_ (em que XX indica diferentes servidores de conteúdos).

As constantes especificadas encontram-se na forma de DK_CM_ (Constantes comuns) ou DK_XX_ (em que XX indica diferentes servidores de conteúdos). Consulte a Tabela 4 na página 33 para obter uma lista das extensões (extensões anexadas a cada DKDatastore).

Ao especificar constantes de DDO, utilize DK_CM_DATAITEM_TYPE_ ... (por exemplo, DK_CM_DATAITEM_TYPE_STRING) para tipos de propriedades. Para tipos de atributos, utilize as constantes DK_CM_...type (por exemplo, DK_CM_INTEGER).

Java

As constantes comuns estão definidas em DKConstant.java. Também pode rever uma versão de texto destas constantes em DKConstant.txt. As constantes de servidores de conteúdos específicos são definidas em ficheiros DKConstantXX.java; por exemplo, as constantes exclusivas de Content Manager encontram-se em DKConstantICM.java.

C++

As constantes comuns estão definidas em DKConstant.h. Para obter uma lista de constantes e dos valores correspondentes, visualize DKConstant2.h (mas não o inclua no seu programa). As constantes de servidores de conteúdos específicos são definidas em ficheiros de cabeçalho de formato DKConstantX.h; por exemplo, as constantes exclusivas do Content Manager encontram-se no formato DKConstantICM.h.

Ligação a servidores de conteúdos

Um objecto da classe DKDatastorexx (em que xx indica um servidor de conteúdos específico) representa e gere uma ligação a um servidor de conteúdos, facultando suporte de transacção e executa comandos do servidor.

Um objecto da classe DKDatastorexx (em que xx indica um servidor de conteúdos específico) representa e gere uma ligação a um servidor de conteúdos, facultando

suporte de transacção e executa comandos do servidor. Consulte a Tabela 4 para obter as extensões exactas.

Caso a aplicação termine repentina ou anormalmente, tal como quando o utilizador insere Ctrl-C na aplicação, certifique-se de que os arquivo de dados ligados são desligados pela aplicação.

Tabela 4. Tipo de servidor e terminologia do nome da classe

Servidor de conteúdos	Nome da classe
DB2 Content Manager Versão 8	DKDatastoreICM
OnDemand	DKDatastoreOD
DB2 Content Manager for AS/400 (VisualInfo para AS/400)	DKDatastoreV4
ImagePlus for OS/390	DKDatastoreIP

Estabelecer uma ligação

Cada classe de DKDatastorexx fornece métodos de ligação e anulação da ligação à mesma.

O exemplo que se segue utiliza um servidor de bibliotecas do DB2 Content Manager denominado ICMNLSDB, o ID de utilizador ICMADMIN e a palavra-passe PASSWORD. O exemplo cria um objecto de DKDatastoreICM para o servidor de conteúdos do CM, estabelece ligação ao mesmo, trabalha com o mesmo e anula a ligação estabelecida.

Exemplo: Java

```
DKDatastoreICM dsICM = new DKDatastoreICM(); //Create datastore object
dsICM.connect("ICMNLSD", "ICMADMIN", "PASSWORD", ""); //Connect to datastore

System.out.println("Connected to datastore dbase: '" + dsICM.datastoreName() +
    "', UserName '" + dsICM.userName() + "'.");

dsICM.disconnect(); // Disconnect from datastore
dsICM.destroy(); // Destroy reference
```

Para obter a aplicação exemplo completa, consulte o exemplo SConnectDisconnectICM.java.

C++

```
DKDatastoreICM* dsICM = new DKDatastoreICM(); //Create datastore object
dsICM->connect("ICMNLSD", "ICMADMIN", "PASSWORD", ""); //Connect to datastore

cout << "Connected to datastore dbase: '" << dsICM->datastoreName() <<
    "', UserName '" << dsICM->userName() << "'.'" << endl;

dsICM->disconnect(); //Disconnect from datastore
delete(dsICM); //Destroy reference
```

Para obter a aplicação exemplo completa, consulte o exemplo SConnectDisconnectICM.cpp.

Ao estabelecer ligação a um servidor de conteúdos o utilizador terá de ter em atenção os requisitos de cada servidor de conteúdos; por exemplo, a palavra-passe para o ImagePlus para OS/390 pode não permitir mais do que oito caracteres de comprimento.

Estabelecer e anular a ligação a um servidor de conteúdos num cliente

Utilize o mesmo código que utilizou para estabelecer uma ligação para aceder a um servidor de conteúdos a partir de uma aplicação cliente.

Para isso, basta substituir `import com.ibm.mm.sdk.server.*;` por `import com.ibm.mm.sdk.client.*;`. A aplicação cliente terá de processar erros de comunicação que ocorram.

Definir e obter opções do servidor de conteúdos

Pode aceder ou definir opções de processamento num servidor de conteúdos utilizando os métodos incluídos no `DKDatastorexx`.

O exemplo que se segue mostra como definir e obter a opção de estabelecimento de uma sessão administrativa num servidor de bibliotecas do DB2 Content Manager. Consulte a Application Programming Reference para obter a lista de opções e as respectivas descrições.

Neste exemplo de definição e obtenção de uma opção do servidor de conteúdos no Content Manager, a colocação em memória cache está desactivada. No caso de servidores de conteúdos que suportam esta opção, recomenda-se que seja utilizada a predefinição (ON).

Importante: Já se encontra criado um objecto de `DKDatastoreICM` válido numa variável denominada `dsICM`.

Exemplo: Java

```
dsICM.setOption(DKConstant.DK_CM_OPT_CACHE,
    new Integer(DKConstant.DK_CM_FALSE));
Object val = dsICM.getOption(DKConstant.DK_CM_OPT_CACHE);
```

Exemplo: C++

```
DKAny inVal = DK_CM_FALSE
DKAny outVal;
dsICM->setOption(DK_CM_OPT_CACHE,inVal);
dsICM->getOption(DK_CM_OPT_CACHE,outVal);
```

Ao obter uma opção de servidor de conteúdos, `output_option` normalmente é um número inteiro, mas o utilizador pode convertê-lo para um objecto.

Enumerar servidores de conteúdos

O `DKDatastorexx` fornece um método para enumerar os servidores aos quais pode estabelecer ligação.

A lista de servidores é devolvida numa `DKSequentialCollection` de objectos de `DKServerDefxx` (em que `xx` identifica o servidor de conteúdos específico).

Após a obtenção de um objecto de `DKServerDefxx`, pode obter o nome do servidor e o tipo de servidor e utilizar o nome do servidor para estabelecer ligação ao mesmo.

O exemplo seguinte enumera os servidores aos quais está configurado para estabelecer ligação.

Exemplo: Java

```
DKDatastoreICM dsICM = new DKDatastoreICM(); // Create a datastore object
dkCollection coll = dsICM.listDataSources(); // Obtain data source list
dkIterator iter = coll.createIterator(); // Create an iterator
while(iter.more()){ // While there are more
    DKServerDefICM srvrDef = (DKServerDefICM) iter.next();
    System.out.println("Found server '"+srvrDef.getName()+"");
}
```

Exemplo: C++

```
DKDatastoreICM* dsICM = new DKDatastoreICM(); // Create a datastore object
dkCollection* coll = (dkCollection*)dsICM->listDataSources(); // Obtain list
dkIterator* iter = coll->createIterator(); // Create an iterator
while(iter->more()){ // While there are more
    DKServerDefICM* srvrDef = (DKServerDefICM*) iter->next()->value();
    cout << "Found server '" << srvrDef->getName() << "' << endl;
    delete(srvrDef); // Free memory
}
delete(iter); // Free memory
delete(coll);
delete(dsICM);
```

Enumerar as entidades e os atributos de um servidor de conteúdos

O DKDatastorexx fornece métodos para enumerar as entidades e os respectivos atributos para um servidor de conteúdos.

Cada nome de atributo faz parte de um espaço de nome. O espaço de nome predefinido é utilizado para todos os atributos em que não esteja especificado um espaço de nome.

A lista de entidades é devolvida num objecto DKSequentialCollection de objectos dkEntityDef. Os atributos para uma entidade são devolvidos num objecto DKSequentialCollection de objectos dkAttrDef. Após a obtenção de um objecto dkAttrDef, pode obter informações sobre o atributo, tais como o respectivo nome e tipo, e utilizar as informações para criar uma consulta.

Para obter mais informações sobre estes dois métodos, consulte Application Programming Reference.

Exemplo: Java

O exemplo seguinte demonstra como obter a lista de tipos de artigos, bem como a lista de atributos de um servidor DB2 Content Manager.

```
. . .
try {
    DKSequentialCollection pCol = null;
    dkIterator pIter = null;
    DKSequentialCollection pCol2 = null;
    dkIterator pIter2 = null;
    DKServerDefICM pSV = null;
    String strServerName = null;
    String strItemType = null;
    DKComponentTypeDefICM itemTypeDef = null;
    DKAttrDefICM attrDef = null;
    DKDatastoreDefICM dsDefICM = null;
    int i = 0;
    int j = 0;
    // ----- Create the datastore and connect (assumes the
    // parameters for the connection are previously set)
```



```

DKDatastoreICM dsICM = new DKDatastoreICM();
dsICM.connect(db,userId,pw,"");
// ----- List the item types
pCol = (DKSequentialCollection) dsICM.listEntities();
pIter = pCol.createIterator();
i = 0;
while (pIter.more() == true)
{
    i++;
    itemTypeDef = (DKComponentTypeDefICM)pIter.next();
    strItemType = itemTypeDef.getName();
    System.out.println("item type name [" + i + "] - " + strItemType);
    System.out.println("    type " + itemTypeDef.getType());
    System.out.println("    itemTypeId " + itemTypeDef.getId());
    System.out.println("    compID " + itemTypeDef.getComponentTypeId());
    //continued . . .
}
// ----- List the attributes
pCol2 = (DKSequentialCollection) dsICM.listEntityAttrs(strItemType);
pIter2 = pCol2.createIterator();
j = 0;
while (pIter2.more() == true)
{
    j++;
    attrDef = (DKAttrDefICM)pIter2.next();
    System.out.println("Attr name [" + j + "] - " + attrDef.getName());
    System.out.println("    datastoreType " + attrDef.datastoreType());
    System.out.println("    attributeOf " + attrDef.getEntityName());
    System.out.println("    type " + attrDef.getType());
    System.out.println("    size " + attrDef.getSize());
    System.out.println("    id " + attrDef.getId());
    System.out.println("    nullable " + attrDef.isNullable());
    System.out.println("    precision " + attrDef.getPrecision());
    System.out.println("    scale " + attrDef.getScale());
    System.out.println("    stringType " + attrDef.getStringType());
    System.out.println("    sequenceNo " + attrDef.getSequenceNo());
    System.out.println("    userFlag " + attrDef.getUserFlag());
}
}
dsICM.disconnect();
}
catch(DKException exc)
{
}
// ----- Handle the exceptions

```

Um exemplo completo, `SItemTypeRetrievalICM`, inclui como listar definições de tipos de artigo. Outro exemplo completo, `SAttributeDefinitionRetrievalICM`, inclui como listar definições de atributos. Ambos os exemplos estão disponíveis no directório `samples`.

Exemplo: C++

O exemplo seguinte de C++ mostra como obter a lista de classes e atributos de índice a partir de um servidor do DB2 Content Manager:

```

// Get a collection containing all Item Type Definitions.
DKSequentialCollection* itemTypeColl = (DKSequentialCollection*)
    dsICM->listEntities();
// Accessing each and printing the name & description.
cout << "\nItem Type Names in System:
    (" << itemTypeColl->cardinality() << ')' << endl;
// Create an iterator to iterate through the collection
dkIterator* iter = itemTypeColl->createIterator();
// while there are still items in the list, continue
while(iter->more()){
    DKItemTypeDefICM* itemType = (DKItemTypeDefICM*) iter->next()->value();
}

```

```

        cout << " - " << itemType->getName() << ": " <<
            itemType->getDescription() << endl;
        delete(itemType); // Free Memory

    cout << endl;
    delete(iter);
    delete(itemTypeColl);

    ...
    pCol2->apply(deleteDKAttrDefICM);
    delete pCol2;
    ...

```

Para obter mais informações, consulte o exemplo `SItemTypeRetrievalICM`, que inclui indicações sobre como enumerar definições de tipo de artigo. O `SAttributeDefinitionRetrievalICM` inclui informações sobre como enumerar definições de atributo. Os exemplos encontram-se disponíveis no directório `samples`.

Trabalhar com objectos de dados dinâmicos (DDOs)

Utilize a classe de `DKDDO` para objectos de dados dinâmicos (DDOs) nas aplicações do IBM DB2 Information Integrator for Content.

Esta secção descreve como utilizar um DDO e contém exemplos que ajudam o utilizador a compreender como:

1. Associe um `DKDDO` a um servidor de conteúdos.
2. Crie um `DKDDO`.
3. Crie Identificadores Persistentes (PIDs) para atributos `DKDDO`.
4. Adicione atributos e definir propriedades de atributo.
5. Defina o `DKDDO` como pasta ou documento.
6. Defina e visualiza valores das propriedades de atributo.
7. Verifique as propriedades de `DKDDO`.
8. Verifique as propriedades dos atributos.
9. Visualize o conteúdo de `DKDDO`.
10. Elimine o `DKDDO`.

Um objecto de `DKDDO` representa um artigo, que, por exemplo, pode ser um documento ou uma pasta do DB2 Content Manager ou um objecto definido pelo utilizador. Um objecto de `DKDDO` contém atributos. Cada atributo tem um nome, valor e propriedades. Cada atributo é identificado por um ID de dados. Os atributos são numerados consecutivamente, começando pelo 1; o número do atributo é o ID dos dados.

Uma vez que o nome, valor e propriedade de um atributo podem variar, o `DKDDO` fornece mecanismos flexíveis para representar dados originários de uma variedade de servidores de conteúdos e formatos. Por exemplo, artigos de diferentes tipos de artigos em DB2 Content Manager ou linhas de tabelas diferentes numa base de dados relacional. O próprio `DKDDO` pode ter propriedades que se aplicam a todo o `DKDDO`, em vez de apenas a um determinado atributo.

Associe um `DKDDO` a um servidor de conteúdos antes de invocar os métodos `add`, `retrieve`, `update` e `delete` para colocar os respectivos atributos no servidor de conteúdos ou recuperá-los. Defina o servidor de conteúdos como parâmetro que irá criar o objecto de `DKDDO` ou invocando o método `setDatastore`.

Todos os DKDDOs possuem um identificador de objectos persistentes (PID), que contém informações para localização dos atributos no servidor de conteúdos.

Criar um DKDDO

O DKDDO possui vários construtores. Pode criar um DKDDO invocando o respectivo construtor sem quaisquer parâmetros.

Exemplo: Java

```
DKDDO ddo = new DKDDO();
```

Exemplo: C++

```
DKDDO ddo;
```

Este DDO ddo deve aumentar dinamicamente para acomodar mais atributos. Para obter um construtor mais eficiente, transmita o número exacto de atributos que pretende (por exemplo, dez).

Para obter um construtor mais eficiente, transmita o número exacto de atributos que pretende (por exemplo, dez).

Exemplo: Java

```
DKDDO ddo = new DKDDO(10);
```

Exemplo: C++

```
DKDDO ddo = new DKDDO((short)10);
```

Importante: Os utilizadores do DB2 Content Manager V8 (API DKDatastoreICM), bem como os utilizadores da API DKDatastoreOD deverão sempre criar DDOs utilizando os métodos DKDatastoreXX::createDDO() em vez de utilizar construtores do DKDDO directamente. O método createDDO() cria a subclasse apropriada, inicia estruturas de metadados preenche predefinições, etc.

O exemplo seguinte cria um DKDDO transmitindo o servidor de conteúdos e o tipo de objectos ao Content Manager Versão 8:

Exemplo: Java

```
// Given a connected DKDatastoreICM object named "dsICM"  
// Create a DDO representing an instance of an item type  
DKDDO ddo=dsICM.createDDO("ICMSAMPLE",DKConstant.DK_CM_DOCUMENT);
```

Para obter mais informações relativas à criação de DDO, consulte o exemplo SItemCreationICM.

Exemplo: C++

```
// Given a connected DKDatastoreICM object named "dsICM"  
// Create a DDO representing an instance of an item type  
DKDDO* ddo = dsICM->createDDO("ICMSAMPLE", DK_CM_DOCUMENT);
```

Para obter mais informações relativas à criação de DDO, consulte o exemplo SItemCreationICM.

Adicionar propriedades a um DDO

Ao criar um objecto de DKDDO para representar um DDO, é necessário especificar a respectiva propriedade de tipo de artigo (um documento, pasta ou artigo).

Podemos transmitir esta propriedade como uma das opções do método `DKDatastoreXX.createDDO(itemTypeName,itemPropertyType/SemanticType)`.
`DKDatastoreICM.createDDO(itemTypeName,itemPropertyType/SemanticType)`

Criar um identificador persistente (PID)

Cada DDO terá de ter um identificador persistente (PID). O PID contém informações sobre um determinado servidor de conteúdos, como, por exemplo, o nome do arquivo de dados, o tipo de arquivo de dados e o tipo de objecto.

O PID identifica a localização de dados persistentes do DDO. Por exemplo, em servidores de conteúdos de versões anteriores do DB2 Content Manager, o PID continha o ID do artigo. O ID do artigo constitui um dos parâmetros mais importantes para as funções de obtenção, actualização e eliminação em servidores de conteúdos de versões anteriores do DB2 Content Manager. No DB2 Content Manager V8, o PID possui outras informações necessárias, tais como o ID de artigo e o ID de componente.

O método `createDDO()` do arquivo de dados do DB2 Content Manager V8 criará um objecto PID para o objecto DDO criado. Durante a operação de adição no DB2 Content Manager V8, o arquivo de dados preenche o objecto PID com informações como, por exemplo, o ID de artigo.

O exemplo que se segue cria um DDO para obtenção de um artigo conhecido.

Exemplo: Java

```
// Given a connected DKDatastoreICM object named "dsICM"
// Create a new DDO
DKDDO ddo = dsICM.createDDO("book",DKConstant.DK_CM_DOCUMENT);
// PID automatically created by the function above.
ddo.add(); // Adicione o novo artigo ao arquivo de dados.
// PID Completed by the System
DKPidICM pid = (DKPidICM) ddo.getPidObject();
```

Exemplo: C++

```
// Given a connected DKDatastoreICM object named "dsICM"
// Create a new DDO
DKDDO* ddo = dsICM->createDDO("book",DK_CM_DOCUMENT);
// PID automatically created by the function above.
ddo->add(); // Adicione o novo artigo ao arquivo de dados.
// PID Completed by the System
DKPidICM* pid = (DKPidICM*) ddo->getPidObject();
```

O conector do DB2 Content Manager V8 possui uma classe de PID denominada `DKPidICM`, que constitui uma subclasse de `DKPid`. O objecto `DKPidICM` é utilizado pelo `DKDDO` e pelas subclasses de `dkResource`.

Trabalhar com artigos e propriedades de dados

O `DKDDO` fornece métodos para adicionar atributos e propriedades de atributos a um objecto de `DKDDO`.

Suponha que um tipo de artigo `NameOfItemType` tem os atributos `Name` de tipo `integer` e está definido como anulável no repositório do CM V8. O utilizador cria um objecto de `DKDDO` para processar um artigo dessa entidade e pretende adicionar dois artigos de dados ao `DKDDO`.

O exemplo seguinte cria um artigo, define propriedades de atributos e guarda-as no servidor de conteúdos persistente do Content Manager.

Importante: Assume que existe um DKDatstoreICM ligado na variável *dsICM*. O tipo de artigo definido pelo utilizador *S_withChild* terá de ser definido com *varchar* *S_varchar*, *long integer* *S_integer*, *short integer* *S_short* e *time* *S_time*, conforme indicado no exemplo formativo da API *SItemCreationICM*.

Exemplo: Java

```
// Create a new item in memory
DKDDO ddo = dsICM.createDDO("S_withChild", DKConstant.DK_CM_DOCUMENT);

// Set attributes (Additional attributes set in SItemCreationICM sample)
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,"S_varchar"),
    "abcdefg");
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,"S_integer"),
    new Integer("123"));
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,"S_short"),
    new Short("5"));
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,"S_time"),
    java.sql.Time.valueOf("10:00:00"));

// Add to datastore
ddo.add();
```

Este exemplo foi retirado do exemplo *SItemCreationICM*.

Exemplo: C++

```
// Create a new item in memory
DKDDO* ddo = dsICM->createDDO("S_withChild", DK_CM_DOCUMENT);

// Set attributes (Additional attributes set in SItemCreationICM sample)
// NOTE: Values below are automatically converted to type DKAny
ddo->setData(ddo->dataId(DK_CM_NAMESPACE_ATTR,"S_varchar"),
    DKString("this is a string value"));
ddo->setData(ddo->dataId(DK_CM_NAMESPACE_ATTR,"S_integer"),
    (long) 123);
ddo->setData(ddo->dataId(DK_CM_NAMESPACE_ATTR,"S_short"),
    (short) 5);
ddo->setData(ddo->dataId(DK_CM_NAMESPACE_ATTR,"S_time"),
    DKTime("10.00.00"));

// Add to datastore
ddo->add();
```

Este exemplo foi retirado do exemplo *SItemCreationICM*.

O utilizador deve definir o tipo da propriedade de um atributo; passível de ser convertido em nulo e outras propriedades são opcionais.

O exemplo seguinte acede a valores de atributos de um DDO.

Exemplo: Java

```
// Cast operations will enable access as subclass of Object type returned.
// NOTE: Additional attributes accessed in SItemRetrievalICM sample.

String attrVal1 = (String) ddo.getData(ddo.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,"S_varchar"));
Integer attrVal2 = (Integer) ddo.getData(ddo.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,"S_integer"));
```

```

Short  attrVal3 = (Short) ddo.getData(ddo.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,"S_short"));
Time   attrVal4 = (Time)  ddo.getData(ddo.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,"S_time"));

System.out.println("Attr 'S_varchar' value: "+attrVal1);
System.out.println("Attr 'S_integer' value: "+attrVal2);
System.out.println("Attr 'S_short'   value: "+attrVal3);
System.out.println("Attr 'S_time'    value: "+attrVal4);

```

Este exemplo foi retirado do exemplo SItemRetrievalICM.

Exemplo: C++

```

// Assignment and cast operations converts values from DKAny to each type.
// NOTE: Additional attributes accessed in SItemRetrievalICM sample.

DKString attrVal1 = ddo->getData(ddo->dataId(DK_CM_NAMESPACE_ATTR,
    DKString("S_varchar"))).toString();
long      attrVal2 = (long) ddo->getData(ddo->dataId(DK_CM_NAMESPACE_ATTR,
    DKString("S_integer")));
short     attrVal3 = (short) ddo->getData(ddo->dataId(DK_CM_NAMESPACE_ATTR,
    DKString("S_short")));
DKTimestamp attrVal4 = (DKTimestamp) ddo->getData(ddo->dataId(
    DK_CM_NAMESPACE_ATTR, DKString("S_time")));

cout << "Attr 'S_varchar' value: " << attrVal1 << endl;
cout << "Attr 'S_integer' value: " << attrVal2 << endl;
cout << "Attr 'S_short'   value: " << attrVal3 << endl;
cout << "Attr 'S_time'    value: " << attrVal4 << endl;

```

Este exemplo foi retirado do exemplo SItemRetrievalICM.

Obter o DKDDO e as propriedades de atributo

Ao processar um DKDDO, é necessário saber primeiro qual o respectivo tipo: documento, pasta ou artigo.

Ao processar um DKDDO, é necessário saber primeiro qual o respectivo tipo: documento, pasta ou artigo. O código exemplo que se segue demonstra como determinar os tipos de DDO.

Exemplo: Java

```

short prop_id = ddo.propertyId(DK_CM_PROPERTY_ITEM_TYPE);
if (prop_id > 0) {
    short type = ((Short) ddo.getProperty(prop_id)).shortValue();
    switch(type) {
        case DK_CM_DOCUMENT:
            // --- process a document
            ....
            break;
        case DK_CM_FOLDER:
            // --- process a folder
        case DK_CM_ITEM:
            // --- Process an item in Content Manager
            ....
            break;
    }
}

```

Para obter mais informações relativas ao acesso de propriedades do tipo de artigo, consulte o Exemplo Formativo de API SItemRetrievalICM.

Exemplo: C++

```
unsigned short prop_id =
    ddo->propertyId(DK_CM_PROPERTY_ITEM_TYPE);
if (prop_id > 0) {
    unsigned short type = (unsigned short) ddo->getProperty(prop_id);
    switch(type) {
        case DK_CM_DOCUMENT:
            // process document
            ...
            break;
        case DK_CM_FOLDER:
            // process folder
            ...
            break;
    }
}
```

Para obter mais informações relativas ao acesso de propriedades do tipo de artigo, consulte o Exemplo Formativo de API SItemRetrievalICM.

Para obter propriedades de um atributo, o utilizador deve obter o data_id do atributo e, de seguida, poderá obter as propriedades.

Tanto o data_id como o property_id começam por 1. Se especificar 0, receberá uma exceção.

Exemplo: Java

```
data_id = ddo.dataId("Title"); // get data_id of Title
// ----- Obter número de propriedades para atributo
short number_of_data_prop = ddo.dataPropertyCount(data_id);
// ----- Apresentar todas as propriedades de dados pertencentes a este
// atributo com um ciclo; índice inicia em 1
for(short i = 1; i <= number_of_data_prop; i++) {
    System.out.println(i + " Property Name = " +
        ddo.getDataPropertyName(data_id,i)
        + " value = " + ddo.getDataProperty(data_id,i));
}
```

Para obter a aplicação exemplo completa, consulte o exemplo SItemRetrievalICM.

Exemplo: C++

```
// get data_id of Title
data_id = ddo->dataId("Title");
// how many props does it have?
unsigned short number_of_data_prop = ddo->dataPropertyCount(data_id);
// displays all data properties belonging to this attribute
// notice that the loop index starts from 1, where
// 1 <= i <= number_of_data_prop
for (unsigned short i = 1; i <= number_of_data_prop; i++) {
    cout << i << " Property Name = " << ddo->
        getDataPropertyName(data_id, i) << " value = " << ddo->
        getDataProperty(data_id, i) << endl;
}
```

Para obter a aplicação exemplo completa, consulte o exemplo SItemRetrievalICM.

Apresentar a totalidade do DDO

Durante o desenvolvimento da aplicação, poderá ser necessário apresentar os conteúdos de um DKDDO para fins de depuração.

Exemplo: Java

```
short number_of_attribute = ddo.dataCount();
short number_of_prop      = ddo.propertyCount();
short number_of_data_prop;
// list DDO properties
for (short k = 1; k <= number_of_prop; k++) {
    System.out.println( k + " Property Name = " + ddo.getPropertyName(k) +
        ",\t value = " + ddo.getProperty(k));
}
// list data-items and their properties
for (short i = 1; i <= number_of_attribute; i++) {
    System.out.println( i + " Attr. Name = " + ddo.getDataName(i) +
        ",\t value = " +
        number_of_data_prop = ddo.dataPropertyCount(i);
    for (short j = 1; j <= number_of_data_prop; j++) {
        System.out.println( "\t" + j + " Data Prop. Name = " +
            ddo.getDataPropertyName(i,j) +
            ",\t value = " +
            ddo.getDataProperty(i,j));
    }
}
```

Para obter um exemplo completo de acesso e impressão de um DDO (e todos os subcomponentes), consulte `SItemRetrievalICM` e a sua função estática `printDDO()`.

Exemplo: C++

```
unsigned short number_of_attribute = ddo->dataCount();
unsigned short number_of_prop;
unsigned short number_of_data_prop;
// list DDO properties
for (short k = 1; k <= number_of_prop; k++) {
    cout << k << " Property Name = " << ddo->getPropertyName(k) <<
        ",\t value = " << ddo->getProperty(k) << endl;
}
// list data-items and their properties
for (unsigned short i = 1; i <= number_of_attribute; i++) {
    cout << i << " Attr. Name = " << ddo->getDataName(i) <<
        << ",\t value = " << ddo->getData(i) << endl;
    number_of_data_prop = ddo->dataPropertyCount(i);
    for (unsigned short j = 1; j <= number_of_data_prop; j++) {
        cout << "\t" << j << " Data Prop. Name = "
            << ddo->getDataPropertyName(i, j)
            << ",\t value = " << ddo->getDataProperty(i, j)
            << endl;
    }
}
```

Para obter um exemplo completo de acesso e impressão de um DDO (e todos os subcomponentes), consulte `SItemRetrievalICM` e a sua função estática `printDDO()`.

Eliminar um DDO (Apenas para C++)

Um DKDDO tem duas representações: a da memória e a cópia persistente. Para eliminar o DKDDO da memória, invoque o respectivo destruidor (`delete myddo`);. Tenha em atenção que esta acção não altera a cópia persistente no servidor de conteúdos.

Para eliminar a cópia persistente no servidor de conteúdos, utilize a função `dkddo:del()` (`delete dkddo`). Este procedimento não afecta a representação do DKDDO na memória. O destruidor elimina todo o objecto, todas as estruturas de metadados internos e o conteúdo da memória e todos os artigos e estruturas das quais o DDO é proprietário.

Trabalhar com objectos de dados expandidos (XDOs)

Um XDO representa um componente que pode armazenar conteúdos de recurso como, por exemplo, um artigo de recurso ou uma parte de documento.

Os artigos de recurso (XDOs) expandem artigos que não são de recurso (DDOs). A criação de artigos de recurso é idêntica aos artigos comuns. Dependendo do tipo de artigo de recurso, o XDO pode ser ainda mais expandido.

DB2 Content Manager V8: O DB2 Content Manager V8 requer que os XDOs pertençam à subclasse correcta de DKLobICM com base na classificação e subclasse associada de XDO do tipo de artigo. É sempre necessário criar DKDDOs utilizando os métodos createDDO() do objecto de arquivo de dados (DKDatastoreICM::createDDO()). Este procedimento permite ao sistema criar automaticamente uma instância da subclasse apropriada de DKDDO ou DKLobICM, iniciar outras informações importantes e configurar adequadamente uma maior funcionalidade, como, por exemplo, recursos, modelo de documentos e pastas. Os DKDDOs de classificações de tipos de artigo de parte de recurso ou documento (devolvidos dos métodos DKDatastoreICM.createDDO()) podem ser convertidos na subclasse real, dependendo da classificação de tipo de artigo e da classe de XDO. Para obter mais informações sobre a criação de artigos em geral e a função DKDatastoreICM.createDDO(), consulte os exemplos de API SItemCreationICM e SResourceItemCreationICM.

Exemplo: Java

Hierarquia de Classes

Tipo	DDO	XDO	Extensão
-----	-----	-----	-----
Lob	DKDDO	-> DKLobICM	
Texto	DKDDO	-> DKLobICM	-> DKTextICM
Imagem	DKDDO	-> DKLobICM	-> DKImageICM
Cadeia	DKDDO	-> DKLobICM	-> DKStreamICM

Utilizar um identificador persistente (PID) de XDO

Um XDO necessita de um PID para armazenar os dados de forma persistente. Para utilizar um XDO de forma a localizar e armazenar dados, é necessários fornecer um PID para o DKBlobxx, utilizando um DKPidXDOxx.

As bases de dados relacionais requerem a tabela, coluna e cadeia de predicado de dados para localizar os dados persistentes num servidor de conteúdos. Para bases de dados relacionais (RDB), o nome da tabela, o nome da coluna e o predicado de dados são requeridos para o DKPidXDOxx.

No Conector do DB2 Content Manager V8, utilize DKPidICM para representar o PID de um objecto de dkResource (que consiste num XDO normalmente instanciado como DKLobICM ou subclasse).

Não se esqueça: Utilize os métodos DKDatastoreICM::createDDO() para construir DDOs activados através de recursos, o que cria também o PID.

Sugestões de programação de Java

Para o Content Manager V8 e posteriores, um XDO é um objecto de dkResource. Utilize o DKPidICM para representar o PID do objecto de recurso.

No caso de versões anteriores do Content Manager, DB2 Content Manager for AS/400 e IP 390, o utilizador identifica um XDO através da combinação do ID de

artigo, do ID de parte e do RepType. Para RDB, a chave para a identificação de um XDO é a combinação da cadeia de tabela, coluna e predicado de dados. Para processar um XDO autónomo, é necessário fornecer o ID do artigo e o ID da parte. O RepType é opcional, uma vez que o sistema lhe fornece um valor predefinido.

Utilize o método add do DKBlobxx para adicionar o conteúdo actual a um servidor de conteúdos. Poderá obter o valor do ID da parte depois de adicionar caso pretenda efectuar mais alguma operação com esse objecto posteriormente.

Utilize o método getPidObject() no dkXDO para obter o objecto de DKPid.

Poderá utilizar a seguinte instrução depois de add para obter o ID da parte atribuído ao sistema:

Aviso: Em versões anteriores do Content Manager, é necessário um ID de parte válido para adicionar uma parte a ser indexada pelo gestor de pesquisas (não é possível definir o ID de parte como 0).

Nesta edição, foram modificados dois métodos em dkXDO: DKPid dkXDO.getPid() está obsoleto e foi substituído por getPidObject. DKPid dkXDO.getPidObject(). Estes métodos devolviam um DKPidXDO, mas actualmente devolvem um objecto de DKPid.

Exemplo: Java

```
int partID = ((DKPidXDOICM)(axdo.getPidObject())).getPartId();
```

Programar um XDO como parte de DDO

Um XDO representa um objecto de parte único, se possuir um DDO a representar um documento, que consiste numa recolha de objectos de conteúdo de recurso. Pode manipular o XDO enquanto componente do DDO ou como um objecto autónomo.

Quando aceder ao XDO como parte do DDO, o DDO fornece o ID do artigo. Ao utilizar o XDO como um objecto autónomo, terá de utilizar o ID do artigo existente para o XDO.

O exemplo seguinte cria um documento e adiciona partes de documento em Content Manager.

Importante: O tipo de artigo definido pelo utilizador, S_docModel, terá de ser definido no sistema, classificado como Modelo de Documentos, e suportar tipos de parte ICMBASE e ICMBASETEXT, conforme definido no exemplo formativo da API SItemTypeCreationICM

Exemplo: Java

```
// Create a document
DKDDO ddoDocument = dsICM.createDDO("S_docModel",
DKConstant.DK_CM_DOCUMENT);
// Create parts
DKLobICM base = (DKLobICM) dsICM.createDDO("ICMBASE",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASE);
DKTextICM baseText1 = (DKTextICM) dsICM.createDDO("ICMBASETEXT",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASETEXT);
DKTextICM baseText2 = (DKTextICM) dsICM.createDDO("ICMBASETEXT",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASETEXT);

// Set parts' MIME type (SResourceItemMimeTypesICM.txt sample)
```

```

base.setMimeType("application/msword");
baseText1.setMimeType("text/plain");
baseText2.setMimeType("text/plain");

// Load content into parts (SResourceItemCreationICM sample)
base.setContentFromClientFile("SResourceItemICM_Document1.doc");
// Load file
baseText1.setContentFromClientFile("SResourceItemICM_Text1.txt");
// into memory
baseText2.setContentFromClientFile("SResourceItemICM_Text2.txt");

// Access the DKParts attribute
DKParts dkParts = (DKParts) ddoDocument.getData(ddoDocument.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,DKConstant.DK_CM_DKPARTS));

// Add parts to document
dkParts.addElement(base);
dkParts.addElement(baseText1);
dkParts.addElement(baseText2);

// Add new document to persistent datastore
ddoDocument.add();

```

Para obter a aplicação exemplo completa, consulte o exemplo SDocModelItemICM.java. SResourceItemCreationICM apresenta mais exemplos da utilização de XDO.

Exemplo: C++

```

// Create a document
DKDDO* ddoDocument = dsICM->createDDO("S_docModel", DK_CM_DOCUMENT);
// Create Parts
DKLobICM* base = (DKLobICM*) dsICM->createDDO("ICMBASE",
    DK_ICM_SEMANTIC_TYPE_BASE);
DKTextICM* baseText1 = (DKTextICM*) dsICM->createDDO("ICMBASETEXT",
    DK_ICM_SEMANTIC_TYPE_BASSETEXT);
DKTextICM* baseText2 = (DKTextICM*) dsICM->createDDO("ICMBASETEXT",
    DK_ICM_SEMANTIC_TYPE_BASSETEXT);

// Set parts' MIME type (SResourceItemMimeTypesICM.txt sample)
base->setMimeType("application/msword");
baseText1->setMimeType("text/plain");
baseText2->setMimeType("text/plain");
// Load content into parts (SResourceItemCreationICM sample)
// Load the file into memory.
base->setContentFromClientFile("SResourceItemICM_Document1.doc");
baseText1->setContentFromClientFile("SResourceItemICM_Text1.txt");
baseText2->setContentFromClientFile("SResourceItemICM_Text2.txt");
// Access the DKParts attribute
DKParts* dkParts = (DKParts*)(dkCollection*) ddoDocument->getData(
    ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS));

// Add parts to document
dkParts->addElement((dkDataObjectBase*)(DKDDO*)base);
dkParts->addElement((dkDataObjectBase*)(DKDDO*)baseText1);
dkParts->addElement((dkDataObjectBase*)(DKDDO*)baseText2);
// Add new document to persistent datastore
ddoDocument->add();

```

Para obter a aplicação exemplo completa, consulte o exemplo SDocModelItemICM.java. SResourceItemCreationICM apresenta mais exemplos da utilização de XDO.

Programar um XDO autónomo

Todos os exemplos que se seguem são específicos do DB2 Content Manager Versão 8. Para obter exemplos para versões anteriores do Content Manager e outros servidores de conteúdos, consulte a secção *Representar itens utilizando DDO*, se trabalhar com outros servidores de conteúdos, e consulte os programas exemplo no directório samples.

Adicionar um XDO a partir da memória tampão

Este exemplo mostra como adicionar um XDO a partir de uma memória tampão no Content Manager. Cria um XDO, carrega conteúdo na memória e armazena conteúdo persistente na memória.

Importante: O tipo de artigo definido pelo utilizador S_lob de recurso de classificação terá de ser definido no sistema, tal como definido pelo exemplo formativo da API SItemTypeCreationICM. Além disso, as definições do gestor de recursos e da recolha SMS devem ser configuradas e estabelecidas como predefinições do tipo de artigo ou do utilizador, tal como os exemplos SResourceMgrDefCreationICM, SSMSCollectionDefCreationICM, SResourceMgrDefSetDefaultICM e SSMSCollectionDefSetDefaultICM demonstram.

Exemplo: Java

```
// Create an empty resource object
DKLobICM lob = (DKLobICM) dsICM.createDDO("S_lob",
DKConstant.DK_CM_DOCUMENT);
// Set the MIME type (SResourceItemMimeTypesICM.txt sample)
lob.setMimeType("application/msword");
// Load content into item's local memory
lob.setContentFromClientFile
("SResourceItemICM_Document1.doc");
// Add to datastore with content already in memory
lob.add();
```

Para obter a aplicação exemplo completa, consulte o exemplo SResourceItemCreationICM.

Exemplo: C++

```
// Create an empty resource object
DKLobICM* lob = (DKLobICM*)
dsICM->createDDO("S_lob", DK_CM_DOCUMENT);
// Set the MIME type (SResourceItemMimeTypesICM.txt sample)
lob->setMimeType("application/msword");
// Carregue o conteúdo para a memória local do artigo
lob->setContentFromClientFile("SResourceItemICM_Document1.doc");
("SResourceItemICM_Document1.doc");
// Add to datastore With content already in memory
lob->add();
```

Para obter a aplicação exemplo completa, consulte o exemplo SResourceItemCreationICM.

Adicionar um XDO a partir de um ficheiro

O exemplo seguinte adiciona um XDO ao servidor de conteúdos (armazenar o conteúdo directamente do ficheiro) do Content Manager.

Ao armazenar objectos, pode especificar uma recolha de gestor de recursos alternativa na aplicação personalizada. Para obter mais informações, consulte DKLobICM no manual *Application Programming Reference*.

Importante: O tipo de artigo definido pelo utilizador S_lob de recurso de classificação terá de ser definido no sistema, tal como definido pelo exemplo formativo da API SItemCreationICM. Além disso, as definições do gestor de recursos e da recolha SMS devem ser configuradas e estabelecidas como predefinições do tipo de artigo ou do utilizador, tal como os exemplos SResourceMgrDefCreationICM, SSMSCollectionDefCreationICM, SResourceMgrDefSetDefaultICM e SSMSCollectionDefSetDefaultICM demonstram.

Exemplo: Java

```
// Create an empty resource object
DKTextICM text = (DKTextICM) dsICM.createDDO("S_text", DKConstant.DK_CM_ITEM);

// Set the MIME type (SResourceItemMimeTypesICM.txt sample)
text.setMimeType("text/plain");

// Store content directly from a file
text.add("SResourceItemICM_Text1.txt");
```

Para obter a aplicação exemplo completa, consulte o exemplo SResourceItemCreationICM.

Exemplo: C++

```
// Create an empty resource object
DKTextICM* text = (DKTextICM*) dsICM->createDDO("S_text",
    DK_CM_ITEM);
// Set the MIME type (SResourceItemMimeTypesICM.txt sample)
text->setMimeType("text/plain");

// Store content directly from a file
text->add("SResourceItemICM_Text1.txt");
```

Para obter a aplicação exemplo completa, consulte o exemplo SResourceItemCreationICM.

Adicionar um objecto de anotação a um XDO

O exemplo seguinte adiciona uma parte de anotação a um documento no Content Manager.

Importante: O tipo de artigo definido pelo utilizador, S_docModel, terá de ser definido no sistema, classificado como Modelo de Documentos, e suportar o tipo de parte ICMANNOTATION, conforme definido no exemplo SItemCreationICM. Parta do princípio que lhe foi atribuída uma instância de um documento já armazenado persistentemente na variável *ddoDocument*. Além disso, presuma que lhe é atribuído um objecto de DKDatastoreICM ligado na variável de *dsICM*.

Exemplo: Java

```
// Check out / lock the item for update
dsICM.checkOut(ddoDocument);

// Create annotation part
DKLobICM annot = (DKLobICM) dsICM.createDDO("ICMANNOATATION",
    DKConstantICM.DK_ICM_SEMANTIC_TYPE_ANNOTATION);
```

```
// Set annotation MIME type (SResourceItemMimeTypesICM.txt sample)
annot.setMimeType("image/bmp");

// Load content into parts (SResourceItemCreationICM sample)
annot.setContentFromClientFile("myAnnotation.bmp");

// Access the DKParts attribute
DKParts dkParts = (DKParts) ddoDocument.getData(ddoDocument.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,DKConstant.DK_CM_DKPARTS));

// Add parts to document
dkParts.addElement(annot);

// Save the changes to the persistent datastore
ddoDocument.update();

// Check in / unlock the item after update
dsICM.checkIn(ddoDocument);
```

Para obter a aplicação exemplo completa, consulte o exemplo SDocModelItemICM.

Exemplo: C++

```
// Check out / lock the item for update
dsICM->checkOut(ddoDocument);

// Create annotation part
DKLobICM* annot = (DKLobICM*) dsICM->createDDO("ICMANNOTATION",
    DK_ICM_SEMANTIC_TYPE_ANNOTATION);

// Set annotation MIME type (SResourceItemMimeTypesICM.txt sample)
annot->setMimeType("image/bmp");

// Load content into parts (SResourceItemCreationICM sample)
annot->setContentFromClientFile("myAnnotation.bmp");

// Access the DKParts attribute
DKParts* dkParts = (DKParts*)(dkCollection*) ddoDocument->getData(
    ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS));

// Add parts to document
dkParts->addElement((dkDataObjectBase*)(DKDDO*)annot);

// Save the changes to the persistent datastore
ddoDocument->update();

// Check in / unlock the item after update
dsICM->checkIn(ddoDocument);
```

Para obter a aplicação exemplo completa, consulte o exemplo SDocModelItemICM.

Exemplos de trabalho com um XDO

O exemplo seguinte ilustra a utilização de um XDO autónomo.

Obter, actualizar e eliminar um XDO

O exemplo seguinte obtém, actualiza e elimina um XDO no Content Manager.

Importante: O tipo de artigo definido pelo utilizador S_text de recurso de classificação tem de ser definido no sistema conforme indicado pelo exemplo formativo de API SItemTypeCreationICM. Além disso, as definições do gestor de recursos e da recolha SMS devem ser configuradas e estabelecidas como predefinições do tipo de artigo ou

do utilizador, tal como os exemplos SResourceMgrDefCreationICM, SSMSCollectionDefCreationICM, SResourceMgrDefSetDefaultICM e SSMSCollectionDefSetDefaultICM demonstram. Parta ainda do princípio que lhe é atribuída uma cadeia de PID na variável *pidString* para um artigo de recurso que já existe no servidor de conteúdos.

Exemplo: Java

```
// Given: String pidString

// Re-create Blank DDOs for Existing Item (SItemRetrievalICM sample)

DKLobICM lob = (DKLobICM) dsICM.createDDOFromPID(pidString);

// Retrieve the item with the resource content
DKRetrieveOptionsICM retrOpts = DKRetrieveOptionsICM::createInstance(dsICM);
retrOpts.baseAttributes(true);
retrOpts.resourceContent(true); // Load into memory
lob.retrieve(retrOpts.dkNVPair());

// Check out / lock the item for update (SItemUpdateICM sample)
dsICM.checkOut(lob);

// Set the new MIME type (SResourceItemMimeTypesICM.txt sample)
lob.setMimeType("application/msword");

// Update datastore with new content
lob.update("SResourceItemICM_Document2.doc");

// Check in / unlock the item after update
dsICM.checkIn(lob);

// Delete item
lob.del();
```

Este exemplo de código foi retirado dos exemplos SResourceItemRetrievalICM, SResourceItemUpdateICM e SResourceItemDeletionICM.

Exemplo: C++

```
// Given: DKString pidString

// Re-create Blank DDOs for Existing Item (SItemRetrievalICM sample)
DKLobICM* lob = (DKLobICM*) dsICM->createDDOFromPID(pidString);

// Retrieve the item with the resource content
DKRetrieveOptionsICM* retrOpts = DKRetrieveOptionsICM::createInstance(dsICM);
retrOpts->baseAttributes(TRUE);
retrOpts->resourceContent(TRUE); // Load into memory
lob->retrieve(retrOpts->dkNVPair(), retrOpts->dkNVPairLength());

// Check out / lock the item for update (SItemUpdateICM sample)
dsICM->checkOut(lob);

// Set the new MIME type (SResourceItemMimeTypesICM.txt sample)
lob->setMimeType("application/msword");

// Update datastore with new content
lob->update("SResourceItemICM_Document2.doc");

// Check in / unlock the item after update
dsICM->checkIn(lob);

// Delete item
```



```
lob->del();

// Free Memory
delete(lob);    delete(retrOpts);
```

Este exemplo de código foi retirado dos exemplos SResourceItemRetrievalICM, SResourceItemUpdateICM e SResourceItemDeletionICM.

Criar documentos e utilizar o atributo DKPARTS

O atributo DKPARTS num DDO representa a recolha de partes de um documento. O valor deste atributo é um objecto DKParts, que consiste numa recolha de objectos DKPart. Os objectos DKPart são itens de tipos de itens classificados como parte de documento e incluem conteúdo de recurso.

Apenas para o DB2 Content Manager: Um documento é um artigo que pode ser armazenado, obtido e trocado entre sistemas do Content Manager e utilizadores como unidade separada. Um artigo ao qual é atribuído o tipo semântico *documento* deve conter informações que compõem um documento, o que não significa uma implementação de um modelo de documentos específico. Um artigo criado a partir de um tipo de artigo classificado documento (também conhecido como modelo de documentos) significa que o artigo irá conter partes de documento, uma implementação específica de um modelo de documentos facultada pelo Content Manager. Os tipos de artigo classificados documento podem criar artigos com o tipo semântico documento ou pasta. As partes de documento incluem vários tipos de conteúdo, por exemplo, texto, imagens e folhas de cálculo.

O exemplo seguinte cria um documento e adiciona partes de documento no Content Manager.

Importante: O tipo de artigo definido pelo utilizador, S_docModel, terá de ser definido no sistema, classificado como Document Model. Também suporta tipos de parte ICMBASE e ICMBASETEXT, tal como o exemplo formativo de API SItemTypeCreationICM demonstra.

Exemplo: Java

```
// Create a document
DKDDO ddoDocument = dsICM.createDDO("S_docModel",
DKConstant.DK_CM_DOCUMENT);
// Create parts
DKLobICM base = (DKLobICM) dsICM.createDDO("ICMBASE",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASE);
DKTextICM baseText1 = (DKTextICM) dsICM.createDDO("ICMBASETEXT",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASETEXT);
DKTextICM baseText2 = (DKTextICM) dsICM.createDDO("ICMBASETEXT",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASETEXT);

//Set parts' MIME type (SResourceItemMimeTypesICM.txt sample)
base.setMimeType("application/msword");
baseText1.setMimeType("text/plain");
baseText2.setMimeType("text/plain");

// Load content into parts (SResourceItemCreationICM sample)
base.setContentFromClientFile("SResourceItemICM_Document1.doc");
// Load file
baseText1.setContentFromClientFile("SResourceItemICM_Text1.txt");
// into memory
baseText2.setContentFromClientFile("SResourceItemICM_Text2.txt");

// Access the DKParts attribute
```



```

DKParts dkParts = (DKParts) ddoDocument.getData(ddoDocument.dataId(
    DKConstant.DK_CM_NAMESPACE_ATTR,DKConstant.DK_CM_DKPARTS));

// Add parts to document
dkParts.addElement(base);
dkParts.addElement(baseText1);
dkParts.addElement(baseText2);

// Add new document to persistent datastore
ddoDocument.add();

```

Para obter mais informações relativas à criação de documentos com partes, consulte o Exemplo Formativo de API SDocModelItemICM.

Exemplo: C++

```

// Create a document
DKDDO* ddoDocument = dsICM->createDDO("S_docModel", DK_CM_DOCUMENT);

// Crie Partes
DKLobICM* base = (DKLobICM*)
    dsICM->createDDO("ICMBASE",
        DK_ICM_SEMANTIC_TYPE_BASE);
DKTextICM* baseText1 = (DKTextICM*)
    dsICM->createDDO("ICMBASETEXT",
        DK_ICM_SEMANTIC_TYPE_BASETEXT);
DKTextICM* baseText2 = (DKTextICM*)
    dsICM->createDDO("ICMBASETEXT",
        DK_ICM_SEMANTIC_TYPE_BASETEXT);

// Set parts' MIME type (SResourceItemMimeTypesICM.txt sample)
base->setMimeType("application/msword");
baseText1->setMimeType("text/plain");
baseText2->setMimeType("text/plain");

// Load content into parts
// (SResourceItemCreationICM sample)
base->setContentFromClientFile("SResourceItemICM_Document1.doc");
// Load the file into memory.
baseText1->setContentFromClientFile("SResourceItemICM_Text1.txt");
baseText2->setContentFromClientFile("SResourceItemICM_Text2.txt");

// Access the DKParts attribute
DKParts* dkParts = (DKParts*)(dkCollection*)
    ddoDocument->getData( ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS));
// Add parts to document
dkParts->addElement((dkDataObjectBase*)(DKDDO*)base);
dkParts->addElement((dkDataObjectBase*)(DKDDO*)baseText1);
dkParts->addElement((dkDataObjectBase*)(DKDDO*)baseText2);

// Add new document to persistent datastore
ddoDocument->add();

```

Para obter mais informações relativas à criação de documentos com partes, consulte o Exemplo Formativo da API SDocModelItemICM.

O DDO possui todas as partes na recolha de partes. Actualize e elimine as partes através do DDO de documento.

O exemplo seguinte demonstra como obter e aceder a partes de um DDO.

Exemplo: Java

```
// NOTE: Print function provided in SDocModelItemICM sample

// Get the DKParts object.
short dataid = ddoDocument.dataId(DKConstant.DK_CM_NAMESPACE_ATTR);
if(dataid==0)
    throw new Exception("No DKParts Attribute Found!
    Either item type does not
    support parts or the document has not been
    explicitly retrieved.");
DKParts dkParts = (DKParts) ddoDocument.getData(dataid);

// Go through part list
dkIterator iter = dkParts.createIterator();
// Create an Iterator

while(iter.more()){
    // While there are items left
    DKDDO part = (DKDDO) iter.next();
    // Move pointer & return next
    System.out.println("Item Id: "+((DKPidICM)part.
    getPidObject()).getItemId());
}
```

Para obter a aplicação exemplo completa, consulte o exemplo SDocModelItemICM.

Exemplo: C++

```
//NOTE: Print function provided in SDocModelItemICM sample

// Get the DKParts object.
short dataid = ddoDocument->dataId
(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS);
if(dataid==0)
    throw DKException("No DKParts Attribute Found!
    Either item type does not
    support parts or the document has not been
    explicitly retrieved.");
DKParts* dkParts = (DKParts*)(dkCollection*)
ddoDocument->getData(dataid);

// Go through part list
dkIterator* iter = dkParts->createIterator();
// Create an Iterator
while(iter->more()){
    // While there are items
    DKDDO* part = (DKDDO*) iter->next()->value();
    // Move pointer & return next
    cout << "Item Id:" << ((DKPidICM*)part->getPidObject
    ())->getItemId()<< endl;
}
delete(iter);
// Free Memory
```

Para obter mais informações relativas à criação de documentos com partes, consulte o Exemplo Formativo da API SDocModelItemICM.

Criar pastas e utilizar o atributo DKFOLDER

Num DDO de pasta, utilize o atributo DKFOLDER para representar a recolha de documentos e outras pastas que pertençam à pasta. O valor deste atributo é um objecto de DKFolder, que é uma recolha de DDOs.

Tal como abaixo indicado, o atributo de DKFolder é definido da mesma forma que o atributo DKParts.

Importante: Apenas para DB2 Content Manager: Uma pasta é um artigo de qualquer tipo de artigo, independentemente da classificação, com o tipo semântico de *pasta*. Qualquer artigo com o tipo semântico de pasta irá conter uma funcionalidade de pasta específica facultada pelo Content Manager, para além de todas as capacidades de artigo que não é de recurso e todas as funcionalidades adicionais disponíveis a partir de uma classificação de tipo de artigo como, por exemplo, modelo de documentos ou recurso. As pastas podem conter um número indeterminado de artigos de qualquer tipo, incluindo documentos e subpastas. Uma pasta é indexada por atributos.

O exemplo seguinte cria uma pasta e adiciona conteúdos ao Content Manager.

Importante: O tipo de artigo definido pelo utilizador, S_simple, terá de ser definido no sistema, conforme indicado no exemplo formativo da API SItemTypeCreationICM.

Exemplo: Java

```
// Create new folder in memory
DKDDO ddoFolder = dsICM.createDDO("S_simple", DKConstant.DK_CM_FOLDER);

// Create and save contents to place in folder
DKDDO ddoDocument = dsICM.createDDO("S_simple", DKConstant.DK_CM_DOCUMENT);
DKDDO ddoFolder2 = dsICM.createDDO("S_simple", DKConstant.DK_CM_FOLDER);
DKDDO ddoItem = dsICM.createDDO("S_simple", DKConstant.DK_CM_ITEM);
ddoDocument.add();
ddoItem.add();
ddoFolder2.add();

// Access the DKFolder attribute
DKFolder dkFolder = (DKFolder)
ddoFolder.getData(ddoFolder.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
                                   DKConstant.DK_CM_DKFOLDER));

// Add contents to folder
dkFolder.addElement(ddoDocument);
dkFolder.addElement(ddoItem);
dkFolder.addElement(ddoFolder2); // Note, Folders can contain sub-folders.

// Save the folder in the persistent datastore.
ddoFolder.add();
```

Para obter mais informações relativas à criação de Pastas, consulte o Exemplo Formativo da API SFolderICM.

Exemplo: C++

```
// Create new folder in memory
DKDDO* ddoFolder = dsICM->createDDO("S_simple", DK_CM_FOLDER);

// Create and save contents to place in folder
DKDDO* ddoDocument = dsICM->createDDO("S_simple", DK_CM_DOCUMENT);
DKDDO* ddoFolder2 = dsICM->createDDO("S_simple", DK_CM_FOLDER);
DKDDO* ddoItem = dsICM->createDDO("S_simple", DK_CM_ITEM);
ddoDocument->add();
ddoItem->add();
ddoFolder2->add();

// Access the DKFolder attribute
```

```

DKFolder* dkFolder = (DKFolder*)(dkCollection*) ddoFolder->getData(
    ddoFolder->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKFOLDER));
// Add contents to folder
dkFolder->addElement(ddoDocument);
dkFolder->addElement(ddoItem);
dkFolder->addElement(ddoFolder2); //Note,Folders can contain sub-folders.

// Save the folder in the persistent datastore.
ddoFolder->add();

```

Para obter mais informações relativas à criação de Pastas, consulte o Exemplo Formativo da API SFolderICM.

No DB2 Content Manager Versão 8, o artigo de pasta não possui os conteúdos de pasta. Um artigo pode ser adicionado a várias pastas. Remover um artigo de uma pasta apenas quebra a relação pasta-conteúdo gerida pelo sistema. Os artigos da pasta devem ser actualizados e eliminados independentemente. Em versões anteriores do Content Manager, o DDO é proprietário dos conteúdos na recolha.

O exemplo seguinte demonstra como obter e aceder a conteúdos de pastas de um DDO.

Exemplo: Java

```

// NOTE: Print function provided in SFolderICM API Education Sample

// Get the DKFolder object.
short dataid = folder.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
    DKConstant.DK_CM_DKFOLDER);
if(dataid==0)
    throw new Exception("No DKFolder Attribute Found!
        DDO is either not a
        Folder or Folder Contents have not been explicitly retrieved.");
DKFolder dkFolder = (DKFolder) folder.getData(dataid);

// Access contents
dkIterator iter = dkFolder.createIterator(); //Create Iterator
while(iter.more()){ // Enquanto houver artigos
    DKDDO ddo = (DKDDO) iter.next(); //Move to & return next element
    System.out.println("Item Id: "+((DKPidICM)ddo.getPidObject())
        .getItemId());
}

```

Para obter a aplicação exemplo completa, consulte o exemplo de instrução SFolderICM.

Exemplo: C++

```

// NOTE: Print function provided in SFolderICM API Education Sample

// Get the DKFolder object.
short dataid = folder->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKFOLDER);
if(dataid==0)
    throw DKException("No DKFolder Attribute Found!
        DDO is either not a Folder
        or Folder Contents have not been explicitly retrieved.");
DKFolder* dkFolder = (DKFolder*)(dkCollection*)folder->getData(dataid);
// Access contents
dkIterator* iter = dkFolder->createIterator(); //Create an Iterator
while(iter->more()){ //Enquanto houver artigos
    DKDDO* ddo = (DKDDO*) iter->next()->value();
    //Move to & return next element
    cout << "Item Id: " << ((DKPidICM*)ddo->getPidObject())->
        getItemId()<< endl;
}

```

Para obter a aplicação exemplo completa, consulte o exemplo de instrução SFolderICM.

Utilizar DKAny (apenas para C++)

O DKAny contém objectos cujo tipo pode variar no tempo de execução.

Um objecto DKAny pode corresponder a qualquer um dos seguintes tipos:

- null
- (unsigned) short
- short
- (unsigned) long
- long
- float
- double
- char
- TypeCode
- DKBoolean
- DKString
- DKDate
- DKTime
- DKTimestamp
- DKByteArray
- DKDecimal
- DKNVPair

Um DKAny pode apenas ser NULL caso não lhe tenha sido atribuído um valor ou o método `DKAny::setNull()` tenha sido invocado (DKAny 1969 any). Após ter-lhe sido atribuído um valor, `DKAny::isNull()`, devolve FALSE.

Além dos tipos acima mencionados, um objecto DKAny pode conter igualmente os seguintes tipos de referência de objectos:

- `dkDataObjectBase*`
- `dkCollection*`
- `void*`

Utilizar código de tipo

Pode determinar o tipo actual de um objecto DKAny invocando a função `typeCode`, que devolve um objecto `TypeCode`, ou seja, `tc_null` para nulo, `tc_short` para curto, e assim sucessivamente.

Consulte a Application Programming Reference para obter uma listagem completa de tipos de códigos.

Gerir memória em DKAny

O DKAny gere a memória para o objecto que contém, excepto se o objecto contido for um um tipo de referência de objectos.

A cópia de operações relacionadas que envolvem referências de objectos irá criar só uma cópia do indicador. O utilizador terá de ter sempre presente a localização dos tipos de referência de objectos durante a cópia e a eliminação.

Utilizar construtores

O DKAny fornece um construtor para cada tipo suportado.

O exemplo que se segue indica como criar um objecto de DKAny que contém alguns dos tipos apresentados na secção anterior.

Exemplo: C++

```
DKAny any1((unsigned short) 10);           // contains unsigned short 10
DKAny any2((long) 200);                     // contains long 200
DKAny any3(DKString("any string"));         // contains DKString
DKAny any4(DKTime(10,20,30));               // contains DKTime
DKAny any5((dkDataObjectBase*) new DKDDO); // contains DKDDO
DKAny any6(new MyObject(5,"abc"));          // contains MyObject
DKAny any7(new DKDDO);                     // shorter form of any5
```

Obter o código de tipo

Utilize a função typeCode para localizar o código de tipo do objecto em DKAny.

Exemplo: C++

```
DKAny::TypeCode type_code;
type_code = any1.typeCode(); // type_code is tc_ushort
type_code = any4.typeCode(); // type_code is tc_time
type_code = any5.typeCode(); // type_code is tc_dobase (object ref)
type_code = any6.typeCode(); // type_code is tc_voidptr since
                             // MyObject is not recognized by DKAny
```

Atribuir um novo valor a DKAny

Para atribuir um novo valor a um objecto DKAny existente, utilize o operador de atribuição do sinal de igual (=). O DKAny fornece uma atribuição para cada código de tipo.

Exemplo: C++

```
DKAny any;           // any contains null
long vlong = 300;
DKTimestamp vts(1997,8,28,10,11,12,999);
dkDataObjectBase* dobase =
(dkDataObjectBase*) new DKDDO;
any = vlong;         // any contains long 300
any = vts;           // any contains timestamp
any = dobase;        // any contains ddo
any = new DKDDO;     // any contains ddo
```

Atribuir um valor de DKAny

A atribuição de um DKAny novamente para um tipo normal requer um operador de conversão.

Exemplo: C++

```
vlong      = (long) any2;           // sets vlong to 200
DKTime at  = (DKTime) any4;         // sets at to (10,20,30)
DKDDO* ddo = (DKDDO*) ((dkDataObjectBase*) any5); // extract the ddo
dkDataObjectBase* dobase = any7;    // extract the DDO
```

O utilizador irá gerar uma excepção de conversão de tipo inválido caso o tipo não seja correspondente. Deste modo, é necessário verificar o código de tipo antes de converter o DKAny para um tipo normal:

```
if (any5.typeCode() == DKAny::tc_dobase)
    dobase = (dkDataObjectBase*) any5;
```

Pode criar uma instrução de tipo de letra para verificar o tipo de DKAny, tal como se segue:

```
switch(any.typeCode()) {
    case DKAny::tc_short:
        // operation for short
        ...
        break;
    case DKAny::tc_ushort:
        // operation for unsigned short
        ...
        break;
    ... etc.
}
```

Se o objecto de DKAny contiver uma referência de objectos, pode obter o conteúdo de DKAny como indicador de vazio, convertendo-o, em seguida, para o tipo apropriado. Contudo, deve utilizar esta operação apenas se conhecer o código de tipo utilizado no DKAny:

```
// knows exactly any5 contains DKDDO
ddo = (DKDDO*) any5.value();
```

Apresentar DKAny

Pode utilizar cout para apresentar o conteúdo de um objecto DKAny.

Exemplo: C++

```
cout << any3 << endl; // displays "any string"
cout << any4 << endl; // displays "10:20:30"
cout << any5 << endl; // displays "(dkDataObjectBase*) <address>",
                      // where address is the memory location of the ddo
```

Destruir o DKAny

Uma vez que o DKAny pode manter uma referência de objecto mas não gere memória para tipos de referência de objecto, é necessário gerir a memória para estes tipos.

O exemplo seguinte gere a memória para um objecto DKAny:

Exemplo: C++

```
DKDDO* ddo = new DKDDO; // creates a DKDDO in the heap
DKAny anyA((dkDataObjectBase*) ddo);
DKAny* anyB = new DKAny(anyA); // creates anyB in the heap
                                // anyA and anyB contains a
                                // reference to the same ddo
...
delete anyB; // delete anyB, does not delete ddo
if (anyA.typeCode() == DKAny::tc_dobase)
    delete ((dkDataObjectBase*) anyA.value()); // deletes the ddo
```

A última instrução de eliminação tem de ser executada antes de sair do âmbito, caso contrário anyA é eliminado, deixando o DDO como uma fuga na memória.

Sugestões de programação

Ao converter um literal inteiro num DKAny, recomenda-se que indique o tipo explicitamente para evitar uma conversão para um tipo não pretendido.

Exemplo: C++

```
any = 10;                // ambiguous
any = (unsigned long) 10; // unambiguous
any = (short) 4;         // unambiguous
```

Utilizar recolhas e iteradores

A dkCollection é uma classe abstracta que fornece os métodos para trabalhar com uma recolha. A DKSequentialCollection é a implementação concreta da dkCollection. As outras recolhas são implementadas como subclasses de DKSequentialCollection. Estas recolhas contêm os objectos de dados como membros.

Os membros da recolha são normalmente objectos do mesmo tipo; no entanto, uma recolha pode conter membros de tipos diferentes.

Importante: Apenas para C++, quando é adicionado um novo membro, a recolha passa a ser proprietária do mesmo. Quando o membro é obtido, o utilizador obtém um indicador para o objecto DKAny no interior da recolha. Este objecto pertence à recolha, o que significa que esta gere a memória para os respectivos membros de DKAny. Um objecto DKAny pode manter uma referência de objectos mas não pode gerir memória para tipos de referência de objectos. Terá de ser o utilizador a gerir a memória para estes tipos.

Utilizar métodos de recolha sequencial

O DKSequentialCollection fornece métodos para adicionar, obter, remover e substituir os respectivos membros. Além disso, possui também um método sort.

O exemplo seguinte ilustra a forma como adicionar um novo membro a uma recolha (o método addElement assume um objecto como parâmetro).

Exemplo: Java

```
DKSequentialCollection sq = new DKSequentialCollection();
String str = " first member ";
sq.addElement(str);          // add a new element at the last position
```

Exemplo: C++

```
DKSequentialCollection sq;
DKAny any = DKString(" first member ");
sq.addElement(any);          // add a new element at last position
                             // any will be copied into the collection
                             // you own the original any, the collection
                             // owns the copy
```

Utilizar o iterador sequencial

O utilizador irá iterar sobre membros da recolha através de iteradores.

As APIs têm dois tipos de iteradores: dkIterator e DKSequentialIterator.

Java

O `dkIterator`, o iterador base, suporta os métodos `next`, `more` e `reset`. A subclasse `DKSequentialIterator` contém mais métodos. Crie um iterador invocando o método `createIterator` na recolha. Depois de ter criado o iterador, pode utilizar os métodos abaixo para cruzar a recolha. O exemplo seguinte ilustra como utilizar um iterador:

```
dkIterator iter = sq.createIterator(); //create an iterator for sq
Object member;
while(iter.more()) { //While there are more members
    member = iter.next(); //move to the next member and get it
    System.out.println(member);
    ....
}
```

C++

Os iteradores são fornecidos para lhe permitir a iteração sobre membros da recolha. Existem dois tipos de iteradores: o iterador base `dkIterator`, que suporta as funções `next`, `more` e `reset`, e a respectiva subclasse `DKSequentialIterator`, que contém mais funções. É criado um iterador através da chamada da função `createIterator` na recolha. Esta função cria um novo iterador e devolve-o ao utilizador. Utilize o código seguinte para iterar numa recolha:

```
dkIterator* iter = sq.createIterator(); //create an iterator for sq
DKAny* member;

// while there are more members
// get the current member and
//advance iter to the next member

while(iter->more()) {
    member = iter->next();

    cout << *member << endl; //display it, if you want to
    ...                      //do other processing
}

delete iter; //do not forget to delete iter
```

Este código permite-lhe executar algumas operações no membro actual antes de mover para o membro seguinte. Uma operação deste tipo poderá ser a substituição de um membro por outro novo, ou removê-lo.

Exemplo: Java

```
String st1 = "the new first member";
sq.replaceElementAt(st1, iter); // replace current member with a new one
.... // or
sq.removeElementAt(iter); // remove the current member
....
```

Exemplo: C++

```
any = DKString("the new first member");

sq.replaceElementAt(any, *iter); // replace current member with a new one
... // or
sq.removeElementAt(); // remove the current member
...
```

Sugestão: Ao remover o membro actual, o iterador avança para o membro seguinte.

Ao remover um membro dentro de um ciclo, verifique-o segundo a forma ilustrada no exemplo seguinte. Deve criar um novo iterador quando um artigo é eliminado do iterador.

Exemplo: Java

```
....
if (removeCondition == true)
    sq.removeElementAt(iter); //remove current member, do not advance the
                              // iterator since it is advanced to the next
                              // after the removal operation
else
    iter.setToNext(); //if no removal, advance the iterator to the
....                  //next position
```

Exemplo: C++

```
...
if (removeCondition == TRUE)
    sq.removeElementAt(*iter); // remove current member, do not advance iter
                              // since it is advanced to the next after
                              // the removal operation
else
    iter->setToNext(); // no removal, advance the iterator
...                  // to the next position
```

Gerir memória em recolhas (apenas para C++)

A recolha gere a memória para os respectivos membros, que são os objectos de DKAny.

As mesmas regras que regem os objectos de DKAny aplicam-se aqui. Caso o objecto incluído no DKAny seja um tipo de referência de objectos, o utilizador é responsável pela gestão da memória quando:

- Destrói a recolha.
- Substitui um membro.
- Remove um membro.

Exemplo: C++

Este exemplo demonstra como gerir a memória nestas situações:

```
// retrieve the member and hang-on to it
member = iter->at();

// code to handle this member as to prevent memory leaks
if (member->typeCode() == DKAny::tc_dobase) {
    // delete it if no longer needed
    delete ((dkDataObjectBase*) member->value());
}

sq.removeElementAt(*iter); // remove it from the collection
```

Em vez de eliminar o membro, poderá adicioná-lo a outra recolha. Deve executar passos idênticos antes de utilizar as funções `replaceElementAt` e `removeAllElement`.

Antes de destruir uma recolha, elimine os seus membros. Poderá escrever uma função de forma a executar esta tarefa e passar esta função para a função `aplicar` da recolha. Suponha que tem uma recolha de objectos de DKAny que contém objectos de DKAttributeDef. O exemplo seguinte elimina a recolha:

```
DKDatastoreICM dsICM;
...
DKAny any = dsICM.listSchemaAttributes("GRANDPA");
```

```

dkCollection* acoll = (dkCollection*) any;
... // use the attributes
acoll->apply(deleteDKAttributeDef); // deletes all members
delete acoll;

```

Neste exemplo, `deleteDKAttributeDef` é a função que assume o objecto de `DKAny` como parâmetro. Está definido da seguinte forma:

```

void deleteDKAttributeDef(DKAny& any) {
    delete ((DKAttributeDef*) any.value());
    any.setNull(); // good practice
}

```

O utilizador poderá escrever a sua função de eliminação para eliminar a recolha ou remover alguns membros antes de eliminar a recolha.

Os destruidores de algumas recolhas conhecidas, como `DKParts`, `DKFolder` e `DKResults`, executam estes passos de remoção necessários. Contudo, estes não gerem a memória ao executar as funções `replaceElementAt`, `removeElementAt` ou `removeAllElement`.

Compreender a recolha e o iterador federados

Utilize uma recolha federada na sua aplicação para processar objectos de dados resultantes de uma consulta como se fossem uma recolha. A recolha federada preserva as relações de subagrupamento que existem entre os objectos de dados.

Uma recolha federada é uma recolha de objectos de `DKResults`. Esta é criada para manter os resultados de `DKFederatedQuery`, que podem ter origem em vários servidores de conteúdos heterogêneos. Cada objecto de `DKResults` contém os resultados de pesquisa de um servidor de conteúdos específico. Uma recolha federada pode conter um número infinito de recolhas imbricadas.

Para concluir as várias etapas de uma recolha federada, crie e utilize um `dkIterator` ou `DKSequentialIterator`. Em seguida, crie outro `dkIterator` para percorrer cada objecto de `DKResults` de forma a iterá-lo e processá-lo de acordo com o servidor de conteúdos do qual o objecto é originário.

Pode criar igualmente um iterador federado, `dkFederatedIterator`, e utilize-o para percorrer todos os membros da recolha, independentemente do servidor de conteúdos do qual do resultado é originário.

Restrição: Não é possível consultar uma recolha federada.

A Figura 5 na página 63 apresenta a estrutura e o comportamento da `DKFederatedCollection`.

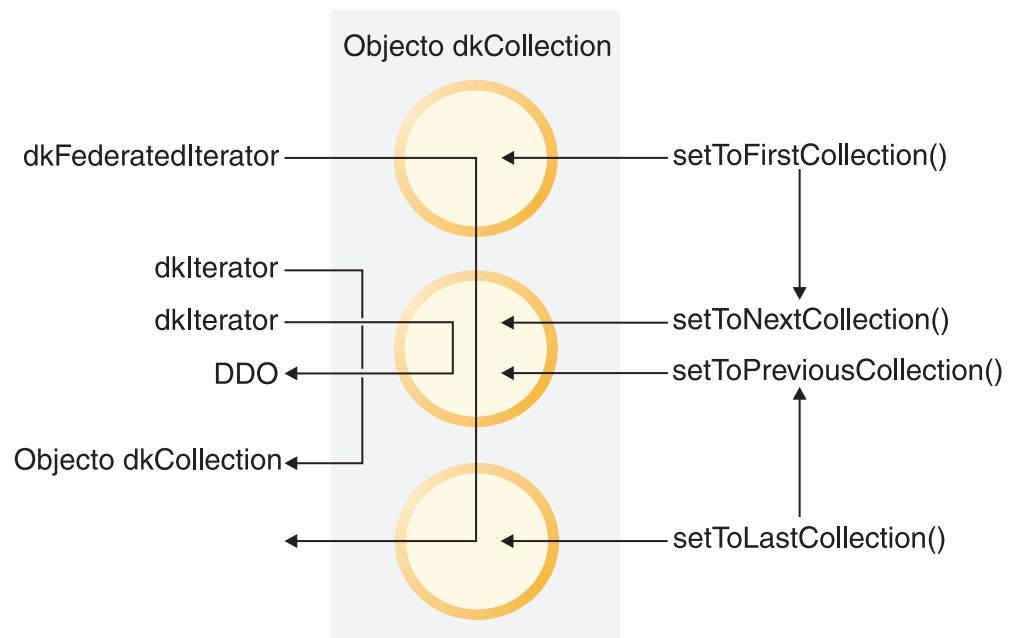


Figura 5. Estrutura e comportamento de DKFederatedCollection

Na Figura 5, o rectângulo representa a DKFederatedCollection, contendo vários círculos menores que são objectos de DKResults. O dkFederatedIterator ultrapassa os limites da recolha e devolve sempre um DDO.

O primeiro dkIterator é um iterador para a DKFederatedCollection e devolve sempre um objecto de DKResults. O segundo dkIterator é um iterador para o segundo objecto de DKResults e devolve um DDO para cada membro da recolha de DKResults.

A função setToFirstCollection no dkFederatedIterator define a posição como o primeiro DDO de DKFederatedCollection. Neste caso, é o primeiro elemento do primeiro objecto da recolha de DKResults. Neste momento, se a função setToNextCollection for invocada, este define a posição do iterador como o primeiro DDO da segunda recolha de DKResults.

A função setToLastCollection no dkFederatedIterator define a posição do iterador como o último DDO de DKFederatedCollection. Neste caso, é o último elemento do último objecto da recolha de DKResults. ~Se a função setToPreviousCollection for invocada, este define a posição do iterador como o último DDO da recolha anterior de DKResults.

Consultar um servidor de conteúdos

Pode pesquisar um servidor de conteúdos e receber resultados num objecto dkResultSetCursor ou DKResults.

No caso de alguns servidores, o utilizador pode criar um objecto de consulta para representar a sua consulta e, de seguida, invocar as funções executar ou avaliar do objecto de consulta. Com o auxílio do servidores de conteúdos, o objecto da consulta executa tarefas de processamento de consultas, tais como preparar e executar uma consulta, supervisionar o estado da execução de uma consulta e armazenar os resultados. Alguns servidores de conteúdos suportam a utilização de

um objecto de consulta como alternativa; dois destes servidores são o das versões anteriores do Content Manager e o servidor de conteúdos federado.

Para os servidores de conteúdos que suportam objectos de consulta, existem quatro tipos de objectos de consulta: paramétricas, de texto, de imagem e combinadas. A consulta combinada é composta por consultas de texto e paramétricas. Nem todos os servidores de conteúdos podem executar consultas combinadas. O DB2 Content Manager anterior suporta a consulta de imagens.

Para o Content Manager, estão integradas consultas paramétricas e de texto. Não deverá utilizar objectos de consulta.

Um servidor de conteúdos utiliza dois métodos para executar uma consulta: executar e avaliar. A função de execução devolve um objecto `dkResultSetCursor` e a função de avaliação devolve um objecto `DKResults`. O objecto `dkResultSetCursor` é utilizado para processar grandes conjuntos de resultados e executar funções de eliminação e actualização na posição actual do cursor do conjunto de resultados. Poderá utilizar a função `fetchNextN` para obter um grupo de objectos numa recolha.

O `dkResultSetCursor` pode ainda ser utilizado para executar novamente uma consulta através da chamada dos métodos `close` e `open`.

O `DKResults` contém todos os resultados da consulta. É possível iterar os artigos na recolha para a frente ou para trás e executar a recolha ou utilizá-la como um âmbito para outra consulta.

Diferenças entre `dkResultSetCursor` e `DKResults`

Existem algumas diferenças fundamentais entre `dkResultSetCursor` e uma recolha `DKResults`.

Uma recolha de `dkResultSetCursor` e de `DKResults` têm as seguintes diferenças:

- O `dkResultSetCursor` funciona como um cursor de servidor de conteúdos. Pode ser utilizado para conjuntos de resultados de grandes dimensões, uma vez que os `DKDDOs` que contém são obtidos um de cada vez. Também pode ser utilizado para voltar a executar uma consulta, de forma a obter os resultados mais recentes.
- O `DKResults` contém o conjunto de resultados total e suporta um iterador bidireccional.
- Deixar um `dkResultSetCursor` aberto durante um longo período de tempo poderá degradar o desempenho de utilizadores simultâneos em alguns servidores de conteúdos.

Utilizar consultas paramétricas

Uma consulta paramétrica é uma consulta que necessita de uma correspondência exacta em relação à condição especificada no predicado da consulta e aos valores de dados armazenados no servidor de conteúdos.

Nota: Para obter informações de consulta sobre o DB2 Content Manager V8, consulte a secção “Consultar um servidor de conteúdos” na página 63.

Trabalhar com o DB2 Content Manager Versão 8.4

Esta secção descreve as interfaces de programação de aplicações (APIs) do conector do DB2 Content Manager Versão 8 Edição 4 (conector do ICM). O conector do ICM é uma extensão da estrutura do DB2 Information Integrator for Content, pelo que se torna essencial que compreenda a estrutura do DB2 Information Integrator for Content.

Pode utilizar as APIs do conector de ICM para construir e implementar aplicações personalizadas que acedem a um servidor de conteúdos do DB2 Content Manager. Pode também utilizar as APIs para integrar as aplicações existentes num servidor de conteúdos do DB2 Content Manager,

Importante: O conector do ICM não suporta a configuração de RMI.

Compreender o sistema do DB2 Content Manager

Os componentes principais do sistema DB2 Content Manager incluem um servidor de bibliotecas, um ou mais objectos de gestor de recursos e um conjunto de interfaces de programação de aplicações (APIs) orientadas para objectos. Para administrar o sistema do DB2 Content Manager, é facultado também ao utilizador um cliente de administração do sistema com base em Java.

O servidor de bibliotecas fornece-lhe capacidades de exemplificação de dados flexíveis, acesso protegido ao seu sistema, gestão eficiente do conteúdo e outras funções. O servidor de bibliotecas gere as relações entre artigos no sistema e controla o acesso a todas as informações do sistema, incluindo as informações armazenadas no gestor de recursos.

O gestor de recursos é o componente que armazena o conteúdo actual de qualquer objecto binário, como uma imagem digitalizada, um documento de office ou um video. Pode integrar outros tipos de gestor de recursos, como o DB2 Content Manager VideoCharger ou outros produtos que não sejam IBM, no sistema do DB2 Content Manager. Com o gestor de recursos poderá concluir as seguintes tarefas:

- Mova automaticamente o conteúdo em suportes de dados de alta velocidade dispendiosos para um suporte de dados mais lento e menos dispendioso, com o System Managed Storage (SMS).
- Aceda ao gestor de recursos directamente a partir do navegador da Web.
- Obtenha a totalidade ou parte de um objecto.
- Sincronize os seus dados com o servidor de bibliotecas.

As APIs facultam aplicações com acesso ao sistema do DB2 Content Manager. As APIs estão disponíveis para Java e C++. Utilizando as APIs, as aplicações do utilizador podem tirar partido de toda a funcionalidade do DB2 Content Manager como, modelação de dados, pesquisa de texto e pesquisa paramétrica integradas, acesso e entrega de dados de terceiros, etc.

O diagrama da Figura 6 na página 66 ilustra como os componentes do sistema se encaixam. Tenha presente que esta é apenas uma implementação de um sistema DB2 Content Manager. Noutra configuração de sistema, pode ter quatro exemplos de gestor de recursos.

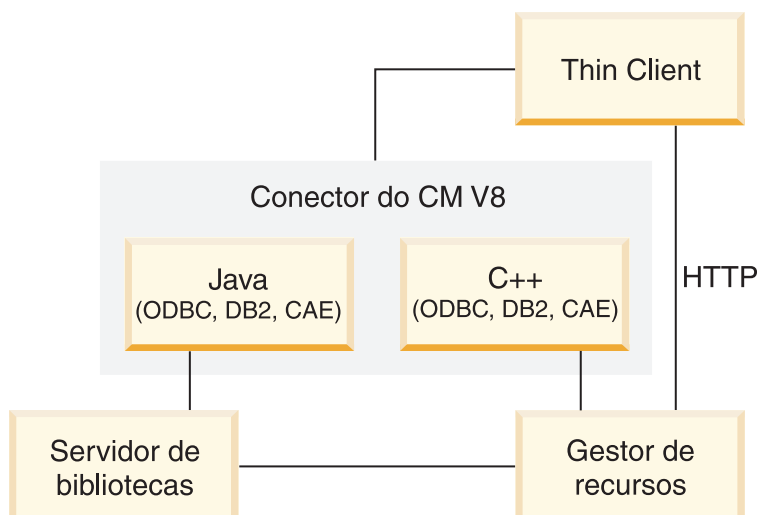


Figura 6. Configuração do sistema

Compreender os conceitos do DB2 Content Manager

Esta secção descreve conceitos importantes do DB2 Content Manager.

É importante que compreenda os conceitos do DB2 Content Manager antes de prosseguir com as tarefas de programação.

Itens

Um *artigo* é a entidade básica gerida pelo servidor de bibliotecas.

Os exemplos de itens incluem uma política, uma participação, um número de telefone, etc. Um artigo é um termo genérico para uma instância de um tipo de artigo. Caso um objecto seja um pedaço discreto de conteúdo digital, um artigo será uma representação desse objecto. O artigo não é o objecto, mas identifica de forma precisa como e onde o localizar. No sistema, os artigos representam objectos incluindo documentos e pastas. Para definir objectos empresariais, como um documento, trabalhe com as definições do artigo.

Quando uma aplicação cria um artigo, o DB2 Content Manager atribui ao artigo vários atributos definidos pelo sistema e permite ao utilizador definir os seus próprios atributos.

Os atributos definidos pelo sistema incluem uma marca de hora de criação e um identificador de artigo (ID do artigo). O ID do artigo é único para cada artigo. O ID de artigo é armazenado pelo DB2 Content Manager e utilizado para localizar o artigo no servidor de bibliotecas. Ao escrever a aplicação, utilize o ID de artigo para aceder a todos os dados associados ao artigo.

Atributos

Um *atributo* é uma unidade de dados que descreve uma determinada característica ou propriedade (por exemplo, nome, endereço, idade, etc.) de um artigo e pode ser pesquisada de modo a localizar o artigo.

Pode agrupar atributos para constituir grupos de atributos. Por exemplo, o atributo endereço pode ser constituído por um grupo de atributos que incluem a rua, cidade, distrito e código postal.

O utilizador pode também definir atributos que possuem vários valores. Estes atributos são denominados atributos de vários valores, que são implementados como componentes descendentes. Por exemplo, pode armazenar vários endereços, o endereço particular, o endereço do local de trabalho, etc., para um proprietário de política.

Para obter informações adicionais, consulte o exemplo `SAttributeDefinitionCreationICM`.

Tipos de artigos

Um *tipo de artigo* (classe de índice em versões anteriores do DB2 Content Manager) é, essencialmente, um modelo para definir e, posteriormente, localizar artigos semelhantes. Um tipo de artigo é formado por um componente raiz, zero ou mais componentes descendentes e uma classificação.

Um tipo de artigo é a estrutura geral que contém todos os componentes e dados associados. Por exemplo, num caso de seguro, um tipo de artigo de apólice contém artigos com atributos como, por exemplo, o número da apólice, o nome, participação, etc.

No Content Manager, existem quatro classificações de tipos de artigo: não de recurso, de recurso, documento (também denominados modelo de documentos) e parte de documento. Um tipo de artigo que não é de recurso representa entidades que não se encontram armazenadas num gestor de recursos. Um tipo de artigo de recurso representa objectos armazenados num gestor de recursos como, por exemplo, ficheiros num sistema de ficheiros, vídeo clips num servidor de vídeo, LOBs (objectos grandes) em tabelas de base de dados, etc. Um tipo de artigo de documento representa entidades que contêm partes de documentos, que por sua vez contêm conteúdos de recurso, tal como um tipo de artigo de recurso. Um tipo de artigo de parte de documento representa objectos armazenados no gestor de recursos, mas são partes de um documento, contidas e detidas por um tipo de artigo de documento. O DB2 Content Manager faculta um conjunto base de tipos de artigo de recurso: objectos de LOB, de texto, imagem, sequência e de vídeo.

Para obter mais informações relativas a tipos de artigo, consulte o exemplo `SItemTypeCreationICM`.

Componentes raiz e descendentes

Um tipo de artigo é composto por componentes: um componente raiz e qualquer número de componentes descendentes, que são opcionais.

Um *componente raiz* é o primeiro ou único nível de um tipo de artigo hierárquico. Um tipo de artigo é formado por atributos definidos pelo sistema e pelo utilizador. Internamente, o tipo de artigo mais básico contém apenas um componente.

Um *componente descendente* é um segundo nível opcional(ou um nível mais baixo) de um tipo de artigo hierárquico. Cada componente descendente está directamente associado ao seu nível superior. A Figura 7 na página 68 apresenta o diagrama do meta-modelo do DB2 Content Manager. Apresenta os componentes raiz e descendente e as suas relações na formação da hierarquia de um artigo. O diagrama também mostra ligações, referências, artigos de recurso e objectos de

recurso.

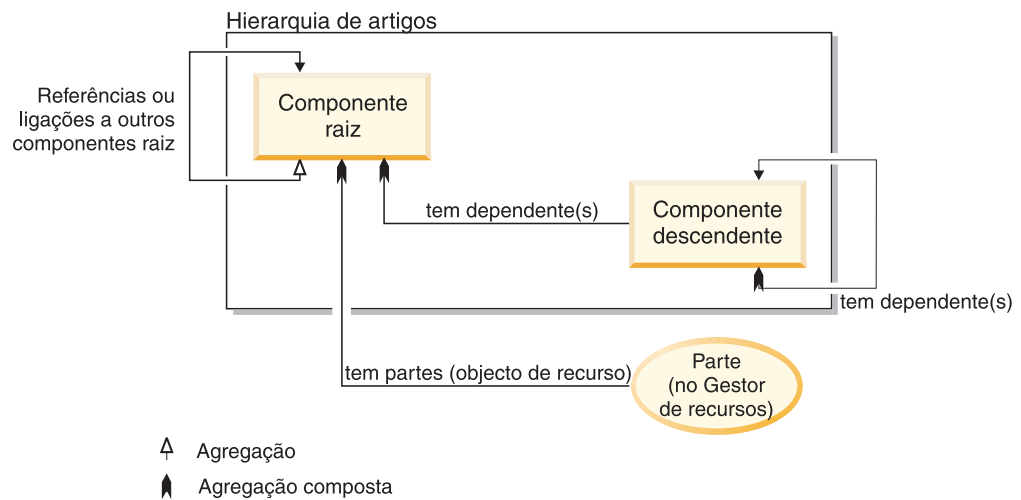


Figura 7. O meta-modelo de DB2 Content Manager: Descrição lógica.

A Figura 8 apresenta um exemplo do modelo de dados para uma aplicação de seguro com a apólice como componente raiz, tendo a participação, o relatório da polícia e a estimativa de danos como componentes descendentes.

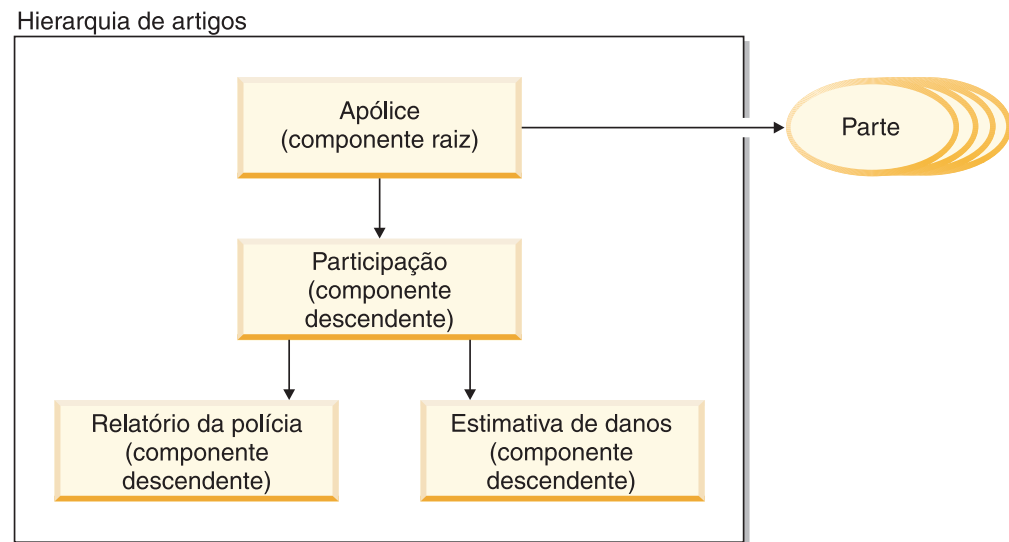


Figura 8. Hierarquia de componentes

Para obter mais informações sobre conceitos de modelação de dados do DB2 Content Manager, consulte o exemplo SItemCreationICM.

Objectos

Um *objecto* (também denominado por conteúdo de recurso) é qualquer conteúdo digital que um utilizador pode armazenar, obter e manipular como uma única unidade (por exemplo, imagens de JPEG, áudio MP3, vídeo AVI e um bloco de texto de um livro).

Um objecto é sempre armazenado num gestor de recursos. O acesso a um objecto é controlado pelo servidor de bibliotecas.

Para obter mais informações, consulte o exemplo `SResourceItemCreationICM`.

Ligações e referências

Uma *ligação* associa apenas dois itens e fornece meios para aceder aos itens que liga ou a outros itens que podem estabelecer ligação a esses dois artigos.

Pode utilizar uma ligação para modelar ou várias associações entre itens. Tal como foi demonstrado na Figura 7 na página 68, as ligações apenas podem ser utilizadas para associar componentes raiz de artigos. As ligações não estão relacionadas com um versão específica de um artigo. Pode definir um número indeterminado de ligações. As ligações são uma forma flexível de ligar uma fonte a um destino. Uma ligação apenas associa dois artigos e faculta formas de aceder aos artigos que liga ou de aceder a artigos que podem estabelecer ligação a esses dois artigos. A utilização de ligações é determinada no tempo de execução pela aplicação do utilizador. Pode utilizar um número indeterminado de ligações na sua aplicação.

As ligações são blocos de construção flexíveis, abertos e irrestritos, que pode utilizar nas suas aplicações. Como tal, o utilizador tem como opção colocar restrições a ligações da sua aplicação.

Uma das formas através das quais pode utilizar uma ligação é representando a utilização de pastas ou a relação contentor-contido. Se optar por implementar a utilização de pastas recorrendo a ligações, tenha em atenção que o contentor não é proprietário do contido, o que significa que os itens do contentor não são eliminados quando o contentor é eliminado. Para obter mais informações relativas a ligações, consulte o exemplo `SLinksICM`.

Uma *referência* encontra-se entre um componente (quer seja raiz ou descendente) e outro componente raiz. Uma referência é representada como um atributo de referência num componente definido na altura da concepção. Uma definição de componente pode possuir qualquer número, especificado durante a definição do tipo de componente, de atributos de referência que remetem para outros componentes raiz. Uma referência normalmente não indica propriedade, mas o utilizador pode implementar uma relação de propriedade, caso seja necessário.

Ao adicionar uma referência a um tipo de componente, os artigos desse tipo de componente podem fazer referência a outro artigo. Em termos do DDO, este possui um atributo que é identificado pelo nome da referência. O valor do atributo pode ser definido para outro DDO. O valor do atributo para DDO consiste no DDO que é referido pela referência.

As referências podem ser definidas quer em tipos de componente raiz, quer em tipos de componente descendente, fazendo referência a outro tipo de componente raiz. Consulte a Figura 7 na página 68. As referências podem também referir uma versão específica de um artigo, enquanto as ligações fazem referência a todas as versões. Para obter mais informações relativas a atributos de referência, consulte o exemplo `SReferenceAttrDefCreationICM`.

Activar a criação automática de ligações com tipos de itens activados para versões

O DB2 Content Manager fornece a criação automática de ligações com tipos de itens activados para versões.

As versões mais antigas do DB2 Content Manager fornecem uma implementação mais restrita da criação automática de ligações, a qual era denominada de criação

automática de pastas. A criação automática de pastas foi restringida para a ligação de pastas apenas. Contudo, o DB2 Content Manager permite a criação automática de ligações com tipos de itens activados para versões desde o Fix Pack 4 da Versão 3.

A função de criação automática de ligações permite que dois tipos de itens sejam ligados por um ou mais atributos. O tipo de artigo origem é o contentor (como, por exemplo, a pasta) e o destino é o contido (como, por exemplo, o tipo de documento cujos itens estão contidos nos itens de pasta). Se dois itens forem ligados, todos os atributos de ligação terão de corresponder. São possíveis vários níveis de criação automática de ligações. Por exemplo, o tipo de documento Doc pode ser automaticamente ligado à Pasta 1 e a Pasta 1 pode ser ligada à Pasta 2, que é automaticamente ligada à Pasta 3, etc.

A criação automática de ligações pode ser activada ao nível do componente raiz ou descendente, ou ambos. Um tipo de artigo de documento pode também ser ligado a um tipo de artigo de pasta através de atributos na raiz ou no componente descendente do primeiro nível na estrutura hierárquica.

Desde o Fix Pack 4, o DB2 Content Manager permite a ligação de tipos de itens destino activados para versões. O tipo de artigo origem (ou pasta) ou qual será estabelecida a ligação tem de estar activado para criação de versões. As ligações para o tipo de artigo destino (ou documento) são mantidas pela utilização de atributos da versão actual do artigo.

Durante a criação de ligações automática de um tipo de artigo destino activado para versões, o DB2 Content Manager aplica as seguintes regras:

Actualização para a versão mais recente do artigo destino

Se a actualização se destinar a um atributo de criação de ligações, o DB2 Content Manager:

- Remove todas as ligações às versões anteriores
- Cria a nova versão com a actualização
- Cria novamente as ligações utilizando a versão mais recente do artigo destino para qualquer um dos respectivos tipos origem

Eliminação da versão mais recente de um documento

Para eliminar a versão mais recente de um documento, o DB2 Content Manager:

- Remove todas as ligações aos itens
- Elimina a versão mais recente
- Restaura a ligação entre a versão anterior e qualquer um dos tipos de artigo origem

Reindexação de um documento

Para reindexar um documento, o DB2 Content Manager:

- Move apenas a versão mais recente para o novo tipo de artigo. As ligações entre o tipo de artigo antigo e quaisquer tipos de artigo origem para esse artigo são quebradas.
- Estabelece novas ligações entre o novo tipo de artigo e quaisquer tipos de artigo origem que possam existir

Restrições:

Ao definir uma regra de criação automática de ligações que envolva um atributo de componente descendente ligado a um atributo raiz origem, a cardinalidade mínima do componente descendente tem de ser definida como 1. A cardinalidade máxima pode ser definida como 1 ou um valor superior. No entanto, terá de manter a exclusividade do atributo dos tipos de itens origem, caso o atributo seja exclusivo.

Existem algumas restrições relativamente aos atributos de ligação. Não é possível atribuir uma marca de hora ao tipo de atributos de ligação. Ao definir uma ligação de pasta automática entre dois tipos de artigo, poderá ter uma ligação raiz-raiz e uma ligação descendente-raiz em simultâneo, mas o atributo de ligação não poderá ser o mesmo para a raiz e o descendente. Para activar a criação automática de ligações ao nível do componente descendente, o atributo de ligação tem de ser igualmente definido ao nível da raiz.

Documentos

Existem dois tipos de documentos que poderá possuir no sistema.

O primeiro tipo de documento é um artigo de tipo semântico document, que deve conter informações que componham um documento. Este tipo de documento pode ser autónomo ou conter partes, caso tenha implementado o modelo de documentos no modelo de dados. Para obter mais informações sobre este tipo de documento, consulte o exemplo SItemCreationICM no directório samples.

O outro tipo de documento é um artigo criado a partir de um documento classificado como tipo de artigo (também denominado "modelo de documentos"). Este tipo de documento contém partes de documento, uma implementação específica do modelo de documentos do DB2 Content Manager. No modelo de documento, os artigos são uma extensão de artigos que não de recurso. As partes de modelo de documento são artigos de recurso. As partes de documento podem incluir vários tipos de conteúdos incluindo, por exemplo, texto, imagens e folhas de cálculo. Para obter mais informações relativas ao modelo de documentos, consulte o exemplo SDocModelItemICM.

Pastas

Uma *pasta* é um artigo que pode conter outros artigos de qualquer tipo.

Isto implica que poderá também incluir outras pastas. No DB2 Content Manager Versão 8, o conceito de pastas é implementado utilizando relações de ligação entre artigos. Os artigos podem conter outros artigos para formar uma hierarquia de contenção, denominada hierarquia de pastas. Por exemplo, um artigo de políticas pertence ao tipo de artigo de políticas e, potencialmente, possui várias participações, fazendo das políticas uma pasta que detém outros artigos como, por exemplo, uma fotografia, um número de segurança social, etc. As pastas são muito flexíveis porque um artigo qualquer pode ser uma pasta e pode conter um número qualquer de outros artigos. Para obter mais informações, consulte o exemplo SFolderICM.

Criação de versões

A *criação de versões* é a capacidade de armazenar e manter várias versões de um artigo, incluindo versões de componentes descendentes do artigo. O utilizador

especifica as regras de criação de versões quando define um tipo de artigo. Se um tipo de artigo for activado para criação de versões, são elaboradas versões de todos os artigos desse tipo de artigo.

Existem dois tipos de criação de versões: constante ou por aplicação. Quando um artigo é activado para elaborar constantemente versões, é automaticamente criada uma nova versão do artigo sempre que o artigo é actualizado e armazenado no servidor de conteúdos. Quando um artigo é activado para criação de versões por aplicação, o sistema apenas cria uma nova versão quando especificado pela aplicação do utilizador.

A criação de versões é processada pelo servidor de bibliotecas do DB2 Content Manager. Cada versão de um artigo, cujo conteúdo é armazenado no gestor de recursos, possuirá a sua própria cópia do conteúdo. Segue-se uma lista de características importantes da criação de versões:

- A criação de versões envolve um componente raiz e toda a sua hierarquia.
- Os tipos de artigos podem ter uma das três políticas de criação de versões possíveis: criar sempre versões, nunca criar versões (predefinição) e criação de versões controlada pela aplicação.
- Todas as versões de um artigo do sistema CM são pesquisáveis e passíveis de serem obtidas.
- Todas as versões de um artigo podem ser actualizadas e eliminadas.
- Para tipos de artigo com criação de versões controlada pela aplicação, quando o artigo é actualizado, o utilizador pode optar entre aplicar as actualizações à versão existente ou criar uma nova versão com base nas actualizações.
- Cada versão de um artigo possui o seu identificador persistente (PID). O PID possui várias partes, das quais duas são relevantes no contexto actual. A primeira parte relevante é o ID do Artigo, que é igual em todas as versões diferentes do artigo. A outra parte é o número da versão. Todas as versões do artigo possuem um número de versão diferente que pode ser obtido e definido como uma cadeia.
- Um tipo de artigo pode ser configurado para manter apenas um número limitado de versões de cada artigo. Se a actualização de um artigo exceder o número máximo permitido de versões, a versão guardada mais antiga é largada e é criada uma nova versão pelo sistema.
- Se um artigo activado para versões for reindexado, todas as versões anteriores do artigo são automaticamente eliminadas.
- Os componentes descendentes de um artigo herdam a versão do componente ascendente.
- A versão de um tipo de componente descendente não pode ser alterada, já que este segue a criação de versões do seu tipo ascendente.
- As regras de criação de versões nível de parte podem ser obtidas no objecto de relação do tipo de artigo que representa os tipos.

Segue-se um exemplo que demonstra como trabalhar com números de versões.

```
DKPidICM pid = (DKPidICM)ddo.getPidObject();
String version = pid.getVersionNumber();
....
pid.setVersionNumber(version);
```

Para obter informações relativas à criação de versões, consulte os exemplos `SItemUpdateICM` e `SItemTypeCreationICM`.

Controlo de acessos

O modelo de controlo de acessos do DB2 Content Manager é constituído pelos seguintes elementos fundamentais: privilégios e conjuntos de privilégios, entidades controladas, utilizadores e grupos de utilizadores e listas de controlo de acessos.

Os vários elementos de controlo de acessos funcionam de acordo com a seguinte descrição. A cada utilizador do DB2 Content Manager é concedido um conjunto de privilégios de utilizador. Estes privilégios definem a operação que um utilizador pode executar. Os direitos de acesso efectivos de um utilizador nunca excederão os privilégios definidos pelo utilizador.

O modelo de controlo de acessos do DB2 Content Manager é aplicado à entidade controlada. Uma *entidade controlada* é uma unidade de dados de utilizador protegidos. No DB2 Content Manager, a entidade controlada pode estar ao nível de um artigo, tipo de artigo, ou de toda a biblioteca. Por exemplo, pode associar uma ACL a um tipo de artigo para reforçar o controlo de acessos a nível do tipo do artigo. As operações sobre entidades controladas são reguladas por uma ou mais regras de controlo, denominadas listas de controlo de acessos (ACLs). Todas as entidades controladas no sistema do Content Manager devem estar ligadas a uma ACL.

Quando um utilizador indica uma operação sobre um artigo, o sistema verifica o privilégio do utilizador e a ACL associada ao artigo para determinar se o utilizador tem direito de efectuar uma operação no artigo. Evidentemente, o direito de aceder a um artigo também requer o direito de aceder ao tipo de artigo em que o artigo se encontra definido. A Figura 9 demonstra um exemplo de como o sistema determina os direitos de acesso do utilizador a um artigo com base em privilégios e ACLs.

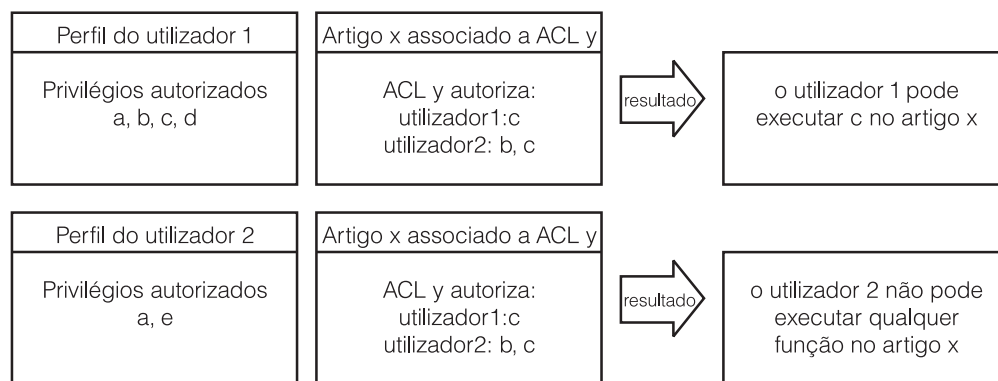


Figura 9. Diagrama de controlo de acessos

Privilégios e conjuntos de privilégios

Os privilégios permitem a um utilizador executar uma acção específica num artigo do sistema como, por exemplo, criá-lo ou eliminá-lo.

A cada utilizador do DB2 Content Manager é concedido um conjunto de privilégios de utilizador. Os privilégios definem o máximo de operações que um utilizador pode executar em informações do sistema DB2 Content Manager. Os direitos de acesso de um utilizador não excedem os privilégios definidos pelo utilizador.

O DB2 Content Manager faculta um determinado número de privilégios predefinidos que o utilizador não pode alterar, denominados privilégios definidos pelo sistema. Pode também definir os privilégios do utilizador, denominados privilégios definidos pelo utilizador. O utilizador aplica os privilégios definidos pelo utilizador na aplicação utilizando as rotinas de saída do utilizador.

Todos os privilégios possuem um código único gerado pelo sistema, denominado código de definição de privilégios. Os códigos de definição de privilégios entre 0 e 999 estão reservados para privilégios definidos pelo sistema. Pode utilizar códigos de 1000, e acima de 1000, para privilégios definidos pelo utilizador.

Os privilégios definidos pelo sistema são classificados em duas categorias: privilégios de administração do sistema e privilégios de acesso a dados. Pode utilizar os privilégios de administração do sistema para modelar dados do utilizador e administrar e manter o sistema DB2 Content Manager. O utilizador necessita de privilégios de administração para concluir tarefas como, por exemplo, configurar o sistema, gerir a configuração de servidor de bibliotecas e gerir tipos de artigo. Pode utilizar os privilégios de acesso a dados para aceder e alterar os dados do sistema como, por exemplo, artigos e tipos de artigos.

No modelo de ACL do Content Manager, existem três privilégios sempre seleccionados no código de ACL do tipo de artigo, mesmo quando é utilizada a ACL de nível de artigo. Os três privilégios são: `ItemTypeQuery`, `ItemAdd` e `ItemMove`.

O privilégio `ItemTypeQuery` para um tipo de artigo (ou vista de um tipo de artigo) determina se o utilizador pode obter o tipo de artigo ou vista. Se não tiver este privilégio na ACL de tipo de artigo, não poderá visualizar o tipo de artigo ao enumerar tipos de itens. Isto significa não será possível aceder a nenhum dos itens neste tipo de artigo. Por conseguinte, é impossível obter um artigo a partir de um tipo de artigo através de obtenção ou consulte, independentemente da ACL do artigo.

Os privilégios `ItemAdd` e `ItemMove` estão sempre seleccionados na ACL do tipo de artigo, mesmo ao utilizar a ACL de nível de artigo. É necessário ter o privilégio `ItemAdd` na ACL de tipo de artigo para poder criar itens no tipo de artigo. Da mesma forma, é necessário ter o privilégio `ItemMove` na ACL de tipo de artigo destino para poder reindexar um artigo neste tipo de artigo.

Um grupo de privilégios atribuídos a um utilizador é um conjunto de privilégios. O conjunto de privilégios gerais de um utilizador determina a quantidade máxima de operações que o utilizador pode executar. Por exemplo, um conjunto de privilégios pode conter os privilégios de criação, actualização e eliminação. Os conjuntos de privilégios permitem uma administração do sistema mais fácil. O utilizador deve agrupar privilégios num conjunto antes de os utilizar. Não existe um limite de privilégios que um conjunto pode conter.

Os conjuntos de privilégios predefinidos do DB2 Content Manager incluem:
Privilégios de administração do sistema

AllPrivSet; PrivSetCode: 1

Um utilizador com este conjunto de privilégios pode executar todas as funções em todas as entidades do DB2 Content Manager. Os privilégios contidos neste conjunto de privilégios incluem: Todos os privilégios definidos pelo sistema e definidos pelo utilizador.

NoPrivSet; PrivSetCode: 2

Os utilizadores que possuam este conjunto de privilégios não podem executar quaisquer acções em entidades do DB2 Content Manager. Os privilégios contidos neste conjunto incluem: Nenhum.

SystemAdminPrivSet; PrivSetCode: 3

Os utilizadores com este conjunto de privilégios podem executar todas as funções de administração do sistema e modelação de dados do DB2 Content Manager. Os privilégios incluídos neste conjunto de privilégios incluem: Nenhum

ItemAdminPrivSet; PrivSetCode: 4

Os utilizadores com este conjunto de privilégios podem executar todas as funções de modelação de dados e acesso a artigo do DB2 Content Manager. Os privilégios contidos neste conjunto incluem:

- Privilégio de tipo de artigo definido pelo sistema
- Privilégio de selecção de SQL de artigo
- Privilégio de consulta de tipo de artigo
- Privilégios de consulta de artigo
- Privilégio de adição de artigo
- Privilégio de atributo definido pelo utilizador de conjunto de artigos
- Privilégio de atributo definido pelo sistema de conjunto de artigos
- Privilégio de eliminação de artigo
- Privilégio de deslocação de artigo
- Privilégio de artigo de ligação a
- Privilégio de artigo ligado
- Privilégio de artigo detentor
- Privilégio de artigo detido
- Privilégio de artigo de adição de ligação
- Privilégio de artigo de alteração de ligação
- Privilégio de artigo de remoção de ligação
- Privilégio de saída de artigo

Tabela 5. Códigos dos Privilégios

Nome do privilégio	Código
ICMLogon	1
SystemAdmin	40
SystemDefineItemType	45
ItemSQLSelect	121
ItemTypeQuery	122
ItemQuery	123
ItemAdd	124
ItemSetUserAttr	125
ItemSetSysAttr	126

Utilizadores e grupos de utilizadores

Um grupo de utilizadores é apenas um agrupamento de conveniência de utilizadores individuais que executam tarefas semelhantes.

É quase certo que o utilizador tem um grupo de utilizadores que exigem o mesmo tipo de acesso ao sistema. Por exemplo, todos os seguradores de uma empresa de seguros necessitam de privilégios de pesquisa, obtenção e actualização para o tipo de artigo de participações. Poderá agrupar os seguradores e outros utilizadores com necessidades comuns de acesso num grupo de utilizadores. No entanto, não poderá colocar um grupo de utilizadores noutra grupo de utilizadores.

Um grupo de utilizadores é formado por nenhum ou mais utilizadores. Não tem de atribuir um conjunto de privilégios a um grupo de utilizadores. Cada utilizador num grupo de utilizadores possui um conjunto de privilégios. Um grupo de utilizadores facilita a criação de listas de controlo do acesso a objectos no seu sistema. Um grupo de utilizadores não pode pertencer a outros grupos.

Adicionar e remover utilizadores em grandes grupos de utilizadores utilizando as APIs de C++

Ao adicionar um utilizador a um grupo de utilizadores de grandes dimensões, com cerca de 1000 utilizadores, serão necessários cerca de 45 minutos para concluir a actualização. Esta demora ocorre em C++ apenas quando são utilizados os seguintes métodos:

```
userGroup->addUser(userDef);  
userMgmt->update(userGroup);
```

Para diminuir o tempo necessário para concluir a actualização, crie as duas novas APIs que se seguem:

```
void DKUserMgmtICM::update(dkUserGroupDef* pUserGroup, dkCollection* pUserColl,  
                           short sAction)  
void DKUserMgmtICM::update(dkUserGroupDef* pUserGroup, dkCollection* pUserColl,  
                           short sAction, short sOption)
```

A utilização das APIs acima mencionadas podem reduzir significativamente o tempo necessário para concluir a tarefa.

Listas de controlos de acessos (ACLs) e e regras de nível de associação de ACL

Quando um utilizador cria um artigo no sistema DB2 Content Manager, esse utilizador deve definir o acesso que outros utilizadores possuirão a esse artigo e quais as operações que nele poderão executar.

Listas de controlo de acessos (ACLs)

A lista de utilizadores que possuem acesso ao artigo e as operações que estes podem executar no artigo são denominadas *lista de controlo de acessos* (ACL). Uma ACL é utilizada no tempo de execução para determinar quais as operações de CRUD que um utilizador pode executar num determinado artigo ou tipo de artigo. Uma ACL é uma lista que contém num ou mais IDs de utilizador individual ou grupos de utilizadores e os seus privilégios associados. O utilizador pode associar artigos, tipos de artigos e listas de trabalho a uma ACL. Os conjuntos de privilégios definem a capacidade máxima de utilização do sistema de um utilizador individual, uma ACL restringe o acesso desse utilizador individual a um artigo. Por exemplo, se a ACL permitir que o artigo fotografia seja eliminado mas João não tiver o privilégio de eliminação no seu conjunto de privilégios, João não pode eliminar a fotografia.

Uma ACL de utilizador consiste numa ACL criada por um utilizador final com o privilégio UserACLOwner e que pode ser atribuída apenas a itens. Os utilizadores

podem pesquisar em ACLs de utilizador. As ACLs de utilizador não são apresentadas no cliente de administração do sistema. Um utilizador incluído na ACL de utilizador e com o privilégio UserACLOwner, ou um administrador, pode modificar uma ACL de utilizador utilizando as APIs.

Uma entidade controlada está associada a uma ACL específica através do código de ACL. Quando associadas a entidades controladas, as ACLs definem a autorização das entidades associadas e não englobam os privilégios do utilizador. É aplicada uma ACL e os privilégios de utilizador são verificados.

Uma ACL contém uma ou mais regras de ACL. Os utilizadores especificados nas regras de controlo de acessos podem ser utilizadores individuais, grupos de utilizadores ou públicos. A interpretação é determinada pelo campo Tipo de Utilizador de uma regra. Os tipos de regras, por uma questão de ilustração, podem possuir os nomes Regra de ACL para Utilizador e Regra de ACL para Grupo e Regra de ACL para Público, respectivamente. Ao especificar público, a Regra de ACL para Público autoriza todos os utilizadores a executar operações especificadas nos Privilégios de ACL na entidade associada, desde que o utilizador passe na verificação de Privilégios do Utilizador. Os privilégios de ACL na entidade limite em Público podem ser configurados no nível Sistema. A capacidade de abrir uma entidade limite em Público pode ser configurada a nível do sistema. O parâmetro de configuração é denominado **PubAccessEnabled** (definido na tabela ICMSTSysControl). Quando desactivadas, todas as Regras de ACL para Público são ignoradas durante o processo de controlo de acessos.

Dentro da mesma ACL, um utilizador pode ser especificado em mais do que um tipo de regra. A prioridade dos três tipos, do mais alto ao mais baixo, é Regra de ACL para Público, Regra de ACL para Utilizador e Regra de ACL para Grupo. Ao aplicar verificações de ACL, se passar um tipo de regra de prioridade superior, a autorização é processada e o processo pára. Se a verificação de Regra de ACL para Público tiver falhado, o seguinte processo irá continuar nos tipos de regra de prioridade inferior.

Se a verificação de Regra de ACL para Utilizador tiver falhado, a verificação pára. A Regra de ACL para Grupo não é verificada. Não há necessidade de prosseguir com a verificação no tipo Grupo pois, caso um utilizador efectue uma verificação de utilizador individual, este será excluído do acesso ao tipo de grupo com base no algoritmo de controlo de acessos. A verificação de controlo de acessos de Tipo de utilizador individual e Tipo de grupo não é um processo sequencial. Trata-se de uma situação com duas opções, apesar de não ser prejudicial a realização de uma verificação sequencial.

Se o utilizador não tiver passado na verificação de tipo de utilizador individual (ou o utilizador não possuir uma regra na tabela Lista de Acessos), o processo de verificação irá avançar para o tipo de grupo. Se o utilizador pertencer a um dos grupos e a verificação do privilégio passar, a autorização é processada e o processo pára. Caso contrário, é negado acesso e o processo pára igualmente. Quando se especifica um utilizador em mais do que uma Regra de ACL para um Grupo, o utilizador é autorizado pela união de todos os Privilégios de ACL dessas regras. Nunca se especifica um utilizador em mais do que uma regra de ACL para Utilizador.

O sistema do CM faculta as seguintes ACLs pré-configuradas: SuperUserACL, NoAccessACL e PublicReadACL.

SuperUserACL

Esta ACL é formada por uma única regra que autoriza o utilizador pré-configurado ICADMIN do CM a executar todas as funções de CM (AllPrivSet) nas entidades associadas.

NoAccessACL

Esta ACL é formada por uma única regra que não permite, a todos os utilizadores de CM (público), a execução de acções (NoPrivSet).

PublicReadACL

Esta ACL é formada por uma única regra que permite, a todos os utilizadores de CM (ICMPUBLIC), a capacidade de leitura (ItemReadPrivSet). Este é o valor predefinido atribuído ao DfltACLCode de um utilizador.

Regras de nível de associação de ACL

O **ACL Binding Level (Nível de associação da ACL)** é um parâmetro de configuração que permite ao utilizador seleccionar a ACL utilizada para verificação de ACL. Uma verificação de ACL é uma verificação adicional utilizada no tempo de execução para determinar quais as operações de CRUD que um utilizador pode executar um determinado artigo ou tipo de artigo. Além de cumprir uma verificação de privilégios gerais, a verificação de ACL tem de ser executada para o artigo ou tipo de artigo específico, dependendo da acção.

A verificação de ACL é baseada nas seguintes regras:

- Os privilégios que os utilizadores obtêm para a ACL X consistem na intersecção dos privilégios do conjunto de privilégios gerais do utilizador com o conjunto de privilégios na ACL.
- Se um utilizador participar numa ACL como utilizador individual e parte de um grupo, a regra de ACL do utilizador substituirá a regra de ACL do grupo do utilizador.
- Quando o parâmetro **PubAccessEnabled** está activado e uma ACL X contém uma regra de ACL para ICMPUBLIC, o conjunto de privilégios que o utilizador obtém na ACL X corresponde à UNIÃO de:
 - os privilégios atribuídos pelas regras de ACL do utilizador e do grupo do utilizador
 - os privilégios atribuídos pela regra de ACL a ICMPUBLIC
- Os utilizadores com o privilégio ItemSuperAccess podem ignorar a verificação de ACL. O utilizador será automaticamente adicionado a todas as ACLs, com todos os privilégios incluídos no conjunto de privilégios gerais do respectivo perfil de utilizador.

O parâmetro de configuração **ACL Binding Level (Nível de associação da ACL)** permite ao utilizador seleccionar a ACL utilizada para efectuar a verificação de privilégios em operações de CRUD. As opções de nível de associação são:

Nível de tipo de artigo

A ACL é retirada da definição da vista de tipo de artigo. Se não for especificada qualquer vista ou se o tipo de artigo for uma parte, a ACL é retirada da vista de tipo de artigo base (que é igual à ACL de tipo de artigo).

Nível de artigo

A ACL é retirada da ACL de artigo. A ACL do artigo é definida pela

aplicação durante a criação do artigo ou predefinida pela ACL de tipo de artigo ou pela ACL predefinida do utilizador, dependendo da definição do tipo de artigo.

Misto (Artigo ou tipo de artigo)

A ACL é retirada da ACL da vista de tipo de artigo ou a ACL de artigo, dependendo do sinalizador da ACL de nível de artigo definido durante a definição do tipo de artigo. O nível de associação da ACL corresponde à predefinição para o Content Manager Versão 8.

Nível de biblioteca

A ACL é retirada do código da ACL de biblioteca a partir do parâmetro de configuração do servidor de bibliotecas.

As seguintes “regras” descrevem a forma como a verificação de privilégios da ACL é executada com base no nível de associação da ACL e a classificação do tipo de artigo.

Regras de artigo

Nível de tipo de artigo

Verifique os privilégios utilizando a ACL da vista de tipo de artigo fornecida ou a vista base, se não tiver sido fornecida outra vista (ACLCODE da tabela ICMSTITVIEWDEFS).

Nível de artigo

Verifique o privilégio utilizando a ACL do artigo (ACLCODE da tabela ICMSTITEMS001001).

Nível misto

Verifique o sinalizador da ACL de nível de artigo da definição de tipo de artigo (ITEMLEVELACL da tabela ICMSTITEMTYPEDEFS) para determinar se a associação da ACL é efectuada ao nível do tipo de artigo ou ao nível do artigo. Para tipos de artigo de encaminhamento de documentos predefinidos pelo sistema (ROUTINGPROCESS, WORKNODE, WORKLIST), o sinalizador da ACL de nível de artigo está definido como true (a ACL é verificada ao nível do artigo).

Nível de biblioteca

Verifique o privilégio utilizando o código da ACL de biblioteca a partir da configuração do servidor de bibliotecas (LIBRARYACLCODE da tabela ICMSTSYSCONTROL).

Regras de partes

Nível de tipo de artigo

Verifique o privilégio utilizando a ACL da relação de tipo de artigo para a parte (DFLTACLCODE da tabela ICMSTITEMTYPEREL).

Nível de artigo

Verifique o privilégio utilizando a ACL do artigo (ACLCODE da tabela ICMSTITEMS001001).

Nível misto

Verifique o sinalizador da ACL de nível de artigo a partir da definição do tipo de artigo de parte (ITEMLEVELACL da tabela ICMSTITEMTYPEDEFS) para determinar se a associação da ACL é efectuada ao nível do tipo de artigo ou ao nível do artigo. Para tipos de partes predefinidos pelo sistema (ICMBASE, ICMBASETEXT, ICMBASESTREAM, ICMNOTELOG,

ICMANNOTATION), o sinalizador da ACL de nível de artigo está definido como false (a ACL é verificada ao nível do tipo de artigo).

Nível de biblioteca

Verifique o privilégio utilizando o código da ACL de biblioteca a partir da configuração do servidor de bibliotecas (LIBRARYACLCODE da tabela ICMSTSYSCONTROL).

Configuração de ACLs para o modelo de documentos

No tempo de execução, o servidor de bibliotecas verifica sempre os privilégios do utilizador para a ACL.

Cada artigo e vista de tipo de artigo tem uma ACL associada. A ACL que deve ser utilizada a verificação de controlo de acessos depende da associação de ACL. No tempo de execução, o servidor de bibliotecas:

- Verifica a ACL ao nível do tipo de artigo ao criar um documento ou ao obter informações de tipo de artigo. Se aceder a uma vista de tipo de artigo, o servidor de bibliotecas verifica a ACL da vista de tipo de artigo. No entanto, para obter partes definidas pelo sistema, as informações de tipo de artigo não requerem qualquer verificação de ACL ou de privilégios. Para tipos de partes definidos pelo utilizador, o servidor de bibliotecas tem de verificar a ACL de tipo de parte.
- Obtém a ACL com base na associação de ACL para verificar a ACL e os privilégios quando o utilizador está a pesquisar ou a obter um documento, a actualizar um documento com partes e para todas as outras funções de tempo de execução. Pode definir a associação de ACL e a ACL predefinida na janela ACL de uma definição de tipo de artigo a partir do cliente de administração do sistema.

A verificação de ACL para o documento e para as partes é separada e independente. Por conseguinte, um utilizador poderá actualizar o documento ou as partes de recurso com base na ACL ou no privilégio.

A cada vista é atribuída uma ACL para um tipo de artigo. A verificação de ACL para partes de documento é independente das vistas de documentos. Não existe qualquer vista para partes de documentos. No entanto, é possível criar vistas em tipos de partes definidos pelo utilizador.

A ACL de nível de tipo de artigo que o servidor de bibliotecas utiliza para o documento é definida na janela Access Control (Controlo de acessos). Para a vista base e para as outras vistas de tipo de artigo, a ACL é definida na janela Item Type Subset (Subconjunto de tipo de artigo) do cliente de administração do sistema.

A ACL é especificada na relação de tipo de artigo para cada tipo de parte quando o tipo de artigo do documento é criado. A ACL de nível de tipo de artigo utilizada pelo servidor de bibliotecas para as partes de recurso têm origem na janela Document Management (Gestão de documentos) no cliente de administração do sistema.

Planear uma aplicação do DB2 Content Manager

Esta secção ajuda-o a identificar os requisitos para a criação de uma aplicação do DB2 Content Manager e fornece informações sobre a forma de operar do DB2 Content Manager.

Uma parte essencial do planeamento da sua aplicação está em criar um modelo de dados que corresponde às necessidades da sua empresa. Para obter mais informações sobre o modelo de dados do DB2 Content Manager, consulte o manual *Planning and Installing Your Content Management System* e o exemplo `tsItemCreationICM` no directório `samples`.

Determinar as funções da aplicação

As funções da aplicação dependem das necessidades da organização.

Para produzir uma aplicação efectiva, todas as partes interessadas na sua organização devem contribuir para o planeamento e concepção da aplicação.

Antes de poder criar a aplicação, deverá poder responder a todas ou a maior parte das seguintes questões:

- Quais os tipos de documentos que a sua organização utiliza?
- Que tipos de conteúdo têm os documentos existentes?
- Como processa documentos?
- Pode automatizar o seu processamento de documentos?
- Como recebe, apresenta, armazena e distribui os documentos?
- Com que frequência obtém os documentos após serem armazenados?
- Qual é o volume de documentos que a sua organização gere?
- Quais os tipos de suporte de memória que pretende utilizar para armazenar os seus objectos grandes?
- Existem outras aplicações que a sua organização utilize?
- Quantos utilizadores e qual o tipo de controlo de acessos que pretende utilizar?

Utilize as respostas para as perguntas colocadas acima para o ajudar a determinar quais as funções a incluir na sua aplicação.

Tratamento de erros

Ao efectuar o tratamento de erros, a excepção mais importante a deter é a classe `DKException`.

Não utilize excepções para lógicas de programa e não tenha em conta excepções de detenção para detectar se existe algo no servidor de conteúdos ou noutra situação que não seja pontual. Utilizar excepções no lógico do programa diminui o rendimento e pode inutilizar as informações de rastreio e registo para a depuração e suporte.

Reveja cuidadosamente todas as informações da excepção. Existem várias subclasses de `DKException` e, dependendo do programa, o método mais indicado será o tratamento de cada excepção individualmente. A Tabela 6 contém informações de `DKException`.

Tabela 6. Informações de `DKException`

<code>DKException</code>	Descrição
Nome	Nome da Classe de Excepção. Contém o nome da subclasse.
Mensagem	Uma mensagem específica explica o erro. A mensagem pode conter uma grande quantidade de informações, encapsulando, por vezes, estados importantes de variáveis na altura em que o erro foi detectado.

Tabela 6. Informações de *DKException* (continuação)

DKException	Descrição
ID da Mensagem	Um ID de Mensagem único identifica este tipo de erro e estabelece uma correspondência entre este e uma mensagem de núcleo utilizada acima.
Estado do Erro	Poderá conter informações adicionais relativas ao estado da API de OO ou ao erro de servidor de bibliotecas. Se o servidor de bibliotecas detectar um erro, as seguintes quatro informações são agrupadas aqui: Código de retorno Código da razão Código de retorno de Ext / SQL Código da razão de Ext / SQL
Código do Erro	Poderá conter o código de retorno do servidor de bibliotecas.
Rastreo da Pilha de Memória	Informações importantes que indicam o ponto da falha no programa do utilizador e o local exacto onde o erro foi detectado ou tratado pela última vez pela API de OO.

Ao trabalhar em Java, também deve tratar `java.lang.Exception`. O exemplo `SConnectDisconnectICM` do directório `samples` demonstra como apanhar e imprimir erros. Para obter informações relativas ao registo e rastreio, consulte `Messages and Codes`.

Activar o suporte de pilha dupla IPv6

O DB2 Content Manager Versão 8.4 faculta suporte de pilha dupla de Internet Protocol Versão 6 (IPv6) juntamente com o Internet Protocol Versão 4 (IPv4).

O IPv6 é o protocolo da próxima geração, criado pela IETF (Internet Engineering Task Force) para substituir o protocolo actual, IPv4. O IPv6 elimina a escassez de endereços IPv4 necessários para todas as máquinas ligadas à Internet. O IPv6 melhora ainda a configuração automática do encaminhamento e da rede do IPv4.

O IPv4 possui endereços de 32 bits. O IPv6 faculta um número quase ilimitado de endereços. Os endereços IPv6 são de 128 bits. Além de resolver o problema da escassez de endereços de rede, o IPv6 faculta ainda:

- Configuração automática
- Renumeração de sítios
- Segurança de IP extremidade a extremidade
- Mobilidade com optimização de encaminhamento

Importante: O DB2 Content Manager Versão 8.4 fornece suporte de IPv6 dual-stack para conectores ICM, apenas para APIs Java e C++. Por exemplo, os conectores Fed não são suportados pelo IPv6.

Importante: As directrizes de utilização de endereços IPv6 com cadeias de URL são:

- Numa cadeia de URL, o endereço IPv6 encontra-se entre parênteses. Se estiver a utilizar um nome do sistema central, não são necessários parênteses embora o sistema central tenha um endereço IPv6. Por exemplo, o URL que utiliza o endereço IPv4 9.67.122.66 is `http://9.67.122.66:8080/index.html`. Se estiver a utilizar um nome

do sistema central, como myhost.company.com, o URL é `http://myhost.company.com:8080/index.html`. Contudo, o URL que utiliza o endereço IPv6 `2001:0DB8:4545:2::09FF:FEF7:62DC` é `http://[2001:0DB8:4545:2::09FF:FEF7:62DC]:8080/index.html`

- Se estiver a utilizar o endereço IPv6 na cadeia de URL DB2 JDBC, o endereço IPv6 terá de estar entre parênteses rectos. Por exemplo, a cadeia de ligação tem de estar no formato:

`JDBCURL=jdbc:db2://[2007::9:181:141:150]:50000/icmn1sdb`

Para activar o suporte de IPv6, o Java (1.4 ou posterior) faculta duas propriedades de sistema:

java.net.preferIPv4Stack

Se uma aplicação tiver como preferência a utilização de sockets IPv4 apenas, defina esta propriedade como true. Se esta propriedade for definida como true, a aplicação não comunica com sistemas centrais IPv6.

java.net.preferIPv6Addresses

Defina esta propriedade para alterar a preferência para utilização de endereços IPv6 em vez de endereços IPv4.

Durante a execução de uma aplicação Java através da utilização de um conector ICM, defina as propriedades `java.net.preferIPv4Stack` e `java.net.preferIPv6Addresses` de forma a que a aplicação Java comunique com sistemas centrais de pilha dupla de IPv4, de IPv6 ou de ambos.

A tabela que se segue resume os resultados de várias combinações de propriedades `java.net.preferIPv4Stack` e `java.net.preferIPv6Addresses` ao executar uma classe de Java num nó de pilha dupla. Exemplo: `java -Djava.net.preferIPv4Stack=false -Djava.net.preferIPv6Addresses=true <class name>`

Tabela 7. Preferências para utilização de IPv4 ou IPv6

<code>java.net.preferIPv4Stack</code>	<code>java.net.preferIPv6Addresses</code>	endereço predefinido
true	true	IPv4
true	false	IPv4
false	false	IPv4
false	true	IPv6

Controlador do IBM DB2 para suporte de JDBC e SQLJ no DB2 Content Manager Versão 8.4

O DB2 Content Manager Versão 8.4 suporta o Controlador IBM® DB2® para JDBC e SQLJ (tipo 2 e tipo 4). Este controlador é, por vezes, denominado como controlador JDBC de JCC. Esta alteração de controlador afecta o conector do CM 8 (ICM) e o conector federado.

A base de dados DB2 tornou obsoleto o controlador JDBC requerido pelo DB2 Content Manager para a Versão 8.3. O DB2 Content Manager Versão 8.4 já não suporta o Controlador JDBC Tipo 2 da DB2 (`com.ibm.db2.jdbc.app.DB2Driver`).

Na Versão 8.4, o DB2 Content Manager suporta apenas o Controlador IBM DB2 para JDBC e SQLJ (tipo 2 e tipo 4).

O controlador de tipo 4 é um controlador de Java puro, o que significa que a aplicação requer apenas ficheiros JAR e não requer o cliente de tempo de execução de DB2. Contudo, neste modo, quando a aplicação do DB2 Content Manager utiliza o controlador de JDBC, é necessário especificar explicitamente o nome do sistema central e a porta da base de dados à qual está a ser estabelecida ligação. A especificação do nome do sistema central e da porta elimina a necessidade de utilização do catálogo de bases de dados. No entanto, se o nome do sistema central, a porta ou a base de dados não forem especificados no ficheiro `cmbicmsrvs.ini` para o conector ICM e no ficheiro `cmbds.ini` para o conector federado, será utilizada a ligação de tipo 2, por predefinição. A ligação de tipo 2 requer o cliente de tempo de execução de DB2 e uma base de dados catalogada no sistema local.

Importante: Se o nome do sistema central, a porta e o nome da base de dados remota forem especificados mas os valores estiverem incorrectos, ocorrerá um erro quando o DB2 Content Manager tentar efectuar uma ligação de tipo 4. Para corrigir este erro, é necessário actualizar o ficheiro INI para incluir as informações correctas.

Importante: Se efectuar a actualização para o DB2 Content Manager Versão 8.4, pode actualizar o nome do sistema central, a porta e a base de dados remota no ficheiro `cmbicmsrvs.ini` para o conector ICM e no ficheiro `cmbds.ini` para o conector federado para utilizar o controlador de tipo 4.

Os ficheiros JAR necessários para o Controlador IBM DB2 para JDBC e SQLJ (tipo 2 e tipo 4) são:

- `db2jcc.jar`
- `db2jcc_license_cu.jar`

Estes ficheiros JAR são fornecidos com o DB2 Content Manager. É necessário um ficheiro adicional JAR, `db2jcc_license_cisuz.jar`, para estabelecer ligação a servidores zSeries. O ficheiro `db2jcc_license_cisuz.jar` é também fornecido com o DB2 Content Manager.

Importante: O ficheiro anterior (`db2java.zip`) já não é utilizado e pode ser removido do caminho de classe de quaisquer scripts.

Modificações aos ficheiros `cmbicmsrvs.ini` e `cmbds.ini`

O ficheiro `cmbicmsrvs.ini` existente para o conector ICM e o ficheiro `cmbds.ini` para o conector federado suporta, o Controlador IBM DB2 para JDBC e SQLJ.

O DB2 Content Manager faculta ainda a capacidade de substituir os valores do nome do sistema central, da porta e do nome da base de dados remota, especificando o controlador de JDBC e o URL de URL directamente. Para especificar o controlador de JDBC e o URL de JDBC directamente, o DB2 Content Manager Versão 8.4 faculta duas propriedades, `ICMJDBCDRIVER` e `ICMJDBCURL`, no ficheiro `cmbicmsrvs.ini` para o conector ICM e no ficheiro `cmbds.ini` para o conector federado. Pode igualmente fornecer as propriedades `JDBCURL` e `JBCDRIVER` utilizando o parâmetro **connect_string** associado ao método `DKDatastoreICM::connect`.

No DB2 Content Manager, estas propriedades são opcionais e não possuem quaisquer valores. Pode especificar os valores destas propriedades optando por uma das seguintes formas:

- Se a propriedade ICMJDBC DRIVER não for especificada ou estiver vazia, é utilizado o Controlador IBM DB2 para JDBC e SQLJ (tipo 2 e tipo 4), por predefinição. O novo Controlador IBM DB2 para JDBC e SQLJ (tipo 2 e tipo 4) é carregado através da classe `com.ibm.db2.jcc.DB2Driver`.
- Se a propriedade ICMJDBCURL, bem como as informações ICMHOST, ICMPORT e OCMREMOTEDB, forem especificadas no ficheiro `cmbicmsrvs.ini` e no ficheiro `cmbds.ini`, o valor associado à propriedade ICMJDBCURL toma precedência e é utilizado como o URL para estabelecer ligação à base de dados.
- Se a propriedade JDBCURL for especificada através da utilização do parâmetro **connect_string** do método `connect()` e a propriedade ICMJDBCURL for especificada no ficheiro `cmbicmsrvs.ini` e no ficheiro `cmbds.ini`, a propriedade JDBCURL no ficheiro **connect_string** tem precedência. Da mesma forma, a propriedade JDBC DRIVER transferida através do parâmetro **connect_string** tem precedência sobre a propriedade ICMJDBC DRIVER no ficheiro `cmbicmsrvs.ini` e no ficheiro `cmbds.ini`.

A definição do URL e do controlador em simultâneo não é um requisito no DB2 Content Manager Versão 8.4. Poderá especificar o URL ou o controlador.

Para especificar um controlador diferente do Controlador IBM DB2 para JDBC e SQLJ, os valores para as propriedades ICMJDBC DRIVER e ICMJDBCURL têm de ser definidos manualmente nos ficheiros `cmbicmsrvs.ini` e `cmbds.ini`. Na Versão 8.4, o DB2 Content Manager suporta apenas o Controlador IBM DB2 para JDBC e SQLJ ao estabelecer ligação a uma base de dados DB2. O DB2 Content Manager não suporta qualquer outro controlador para a base de dados DB2.

O utilizador pode definir os valores para as propriedades ICMJDBC DRIVER e ICMJDBCURL no ficheiro INI, conforme indicado em seguida:

```
ICMJDBC DRIVER=com.ibm.db2.jcc.DB2Driver
ICMJDBCURL=jdbc:db2://myhostname.ibm.com:50000/icm1sdb
```

Não se esqueça:

Quando um endereço de IP é utilizado em vez de um nome do sistema central no URL de JDBC, [e] são necessários ao endereço se for utilizado IPv6. Por exemplo, a cadeia de ligação de um endereço IPv6 2007::9:181:141:150 tem de estar entre [e] no seguinte formato:

```
JDBCURL=jdbc:db2://[2007::9:181:141:150]:50000/icm1sdb
```

Importante: As propriedades do ficheiro INI e o parâmetro **connect_string** aplicam-se a DB2..

Informações relacionadas



Controladores suportados para JDBC e SQLJ

Trabalhar com exemplos do DB2 Content Manager

O DB2 Content Manager fornece um conjunto abrangente de exemplos de código para ajudar a executar tarefas essenciais do DB2 Content Manager.

Os exemplos são uma ótima fonte de instrução de API ICM, pois facultam informações de consulta, orientações de programação, exemplos de utilização de API e ferramentas.

Pode visualizar os exemplos no manual *Application Programming Reference*, no Information Center do produto. Além disso, os exemplo encontram-se nos directórios *IBMCMROOT/samples/cpp/icm* e *IBMCMROOT/samples/java/icm*. No entanto, tenha em conta que deve seleccionar o componente Exemplos e Ferramentas durante a instalação do DB2 Information Integrator for Content de modo a dispor dos exemplos no directório.

Nota: Os exemplos do DB2 Content Manager Express encontram-se no seguinte directório: `\samples\java\cmx` e `\samples\cpp\cmx`

Para tirar o maior partido possível dos exemplos leia o ficheiro Readme de Exemplos. Contém um índice completo de consulta para o ajudar a encontrar rapidamente o exemplo que contém o conceito, ou tópico, que procura. Todos os exemplos são documentados com precisão e facultam informações conceptuais aprofundadas e uma explicação de todos os passos de uma tarefa. As informações adicionais contidas em cada exemplo incluem:

- Informações de cabeçalho detalhadas que explicam os conceitos apresentados no exemplo.
- Uma descrição do ficheiro de exemplos incluindo informações de pré-requisitos e utilização de linhas de comandos.
- Um código totalmente comentado que facilmente pode ser eliminado, personalizado e utilizado nas suas aplicações.
- Uma função de utilitário que pode utilizar ao desenvolver as suas aplicações.

A secção *Introdução* no ficheiro Readme de Exemplos ajudará a compreender rapidamente como executar as seguintes tarefas gerais:

- Modelação de dados.
- Estabelecer ligação a um servidor e tratar erros.
- Definir atributos e grupos de atributos.
- Trabalhar com atributos de referência.
- Definir o seu modelo de dados.
- Trabalhar com artigos.
- Trabalhar com artigos de recurso.
- Trabalhar com pastas.
- Trabalhar com ligações.
- Definir uma recolha SMS.
- Pesquisar artigos.

O exemplo da companhia de seguros

O DB2 Content Manager faculta exemplos de códigos para uma possível implementação prática em que se recorre a uma seguradora. As informações utilizadas para criar o exemplo de seguradora são concebidas e criadas com o único intuito de ajudar a clarificar as funções essenciais do Content Manager. Para obter uma lista completa dos exemplos que compõem o cenário de seguro, consulte o Readme de Exemplos.

Criar uma aplicação do DB2 Content Manager

As APIs que implementam a funcionalidade do DB2 Content Manager Versão 8.4 estão agrupadas no que é denominado conector do ICM.

As APIs do conector de ICM possuem um sufixo de ICM, como no exemplo DKDatastoreICM.

Compreender os componentes do software

Pode categorizar as APIs OO nos seguintes grupos de serviços: modelação de dados e documentos, pesquisa e obtenção, importação e apresentação de dados, gestão do sistema e encaminhamento de documentos.

O módulo de elaboração de modelos de dados e de documentos contém as APIs que permitem correlacionar os seus modelos de dados empresariais para o modelo de dados hierárquico do DB2 Content Manager subjacente. Por exemplo, o modelo de dados de uma companhia de seguros inclui apólices, que, no modelo de dados do DB2 Content Manager correspondem, essencialmente, a artigos. As APIs do módulo de modelação de dados e documentos facultam interfaces para definir artigos que representam apólices.

O módulo de pesquisa e obtenção processa pedidos relativos a artigos geridos como, por exemplo, documentos e pastas. As APIs do módulo de pesquisa permite-lhe executar pesquisas de texto e paramétricas combinadas para artigos que constam no sistema DB2 Content Manager. Os resultados da pesquisa são devolvidos à aplicação na forma de conjuntos de resultados da pesquisa.

O módulo de importação e entrega de dados faculta as APIs que permitem ao utilizador importar dados para o sistema e entregar esses dados através de vários meios de transmissão como, por exemplo, uma rede ou a Web.

O módulo de gestão do sistema fornece-lhe as interfaces para configurar e manter um sistema do DB2 Content Manager eficiente e protegido. Por exemplo, pode incorporar as APIs de gestão do sistema na sua aplicação de forma a poder ajustar as definições de controlo do sistema, gerir utilizadores, atribuir privilégios de utilizador, permitir acesso ao sistema e assim por diante.

As APIs do módulo de encaminhamento de documentos ajudam o utilizador a encaminhar objectos empresariais como, por exemplo, documentos, através de um processo, segundo as necessidades definidas pela empresa do utilizador.

Representar itens através de DDOs

Antes de criar uma aplicação, é necessário compreender os conceitos do protocolo de DDO/XDO

Um DDO é essencialmente um contentor de atributos. Um atributo tem um nome, valor e várias propriedades. Uma das propriedades mais importantes dos atributos é o tipo de atributo. Um DDO possui um identificador persistente (PID) para indicar onde o objecto reside no arquivo persistente. Um DDO tem alguns métodos para se multiplicar e métodos correspondentes para obter as informações de um artigo. Os métodos de DDO incluem adicionar, obter, actualizar e eliminar. O utilizador recorre a estes métodos para colocar e retirar os dados de um artigo do arquivo persistente como, por exemplo, DB2 Content Manager.

Na memória, os artigos do DB2 Content Manager são representados como DDOs. Os atributos do artigo são representados como atributos do DDO com um nome, tipo e valor. As ligações e referências são representadas como tipos especiais de atributos. No entanto, a diferença entre um atributo de ligação e um atributo de referência é que o atributo de referência remete para outro (único) DDO ou XDO, e um atributo de ligação remete para uma recolha (vários) de DDOs ou XDOs. Os XDOs são utilizados para representar objectos volumosos (LOBs).

Uma referência a um artigo, quer seja XDO ou outro DDO, tem um nome com a propriedade tipo definida como referência do objecto e o valor definido como a instância do objecto referenciado. Os componentes descendentes e as ligações são também representados como atributos do DDO com a propriedade tipo definida como uma recolha de objectos de dados e o valor definido como uma recolha de DDOs. No caso de um componente descendente, o nome do atributo é o nome do componente descendente. O valor é a recolha de componentes descendentes que pertencem ao componente raiz. Se o artigo raiz for eliminado, todos os componentes de raiz do artigo raiz são também eliminados.

Estabelecer ligação ao sistema do DB2 Content Manager

Uma das primeiras coisas que o utilizador deverá efectuar quando constrói uma aplicação do DB2 Content Manager é estabelecer ligação ao servidor.

Esta secção auxilia o utilizador na execução de várias tarefas incluídas no estabelecimento e na conclusão da ligação a um servidor de DB2 Content Manager.

Dependendo da configuração do sistema, poderá ter vários servidores de bibliotecas e gestores de recursos aos quais pode estabelecer ligação. Para visualizar uma lista dos nomes dos servidores de bibliotecas aos quais pode estabelecer ligação, utilize o DKDatastoreICM e invoque o método listDataSourceNames() e, em seguida, o método listDataSources(). O método listDataSources() cria uma lista dos servidores de bibliotecas actualmente disponíveis para ligação.

Para aceder a um servidor do DB2 Content Manager, a sua aplicação deve criar um servidor de conteúdos, que funciona como um servidor comum. Para criar e estabelecer ligação a um servidor de conteúdos:

1. Crie um objecto de servidor de conteúdos.

Java

```
DKDatastoreICM dsICM = new DKDatastoreICM();
```

C++

```
DKDatastoreICM *dsICM =new DKDatastoreICM();
```

2. Configure os parâmetros de ligação.

Java

```
String database = "icm1sdb";  
String userName = "icmadmin";  
String password = "password";
```

C++

```
char * database = "icm1sdb";  
char * userName = "icmadmin";  
char * password ="password";
```

3. Invoque a operação connect no servidor de conteúdos. databaseNameStr é o nome da base de dados à qual pretende estabelecer ligação.

Java

```
dsICM.connect(databaseNameStr,usridStr,pwStr,"");
```

C++

```
dsICM->connect(database, userName, password, "");
```

Após estabelecer ligação a um servidor de bibliotecas, utilize DKRMConfiguration e invoque o método listResourceMgrs(), para obter uma lista de gestores de recursos associados a esse servidor de bibliotecas.

Para anular a ligação ao sistema, invoque a operação disconnect no servidor de conteúdos.

Para obter mais informações, consulte o SConnectDisconnectICM, o exemplo completo do qual foram retirados os fragmentos de código acima mencionados.

Alterar uma palavra-passe

Pode permitir aos utilizadores a alteração da palavra-passe sempre que iniciem uma nova sessão do servidor de bibliotecas.

Para implementar a opção de alteração da palavra-passe, utilize DKDatastoreICM e invoque o método changePassword() .

Java

```
changePassword(String userID, String oldPwd, String newPwd)
```

C++

```
changePassword(const char* userID, const char* oldPwd, const char* newPwd)
```

Trabalhar com artigos

Esta secção descreve como criar, actualizar e eliminar artigos. Para obter mais informações sobre o trabalho com estes artigos, consulte o exemplo de SItemCreationICM.

Criar um tipo de artigo

Antes de criar quaisquer artigos, deve já ter criado tipos de artigo, muito semelhantes a categorias, para colocar os artigos.

Por exemplo, um artigo de participação pode ser colocado sob o tipo de artigo apólice. Esta relação pode ser equiparada a uma relação ascendente-descendente, na qual o descendente é o artigo, ou a participação, e o ascendente é o tipo de artigo, ou a apólice.

Quando cria um tipo de artigo, pode definir uma classificação para o tipo de artigo. Uma *classificação de tipo de artigo* é a categorização dentro de um tipo de artigo que aprofunda a identificação dos artigos desse tipo de artigo. Todos os artigos do mesmo tipo de artigo possuem a mesma classificação de tipo de artigo. O Content Manager fornece as seguintes classificações de tipo de artigo: artigo, artigo de recurso docmodel e docpart. Se não especificar uma classificação de tipo de artigo quando cria um tipo de artigo, a classificação do tipo de artigo assume a predefinição do artigo (DDO). O Tabela 8 na página 90 contém as classificações de tipo de artigo predefinidas e as suas constantes de ID correspondente.

Tabela 8. Classificações de tipo de artigo

Classificação	Constante de ID	Número	Descrição
Artigo	DK_ICM_ITEMTY PE_CLASS_ITEM	0	Um artigo padrão (DDO).
Recurso	DK_ICM_ITEMTY PE_CLASS_RES OURCE_ITEM	1	Um artigo de recurso que descreve e contém dados que se encontram armazenados no gestor de recursos. Os dados de vídeo, imagens, documento e outros dados arquivados no gestor de recursos constituem exemplos de artigos de recurso.
Modelo de documento	DK_ICM_ITEMT YPE_CLASS_DOC _MODEL	2	Um artigo que modela documentos utilizando partes. Um documento é composto por um determinado número de partes, que se encontram contidas no atributo <code>DKConstant.DK_CM_DKPARTS</code> .
Parte	DK_ICM_ITEMTY PE_CLASS_DOC_ PART	3	Os artigos (partes) na classificação do modelo de documento.

Notas: Nota: As constantes encontram-se em `com\ibm\mm\sdk\common\DKConstantICM.java` para Java e em `dk\icm\DKConstantICM.h` para C++.

Para criar um tipo de artigo (consulte o exemplo de código em baixo):

1. Criar um tipo de artigo e transmita-o como referência ao servidor de conteúdos.
2. Atribua um nome ao tipo de artigo. Os nome de tipo de artigo têm de ter menos do que 15 caracteres de comprimento. O nome do tipo de artigo é utilizado para criar DDOs do tipo de artigo.
3. Adicione atributos ao tipo de artigo e defina os qualificadores pretendidos, como, por exemplo, nullable, textsearchable e unique.
4. Adicione o tipo de artigo ao servidor de conteúdos persistente.

Exemplos

Java

```
//This example creates an item type definition and names it.
//The item type name must be less than 15 characters in length.
DKItemTypeDefICM bookItemType = new DKItemTypeDefICM(dsICM);
    bookItemType.setName("book");
    bookItemType.setDescription("This is an example item type name.");
//Below, a text-searchable attribute called book title is added. The
//attribute is defined to require a unique name and also a value. The value
//does not have to be unique.
DKAttrDefICM attr = (DKAttrDefICM)_dsDefICM.retrieveAttr("book_title");
    attr.setTextSearchable(true);
    attr.setUnique(true);
    attr.setNullable(false);
    bookItemType.add(attr);
//Here, a book_num_pages attribute is added to the book item
//type with the following characteristics: text searchable, unique,
//and a value is not required.
DKAttrDefICM attr = (DKAttrDefICM)_dsDefICM.retrieveAttr("book_num_pages");
    attr.setTextSearchable(false);
```



```

attr.setUnique(false);
attr.setNullable(false);
    bookItemType.addAttr(attr);
    bookItemType.add();

```

C++

```

DKDatastoreICM* pDs;
DKDatastoreICM* pDs;

...
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*)pDs->datastoreDef();
//create new ItemType
DKItemTypeDefICM * bookItemType = new DKItemTypeDefICM(pDs);
bookItemType->setName("book");
bookItemType->setDescription("This is an example item type name.");
//Create new Attribute; add it to datastore and to the ItemType
DKAttrDefICM * attr = (DKAttrDefICM *)dsDefICM->createAttr();
    attr->setName("book_title");
    attr->setType(DK_CM_VARCHAR);
    attr->setSize(80);
    attr->setTextSearchable(TRUE);
    attr->setUnique(TRUE);
    attr->setNullable(FALSE);
//Persist the attribute to the datastore
attr->add();
//Add the newly created attribute to the item type.
bookItemType->addAttr(attr);
//Create new Attribute; add it to datastore and to ItemType
attr = (DKAttrDefICM *)dsDefICM->createAttr();
    attr->setName("book_num_pages");
    attr->setType(DK_CM_INTEGER);
    attr->setTextSearchable(FALSE);
    attr->setUnique(FALSE);
    attr->setNullable(FALSE);
    attr->add();
bookItemType->addAttr(attr);
//Add the entity, bookItemType, to the datastore.
bookItemType->add();

```

Consulte o exemplo SItemTypeCreationICM para obter mais informações.

Enumerar tipos de itens

Para enumerar tipos de itens, terá primeiro de estabelecer ligação a um servidor de conteúdos.

Para obter uma lista de tipos de artigo definidos:

1. Estabeleça ligação ao servidor de conteúdos DKDatastoreICM.
2. Obtenha uma referência à definição do servidor de conteúdos.
3. Invoque o método listEntityNames no objecto de definição do servidor de conteúdos para obter uma matriz de cadeias dos nomes dos tipos de artigo.
4. Utilize um ciclo para enumerar todos os nomes.

Exemplos

Java

```

String itemTypeName[] = dsICM.listEntityNames();
DKSequentialCollection itemTypeColl =
    (DKSequentialCollection) dsICM.listEntities();
dkIterator iter = itemTypeColl.createIterator();
while(iter.more()){
    dkEntityDef itemType = (dkEntityDef) iter.next();
    System.out.println(" Item type name : " + itemType.getName());
}

```


C++ Exemplo 1

```
long larraySize = 0;
DKString * itemTypeNames = dsICM->listEntityNames(larraySize);
for (int i = 0; i < larraySize; i++) {
    cout <<(char*)itemTypeNames[i] <<endl;
}
delete [] itemTypeNames;
```

C++ Exemplo 2

```
DKSequentialCollection * itemTypeColl =
    (DKSequentialCollection *)
        dsICM->listEntities();
dkIterator * iter = itemTypeColl->createIterator();
while (iter->more()) {
    dkEntityDef* itemType=(dkEntityDef*)((void*)(*iter->next()));
    cout <<(char*)itemType->getName() < delete(itemType);
}
delete(iter);
delete(itemTypeColl);
```

Para obter mais informações, consulte o exemplo SItemTypeInfoRetrievalICM.

Limitações da actualização de tipos de artigo

Existem algumas limitações na actualização de tipos de artigo.

Segue-se a lista de limitações à actualização de tipos de artigo:

- Não é possível actualizar o nome de um tipo de artigo. Por exemplo, se definir um tipo de artigo com o nome Book, não poderá actualizar o nome deste tipo de artigo para BookNew.
- Não é possível alterar o ID de identificação de um tipo de artigo. Por exemplo, se definir um tipo de artigo Book e o respectivo ID for 1000, não pode actualizar o ID de Book para 1002.
- Não é possível actualizar a classificação de um tipo de artigo. Por exemplo, se definir um tipo de artigo Book e a respectiva classificação for DK_ICM_ITEMTYPE_CLASS_RESOURCE_ITEM, não pode actualizá-la para DK_ICM_ITEMTYPE_CLASS_DOC_MODEL.
- O tipo de artigo de parte de documento não pode ser activado para ligação automática durante a criação nem durante a actualização.
- Não é possível remover atributos ou grupos de atributos.
- Não é possível alterar atributos de requerido para não requerido ou de não requerido para requerido.
- Não é possível alterar atributos de exclusivo para não exclusivo e de não exclusivo para exclusivo.
- Não é possível alterar a descrição de um atributo. Por exemplo, se definir um atributo Address e a respectiva descrição for "This is an address attribute", não poderá actualizar a descrição do atributo para "This is a new address attribute".
- Não é possível alterar uma regra de eliminação de tipos de componente.
- Não é possível alterar o campo pesquisável por texto de true para false.
- No campo pesquisável por texto, não é possível alterar o valor de IndexCCSID e IndexLangCode. Por exemplo, se definir o valor IndexCCSID como 806 e IndexLangCode como EN_US, não pode alterar o valor IndexCCSID para 852 e indexLangCode para EN_AU.
- Não é possível alterar a classificação do XDO de tipo de artigo. Por exemplo, se definir um tipo de artigo Book, Book será um tipo de artigo de recurso e o respectivo ID de classificação de XDO será DK_ICM_XDO_LOB_CLASS_ID. Não poderá alterá-lo para DK_ICM_XDO_TEXT_CLASS_ID.

- Não é possível alterar o comprimento de atributo de VARCHAR, CHAR, BLOB e CLOB. O número total e de casas fixas para o comprimento decimal não pode ser alterado.
- Não é possível remover um tipo de componente descendente se este já tiver dados carregados.

Criar atributos

Os atributo no DB2 Content Manager são criados como objectos independentes e têm propriedades genéricas.

Quando cria um tipo de artigo, determina quais os atributos a incluir no tipo de artigo. Depois do atributo ter sido incluído no tipo de artigo, pode continuar a definir propriedades de atributos para esse tipo de artigo. Por exemplo, pode definir o atributo ISBN no tipo de artigo Journal como anulável e, no tipo de artigo Book, como não anulável. Para obter mais informações sobre a criação de atributos, consulte o exemplo SItemTypeCreationICM.

Para criar um atributo, efectue os seguintes passos:

1. Crie um objecto de definição de atributos utilizando a classe DKAttrDefICM.
2. Descreva o objecto que criou, definindo o seu nome, descrição, tipo, tamanho, etc. Os nomes de atributos têm um limite de 15 caracteres. Se precisa de incluir mais informações, utilize o campo descrição. A tabela seguinte contém exemplos de tipos de atributo que pode criar:

Tabela 9. Tipos de atributos

Tipo	Constante	Objecto
BLOB	DKConstant.DK_CM_BLOB	byte bytes[]
Carácter	DKConstant.DK_CM_CHAR	java.lang.String
CLOB	DKConstant.DK_CM_CLOB	java.lang.String
Data	DKConstant.DK_CM_DATE	java.sql.Date
Decimal	DKConstant.DK_CM_DECIMAL	java.math.BigDecimal
Duplo	DKConstant.DK_CM_DOUBLE	java.lang.Double
Número inteiro	DKConstant.DK_CM_INTEGER	java.lang.Integer
Curto	DKConstant.DK_CM_SHORT	java.lang.Short
Hora	DKConstant.DK_CM_TIME	java.sql.Time
Marca de hora	DKConstant.DK_CM_TIMESTAMP	java.sql.Timestamp
Varchar	DKConstant.DK_CM_VARCHAR	java.lang.String

3. Adicione a nova definição de atributo ao servidor de conteúdos persistente.

Exemplos

Java

```
//This example defines an attribute for the title of a book.
attr = new DKAttrDefICM(dsICM);
attr.setName("book_title");
attr.setDescription("The title of the book.");
attr.setType(DKConstant.DK_CM_VARCHAR);
attr.setSize(100);
attr.add();
//This example defines an attribute for the number of pages in a book.
attr = new DKAttrDefICM(dsICM);
attr.setName("book_num_pages");
attr.setDescription("The number of pages in the book.");
```

```

attr.setType(DKConstant.DK_CM_INTEGER;
    attr.setMin((short) 0);
    attr.setMax((short) 100000);
    attr.add();

```

C++

```

//This example defines an attribute for the title of a book.
DKDatastoreICM * dsICM; .....
DKAttrDefICM * attr = new DKAttrDefICM(dsICM);
    attr->setName("book_title");
    attr->setDescription("The title of the book.");
    attr->setType(DK_CM_VARCHAR);
    attr->setSize((long) 100);
    attr->add();
//This example defines an attribute for the number of pages in a book.
DKAttrDefICM * attr = new DKAttrDefICM(dsICM);
    attr->setName("book_num_pages");
    attr->setDescription("The number of pages in the book.");
    attr->setType(DK_CM_INTEGER);
    attr->setMin((long) 0);
    attr->setMax((long) 100000);
    attr->add();

```

Para obter mais informações relativas à criação de atributos, consulte o exemplo `SAttributeDefinitionCreationICM`.

Criar, actualizar e eliminar grupos de atributos

Os grupos de atributos tornam mais fácil a tarefa de adicionar todo um grupo de atributos a artigos e subcomponentes.

Um atributo pode pertencer a qualquer número de grupo de atributos, desde zero até qualquer número. Um atributo pode ser adicionado a grupos de atributos múltiplos. Um artigo de dados (DDO) pode conter grupos de atributos múltiplos. O mesmo nome de atributo pode aparecer no DDO, mas cada atributo é completamente independente, com base no espaço do nome. Quando um atributo é adicionado a um grupo de atributos, este apenas vai ter impacto nos tipos de componentes que forem criados após a adição. Os tipos de componentes pré-existentes permanecem inalterados.

Para actualizar o nome e a descrição de um grupo de atributos invoque o método `update()` no `DKAttrGroupDefICM` e forneça uma matriz de novos IDs de atributo. Se este grupo de atributos já tiver sido associado a um tipo de componente, não poderá actualizar este grupo de atributos.

Para eliminar um grupo de atributos, o utilizador também trabalha com a classe `DKAttrGroupDefICM`. Ao eliminar o grupo de atributos, os atributos primários que anteriormente formavam o grupo de atributos permanecem no servidor de bibliotecas. As seguintes excepções aplicam-se quando elimina um grupo de atributos:

- Um grupo de atributos não pode ser eliminado, caso esteja associado a um tipo de componente e seja constante.
- Um atributo não pode ser removido de um grupo de atributos, caso este grupo esteja associado a um tipo de componente e seja constante.
- O utilizador não pode adicionar um atributo a um grupo de atributos caso este grupo esteja associado a um tipo de componente e seja constante.

O exemplo seguinte demonstra como criar um grupo de atributos:

Exemplo: Java

```
// Create a datastore definition object given the connected datastore
DKDatastoreDefICM dsDefICM = (DKDatastoreDefICM)dsICM.datastoreDef();
//Creating a new attribute group
DKAttrGroupDefICM attrGroup = new DKAttrGroupDefICM(dsICM);
// Set a name, maximum 15 characters
attrGroup.setName("Book_Details");
//Set a description for the new attribute group
attrGroup.setDescription("Detailed book information");
// Retrieve the definition of an attribute that will be
//added to this attribute group.
DKAttrDefICM title = (DKAttrDefICM) dsDefICM.retrieveAttr("book_title");
// Retrieve the definition of another attribute that will be added
//to this attribute group.
DKAttrDefICM publisher = (DKAttrDefICM) dsDefICM.retrieveAttr("publisher");
// Add the Attribute Definitions to the Attribute Group
attrGroup.addAttr(title);
attrGroup.addAttr(publisher);
// add it to the persistent datastore
attrGroup.add();
```

Exemplo: C++

```
// Create a datastore definition object given the connected datastore
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*) dsICM->datastoreDef();
//Creating a new attribute group
DKAttrGroupDefICM* attrGroup = new DKAttrGroupDefICM(dsICM);
// Set a name, maximum 15 characters
attrGroup->setName("Book_Details");
//Set a description for the new attribute group
attrGroup->setDescription("Detailed book information");
// Retrieve the definition of an attribute that will be
//added to this attribute group.
DKAttrDefICM* title =
    (DKAttrDefICM *) dsDefICM->retrieveAttr("book_title");
// Retrieve the definition of another attribute that will be added
//to this attribute group.
DKAttrDefICM* publisher =
    (DKAttrDefICM *) dsDefICM->retrieveAttr("publisher");
// Add the Attribute Definitions to the Attribute Group
attrGroup->addAttr(title);
attrGroup->addAttr(publisher);
// add it to the persistent datastore
attrGroup->add();

delete(attrGroup);
```

Para obter mais informações, consulte o exemplo SAttributeGroupDefCreationICM.

Enumerar os atributos num servidor de conteúdos

O exemplo demonstra como obter uma lista de atributos num servidor de conteúdos para Java e C++.

Exemplo: Java

Para Java

```
DKDatastoreICM dsICM;
DKDatastoreDefICM dsDefICM=DKDatastoreDefICM(dsICM.datastoreDef());
//Get a collection containingall Attribute Definitions.
DKSequentialCollection attrDefColl =
    (DKSequentialCollection)dsDefICM.listAttrs();
if ((attrDefColl!=null) &&(attrDefColl.cardinality(>0))){
    //Create an iterator to iterate through the collection
    dkIterator iter = attrDefColl.createIterator();
    while(iter.more()){
```

```

        //while there are still items in the list,continue
        dkAttrDef attrDef = (dkAttrDef) iter.next();
        System.out.println("-"+attr.getName()+":"+attr.getDescription());
    }
}

```

Exemplo: C++

Para C++

```

DKDatastoreICM * dsICM; .....
DKDatastoreDefICM*dsDefICM =(DKDatastoreDefICM *)dsICM->datastoreDef();
//Obtenha um recolha com todas as Definições de Atributos.
DKSequentialCollection*attrDefColl =
    (DKSequentialCollection *)dsDefICM->listAttrs();
if (attrDefColl &&(attrDefColl->cardinality()>0)){
    //Create an iterator to iterate through the collection
    dkIterator*iter =attrDefColl->createIterator();
    while(iter->more()){
        //while there are still items in the list,continue
        dkAttrDef* attrDef =(dkAttrDef *)iter->next()->value();
        cout <<"- "<<attrDef->getName()<<": "<<attrDef->getDescription()<<endl;
        delete(attrDef);
    }
    delete(iter);
    delete(attrDefColl);
}
delete(dsDefICM);

```

Enumerar nomes de atributos para um tipo de artigo

O exemplo seguinte demonstra como obter uma lista de nomes de atributos para um tipo de artigo.

Exemplo: Java

```

//Get a collection containing all attr. defs for the Item Type.
DKSequentialCollection attrColl = (DKSequentialCollection)
    dsICM.listEntityAttrs(itemTypeName);
//Accessing attribute each and printing the name and description.
System.out.println("\nAttributes of Item Type '"+itemTypeName+":
    (" +attrColl.cardinality()+')');
//Create an iterator to iterate through the collection
dkIterator iter = attrColl.createIterator();
while(iter.more()) {
    //while there are still items in the list, continue
    DKAttrDefICM attr = (DKAttrDefICM)iter.next();
    System.out.println("-"+attr.getName()+":"+attr.getDescription());
}

```

Exemplo: C++

```

//Get a collection containing all Attribute Definitions for the Item Type.
DKSequentialCollection*attrColl =(DKSequentialCollection*)
    dsICM->listEntityAttrs(itemTypeName);
//Accessing attribute each and printing the name and description.
cout <<"\nAttributes of Item Type '"<<itemTypeName <<":
    ("<<attrColl->cardinality()<<')'<<
//Create an iterator to iterate through the collection
dkIterator* iter =attrColl->createIterator();
while(iter->more()) {
    //while there are still items in the list, continue
    DKAttrDefICM* attr =(DKAttrDefICM*)iter->next()->value();
    cout <<"- "<<attr->getName()<<": "<<attr->getDescription()<<endl;
}

```

```
delete(attr);
}
delete(iter);
delete(attrColl);
```

Para obter mais informações, consulte o exemplo formativo de API `SItemTypeRetrievalICM`.

Criar um artigo

Um artigo corresponde à árvore completa de DDOs de componentes com, pelo menos, um componente raiz.

Um artigo ou DDO deve conter uma propriedade de tipo genérico associada ao mesmo. Este tipo genérico de DDO consiste num documento, pasta ou artigo. Ao criar artigos, a propriedade artigo é especificada através do segundo parâmetro da função `createDDO` do arquivo de dados em conjunto com o tipo semântico. É apenas submetido um valor, no entanto a propriedade artigo e os valores dos tipos semânticos são derivados e definidos em separado no DDO. Os tipos de propriedade artigo constituem um subconjunto de tipos semânticos. Este valor é armazenado na propriedade de DDO, `DK_CM_PROPERTY_ITEM_TYPE`, que não deverá ser confundida com a definição geral de tipos de artigos, que o descreve como sendo a estrutura deste artigo. `PROPERTY_ITEM_TYPE` refere-se à propriedade artigo e não à definição de tipo de artigo que descreve a estrutura e as definições do artigo na totalidade.

A Tabela 10 apresenta uma lista dos tipos de propriedades de artigo disponíveis.

Tabela 10. Definições de propriedade de tipo de artigo

Tipo de propriedade	Constante	Definição
Documento	<code>DK_CM_DOCUMENT</code>	O artigo representa um documento ou dados armazenados. As informações contidas neste artigo poderão ser utilizadas para formar um documento. Este artigo pode ser considerado um documento comum, uma vez que não significa obrigatoriamente uma implementação de um modelo de documento específico. A definição de um tipo de documento não representa necessariamente um documento com partes que utilizam o modelo de documentos DB2 Content Manager V8, que é controlado pela classificação "documento" na definição do tipo de artigo.

Tabela 10. Definições de propriedade de tipo de artigo (continuação)

Tipo de propriedade	Constante	Definição
Pasta	DK_CM_FOLDER	O artigo representa uma pasta que pode conter outros artigos. Este artigo inclui um conjunto incorporado de DKFolder, que pode conter o conteúdo das pastas, que por sua vez formam ligações de pastas para outros artigos. Um conjunto DKFolder será automaticamente adicionado. Todos os artigos com este valor têm a capacidade de utilizar o mecanismo DKFolder. No entanto, este valor não significa necessariamente que uma solução utiliza o mecanismo DKFolder e pode, em alternativa, implementar a sua própria solução de pastas. Este valor consiste em qualquer interpretação de uma pasta, contudo o DKFolder incorporado está disponível para tais artigos. Nota: É possível criar ligações que não sejam para pastas e referências de artigos de qualquer tipo semântico ou tipo de propriedade de artigos.
Artigo (predefinição)	DK_CM_ITEM	Artigo genérico. Este artigo corresponde ao tipo de propriedades de artigos para todos os tipos semânticos que não sejam documentos ou pastas.

Os artigos são criados como DKDDOs. Utilize sempre os métodos createDDO() do DKDatastoreICM para criar DKDDOs, uma vez que o sistema utiliza os métodos createDDO do DKDatastoreICM para configurar automaticamente informações importantes na estrutura do DKDDO. Os construtores DDO não são recomendados. A utilização dos vários métodos DKDatastoreICM::createDDO() é uma das formas recomendadas de criação de ocorrências DKDDO (ou subclasses). Não se recomenda a criação de DDOs utilizando novos ou quaisquer meios de criação de ocorrências DDO directamente a partir da classe de DDO. Os métodos createDDO() criam instâncias que corrigem as subclasses e configuram quaisquer outras informações necessárias nos DDOs para funcionar correctamente noutras operações.

Ao criar artigos, um tipo semântico pode ser especificado como segundo parâmetro da função createDDO do arquivo de dados. Um tipo semântico corresponde geralmente a uma metáfora de um artigo. Os tipos semânticos constituem uma função para utilização de aplicações e, geralmente, não são impostos, nem especialmente significativos no servidor DB2 Content Manager. Isto confere aos artigos uma característica imposta para determinadas operações básicas e faculta aplicações com a capacidade de aplicar um tipo semântico. Onde aplicável, os tipos semânticos são utilizados com DB2 Content Manager, contudo o único tipo semântico que contém algum significado, validação ou imposição é PASTA.

Não se esqueça: Não se baseie nos valores de tipos semânticos, se a aplicação tiver a capacidade de ler artigos criados por outras aplicações ou ferramentas, uma vez que o significado e a imposição podem variar entre as aplicações. Por exemplo, não se baseie nas várias partes dos tipos semânticos, uma vez que as partes podem ser definidas com qualquer valor de tipo semântico. Em alternativa,

baseie-se do tipo de objecto de PID, que identifica o tipo de artigo exacto, ou poderá obter a definição, que facultará mais informações, tais como a classificação como parte, a classe XDO, etc. Poderá basear-se no tipo semântico como sendo uma medida secundária, na eventualidade de não reconhecer o tipo de objecto.

O valor é armazenado na propriedade DKConstantICM.DK_ICM_PROPERTY_SEMANTIC_TYPE do DDO, que, em muitos casos, poderá conter o mesmo valor que o tipo de propriedade do artigo. A Tabela 11 enumera os tipos semânticos disponíveis.

Tabela 11. Tipos semânticos predeterminados

Tipo semântico	Constante	Definição
Qualquer	Específico do utilizador	CreateDDO irá atribuir qualquer valor especificado ao mesmo valor de tipo semântico. Uma vez que as aplicações podem utilizar a semântica do modo que pretenderem, todos os valores são permitidos, excepto no caso dos tipos de artigos classificados como "documento" (modelo de documento).
Documento	DK_ICM_SEMANTIC_TYPE_DOCUMENT	O artigo normalmente representa um documento. A definição de um tipo semântico num documento não representa necessariamente um documento com partes que utilizam o "modelo de documentos" DB2 Content Manager V8, que, pelo contrário, é controlado pela classificação "documento" na definição de tipos de artigos. Equivalente ao tipo de propriedade de artigos DK_CM_DOCUMENT.
Pasta	DK_ICM_SEMANTIC_TYPE_FOLDER	O artigo representa uma pasta que pode conter outros artigos. Este artigo inclui um conjunto incorporado de DKFolder, que pode conter o conteúdo das pastas, que por sua vez formam ligações de pastas para outros artigos. Um conjunto DKFolder será automaticamente adicionado. Todos os artigos com este valor têm a capacidade de utilizar o mecanismo DKFolder. No entanto, este valor não significa necessariamente que uma solução utiliza o mecanismo DKFolder e pode, em alternativa, implementar a sua própria solução de pastas. Este valor consiste em qualquer interpretação de uma pasta, contudo o DKFolder incorporado está disponível para tais artigos. Tenha em atenção que é possível criar ligações que não sejam para pastas e referências de artigos de qualquer tipo semântico ou tipo de propriedade de artigos. Equivalente ao tipo de propriedade de artigos DK_CM_FOLDER.

Tabela 11. Tipos semânticos predeterminados (continuação)

Tipo semântico	Constante	Definição
Contentor	DK_ICM_SEMANTIC_TYPE_CONTAINER	O artigo representa um contentor genérico ou um artigo genérico. Este é equivalente ao tipo de propriedade de artigos DK_CM_ITEM. AS soluções podem implementar uma relação de restrição em qualquer solução que escolherem, excepto no caso do mecanismo DKFolder incorporado (ligações a pastas). Pode utilizar ligações de qualquer outro tipo de ligação, atributos de referências ou qualquer solução personalizada da sua escolha.
Base	DK_ICM_SEMANTIC_TYPE_BASE	Atribuído alegadamente à maioria das partes de documentos semelhantes do tipo "BASE" para utilização no modelo de documentos incorporado onde as partes são criadas e adicionadas para constituir um documento. No entanto, este valor não identifica nem torna isto numa parte BASE. Este valor não é validado ou imposto. O tipo de artigo da parte (identificado no tipo de objecto de PID para a parte DDO) é o que o torna uma parte BASE. As partes BASE podem ser criadas com qualquer tipo semântico escolhido pelo utilizador. Defina este tipo semântico para as partes BASE, no entanto, quando forem detectados tipos de partes, utilize, em primeiro lugar, o tipo de objecto de PID. Utilize este valor, caso tenha criado tipos de artigos de partes personalizadas, de modo a que outras aplicações que possam não ter conhecimento do tipo de parte personalizado o possam correlacionar ao tipo de parte com maior semelhança que este tem a capacidade de compreender.
BaseText	DK_ICM_SEMANTIC_TYPE_BASETEXT	Igual a BASE (Consulte a documentação sobre "Base" para obter mais detalhes), excepto pelo facto de ser atribuído ao "BASETEXT" com maior semelhança.
BaseStream	DK_ICM_SEMANTIC_TYPE_BASESTREAM	Igual a BASE (Consulte a documentação sobre "Base" para obter mais detalhes), excepto pelo facto de ser atribuído ao "BASESTREAM" com maior semelhança.
Anotação	DK_ICM_SEMANTIC_TYPE_ANNOTATION	Igual a BASE (Consulte a documentação sobre "Base" para obter mais detalhes), excepto pelo facto de ser atribuído ao "ANNOTATION" com maior semelhança.
Histórico	DK_ICM_SEMANTIC_TYPE_HISTORY	Igual a BASE (Consulte a documentação sobre "Base" para obter mais detalhes), excepto pelo facto de ser atribuído ao "HISTORY" com maior semelhança.
Nota	DK_ICM_SEMANTIC_TYPE_NOTE	Igual a BASE (Consulte a documentação sobre "Base" para obter mais detalhes), excepto pelo facto de ser atribuído ao "NOTE" com maior semelhança.

Tabela 11. Tipos semânticos predeterminados (continuação)

Tipo semântico	Constante	Definição
Definido pelo utilizador	Definido pelo utilizador	Um tipo semântico definido pelo utilizador. Consulte os Exemplos de tipos semânticos para obter mais informações.

Nota:

- Em Java, as constantes estão definidas no ficheiro DKConstantICM.java.
- Em C++, as constantes estão definidas no ficheiro DKConstantICM.h.

Quando cria um artigo, o utilizador cria os componentes que formam o artigo. Por exemplo, se optar por criar um artigo hierárquico, terá de criar componentes descendentes.

1. Ao utilizar os métodos createDDO e createChildDDO do servidor de conteúdos, crie um DDO de artigo e defina todos os seus atributos e outras informações necessárias. Este exemplo utiliza o objecto de DKDatastoreICM com sessão iniciada e ligado denominado dsICM.
2. Crie um componente raiz.

Exemplo 1 de Java

```
DKDDO myDocumentDDO = dsICM.createDDO("EmployeeDoc",
    DKConstantICM.DK_CM_DOCUMENT);
```

Exemplo 2 de Java

```
DKDDO myFolderDDO = dsICM.createDDO("DeptFolder",
    DKConstantICM.DK_CM_FOLDER);
```

C++ Exemplo 1

```
DKDDO* myDocumentDDO = dsICM->createDDO("EmployeeDoc",DK_CM_DOCUMENT);
```

C++ Exemplo 2

```
DKDDO* myFolderDDO = dsICM->createDDO("DeptFolder",DK_CM_FOLDER);
```

3. Crie um componente descendente sob o componente raiz "EmployeeDoc" como, por exemplo, "Dependent".

Java

```
DKDDO myDDO =dsICM.createChildDDO("EmployeeDoc","Dependent");
```

C++

```
DKDDO* myDDO = dsICM.createChildDDO("EmployeeDoc","Dependent");
```

4. Adicione o componente descendente ao ascendente.

Java

```
short dataid = myDocumentDDO.dataId(DK_CM_NAMESPACE_CHILD,"Dependent");
DKChildCollection children =
    (DKChildCollection) myDocumentDDO.getData(dataid);
children.addElement(myDDO);
```

C++

```
short dataid = myDocumentDDO->dataId(DK_CM_NAMESPACE_CHILD,"Dependent");
DKChildCollection* children =
    (DKChildCollection*)(dkCollection*)
myDocumentDDO->getData(dataid);
children->addElement(myDDO);
```

5. Utilize o método setData para preencher o DDO com os valores adequados.

Java

```
myDDO.setData(.....);
```

C++

```
myDDO->setData(....);
```

6. Guarde este artigo na memória persistente.

Java

```
myDocumentDDO.add();
```

C++

```
myDocumentDDO->add();
```

No exemplo anterior, o último passo criou um documento no servidor de conteúdos. Quando um DDO de documento é adicionado a um servidor de conteúdo, todos os respectivos atributos são adicionados, incluindo todas as partes que se encontram dentro do conjunto DKParts. Quando um DDO de documento é adicionado a um servidor de conteúdos, todos os respectivos atributos são adicionados, incluindo todos os descendentes e todas as partes que se encontram dentro do conjunto DKParts.

Se criar um artigo num tipo de artigo que foi definido como tipo de artigo de recurso, é devolvido o XDO correcto. Para obter mais informações relativas a recursos, consulte o exemplo SResourceItemCreationICM. Poderá também revelar-se útil rever as informações relativas à criação de artigos no exemplo SItemCreationICM.

Definir e obter valores de atributo de artigo

O exemplo mostra como definir e obter valores de atributo de artigo.

Exemplo: Java

Os valores de atributo são armazenados e obtidos como java.lang.Objects. Para definir ou obter valores de atributo, utilize s métodos setData e getData() do DDO, respectivamente. Defina o valor utilizando um objecto com o tipo que corresponde ao tipo de atributo. Exemplo:

```
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
    nameOfAttrStr),valueObj);
Object obj = ddo.getData(ddo.dataId(DK_CM_NAMESPACE_ATTR,nameOfAttrStr));
```

A instrução anterior define o valor de atributo como o valor transmitido como um java.lang.Object valueObj. O nameOfAttrStr é o nome da cadeia do atributo. Este atributo foi definido e especificado no tipo de artigo quando o tipo de artigo deste DDO foi especificado. valueObj é o valor que deve definir e que deve ser o tipo certo para este atributo.

Exemplo: C++

Ao definir valores para atributos individuais, utilize o nome de definição do atributo individual. Para aceder a atributos individuais que pertencem a um grupo de atributos, utilize este formato: <Nome Grupo Atributo>.<Nome Atributo>.

Exemplo:

```
//The code snippet below shows the value of the character attribute
// "book_title" for the item represented by the DDO ddoDocument is set to
// the value "Effective C++"
ddoDocument->setData(ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,
    DKString("book_title")),DKString("Effective C++"));
//The code snippet below shows the value of the integer attribute
// "book_num_pages" for the item represented
// by the DDO ddoDocument is set to the value "250"
ddoDocument->setData(ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,
```

```

        DKString("book_num_pages")), (long)250);
//This code snippet shows how the value of the "book_title" attribute of the
//item represented by the DDO ddoDocument is retrieved into the "title"
//string variable.
DKString title = (DKString) ddoDocument->getData(ddoDocument->
    dataId(DK_CM_NAMESPACE_ATTR, DKString("book_title")));

```

Definir e obter valores de atributo de chave remota

No DB2 Content Manager Versão 8, pode predefinir valores de atributo de chave remota para preencher menus pendentes de cliente com opções específicas.

Estes valores serão associados a um atributo de um tipo de artigo. Por exemplo, um tipo de artigo de participação auto pode ter um atributo de cidade no valores pendentes de New York e Chicago. DO mesmo modo, um tipo de artigo de participação vida pode ter um atributo de cidade no valores pendentes de Boston, Atlanta e Dallas.

O DB2 Content Manager Versão 8 faculta métodos na classe DKForeignKeyDefICM que podem obter valores de atributo de chave remota para um artigo. Com a Versão 8.3, o cliente e o servidor de biblioteca de administração do sistema suportam várias definições de atributos numa restrição de chave remota.

Além disso, pode ver em cascata as relações entre as listas pendentes. Por exemplo, se seleccionar Califórnia para o valor de atributo Estado, pode preencher a lista pendente Cidade com San Jose e Los Angeles. Para tal, defina um tipo de artigo de chave remota chamado EstadoCidade com dois atributos: Estado e Cidade. Pode, então, definir artigos para todas as combinações válidas de estado e cidade. Os outros tipos de artigo que tenham atributos Estado e Cidade pode utilizar o tipo de artigo EstadoCidade como chave remota e restringir os valores permitidos para cidade e estado. Pode ordenar os campos de chave remota para aparecerem por ordem de precedência (Estado e Cidade).

As chaves remotas pode ter uma ou mais relações de atributos, utilizando os conceitos seguintes:

Número de sequência da coluna

Identificador único atribuído para descrever a relação de atributo origem e atributo destino numa chave remota.

Nome de atributo origem

Referência a um atributo num artigo origem ou numa tabela origem. Por exemplo, State.

Nome de atributo destino

Referência a um atributo num artigo destino ou numa tabela destino. Por exemplo, City.

Apresentar sinalizador

Valor booleano que identifica os atributos da chave remota para preencher os menus pendentes. Se o definir como FALSE, é apresentado, em alternativa, um campo de entrada de texto.

Os exemplos que se seguem utilizam getSpecificAttrsForForeignKey() para devolver um conjunto de resultados com base numa pesquisa de nomes de atributos (coluna) fornecidos na chave remota e respectivos valores correspondentes. Este método só pode ser usado em conjunto com chaves remotas relacionadas com as tabelas definidas no DB2 Content Manager.

Exemplo: Java

```
ArrayList getSpecificAttrsForForeignKey(DKNVPair[] fkeyValues,  
String[] columnNames ) throws Exception, DKException
```

Devolve uma ArrayList de matrizes de Cadeia.

Exemplo: C++

```
dkCollection * getSpecificAttrsForForeignKey(DKNVPair[] fkeyValues,  
long fkeyValuesLen, char * columnNames[], long numOfColumns)
```

Devolve um dkCollection de objectos SpecificAttrResults, no qual cada objecto conterá um resultado char* com os resultados para cada coluna e o comprimento desta matriz de resultados. Pode, então, obter os resultados utilizando getResultArray() e getNumOfResults(), criando uma instância da classe SpecificAttrResults:

```
DKExport void SpecificAttrResults (long no_of_results) //construtor  
DKExport void addResult( char * res) //adds a string to the results[]  
char *[] getResultArray() //retrieves the results[] for traversing  
long getNumOfResults() //retrieves the number of length of no_of_array  
~ SpecificAttrResults() //desconstrutor
```

O parâmetro **DKNVPair** representa uma matriz de DKNVPairs que contém os nomes de atributos (colunas) nas chaves remotas e os respectivos valores correspondentes que serão utilizados para formular uma consulta. Opcionalmente, pode especificar atributos no parâmetro **columnNames** para limitar quais os valores de colunas múltiplas a devolver. Por predefinição devolve todos os resultados. Para ver um exemplo de resultados, consulte Tabela 12.

Tabela 12. Exemplo de resultados para parâmetros em getSpecificAttrsForForeignKey()

DKNVPairs	Nomes de atributo	Resultado
DKNVPair fKeyValues[0]("Country", "US");	columnNames[0] = "city"	Devolve os nomes de todas as cidades nos Estados Unidos
DKNVPair fKeyValues[0]("Country", "US") DKNVPair fKeyValues[1]("State", "California")	columnNames[0] = "City" columnNames[1] = "Zip"	Devolve os nomes de todas as cidades (e códigos respectivos) só no Estado da Califórnia
DKNVPair fKeyValues[0]("Country", "US") DKNVPair fKeyValues[1]("State", "California") DKNVPair fKeyValues[2]("City", "San Jose")	columnNames = null;	Como a chave remota contém os atributos: País (Country), Estado (State), Cidade (City), endereço (addresses), código postal (zip) e número de telefone (phone number), devolve todos os valores para address, zip e phone number.

Os exemplos seguintes devolver matrizes de cadeia de chaves remotas na seguinte ordem: número de sequência de coluna, atributo origem, atributo destino e valor de sinalizador de apresentação.

Exemplo: Java

```
public Vector listForeignKeyAttrDetails( ) throws Exception, DKException
```

Exemplo: C++

```
dkCollection* listForeignKeyAttrDetails()
```

Os seguintes exemplos definem novas associações entre atributos de dois tipos de componentes numa definição de chave remota.

Exemplo: Java

```
public void addSrcAndTgtAttrName(String srcAttrName, String tgtAttrName,  
int columnSeq, boolean displayFlag) throws Exception, DKException
```

Exemplo: C++

```
void addSrcAndTgtAttrName(char * srcAttrName, char* tgtAttrName, long  
columnSeq, DKBoolean displayFlag)
```

Os seguintes exemplos actualizam o valor do sinalizador de apresentação da definição da chave remota em memória para a relação de atributo especificada no número de sequência de coluna. A definição da chave remota precisa ainda de ser actualizada na memória constante para para que o novo valor fique constante. Este método só pode ser usado em conjunto com chaves remotas relacionadas com as tabelas definidas no DB2 Content Manager.

Exemplo: Java

```
public void setDisplayFlag(int columnSeq, boolean displayFlag)  
throws Exception, DKException
```

Exemplo: C++

```
void setDisplayFlag(long columnSeq, DKBoolean displayFlag)
```

Os seguintes exemplos devolvem o valor booleano para o sinalizador de apresentação (da definição de chave remota) para a relação de atributos especificada no número de sequência de coluna.

Exemplo: Java

```
public boolean getDisplayFlag(int columnSeq) throws Exception, DKException
```

Exemplo: C++

```
DKBoolean getDisplayFlag(long columnSeq)
```

Além disso, pode utilizar os seguintes métodos `CMBAttribute` para determinar valores predefinidos, que são valores atribuídos ao atributo origem que pode ser referenciado posteriormente. Por exemplo, o atributo `City` pode ter valores predeterminados como, por exemplo `Lisboa` e `Porto` que podem ser apresentados num menu pendente.

hasPredefinedValues()

Valor booleano que indica se um atributo tem ou não valores predeterminados.

getPredefinedValues()

Devolve os valores predeterminados de um atributo.

getDependentValues()

Devolve uma lista de `CMBAttributes` cujos valores predefinidos dependem deste atributo.

getControllingAttributes()

Devolve uma lista de CMBAttributes dos quais os valores predefinidos deste atributo dependem.

Para mais informações sobre estes métodos, consulte os exemplos TGetPreDefValues.java e TGetPreDefValues.cpp.

Modificar atributos de um artigo

Para modificar atributos de um artigo, utilize o método setData do DDO

Defina o método setData como o valor requerido especificando esse valor através da utilização java.lang.Object, conforme indicado no exemplo que se segue.

Exemplo: Java

No exemplo seguinte, nameOfAttrStr é o nome de cadeia do atributo e valueObj corresponde ao valor.

```
ddo.setData(ddo.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
    nameOfAttrStr),valueObj);
```

Exemplo: C++

```
ddoDocument->setData(ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,"book_title"),
    DKString("More Effective C++"));
```

Actualizar um artigo

Para actualizar um artigo, é necessário obter o artigo, modificar os respectivos atributos e invocar a função de actualização.

Para actualizar um artigo:

1. Obtenha um artigo ou crie um artigo e adicione-o ao servidor de conteúdos.
2. Modifique o artigo, os seus atributos, recolhas de ligações e assim por diante.
3. Invoque a operação update do DDO.

Exemplo: Criar e actualizar uma versão

Se um artigo for activado para criação de versões, o utilizador pode criar uma nova versão do artigo em vez de actualizar o artigo actual, tal como o exemplo seguinte demonstra.

Java

```
ddo.update(DKConstant.DK_CM_VERSION_NEW);
```

C++

```
ddo->update(DK_CM_VERSION_NEW);
```

Para actualizar a versão mais recente de um artigo, utilize o formato apresentado no exemplo seguinte:

Java

```
ddo.update(DKConstant.DK_CM_VERSION_LATEST);
```

C++

```
ddo->update(DK_CM_VERSION_LATEST);
```

Pode também actualizar um DDO invocando o método updateObject no DKDatastoreICM com as opções apropriadas, conforme indicado no exemplo que se segue:

Java

```
// Connected DKDatastoreICM object named ds;  
// DKDDO object named ddo;  
int options = DK_CM_VERSION_NEW + DK_CM_CHECKIN;  
ds.updateObject(ddo, options);
```

C++

```
int options = DK_CM_VERSION_NEW + DK_CM_CHECKIN;  
ds->updateObject(ddo, options);
```

Importante: Terá de dar saída do artigo antes de invocar o método update e voltar a dar entrada do artigo quando tiver terminado a actualização. Consulte. Para obter mais informações sobre a actualização de itens, consulte o exemplo formativo de API SItemUpdateICM.

Definir um tipo de artigo de recurso

Um *artigo de recurso* é um artigo com atributos de sistema adicionais que definem a localização, o tipo, o tamanho, etc., do objecto que o artigo representa.

O objecto é, por vezes, denominado recurso e pode ser um ficheiro de vídeo, uma imagem, um documento de processador de texto, etc. Para obter informações adicionais, consulte o exemplo SItemTypeCreationICM (no directório samples).

Os passos seguintes explicam-lhe o processo de definição de um tipo de artigo de recurso.

1. Obtenha o objecto de definição do servidor de conteúdos a partir do servidor de conteúdos ligado.

Java

```
DKDatastoreDefICM dsDefICM = (DKDatastoreDefICM)dsICM.datastoreDef();
```

C++

```
DKDatastoreDefICM* dsDefICM = (DKDatastoreDefICM*)  
dsICM->datastoreDef();
```

2. Crie uma nova definição de tipo de artigo.

Java

```
DKItemTypeDefICM itemType = new DKItemTypeDefICM(dsICM);  
itemType.setName("SampleResource");  
itemType.setDescription("Simple Resource Lob Item Type");
```

C++

```
DKItemTypeDefICM * itemType = new DKItemTypeDefICM(dsICM);  
itemType->setName("SampleResource");  
itemType->setDescription("Simple Resource Lob Item Type");
```

3. Adicione um atributo.

Java

```
DKAttrDefICM attr =(DKAttrDefICM)dsDefICM.retrieveAttr("S_varchar");  
itemType.addAttr(attr);  
//Resource classification indicates that this class will  
//contain data file  
itemType.setClassification  
(DKConstantICM.DK_ICM_ITEMTYPE_CLASS_RESOURCE_ITEM);
```

C++

```
DKAttrDefICM * attr = (DKAttrDefICM*) dsDefICM->  
retrieveAttr("S_varchar");  
itemType->addAttr(attr);  
// Resource classification indicates that this class will contain  
//data file  
itemType->setClassification(DK_ICM_ITEMTYPE_CLASS_RESOURCE_ITEM);
```


4. Especifique a classe de XDO e o tipo de recurso deste tipo de artigo.

Java

```
itemType.setXDOClassName(DKConstantICM.DK_ICM_XDO_LOB_CLASS_NAME);
itemType.setXDOClassID(DKConstantICM.DK_ICM_XDO_LOB_CLASS_ID);
itemType.add();
```

C++

```
itemType->setXDOClassName(DK_ICM_XDO_LOB_CLASS_NAME);
itemType->setXDOClassID(DK_ICM_XDO_LOB_CLASS_ID);
itemType->add();
```

Criar um artigo de recurso

Criar artigos de recurso é muito semelhante a criar artigos comuns. Os XDOs expandem os DDOs e, dependendo do tipo de artigo de recurso, o XDO pode ser ainda mais expandido.

A Tabela 13 contém os tipos de artigo do recurso e a hierarquia de classe utilizada para os criar. Para obter mais informações, consulte o exemplo `SResourceItemCreationICM` (no directório `samples`).

Tabela 13. Hierarquia de classe do tipo de artigo do recurso

Tipo	DDO	XDO	Extensão
LOB	DKDDO -> DKLobICM		
Texto	DKDDO ->	DKLobICM ->	DKTextICM
Imagem	DKDDO ->	DKLobICM ->	DKImageICM
Sequência	DKDDO ->	DKLobICM ->	DKStreamICM
Sequência vídeo	DKDDO ->	DKLobICM ->	DKVideoStreamICM

Os passos seguintes explicam-lhe o processo de criação de um artigo de recurso. Note que há várias formas de definir e armazenar conteúdos de recurso.

1. Crie o artigo do recurso.

Importante: O artigo de recurso pode ser de qualquer tipo semântico e a chamada de `DKDatastoreICM::createDDO` é também utilizada para criar um artigo de recurso da mesma forma que é utilizada para criar um artigo comum.

O tipo de recurso devolvido por `DKDatastoreICM.createDDO` é convertido na subclasse correcta (neste caso, `DKLobICM`) com base na classificação de XDO definida durante a criação do tipo de artigo no qual este artigo de recurso é baseado.

Java

```
DKLobICM lob = (DKLobICM)dsICM.createDDO
("SampleResource",DKConstant.DK_CM_DOCUMENT);
```

C++

```
DKLobICM* lob = (DKLobICM*)
dsICM->createDDO("SampleResource",DK_CM_DOCUMENT);
```

2. Defina o conteúdo ou dados no objecto. Após a criação do LOB, pode definir o tipo MIME do recurso. Neste caso, o recurso é um documento de MS Word. O tipo MIME descreve o tipo de conteúdo que está a ser armazenado.

Java

```
lob.setContentFromClientFile(fileName);
lob.setMimeType("application/msword");
```

C++

```
lob->setContentFromClientFile(fileName);  
lob->setMimeType("application/msword");
```

Para obter mais informações sobre tipos MIME, consulte o exemplo `SResourceItemMimeTypesICM`.

3. Adicione os dados ao servidor de conteúdos. Neste caso, trata-se de um documento Word exemplo.

Não se esqueça: O conteúdo do artigo de recurso é armazenado no gestor de recursos.

Java

```
lob.add("SResourceItemICM_Document1.doc");
```

C++

```
lob->add("SResourceItemICM_Document1.doc");
```

Importante: Em C++, é necessário limpar a memória.
`delete(lob);`

Para obter mais informações, consulte o exemplo `SItemUpdateICM`.

O nome original, o nome do artigo antes de ser armazenado no DB2 Content Manager, de um artigo de recursos ou de uma parte de documento é armazenado no servidor de bibliotecas como um atributo do artigo de recursos ou da parte de documento. Para obter o nome do ficheiro original (se ainda não o tiver obtido ou pretender actualizar o valor), invoque `retrieve()` (artigo único ou artigos múltiplos) e solicite opções que incluam, no mínimo os atributos de recursos do sistema, tais como `DKRetrieveOptionsICM::baseAttributes(true)`. Opcionalmente, pode adicionar um filtro apenas para atributos do sistema (`DKRetrieveOptionsICM::attributeFilters()`). No entanto, para actualizar ou obter atributos de recursos do sistema, não é necessário obter o conteúdo dos recursos. Por exemplo, pode utilizar `DKRetrieveOptionsICM::resourceContent(false)` (predefinição).

Pesquisar itens

Pode pesquisar os itens que correspondam aos critérios.

Para localizar artigos que correspondam a um conjunto de critérios, conclua os seguintes passos:

1. Estabeleça ligação ao um objecto `DKDatastoreICM`.
2. Processe a consulta correcta. Certifique-se de que a sua consulta está de acordo com a linguagem de consulta.
3. Guarde os resultados da consulta num objecto `DKResult`.

Exemplo

Java

```
DKNVPair options[] = new DKNVPair[3];  
options[0] =  
    new DKNVPair(DKConstant.DK_CM_PARM_MAX_RESULTS, "0"); // No Max (Default)  
DKRetrieveOptionsICM dkRetrieveOptions =  
    DKRetrieveOptionsICM::createInstance(dsICM);  
dkRetrieveOptions.baseAttributes(true);  
options[1] = new DKNVPair(DKConstant.DK_CM_PARM_RETRIEVE,  
    dkRetrieveOptions);
```

```
options[2] = new DKNVPair(DKConstant.DK_CM_PARM_END, null);
DKResults results = (DKResults)dsICM.evaluate(query,
DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
```

C++

```
DKNVPair *options = new DKNVPair[3];
// only one result will be returned
options[0].set(DK_CM_PARM_MAX_RESULTS, "1");// Retrieve only one item
DKRetrieveOptionsICM* dkRetrieveOptions =
DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOptions->baseAttributes(TRUE);
options[1].set(DK_CM_PARM_RETRIEVE, dkRetrieveOptions);
// Last option has to be this value
options[2].set(DK_CM_PARM_END, (long)0);
//Note if a query is expected to return more than one result,
//the DKDatastoreICM::execute method
//should be used. The execute method returns a dkResultSetCursor.
DKResults* results = (DKResults*)(dkCollection*)dsICM->evaluate
(query, DK_CM_XQPE_QL_TYPE, options);
```

Opção de tempo limite da consulta

A opção de tempo limite de consulta, DKConstant.DK_CM_SQL_TIMEOUT, pode ser adicionada como DKNVPair na matriz de opções de consulta. É necessário um objecto de número inteiro que represente um número de milissegundos antes de se esgotar tempo de espera da consulta ou de esta ser cancelada. Esta opção é suportada em todos os métodos de consulta DKDatastoreICM: execute, evaluate, executeCount e executeWithCallback.

Importante: Esta opção tem uma base por consulta e deve ser passada com cada pedido de consulta de API de DB2 Content Manager.

Consulte o seguinte fragmento de código Java™ para obter um exemplo da definição da opção de tempo limite e para invocar o método consulta. A opção de tempo limite está disponível para APIs de Java e de C++.

```
// Create the query timeout option.
//Set to 5 seconds (5000 milliseconds)
DKNVPair options[] = null;
options = new DKNVPair[2];
options[0] = new DKNVPair( DKConstant.DK_CM_SQL_TIMEOUT,
new Integer(5000));
options[1] = new DKNVPair(DKConstant.DK_CM_PARM_END,null);
// Use one of the query methods to perform the query
dkResultSetCursor cursor = dsICM.execute(
"/NOINDEX", DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
DKResults results = (DKResults) dsICM.evaluate(
"/NOINDEX", DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
dsICM.executeWithCallback(
"/NOINDEX", DKConstantICM.DK_CM_XQPE_QL_TYPE, options, callbackObject);
long count = dsICM.executeCount(
"/NOINDEX", DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
```

Para obter informações adicionais, consulte o exemplo SSearchICM.

Obter um artigo

O DB2 Content Manager fornece várias opções de obtenção.

Para obter um artigo, terá primeiro de obter um objecto DDO com informações de PID completas que representem o artigo (ou o o componente). Se tiver acesso a um objecto PID concluído ou souber a cadeia PID do item a obter, poderá reconstruir um objecto DDO utilizando o método DKDatastoreICM::createDDOFromPID(). O

método `DKDatastoreICM::createDDOFromPID()` aceita um objecto PID ou uma cadeia PID. Pode também utilizar uma consulta para localizar itens e devolver DDOs com informações de PID completas.

Nota: A consulta determina a lista de PIDs e cria DDOs em branco com os PIDs completos. No entanto, a consulta pode invocar o método `retrieve` multi-artigo antes de devolver os DDOs com base nas opções de consulta.

Após ter obtido um objecto DDO válido, pode obter ou actualizar os respectivos conteúdos. Pode obter directamente qualquer DDO que inclua o componente raiz, o componente descendente e DDOs de partes de documentos individuais. Um artigo é acedido através do DDO do componente raiz.

Existem dois tipos de métodos `retrieve`: `retrieve` multi-artigo e `retrieve` de artigo único. A operação `retrieve` multi-artigo proporciona vantagens significativas em termos de desempenho relativamente à operação `retrieve` de artigo único, que efectue chamadas separadas ao servidor. Sempre que pretender obter mais do que um DDO, terá de utilizar a operação `retrieve` multi-artigo, excepto se necessitar de dados que apenas podem ser acedidos através da operação `retrieve` de artigo único.

Por exemplo, o conteúdo de recurso apenas pode ser obtido através da operação `retrieve` de artigo único. Ao especificar as opções de obtenção numa consulta (ou pesquisa), a consulta utiliza a operação `retrieve` multi-artigo nos resultados. Deverá utilizar a operação `retrieve` multi-artigo para todos os metadados comuns requeridos de todos os DDOs. Quando um DDO requer conteúdo adicional, como, por exemplo, conteúdo de recurso para um artigo específico, esse artigo deverá ser obtido com o método `retrieve` de artigo único. Para obter exemplos de assinaturas de método de operações `retrieve` de artigo único e `retrieve` multi-artigo e consultas, consulte o `DKDatastoreICM::retrieveObjects(dkCollection,DKNVPair[])` no manual *Application Programming Reference*.

O DB2 Content Manager fornece uma classe `DKRetrieveOptionsICM` e uma classe integrada `DKProjectionListICM`. As opções de obtenção granular foram concebidas para seleccionar dados necessários. Além das optimizações incorporadas activadas automaticamente através do método `DKRetrieveOptions`, pode conseguir um melhor desempenho de obtenção, obtendo apenas o subconjunto de dados que pretende utilizar.

Obter componentes descendentes e partes de documentos

Pode obter ou actualizar directamente componentes descendentes e DDOs de partes de documentos sem invocar a operação `retrieve` nos DDOs ascendentes. Contudo, não é possível actualizar os itens obtendo o componente e invocando o método `DKDDO.update()`. Todas as actualizações têm de ser efectuadas através do componente raiz do artigo ou documento. Tem de modificar o componente descendente ou a parte do documento dentro da estrutura obtida do componente raiz invocando o método `DKDDO.update()` no componente raiz.

Especificar pedidos e exclusões utilizando opções de obtenção

Especifique sempre a lista de pedidos e exclusões para obter um melhor desempenho com base nas considerações de desempenho documentadas na classe `DKRetrieveOptionsICM`, no manual *Application Programming Reference*. A emissão de uma instância de `DKRetrieveOptionsICM` permite optimizações de obtenção

adicionais que não se encontram disponíveis através do comportamento predefinido ou relativo a bits, caso não seja emitida uma instância de `DKRetrieveOptionsICM`.

Importante: Se não especificar pedidos e exclusões, o desempenho da aplicação pode ser fortemente afectado.

É necessário utilizar opções de obtenção especificadas através da classe `DKRetrieveOptionsICM` para especificar qual o subconjunto de dados requerido. Deverá obter apenas os dados que pretende utilizar. Pode melhorar o desempenho utilizando cuidadosamente as opções de obtenção e utilizando custos de desempenho apenas para os dados que utiliza.

Para dados que possa utilizar apenas para um artigo, adie as obtenções adicionais até que os dados sejam necessários. Por exemplo, se está a enumerar 10 000 resultados de consulta, considere a obtenção de apenas atributos mínimos para a primeira página de resultados e obtenham mais dados apenas se o utilizador pedir a página seguinte ou fizer clique num documento para visualização.

Pode evitar chamadas desnecessárias ao servidor, à comunicação da rede, à utilização da memória e às contagens de objectos desnecessários, bem como outras despesas gerais. As implicações no desempenho de cada definição de opção de obtenção são apresentadas na documentação de referência de Javadoc, na classe `DKRetrieveOptionsICM`.

É necessário fornecer sempre opções de obtenção, mesmo se não obtiver quaisquer dados novos. As opções de obtenção que definir são construídas de forma incremental, sem nenhuma opção seleccionada por predefinição. Cada opção de obtenção que definir como `true` é considerada um pedido específico de dados e qualquer opção definida como `false` (predefinição) é considerada um pedido específico para excluir esses dados. É aconselhável submeter um objecto de opções sem alterações para indicar que não está a pedir quaisquer dados. Se não forem especificadas opções de obtenção, serão incluídas, por predefinição, várias selecções baseadas nas variantes de opções de obtenção de números inteiros relativos a bits. A predefinição varia de acordo com o método utilizado.

Importante: Utilize sempre que possível a classe `DKRetrieveOptionsICM` em variantes de opções de obtenção longas ou de números inteiros relativos a bits ou variantes de não opção. Utilizando a classe `DKRetrieveOptionsICM`, o utilizador activa um determinado número de optimizações e actualizações de produto que não estão disponível ao utilizar as opções anteriores de obtenção de números inteiros relativos a bits, mesmo com as definições de obtenção equivalentes. Se já tiver utilizando as opções de obtenção de números inteiros relativos a bits, consulte a classe `DKRetrieveOptionsICM` no manual *Application Programming Reference*.

Para obter mais informações, consulte as informações do Javadoc relativas a:

- `DKRetrieveOptionsICM`
- `DKDatastoreICM::retrieveObjects(dkCollection, DKNVPair[])`

São facultados documentação e exemplos adicionais no exemplo `SItemRetrievalICM`.

Obter um artigo: Java:

1. Após o estabelecimento de uma ligação através de um objecto DKDatastoreICM, pesquise o artigo utilizando o ID de artigo apropriado ou outras condições de atributo.
2. Guarde os resultados da pesquisa num objecto DKResults se estiver a utilizar evaluate(). No exemplo de código seguinte, queryStr é a cadeia de pesquisa no formato de linguagem de consulta correcto.

```
DKRetrieveOptionsICM dkRetrieveOpts =
DKRetrieveOptionsICM::createInstance(dsICM);
DKNVPair options[] = new DKNVPair[2];
options[0] = new DKNVPair(DKConstant.DK_CM_PARM_RETRIEVE,dkRetrieveOpts);
options[1] = new DKNVPair(DKConstant.DK_CM_PARM_END,null);
DKResults results = (DKResults)dsICM.evaluate(queryStr,
DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
```

3. Crie um iterador em DKResults e utilize-o para obter DDOs, com PIDs completos
4. Em vez de utilizar a consulta, se tiver uma cadeia de PID obtida anteriormente a partir de um DDO e a utilizar para reconstruir um novo DDO para um artigo existente, utilize os métodos DKDatastoreICM's createDDOFromPID(DKPidICM pid) ou createDDOFromPID(String pidString) do DKDatastoreICM, conforme indicado no exemplo que se segue. Não utilize o conector DKDDO. Pode utilizar este método para criar novamente um DDO em branco se pretender limpar completamente os dados existentes num DDO existente.

Nota: Se executou simplesmente a consulta, pode utilizar os DDOs devolvidos pela consulta directamente.

```
DKPidICM pid = ddo.getPidObject();
String pidStr = pid.pidString();DKDDO ddo = dsICM.createDDOFromPID(pid);
DKDDO ddo2 = dsICM.createDDOFromPID(pidStr);
```

Após ter um DDO, do Passo 4, que pretende obter, se pretender obter atributos raiz, pode utilizar a operação de obtenção de artigo único.

```
DKRetrieveOptionsICM dkRetrieveOpts =
DKRetrieveOptionsICM.createInstance(dsICM);
dkRetrieveOpts.baseAttributes(true);
ddo.retrieve(dkRetrieveOpts.dKNVPair());
```

Não se esqueça: Utilize a obtenção multi-artigo para mais do que um DDO.

Em alternativa, pode utilizar a obtenção multi-artigo para obter vários DDOs e reutilizar as mesmas opções do último exemplo de código.

```
dkCollection ddoColl = new DKSequentialCollection();
ddoColl.addElement(ddo);
dsICM.retrieveObjects(ddoColl,dkRetrieveOpts);
```

Se um artigo estiver activado para criação de versões, pode obter uma versão específica do artigo utilizando a consulta com condições no ID de versão ou modificando o número da versão no PID antes da obtenção. Pode ainda pedir a versão mais recente, tal como indicado em seguida.

```
DKRetrieveOptionsICM dkRetrieveOpts =
DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOpts.baseAttributes(true);
dkRetrieveOpts.functionVersionLatest(true);
ddo.retrieve(dkRetrieveOpts.dKNVPair());
```


Para a obtenção de artigo único, pode também invocar directamente o método de obtenção de artigo único do arquivo de dados. Este é, na realidade, o método de obtenção de artigo único que todos os outros métodos de obtenção de artigo único invocam.

```
dsICM.retrieveObject(ddo,dkRetrieveOpts);
```

Obter um artigo: C++:

1. Após o estabelecimento de uma ligação através de um objecto DKDatastoreICM, pesquise o artigo utilizando o ID de artigo apropriado ou outras condições de atributo.
2. Guarde os resultados da pesquisa num objecto DKResults se estiver a utilizar evaluate(). No exemplo de código seguinte, queryStr é a cadeia de pesquisa no formato de linguagem de consulta correcto.

```
DKNVPair* options = new DKNVPair[2];
options[0]->set(DK_CM_PARM_RETRIEVE, dkRetrieveOpts);
options[1]->set(DKConstant.DK_CM_PARM_END,(long)0);
DKResults* results = (DKResults*)(dkCollection*)
dsICM->evaluate(queryStr, DK_CM_XQPE_QL_TYPE, options);
```

3. Crie um iterador em DKResults e utilize-o para obter DDOs com PIDs completos.
4. Em vez de utilizar a consulta, se tiver uma cadeia de PID obtida anteriormente a partir de um DDO e pretender utilizá-la para reconstruir um novo DDO para um artigo existente, utilize os métodos createDDOFromPID(DKPidICM pid) ou createDDOFromPID(String pidString) do DKDatastoreICM. Não utilize o conector DKDDO. Pode utilizar este método para criar novamente um DDO em branco se pretender limpar completamente os dados existentes num DDO existente.

Nota: Se executou simplesmente a consulta, pode utilizar os DDOs devolvidos pela consulta directamente.

```
DKPidICM* pid = (DKPidICM*) ddo->getPidObject();
DKString pidStr = pid->pidString();
DKDDO* ddo = dsICM->createDDOFromPID(pid);
DKDDO* ddo2 = dsICM->createDDOFromPID(pidStr);
```

Após ter um DDO, do Passo 4, que pretende obter, se pretender obter atributos raiz, pode utilizar a operação de obtenção de artigo único.

```
DKRetrieveOptionsICM* dkRetrieveOpts =
DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOpts->baseAttributes(TRUE);
ddo->retrieve(dkRetrieveOpts->dkNVPair(),dkRetrieveOpts->dkNVPairLength());
```

Poderá pretender utilizar a obtenção multi-artigo para obter vários DDOs e reutilizar as mesmas opções do exemplo de código anterior.

```
dkCollection* ddoColl = new DKSequentialCollection();
ddoColl->addElement(ddo);
dsICM->retrieveObjects(ddoColl,dkRetrieveOpts);
```

Se um artigo estiver activado para criação de versões, pode obter uma versão específica do artigo utilizando a consulta com condições no ID de versão ou modificando o número da versão no PID antes da obtenção. Pode ainda pedir a versão mais recente, tal como indicado em seguida.

```
DKRetrieveOptionsICM* dkRetrieveOpts =
DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOpts->baseAttributes(TRUE);
dkRetrieveOpts->functionVersionLatest(TRUE);
ddo->retrieve(dkRetrieveOpts->dkNVPair(),dkRetrieveOpts->dkNVPairLength());
```

Para a obtenção de artigo único, pode também invocar directamente o método de obtenção de artigo único do arquivo de dados. Este é, na realidade, o método de obtenção de artigo único que todos os outros métodos de obtenção de artigo único invocam.

```
dsICM->retrieveObject(ddo,dkRetrieveOpts);
```

Eliminar artigos

Utilize o método delete no DDO para eliminar um artigo do servidor de conteúdos.

O DDO deve ter definidos o seu ID de artigo e tipo de objecto, bem como possuir uma ligação válida ao servidor de conteúdos.

Exemplo: Java

```
ddoDocument.del();
```

Exemplo: C++

```
ddoDocument->del();
```

Para obter mais informações sobre eliminação de artigos, consulte o exemplo formativo SItemDeletionICMAPI.

Eliminar versões de artigos

Se activar um artigo para criação de versões, tenha em atenção que tem várias versões desse artigo com as quais pode trabalhar. Se pretende eliminar completamente uma versão de artigo, tem de eliminar todas as versões desse artigo. Esta secção descreve como eliminar todas as versões de um artigo ou determinadas versões de um artigo.

Para eliminar determinadas versões de um artigo, efectue uma pesquisa de todos os artigos correspondente a alguns critérios de pesquisa como, por exemplo, ITEMID. Se não especificar uma versão nem a última versão, todas as versões correspondentes aos critérios de pesquisa são devolvidas. Se apenas especificou ITEMID como critério, todas as versões desse artigo são devolvidas. Pode, então, passar pela recolha de resultados e eliminar cada artigo individualmente.

Para eliminar todas as versões de um artigo em simultâneo, especifique a opção DKConstantICM::DK_ICM_DELETE_ALL_VERSIONS ao eliminar qualquer versão de um artigo individual. É importante especificar a opção DKConstantICM::DK_ICM_DELETE_ALL_VERSIONS, uma vez que, se não o fizer, apenas a versão do artigo representado pelo DDO do componente raiz é eliminada.

Para visualizar um código exemplo que utilize pesquisa e eliminação, consulte o exemplo SItemDeletionICM no directório samples.

Dar entrada e saída de artigos

Para impedir que dois utilizadores efectuem alterações ao mesmo artigo em simultâneo, é necessário dar saída ou bloquear um artigo antes de o actualizar.

Dar saída de um artigo concede os direitos exclusivos de actualização ao utilizador que tenha dado saída do artigo. Quando dá saída de um artigo, o utilizador mantém um bloqueio de escrita persistente sobre esse artigo. A totalidade do artigo é bloqueada e desbloqueada, incluindo todas as versões, componentes descendentes, conteúdo de recurso e partes de documentos. No entanto, os artigos ligados e referenciados não são bloqueados com o artigo, uma vez que apenas é

possível referenciar e não ser proprietário de artigos. O bloqueio de escrita persistente no artigo é libertado quando é dada entrada do artigo.

As operações de dar saída e dar entrada são semelhantes a bloquear ou desbloquear, respectivamente. Estas operações não recuperam as informações mais recentes ao ser dada saída e não mantêm as actualizações ao ser dada entrada, excepto quando transferidas como opções para operações de recuperação ou actualização. Deste modo, ao dar saída (bloquear) um artigo, lembre-se de que qualquer outro utilizador ou sessão iniciada pode ter efectuada alterações ao artigo no período de tempo entre a última recuperação do documento e a obtenção do bloqueio. Poderá actualizar o DDO com as informações mais recentes após a obtenção do bloqueio (o qual garante que o utilizador é o único que pode efectuar alterações) ou pedir que seja dada saída como parte da operação de obtenção original, utilizando as opções de obtenção.

O bloqueio de escrita é persistente e associado ao nome de utilizador e não à sessão iniciada. Os bloqueios permanecem activos até que o utilizador ou um superutilizador com autoridade suficiente desbloqueie o artigo. Além disso o mesmo utilizador pode efectuar alterações em qualquer sessão autenticada. Se estabelecer ligação com o mesmo nome de utilizador em várias ligações, todas as ligações terão autorizações para o bloqueio de escrita e podem efectuar alterações.

Pode obter o estado de saída dada que indica o proprietário actual do bloqueio (se existir) e a marca de hora do bloqueio através da opção de obtenção, `DKRetrieveOptions::basePropertyCheckedOutDetails(Boolean)`. Consulte as informações detalhadas no manual *Application Programming Reference* correspondente a este método para obter informações sobre utilização, comportamento e as propriedades de DDO nas quais as informações estão armazenadas. Pode também utilizar métodos de `DKDatastoreExtICM` para efectuar uma chamada separada de modo a verificar o estado de saída dada actual e localizar o proprietário actual do bloqueio. No entanto, este procedimento tem a desvantagem de uma chamada de servidor por verificação.

Para dar entrada e saída de artigos individuais, utilize as operações `checkIn` e `checkOut` do `DKDatastoreICM`, conforme indicado nos passos abaixo. Pode combinar operações de dar saída com obtenção ou combinar operações de dar entrada com actualização, conforme abaixo indicado

1. Se tiver estabelecido ligação a um objecto do `DKDatastoreICM` denominado `dsICM` e estiver a utilizar um DDO denominado `myDDO`, dê saída do artigo e, em seguida, dê novamente entrada do mesmo.

Nota: Normalmente, são executadas operações entre as operações de dar saída e dar entrada.

Java

```
dsICM.checkOut(myddo);  
dsICM.checkIn(myddo);
```

C++

```
dsICM->checkOut(myddo);  
dsICM->checkIn(myddo);
```

2. Se pretender combinar a operação de dar saída com obtenção e combinar a operação da dar entrada com actualização, siga o exemplo abaixo apresentado:

Nota: Normalmente, são executadas operações entre a obtenção e a actualização.

Se necessário, pode obter o estado de saída dada, que confirmará a propriedade do bloqueio.

Não se esqueça: Solicite apenas informações que pretende utilizar para minimizar custos.

Java

```
DKRetrieveOptionsICM dkRetrieveOpts =  
DKRetrieveOptionsICM.createInstance(dsICM);  
dkRetrieveOpts.baseAttributes(true);  
dkRetrieveOpts.basePropertyCheckedOutDetails(true);  
dkRetrieveOpts.functionCheckOut(true);  
ddo.retrieve(dkRetrieveOpts.dkNVPair());  
ddo.update(DKConstant.DK_CM_CHECKIN);
```

C++

```
DKRetrieveOptionsICM* dkRetrieveOpts =  
    DKRetrieveOptionsICM::createInstance(dsICM);  
dkRetrieveOpts->baseAttributes(TRUE);  
dkRetrieveOpts->basePropertyCheckedOutDetails(TRUE);  
dkRetrieveOpts->functionCheckOut(TRUE);  
ddo->retrieve(dkRetrieveOpts->dkNVPair(),  
    dkRetrieveOpts->dkNVPairLength());  
ddo->update(DK_CM_CHECKIN);
```

3. Para dar saída (bloquear) vários artigos em simultâneo, pode utilizar a obtenção de artigos múltiplos com a opção `DKRetrieveOptionsICM::functionCheckout(true)`. O exemplo seguinte reutiliza as mesmas opções do Passo 2.

Java

```
dkCollection ddoColl = new DKSequentialCollection();  
ddoColl.addElement(ddo1);  
ddoColl.addElement(ddo2);  
ddoColl.addElement(ddo3);  
dsICM.retrieveObjects(ddoColl, dkRetrieveOpts.dkNVPair());
```

C++

```
dkCollection* ddoColl = new DKSequentialCollection();  
ddoColl->addElement(ddo1);  
ddoColl->addElement(ddo2);  
ddoColl->addElement(ddo3);  
dsICM->retrieveObjects(ddoColl,  
    dkRetrieveOpts->dkNVPair(), dkRetrieveOpts->dkNVPairLength());
```

Para obter mais informações, notas de comportamento, considerações de desempenho, etc, consulte o manual *Application Programming Reference* relativo ao método `DKRetrieveOptionsICM::functionCheckOut(Boolean)`.

Para obter mais instruções, exemplos e demonstrações sobre como dar entrada e saída de artigos, consulte o exemplo formativo de API `SItemUpdateICM`.

Definir e obter as propriedades de criação de versões de um artigo

O exemplo a seguir apresentado demonstra como definir as propriedades de atribuição de versão de um artigo e como obter as propriedades de atribuição de versão de um artigo.

Exemplo: Java

```
DKPidICM pid = (DKPidICM)ddo.getPidObject();
// Accessing version information
String version = pid.getVersionNumber();
...
// Setting the version number in the DDO
pid.setVersionNumber(version);
```

Exemplo: C++

```
DKPidICM * pid = (DKPidICM *)ddo->getPidObject();
// Accessing version information
DKString version = pid->getVersionNumber();
....
//Setting the version number for the PID associated with an item (DDO).
pid->setVersionNumber((char *)version);
```

Trabalhar com versões de itens

Pode manter várias versões de artigos e objectos, activando as opções para criação de versões do DB2 Content Manager.

Por exemplo, se um artigo tiver a opção de criação de versões definida como always, é criada uma nova versão sempre que o artigo for actualizado, em vez de a versão original ou existente ser actualizada. Pode fazer algo para garantir que definiu a política de versões da forma que mais convém às suas necessidades. Por exemplo, pode desactivar o controlo de versões para atributos, e deixar este controlo de versões para o conteúdo de recurso. Pode também permitir a uma aplicação de utilizador final para especificar quando é criada uma nova versão.

Esta secção faculta exemplos para auxiliar o utilizador a trabalhar com propriedades de criação de versões.

Obter a política de controlo de versões de um tipo de artigo

Um artigo tem uma política de controlo de versão, que contém a propriedade de atribuição de versão para o artigo. Apresenta-se a seguir a lista das três propriedades de atribuição de versão disponíveis e o valor utilizado para representar cada propriedade na política de controlo de versões.

DK_ICM_VERSION_CONTROL_ALWAYS

Elaboração constante de versões.

DK_ICM_VERSION_CONTROL_NEVER

A criação de versões não é suportada neste tipo de artigo (predefinição).

DK_ICM_VERSION_CONTROL_BY_APPLICATION

A aplicação determina o esquema de criação de versões.

Exemplo: Java

```
short versionControlPolicy;
DKItemTypeDefICM item = newDKItemTypeDefICM();
....
versionControlPolicy = item.getVersionControl();
```

Exemplo: C++

```
short versionControlPolicy = 0;
DKItemTypeDefICM * item = NULL;
...
versionControlPolicy = item->getVersionControl();
```

Definir o controlo de versão para um tipo de artigo

Exemplo: Java

```
DKItemTypeDefICM item = new DKItemTypeDefICM();
short versionControl = DK_ICM_VERSION_CONTROL_ALWAYS;
....
item.setVersionControl(versionControl);
```

Exemplo: C++

```
DKItemTypeDefICM * item = NULL;
short versionControl = DK_ICM_VERSION_CONTROL_ALWAYS;
...
item->setVersionControl(versionControl);
```

Para obter um exemplo completo de acesso a informações de controlo de versões relativas a definições de tipo de artigo no sistema, consulte o exemplo formativo `SItemTypeRetrievalICMAPI`.

Obter o número máximo permitido de versões para um tipo de artigo

Exemplo: Java

```
short versionMax;
DKItemTypeDefICM item = new DKItemTypeDefICM();
....
versionMax = item.getVersionMax();
```

Exemplo: C++

```
short versionMax = 0;
DKItemTypeDefICM * item = NULL;
....
versionMax = item->getVersionMax();
```

Definir o número máximo permitido de versões permitido para um tipo de artigo

Neste exemplo, apenas dez versões de um artigo com base neste tipo de artigo são mantidas pelo sistema.

Exemplo: Java

```
short versionMax=10;
DKItemTypeDefICM item = new DKItemTypeDefICM();
...
item.setVersionMax(versionMax);
```

Exemplo: C++

```
short versionMax = 10;
DKItemTypeDefICM * item = NULL;
....
item->setVersionMax(versionMax);
```

Definir o valor do tipo de criação de versões para um tipo de artigo

Exemplo: C++

```
DKItemTypeDefICM * item = NULL;
short versionType = DK_ICM_ITEM_VERSIONING_OPTIMIZED;
...
item->setVersioningType(versioningType);
```

Trabalhar com pastas

Uma pasta é semelhante a outro artigo criado a partir de qualquer tipo de artigo no tempo de execução.

Ao criar um artigo de tipo semântico, obtém um atributo, DKFolder com um valor de uma recolha DKFolder. Este procedimento é denominado mecanismo de pasta. Pode adicionar DDOs dos conteúdos da pasta a esta recolha DKFolder. A recolha armazena as referências de DDO do componente raiz nos itens contidos na pasta.

Importante: As relações da pasta com os respectivos conteúdos são relações de ligação. As relações de ligação não são específicas de versões e não significa, propriedade. Por conseguinte, a relação da pasta com os respectivos conteúdos não é específica da versão, não sendo a pasta proprietária dos conteúdos. A pasta pode suportar o modelo de dados de itens padrão, itens de recurso e documentos, dependendo da classificação de tipo de artigo.

Embora a função de pasta seja flexível e funcione em qualquer tipo de artigo, é comum definir um tipo de artigo separada para as pastas. A combinação do mecanismo de pasta e de conteúdos de recurso ou partes de documentos é, por vezes, desnecessária e mais fácil de gerir como conceitos separados. As pastas são flexíveis, sendo possível decidir a quantidade de flexibilidade a utilizar. O mecanismo de pasta incorporado funciona em itens de qualquer tipo ou de qualquer classificação de tipo de artigo (não recurso, recurso e modelo de documentos). Contudo, a maioria dos utilizadores prefere ter um tipo de artigo separado para utilizar com a respectiva aplicação. As pastas são flexíveis, sendo possível decidir qual a quantidade de flexibilidade necessária.

Se preferir relações de pastas específicas de versões ou aplicar regras de eliminação alternada à relação, pode modelar o seu próprio mecanismo de pasta modelando um tipo de componente descendente que contenha um atributo de referência que seja específico de versão e possa ter regras de eliminação alternadas. A recolha descendente age como atributo de valor múltiplo, o que permite ao utilizador ter um número indiferenciado de atributos de referência que apontem para os conteúdos da pasta personalizada em vez de utilizar a recolha DKFolder incorporada.

Os procedimentos seguintes incluem fragmentos de códigos apenas para fins informativos.

Importante: Não copie estes fragmentos de código e não os cole directamente no programa de aplicação, uma vez que não funcionarão no formato actual. Para obter um exemplo completo de pastas que possa executar, consulte o exemplo SFolderICM no directório samples.

Criar uma pasta

O método DKDatastoreICM.createDDO() é utilizado para criar DDOs em tempo de execução.

O tipo de propriedade ou o tipo semântico de artigos tem de ser especificado como DKConstant.DK_CM_FOLDER ao criar artigos de pastas. Para obter mais informações, consulte o exemplo SItemCreationICM (no directório samples).

Um artigo de pasta é criado através da utilização do método `DKDatastoreICM.createDDO` com o tipo semântico `DK_CM_FOLDER`. Pode criar uma pasta especificando o `DKConstant.DK_CM_FOLDER` como segundo parâmetro para o método `createDDO`.

Exemplo: Java

```
DKDDO ddoFolder = dsICM.createDDO("S_simple", DKConstant.DK_CM_FOLDER);
```

Exemplo: C++

```
DKDDO* ddoFolder = dsICM->createDDO("S_simple", DK_CM_FOLDER);
```

Utilize o método `setData` para preencher o `ddoFolder`. Após a criação do DDO de pasta, este é guardado no servidor de conteúdos persistente.

Exemplo: Java

```
ddoFolder.add();
```

Exemplo: C++

```
ddoFolder->add();
```

Adicionar conteúdos a uma pasta

Os conteúdos de `DKFolder` têm de ser persistentes no servidor de conteúdos para que seja possível invocar o método `add()` ou `update()` na pasta DDO. Para tornar os conteúdos persistentes, invoque os respectivos métodos `DKDDO.add()`.

Para adicionar artigos a uma pasta, utilize a função `addElement()` de `DKFolder`. Após adicionar membros suficientes à pasta, pode invocar o método `add` ou `update` para guardar as relações pasta-conteúdo no armazenamento persistente. Esta acção agrupa todas as modificações de pasta e numa única chamada do servidor de bibliotecas. Ao adicionar artigos a uma pasta, não adicione várias cópias do mesmo artigo a `DKFolder`. A adição de várias cópias do mesmo artigo a `DKFolder` cria modificações divergentes ou duplicada, uma vez que todas as modificações de pastas são rastreadas.

Siga os passos seguintes para adicionar conteúdos a uma pasta. Tenha em atenção que uma pasta pode conter outra pasta (subpasta) como um dos seus artigos de conteúdo.

Exemplo de adição de conteúdos a uma pasta: Java:

Para adicionar conteúdos a uma pasta:

1. Crie três novos artigos. Estes artigos serão adicionados a uma pasta como sendo os seus conteúdos. Os conteúdos de pasta podem ser de qualquer tipo semântico.

```
DKDDO ddoDocument=dsICM.createDDO("S_simple",DKConstant.DK_CM_DOCUMENT);  
DKDDO ddoFolder2=dsICM.createDDO("S_simple",DKConstant.DK_CM_FOLDER);  
DKDDO ddoItem=dsICM.createDDO("S_simple",DKConstant.DK_CM_ITEM);
```

2. Utilize o método `setData` para preencher os DDOs. Os artigos que forem adicionados como conteúdos de pasta devem ser tornados persistentes no arquivo de dados.

```
ddoDocument.add();  
ddoItem.add();  
ddoFolder2.add();
```

.

3. Obtenha o atributo DKFolder do DDO de pasta anteriormente criado e tornado persistente.

```
short dataid = ddoFolder.dataId  
(DKConstant.DK_CM_NAMESPACE_ATTR,DKConstant.DK_CM_DKFOLDER);  
dkFolder = (DKFolder) ddoFolder.getData(dataid);
```

4. Dê saída da pasta antes de a actualizar (adicionando conteúdos).

```
dsICM.checkOut(ddoFolder);
```

5. Adicione os artigos anteriormente criados à pasta.

```
dkFolder.addElement(ddoDocument);  
dkFolder.addElement(ddoItem);  
dkFolder.addElement(ddoFolder2);
```

6. Guarde as alterações na pasta.

```
ddoFolder.update();
```

7. Dê entrada das alterações.

```
dsICM.checkIn(ddoFolder);
```

Exemplo de adição de conteúdos a uma pastas: C++:

Para adicionar os conteúdos a uma pasta:

1. Crie os artigos que irão ser adicionados à pasta.

```
DKDDO * ddoDocument = dsICM->createDDO("book", DK_CM_DOCUMENT);  
DKDDO * ddoFolder2 = dsICM->createDDO("book", DK_CM_FOLDER);  
DKDDO * ddoItem = dsICM->createDDO("book", DK_CM_ITEM);
```

2. Torne persistentes os artigos criados no arquivo de dados.

```
ddoDocument->add();  
ddoItem->add();  
ddoFolder2->add();
```

3. Obtenha o atributo DKFolder para a pasta.

```
DKFolder * dkFolder;  
short dataid = ddoFolder->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKFOLDER);  
if (dataid!=0) dkFolder =  
(DKFolder*)(dkCollection*) ddoFolder->getData(dataid);
```

4. Dê saída da pasta. É necessário dar saída da pasta antes de a actualizar.

```
dsICM->checkOut(ddoFolder);
```

5. Adicione os artigos criados à pasta.

```
dkFolder->addElement(ddoDocument);  
dkFolder->addElement(ddoItem);  
dkFolder->addElement(ddoFolder2); // Folders can contain sub-folders.
```

6. Actualize a pasta. Este procedimento também dá entrada da pasta (desbloqueia-a).

```
ddoFolder->update();
```

7. Dê entrada da pasta.

```
dsICM->checkIn(ddoFolder);
```

Remover conteúdo de uma pasta

Pode remover conteúdo de uma pasta.

Os itens podem ser removidos de uma pasta de uma das duas seguintes formas:

Remoção diferida

Um remoção diferida é efectuada utilizando o(s) método(s) `removeElementXX`. A remoção real é efectuada quando o DDO da pasta é actualizado e as várias chamadas do servidor de conteúdos são combinadas numa só.

Remoção imediata

Uma remoção imediata pode ser efectuada utilizando o método `dkFolder.removeMember`. Esta remoção é dispendiosa, uma vez que são efectuadas várias chamadas ao servidor de conteúdos.

Consulte o exemplo `SFolderICM` (no directório `samples`) para obter mais informações sobre como trabalhar com pastas.

Remover conteúdos de uma pasta: Java:

Para remover os conteúdos de uma pasta:

1. Dê saída do DDO de pasta da qual pretende remover um artigo.
`dsICM.checkOut(ddoFolder);`
2. Procure o artigo a remover criando um iterador para iterar o conteúdo da pasta. Neste caso, procure o DDO `docItem` anteriormente criado.

```
dkIterator iter = dkFolder.createIterator();
String itemPidString = ddoItem.getPidObject().pidString();
while(iter.more()) {
    // Move the pointer to next element and return that object.
    // Compare the PID strings of the DDO returned from the iterator
    // with the DDO that is to be removed.
    DKDDO ddo = (DKDDO) iter.next();
    if(ddo.getPidObject().pidString().equals(itemPidString)) {
        // The item to be removed is found.
        // Remove it (current element in the iterator)
        dkFolder.removeElementAt(iter);
    }
}
```
3. Torne a alteração persistente e dê entrada no DDO de pasta.
`ddoFolder.update();`
`dsICM.checkIn(ddoFolder);`

Remover conteúdos de uma pasta: C++:

1. Crie um artigo e adicione-o a uma pasta.

Nota:

A pasta foi criada anteriormente.

- ```
DKDDO * ddoItem = dsICM->createDDO("book", DK_CM_ITEM);
ddoItem->add();
DKFolder * dkFolder;
short dataid = ddoFolder->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKFOLDER);
if (dataid!=0) dkFolder =
 (DKFolder*)(dkCollection*) ddoFolder->getData(dataid);
dsICM->checkOut(ddoFolder);
dkFolder->addElement(ddoItem);
ddoFolder->update();
```
2. Dê explicitamente entrada da pasta.  
`dsICM->checkIn(ddoFolder);`
  3. Crie um iterador para a pasta.  
`dkIterator* iter = dkFolder->createIterator();`
  4. Utilize a iteração através do conteúdo da pasta até que seja localizado o artigo que irá ser removido. Remova o artigo.  

```
while(iter->more())
{
 DKDDO* ddo = (DKDDO*) iter->next()->value();
 if (((DKPidICM*) ddo->getPidObject())->pidString() ==
 ((DKPidICM*) ddoItem->getPidObject())->pidString())
 {
 //Found the element to be removed. Remove it
 }
}
```



```

 dkFolder->removeElementAt(*iter);
 // Now that we found the ddoItem, remove it.
 }

 delete(iter);

```

## Sugestões para adicionar e remover conteúdos de pastas

Ao adicionar e remover conteúdos de pastas, pode obter um melhor desempenho se utilizar métodos de elemento para agrupar várias actualizações e utilizar operações de criação e eliminação únicas e periódicas.

Pode facilitar o processo de adição, actualização e remoção de conteúdos de pastas se não esquecer os seguintes pontos:

### Utilize métodos de elemento para agrupar várias actualizações numa única actualização

A forma mais eficaz de adicionar e remover qualquer quantidade de conteúdos de uma pasta é agrupar todas as adições e modificações numa única actualização no servidor de bibliotecas. Utilize as funções `addelement` ou `removeelement` de DKFolder. Quando todas as modificações estiverem concluídas, invoque o método `add()` ou `update()` do DKDDO.

### Utilize a criação e eliminação única e periódica, uma vez que os métodos imediatos proporcionam um melhor desempenho

Se adicionar e remover artigos, um de cada vez, sem fazer qualquer outra alteração à pasta, os métodos imediatos funcionam melhor, dado que evitam operações adicionais que ocorrem numa `DKDDO::update()`. Para resultados imediatos, utilize as funções `DKFolder.addMember()` ou `removeMember()` ou `DKDatastoreExtICM's addToFolder()` ou `removeFromFolder()`, que torna as funções imediatamente persistentes à custa de uma chamada para o servidor de bibliotecas por cada alteração. No entanto, com este processo demora mais tempo se adicionar e eliminar mais do que um artigo.

## Obter os conteúdos de uma pasta

Ao obter itens existentes, o conjunto DKFolder não é incluído automaticamente. Pode pedir a enumeração de conteúdos de pastas que preenche o conjunto DKFolder e a lista de conteúdos mais recente.

**Importante:** Existe uma diferença entre a enumeração de conteúdos de pastas e a obtenção de dados para os conteúdos de uma pasta.

Primeiro, tem de enumerar os conteúdos da pasta preenchendo a recolha DKFolder com DDOs em branco e os respectivos PIDs preenchidos, que representam os conteúdos. Após a enumeração dos conteúdos de pasta, pode obter atributos ou outras informações para os conteúdos da pasta. Pode transferir a recolha DKFolder directamente para o método de obtenção multi-artigo (`DKDatastoreICM::retrieveObjects(dkCollection,DKNVPair[])`), uma vez que a recolha DKFolder é um `dkCollection` que contém DKDDOs.

Para obter a lista mais recente de conteúdos de pasta, é necessário obter ligações de partida, uma vez que as relações de pastas são relações de ligações. Opcionalmente, pode limitar os tipos de ligação obtidos. Se pedir apenas o tipo de ligação DKFolder, obterá apenas a recolha DKFolder e não outros tipos de ligação ou recolhas de ligações. Se não forem encontrados quaisquer conteúdos de pasta, é apresentado uma recolha DKFolder vazio no DDO. As seguintes opções de obtenção aplicam-se à enumeração de conteúdos de pasta:

- DKRetrieveOptionsICM::linksOutbound()
- DKRetrieveOptionsICM::linksTypeFilter() (limita, opcionalmente, o tipo)
- DKRetrieveOptionsICM::linksLevelTwoCount()
- DKRetrieveOptionsICM::linksLevelTwo() (método apenas para Java)

Na referência de programação relativa a opções de obtenção, é possível encontrar uma explicação completa de considerações de desempenho, descrições e explicações da localização exacta na qual os dados são obtidos para o DDO. Consulte o método DKRetrieveOptionsICM::linksOutbound() no manual *Application Programming Reference*.

**Importante:** Se estiver a obter a enumeração dos conteúdos de pasta devido ao facto de a funcionalidade de pasta depender da pasta de tipo semântico, terá de obter a propriedade de tipo semântico. Pode incluir atributos ou, caso pretenda, filtrar a obtenção de atributos para apenas atributos de sistema. Consulte o método DKRetrieveOptionsICM::attributeFilters() no manual *Application Programming Reference*.

O exemplo seguinte demonstra como obter a lista de conteúdos de pasta.

#### Exemplo: Java

```
DKRetrieveOptionsICM dkRetrieveOpts =
DKRetrieveOptionsICM.createInstance(dsICM);
dkRetrieveOpts.baseAttributes(true);
dkRetrieveOpts.linksOutbound(true);
ddoFolder.retrieve(dkRetrieveOpts.dkNVPair());
```

#### Exemplo: C++

```
DKRetrieveOptionsICM* dkRetrieveOpts =
DKRetrieveOptionsICM.createInstance(dsICM);
dkRetrieveOpts->baseAttributes(TRUE);
dkRetrieveOpts->linksOutbound(TRUE);
ddoFolder->retrieve(dkRetrieveOpts->dkNVPair(),
dkRetrieveOpts->dkNVPairLength());
```

### Obter todas pastas que contêm um DDO específico

Várias pastas podem conter o mesmo artigo (DDO), uma vez que as relações de pastas não implicam propriedade. O processo para determinar quais as pastas que contêm actualmente um DDO específico é semelhante a obter e aceder à recolha de DKFolder, excepto no facto de as origens das pastas estarem armazenadas numa recolha diferente e serem solicitadas através de uma opção de obtenção diferente.

No caso de qualquer DDO de componente raiz de um artigo, pode verificar se está enumerado em quaisquer pastas obtendo as ligações de chegada e pedindo as origens de pastas de chegada. Opcionalmente, pode limitar a obtenção de ligações ao tipo de ligação de DKFolder. As opções de obtenção que se seguem aplicam-se à enumeração de origens de pastas que contêm o DDO actual:

- DKRetrieveOptionsICM::linksInbound()
- DKRetrieveOptionsICM::linksInboundFolderSources()
- DKRetrieveOptionsICM::linksTypeFilter() (limita, opcionalmente, o tipo)

Para enumerar as origens de pastas, ao contrário do que ocorre com a enumeração de conteúdos de pastas, não é necessário obter o tipo semântico do DDO actual. No entanto, tal como a recolha de DKFolder, a recolha de origens de pastas contém

DDOs em branco com PIDs preenchidos. Pode, posteriormente, invocar o método `retrieve` de artigos múltiplos e transferir a `dkCollection` das origens de pastas.

Para obter mais informações sobre considerações de desempenho e detalhes sobre a localização exacta para a qual a recolha de origens de pastas é recuperada (incluindo todos os nomes e tipos de valores), consulte o método `DKRetrieveOptionsICM::linksInboundFolderSources(Boolean)`, no manual *Application Programming Reference*.

Os exemplos seguintes demonstra a obtenção e o acesso à lista de origens de pastas que contém o DDO actual. Pode ainda utilizar o método `DKDatastoreExtICM::GetFoldersContainingDDO()`, embora seja menos eficiente. Consulte o exemplo `SFolderICM` (no directório `samples`) para obter mais informações sobre como trabalhar com pastas.

#### **Obter todas as pasta que contém um DDO específico: Java:**

Para obter todas as pastas que contém um DDO específico:

1. Se tiver um DDO denominado `ddo`, solicite uma lista de origens de pasta que contenham este DDO.

```
DKRetrieveOptionsICM dkRetrieveOpts =
DKRetrieveOptionsICM.createInstance(dsICM);
dkRetrieveOpts.linksInbound(true);
dkRetrieveOpts.linksInboundFolderSources(true);
dkRetrieveOpts.linksTypeFilter(DKConstantICM.DK_ICM_LINKTYPENAME_DKFOLDER);
ddo.retrieve(dkRetrieveOpts.dkNVPair());
```

2. Obtenha a recolha de origens de pasta

```
dkCollection foundSourceFoldersColl = null;
propId = ddo.propertyId(DKConstant.DK_CM_PROPERTY_FOLDER_SOURCES);
if(propId > 0)
 foundSourceFolderColl = (dkCollection) ddo.getProperty(propId);
```

3. Examine os resultados.

```
if(foundSourceFolderColl==null)
 System.out.println("No source folders found.");
else
 System.out.println("Found '"+foundSourceFolderColl.cardinality()
 +"' source folders that contain this item.");
// Want to access the source folders?
dkIterator iter = foundSourceFolderColl.createIterator();
while(iter.more()){
 DKDDO folder = (DKDDO) iter.next();
}
```

#### **Obter todas as pastas que contém um DDO específico: C++:**

Para obter todas as pastas que contém um DDO específico:

1. Se tiver um DDO denominado `ddo`, solicite uma lista de origens de pasta que contenham este DDO.

```
DKRetrieveOptionsICM* dkRetrieveOpts =
DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOpts->linksInbound(TRUE);
dkRetrieveOpts->linksInboundFolderSources(TRUE);
dkRetrieveOpts->linksTypeFilter(DK_ICM_LINKTYPENAME_DKFOLDER);
ddo->retrieve(dkRetrieveOpts->dkNVPair(),dkRetrieveOpts->dkNVPairLength());
```

2. Obtenha a recolha de origens de pasta

```
dkCollection* foundSourceFoldersColl = NULL;
propId = ddo->propertyId(DK_CM_PROPERTY_FOLDER_SOURCES);
if(propId > 0)
 foundSourceFolderColl = (dkCollection*) ddo->getProperty(propId);
```

3. Examine os resultados

```

if(foundSourceFolderColl==NULL)
 cout << "No source folders found." << endl;
else
 cout << "Found '" << foundSourceFolderColl->cardinality() <<"'
 source folders that contain this item." << endl;
// Want to access the source folders?
std::auto_ptr iter(foundSourceFolderColl->createIterator());
while(iter->more()){
 DKDDO* folder = (DKDDO*)(dkDataObjectBase*)(iter->next());
}

```

## Definir ligações entre itens

Pode utilizar ligações para associar um artigo origem a um destino com um artigo de descrição facultativo.

Defina ligações em objectos DKLink, nos quais pode especificar os seguintes tipos de elementos de ligação:

*Tabela 14. Estrutura de dados de ligação*

| DKLink                  | Definição                           |
|-------------------------|-------------------------------------|
| Nome do Tipo de Ligação | O tipo de ligação.                  |
| Origem                  | O artigo origem de uma ligação.     |
| Destino                 | O artigo destino de uma ligação.    |
| Artigo de Ligação       | A descrição do artigo. Facultativo. |

Uma vez que uma ligação representa um relação de um para muitos, é representada num DDO apor um artigo de dados no espaço de nomes LINK, com o valor DKLinkCollection. Para obter mais informações relativas ao trabalho com ligações, consulte o exemplo SLinksICM no directório samples.

Uma ligação não pertence à origem nem ao destino. Uma ligação apenas liga uma origem e um destino. Por exemplo, se a origem A for ligada ao destino B, A será sempre a origem e B será sempre o destino, independentemente do DDO a que A ou B estão a ser referenciados.

## Exemplo de definição de DKLink

Na memória, tanto os DDOs origem como os DDOs destino contêm cópias do mesmo objecto DKLink. Uma vez que o objecto DKLink contém uma referência à origem e ao destino, um DDO que contenha uma ligação também contém uma ligação que o referencia a si próprio, bem como a outro DDO. Por exemplo, se a Origem A estiver ligada ao Destino B, tanto A como B contêm a mesma ligação, como mostra o exemplo da Tabela 15.

*Tabela 15. Exemplo de definição de DKLink*

| DKLink Definido de A para B | DDO A         | DDO B         |
|-----------------------------|---------------|---------------|
| A origem é A                | A origem é A  | A origem é A  |
| O destino é B               | O destino é B | O destino é B |

Ao adicionar ou remover ligações da memória persistente, o utilizador apenas tem de executar a operação num dos dois artigos, no origem ou no destino. A ligação é automaticamente actualizada para o outro artigo.

Para obter mais informações relativas a ligações e um exemplo de ligações completo que possa executar, consulte o exemplo `SLinksICM` no directório `samples`.

**Não se esqueça:** Antes de efectuar uma chamada aos métodos `add()` ou `update()` de DDO, todos os itens associados às ligações, têm de ser persistentes.

### **Ligações de chegada e partida**

Ao utilizar ligações para referenciar um DDO específico, origem ou destino, pode considerar as ligações como sendo de chegada ou de partida.

Ao considerar uma ligação segundo a perspectiva de um determinado DDO, caso esse DDO seja o destino, este último considera-a uma ligação de chegada. Entretanto, segundo a perspectiva do DDO no outro extremo dessa ligação, a mesma ligação é de partida.

No exemplo em Tabela 15 na página 127, a ligação do DDO A ao DDO B é de partida, enquanto a mesma ligação ao DDO B é de chegada.

### **Nomes de tipos de ligação**

As relações entre ligações possuem nomes.

Dentro de um DDO, estão agrupadas ligações em recolhas de ligações. Existem nomes de tipos de ligação definidos pelo sistema e o utilizador pode definir os seus próprios nomes. Pode utilizar todos os nomes de tipos de ligação definidos pelo utilizador ou os nomes definidos pelo sistema. Os seguintes nomes de tipo de artigo definidos pelo sistema incluem:

#### **O nome de tipo de ligação contém**

##### **Constante de Java**

`DKConstant.DK_ICM_LINKTYPENAME_CONTAINS`

##### **Constante de C++**

`DK_ICM_LINKTYPENAME_CONTAINS`

#### **Nome do tipo de ligação: DKFolder**

##### **Constante de Java**

`DKConstant.DK_ICM_LINKTYPENAME_DKFOLDER`

##### **Constante de C++**

`DK_ICM_LINKTYPENAME_DKFOLDER`

O nome de tipo de ligação `DKFolder` é fornecido e utilizado pelo sistema para gerir pastas. O objecto `DKFolder` é uma interface simplificada para uma recolha de ligações de partida para facilitar o trabalho com pastas. Deste modo, não deverá utilizar o nome de tipo de ligação `DKFolder` para definir ou eliminar ligações. Não deverá utilizar o nome de tipo de ligação `DKFolder` de modo algum com uma `DKLinkCollection`. No entanto, terá de especificar o nome de tipo de ligação `DKFolder` de modo a pesquisar pastas utilizando ligações.

Consulte o exemplo `SLinksICM`, no directório `IBMCMROOT/samples/cpp/icm` ou `IBMCMROOT/samples/java/icm`, para obter mais informações sobre ligações. Para obter informações relativas à criação de tipos de ligação definidos pelo utilizador, consulte o exemplo `SLinkTypeDefinitionCreationICM`.

## Obter itens ligados

Existe uma diferença entre a enumeração de ligações e a obtenção de dados de itens ligados. Para obter atributos e outras informações, preencha `DKLinkCollection` com `DKLinks` e, em seguida, invoque o método de obtenção. O método de múltiplos itens `retrieve` suporta igualmente colecções de ligações e o utilizador não tem de interagir a lista de colecção de ligações e obter cada ligação individualmente.

Primeiro, terá de enumerar as ligações preenchendo as recolhas `DKLinkCollection` com `DKLinks`, cada uma das quais contém DDOs em branco e respectivos PIDs completos, que representam a origem, o destino e o descritor ligados. Em seguida, se pretender, pode efectuar uma chamada subsequente para obter atributos ou outras informações para os próprios itens ligados. Pode transferir ou criar um `dkCollection` próprio, criando um novo `DKSequentialCollection`, iterar todas as ligações em todas as recolhas de ligações pretendidas e adicionar outro DDO (a referência do DDO actual é apresentada como a origem ou o destino, dependendo da direcção de chegada ou de partida) e, em seguida, transferir a nova recolha directamente para a obtenção multi-artigo (`DKDatastoreICM::retrieveObjects(dkCollection,DKNVPair[])`).

O método `DKDatastoreICM::retrieveObjects(dkCollection)` processa qualquer `dkCollection` de objectos `DKDDO`. O utilizador não tem de iterar os objectos `DKLinkCollection`, obtendo os itens de origem e de destino individualmente. Contudo, o utilizador tem de fornecer `DKDatastoreICM::retrieveObjects(dkCollection)` com uma colecção de objectos `DKDDO`. Visto que o `DKLinkCollection` contém objectos `DKLink`, o utilizador terá de criar as suas próprias `dkCollection` (utilizando `DKSequentialCollection`) e inserir os objectos `DKDDO` que pretende obter. Em seguida, passe `dkCollection` para o método de artigos múltiplos `retrieve`.

**Não se esqueça:** O objecto `DKLink` contém uma referência aos objectos `DKDDO` do artigo de origem e de destino.

Para além disso, um dos dois objectos `DKDDO` (origem ou destino) é o mesmo `DKDDO` do artigo já obtido, a partir do qual obteve as colecções de ligações. O utilizador necessita de obter apenas os `DKDDOs` não obtidos, como o destino de uma ligação externa ou a origem de uma ligação interna. Para obter mais informações sobre ligações internas e externas, consulte a amostra de aprendizagem da API `SLinksICM`.

Existem várias considerações e opções no que respeita à obtenção da lista de ligações. Existem considerações muito importantes e detalhes completos documentados para cada opção de ligações. Consulte as seguintes opções relacionadas com ligações para obter detalhes completos no manual *Application Programming Reference*. Pode obter apenas ligações de chegada, de partida, de ambas as direcções, limitar a obtenção a apenas um tipo de ligação específico, obter dois níveis de ligações de uma só vez (apenas para Java), obter a contagem de ligações para o segundo nível, etc.

- `DKRetrieveOptions::linksOutbound()`
- `DKRetrieveOptions::linksInbound()`
- `DKRetrieveOptions::linksInboundFolderSources()`
- `DKRetrieveOptions::linksDescriptors()`
- `DKRetrieveOptions::linksTypeFilter()`
- `DKRetrieveOptions::linksLevelTwoCount()`

- `DKRetrieveOptions::linksLevelTwo()` (apenas para Java)

---

## Trabalhar com controlo de acessos

Se tiver acesso ao cliente de administração do sistema do DB2 Content Manager ou às APIs do DB2 Content Manager para escrever um programa de administração próprio, pode utilizar as funções de controlo de acessos para controlar o acesso às informações do sistema do DB2 Content Manager.

As várias APIs de controlo de acessos permitem ao utilizador controlar o acesso a todo o sistema, a um conjunto de operações intimamente relacionado do sistema ou a um artigo individual.

Algumas das tarefas que pode concluir utilizando as APIs de controlo de cesso incluem:

- Crie conjuntos de privilégios para agrupar privilégios
- Associe uma lista de acções com informações do sistema
- Conceda autorização a utilizadores para executar acções nas informações do servidor de bibliotecas

### Criar um privilégio

Um privilégio é representado pela classe `DKPrivilegeICM`.

Os passos e exemplos de código seguintes mostram como criar um novo objecto de privilégio, torná-lo persistente e obter um privilégio. Para criar um privilégio, siga os seguintes passos:

#### Criar um privilégio: Java

1. Estabelecer uma ligação a um arquivo de dados

```
DKDatastoreICM ds = new DKDatastoreICM();
ds.connect("ICMNLSDDB","icmadmin",password,"");
dkDatastoreDef dsDef = (dkDatastoreDef)ds.datastoreDef();
DKDatastoreAdminICM dsAdmin = (DKDatastoreAdminICM)
dsDef.datastoreAdmin();
```
2. Obtenha um objecto `DKAuthorizationMgmtICM`. Esta classe processa tarefas de gestão de autorizações.

```
DKAuthorizationMgmtICM aclMgmt = (DKAuthorizationMgmtICM)
dsAdmin.authorizationMgmt();
```
3. Criar um novo objecto de privilégio.

```
DKPrivilegeICM priv = new DKPrivilegeICM(ds);
```
4. Atribuir um nome ao objecto de privilégio.

```
priv.setName("UserPriv");
```
5. Atribuir uma descrição ao objecto de privilégio.

```
priv.setDescription("Este é um privilégio definido pelo utilizador");
```
6. Adicionar o novo privilégio objecto do gestor de autorizações.

```
aclMgmt.add(priv);
```
7. Obter o privilégio recém-criado do objecto do gestor de autorizações.

```
dkPrivilege aPriv = aclMgmt.retrievePrivilege("UserPriv");
```
8. Apresentar informações relativas ao objecto de privilégio.

```
System.out.println("privilege name = " + aPriv.getName());
System.out.println("privilege description = " +
aPriv.getDescription());
```



## Criar um privilégio: C++

1. Criar o objecto de arquivo de dados, bem como os objectos relacionados com a gestão da administração.

```
//Create the datastore object
DKDatastoreICM * ds = new DKDatastoreICM();
//Connect to the underlying datastore
ds->connect("ICMNLSDb", "icmdmin", "password", "");
//Retrieve the datastore definition object (used to access
//and manipulate CM meta-data)
dkDatastoreDef * dsDef = (dkDatastoreDef *) ds->datastoreDef();
//Create the class used to represent and process datastore
//administration functions.
DKDatastoreAdminICM * dsAdmin = (DKDatastoreAdminICM *)
 dsDef->datastoreAdmin();
//Retrieve the class used to represent and manage the authorization
//related functionality of the ICM datastore
DKAuthorizationMgmtICM * aclMgmt = (DKAuthorizationMgmtICM *)
 dsAdmin->authorizationMgmt();
```

2. Crie um novo objecto de privilégio e defina as respectivas propriedades.

```
DKPrivilegeICM * priv = new DKPrivilegeICM(ds);
//Set the name of the privilege object
priv->setName("UserPriv");
//Set the privilege description
priv->setDescription("This is user-defined privilege");
```

3. Adicionar o privilégio ao arquivo de dados através do objecto de gestão de autorizações.

```
aclMgmt->add(priv);
```

## Criar um conjunto de privilégios

Um conjunto de privilégios é representado pela classe DKPrivilegeSetICM.

Antes de poder iniciar a criação de conjuntos de privilégios, deve estar ligado a um servidor de conteúdos. Para criar um conjunto de privilégios, execute os passos apresentados nos tópicos seguintes:

### Criar um conjunto de privilégios: Java

1. Criar novos privilégios (ou obter privilégios existentes) para adicionar ao novo conjunto de privilégios.

```
DKPrivilegeICM priv_1 = new DKPrivilegeICM(ds);
priv_1.setName("ItemCheckOut");
DKPrivilegeICM priv_2 = new DKPrivilegeICM(ds);
priv_2.setName("ItemQuery");
DKPrivilegeICM priv_3 = new DKPrivilegeICM(ds);
priv_3.setName("ItemAdd");
```

2. Criar um novo conjunto de privilégios.

```
DKPrivilegeSetICM privSet1 = new DKPrivilegeSetICM(ds);
```

3. Atribuir um nome ao conjunto de privilégios.

```
privSet1.setName("UserPrivSet");
```

4. Atribuir uma descrição ao conjunto de privilégios.

```
privSet1.setDescription("This is a user-defined priv set");
```

5. Adicionar os privilégios ao conjunto de privilégios.

```
privSet1.addPrivilege(priv_1);
privSet1.addPrivilege(priv_2);
privSet1.addPrivilege(priv_3);
```

6. Adicionar o conjunto de privilégios recém-criado ao gestor de autorizações.

```
AclMgmt.add(privSet1);
```



7. Apresentar informações relativas ao conjunto de privilégios recém-criado.

```
DKPrivilegeSetICM aPrivSet = (DKPrivilegeSetICM)
 aclMgmt.retrievePrivilegeSet("UserPrivSet");
System.out.println("privilege set name = " + aPrivSet.getName());
System.out.println("privilege set description=" +
 aPrivSet.getDescription());
dkCollection coll = aPrivSet.listPrivileges();
dkIterator iter = coll.createIterator();
while (iter.more()) {
 DKPrivilegeICM _priv = (DKPrivilegeICM) iter.next();
 System.out.println(" privilege name = " + _priv.getName());
}
```

## Criar um conjunto de privilégios: C++

1. Criar o objecto de arquivo de dados, bem como os objectos relacionados com a gestão da administração.

```
//Create the datastore object
DKDatastoreICM * ds = new DKDatastoreICM();
//Connect to the underlying datastore
ds->connect("ICMNLSDb", "icmdmin", "password", "");
//Retrieve the datastore definition object
//(used to access and manipulate CM meta-data)
dkDatastoreDef * dsDef = (dkDatastoreDef *) ds->datastoreDef();
//Create the class used to represent and process
// datastore administration functions.
DKDatastoreAdminICM * dsAdmin = (DKDatastoreAdminICM *)
 dsDef->datastoreAdmin();
//Retrieve the class used to represent and manage the authorization
//related functionality of the ICM datastore
DKAuthorizationMgmtICM * aclMgmt = (DKAuthorizationMgmtICM *)
 dsAdmin->authorizationMgmt();
```

2. Criar três privilégios e definir as suas propriedades.

```
DKPrivilegeICM * priv_1 = new DKPrivilegeICM(ds);
priv_1->setName("ItemCheckOut");
DKPrivilegeICM * priv_2 = new DKPrivilegeICM(ds);
priv_2->setName("ItemQuery");
DKPrivilegeICM * priv_3 = new DKPrivilegeICM(ds);
priv_3->setName("ItemAdd");
```

3. Criar um novo conjunto de privilégios e definir as suas propriedades.

```
DKPrivilegeSetICM * privSet1 = new DKPrivilegeSetICM(ds);
privSet1->setName("UserPrivSet");
privSet1->setDescription("This is a user-defined priv set");
```

4. Adicionar os privilégios criados ao novo conjunto de privilégios.

```
privSet1->addPrivilege(priv_1);
privSet1->addPrivilege(priv_2);
privSet1->addPrivilege(priv_3);
```

5. Adicionar o conjunto de privilégios ao arquivo de dados com o objecto de gestão de autorizações.

```
aclMgmt->add(privSet1);
```

## Apresentar propriedades do conjunto de privilégios

O exemplo indica como apresentar propriedades de um conjunto de privilégios.

### Exemplo: C++

```
//Retrieve a privilege set using its name
DKPrivilegeSetICM * sPrivSet = (DKPrivilegeSetICM *)
 aclMgmt->retrievePrivilegeSet("UserPrivSet");
//Display privilege set properties
cout<<"privilege set name = "<< (char *)sPrivSet->getName() << endl;
cout<<"priv set descrip="<< (char *)sPrivSet->getDescription() << endl;
```

```
//Retrieve the list of privileges that are a part of this privilege set
dkCollection * coll = sPrivSet->listPrivileges();
dkIterator * iter = coll->createIterator();
while (iter->more())
{
 DKPrivilegeICM* _priv = (DKPrivilegeICM *) (void *) (*iter->next());
 cout<<" privilege name = "<< (char *)_priv->getName() << endl;
}
delete(iter);
```

## Definir uma lista de controlo de acessos (ACL)

O modelo de controlo de acessos do DB2 Content Manager é aplicado ao nível da entidade controlada.

Uma entidade controlada é uma unidade de dados de utilizador protegidos. Uma entidade controlada pode ser um artigo individual, um tipo de artigo ou uma biblioteca completa. As operações sobre entidades controladas são reguladas por uma ou mais regras de controlo. A ACL é o contentor destas regras de controlo. A classe DKAccessControlListICM representa uma ACL. Cada entidade controlada num sistema DB2 Content Manager deve estar associada a uma ACL.

O exemplo seguinte demonstra como definir uma ACL.

### Definir uma lista de controlo de acessos (ACL): Java

1. Defina um novo objecto DKACLData. Uma classe DKACLData é utilizada para manter dados relacionados com a ACL.  

```
DKACLData aclData1 = new DKACLData();
```
2. Defina o nome do grupo de utilizadores para a ACL  

```
aclData1.setUserGroupName("ICMADMIN");
```
3. Defina o tipo patron da ACL.  

```
aclData1.setPatronType(DK_CM_USER_KIND_USER);
```
4. Defina o conjunto de privilégios associados a esta ACL  

```
aclData1.setPrivilegeSet(privSet_1);
```
5. Crie outro objecto DKACLData.  

```
DKACLData aclData2 = new DKACLData();
aclData2.setUserGroupName("ICMPUBLIC");
aclData2.setPatronType(DK_CM_USER_KIND_GROUP);
aclData2.setPrivilegeSet(privSet_2);
```
6. Crie uma nova ACL. Um DKAccessControlListICM representa uma ACL do CM. V8  

```
DKAccessControlListICM acl1 = new DKAccessControlListICM(ds);
```
7. Atribua um novo nome à nova ACL  

```
acl1.setName("UserACL");
```
8. Atribua uma descrição à nova ACL  

```
acl1.setDescription("This is a user-defined ACL");
```
9. Adicione objectos de dados de ACL criados anteriormente à ACL  

```
acl1.addACLData(aclData1);
acl1.addACLData(aclData2);
```
10. Adicione a nova ACL ao gestor de autorizações, aclMgmt.add(acl1);
11. Obtenha e apresenta informações sobre a nova ACL  

```
DKAccessControlListICM acl_1 = (DKAccessControlListICM)
aclMgmt.retrieveAccessControlList("UserACL");
System.out.println("ACL name = " + acl_1.getName());
System.out.println(" desc = " + acl_1.getDescription());
```

```

dkCollection coll = acl_1.listACLData();
dkIterator iter = coll.createIterator();
while (iter.more()) {
 DKACLData aclData = (DKACLData) iter.next();
 DKPrivilegeSetICM _privSet = (DKPrivilegeSetICM)
 aclData.getPrivilegeSet();
 System.out.println(" PrivSet name = " + _privSet.getName());
 System.out.println(" PrivSet desc = " + _privSet.getDescription());
 String usrGrpName = aclData.getUserGroupName();
 System.out.println(" UserGroupName = " + usrGrpName);
 short patronType = aclData.getPatronType();
 System.out.println(" Patron type = " + patronType);
}

```

O programa exemplo completo encontra-se no directório samples.

### Definir uma lista de controlo de acessos (ACL): C++

1. Defina novos objectos DKACLData. Cada objecto DKACLData é utilizado para manter dados relacionados com a ACL.

```

DKACLData * aclData1 = new DKACLData();
// set the name of the user group to be associated with this ACL
aclData1->setUserGroupName("ICMADMIN");
// Set the ACL patron type. Can be one of 3 possible
//values:DK_CM_USER_KIND_USER or DK_CM_USER_KIND_GROUP or
//DK_CM_USER_KIND_PUBLIC.
aclData1->setPatronType(DK_CM_USER_KIND_USER);
// Set the privilege set associated with the ACL.
DKPrivilegeSetICM * privSet_1 = (DKPrivilegeSetICM *)
 aclMgmt->retrievePrivilegeSet("UserPrivSet");
aclData1->setPrivilegeSet(privSet_1);

```

2. Crie outro objecto DKACLdata.

```

DKACLData * aclData2 = new DKACLData();
aclData2->setUserGroupName("ICMPUBLIC");
aclData2->setPatronType(DK_CM_USER_KIND_GROUP);
DKPrivilegeSetICM * privSet_2 = (DKPrivilegeSetICM *)
 aclMgmt->retrievePrivilegeSet("PublicPrivSet");
aclData2->setPrivilegeSet(privSet_2);

```

3. Criar uma nova ACL. Definir valores de propriedade para esta nova ACL.

```

DKAccessControlListICM * acl1 = new DKAccessControlListICM(ds);
//Assign a new name to the newly-created ACL.
acl1->setName("UserACL");
// Assign a description to the newly-created ACL.
acl1->setDescription("This is a user-defined ACL");

```

4. Adicione os objectos de dados da ACL, criada no passo três, à ACL.

```

acl1->addACLData(aclData1);
acl1->addACLData(aclData2);

```

5. Adicionar a ACL, criada no passo três, ao gestor de autorizações.

```

aclMgmt->add(acl1);

```

O programa exemplo completo encontra-se no directório samples.

### Obter e apresentar informações sobre a ACL

Pode obter e apresentar informações sobre a ACL obtendo, primeiro a ACL e, em seguida, os dados da ACL.

Para obter e apresentar as informações sobre a ACL, conclua os passos seguintes.

1. Obtenha a ACL a partir do objecto de gestão de autorizações com o nome da ACL. Para C++:

```

DKAccessControlListICM * acl_1 = (DKAccessControlListICM *) aclMgmt->
 retrieveAccessControlList("UserACL");
cout<<"ACL name = "<< (char *)acl_1->getName() << endl;
cout<<" desc = "<< (char *)acl_1->getDescription() << endl;
2. Obter os dados da ACL associados à ACL. Para C++:
dkCollection * coll = acl_1->listACLData();
dkIterator * iter = coll->createIterator();
char szpatronType[12] = {0x00};
while (iter->more())
{
 DKACLData * aclData = (DKACLData *) (void *) (*iter->next());
 DKPrivilegeSetICM * _privSet = (DKPrivilegeSetICM *) aclData->
 getPrivilegeSet();
 cout<<"privSet name = "<< (char *) _privSet->getName() << endl;
 cout<<"privSet desc="<< (char *) _privSet->getDescription()
 << endl;
 DKString usrGrpName = aclData->getUserGroupName();
 cout<<" UserGroupName = "<< (char *)usrGrpName << endl;
 short patronType = aclData->getPatronType();
 sprintf(szpatronType, "
 cout<<" Patron type = "<< szpatronType << endl;
}
delete(iter);

```

## Atribuir uma ACL a um tipo de artigo

Após a criação de uma ACL, pode associá-la a um tipo de artigo específico.

De seguida são apresentados os passos que terá de seguir para atribuir uma ACL a um tipo de artigo:

1. Configura um novo tipo de artigo.

### Java

```

DKItemTypeDefICM it = new DKItemTypeDefICM(dsICM);
//Assign a name to this item type
it.setName("TextResource1");
//Assign a description to this item type
it.setDescription("CMv8 Text Resource Item Type.");
//Assign an ACL code to this item type
it.setItemTypeACLCode((int)DK_ICM_ITEMACL_BIND_AT_ITEM);
....
it.add(); // make the item type persistent
...

```

### C++

```

DKItemTypeDefICM * itemType = new DKItemTypeDefICM(dsICM);
// Assign a name to this item type.
itemType->setName("TextResource1");
// Assign a description to this item type.
itemType->setDescription("CMv8 Text Resource Item Type.");

```

2. Obter os dados da ACL associados à ACL.

### Java

```

int itemTypeACLCode = it.getItemTypeACLCode();
// By setting the item type's ACL flag to TRUE(1) we confirm that ACL
// binding is at the item type level. By setting the item type's ACL
// flag was set to FALSE(0), we would be saying that the ACL binding
// is not at the item type level
it.setItemLevelACLFlag(1); // - true
//Determine whether the ACL binding is at the item type level or not
int itemTypeACLFlag = it.getItemLevelACLFlag();

```

C++

```
itemType->setItemTypeACLCode((long) 1);
// By setting the item type's ACL flag to TRUE (1),
//we confirm that ACL binding is at the item type level.
//By setting the item type's ACL flag to FALSE(0),
// we would be saying that the ACL binding is not at
// the item type level.
itemType->setItemLevelACLFlag((short)1);
itemType->add();
// make the item type persistent.
```

O programa exemplo completo está disponível no directório samples.

## Atribuir uma ACL a um artigo

Para activar o controlo de acessos ao nível do artigo, deve associar um ACL ao artigo.

Para tal, é necessário criar o artigo utilizando o método add() no DDO, conforme indicado no seguinte fragmento de código. No fragmento de código seguinte, parte-se do princípio que já possui uma ligação ao servidor de conteúdos e que criou a ACL. O programa completo encontra-se no directório de samples.

### Atribuir uma ACL a um artigo: Java

1. Adicionar uma nova propriedade ao DDO. Parta do princípio que uma ACL (Lista de controlo de acesso) denominada "MyACL" // já existe no sistema. Esta propriedade será denominada DK\_ICM\_PROPERTY\_ACL.  
`int propId =ddoItem.addProperty(DK_ICM_PROPERTY_ACL);`
2. Defina a ACL anteriormente criada (ou obtida) como o valor desta propriedade.  
`ddoItem.setProperty(propId,"MyACL");`
3. Torne persistente o DDO no arquivo de dados.  
`ddoItem.add();`

### Atribuir uma ACL a um artigo: C++

1. Crie um novo artigo (DDO) baseado num tipo de artigo existente.  
`DKDDO * ddoItem = dsICM->createDDO("book", DK_CM_ITEM);`
2. Parta do princípio de que uma ACL (Lista de Controlo de Acessos) com o nome "MyACL" já existe no sistema.
3. Adicione uma nova propriedade ao DDO. Esta propriedade será denominada DK\_ICM\_PROPERTY\_ACL.  
`ushort propId = ddoItem->addProperty(DK_ICM_PROPERTY_ACL);`
4. Defina a ACL, criada anteriormente, como valor para esta propriedade.

**Importante:** Um utilizador pode optar por obter uma ACL existente e utilizá-la como a ACL deste artigo.

```
ddoItem->setProperty(propId, "MyACL");
```

5. Torne persistente o DDO no arquivo de dados.  
`ddoItem->add();`

## Utilizar ACLs de vistas em operações de eliminação, adição ou remoção de ligações

Os tipos de itens podem pertencer a uma vista base ou a mais vistas.

A cada vista de tipo de artigo é atribuída uma lista de controlo de acessos (ACL) própria. Apenas pode utilizar a ACL de vista quando a associação de ACL é de tipo de artigo ou misto com o sinalizador de ACL de nível de artigo definido como False. Deste modo, para utilizar uma ACL de vista, a ACL tem de ser seleccionada na ACL de vista de tipo de artigo em vez de ser seleccionada na ACL de tipo de artigo. Se não for especificada qualquer vista, a ACL de vista base (que é a ACL de tipo de artigo) é seleccionada.

Ao adicionar ou remover uma ligação ou ao eliminar um artigo, a ACL da vista é seleccionada. Deste modo, para estabelecer ligação a um artigo, os utilizadores com acesso ao artigo a partir do qual pretende estabelecer ligação têm igualmente de ter acesso ao artigo ao qual pretende estabelecer ligação.

Se um utilizador fizer parte de uma ACL (tal como a ACL 1) que concede ao utilizador acesso de leitura e escrita para todos os itens de um tipo de artigo, o mesmo utilizador pertencer a uma segunda ACL (tal como a ACL 2) que concede ao utilizador apenas acesso de leitura a todos os itens de uma vista de tipo de artigo. Se o utilizador enviar uma consulta utilizando a vista de tipo de artigo e, em seguida, executar uma acção de ligação, o utilizador não conseguirá estabelecer ligação ou anular a ligação aos itens, uma vez que a ACL da vista de tipo de artigo é aplicada pelo sistema do DB2 Content Manager.

Para estabelecer ligação aos itens a que um utilizador poderá não ter acesso, defina um bit de sinalizador de sistema na tabela de controlo do sistema. As secções que se seguem demonstram como estabelecer ligação aos itens aos quais não tem acesso, definindo um sinalizador de sistema:

### Exemplo: Java

```
DKDatastoreICM dsICM = new DKDatastoreICM();
DKDatastoreDefICM dsDef = null;
DKDatastoreAdminICM dsAdmin = null;
DKConfigurationMgmtICM dsCfg = null;
DKLSCfgDefICM lsDef = null;
short systemFlag = 0;
int iSystemFlag = 0;
dsICM.connect("ICMNLSDDB","icmadmin","password","");
dsDef = (DKDatastoreDefICM) dsICM.datastoreDef();
dsAdmin = (DKDatastoreAdminICM) dsDef.datastoreAdmin();
dsCfg = (DKConfigurationMgmtICM) dsAdmin.configurationManagement();
lsCfg = (DKLSCfgDefICM) dsCfg.lscfgMgmt();
lsDef = lsCfg.retrieveConfigParameters();
systemFlag = lsDef.getSystemFlag();
iSystemFlag = systemFlag | DKLSCfgDefICM.USE_VIEWID_LINK_AND_DELETE_DISABLED;
systemFlag = (short)iSystemFlag;
lsDef.setSystemFlag(systemFlag); // disable view acl for link and delete
lsCfg.defineConfigParameters(lsDef); // update library server with information
dsICM.disconnect();
dsICM.destroy();
DKLSCfgDefICM.java
class DKLSCfgDefICM
/**
 * Returns Use view for link and delete enabled state
 * @return true if the use view for link and delete is enabled,
 * false otherwise
 */
public boolean isUseViewForLinkAndDeleteEnabled();
// System Flag
public static final int USE_VIEWID_LINK_AND_DELETE_DISABLED = 16384;
```

### Exemplo: C++

```
DKDatastoreDefICM* dsDef = NULL;
DKDatastoreAdminICM* dsAdmin = NULL;
DKConfigurationMgmtICM* dsCfg = NULL;
DKLSConfigurationMgmtICM* lsCfg = NULL;
DKLSCfgDefICM* lsDef = NULL;
short systemFlag = 0;
long iSystemFlag = 0;
DKDatastoreICM* dsICM = new DKDatastoreICM();
dsICM->connect("ICMNLSDb","icmadmin","password","");
dsDef = (DKDatastoreDefICM*) dsICM->datastoreDef();
dsAdmin = (DKDatastoreAdminICM*) dsDef->datastoreAdmin();
dsCfg = (DKConfigurationMgmtICM*) dsAdmin->configurationManagement();
lsCfg = (DKLSConfigurationMgmtICM*) dsCfg->lsConfigurationMgmt();
lsDef = lsCfg->retrieveConfigParameters();
systemFlag = lsDef->getSystemFlag();
iSystemFlag = systemFlag | USE_VIEWID_LINK_AND_DELETE_DISABLED;
systemFlag = (short)iSystemFlag;
lsDef->setSystemFlag(systemFlag); // disable view acl for link and delete
lsCfg->defineConfigParameters(lsDef); // update library server with information
dsICM->disconnect();
delete dsICM;
DKLSCfgDefICM.hpp
// System Flag
#define USE_VIEWID_LINK_AND_DELETE_DISABLED (short)16384
class DKLSCfgDefICM
/**
 * Returns Use view for link and delete enabled state
 * @return true if the use view for link and delete is enabled,
 * false otherwise
 */
DKEXPORT DKBoolean isUseViewForLinkAndDeleteEnabled();
```

---

## Implementar início de sessão único nas aplicações

É possível utilizar o início de sessão único na Web ou em aplicações do ambiente de trabalho no ambiente do WebSphere Application Server.

Em seguida, encontra o caminho de fluxo de autenticação, através das APIs, que deverá percorrer para implementar esta sessão na aplicação:

1. O utilizador insere o respectivo ID de utilizador e palavra-passe numa aplicação de navegador e o pedido é enviado para o WebSphere Application Server.
2. O WebSphere Application Server autentica o utilizador mediante a verificação das informações sobre o utilizador junto de um servidor LDAP.
3. Se estas informações sobre o utilizador forem válidas, o WebSphere Application Server gera um cookie denominado LTPACookie. Esse cookie contém o token LTPA (Lightweight Third Party Authentication).
4. A aplicação inicia sessão no DB2 Information Integrator for Content convocando `connectWithCredential()` no `DKDatastoreICM` e usando o cookie LTPA. Exemplo:

```
DKAuthenticationData credential = new DKAuthenticationData();
credential.setCredential(sLTPAToken);
//sLTPAToken can be a string representing the LTPA token generated

DKDatastoreICM dsICM = new DKDatastoreICM();

if (credential !=null){
```



```

dsICM.connectWithCredential("ICMNLSDb", credential, "");
} else{
dsICM.connect("ICMNLSDb", UserId, PWD, "") // normal connection with ID and PWD
}

```

5. A API do DB2 Information Integrator for Content efectua uma chamada ao WebSphere Application Server para validar o token LTPA.
6. O WebSphere Application Server valida o token LTPA. Se for válido, a credencial é criada, caso contrário é lançada uma excepção.
7. Se for criada uma credencial, esta é translineada no DKAuthenticationData, e o ID de utilizador do WebSphere Application Server é extraído para utilização ao iniciar sessão no DB2 Content Manager.

A figura seguinte ilustra o caminho do fluxo de autenticação através das APIs acima descritas:

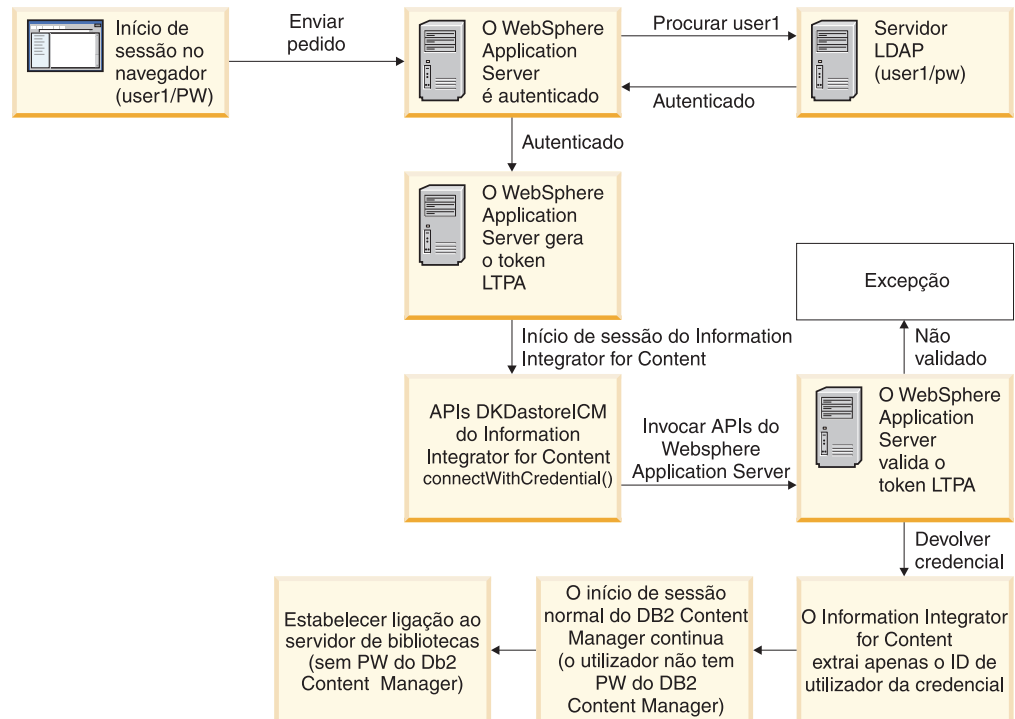


Figura 10. Fluxo de autenticação de início de sessão único através das APIs num ambiente do WebSphere Application Server

Pode também utilizar o bean CMBConnection para estabelecer uma ligação com início de sessão único. Segue-se um exemplo de utilização do bean CMBConnection:

```

public static void connect(CMBConnection connection, String dstype,
String server, String ltpaToken) throws Exception {

// If the server name is followed by a parenthesized string,
// use that string for the connect string.
if (server.indexOf("(") > 0) {
String connectString = server.substring(server.indexOf("(") + 1);
server = server.substring(0, server.indexOf("("));
if (connectString.endsWith(")")) {
connectString = connectString.substring(0, connectString.length() - 1);
}
connection.setConnectString(connectString);
}
}

```

```
// Set properties on connection bean
connection.setDsType(dstype);

// Use the connect method to connect. This will
// create the correct type of DKDatastore object (see Java API)
// and call its connect method.
System.out.println("Connecting to server");
connection.connectWithCredential(server, ltpaToken, "");
System.out.println("OK, Connected");
// Enable display names support. This will cause CMBSchemaManagement,
// CMBEntity, CMBAttribute, CMBItem to use display names when working
//with DB2 Content Manager 8 (default is to use non-display names).
connection.setDisplayNamesEnabled(true);

}
```

Para utilizar início de sessão único da forma descrita na secção anterior, é necessário configurar o sistema para permitir início de sessão fidedigno, executando os seguintes passos:

1. Defina o sinalizador **Allow trusted logon (Permitir início de sessão fidedigno)** na configuração do servidor de bibliotecas. Com o cliente de administração do sistema, siga para **Library Server Parameters (Parâmetros do Servidor de Bibliotecas) → Configurations (Configurações) → Library Server Configuration (Configuração do Servidor de Bibliotecas)**. Certifique-se de que seleccionou **Allow trusted logon (Permitir início de sessão fidedigno)**.
2. Certifique-se de que o ID de ligação da base de dados possui o **privilegio UserDB2TrustedConnect** definido. Para tal, com o cliente de administração do sistema, siga para **Tools (Ferramentas) → Manage Database Connection ID (Gerir ID de Ligação à Base de Dados) → Change Shared Database Connection ID (Alterar ID de Ligação à Base de Dados Partilhada)**. Certifique-se de que a opção **Password is required for all users (A palavra-passe é necessária para todos os utilizadores)** não está seleccionada.
3. Caso ainda não o tenha efectuado, importe os utilizadores do LDAP para o DB2 Content Manager, uma vez que o WebSphere Application Server autentica o utilizador com um servidor LDAP.
4. Certifique-se de que o conjunto de privilégios para todos os utilizadores do DB2 Content Manager contém o privilégio **AllowTrustedLogon**. Para tal, siga para o cliente de administração do sistema e abra o painel de propriedades do utilizador. Certifique-se de que o conjunto de privilégios para todos o utilizador contém o privilégio **AllowTrustedLogon**. Para obter mais informações relativas a privilégios, consulte as informações sobre administração no Information Center.

---

## Saídas de utilizador do servidor de bibliotecas do DB2 Content Manager

Os utilizadores podem personalizar o comportamento do servidor de bibliotecas utilizando as saídas de utilizador do servidor de bibliotecas.

### Importante:

- Para sistemas não zLinux, todas as saídas do utilizador C têm de ser concebidas utilizando 32 bits.
- Para Linux no sistema z, todas as saídas do utilizador têm de ser concebidas utilizando 64 bits, excepto **ICMFetchContentExit**, que ainda tem de ser concebido com 32 bits.

- ICMACLPPrivExit é um UDF e pode ser escrito com 32 ou 64 bits. O UDF de 64 bits é suportado ao funcionar numa ocorrência de db2 de 64 bits.

O directório de exemplos de saídas do utilizador inclui o script bldUX.bat para Windows e o script bldUX para Unix, os quais constituem os exemplos de compilação e os parâmetros de ligação que podem ser utilizados para criar saídas de utilizadores de 32 ou 64 bits.

## Saída de utilizador ICMFetchContentExit para alteração de texto

Esta saída de utilizador C faculta a opção de alteração de texto obtido a partir do gestor de recursos antes de ser transferido para o serviço para indexação de texto completo. A conversão poderá incluir descodificação, adição de identificadores para criar um documento estruturado ou a selecção de apenas um subconjunto do objecto para indexação. Esta saída de utilizador é suportada apenas na plataforma DB2. Desde o Fix pack 2, a pesquisa de texto do DB2 Content Manager foi melhorada através da utilização de ICMFetchContentExit com uma interface diferente.

### ICMFetchContentExit (antiga)

O objecto obtido para indexação de texto completo é fornecido como entrada de dados. Esta saída pode transformar o objecto de qualquer forma necessária, obtendo o novo conteúdo no campo de entrada, caso o comprimento não tenha sido aumentado.

### Finalidade

Se o comprimento de saída for desconhecido no início do processamento ou tiver sido aumentado, o código de saída deverá atribuir o espaço necessário utilizando o endereço da função malloc() transferida. Isto é necessário para evitar problemas quando o código do DB2 Content Manager e a saída são construídos com compiladores diferentes.

**Importante:** Se o servidor de bibliotecas tentar carregar a saída de utilizador e esta não for localizada, as saídas continuam o funcionamento normal, não indicando qualquer erro ao encerrar.

Se a saída existir, é automaticamente carregada. No entanto, por motivos de desempenho, a saída deve ser apenas carregada uma vez, pelo que deve executar um comando db2stop/start antes de tentar carregar a saída.

Pode personalizar a saída de utilizador das seguintes formas:

- A função de saída de utilizador atribui memória ao objecto descodificado antes de o enviar novamente para a função de pesquisa de texto definida pelo utilizador. Neste caso, o servidor de bibliotecas recebe o objecto na memória tampão char \*\* pszDeObj.
- A função de saída de utilizador utiliza descodificação no local, ou seja, no local em que o objecto descodificado está incluído na mesma memória tampão que o objecto de entrada. Neste caso, o servidor de bibliotecas recebe o objecto na memória tampão pszEnObj original.

## Parâmetros

A saída de utilizador é apresentada como:

```
extern C ICMFetchContentExit(long *psTIEFLAG, void *pvInputObject,
long *plInputObjectLength,void *ppvOutputObject,
long *plOutputObjectLength, void *malloc, long *plRC)
```

### Parâmetros de entrada

**long \*psTIEFLAG**

Ponteiro para o valor TIEFLAG transferido do cliente

**char \*pvInputObject**

Ponteiro para o objecto codificado enviado à saída

**long \* plInputObjectLength**

Comprimento do objecto codificado

**char \*\*ppvOutputObject**

Ponteiro para o objecto decodificado definido pela saída

**long \* plOutputObjectLength**

Comprimento do objecto decodificado

**void \* mallocf**

Função de atribuição de memória transferida para a saída de utilizador

### Parâmetros de saída

**long \*plRC**

Ponteiro para o código de retorno

## Códigos exemplo

A rotina de saída de utilizador e os ficheiros .mak associados encontram-se no directório samples/server/exit do CD-ROM do produto. Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT\samples\server\exit*:

*Tabela 16. Localizações dos ficheiros*

| Nome do exemplo | Finalidade                                         |
|-----------------|----------------------------------------------------|
| icmdecxt.c      | Código origem C para saída de utilizador           |
| icmdecxt.def    | Ficheiro DEF padrão para a plataforma Windows      |
| icmldecxt.mak   | Ficheiro de procedimento para a plataforma Linux   |
| icmndecxt.mak   | Ficheiro de procedimento para a plataforma Windows |
| icmsdecxt.mak   | Ficheiro de procedimento para a plataforma Sun     |
| icmdecxt.mak    | Ficheiro de procedimento para a plataforma AIX     |

A biblioteca tem de ser copiada para *ICMDLL\databasename\DLL*, em que *ICMDLL* corresponde a uma coluna PATHICMDLL na tabela ICMSTSYSCONTROL e *databasename* corresponde a nome da base de dados do servidor de bibliotecas.

Origem:icmdecxt.c

## Códigos de retorno

- 0 O objecto foi convertido no local ou não foi convertido. O campo de entrada deverá ser devolvido para indexação de texto completo.
- 1 O objecto foi convertido. Devolva os dados no campo de saída para indexação de texto completo e, em seguida, liberte a memória.
- 2 Não é necessária conversão

Quaisquer outros códigos de retorno resultam num erro de conversão. Se plOutputObjectLength não corresponder a zero, ppvOutputObject contém texto de mensagem para gravar no registo do servidor do DB2 Content Manager.

## ICMFetchContentExit (nova)

A partir do Fix Pack 2, o DB2 Content Manager melhora a função de pesquisa de texto utilizando a função ICMFetchContentExit.

DB2 Content Manager:

- Permite a indexação de metadados fornecendo um mecanismo no ICMFetchFilter
- Suporta uma rotina de utilizador diferente para cada índice de texto

A função ICMFetchContentExit já está disponível no DLL icmdecxt (em UNIX®) ou icmdecxt.dll (em Windows®). A nova função é a mesma função com uma interface diferente. Os novos DLLs são icmxlsfcn em sistemas UNIX e icmxlsfcn.dll em Windows. O último carácter, n, deriva do valor do parâmetro TIEFLAG.

Ao contrário da rotina de saída de utilizador existente, que é invocada apenas após a obtenção do objecto do gestor de recursos, a nova rotina de saída de utilizador é invocada antes da obtenção do objecto. Dependendo do que é devolvido, a nova rotina de saída de utilizador é invocada novamente após a obtenção.

A utilização da função ICMFetchContentExit existente (do DLL icmdecxt) não foi danificada pela nova implementação. A precedência de carregamento é: Se não for localizada qualquer rotina de saída de utilizador, nenhuma será carregada. Se a nova rotina de saída de utilizador for localizada, com base no valor do parâmetro TIEFLAG, é carregada a nova rotina de saída de utilizador (do DLL icmxlsfcn). Caso contrário, é carregada a rotina de saída de utilizador existente (do DLL icmdecxt). A nova rotina de saída de utilizador utiliza o mesmo ficheiro de registo que o ICMFetchFilter.

A saída de utilizador ICMFetchContentExit (ICMFetchContentParms1 \*IFP) melhora a pesquisa de texto.

```
/* ----- */
/* ICMStFetchContentParms1 data structure */
/* ----- */

typedef enum
{
 /* Return codes for user exit when called BEFORE retrieving */
 /* the object from the resource manager */
 ICMRcContinueRetrieving = 0, /* No text returned - continue with */
 ICMRcTextReturned = 1, /* retrieve from the resource manager */
 ICMRcTextReturnedSkipInso = 4, /* Text returned - call INSO to convert */
 /* Object will not be retrieved from RM */
 /* Text returned - do not call INSO */
 /* Object will not be retrieved from RM */
}
```

```

/* Return codes for user exit when called AFTER retrieving */
/* the object from the resource manager */
/*
ICMRcInplaceConversion = 0, /* Object was not converted or was */
/* converted in place. Continue processing */
/* by calling INSO if the MIMETYPE is for */
/* a text object. */
ICMRcOutputGenerated = 1, /* Object was converted and returned */
/* in memory allocated by the function */
/* pfMalloc. INSO conversion should not */
/* be called if MIMETYPE indicates text. */

ICMRcConversionError=2, /* Error in conversion; No text returned */
/* to NSE; output object written to UDF log*/

ICMRcInplaceConversionSkipInso=3, /* Same as 0 but skip INSO conversion */

ICMRcOutputGeneratedSkipInso=4, /* Same as 1 but skip INSO conversion */

ICMRcWarning = -1 /* The object cannot be converted, and */
/* a "warning" should be returned to NSE */
/* to skip this document */

} ICMEnRc;

typedef struct ICMStFetchContentParmsV1
{
/*
/*****
/* The following parameters are passed into the exit and should not
/* be changed.
*****/
size_t uSizeOfStruct; /* Size of the structure */
/* Used for consistency checking */
short sExitVersion; /* Version number of the exit. The */
/* structure shown here is for V1. */
short sExitType; /* 1: Pre-retrieve (called before */
/* object is retrieved from RM) */
/* 2: Post-retrieve (called after */
/* object was retrieved from RM) */
long lComponentTypeID; /* Not currently supported */
char szContentType[ICM_SIZE_CONTENT_TYPE]; /* MIME type for post-retrieve */
short sTIEFlag; /* TIEFLAG value can be used to */
/* control processing in the exit */
char szItemId[ICM_SIZE_ITEM_ID]; /* ITEMID being processed */
short sItemVersionId; /* VersionID being processed */
void *pvInputObject; /* Pointer to the object to be */
/* processed. The object may be */
/* changed by the exit if RC=0 or 2 */
size_t uInputObjectSize; /* Size of the input object in bytes */
void *pfMalloc; /* This pointer to malloc must be used*/
/* to allocate memory returned in */
/* pvOutputObject */
FILE * pstDebugFile; /* If not null, pointer to a file */
/* that may be used for debugging via */
/* fprintf, fflush. DO NOT CLOSE. */

/*****
/* The following parameters are set by the exit
*****/
void * pvOutputObject; /* Pointer to the output object when */
/* RC=1 or 2 */
size_t uOutputObjectSize; /* Length of the object in bytes */

```

```

char szErrorMessageText[ICM_SIZE_MESSAGE_TEXT]; /* Error message */
ICMEnRc ReturnCode; /* Return code */
} ICMStFetchContentParmsV1;

```

## Saída de utilizador ICMLogonExit para início de sessão

Esta saída de utilizador C é invocada no início de sessão e faculta uma interface para validar um ID de utilizador, palavra-passe, contagem de utilizadores e autenticação de LDAP.

### Finalidade

O nome externo desta saída de utilizador é ICMXLSLG.DLL. Coloque o DLL no directório: ICMROOT/database name/DLL para utilizar esta saída de utilizador. Esta saída de utilizador é suportada em DB2 e z/OS.

### Parâmetros

```

extern C ICMLogonExit(pszLanguage, pszUserID, pszPassword,
pszNewPassword, pszApplication, psUserDomain, pszLDAPInfo,
pszDatabaseName, pszSchemaName, plRC, plReason, plExtRC, plExtReason)

```

#### Parâmetros de entrada

Os parâmetros que se seguem são os parâmetros de entrada:

```

char *pszLanguage
 Código de linguagem

char *pszUserID
 ID de utilizador

char *pszPassword
 Palavra-passe decodificada

char *pszNewPassword
 Nova palavra-passe decodificada (se tiver sido fornecida uma nova
 palavra-passe)

char *pszApplication
 Nome da aplicação

short *psUserDomain
 Domínio do utilizador

char *pszLDAPInfo
 O caminho no servidor de LDAP para este ID de utilizador

char *pszDatabaseName
 O nome da base de dados do servidor de bibliotecas

char *pszSchemaName
 O nome do esquema de base de dados do servidor de bibliotecas

```

#### Parâmetros de saída

Os parâmetros que se seguem são os parâmetros de saída:

```

long *plRC
 Código de retorno

long *plReason
 Código da razão

```



**long \*plExtRC**

Utilize este parâmetro para devolver quaisquer informações adicionais de erro específicas da aplicação

**long \*plExtReason**

Utilize este parâmetro para devolver quaisquer informações adicionais de erro específicas da aplicação

Defina o código da razão (plReason ) de acordo com as seguintes regras:

**\*plReason = 0**

ICMLogon efectuará a validação da palavra-passe

**\*plReason = 1**

ICMLogon ignorará a validação da palavra-passe.

## Códigos exemplo

É fornecido um código exemplo no directório raiz do ICM, em:

/samples/icmxslg.c. Para EIP, o código exemplo encontra-se no directório raiz do EIP, em: /samples/icmxslg.c.

Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT*\samples\server\exit:

*Tabela 17. Localizações dos ficheiros*

| Nome do exemplo | Finalidade                                            |
|-----------------|-------------------------------------------------------|
| icmxslg.c       | Código origem C para saída de utilizador              |
| icmxslg.def     | Ficheiro DEF padrão para a plataforma Windows         |
| icmnlsg.mak     | Ficheiro de procedimento para a plataforma Windows(R) |
| icmxslg.mak     | Ficheiro de procedimento para a plataforma Sun        |
| icmxslg.mak     | Ficheiro de procedimento para a plataforma AIX        |

A biblioteca tem de ser copiada para *ICMDLL\databasename\DLL*, em que *ICMDLL* é uma coluna *PATHICMDLL* na tabela *ICMSTSYSCONTROL* e *databasename* é o nome da base de dados do servidor de bibliotecas.

## Códigos de retorno

A lista seguinte contém os códigos de retorno (\*plRC) esperados ao iniciar sessão no DB2 Content Manager:

### **0 - RC\_OK**

Validação transferida. O ICMLogon continua a execução normal.

### **7123 - RC\_INVALID\_PARAMETER**

O nome de um ponteiro ou valor não válido tem sessão iniciada. O início de sessão é negado.

### **7015 - RC\_UNEXPECTED\_SQL\_ERROR**

Erro de SQL. O início de sessão é negado.

### **7172 - RC\_INVALID\_PASSWORD**

A palavra-passe não corresponde á palavra-passe definida por este utilizador. O início de sessão é negado.

**7173 - RC\_MAX\_LOGON\_PASSWORD\_RETRY**

Foi atingido o número máximo de tentativas com uma palavra-passe incorrecta. O início de sessão é negado.

**7098 - RC\_INVALID\_NEWPASSWORD**

Definido apenas pela saída, caso a nova palavra-passe não corresponda às regras do cliente. O início de sessão é negado.

**7171 - RC\_PASSWORD\_EXPIRED**

A palavra-passe tem de ser alterada. Invoque novamente ICMLogon com uma nova palavra-passe. O início de sessão é negado.

**7160 - RC\_LOGON\_MAX\_USER\_ERROR**

O número máximo de utilizadores simultâneos foi atingido. O início de sessão é negado.

**7094 - RC\_ALREADY\_LOGGED\_ON**

O ID de utilizador já tem sessão iniciada no sistema do Content Manager. O início de sessão é negado.

**7006 - RC\_DLL\_LOAD\_ERROR**

Não foi possível carregar um DLL. O início de sessão é negado.

**7011 - RC\_GET\_PROC\_ADDRESS\_ERROR**

Não foi possível obter um procedimento. O início de sessão é negado.

**4751 - RC\_LOGON\_MAX\_USER\_WARNING**

O número máximo de utilizadores simultâneos foi atingido. O início de sessão é permitido, mas deverá ser apresentada uma mensagem de aviso.

**Informações relacionadas**

Melhoramentos à pesquisa de texto

## Saída de utilizador myExit para encaminhamento de documentos

Esta saída de utilizador é invocada após a criação de um pacote de trabalho (utilizando o processo de início), quando um pacote de trabalho é retirado de um nó de trabalho, para o mover para outro nó de trabalho, ou quando o limite de sobrecarga é atingido.

**Finalidade**

No momento em que a saída é invocada, o pacote de trabalho existem na tabela de pacotes de trabalho e pode ser obtido utilizando o ID de componente como identificador da linha na tabela ICMUT00204001.

**Parâmetros**

```
typedef struct ICMCONTAINERDATA_STRUCT
{
 char szContainerName[33]; Container Data variable name
 char szContainerVal[255]; Container Data variable value
} ICMCONTAINERDATA_STRUCT;
typedef struct _ICMUSERSTRUCT
{
 long lUserEvent;
 char szWPCompID[19];
 char szWPItemID[27];
 short sWPVersionID;
 char szRouteSel[33];
 short sUpdateFlag;
 short sNumContainerData;
 pContainerDataStructIn
```

```

pContainerDataStructOut
const ICMCONTAINERDATA_STRUCT *pContainerDataStructIn;
ICMCONTAINERDATA_STRUCT *pContainerDataStructOut;
void * mallocf;
} ICMUSERSTRUCT;

```

## Parâmetros de entrada

### IUserEvent

Indica o evento de nó de trabalho (Work Node Event) no qual esta saída de utilizador foi invocada. Valores:

- 1 O pacote de trabalho está a entrar no nó de trabalho.
- 2 O pacote de trabalho está a sair do nó de trabalho.
- 3 O pacote de trabalho está no nó de trabalho, o que excede o limite de sobrecarga deste nó de trabalho.

### szWPCompID

O ID de componente do pacote de trabalho foi transferido para esta saída de utilizador

### szWPItemID

O ID de artigo do pacote de trabalho foi transferido para esta saída de utilizador.

### sWPVersionID

O ID de versão do pacote de trabalho foi transferido para esta saída de utilizador

### sNumContainerData

Número de estruturadas de dados do contentor do ICM enviadas para esta saída no pContainerDataStructIn

### pContainerDataStructIn

Contém dados de contentor enviados para a saída de utilizador

## Parâmetros de saída

### szRouteSel

O caminho efectuado pelo pacote de trabalho ao regressar desta saída de utilizador

### sUpdateFlag

- 1 A saída de utilizador está a devolver dados de contentor actualizados
- 0 A saída de utilizador não está a devolver dados de contentor actualizados

### sNumContainerData

Número de estruturas de dados do contentor do ICM devolvidas no pContainerDataStructOut

### pContainerDataStructOut

Contém dados de contentor devolvidos pela saída de utilizador

### mallocf

O ponteiro da função malloc de C, de que a saída de utilizador necessita para atribuir memória à estrutura de dados de contentor de saída, será devolvido no pContainerDataStructOut

## Códigos exemplo

É fornecido um código exemplo no directório *IBMCMROOT\samples\server\exit*. Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT\samples\server\exit*:

Tabela 18. Localizações dos ficheiros

| Nome do exemplo | Finalidade                                                                                                                              |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| icmdruext.c     | Código origem C para saída de utilizador                                                                                                |
| icmdruext.def   | Ficheiro DEF padrão para a plataforma Windows                                                                                           |
| wxv2tue.h       | Ficheiro de cabeçalho para constantes e protótipos de saída de utilizador                                                               |
| bldexit.bat     | Efectue a compilação da saída de utilizador do servidor de bibliotecas do DB2 Content Manager utilizando o Microsoft Visual Studio 2003 |

A biblioteca tem de ser copiada para *ICMDLL\databasename\DLL*, em que *ICMDLL* é uma coluna *PATHICMDLL* na tabela *ICMSTSYSCONTROL* e *databasename* é o nome da base de dados do servidor de bibliotecas.

## Saída do utilizador ICMValidatePassword para validar palavras-passe

A saída de utilizador para validação de palavras-passe é invocada para validar se a nova palavra-passe cumpre as directrizes necessárias.

### Finalidade

Pode personalizar a saída de utilizador de modo a que cumpra as directrizes de palavras-passe. Esta saída de utilizador é suportada em DB2 e z/OS.

O servidor de bibliotecas permite aos utilizadores fazer cumprir as regras de validação da sintaxe de palavras-passe através da utilização da biblioteca *ICMPLSVP* da saída de utilizador. A saída de utilizador da biblioteca *ICMPLSVP* é invocada quando:

- Os utilizadores estão a criar o respectivo perfil ou a actualizar a palavra-passe de utilizador
- Os utilizadores estão a alterar a respectiva DB2 Content Managerpalavra-passe de início de sessão

Todas as declarações e definições encontram-se no ficheiro de cabeçalho *icmuxcom.h*.

Para esta saída de utilizador, o servidor de bibliotecas é iniciado assim que o perfil de utilizador é criado ou a palavra-passe do utilizador é alterada. Se o servidor de bibliotecas localizar a biblioteca *ICMPLSVP* na localização especificada, carrega a saída de utilizador e executa a função *ICMValidatePassword*. Se o servidor de bibliotecas não localizar a biblioteca *ICMPLSVP* na localização especificada, o servidor de bibliotecas não verifica novamente a saída de utilizador até que a base de dados seja parada ou iniciada.

Deste modo, para implementar a saída de utilizador, é necessário parar ou iniciar o servidor de base de dados para que o servidor de bibliotecas utilize a nova saída de utilizador.

Após a invocação da saída de utilizador, esta devolve UX\_HOLD ou UX\_RELEASE. Se a saída de utilizador devolver UX\_HOLD, o servidor de bibliotecas não descarrega a saída de utilizador, mantendo-a na memória para melhorar o desempenho. O valor de retorno UX\_RELEASE faz com que o servidor de bibliotecas liberte a saída de utilizador e tente carregá-la novamente na invocação seguinte. Neste caso, a saída de utilizador é sempre recarregada.

**Importante:** Se a saída de utilizador da palavra-passe for colocada em cmgmt/ls/icmnlstdb/DLL, a saída de utilizador é utilizada pelo servidor da biblioteca para modificar a palavra-passe para os utilizadores do DB2 Content Manager. Contudo, se estiver a tentar modificar a palavra-passe do utilizador administrativo do DB2 Content Manager ( icmadmin) a partir da janela de início de sessão do DB2 Content Manager ou da janela de início de sessão do pClient e do eClient, a saída de utilizador não é aplicada porque a API invoca directamente a base de dados de DB2 em vez do servidor da biblioteca para alterar a palavra-passe.

## Parâmetros

A função invocada na saída de utilizador ICMValidatePassword é declarada extern C, tal como em seguida indicado:

```
int ICMValidatePassword (ICM_UX_PASSWORD_VALIDATION_STRUCT *pInputParams,
 ICM_UX_DIAGNOSTIC_RETURN_STRUCT *pReturnParams);
```

### Parâmetros de saída

Parâmetros de saída comuns da saída de utilizador, códigos de retorno e cadeia de diagnóstico.

```
typedef struct ICM_UX_DIAGNOSTIC_RETURN_STRUCT {
 int returnCode; //output: RC_OK, RC_ERROR
 int returnReasonCode; //output: user-defined return codes
 char diagString[2048+1]; //output. additional text info
} ICM_UX_DIAGNOSTIC_RETURN_STRUCT;
```

#### returnCode

Código de retorno. Os códigos de retorno possíveis da saída de utilizador ICMValidatePassword são:

##### RC\_OK(value 0)

Validação da palavra-passe com êxito.

##### RC\_ERROR(value -1):

Falha na validação da palavra-passe.

#### returnReasonCode

Código da razão definido pelo utilizador. O returnReasonCode indica o motivo pelo qual ocorreu uma falha na validação da palavra-passe. Este código de razão é devolvido ao cliente ou à aplicação.

#### diagString

É registado um erro ou mensagem de aviso definidos pelo utilizador no registo do servidor de bibliotecas caso o valor não seja nulo.

### Parâmetros de entrada

Parâmetros de entrada para a saída de utilizador de validação de palavras-passe.

```
typedef struct _ICM_UX_PASSWORD_VALIDATION_STRUCT {
 char currentLSVersion[128+1];
 char userId[32+1];
 char newPassword[48+1];
} ICM_UX_PASSWORD_VALIDATION_STRUCT;
```

#### **currentLSVersion**

Nível do servidor de bibliotecas. Este parâmetro pode ser utilizado para verificar a compatibilidade da saída de utilizador e da versão do servidor de bibliotecas.

#### **userId**

ID de utilizador do DB2 Content Manager.

#### **newPassword**

Nova palavra-passe.

## **Valores de retorno**

A função de saída de utilizador devolve:

#### **UX\_HOLD (valor 0)**

A biblioteca da saída de utilizador não é libertada após o retorno.

#### **UX\_RELEASE (valor 1)**

A biblioteca da saída de utilizador é libertada após o retorno.

## **Códigos exemplo**

É fornecido um código exemplo no directório *IBMCMROOT\samples\server\exit\icmplsvp\**. Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT\samples\server\exit*:

*Tabela 19. Localizações dos ficheiros*

| <b>Nome do exemplo</b> | <b>Finalidade</b>                                                                                                                       |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| icmplsvp.c             | Código origem C para saída de utilizador                                                                                                |
| icmuxcom.h             | Ficheiro de cabeçalho comum para constantes e protótipos de saída de utilizador                                                         |
| icmplsvp.def           | Ficheiro DEF padrão para a plataforma Windows                                                                                           |
| bldUX.bat              | Efectue a compilação da saída de utilizador do servidor de bibliotecas do DB2 Content Manager utilizando o Microsoft Visual Studio 2003 |
| bldUX                  | Script UNIX para a compilação da saída de utilizador do servidor de bibliotecas do DB2 Content Manager em plataformas AIX, Sun e Linux  |

A biblioteca tem de ser copiada para *ICMDLL\databasename\DLL*, em que *ICMDLL* é uma coluna *PATHICMDLL* na tabela *ICMSTSYSCONTROL* e *databasename* é o nome da base de dados do servidor de bibliotecas.

## **Códigos de retorno**

A lista que se segue contém os códigos de retorno do servidor de bibliotecas ao iniciar sessão no DB2 Content Manager:

#### **7109 - ICM\_PASSWORD\_VALIDATION\_FAILED**

A nova palavra-passe não é válida ao tentar criar um utilizador ou alterar a palavra-passe.

É apresentada a seguinte mensagem de excepção:

```
DGL7280A: The new password failed to pass the validation rules.: : U001;
ICM0001: This is an unexpected internal error or an unrecognized error code.
Run the application again with server trace set to Detail and Data.
Save the server log and see your IBM service representative.
: 7109 (STATE) : [LS RC = 7109, LS reasonCode = 2]
```

#### 7011 - RC\_GET\_PROC\_ADDRESS\_ERROR

A saída de utilizador não contém o nome da função requerida. Consulte o registo do servidor de bibliotecas para obter mais informações.

## Rotinas de saída de utilizador de ACL

O DB2 Content Manager faculta duas rotinas de saída de utilizador de controlo de acessos que substituem o mecanismo de controlo de acessos incorporado: ICMACLPPrivExit e ICMGenPrivExit. O utilizador tem de construir ambas as rotinas para concluir a substituição.

### ICMACLPPrivExit

#### Finalidade

Utilize esta rotina de saída de utilizador para determinar se o utilizador tem autoridade para executar a função solicitada. Esta não é uma rotina de saída de utilizador C, mas sim uma UDF de base de dados C. Esta UDF é utilizada quando o DB2 Content Manager verifica os privilégios incluídos numa instrução SQL. Esta saída de utilizador é suportada em plataformas DB2 e z/OS.

#### Parâmetros

O exemplo que se segue mostra a declaração da função C para a UDF de rotina de saída de privilégios da ACL.

```
void SQL_API_FN ICMACLPPrivExit(
 char *pszUserID,
 char *pszApplicationID,
 char* APIAction,
 char *pszProcedureName,
 char *InfoType,
 char *ItemID,
 char *ViewID,
 long *plGenPrivCode,
 short * pszUserID _ind,
 short * pszApplicationID _ind,
 short * APIAction _ind,
 short * pszProcedureName _ind,
 short * InfoType _ind,
 short * ItemID _ind,
 short * ViewID _ind,
 short * plGenPrivCode _ind,
 char *sqlstate, /* SQL STATE */
 char *funcname, /* function name */
 char *specname, /* specific function name */
 char *mesgtext /* message text insert */
)
```

#### Parâmetros de entrada

##### pszUserID

ID de utilizador (o número máximo de caracteres é 32)



**pszApplicationID**

ID da aplicação (o número máximo de caracteres é 32667 - reservados)

**pszHostname**

Nome do sistema central do cliente (o número máximo de caracteres é 128)

**APIAction**

As acções da API são:

- 0 Obter
- 2 Actualizar
- 3 Eliminar
- 4 Não disponível

**pszProcedureName**

Nome do procedimento armazenado do servidor de bibliotecas (20 caracteres)

**InfoType**

Defina o seguinte parâmetro para o tipo de informações relacionadas com:

- 0 Artigos
- 2 Vistas
- 3 Tipos de artigo ou vistas de tipo de artigo

**ItemID**

ID de artigo (26 caracteres)

**ViewID**

ID de tipo de artigo, ID de vista de tipo de artigo, ou ID de vista

**pIGenPrivCode**

Privilégio geral requerido para a acção

**Parâmetros de saída****pIRC**

Ponteiro para o código de retorno

**Códigos exemplo**

É fornecido um código exemplo no directório *IBMCMROOT\samples\server\exit*. Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT\samples\server\exit*:

*Tabela 20. Localizações dos ficheiros*

| Nome do exemplo | Finalidade                                                                                      |
|-----------------|-------------------------------------------------------------------------------------------------|
| icmaclxt.c      | Código origem C para saída de utilizador                                                        |
| icmaclxt.def    | Ficheiro DEF padrão para a plataforma Windows                                                   |
| icmaclxt.exp    | Ficheiro EXP padrão para a plataforma Windows                                                   |
| aclnvar.mak     | Variáveis globais necessárias para o ficheiro de procedimento da saída de utilizador Windows(R) |
| icmlaclxt.mak   | Ficheiro de procedimento para a plataforma Linux(R)                                             |
| icmnaclxt.mak   | Ficheiro de procedimento para a plataforma Windows(R)                                           |

Tabela 20. Localizações dos ficheiros (continuação)

| Nome do exemplo | Finalidade                                        |
|-----------------|---------------------------------------------------|
| icmsaclxt.mak   | Ficheiro de procedimento para a plataforma Sun    |
| icmxaclxt.mak   | Ficheiro de procedimento para a plataforma AIX(R) |

A biblioteca tem de ser copiada para `<sqlib>\function`, em que `<sqlib>` corresponde ao directório inicial da instância de base de dados para DB2.

## Códigos de retorno

**0** Validação RC\_OK que a rotina ICMACLPPrivExit concede à ACL para consulta

### Diferente de zero

Declina o acesso à consulta da ACL

## ICMGenPrivExit

### Finalidade

Utilize esta rotina de saída para determinar se o utilizador tem o privilégio geral para executar a função solicitada num determinado artigo ou vista. Esta saída de utilizador é suportada em DB2 e z/OS.

### Parâmetros

O exemplo seguinte apresenta a rotina de saída de utilizador ICMGenPrivExit. Os parâmetros fornecem informações que pode utilizar na lógica de acesso de segurança.

```
extern long ICMGenPrivExit(
 char *pszUserID,
 char *pszApplicationID,
 long *pIGenPrivCode,
 char *pszProcedureName,
 long *pICMDecision,
 long *pICMDecReason,
 long *pIRC
)
```

### Parâmetros de entrada

#### pszUserID

ID de utilizador (o número máximo de caracteres é 32)

#### pszApplicationID

ID da aplicação (o número máximo de caracteres é 32667 - reservados)

#### pIGenPrivCode

Privilégio geral requerido para a acção

#### pszProcedureName

Nome do procedimento armazenado do servidor de bibliotecas (20 caracteres)

#### pICMDecision

Decisão do servidor de bibliotecas

**0** Conceder acesso

#### Outros códigos

Procurar códigos de erro do servidor de bibliotecas

### **pICMDecReason**

Código de retorno expandido, se disponível

### **Parâmetros de saída**

#### **pIRC**

Ponteiro para o código de retorno

### **Códigos exemplo**

É fornecido um código exemplo no directório *IBMCMROOT\samples\server\exit*. Para a saída de utilizador exemplo, os ficheiros seguintes encontra-se instalados no directório *IBMCMROOT\samples\server\exit*:

*Tabela 21. Localizações dos ficheiros*

| Nome do exemplo | Finalidade                                            |
|-----------------|-------------------------------------------------------|
| icmgenxt.c      | Código origem C para saída de utilizador              |
| icmgenxt.def    | Ficheiro DEF padrão para a plataforma Windows         |
| icmgenxt.exp    | Ficheiro EXP padrão para a plataforma Windows         |
| icmlgenxt.mak   | Ficheiro de procedimento para a plataforma Linux(R)   |
| icmngenxt.mak   | Ficheiro de procedimento para a plataforma Windows(R) |
| icmsgenxt.mak   | Ficheiro de procedimento para a plataforma Sun        |
| icmxgenxt.mak   | Ficheiro de procedimento para a plataforma AIX(R)     |

A biblioteca tem de ser copiada para *ICMDLL\databasename\DLL*, em que *ICMDLL* é uma coluna PATHICMDLL na tabela ICMSTSYSCONTROL e *databasename* é o nome da base de dados do servidor de bibliotecas.

### **Códigos de retorno**

0 Validação RC\_OK que a rotina ICMACLPrivExit concede à ACL para consulta

#### **Diferente de zero**

Declina o acesso à consulta da ACL

---

## **Limitações do servidor de bibliotecas e da base de dados federada**

Quando se liga a um servidor de bibliotecas com um ID de utilizador de DB2 Content Manager ou de base de dados federada que não seja ID de utilizador de base de dados, o conector tenta ligar à base de dados com esse ID de utilizador e a ligação irá falhar. O conector irá em seguida ligar com o ID de utilizador de ligação à base de dados e transmitir o ID de utilizador do DB2 Content Manager ou da base de dados federada ao servidor.

Para evitar que a ligação à base de dados falhe quando se usa um ID de utilizador de DB2 Content Manager ou de base de dados federada que não seja um ID de utilizador de base de dados, pode efectuar as seguintes alterações:

#### **Para uma base de dados federada**

No ficheiro cmbds.ini, atualize o valor FEDSERVERREPTYPE para DB2CON em vez de DB2 para uma base de dados específica. Deve ser algo como: FEDSERVERREPTYPE=DB2CON. Quando o FEDSERVERREPTYPE é definido como DB2CON, o conector inicia sessão utilizando primeiro o ID

de ligação da base de dados e, em seguida, transmite o ID de utilizador da base de dados ao servidor relativo a essa base de dados federada.

Se o ID de utilizador da base de dados federada for também ID de utilizador de base de dados, a chamada ao servidor falha. O conector desliga automaticamente da base de dados e volta a ligar com o ID de utilizador federado para iniciar sessão na base de dados e, de seguida, transmite o ID de utilizador de base de dados federada ao servidor.

Para o método connect do conector DKDatastoreFed de Java e C++, existe outra opção de cadeia de ligação (REPTYPE=DB2CON) que pode ser especificada, de forma a que uma determinada instância do arquivo de dados tenha o comportamento acima mencionado.

#### **Para uma base de dados do DB2 Content Manager Versão 8**

No ficheiro cmbicmsrvs.ini, actualize o valor ICMSEVERREPTYPE para DB2CON em vez de DB2 para uma determinada base de dados do DB2 Content Manager Versão 8 no ficheiro cmbicmsrvs.ini, tal como se segue: ICMSEVERREPTYPE=DB2CON

Após a alteração do valor ICMSEVERREPTYPE, o conector do DB2 Content Manager inicia sessão no servidor utilizando primeiro o ID de ligação da base de dados e, em seguida, transmite o ID de utilizador do DB2 Content Manager Versão 8 ao servidor para essa base de dados do DB2 Content Manager Versão 8.

Se o o ID de utilizador do DB2 Content Manager Versão 8 for também ID de utilizador de base de dados, a chamada ao servidor falha. O conector desliga automaticamente da base de dados e volta a ligar com o ID de utilizador do DB2 Content Manager Versão 8 para iniciar sessão na base de dados e, de seguida, transmite o ID de utilizador do DB2 Content Manager Versão 8 ao servidor.

Para o método connect do conector DKDatastoreICM de Java e C++, pode especificar uma opção de cadeia de ligação (REPTYPE=DB2CON), de forma a que uma determinada instância do arquivo de dados tenha o comportamento acima mencionado.

---

## **Trabalhar com o gestor de recursos**

Um gestor de recursos de DB2 Content Manager controla uma recolha de recursos geridos (objectos). Gere também a memória necessária e infra-estrutura de HSM (Hierarchical Storage Management), mas primeiro é necessário configurar o gestor de recursos para suportar HSM. Os gestores de recursos possuem funções para suportar serviços específicos de tipos para mais do que um tipo de objecto como, por exemplo, sequenciar, compactar, descompactar, codificar, converter códigos, pesquisar ou extrair texto.

Um único gestor de recursos é utilizado exclusivamente por um servidor de bibliotecas. Cada gestor de recursos facultado pelo sistema de DB2 Content Manager faculta um subconjunto comum de APIs de acesso a dados nativos, que pode ser acedido através da servidor de bibliotecas controladora, de outros componentes de DB2 Content Manager e de aplicações, quer local (no mesmo nó de rede) ou remotamente.

Outras APIs de acesso a dados permitem acesso remoto ao gestor de recursos utilizando o suporte de cliente do próprio gestor de recursos e um protocolo padrão de acesso de rede como, por exemplo, CIFS, NFS ou FTP. Para obter o

acesso remoto, utilize uma ligação entre cliente e servidor. Os clientes comunicam com os gestores de recursos do Content Manager que utilizam HTTP através da utilização de um servidor da Web padrão. A entrega de dados tem por base os protocolos de transferência de dados de HTTP, FTP e FS. Utilizando HTTP, todas as aplicações ou componentes do Content Manager que necessitem de aceder aos conteúdos geridos do Content Manager podem formar dinamicamente um triângulo com um servidor de bibliotecas e um gestor de recursos. Este triângulo forma um caminho de acesso a dados directo entre a aplicação e cada gestor de recursos, mas também um caminho de controlo entre o servidor de bibliotecas e o gestor de recursos. Pode correlacionar este triângulo conceptual para qualquer configuração de rede, que vá desde a configuração de apenas um nó até uma distribuída geograficamente.

A nova arquitectura aloja ainda gestores de recursos a que uma aplicação não consegue aceder directamente, tais como um subsistema baseado no sistema central, um sistema de utilizador único que não processe controlo de acesso ou um sistema que contenha informações de elevada sensibilidade no qual o acesso directo através de uma aplicação não é permitido pela política empresarial. Neste caso, o acesso a estes gestores de recursos é indirecto. Tanto o paradigma pull como o paradigma push da transferência de dados são adaptados pelo sistema Content Manager, bem como as chamadas síncronas e assíncronas.

Para obter informações relativas à configuração de um gestor de recursos, consulte *Planning and Installing Your Content Management System* e o exemplo `SResourceMgrDefCreationICM` no directório `samples`.

## Trabalhar com objectos do gestor de recursos

Num DB2 Content Manager, cada entidade gerida chama-se um artigo. Existem dois tipos de artigos, o tipo que representa entidades lógicas puras como, por exemplo, documentos ou pastas, ou entidades que representam objectos de dados físicos como, por exemplo, os dados de texto de um documento de processamento de texto, a imagem digitalizada de uma participação ou o clip de vídeo de um acidente de viação. Os objectos possuem um estado e comportamento necessários para processar os dados físicos associados a um documento lógico.

Os objectos de recurso também representam coisas como ficheiros num sistema de ficheiros, clips de vídeo num servidor de vídeo e BLOBs. Durante o tempo de execução, os objectos de recurso são utilizados para aceder aos dados físicos para os quais remetem. Por essa razão, os objectos de recurso no Content Manager possuem um tipo. Ou seja, têm um determinado estado e comportamento. O servidor de bibliotecas e o gestor de recursos partilham um esquema para armazenar o estado de um objecto. Os tipos de objectos de base facultados pelo Content Manager são: BLOBs ou CLOBs genéricos e objectos de conteúdo de Texto, Imagem e Vídeo. O utilizador pode também criar subclasses dos tipos predeterminados. Um objecto de recurso também pode ter atributos definidos pelo utilizador, que são utilizados para pesquisa e obtenção.

Segundo a perspectiva do sistema Content Manager, cada objecto é representado por um identificador lógico único, o seu Uniform Resource Identifier (URI). O servidor de bibliotecas gere o espaço de nome do URI. Se assim for solicitado, o servidor de bibliotecas correlaciona URIs com Uniform Resource Locators (URL). Os URLs são utilizados para obter acesso aos dados físicos. Os URLs não remetem directamente para uma área de memória gerida pelo gestor de recursos. Pelo contrário, o gestor de recursos utiliza um espaço de nome local para converter nomes de objectos lógicos em nomes de ficheiros físicos. Os URIs de objectos são

criados pelo gestor de recursos específico. O servidor de bibliotecas ou o utilizador final podem sugerir um URI de objecto (o seu nome), mas a decisão é tomada pelo gestor de recursos.

Pode aceder a um objecto utilizando as APIs do gestor de recursos do DB2 Content Manager (armazenar, obter, actualizar, etc.). Em alguns casos, pode utilizar APIs que sejam nativas do objecto (sequência, difusão selectiva e etapa) ou o sistema de ficheiros.

Para obter informações sobre como trabalhar com objectos do gestor de recursos, consulte o exemplo `SResourceItemCreationICM` no directório `samples`, `IBMCMROOT/samples/java/icm` ou `IBMCMROOT/samples/cpp/icm`.

## Obtenção confidencial de objectos de recurso

O DB2 Content Manager suporta Secure Sockets Layer (SSL) através de APIs de Java para a comunicação entre um thin client ou aplicações applet e o gestor de recursos. Utilize SSL se a aplicação aceder a objectos do gestor de recursos e for executado fora de limites seguros, como uma firewall.

### Requisitos

- Antes de utilizar a aplicação para comunicar como gestor de recursos através de SSL, verifique se o servidor HTTP no gestor de recursos está activado para SSL.
- Para applet do visualizador de eClient, certifique-se de que a aplicação de eClient está configurada para utilizar uma ligação HTTPS.
- Ao invocar a API de Java de `DKLobICM.getContentURL` na aplicação, esta devolve um URL. Se o URL devolvido por `DKLobICM.getContentURLs` for um URL HTTPS URL, o SSL está activado. Se o URL devolvido for um URL HTTP, a SSL não está activada.

Para activar o SSL, edite o ficheiro `cmbrm.ini`, que pode encontrar no mesmo directório que os outros ficheiros de propriedades. O ficheiro `cmbcmenv.properties` contém uma entrada denominada `CMCFGDIR`, que aponta para a localização do ficheiro `cmbrm.ini`. No ficheiro `cmbrm.ini`, configure a definição `RM_SSL_FOR_URL_RETRIEVES` como 1. Se a definição não existir, adicione-a.

Após ter verificado que a definição `RM_SSL_FOR_URL_RETRIEVES` está correcta, a aplicação pode utilizar o URL HTTPS para comunicar de forma segura com o gestor de recursos.

Tenha em atenção que `RM_SSL_FOR_URL_RETRIEVES` é uma definição global e que, assim que for definida como 1, todas as obtenções baseadas em URL com o gestor de recursos no sistema vão utilizar SSL. Isto significa que, se o sistema estiver definido para trabalhar com vários gestores de recursos, todos os gestores de recursos têm de suportar SSL.

## Remover conteúdo de objectos de recurso

Quando elimina o conteúdo associado a um artigo de recurso, pode estar a destruir esse conteúdo de forma irreversível.

Esta função, denominada `irrecoverable destroy`, só é suportada no conector do DB2 Content Manager Versão 8 para objectos armazenados num disco fixo num gestor de recursos Windows, AIX, Solaris.

**Importante:** Para utilizar esta função, o conector, o servidor de bibliotecas e o gestor de recursos do DB2 Content Manager Versão 8, terão de ter o mesmo pacote de correcções do DB2 Content Manager Versão 8.2 de nível sete ou posterior. Se um dos componentes do DB2 Content Manager for de níveis anteriores, o irrecoverable destroy é processado como uma eliminação normal.

Para utilizar o irrecoverable destroy, é necessário especificar a opção DK\_CM\_DESTROY\_DELETE de DKConstant em operações que resultem na eliminação de um objecto. Em seguida, encontra-se uma lista do resumo das operações que eliminam objectos e nas quais pode especificar o irrecoverable destroy:

- Ao invocar delete num artigo de recurso ou documento.
- Ao invocar update para eliminar uma parte de documento ou ao actualizar um artigo de recurso de conteúdo para não conteúdo.
- Ao invocar update para um artigo de recurso que tem a criação de versões activada. Neste caso, quando ultrapassa o número máximo de versões armazenadas, e a versão mais antiga deve ser eliminada, a versão mais antiga também é destruída.
- Quando um artigo é reindexado a um novo tipo de artigo, o artigo do tipo de artigo original é destruído.

Consulte Application Programming Reference para obter mais informações sobre os métodos e opções para eliminar objectos. A lista seguinte apresenta um resumo das APIs e métodos associados que pode utilizar para especificar o irrecoverable destroy:

#### **DKDatastoreICM**

```
updateObject(dkDataObject ddo, int option)
deleteObject(dkDataObject ddo, int option)
moveObject(dkDataObject sourceDDO, dkDataObject destinationDDO,
int options)
```

#### **DKLobICM**

```
del(int option)
update(InputStream is,long length,int option)
update(DKThirdPartyServerDef thirdpartyObject, int option)
update(int option)
update(String aFullFileName,int option)
update(int option,DKRMSMPairDefICM[] rmsmpsairs)
updateFrom(String hostname, String userid, String passwd, String
protocol, int port, String filename,int option)
updateFrom(int option)
updateFromAsync(String hostname, String userid, String passwd,
String protocol, int port, String filename,int option)
updateFromAsync(int option)
```

#### **DKDDO**

```
update(int option)
update(DKNVPair[] option)
del(int option)
del(DKNVPair[] option)
```



Na maioria dos casos, pode combinar DK\_CM\_DESTROY\_DELETE com outras opções, excepto no caso de uma operação de eliminação. Para operações de eliminação, tem de ser especificado exclusivamente este sinalizador.

Quando o conteúdo associado a todas as versões de um artigo de recurso tem de ser destruído, o utilizador terá de substituir a opção DK\_CM\_DESTROY\_DELETE pela opção DK\_ICM\_DESTROY\_ALL\_VERSIONS. A opção DK\_ICM\_DESTROY\_ALL\_VERSIONS está disponível apenas para operações de eliminação.

## Replicação assíncrona num gestor de recursos z/OS

A replicação assíncrona copia um objecto de um gestor de recursos do DB2 Content Manager para outro.

Para os objectos a replicar, tem de criar regras de replicação e associar as regras à recolha onde os objectos estão armazenados. Tem de fazê-lo antes de armazenar objectos no DB2 Content Manager. Se os objectos estiverem armazenados no DB2 Content Manager antes da criação e associação das regras de replicação, o DB2 Content Manager faculta um utilitário denominado IMPORTREPLICA para esse efeito. Para obter mais informações relativas ao utilitário IMPORTREPLICA, consulte as informações sobre administração no Information Center.

A funcionalidade de replicação assíncrona está incluída num módulo autónomo do z/OS gestor de recursos. Para concluir um processo de replicação assíncrona, efectue os seguintes passos:

1. Execute ICMMRMAP.JCL para iniciar o processo de replicação assíncrona, ICM MOSAP. A lista de parâmetros de entrada transferida para ICM MOSAP a partir de ICMMRMAP.JCL é verificada. A lista de parâmetros de entrada é apresentada seguinte forma:

### **DB2SUBSYSID (DB2C)**

4 bytes: O **DB2SUBSYSID** indica a localização da base de dados do gestor de recursos z/OS.

### **Base de dados do servidor de bibliotecas do DB2 Content Manager (NQADB2C)**

Nome alternativo utilizado para localizar uma determinada base de dados no subsistema UDB do DB2 especificado.

### **Localização do servidor de bibliotecas (LOCAL/REMOTE)**

Este valor indica se a base de dados do servidor de bibliotecas reside no mesmo subsistema UDB do DB2 que o gestor de recursos z/OS, ou se reside fora.

### **Nome do Plano (OSAPFVTA)**

8 bytes: O plano de associação para o processo de replicação assíncrona.

### **ID de Utilizador (IFVTA)**

Utilizador do DB2 Content Manager com autorização de Administrador do Sistema.

### **Palavra-passe de Utilizador (IFVTA1)**

Palavra-passe do Administrador do Sistema do DB2 Content Manager.

### **Nome do Gestor de Recursos (IMWEBSR1)**

Este nome indica a qual o gestor de recursos do z/OS a que está associado o processo assíncrono.

**Valor de Repouso (300)**

Duração (em segundos) para o processo fazer um pausa antes de reprocessar os artigos marcados para eliminação.

**Nível de Rastreio (0 – 3)**

Nível de registo. Os actuais valores suportados são 0, 1, 2 e 3.

**Nível 0: ERROR**

Regista apenas o relatório de resumo e os erros da replicação assíncrona.

**Nível 1: INFO**

Regista o estado de entrada e saída da função mais ERROR.

**Nível 2: DETAIL**

Regista a entrada e saída da função, resultados de sucessos e fracassos e INFO.

**Nível 3: DEBUG**

Regista instruções extensas utilizadas para depuração e INFO.

2. Preencha a estrutura `ReplicaItemStruct`. Se todos os parâmetros de entrada forem válidos, o processo de replicação assíncrona continuará para ligar à base de dados do servidor de bibliotecas para obter informações da recolha de dados e do gestor de recursos no sistema para configurar o processo de replicação. Além disto, a função `readToBeReplicated` é invocada para buscar as informações dos artigos a replicar na tabela `Asynch` do gestor de recursos e armazená-las na estrutura `ReplicaItemStruct`. É esta estrutura que conduz o processo de replicação assíncrona.
3. Após a estrutura `ReplicaItemStruct` ter sido preenchida, a replicação assíncrona invoca diversas funções de ajudante para concluir as tarefas. As funções mais importantes são:

**mapRMData()**

Esta função toma a estrutura `ReplicaItemStruct` como um dos parâmetros e utiliza as suas informações para validar os gestores de recursos que a tabela `Asynch` apresenta, comparando-as com as informações obtidas do servidor de bibliotecas para assegurar a sua existência. Após boa verificação, os gestores de recursos origem e destino são correlacionados apropriadamente na estrutura `ReplicaItemStruct`.

**mapCollData()**

Esta função também toma a estrutura `ReplicaItemStruct` como um dos parâmetros e utiliza as suas informações para validar as recolhas que a tabela `Asynch` apresenta, comparando-as com as informações de recolha obtidas do servidor de bibliotecas para assegurar a sua existência. Após boa verificação, as recolhas origem e destino são correlacionadas apropriadamente na estrutura `ReplicaItemStruct`. Esta função só é invocada depois dos gestores de recursos de origem e de destino terem sido devidamente correlacionados.

**checkoutItems()**

Esta função também toma a estrutura `ReplicaItemStruct` como um dos parâmetros e utiliza as suas informações para verificar os artigos a replicar. Depois de ter sido verificado, o artigo é bloqueado e as respectivas informações são inscritas na tabela `CHECK OUT` do servidor de bibliotecas.

**sendReplicasToTargetRM()**

Esta função toma a estrutura `ReplicaItemStruct` como um dos

parâmetros e utiliza as suas informações para determinar qual o artigo que tem de ser replicado e de que gestor de recursos e recolha para que gestor de recursos e recolha.

A função `sendReplicasToTargetRM()` cumpre as suas tarefas em vários passos:

- a. Primeiro, indaga junto a OAM para verificar se o objecto existe ou não. Caso exista, tenta obter o objecto.
- b. Depois, verifica qual o gestor de recursos de destino ou recolha de destino para onde este objecto tem de ser enviado.
- c. Finalmente, um pedido HTTP junto das informações do gestor de recursos de destino ou da recolha de destino e o objecto é enviado para o gestor de recursos de destino ou recolha de destino. E assim o objecto é replicado.

## Definições de tabelas e descrições de colunas

*Tabela 22. ICMRMCONTROL*

| ICMRMCONTROL                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |          |     |      |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|-----|------|
| STOPREPLICATOR                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | SMALLINT | NOT | NULL |
| STOPREPLICATOR: (0/1)) O bit de REPLICATOR determina se o processo de replicação assíncrona ICMRMAP pára de processar entradas depois de uma ronda ou não, e é também o método pelo qual o processo de replicação assíncrona é parado. Se estiver definido para um, será parado depois de uma ronda de processamento. Se estiver definido como 0, depende do valor de inactividade para determinar por quanto tempo o processo deverá ser parado até iniciar o processamento novamente. |          |     |      |

*Tabela 23. ICMRMASYNCH*

| ICMRMASYNCH |           |          |
|-------------|-----------|----------|
| ITEMID      | CHAR (26) | NOT NULL |
| VERSIONID   | CHAR (5)  | NOT NULL |
| OBJID       | CHAR (44) | NOT NULL |
| COLLNAME    | CHAR (44) | NOT NULL |
| PRIMARY KEY | (OBJID)   |          |

### ITEMID

Valor do ID de artigo gerado pelo servidor de bibliotecas durante a transacção de armazenamento do objecto. Exemplo:  
A1001001A03L09B64813I92553

### VERSIONID

Se criação de versões estiver activada, este valor apresenta uma determinada instância de um objecto. Exemplo: 1

### OBJID

É o ID de artigo e o ID de versão combinado e delimitado por pontos (.), representa um localizador de objecto exclusivo do DB2 Content Manager no Object Access Method (OAM). Exemplo:  
A1001001.A03L09.B64813.I92553.V001

### COLLNAME

sourcecollname é a recolha na qual o objecto foi originalmente armazenado e a origem da operação de busca prévia. Exemplo: CLLCT001

Tabela 24. ICMSTITEMSTODELETE

| ICMSTITEMSTODELETE |           |          |
|--------------------|-----------|----------|
| ITEMID             | CHAR (26) | NOT NULL |
| VERSIONID          | SMALLINT  |          |
| RCODE              | SMALLINT  |          |

#### RCODE

O nome do gestor de recursos dado pelo utilizador é utilizado para correlacionar o RCODE, que, por sua vez, aponta para uma entrada na tabela. Exemplo: 1

## Armazenar conteúdo numa recolha especificada pelo utilizador através da API de Java ou C++

Ao armazenar objectos, pode especificar uma recolha de gestor de recursos alternativa na aplicação personalizada alterando a classe DKLobICM.

O privilégio ItemSetRMCollection é adicionado a um conjunto existente de privilégios na classe DKLobICM. O privilégio ItemSetRMCollection permite substituir a recolha de gestor de recursos definida pelo administrador. O nome do gestor de recursos e o nome da recolha têm de ser válidos e existir no sistema do DB2 Content Manager. Se não especificar o nome do gestor de recursos e o nome da recolha, poderá obter uma excepção DKUsageError. Terá também de definir o nome da recolha no gestor de recursos especificado e não em outros gestores de recursos. Se não definir o nome da recolha no gestor de recursos especificado, poderá obter uma excepção DKUsageException. Para obter mais informações, consulte a classe DKLobICM, no manual *Application Programming Reference*.

### Exemplo

Para armazenar o conteúdo associado a itens de partes de documento e de recurso com qualquer localização especificada pelo utilizador a um nível de artigo, a API de Java e de C++ fornece o seguinte método:

#### DKStorageManageInfoICM::setStoreLocation( string ResourceManagerName, string collectionName)

Certifique-se de os parâmetros deste método, o nome do gestor de recursos e o nome da recolha são válidos.

##### Artigo de recurso

```
DKTextICM xdo = (DKTextICM) datastore.createDDO(itemTypeName,
(int) DKConstantICM. DK_ICM_XDO_TEXT_CLASS_ID);
DKPidICM apid = (DKPidICM)xdo.getPidObject();
apid.setVersionNumber("1"); xdo.setMimeType(mimeType);
DKStorageManageInfoICM storageInfo = (DKStorageManageInfoICM)
xdo.getExtension("DKStorageManageInfoICM");
storageInfo.setStoreLocation(ResourceManagerName,
collectionName);
xdo.add(inputFileName); // Become persistent
```

##### Parte de documento

```
DKDDO ddo_doc = datastore.createDDO(itemTypeName,
DKConstant.DK_CM_FOLDER);
DKParts partColl = null;
short dataId =
ddo_doc.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
DKConstant.DK_CM_DKPARTS);
if (dataId == 0) {
```

```

 dataId = ddo_doc.addData(
DKConstant.DK_CM_NAMESPACE_ATTR,DKConstant.DK_CM_DKPARTS);
 partColl = new DKParts();
 ddo_doc.setData(dataId, partColl);
 } else {
 partColl = (DKParts)ddo_doc.getData(dataId);
 if (partColl == null){
 partColl = new DKParts();
 ddo_doc.setData(dataId, partColl);
 }
 }
 DKTextICM part_xdo = (DKTextICM)datastore.
 createDDO(partName,
 (int) DKConstantICM. DK_ICM_XDO_TEXT_CLASS_ID);
 DKPidICM apid = (DKPidICM)part_xdo.getPidObject();
 apid.setVersionNumber("1");
 part_xdo.setMimeType(mimeType);
 DKStorageManageInfoICM storageInfo =
 (DKStorageManageInfoICM)
 part_xdo.getExtension("DKStorageManageInfoICM");
 storageInfo.setStoreLocation(resourceManagerName,
 collectionName);
 part_xdo.loadDataFromFile(inputFileName);
 part_xdo.setAddLocationOption("MEMORY");
 partColl.addElement(part_xdo);
 ddo_doc.add();

```

No exemplo anterior, pode substitui o método `DKStorageManageInfoICM::setStoreLocation( string resourceManagerName, string collectionName)` pelo método `DKStorageManageInfoICM::setStoreSite(string resourceManagerName)` e o método `DKStorageManageInfoICM::setCollectionName(string collectionName)`.

**Importante:** Esta função destina-se apenas ao armazenamento de itens criados no gestor de recursos. Para funções update, se especificar os parâmetros configuráveis **RMName** e **SMSCollectionName**, o DB2 Content Manager não considera a entrada de dados e continua a actualizar o par de parâmetros **RMName** e **SMSCollectionName** em que o objecto foi armazenado.

## API para o gestor de recursos na Versão 8.4

A API in DB2 Content Manager Versão 8.4 permite aos clientes gerir remotamente o gestor de recursos através de APIs, do cliente de administração do sistema e do WebSphere Application Server.

O DB2 Content Manager Versão 8.4 fornece uma interface para que as aplicações executem as seguintes funções:

- Configurar as propriedades do Tivoli Storage Manager (TSM) a partir de uma localização remota e armazenar as propriedades na base de dados do gestor de recursos
- Controlar a migração de objectos com base em valores de limiar elevado e baixo
- Parar e iniciar o gestor de recursos e verificar o estado dos diferentes serviços do gestor de recursos

Os melhoramentos de API do gestor de recursos eliminam a necessidade de que os administradores do sistema tenham acesso físico à máquina que aloja o gestor de recursos. No DB2 Content Manager Versão 8.4, o gestor de recursos suporta também conjuntos de unidades de rede do WebSphere Application Server. A

capacidade de gestão dos conjuntos de unidades numa rede proporciona a capacidade de gestão do gestor de recursos a partir de uma localização remota.

**Importante:** Os melhoramentos ao gestor de recursos são permitidos pelas APIs de Java do conector do ICM. É necessário instalar o DB2 Content Manager Versão 8.4 para utilizar os melhoramentos de API do gestor de recursos.

As seguintes classes no pacote com.ibm.mm.sdk.rm foram modificadas para suportar estas interfaces.

## DKRMConstantSMS

As seguintes constantes foram adicionadas á classe DKRMConstantSMS para suportar políticas de migração flexíveis em limiares elevados e baixos.

Tabela 25. Novas constantes na classe DKRMConstantSMS

| Constantes                     | Finalidade                                                                                    | Valor |
|--------------------------------|-----------------------------------------------------------------------------------------------|-------|
| SMS_ALLOWED_MIN_LOW_THRESHOLD  | Indica o valor mínimo permitido para um limiar mínimo (retome o limiar para activar o volume) | 0     |
| SMS_ALLOWED_MAX_HIGH_THRESHOLD | Indica o valor máximo permitido para um limiar elevado (limiar para desactivar o volume)      | 100   |
| SMS_ALLOWED_MIN_HIGH_THRESHOLD | Indica o valor mínimo permitido para um limiar elevado (limiar para desactivar o volume)      | 1     |

## DKRMServerDefinitionDefSMS

Os métodos seguintes foram adicionados à classe DKRMServerDefinitionDefSMS.

Tabela 26. Novos métodos na classe DKRMServerDefinitionDefSMS

| Métodos             | Finalidade                                                                                                                                                                                                                                         |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| setDSMConfig        | Define o caminho totalmente qualificado para o ficheiro de opções do cliente do TSM                                                                                                                                                                |
| getDSMConfig        | Devolve o caminho totalmente qualificado para o ficheiro de opções do cliente do TSM                                                                                                                                                               |
| setTSMBufferSize    | Define o tamanho da memória tampão a utilizar para transferência de dados entre o gestor de recursos e o servidor do TSM                                                                                                                           |
| getTSMBufferSize    | Devolve o tamanho da memória tampão a utilizar para transferência de dados entre o gestor de recursos e o servidor do TSM                                                                                                                          |
| setTSMFileSpaceName | Define o nome do espaço de ficheiros do TSM. O TSM determina o tamanho da memória tampão da localização de conteúdo que deve ser utilizado para transferência de dados com o servidor do TSM através da utilização do método setTSMFileSpaceName.  |
| getTSMFileSpaceName | Devolve o nome do espaço de ficheiros do TSM. O TSM determina o tamanho da memória tampão da localização de conteúdo que deve ser utilizado para transferência de dados com o servidor do TSM através da utilização do método getTSMFileSpaceName. |
| setTSMFileSpaceInfo | Define um identificador dos objectos do gestor de recursos no espaço de ficheiros do TSM                                                                                                                                                           |
| getTSMFileSpaceInfo | Devolve um identificador dos objectos do gestor de recursos no espaço de ficheiros do TSM                                                                                                                                                          |

Consulte a documentação de referência de Javadoc para obter mais informações.

## Exemplo

O exemplo que se segue demonstra a utilização de alguns dos métodos acima mencionados na classe DKRMServerDefinitionDefSMS:

```
DKRMConfigurationMgmtICM _rmCfgMgmt =
new DKRMConfigurationMgmtICM(dsICM);
DKRMServerDefinitionDefSMS _serverdefObj = null;
DKResourceMgrDefICM _resMgrDef =
 (DKResourceMgrDefICM)_rmCfgMgmt.retrieveResourceMgr(rmName);
// Connect to resource manager
DKRMStoreSMS _rmStoreSMS = new DKRMStoreSMS(_resMgrDef);
_rmStoreSMS.connect(_resMgrDef, "");
_serverdefObj = new DKRMServerDefinitionDefSMS(_rmStoreSMS);
_serverdefObj.retrieve();
// Set fully qualified path for TSM client properties file
_serverdefObj.setDSMConfig("/opt/tivoli/tsm/client/api/bin/dsm.opt");
// Sets the Tivoli Storage Manager file space name
_serverdefObj.setTSMFileSpaceName("/dev/hda3");
// Set the identifier of Resource Manager objects within the TSM file space
_serverdefObj.setTSMFileSpaceinfo("RM1_Objects");
// Persist the information in the Resource Manager
_serverdefObj.update();
```

## DKRMVolumeDefSMS

Os novos métodos que se seguem, que foram adicionados à classe DKRMVolumeDefSMS, proporcionam uma forma mais flexível e granular para controlar a migração dos objectos:

Tabela 27. Novos métodos na classe DKRMVolumeDefSMS

| Métodos          | Finalidade                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| setHighThreshold | Define um limiar elevado (limiar para desactivar o volume) a partir do qual o volume é considerado completo. Se este valor for excedido, o migrador poderá mover objectos até que o limiar baixo (retomar o limiar para activar o volume) seja atingido, de modo a manter disponível espaço de disco suficiente. O valor de limiar elevado (limiar para desactivar o volume) tem de estar no intervalo de SMS_ALLOWED_MAX_HIGH_THRESHOLD e SMS_ALLOWED_MIN_HIGH_THRESHOLD . Este valor não pode ser inferior ao valor do limiar baixo (retomar limiar para activar o volume). |
| getHighThreshold | Devolve um limiar elevado (limiar para desactivar o volume) a partir do qual o volume é considerado completo.                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| setLowThreshold  | Define um limiar baixo (retomar limiar para activar o volume) no qual o migrador para a migração de limiar e o volume se torna disponível novamente. Este valor tem de ser superior a SMS_ALLOWED_MIN_LOW_THRESHOLD. Este valor não pode ser superior ao valor do limiar elevado (limiar para desactivar o volume).                                                                                                                                                                                                                                                           |
| getLowThreshold  | Devolve o limiar baixo                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |

Os seguintes métodos na classe DKRMVolumeDefSMS estão obsoletos:

### setThreshold

Obsoleto. Utilize os métodos setHighThreshold e setLowThreshold, em alternativa.



## getThreshold

Obsoleto. Utilize os métodos getHighThreshold e getLowThreshold , em alternativa.

Consulte o Javadoc para obter mais informações.

## Exemplo

O exemplo que se segue demonstra a utilização do método setHighThreshold na classe DKRMVolumeDefSMS:

```
DKRMConfigurationMgmtICM _rmCfgMgmt =
new DKRMConfigurationMgmtICM(dsICM);
DKResourceMgrDefICM _resMgrDef =
 (DKResourceMgrDefICM)_rmCfgMgmt.retrieveResourceMgr(rmName);
// Connect to resource manager
DKRMStoreSMS _rmStoreSMS = new DKRMStoreSMS(_resMgrDef);
_rmStoreSMS.connect(_resMgrDef, "");
DKRMVolumeDefSMS volume = DKRMVolumeDefSMS(_rmStoreSMS);
// Set high threshold value
volume.setHighThreshold(50);
// Set low threshold value
volume.setLowThreshold(10);
// Persist the information in the Resource Manager
volume.update();
```

## DKRMControlServiceSMS

Existe apenas um método significativo, DKRMConstantSMS.CONTROL\_SERVICE\_RESPONSE control, na classe DKRMControlServiceSMS:

```
public DKRMConstantSMS.CONTROL_SERVICE_RESPONSE control
(DKRMConstantSMS.CONTROL_SERVICE_NAME service,
 DKRMConstantSMS.CONTROL_SERVICE_ACTION action)
```

O método DKRMConstantSMS.CONTROL\_SERVICE\_RESPONSE control recebe um nome de serviço e uma acção como parâmetros de entrada e envia uma ordem ao gestor de recursos para executar a acção especificada no serviço indicado. O gestor de recursos envia uma resposta com a indicação do resultado ou o estado da acção executada.

Consulte o Javadoc para obter mais informações.

## Exemplo

O exemplo que se segue demonstra a utilização do método na classe DKRMControlServiceSMS:

```
\DKRMConfigurationMgmtICM _rmCfgMgmt =
new DKRMConfigurationMgmtICM(dsICM);
DKResourceMgrDefICM _resMgrDef =
 (DKResourceMgrDefICM)_rmCfgMgmt.retrieveResourceMgr(rmName);
// Connect to resource manager
DKRMStoreSMS _rmStoreSMS = new DKRMStoreSMS(_resMgrDef);
_rmStoreSMS.connect(_resMgrDef, "");
DKRMControlServiceSMS controlService = DKRMControlServiceSMS(_rmStoreSMS);
// Get the status of migrator
DKRMConstantSMS.CONTROL_SERVICE_RESPONSE response =
 controlService.control(DKRMConstantSMS.CONTROL_SERVICE_NAME.migrator,
 DKRMConstantSMS.CONTROL_SERVICE_ACTION.status);
System.out.println("Response = " + response);
```

---

## Gerir documentos em DB2 Content Manager

O DB2 Content Manager implementa um flexível modelo de dados de gestão de documentos que pode utilizar para gerir objectos empresariais. Os elementos básicos do modelo de dados incluem pastas, documentos e objectos.

Como já foi anteriormente mencionado, os documentos, pastas e outros objectos são todos representados por artigos do sistema de DB2 Content Manager. No nível de API, a única diferença entre um documento e uma pasta é o tipo semântico e a respectiva funcionalidade de DKFolder. Um documento é formado por atributos ou metadados, que descrevem o documento, incluindo atributos de valor único (documento, nome, data, assunto), atributos multi-valor (palavras-chave) e recolhas de atributos multi-valor (morada, formado por rua, cidade e código postal).

O modelo de dados de gestão de documentos utiliza partes de documento para associar objectos (artigos de recurso) ao documento. Este modelo suporta mais do que uma parte para construção de um documento. Por exemplo, cada página pode ser uma parte distinta. Para que uma aplicação determine a ordem das partes que compõem um documento, um número de parte é armazenado nas partes de documento. As partes de documento contêm um indicador do objecto (um atributo de referência), que contém outras informações relativas à parte como, por exemplo, o tipo MIME, o tamanho, o ID de gestor de recursos, que contém a parte, o nome de recolha desse gestor de recursos, etc. Cada objecto pode ter diferentes atributos. Por exemplo, uma anotação pode ter X e Y coordenadas, enquanto um registo de nota pode ter o CCSID do texto da nota.

Para ajudar o utilizador a compreender o modelo de dados de gestão de documentos, tenha em consideração a seguinte hipótese onde um utilizador importa um documento utilizando uma aplicação cliente:

- É apresentada uma janela ao utilizador.
- O utilizador introduz (ou selecciona) o nome do ficheiro a importar para o sistema. Por exemplo, um relatório da polícia.
- O utilizador selecciona o tipo de documento (memorando, participação, plano).
- Abre-se uma nova janela. Nesta janela, estão os campos onde o utilizador pode introduzir atributos que descrevem o documento. Por exemplo, a data, número de participação e número da apólice de seguro.
- O utilizador define uma parte de um documento e introduz alguns valores destinados aos atributos. Por exemplo, o relatório da polícia é uma parte de documento de uma participação.
- O utilizador termina de inserir as descrições do documento e conclui a tarefa. O relatório da polícia é criado.

Utilizando as APIs ou JavaBeans, a aplicação cliente estabelece ligação ao servidor de bibliotecas e ao gestor de recursos. O sistema cria dois artigos (um artigo sem recursos e um artigo de recurso) para armazenar o documento. São criados dois artigos porque o relatório da polícia contém uma fotografia, que é armazenada no gestor de recursos. O documento é criado com uma única chamada de API. O objecto é então armazenado no gestor de recursos e o gestor de recursos devolve a marca de hora e outros metadados relativos ao objecto. O gestor de recursos cria um atributo de referência para o objecto e insere o atributo de referência no componente descendente do documento. É efectuada uma última chamada ao servidor de bibliotecas para armazenar componentes descendentes e para actualizar os atributos. Todo o processo está limitado por uma transacção, de forma a que nenhuma API gere um documento parcialmente criado.

Após ter criado um documento, pode actualizá-lo. Pode executar dois tipos de actualizações: pode alterar os metadados ou alterar o conteúdo. O servidor de bibliotecas cria automaticamente um novo registo de artigo com o número de versão seguinte (se o artigo estiver activado para versões) e copia todos os componentes descendentes que estão associados ao artigo.

---

## Criar o modelo de dados de gestão de documentos

Esta secção ajuda a executar as tarefas principais associadas ao modelo de dados de gestão de documentos.

### Criar um tipo de artigo de documento

Para criar um tipo de artigo de documento, siga os passos indicados nos exemplos que se seguem.

#### Criar um tipo de artigo de documento: Java

1. Crie um ItemType documento com a classificação ItemType = DK\_ICM\_ITEMTYPE\_CLASS\_DOC\_MODEL e defina o seguinte.

```
docItemTypeDef.setVersionControl
 ((short)DKConstantICM.DK_ICM_VERSION_CONTROL_ALWAYS);
docItemTypeDef.setVersioningType
 (DKConstantICM.DK_ICM_ITEM_VERSIONING_FULL);
docItemTypeDef.setDeleteRule(DKConstantICM.DK_ICM_DELETE
 _RULE_CASCADE);
```
2. Crie uma relação de ItemType para adicionar Parts ao documento. Obtenha EntityDef para cada Part.

```
Parttype = (DKItemTypeDefICM) dsDef.retrieveEntity(PartName);
```
3. Para cada parte, crie uma relação de ItemType e defina os valores.

```
DKItemTypeRelationDefICM itRel = new DKItemTypeRelationDefICM(ds);
itRel.setTargetItemTypeID(PartName);
itRel.setDefaultRMCode((short)1);
itRel.setDefaultACLCode(DKConstantICM.DK_ICM_SUPER_USER_ACL);
itRel.setDefaultCollCode((short)1);
itRel.setDefaultPrefetchCollCode((short)1);
itRel.setVersionControl(DK_ICM_VERSION_CONTROL_NERVER);
```
4. Adicione a relação de ItemType ao Document (Source).

```
docItemTypeDef.addItemTypeRelation(itRel);
```
5. Adicione o ItemType documento ao armazenamento persistente.

```
docItemTypeDef.add();
```

#### Criar um exemplo de tipo de artigo de documento: C++

1. Obtenha o objecto de definição do servidor de conteúdos.

```
DKDatastoreDefICM * dsDefICM =
 (DKDatastoreDefICM *)dsICM->datastoreDef();
```
2. Obtenha o objecto de administração do servidor de conteúdos.

```
DKDatastoreAdminICM * pdsAdmin =
 (DKDatastoreAdminICM *)dsDefICM->datastoreAdmin();
DKItemTypeDefICM * itemType = NULL;
DKItemTypeRelationDefICM * itemTypeRel = NULL;
DKAttrDefICM* attr = NULL;
```
3. Obtenha um atributo para o tipo de artigo documento. Se o atributo não existir, crie-o.

```
dkAttrDef *pAttr = dsDefICM->retrieveAttr("docTitle1");
if(pAttr == NULL)
{
 attr = new DKAttrDefICM(dsICM);
```

```

 attr->setName("docTitle1"); //attribute name column name
 attr->setType(DK_CM_CHAR);
 attr->setSize(100);
 attr->setNullable(false);
 attr->setUnique(false);
 attr->add();
 pAttr = attr;
 }
 itemType = new DKItemTypeDefICM(dsICM);
 itemType->setName("DocModelTest");
 itemType->setDescription("This is a test Item Type");
 itemType->setClassification(DK_ICM_ITEMTYPE_CLASS_DOC_MODEL);
 itemType->setAutoLinkEnable(false);
 itemType->setVersionControl((short)DK_ICM_VERSION_CONTROL_ALWAYS);
 itemType->setVersioningType(DK_ICM_ITEM_VERSIONING_FULL);
 itemType->addAttr(pAttr);

```

4. Crie uma relação entre o documento que acabou de criar e part1. Para isso, obtenha primeiro EntityDef para cada Part.

```

DKItemTypeDefICM *itemTypePart1 = (DKItemTypeDefICM *)
 dsDefICM->retrieveEntity("ICMBASE");
//int part1ITypeId = itemTypePart1->getItemTypeId();
int part1ITypeId = itemTypePart1->getIntId();

```

5. Para cada parte, crie uma relação de tipo de artigo e defina os valores.

```

DKItemTypeRelationDefICM *itemTypeRelPart1=
 new DKItemTypeRelationDefICM(dsICM);
itemTypeRelPart1->setTargetItemTypeID(part1ITypeId);
//sets the default resource manager
itemTypeRelPart1->setDefaultRMCode((short)1);
//sets the default ACL code
itemTypeRelPart1->setDefaultACLCode(1);

```

6. Defina a recolha predefinida na qual os recursos de artigos correspondentes a este tipo de artigo devem ser armazenados.

```

itemTypeRelPart1->setDefaultCollCode((short)1);

```

7. Defina a recolha de busca prévia predefinida na qual os recursos de artigos correspondentes a este tipo de artigo devem ser armazenados.

```

itemTypeRelPart1->setDefaultPrefetchCollCode((short)1);
itemTypeRelPart1->setVersionControl((short)DK_ICM_VERSION
 _CONTROL_NEVER);
itemTypeRelPart1->setSourceItemTypeID(itemType->getIntId());

```

8. Adicione a relação de tipo de artigo ao documento (origem).

```

itemType->addItemTypeRelation(itemTypeRelPart1);

```

9. Crie uma relação entre o documento que acabou de criar e part2. Para isso, obtenha primeiro EntityDef para cada parte.

```

DKItemTypeDefICM *itemTypePart2 = (DKItemTypeDefICM *)
 dsDefICM->retrieveEntity("ICMANNOTATION");
int part2ITypeId = itemTypePart2->getIntId();

```

10. Para cada parte, crie uma relação de tipo de artigo e defina os valores.

```

DKItemTypeRelationDefICM * itemTypeRelPart2= new
 DKItemTypeRelationDefICM(dsICM);
itemTypeRelPart2->setTargetItemTypeID(part2ITypeId);
itemTypeRelPart2->setDefaultRMCode((short)1);
itemTypeRelPart2->setDefaultACLCode(1);
itemTypeRelPart2->setDefaultCollCode((short)1);
itemTypeRelPart2->setDefaultPrefetchCollCode((short)1);

itemTypeRelPart2->setVersionControl((short)DK_ICM_VERSION
 _CONTROL_NEVER);

```

```

itemTypeRelPart2->setSourceItemTypeID(itemType->getIntId());

```

11. Adicione a relação de tipo de artigo ao documento (origem).

```
itemType->addItemTypeRelation(itemTypeRelPart2);
```

12. Atualize a definição do tipo de artigo no servidor de bibliotecas.

```
itemType->add();
```

Para obter informações adicionais, consulte o exemplo SItemTypeCreationICM.

## Criar um documento

Um artigo com o tipo semântico "Documento" pode conter atributos (como os artigos dos mesmos tipos semânticos) e várias "partes" (ao contrário de artigos de outros tipos semânticos) no seu interior.

Os passos seguintes explicam o processo de criação de um artigo (com base num tipo de artigo de documento predeterminado) que contém um atributo e uma "parte". Tenha em atenção que, nos passos seguintes, parte-se do princípio que já foi definido um tipo de artigo denominado `s_simple`, com um atributo, denominado `S_varchar` e uma parte ("ICMBASE").

### Criar um exemplo de documento: Java

1. Crie o DDO do documento.

```
DKDDO ddoDocument = dsICM.createDDO("S_simple",
 DKConstant.DK_CM_DOCUMENT);
short dataId = 0;
String attrValue = "Test";
```

2. Defina o atributo do documento. Neste caso, partimos do princípio que o tipo de artigo só possui um atributo.

```
dataId = ddoDocument.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
 "S_varchar");
ddoDocument.setData(dataId, attrValue);
DKParts parts = null;
// Document's parts
```

3. Aceda à recolha de partes do documento.

```
dataId = ddoDocument.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
 DKConstantICM.DK_CM_DKPARTS);
if (dataId == 0) {
 dataId = ddoDocument.addData(DKConstant.DK_CM_NAMESPACE
 _ATTR,
 DKConstantICM.DK_CM_DKPARTS);
 parts = new DKParts();
 ddoDocument.setData(dataId, parts);
}
else
{
 parts = (DKParts)ddoDocument.getData(dataId);
 if (parts == null)
 {
 parts = new DKParts();
 ddoDocument.setData(dataId, parts);
 }
}
```

4. Crie uma parte de tipo predeterminado "ICMBASE". Esta parte será adicionada ao documento criado. Parte-se do princípio que o documento abaixo criado tem por base um tipo de artigo com apenas uma parte.

```
DKLobICM pLobPart = (DKLobICM) dsICM.createDDO("ICMBASE",
 DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASE);
pLobPart.setPartNumber(1);
// Set the mime type for added part
pLobPart.setMimeType("text/plain");
String partValue = "This is a base part";
pLobPart.setContent(partValue.getBytes());
```

5. Adicione a parte criada à recolha de "partes".

**Importante:** Esta é uma acção de guardar diferida (a alteração não é consolidada no arquivo de dados até que o DDO do documento seja tornado persistente).

```
parts.addElement((dkDataObjectBase)((DKDDO) pLobPart));
```

6. Torne persistente o documento no arquivo de dados.

```
ddoDocument.add();
```

## Criar um exemplo de documento: C++

1. Crie um novo DDO de tipo DocModelTest e tipo semântico DK\_CM\_DOCUMENT

```
DKDatastoreDefICM* pdsDef = (DKDatastoreDefICM*) dsICM->datastoreDef();
DKDDO* ddoDocument = dsICM->createDDO("DocModelTest",DK_CM_DOCUMENT);
ddoDocument->setData(ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,
 DKString("docTitle")),DKString("this is a string value"));
DKItemTypeDefICM* itemType = (DKItemTypeDefICM*)
 pdsDef->retrieveEntity("DocModelTest");
```

2. Obtenha o conjunto de objectos DKItemTypeRelationDefICM para o tipo de artigo origem fornecido do arquivo persistente

```
dkCollection* pRelationColl = itemType->retrieveItemTypeRelations();
dkIterator* pIter = pRelationColl->createIterator();
int noOfPartsToCreate = pRelationColl->cardinality();
DKParts* pPartColl= NULL;
```

3. Crie a recolha de partes para o objecto, se ainda não existir

```
short dataId = ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS);
if (dataId ==0) {
 dataId = ddoDocument->addData(DK_CM_NAMESPACE_ATTR,DK_CM_DKPARTS);
 pPartColl = new DKParts();
 ddoDocument->setData(dataId,pPartColl);
}
else {
 pPartColl = (DKParts*) (ddoDocument->getData(dataId).value());
 if (pPartColl ==NULL) {
 pPartColl = new DKParts();
 ddoDocument->setData(dataId,pPartColl);
 }
}
int i=0;
DKItemTypeRelationDefICM* itemTypeRelPart =NULL;
DKItemTypeDefICM* pEnt =NULL;
// CV v8 BLOB
DKLobICM* pPart =NULL;
DKString str = "This is to test the document model with two parts";
while(pIter->more()) {
 i=i+1;
 itemTypeRelPart=(DKItemTypeRelationDefICM*) pIter->next()->value();
 pEnt = (DKItemTypeDefICM*)
 ((DKDatastoreDefICM*) pdsDef)->retrieveEntity(
 (long)itemTypeRelPart->getTargetItemTypeID());
 pPart = (DKLobICM*) dsICM->createDDO(pEnt->getName(), DK_CM_RESOURCE);
 pPart->setPartNumber(i);
 pPart->setContent(str);

 DKAny any = (dkDataObjectBase*) pPart;
 pPartColl->addElement(any);
}
```

4. Adicione o DDO ao arquivo de dados

```
ddoDocument->add()
```

Para obter mais informações, consulte o exemplo SDocModelItemICM.

## Actualizar um documento

Os passos desta secção explicam o processo de actualização de um artigo de tipo semântico "Documento."

Nos passos seguintes, uma parte nova é adicionada e o valor do atributo é actualizado.

### Actualizar um exemplo de documento: Java

1. Actualizar o valor do atributo do artigo de documento.

```
String attrValue = "New Value";
short dataId=ddoDocument.dataId
(DKConstant.DK_CM_NAMESPACE_ATTR,"S_varchar");
ddoDocument.setData(dataId,attrValue);
```

2. Aceda à recolha de partes do documento.

```
DKParts parts = null;
// Document's parts
dataId = ddoDocument.dataId(DKConstant.DK_CM_NAMESPACE_ATTR,
DKConstantICM.DK_CM_DKPARTS);
if (dataId == 0) {
 dataId = ddoDocument.addData(DKConstant.DK_CM_NAMESPACE_ATTR,
DKConstantICM.DK_CM_DKPARTS);
 parts = new DKParts();
 ddoDocument.setData(dataId, parts);
}
else
{
 parts = (DKParts)ddoDocument.getData(dataId);
 if (parts == null)
 {
 parts = new DKParts();
 ddoDocument.setData(dataId, parts);
 }
}
```

3. Criar dados para a nova parte.

```
String partValue = "This is an annotation";
```

4. Criar uma parte de tipo predeterminado "ICMANNOTATION". Esta parte será adicionada ao documento criado. Aqui, parte-se do princípio que o documento a ser criado tem por base um tipo de artigo com apenas uma parte. Após a nova parte ser adicionada, o documento terá duas partes.

```
DKLobICM pLobPart = (DKLobICM)dsICM.createDDO("ICMANNOTATION",
DKConstantICM.DK_ICM_SEMANTIC_TYPE_BASE);
pLobPart.setContent(partValue.getBytes());
pLobPart.setPartNumber(2);
```

5. Adicione a parte criada à recolha de "partes". Esta é uma acção de guardar diferida (a alteração não é consolidada no arquivo de dados até que o DDO do documento seja tornado persistente).

```
parts.addElement((dkDataObjectBase)((DKDDO) pLobPart));
```

6. Torne persistente o documento alterado no arquivo de dados.

```
ddoDocument.update();
```

### Actualizar um exemplo de documento: C++

1. Actualizar o valor do atributo do artigo de documento.

```
DKString attrValue = "New Value";
short dataId=ddoDocument->dataId
(DK_CM_NAMESPACE_ATTR,"S_varchar");
ddoDocument->setData(dataId,attrValue);
```



2. Aceder à colecção de partes do documento

```
// Aceder à colecção DKParts
DKParts* parts = null;
// Document's parts
dataId = ddoDocument->dataId(DK_CM_NAMESPACE_ATTR,
DK_CM_DKPARTS);
if(dataId == 0)
{ dataId = ddoDocument->addData(DK_CM_NAMESPACE_ATTR, DK_CM_DKPARTS);
parts = new DKParts();
ddoDocument->setData(dataId, parts);
}else{
parts = (DKParts*)(dkCollection*)ddoDocument->getData(dataId);
if (parts == null)
{parts = new DKParts();
ddoDocument->setData(dataId, parts);
}
}
```

3. Adicione uma nova parte ao documento.

```
DKLobICM* pPart1 =
(DKLobICM*) dsICM->createDDO ("ICMNOTELOG", DK_CM_RESOURCE);
pPart1->setPartNumber(3);
DKString pTempData = "This is to test the document model with two parts";
pPart1->setContent(pTempData);
DKAny any = (dkDataObjectBase*)(DKDDO*) pPart1;
pParts->addElement(any);
```

4. Actualize as informações de DDO no arquivo de dados.

```
ddoDocument->update();
```

Para obter mais informações, consulte o exemplo SDocModelItemICM.

## Obter e eliminar um documento

Para obter um documento com partes, utilize métodos de obtenção tal como em qualquer outro artigo.

No caso das partes de documentos (recolha DKParts), existe uma diferença entre a enumeração de partes e a obtenção de dados para as partes (incluindo atributos e conteúdo de recursos). Primeiro, terá de enumerar as partes preenchendo a recolha DKParts com DDOs em branco e os respectivos PIDs completos que representam os dados das partes. No entanto, pode também pedir dados de partes adicionais, incluindo atributos e conteúdo de recursos na mesma chamada para otimizar a obtenção. Além disso, pode sempre efectuar uma chamada subsequente para obter atributos ou outras informações para as próprias partes. Pode transferir a recolha DKParts na totalidade directamente para a obtenção multi-artigo (DKDatastoreICM::retrieveObjects(dkCollection,DKNVPair[]), uma vez que DKParts é um dkCollection que contém DKDDOs (como subclasse DKLobICM). Em alternativa, pode invocar a obtenção de artigo único em qualquer parte, individualmente.

Se pretender enumerar as partes, solicite DKRetrieveOptionsICM::partsList(true). Se pretender os atributos das partes (incluindo os atributos de recursos do sistema), inclua também DKRetrieveOptionsICM::partsAttributes(true). Se pretender o conteúdo de recursos das partes, inclua também DKRetrieveOptionsICM::resourceContent(true) (apenas para obtenção de artigo único). Existem considerações de desempenho muito importantes, entre outros detalhes, totalmente explicados no manual *Application Programming Reference*, para opções de obtenção. Para obter mais informações, consulte a documentação de DKRetrieveOptionsICM para cada opção.

Para eliminar um documento com partes, invoque a eliminação (ddo.del()) tal como em qualquer outro artigo. O documento e as partes a este anexadas são eliminados. Para obter mais informações relativas à obtenção e eliminação de documentos com partes, consulte o exemplo de API SDocModelItemICM.

## Criação de versões de partes do modelo de dados de gestão de documentos

As propriedades de criação de versões de partes de documentos são determinadas pelas propriedades de criação de versões da relação de tipo de artigo com cada tipo de parte seleccionado no tipo de artigo do documento.

As características de criação de versões de partes de documentos incluem as seguintes:

- Tal como os documentos normais, as partes podem ter um dos três modelos de criação de versões: criar sempre versões, nunca criar versões (predefinição) e criação de versões controlada pela aplicação.
- Se um tipo de artigo possuir uma política de versões de nunca criar versões, as partes do tipo de artigo possuem a mesma política de versões.
- Se um tipo de artigo T possuir uma política de versões de criar sempre versões e um artigo I de tipo de artigo T e o utilizador modificar (adicionando/eliminando ou actualizando uma parte) os seus atributos ou a sua recolha de partes, é criada uma nova versão do artigo I.
- As partes de documentos, ao contrário dos próprios documentos, não possuem um número máximo de versões.
- Pode obter regras de criação de versões, a nível de partes, no objecto de relação do tipo de artigo relativo à parte em questão (base, nota, anotação, etc.).

Os exemplos seguintes mostram como obter as regras de criação de versões para uma parte de base do tipo de artigo.

### Exemplo: Java

```
String itemTypeName="book"; //example item type
long partId = DK_ICM_PART_BASE;
DKItemTypeDefICM item =null;
DKDatastoreICM ds = new DKDatastoreICM();
...
item = (DKItemTypeDefICM)ds.datastoreDef.retrieveEntity(itemTypeName;
DKItemTypeRelationDefICM itemRelation =
(DKItemTypeRelationDefICM)item.retrieveItemTypeRelation(partId);
versionControlPolicy = itemRelation.getVersionControl();
```

### Exemplo: C++

```
DKString itemTypeName="book"; //example item type
//Specify the part id for which we need versioning information
long partId = DK_ICM_PART_BASE;
DKItemTypeDefICM * itemType;
//Retrieve the entity corresponding to the "book" item type
itemType =
(DKItemTypeDefICM *)dsICM->datastoreDef()->retrieveEntity(itemTypeName);
//Retrieve the relation object for the specified part id
DKItemTypeRelationDefICM * itemTypeRelation =
(DKItemTypeRelationDefICM*)itemType->retrieveItemTypeRelation(partId);
//Retrieve the version control policy for the specified part
int versionControlPolicy = itemTypeRelation->getVersionControl();
```

---

## Trabalhar com transacções

As transacções permitem ao DB2 Content Manager manter a coerência entre o servidor de bibliotecas e todos os gestores de recursos adjacentes.

Uma transacção é uma unidade de trabalho recuperável determinada pelo utilizador, que é formada por uma ou mais chamadas consecutivas de APIs efectuadas através de uma única ligação ao servidor de bibliotecas. A sequência de chamadas consecutivas do método DKDatastoreICM é efectuada directa ou indirectamente, através dos DDOs e XDOs.

O âmbito de uma transacção e a quantidade de trabalho envolvida na mesma é, por predefinição, o trabalho executado por um único método de API (transacção implícita). Este tipo de transacção é recomendado e consiste no melhor âmbito de rendimento de uma transacção. O utilizador pode, no entanto, alterar o âmbito de uma unidade de trabalho, tornando-a maior para incluir várias chamadas de método (transacção), mas a utilização deste tipo de transacção pode gerar algum tempo sistema de rendimento.

Quando termina uma transacção, toda a transacção é consolidada ou sofre um retrocesso. Se for consolidada, todas as alterações ao servidor do Content Manager efectuadas por chamadas de API dentro da transacção, passam a ser persistentes. Se forem removidas alterações a uma transacção ou esta falhar, todas as alterações efectuadas dentro da transacção são invertidas durante o processamento da remoção de alterações.

A consolidação e a remoção de uma transacção são executadas automaticamente no caso de transacções implícitas. O sistema do Content Manager inicia uma remoção de alterações quando ocorre um erro grave ou quando é necessário processar um bloqueio entre o servidor de bibliotecas e a base de dados.

Dentro de uma transacção, as alterações não consolidadas ao gestor de recursos só são visíveis à aplicação que efectuou as alterações após a transacção estar consolidada. Por exemplo, o utilizador efectua alterações a um artigo do gestor de recursos e armazena-o. Se obtiver esse artigo antes da transacção estar consolidada, o artigo não irá reflectir as alterações que acabou de efectuar. Só verá o artigo actualizado depois da consolidação da transacção.

Não são suportadas transacções concorrentes ou sobrepostas através de uma única ligação do servidor de bibliotecas. Para manter transacções concorrentes, o utilizador deve estabelecer várias ligações entre o servidor de bibliotecas e a base de dados ou iniciar vários processos ou módulos clientes, se estiver a trabalhar com uma aplicação cliente. As aplicações como a IBM WebSphere<sup>®</sup> Application Server operam processos, ligações e sessões.

Quando invocados, os métodos execute() e executeWithCallback() no DKDatastoreICM criam automaticamente uma ligação adicional à base de dados. A nova ligação à base de dados é então utilizada para executar a consulta. Visto que as consultas utilizam uma ligação diferente à base de dados, também possuem um âmbito de transacção diferente de outras operações de servidor de conteúdos. A ligação à base de dados é encerrada (ou devolvida ao conjunto, se esta opção estiver activada) quando o DKResultSetCursor é encerrado.

## Pontos a ter em consideração ao conceber transacções na aplicação

É necessário ter alguns pontos em consideração ao conceber transacções nas aplicações.

Se um nó cliente ou um servidor de bibliotecas falhar antes da consolidação da transacção, a função de obtenção da base de dados remove imediatamente as alterações à transacção no servidor de bibliotecas. As alterações ao gestor de recursos efectuadas durante a falha são imediatamente anuladas se o nó cliente e o gestor de recursos estiverem ambos activos. Se a falha tiver ocorrido no nó cliente, o utilizador deve submeter o gestor de recursos a um ciclo do Asynchronous Recovery Utility, de modo a repor a coerência entre o gestor de recursos e o servidor de bibliotecas. Antes da execução do utilitário, os servidores ainda possuem integridade de dados. Serão afectadas as operações dos artigos em curso que evidenciaram a falha e serão também rejeitadas até que o GR seja recuperado. Uma falha no decorrer de uma actualização de um objecto impede que se realize outra actualização desse mesmo objecto antes da primeira falha estar consolidada.

Se o gestor de recursos falhar, o utilizador deve executar o utilitário de obtenção assíncrona para remover incoerências. No OS/390, o gestor de recursos tem capacidades de transacção nativas, tal como o Object Access Method (OAM), que são utilizadas para obter mais rapidamente.

## Precauções ao utilizar transacções

Para transacções, nas quais controla o âmbito da transacção utilizando `DKDatastoreICM.startTransaction()` e `DKDatastoreICM.commit()`, deverá ter precauções ao desenvolver uma aplicação na qual trabalha com documentos do Content Manager com partes e ao executar operações create, retrieve, update e delete (CRUD) de `DKLobICM`.

Ao executar estas operações, deverá executar operações de CRUD próximas, o mais possível, do final da transacção. Também deverá manter uma transacção o mais breve possível, pois uma transacção longa aumenta o potencial de problemas de bloqueio da base de dados.

Os problemas de bloqueio são mais notórios ao actualizar um artigo e quando a aplicação opta por não consolidar imediatamente a transacção. Enquanto a transacção não for consolidada, o artigo que está a ser actualizado continua a ser visível a outras aplicações. Quando outro utilizador tenta aceder ou visualizar o artigo, esse utilizador fica bloqueado até que a transacção de actualização seja consolidada. O mesmo problema (bloqueio da base de dados) ocorre ao criar novos artigos numa pasta. Se a pasta for visível a outro utilizador, e esse utilizador tentar obter o novo artigo, o utilizador é bloqueado até que a transacção seja consolidada. O tempo anterior à consolidação da transacção corresponde ao tempo que o utilizador está bloqueado.

A melhor abordagem para evitar o bloqueio da base de dados é consolidar frequentemente as transacções e evitar transacções de execução longa. Se o utilizador tiver de executar operações de CRUD, recomenda-se que execute estas operações quando tiver a certeza de que ninguém irá aceder aos artigos a serem actualizados.

## Utilizar operações de dar entrada e saída em transacções

O DB2 Content Manager suporta operações check-out e check-in em itens.

A operação check-out é invocada para adquirir um bloqueio de escrita persistente para itens. Quando é dada saída de um artigo por parte de um utilizador, outros utilizadores não podem actualizá-lo, apesar de o poderem obter e visualizar. É necessário invocar a operação check-out antes de actualizar ou reindexar um artigo, independentemente do modo de transacção (implícito ou explícito) que utiliza. Ao terminar a acção no artigo, invoque a operação check-in para libertar o bloqueio persistente e disponibilizar o artigo para que possa ser actualizado por outros utilizadores. Após criar um artigo, o utilizador possui a opção de o manter em estado de saída para impedir que outros utilizadores o alterem antes de concluir o trabalho. Se der saída (ou entrada) de um artigo utilizando uma transacção explícita, esta operação será anulada se forem removidas alterações à transacção. Se der saída de um artigo utilizando uma transacção implícita, esta acção é consolidada. É da responsabilidade da aplicação voltar a dar entrada do artigo, utilizando opções ou métodos de entrada.

## Processar transacções

O âmbito da transacção pode ser controlado por uma chamada de API cliente, mas este controlo deve ser cuidadosamente concebido.

Para agrupar um conjunto de chamadas de API numa transacção, é necessário construir o conjunto executando os seguintes passos:

1. Invoque o método `startTransaction()` da classe `DKDatastoreICM`. O utilizador trabalhará com métodos `DKDatastoreICM` para concluir todos os passos da transacção.
2. Invoque todas as APIs que pretende incluir na transacção segundo a ordem que pretende que estas sejam invocadas.
3. Invoque os métodos de `commit` ou `rollback` para terminar a transacção.

### Importante:

Todas as chamadas de API efectuadas entre `startTransaction()` e `commit()` ou `rollback()`, são consideradas uma única transacção.

Todas as APIs podem ser incluídas em transacções, excepto se existir indicação em contrário. Consulte *Application Programming Reference* para obter mais informações. Algumas APIs administrativas não podem ser incluídas em transacções. Por exemplo, o método para definir ou actualizar tipos de artigos.

Segue-se a lista de métodos de classe envolvidos nas transacções do Content Manager relativamente à criação e actualização de itens:

#### **`DKDatastoreICM.startTransaction()`**

Inicia uma transacção.

#### **`DKDatastoreICM.commit()`**

Consolida as alterações da transacção efectuadas na base de dados.

#### **`DKDatastoreICM.rollback()`**

Remove alterações à transacção da base de dados.

#### **`DKDatastoreICM.checkOut()`**

Adquire um bloqueio de escrita persistente sobre um artigo.

#### **`DKDatastoreICM.checkIn()`**

Liberta um bloqueio de escrita persistente anteriormente adquirido.

**DKDatastoreICM.add()**

Cria um novo artigo na base de dados.

**DKDatastoreICM.updateObject()**

Actualiza um artigo. Deve ser dada saída do artigo antes de invocar este método.

**DKDatastoreICM.retrieveObject()**

Obtém um artigo da base de dados.

**DKDatastoreICM.deleteObject()**

Elimina um artigo da base de dados.

**DKDatastoreICM.moveObject()**

Reindexa um artigo. Move um artigo de um tipo de artigo para outro. Deve ser dada saída do artigo antes de invocar este método.

Outra fonte de informações relativas a transacções é o exemplo SItemUpdateICM.

## Comportamento das transacções explícitas na Versão 8.4

Se o sistema do DB2 Content Manager detectar um erro durante uma operação persistente, o comportamento predefinido do sistema é sempre remover automaticamente a transacção explícita.

Por exemplo, XYZ Seguros tem um sistema de pagamento de apólices online, e um utilizador submeteu as informações do seu cartão de crédito para processamento, para a uma repentina quebra de energia. O desconto do cartão de crédito e a actualização de pagamento da apólice são anulados pelo sistema.

A remoção automática de transacções explícitas é muito importante, dado que não pode depender da sua aplicação para reverter uma série de actualizações efectuadas antes a operação falhar, em particular para operações importantes. Talvez a ligação tenha caído ou sistema tenha bloqueado devido a falha de energia e não pode anular as transacções anteriores. A aplicação pode não memorizar o que estava a fazer antes do bloqueio se não tiver um mecanismo de registo e recuperação. Com uma transacção, o sistema é informado para iniciar o registo das alterações, mas largá-las se não lhe for dito explicitamente que já terminou.

Se pretender substituir o comportamento predefinido do sistema para transacções explícitas, consulte o exemplo STransactionsRollbackChangeICM.

## Comportamento de transacção na eliminação de um utilizador que não pertence a um grupo

Por predefinição, quando tenta eliminar um utilizador de um grupo e o utilizador não existe ou não pertence ao grupo, o método DKUserMgmt::update remove a transacção.

Se não pretende o comportamento predefinido na aplicação, e pretende poder eliminar um utilizador, mesmo que não pertença ao grupo, pode utilizar um novo método. Pode encontrar o novo método em DKUserMgmt. De seguida, é apresentada a assinatura do método:

```
public void update(dkUserGroupDef userGroup, Vector v, int action, int option)
throws DKNotExistException, DKException, Exception
```

A constante DK\_ICM\_DO\_NOT\_ROLLBACK\_ON\_ERROR, de DKConstantICM.java, permite sobrepor o comportamento predefinido. Se o valor para o argumento da acção for ACTION\_DELETE e transmitir zero como valor

para o argumento da opção, este método comporta-se de forma igual à do actual método de actualização, que tem a seguinte assinatura:

```
public void update(dkUserGroupDef userGroup, Vector v, int action)
throws DKNotExistException, DKException, Exception
```

Se o valor do argumento da acção for ACTION\_DELETE e o valor do argumento da opção for DK\_ICM\_DO\_NOT\_ROLLBACK\_ON\_ERROR, o novo método update não remove a transacção, mesmo que o utilizador a eliminar do grupo não pertença ao grupo. Mais uma vez, o comportamento predefinido (se o utilizador não pertencer ao grupo) é lançar uma excepção DKNotExistException.

Para obter mais informações, consulte a DKUserMgmtAPI na Application Programming Reference.



---

## Pesquisar dados

Quando uma aplicação pesquisa artigos armazenados no sistema do DB2 Content Manager, o motor subjacente processa a pesquisa com base numa consulta composta pela utilizador. Para começar a escrever consultas, o utilizador deve compreender os conceitos, sintaxe e gramática da linguagem de consulta.

Para examinar de forma eficiente o modelo de dados do DB2 Content Manager, crie consultas utilizando a linguagem de consulta do DB2 Content Manager. A linguagem de consulta faculta as seguintes vantagens:

- Suporta o modelo de dados completo.
- Suporta a pesquisa de uma determinada versão como, por exemplo, a versão mais actual.
- Permite pesquisas em hierarquias de vista de tipo de componente em artigos ligados e referências.
- Combina a pesquisa paramétrica e de texto.
- Faculta a ordenação de capacidade, com SORTBY.
- Aplica controlo de acessos do DB2 Content Manager.
- Está em conformidade com as regras do XQuery Path Expressions (XQPE), um subconjunto do projecto de trabalho W3C XML Query.
- Efectua pesquisas de alto rendimento.

A linguagem de consulta do Content Manager é uma linguagem com base em XML que, ao contrário das linguagens patenteadas, se adapta ao XQuery Path Expressions (XQPE), uma das bases do projecto de trabalho W3C XML Query. Com a linguagem de consulta, pode pesquisar tipos de artigos hierárquicos e localizar artigos de forma fácil e rápida.

Todas as consultas são realizadas em vistas de tipos de componente. Assim sendo, os nomes que o utilizador irá utilizar para os componentes raiz ou para os componentes descendentes nas cadeias de consulta podem ser os nomes de vistas de tipo de componente base que são criados pelo sistema (por exemplo, Journal, Journal\_Article) ou os nomes de vistas de tipo de componentes definidos pelo utilizador (por exemplo, My\_Journal, My\_Book\_Section). Note que no cliente de administração do sistema, as vistas de tipo de componentes são denominadas subconjuntos de tipo de componentes.

Quando o utilizador submete uma consulta de um documento, pasta, objecto, etc., o seu pedido é dirigido para o motor de consulta do DB2 Content Manager. O motor processa a consulta e converte-a no código SQL adequado. De seguida, o servidor de bibliotecas adiciona as verificações de segurança (ACLs) e executa a consulta.

**Importante:** Os termos consulta e pesquisa são utilizados alternadamente nesta secção e têm, essencialmente, o mesmo significado.

---

## Consultar o servidor de DB2 Content Manager

Pode construir consultas que efectuam os três principais tipos de pesquisa; paramétrica, de texto ou uma combinação de paramétrica e texto. Para executar uma consulta, pode utilizar um dos seguintes métodos: `evaluate`, `execute` ou `executeWithCallback`.

Uma consulta paramétrica utiliza condições como, por exemplo, igualdade e comparação. Uma consulta de texto utiliza as funções de pesquisa de texto que tornam a pesquisa mais precisa. Uma consulta combinada é composta por consultas de texto e paramétricas.

Ao consultar o servidor de bibliotecas do DB2 Content Manager, execute os seguintes passos principais:

1. Crie uma cadeia de consulta para representar as suas condições de pesquisa.
2. Invoque o método `evaluate`, `execute` ou `executeWithCallback` com a cadeia de consulta e opções de consulta para obter os resultados ou invoque o método `executeCount` para obter a contagem dos resultados.
3. Receba os resultados através de um objecto de `DKResults`, um `dkResultSetCursor` ou um objecto de `dkCallback`.

As APIs de consulta executam as tarefas de processamento de consultas como, por exemplo, preparar e executar uma consulta, supervisionar o estado da execução de uma consulta e obter os resultados.

O método `evaluate` devolve todos os resultados como uma recolha de DDOs. Nas opções predefinidas, este método funciona bem para conjuntos de resultados relativamente pequenos, 200 artigos ou menos. Ao trabalhar com conjuntos de resultados maiores, pode especificar a opção `IDONLY retrieve` para impedir que os dados de atributos sejam armazenados nos DDOs devolvidos. Nesse caso, tem de obter explicitamente os dados de atributos para os DDOs como dados necessários à aplicação. Note ainda que pode limitar o tamanho do conjunto de resultados com a opção `max results`.

O método `execute` devolve um objecto `dkResultSetCursor`, que possui as seguintes características:

- O `dkResultSetCursor` funciona como um cursor de servidor de conteúdos.
- Poderá utilizá-lo para grandes conjuntos de resultados porque os DDOs que devolve são obtidos em blocos, tal como o utilizador os solicitar.
- Pode utilizar o `dkResultSetCursor` para executar novamente uma consulta, invocando os métodos `close` e `open`.
- Pode utilizar o `dkResultSetCursor` para eliminar e actualizar a posição actual do cursor de conjunto de resultados.
- O `dkResultSetCursor` mantém aberto um cursor de uma base de dados para obter linhas da base de dados à medida que estas forem necessárias.
- Para evitar problemas de bloqueio e transacção, é utilizada uma nova ligação da base de dados para executar a consulta, o que significa que a aplicação pode utilizar duas ligações enquanto o cursor fica aberto. Para evitar esta ligação adicional, utilize o método `evaluate` para executar a consulta.

O método `executeWithCallback` executa uma consulta de modo assíncrono e envia em blocos os resultados para o objecto de repetição de marcação especificado. Assim, o método `executeWithCallback` liberta o módulo principal de execução da

tarefa de obtenção de resultados de consulta. Consulte a Application Programming Reference para obter uma descrição das opções de consulta no método `execute`.

Com o DB2 Content Manager Versão 8 Edição 3, pode obter a contagem dos resultados da consulta sem ter de obter os próprios resultados. Existem duas formas de proceder: Uma das formas é através do método `executeCount` no `DKDatastoreICM` e a outra é através do método `cardinality` no `dkResultSetCursor`. O método `executeCount` no `DKDatastoreICM` recebe uma cadeia e opções de consulta e devolve a estimativa de contagem dos resultados da consulta. O método `cardinality` no `dkResultSetCursor`, implementado no conector do DB2 Content Manager Versão 8, permite obter a contagem quando o utilizador já tem um `dkResultSetCursor`. Esta chamada precisa de uma deslocação separada para o servidor avaliar a contagem.

Note que a contagem dos resultados de consulta pode ser diferente do número de resultados que receberia ao executar a consulta e obter resultados. Isto porque a contagem só é exacta num momento no tempo. Podem ser adicionados ou eliminados artigos entre a hora da obtenção da contagem e da obtenção dos resultados. Em alguns casos a consulta pode devolver o ID de um artigo, mas a API de obtenção não tem acesso a este. Nestes casos, a contagem também será inexacta. Note ainda que para obter a contagem, tem de ser efectuada uma chamada ao servidor para avaliar a consulta. Significa isto que se quiser contagem e resultados, tem de fazer duas deslocações ao servidor.

Para obter mais informações relativas a métodos de consulta, consulte o exemplo `SSearchICM`. Consulte a Application Programming Reference para os métodos `evaluate`, `execute` ou `executeWithCallback` na classe `DKDatastoreICM`.

---

## Aplicar a linguagem de consulta ao modelo de dados do DB2 Content Manager

A visualização do servidor de bibliotecas como documento XML pode ajudar a compreender a linguagem de consulta.

Contudo, a analogia com o documento XML apenas é utilizada para fins de explicação da linguagem de consulta. Deste modo, é muito importante lembrar que a representação XML do servidor de bibliotecas é apenas uma representação virtual e os artigos num servidor de bibliotecas não são elementos XML, nem o utilizador obtém elementos XML quando executa uma consulta. Na representação XML do servidor de bibliotecas, os elementos do modelo de dados do DB2 Content Manager são representados da seguinte forma:

### Artigos

Em geral, cada artigo do CM é representado por elementos de XML imbricados, em que o elemento XML de nível de topo representa o componente raiz e os elementos XML imbricados representam os componentes descendentes. A imbricação de elementos de XML representa, assim, a hierarquia de componentes.

### Componentes raiz

Um componente raiz é representado pelo primeiro nível de um elemento XML. Um componente raiz possui os seguintes atributos de XML: `ID`, `ITEMID`, `STRING COMPONENTID`, `INTEGER VERSIONID`, `INTEGER SEMANTICTYPE` e qualquer outros atributos de componente definidos pelo utilizador. Em servidor de bibliotecas, o `IDITEM` é único.

Os componentes raiz podem conter um elemento ICMCHECKEDOUT para indicar o estado "de saída dada" do artigo. Tem dois atributos: ICMCHKOUTUSER, para indicar qual o utilizador que deu saída do artigo, e ICMCHKOUTTS, que é a marca de hora em que foi dada saída do artigo, por exemplo 2003-08-02-17.29.23.977001).

### **Componentes descendentes**

Um componente descendente é representado por um elemento de XML imbricado e possui os seguintes atributos: STRING ITEMID, STRING COMPONENTID, INTEGER VERSIONID e quaisquer outros atributos do componente definidos pelo utilizador.

Tenha em atenção que COMPONENTID isolado é apenas exclusivo num componente do DB2 Content Manager. O ITEMID e o VERSIONID são exactamente iguais ao ITEMID e VERSIONID do componente raiz do descendente.

### **Atributos definidos pelo utilizador**

Cada atributo definido pelo utilizador é representado por um atributo XML imbricado no elemento XML, que representa o componente que o contém.

### **Ligações**

Apesar das ligações de chegada e partida não fazerem parte de um artigo incluído no modelo de dados do CM, caso pretenda consultá-las, é conveniente considerá-las conceptualmente como fazendo parte do elemento XML que representa o artigo. Isto faz com que as aplicações não tenham de escrever explicitamente junções nas consultas. Pode utilizar ligações para modelar relações de muitos para muitos entre artigos. Tenha em atenção que esta relação só existe entre componentes raiz (uma ligação não pode originar nem terminar um componente descendente).

As ligações que dão origem a um artigo são representadas por elementos XML<OUTBOUNDLINK> com os seguintes atributos: IDREF LINKITEMREF, IDREF TARGETITEMREF e STRING LINKTYPE. O LINKITEMREF é uma referência a um artigo que contém metadados para a ligação. O TARGETITEMREF é uma referência ao artigo para o qual remete a ligação. O LINKTYPE é o tipo da ligação. De modo idêntico, as ligações que remetem para um artigo são representadas por elementos XML <INBOUNDLINK> com os seguintes atributos: IDREF LINKITEMREF, IDREF SOURCEITEMREF e STRING LINKTYPE. O SOURCEITEMREF é uma referência ao artigo em que a ligação é originada. Para obter mais informações relativas à semântica das ligações, consulte os exemplos SLinksICM e SSearchICM.

### **Referências (atributos de referência)**

Os atributos de referência são representados por atributos XML de tipo IDREF. Uma referência representa uma relação de um para um entre um artigo ou um componente e outro artigo. Consequentemente, o destino de uma referência apenas pode ser um componente raiz e não um componente descendente. Um atributo de referência, no entanto, pode ter origem tanto em componentes raiz como componentes descendentes.

Os atributos de referência podem ser definidos pelo sistema (SYSREFERENCEATTRS) ou definidos pelo utilizador (PublicationRef nas seguintes consultas exemplo). As referências podem ser cruzadas em ambas as direcções.

O cruzamento inverso de referências é executado de forma semelhante ao cruzamento inverso de ligações, tal como foi descrito acima. O utilizador pode conceber o artigo que está a ser referenciado como um elemento de XML denominado REFERENCEDBY, que contém um atributo XML denominado REFERENCER (de tipo IDREF), que indica o componente que faz referência a este artigo. É semelhante ao elemento INBOUNDLINK com o atributo SOURCEITEMREF para fins de cruzamento inverso de ligações.

As referências também suportam semânticas de eliminação (restrição, cascata, definição de nulo ou sem acção). Para obter mais informações relativas a atributos de referência, consulte os exemplos SReferenceAttrDefCreationICM e SSearchICM.

**Pastas** A funcionalidade de pasta facultada no Content Manager através de DKFolder é armazenada como um conjunto de ligações do tipo de ligação DKFolder(DK\_ICM\_LINKTYPENAME\_DKFOLDER). Todas as relações pasta-conteúdo são modeladas como ligações de partida da pasta e como ligações de chegada aos conteúdos, transformando a pasta na origem e os conteúdos nos destinos de cada ligação. Siga a semântica de pesquisa e de cruzamento de ligações para pesquisar e cruzar pastas. Para obter mais informações, consulte os exemplos SFolderICM e SSearchICM.

---

## Compreender a pesquisa paramétrica

Os artigos são normalmente obtidos através da iniciação de uma pesquisa de atributos seleccionados. Uma única consulta pode examinar atributos definidos pelo sistema e atributos definidos pelo utilizador dos artigos no servidor de conteúdos.

As condições de pesquisa simples são formadas por um nome de atributo, um operador e um valor que são combinados numa cláusula. O DB2 Content Manager facultava ao utilizador muitos operadores de comparação para concluir as pesquisas paramétricas. Os operadores incluem:

=  
<  
<=  
>  
>=  
!=  
LIKE  
NOT LIKE  
BETWEEN  
NOT BETWEEN  
IS NULL  
IS NOT NULL  
IN  
NOT IN

Pode simplificar condições de pesquisa complexas combinando condições de pesquisa simples numa cláusula através da utilização dos operadores booleanos AND, OR e NOT. Consulte os exemplos de consulta para obter mais detalhes.

---

## Compreender a pesquisa de texto

O DB2 Content Manager suporta dois tipos de pesquisa de texto: a pesquisa de texto de atributos que contém texto em componentes e a pesquisa de texto de objectos. A principal diferença entre os dois tipos de pesquisa de texto é a forma como o conteúdo é armazenado.

Quando define um atributo para que seja pesquisável por texto, está a indicar que se pode pesquisar texto contido na coluna desse atributo.

Por exemplo, Paulo, um administrador de sistemas, cria um tipo de artigo denominado Jornal com uma vista de tipo de componente descendente Artigo\_Jornal, que pretende activar para pesquisa de texto. Um dos atributos de Artigo\_Jornal é Título, que Paulo activa para pesquisa de texto. Quando Susana, uma subscritora, pesquisa Títulos que contém a palavra "Java", o sistema pesquisa no índice de texto Título ocorrências de "Java".

Para tornar o texto de um atributo (coluna) pesquisável, tem de criar índices de texto no conteúdo do tipo de artigo ou nos atributos específicos, utilizando o cliente de administração do sistema ou as APIs de administração do sistema.

O índice de texto contém informações relativas ao texto que será pesquisado. Estas informações são utilizadas para executar a pesquisa de texto de forma eficaz. O DB2 Content Manager utiliza a seguinte hierarquia de classes para representar estes índices de texto:

### **dkTextIndexICM**

Interface de classes que representa um índice de texto.

### **dkDB2TextIndexICM**

Interface de classes que expande o dkTextIndexICM e representa o índice de texto do IBM DB2 Universal Database Net Search Extender.

Para obter informações sobre DB2 Net Search Extender utilizado no DB2 Content Manager Versão 8.2 ou superior, consulte o DB2 Information Center.

**Importante:** O DB2 Content Manager Versão 8.4 suporta ainda funcionalidades da Versão 8, tais como a classe de definição DKTextIndexDefICM e os métodos que utilizam DKTextIndexDefICM de DKItemTypeDefICM e de DKAttrDefICM.

## Pesquisar conteúdo de objectos

Um utilizador final que execute uma pesquisa não nota diferenças ao pesquisar objectos armazenados num gestor de recursos, mas um administrador de sistema tem de executar um passo adicional para pesquisar conteúdo de objectos.

A pesquisa de conteúdo de objectos funciona de forma diferente. Em vez de indexar directamente uma coluna, o sistema utiliza uma referência à localização do objecto num gestor de recursos. O NSE utiliza a referência para obter o conteúdo quando cria um índice de texto. Um administrador de sistemas, no entanto, tem de configurar uma vista de tipo de artigo de recurso de texto, de forma a que o mecanismo de pesquisa localize o conteúdo no gestor de recursos. A pesquisa de texto é executada no atributo "TIREF" do tipo de artigo de recurso, que faz referência aos conteúdos armazenados no gestor de recursos para fins de pesquisa de texto.

## Pesquisar documentos

Para localizar documentos, pesquise os respectivos conteúdos de texto.

Uma vista "ICMPARTS" de tipo de componente virtual é suportada na consulta como descendente de todos os documentos do sistema. O atributo "TIEREF" na vista "ICMPARTS" de tipo de componente faz referência aos conteúdos de todas as partes pesquisáveis por texto desse documento, para fins de pesquisa de texto. Consulte os exemplos de consulta para obter a utilização específica desta funcionalidade.

## Tornar os atributos definidos pelo utilizador pesquisáveis por texto

Pode tornar os atributos definidos pelo utilizador pesquisáveis por texto utilizando as APIs DKAttrDefICM e DKItemTypeDefICM.

As propriedades predefinidas do índice de texto criado podem ser modificadas utilizando a interface dkDB2TextIndexICM. Pode ainda modificar as propriedades predefinidas do índice de texto criado utilizando a interface . Para obter mais informações relativas às APIs, consulte a Application Programming Reference ou o exemplo SItemTypeCreationICM.

## Compreender a sintaxe da pesquisa de texto

O utilizador pode executar consultas de pesquisa de texto utilizando uma sintaxe pesquisa de texto básica ou avançada.

### Pesquisa de texto básica

Uma vez que a maioria das pesquisas de texto são efectuadas através da listagem de algumas palavras seguidas, a sintaxe de pesquisa de texto básica (simplificada) foi concebida especificamente para simplificar este caso mais comum para os utilizadores.

A sintaxe de pesquisa básica permite a utilização de + e -, bem como a utilização de expressões entre aspas. A pesquisa básica é suportada no IBM DB2 UDB. .

A função "contains-text-basic" é utilizada para pesquisar dentro de atributos e dentro de conteúdos de recurso ou documentos. A função "score-basic" utiliza a mesma sintaxe da função "contains-text-basic" e é útil para ordenar resultados com base na ordenação dos resultados da pesquisa de texto. Tenha em atenção que a função "contains-text-basic" terá de corresponder a 1, caso seja true, e a 0, caso seja false. Consulte os exemplos de consulta para obter mais informações sobre como utilizar estas funções.

As informações adicionais relativas à sintaxe de pesquisa básica de texto incluem:

- Tem de utilizar a pesquisa avançada para efectuar pesquisas dependentes de maiúsculas e minúsculas.
- Parte-se do princípio que os termos entre aspas são uma expressão.
- Utilização de + (mais) - (menos)

**+ (mais)**

o documento tem de incluir esta palavra

**- (menos)**

o documento não pode incluir esta palavra



### Não especificado

Quando não é especificado um + ou -, o motor de consulta utiliza um algoritmo para fazer corresponder as palavras ao texto.

- Os operadores booleanos (AND, OR, NOT) não são válidos e são ignorados.
- Os parêntesis na sintaxe básica não são suportados.
- Caracteres globais válidos

### ? (ponto de interrogação)

representa um único carácter

### \* (asterisco)

representa qualquer número de caracteres arbitrários

Para obter mais informações relativas à pesquisa de texto básica, consulte o exemplo SSearchICM.

## Pesquisa de texto avançada

A sintaxe de pesquisa de texto avançada utiliza um motor de texto subjacente diferente do da pesquisa de texto básica, pelo que a sintaxe de texto é diferente. A pesquisa de texto avançada suporta funções como, por exemplo, pesquisa por proximidade e pesquisas imprecisas.

A sintaxe de pesquisa de texto avançada funções utiliza as funções específicas de base de dados de forma idêntica ao modo como as funções "contains-text-basic" e "score-basic" são utilizadas pela pesquisa de texto básica. A pesquisa avançada utiliza a sintaxe de pesquisa suportada pelo motor de texto subjacente (DB2 Net Search Extender para DB2 UDB ), pelo que a sintaxe de pesquisa é diferente.

### contains-text-db2 e score-db2

Utiliza a sintaxe de pesquisa de texto com um excepção: tem de substituir aspas por plicas e vice-versa. Por exemplo, a condição CONTAINS (description, ' "IBM" ') = 1 no NSE passa a ser contains-text-db2(@description, " 'IBM' ") = 1 na linguagem de consulta do DB2 Content Manager. Esta acção deve ser efectuada para suportar a simplicidade de escrita em consultas com uma utilização mínima de caracteres de abandono. Tenha em atenção que a função contains-text-db2 deve ser igual a 1 para verificar se é true e igual a 0 para verificar se é false. Consulte os exemplos de consulta para obter mais detalhes relativos à pesquisa de texto avançada.

Exemplo:

```
/Book[contains-text-db2(@Title, "ibm")=1] SORTBY(score-db2(@Title, "ibm"))
```

Para efeitos de retrocompatibilidade com o DB2 Content Manager Versão 8 Edição 2, as funções de pesquisa de texto avançadas "contains-text" e "score" continuam a ser suportadas. Estas funções são idênticas às funções contains-text-db2 e score-db2.

Para obter mais informações relativas à pesquisa de texto avançada, consulte o exemplo SSearchICM.

## Pesquisa de texto com a UDF ICMFetchFilter

A saída de utilizador, ICMFetchContentExit (ICMFetchContentParms1 \*IFP) está disponível para melhorar a pesquisa de texto.

Para obter mais informações, consulte a secção dcmcm118.dita#dcmcm118.dita

## Duração do tempo limite do ICMFetchFilter

A activação e definição da duração do tempo limite pode ajudar a evitar os bloqueios/paragens.

Durante a actualização do índice de texto em artigos, o motor NSE obtém o texto para indexação invocando a UDF do ICMFetchFilter. A UDF do ICMFetchFilter invoca o filtro de conversão INSO para converter dados binários (PDF, DOC, etc) em texto simples. Uma vez que estas invocações são efectuadas em sequência, o processo de actualização de índices do NSE pode ser bloqueado quando o INSO para.

O DB2 Content Manager Versão 8.4 faculta um quadrado de confirmação **Enable filter timeout (Activar o tempo limite do filtro)** no separador **Features (Funções)** da janela Library Server Configuration (Configuração do Servidor de Bibliotecas) para configurar a duração do tempo limite do ICMFetchFilter.

Se seleccionar o quadrado de confirmação **Enable filter timeout (Activar tempo limite do filtro)**, pode especificar o valor de tempo limite do filtro na lista **Filter timeout (Tempo limite do filtro)**. Se o quadrado de confirmação **Enable filter timeout (Activar o tempo limite do filtro)** não for seleccionado, a função de tempo limite não é activada.

Quando a conversão do INSO para, o DB2 Content Manager regista as informações sobre o documento que provocou a paragem do INSO na tabela de registo do NSE.

**Importante:** O tempo limite da UDF não se aplica às seguintes plataformas: iSeries e zSeries, nas quais o NSE não está disponível.

Os dois seguintes métodos são adicionados à classe DKLSCfgDefICM:

### getUDFTimeout

Devolve a duração do tempo limite que controla o período de tempo durante o qual o ICMFetchFilter aguarda pelo processo descendente de conversão de texto.

### setUDFTimeout

Modifica a duração do tempo limite que controla o período de tempo durante o qual a UDF do ICMFetchFilter aguarda pelo processo descendente de conversão de texto. Se o valor de tempo limite for positivo, a UDF do ICMFetchFilter aguarda pelo processo descendente de conversão de texto durante o período especificado antes de parar a conversão. Se o valor de tempo limite for zero, o método invoca a conversão de texto directamente na UDF. Um valor de tempo limite negativo resulta num erro.

Para obter mais informações, consulte a classe DKLSCfgDefICM no manual *Application Programming Reference*

## Indexar e pesquisar numa base de dados do servidor de bibliotecas Unicode DB2 UDB

Os utilizadores tiveram problemas com a forma como o DB2 Content Manager processava a indexação de texto de conteúdo de documentos em língua não inglesa ao utilizar uma base de dados do servidor de bibliotecas Unicode DB2 UDB.

Os problemas derivam da forma como o texto dos documentos foi extraído pela UDF do ICMFetchFilter. A UDF do ICMFetchFilter é utilizado para extrair texto de documentos em língua não inglesa, tais como Word ou PDF. Esse texto foi

devolvido na página de códigos do sistema operativo e não na página de códigos da base de dados. O texto tem de estar na página de códigos da base de dados para que seja interpretado correctamente pelo DB2 Net Search Extender durante a indexação. Se a página de códigos não corresponder, apenas as palavras que existem em ambas as páginas de códigos funcionarão. Esta situação poderá fazer parecer que apenas existem palavras em inglês.

Muitos dos problemas são corrigidos adicionando a variável de ambiente do servidor de bibliotecas **ICMCCSID**.

Ao definir a variável de ambiente **ICMCCSID**, já não é necessário que a página de códigos do sistema seja igual à página de códigos da base de dados. Um valor possível de **ICMCCSID** é 1208. A definição da variável de ambiente permite indexar documentos de vários idiomas para uma base de dados Unicode. Deste modo, pode agora utilizar uma base de dados do servidor de bibliotecas Unicode DB2 UDB para indexar e pesquisar o seguinte:

- Documentos que incluam conteúdo em vários idiomas
- Documentos em diferentes idiomas mas que pertençam ao mesmo tipo de artigo, utilizando um único índice de texto UTF-8

### Configurar o ambiente:

Se o ambiente não estiver configurado correctamente, não será possível pesquisar palavras em vários idiomas e a indexação de texto não funcionará correctamente.

Antes de prosseguir, deverá ter instalado o DB2 Content Manager 8.3 Fix Pack 2 ou posterior. Além disso, deverá certificar-se de que a base de dados do servidor de bibliotecas é Unicode (página de códigos 1208). Ao instalar o DB2 Content Manager, foi apresentada a opção de selecção de uma base de dados Unicode. Pode ainda efectuar a verificação através da ligação à base de dados e da emissão do seguinte comando: `db2 get db config`. O comando apresenta a totalidade da configuração da base de dados. Procure a linha que contém Database code page e verifique se o valor corresponde a 1208.

As informações que se seguem incluem as tarefas principais:

- Definir o **CCSID** como 1208 no servidor de bibliotecas
  - Criar novamente índices de pesquisa
  - Certificar-se de que os documentos de texto estão codificados em UTF-8
1. Defina o **CCSID** como 1208 no servidor de bibliotecas. Para incluir conteúdo de documentos num índice de texto numa base de dados Unicode, é necessário utilizar `ICMFetchFilter` para obter conteúdo de documentos. Além disso, o utilizador tem de se certificar que todos os índices de texto são criados com o **CCSID** definido 1208. A variável de ambiente **ICMCCSID** tem igualmente de ser definida como 1208 no ambiente do servidor de bibliotecas. Para tal, execute os seguintes passos:

**UNIX** Exporte a variável de ambiente **ICMCCSID** e adicione-a ao **DB2ENVLIST** executando os seguintes passos:

- a. No ficheiro `/home/db2inst1/sqllib/userprofile`, certifique-se de que as seguintes linhas estão incluídas na ordem apresentada:  

```
ICMCCSID=1208
export ICMCCSID
```
- b. No ficheiro `/home/db2inst1/sqllib/profile.env`, certifique-se de que as seguintes linhas estão incluídas e que o **ICMCCSID** é adicionado ao **DB2ENVLIST**:

```
DB2LIBPATH=/usr/lpp/icm/lib
DB2ENVLIST='LIBPATH ICMROOT ICDLL ICMCOMP EXTSHM
CMCOMMON DB2LIBPATH ICMCCSID'
```

c. Recicle a instância de DB2 UDB introduzindo

```
db2stop force
db2start
```

**Windows**

Num sistema Windows DB2 Content Manager, defina o **ICMCCSID** como uma variável de ambiente do sistema e, em seguida, pare e reinicie o DB2 UDB.

2. Crie novamente os índices de pesquisa. Se tiver tipos de artigo com índices de texto já existentes numa base de dados do servidor de bibliotecas Unicode, é necessário remover os índices de texto e voltar a criá-los com o novo CCSID. Para criar novamente os índices, consulte o script `ManuallyDropDb2CMTextindexes.SQL` no final deste tópico. Terá de consultar os comentários do script e personalizá-lo de acordo com o seu ambiente antes de o utilizar para criar novamente índices de texto. Importante: Este procedimento modifica directamente a base de dados do servidor de bibliotecas. Como tal, deverá efectuar uma cópia de segurança da base de dados e dos índices de texto antes de tentar remover manualmente e voltar a criar os índices de texto.
3. Certifique-se de que os documentos de texto estão codificados em UTF-8. Certifique-se de que todos os documentos de texto simples estão codificados em UTF-8 ao armazená-los. É necessário indicar manualmente que estes documentos estão codificados em UTF-8. Caso contrário, a pesquisa de texto funcionará perfeitamente, mas os utilizadores finais poderão não conseguir visualizar documentos de texto simples que não tenham este campo definido. Para obter mais informações sobre como definir o campo de codificação para o documento, consulte os tópicos `Language and character encoding for imported text documents in eClient` e `Specify language and character encoding information when you import plain text documents`.

Para garantir que os documentos de texto simples são correctamente indexados, é necessário converter também documentos de texto simples que se encontrem na página de códigos local para UTF-8.

Quando todas as tarefas acima indicadas estiverem concluídas, a indexação de texto de conteúdo de documentos irá funcionar correctamente. O texto será obtido em Unicode e, em seguida, convertido para o formato correcto requerido pelo DB2 Net Search Extender. Isto permite a indexação de texto de documento que contém texto em vários idiomas, uma vez que o texto é processado em Unicode.

Para índices de texto novos, a pesquisa pode ser executada em palavras de diferentes idiomas, de acordo com as limitações abaixo apresentadas. Para índices de texto existentes, o índice tem de ser removido e criado novamente, conforme descrito anteriormente. Se forem adicionados documentos de texto simples ao mesmo tipo de nó, não existirá conversão de página de códigos. A codificação dos documentos de texto simples tem de ser UTF-8 (página de códigos 1208) para que sejam incluídos correctamente no índice de texto. Muitos editores permitem que os ficheiros de texto sejam guardados com esta codificação.

Encontra-se disponível suporte para pesquisa de documentos em vários idiomas para bases de dados de servidor de bibliotecas que não estejam em formato Unicode. Para bases de dados em formatos diferentes de Unicode, a pesquisa de texto está limitada à pesquisa de caracteres que se encontrem na página de códigos

do servidor de bibliotecas. A conversão de uma base de dados de servidor de bibliotecas não Unicode para uma base de dados de servidor de bibliotecas Unicode não é suportada pelo DB2 Content Manager, embora seja possível através de um ajuste de serviços.

A indexação e pesquisa de texto de vários idiomas para uma base de dados Unicode é suportada através de ICMFetchFilter. Além disso, é suportada quando um atributo de texto é tornado pesquisável por texto (CHAR, VARCHAR, CLOB). Esta situação não é suportada em índices de texto de atributos que utilizem ICMFilter (atributos BLOB). A API de Java do DB2 Content Manager suporta totalmente a pesquisa de palavras em vários idiomas. Uma vez que o Java representa cadeias em Unicode, os caracteres Unicode incluídos como parte de uma cadeia de pesquisa em Java irão fluir para o gestor de conteúdos sem problemas.

A API de C++ não suporta Unicode completo, embora possa ser utilizada para aceder a um servidor de bibliotecas Unicode. A API de C++ utiliza uma única página de códigos, que não é Unicode. Deste modo, apenas pode efectuar pesquisas utilizando palavras que possam ser representadas na página de códigos que está a utilizar.

Para obter uma lista dos formatos suportados pelo Stellent, consulte o seguinte endereço da Web: [http://www.stellent.com/stellent3/groups/mkt/documents/nativepage/outside\\_in\\_supported\\_platforms.pdf](http://www.stellent.com/stellent3/groups/mkt/documents/nativepage/outside_in_supported_platforms.pdf).

O problema abordado aplica-se apenas aos sistemas DB2 Content Manager 8 que utilizem DB2 UDB. Tenha em atenção que o MULTI-LEXER não pode ser utilizado com tipos de artigo de documento. Além disso, a utilização do BASIC-LEXER não faculta a mesma variedade de suporte de idiomas disponível no servidor de bibliotecas no DB2 UDB com APAR IO02595.

### **Script ManuallyDropDb2CMTextindexes.SQL:**

Existem 6 passos que devem ser executados com êxito para que seja possível criar novamente índices de pesquisa.

```
-- This procedure manually drops an item type text index from CM 8. However,
-- only document item types can recreate the text index, once it is dropped.
-- For resource item types, once the text index is dropped, it can not be
-- recreated using the admin client.
--
-- This procedure directly modifies the library server database. Therefore, you
-- should back up your database and text indexes before you attempt to manually
-- drop and recreate the text indexes.
--
-- In this script, all of the updates and deletes are commented out.
-- You must customize and run this script and look at the output of the queries
-- and then come back and uncomment and customize updates and deletes.
--
-- Connect to the library server database
-- Change this command to fill in your database name, userid, and password
CONNECT TO ICMNLSDB USER icmadmin using password;

-- Step 1:
-- This query selects the item type name and index names for the item type.
-- The example below finds the index names for the item type UniText.
-- To find all item type indexes in the system, remove the last AND condition from
-- the query below. Then query will show text indexes for all item
-- types. Note that the item type index names always end with 'TIE'.
-- If a schema other than 'icmadmin' is used, this must be customized below
select a.keywordname AS CompTypeName, d.indexname AS IndexName, b.itemtypeid
```

```

from icmadmin.icmstnlkeywords a,
 icmadmin.icmstitemtypedefs b,
 icmadmin.icmstcompdefs c,
 icmadmin.icmsttextindexconf d
where a.keywordclass = 2 and
 a.keywordcode = b.itemtypeid and
 b.itemtypeid >= 1000 and
 c.itemtypeid = b.itemtypeid and
 c.componenttypeid = d.componenttypeid AND
 d.indexName LIKE '%TIE'
-- Comment out this line to see all text indexes
 AND a.keywordname = 'Forum'
;

-- Step 2:
-- Manually drop the text index. Use the index name from the query above.
-- Just as in the connect, you must customize database name, userid, and password
-- db2text drop index icmadmin.ICMUU01033001TIE for text connect to ICMNLSDB
-- user icmadmin using password

-- Step 3:
-- Now delete the row from the CM table that holds the configuration of the
-- the text index. This is needed to allow the index to be recreated with a
-- new configuration. Note that this deletes all configuration for this index.
-- You will need to re-enter the configuration information when you recreate
-- the index. You may want to look at the index in the System Admin GUI to
-- see the configuration, before doing this step.
-- Here you must set the index name to the name selected in the first query.
-- DELETE FROM icmadmin.icmsttextindexconf WHERE indexname='ICMUU01033001TIE';

-- Step 4:
-- Now we need to tell CM this item type is no longer searchable. To do this
-- we must set a flag in the component type definition of the attribute which
-- has the text index. For item type text indexes, the attribute
-- is called TIEREF. To see the current value of the flag, do the following
-- query. You must set the component type that contains the index and the name of
-- of the attribute that contains the text index in the sample shown below.
-- For resource item types this is simply the name of the item type. For document
-- item types this name must be derived from the ItemTypeId that was returned
-- from the first query above. The name of the comp type for a document item
-- type is ICMPart + the item type id. This is ICMParts1009 in the example
-- shown below.
SELECT a.keywordname AS CompTypeName, ca.ATTRFLAGS, kw.keywordname AS
 AttributeName, ca.ComponentTypeId, ca.AttributeId, ca.attributeGroup
 FROM icmadmin.ICMSTCompAttrs ca,
 icmadmin.icmstnlkeywords a,
 icmadmin.icmstnlkeywords kw
 WHERE
 a.keywordclass = 5 AND
 a.keywordcode = ca.componenttypeid AND
 -- To list data for all indexes, comment on the following line
 a.keywordname = 'ICMParts1011' AND
 kw.keywordclass = 1 AND
 kw.keywordcode = ca.AttributeId AND
 kw.keywordname = 'TIEREF';
;

-- Step 5:
-- Tell CM this item type is no longer text searchable. To do this you must
-- set the attr flags for the TIEREF attribute to 3 in the database.
-- You must customize the query below to set the component type id from the
-- previous query.
-- UPDATE icmadmin.ICMSTCompAttrs ca
-- SET ca.ATTRFLAGS = 3
-- WHERE ca.componenttypeid = 1033 AND
-- ca.attributeid = 20 AND
-- ca.attributeGroup = 7

```



```
--;

-- Step 6:
-- Now commit all the changes
COMMIT WORK;

-- Disconnect from the database
CONNECT RESET;

-- Now that all this is done you can go into the System Admin GUI and recreate
-- the text index for this item type.
```

---

## Criar uma pesquisa combinada de texto e parâmetros

Existem 3 tipos de pesquisas que podem ser executados em, virtualmente, qualquer parte de um artigo ou componente, texto dentro de um artigo ou componente ou texto dentro de conteúdo de recursos.

Uma pesquisa pode ser de um dos seguintes tipos:

### **Pesquisa paramétrica**

Uma pesquisa que é baseada nas propriedades do artigo e do componente, atributos, referências, ligações, conteúdos de pastas, etc.

### **Pesquisa de Texto**

Pesquisar no texto.

### **Pesquisa Combinada**

Uma pesquisa utilizando a pesquisa paramétrica e de texto.

## Criar uma pesquisa combinada de texto e parâmetros: Java

A combinação de pesquisas de texto e parâmetros em Java implica 5 passos.

1. Criar um arquivo de dados de ICM e estabelecer ligação ao mesmo.
2. Gerar uma cadeia de consulta combinada.

```
String queryString =
 "///Journal_Article [Journal_Author/@LastName = \"Richardt\" +
 \" AND contains-text (@Text, \" 'Java' & 'XML' \")=1]\";
```

3. Utilize a matriz DKNVPair para agrupar as opções. Especifique sempre as opções de obtenção (entre quaisquer outras opções de consulta) para pedir os dados que pretende utilizar e activar optimizações.

```
DKNVPair parms[] = new DKNVPair[3];
String strMax = "5";
parms[0] = new DKNVPair(DKConstant.DK_CM_PARM_MAX_RESULTS, 5);
DKRetrieveOptionsICM dkRetrieveOptions =
 DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOptions.baseAttributes(true);
parms[1] = new DKNVPair(DKConstant.DK_CM_PARM_RETRIEVE, dkRetrieveOptions);
parms[2] = new DKNVPair(DKConstant.DK_CM_PARM_END, null);
```

4. Execute a consulta combinada através de uma de três formas diferentes: evaluate, execute e executeWithCallback.

```
DKResults resultsCollection =
 (DKResults)dsICM.evaluate(queryString,
 DKConstant.DK_CM_XQPE_QL_TYPE,parms);
```

5. Processar os resultados. O procedimento de processamento dos resultados depende do método de execução que utilizou.

Para obter um exemplo completo e documentação adicional, consulte o exemplo formativo SSearchICMAPI em *IBMCMROOT/samples/java/icm*.



## Criar uma pesquisa combinada de texto e parâmetros: C++

A combinação de pesquisas de texto e parâmetros em C++ implica 5 passos.

1. Especifique as opções de consulta. Especifique sempre as opções de obtenção para pedir os dados que pretende utilizar e activar optimizações. Em C++, a matriz de opções `DKNVPair[]` tem sempre de ter um elemento `END` especial para que a operação de consulta detecte o final. Por exemplo, se pretender especificar duas opções, a matriz de opções tem de ter três elementos nos quais o último elemento é o elemento `END` especial.

```
DKNVPair * parms = new DKNVPair[3];
//Allow a maximum of 5 items to be returned from the search
pparm[0].(DK_CM_PARM_MAX_RESULTS, DKString("5"));
//Specify what content is to be retrieved
DKRetrieveOptionsICM* dkRetrieveOptions =
 DKRetrieveOptionsICM::createInstance(dsICM);
dkRetrieveOptions->baseAttributes(TRUE);
pparm[1].set(DK_CM_PARM_RETRIEVE, dkRetrieveOptions);
pparm[2].set(DK_CM_PARM_END, (long) 0);
```

2. Executar a pesquisa. Existem três modos para executar uma pesquisa:

### **avaliar**

Devolve todos os resultados como uma recolha; é bom para conjuntos pequenos.

### **executar**

Devolve um cursor de conjunto de resultados, que o programa de chamada utiliza para executar uma iteração sobre os resultados.

### **executeWithCallback**

Cria um módulo que executa uma iteração sobre o conjunto de resultados e invoca o objecto de repetição de marcação para cada bloco de resultados. O programa de chamada utiliza o objecto de repetição de marcação para obter os resultados.

No exemplo que se segue, apenas são necessários cinco resultados. Deste modo, é utilizando o método `DKDatastoreICM.evaluate`.

```
DKResults * resultsCollection = (DKResults *) (dkCollection *)
 dsICM->evaluate(queryString, DK_CM_XQPE_QL_TYPE, parms);
```

3. Apresentar os resultados da pesquisa.

```
// Create an iterator to go through Results collection.
dkIterator* iter = resultsCollection->createIterator();

cout << "Results:" << endl;
cout << " - Total: " << results->cardinality() << endl;

while(iter->more())
{
 //Each element in the returned array is an item (DDO)
 DKDDO* ddo = (DKDDO*) iter->next()->value();
 cout << " - Item ID: " << ((DKPidICM*) ddo->getPidObject())
 << endl;
 cout << " - Item ID: " << ((DKPidICM*) ddo->getPidObject())
 << endl;
 cout << " - Item ID: " << ((DKPidICM*) ddo->getPidObject())
 << endl;
 cout << " - Item ID: " << ((DKPidICM*) ddo->getPidObject())
 << endl;
}

delete(iter);
delete[] parms;
....
```

4. Limpar.

Para obter um exemplo completo e documentação adicional, consulte o exemplo formativo `SSearchICMAPI` em `IBMCMROOT/samples/cpp/icm`.

---

## Pesquisas exemplo com linguagem de consulta

Para começar a escrever consultas, o utilizador deve compreender os conceitos, sintaxe e gramática da linguagem de consulta.

O modelo de dados exemplo fornecido é uma representação de um documento XML do modelo de dados, incluindo consultas exemplo, ajudá-lo-ão a compreender melhor a linguagem de consulta e a iniciar a escrita de consultas.

As consultas exemplo nas secções que se seguem baseiam-se no modelo de dados de exemplos de consulta descritos na Figura 11 na página 197. Consulte a ilustração do modelo de dados quando revir as consultas exemplo.

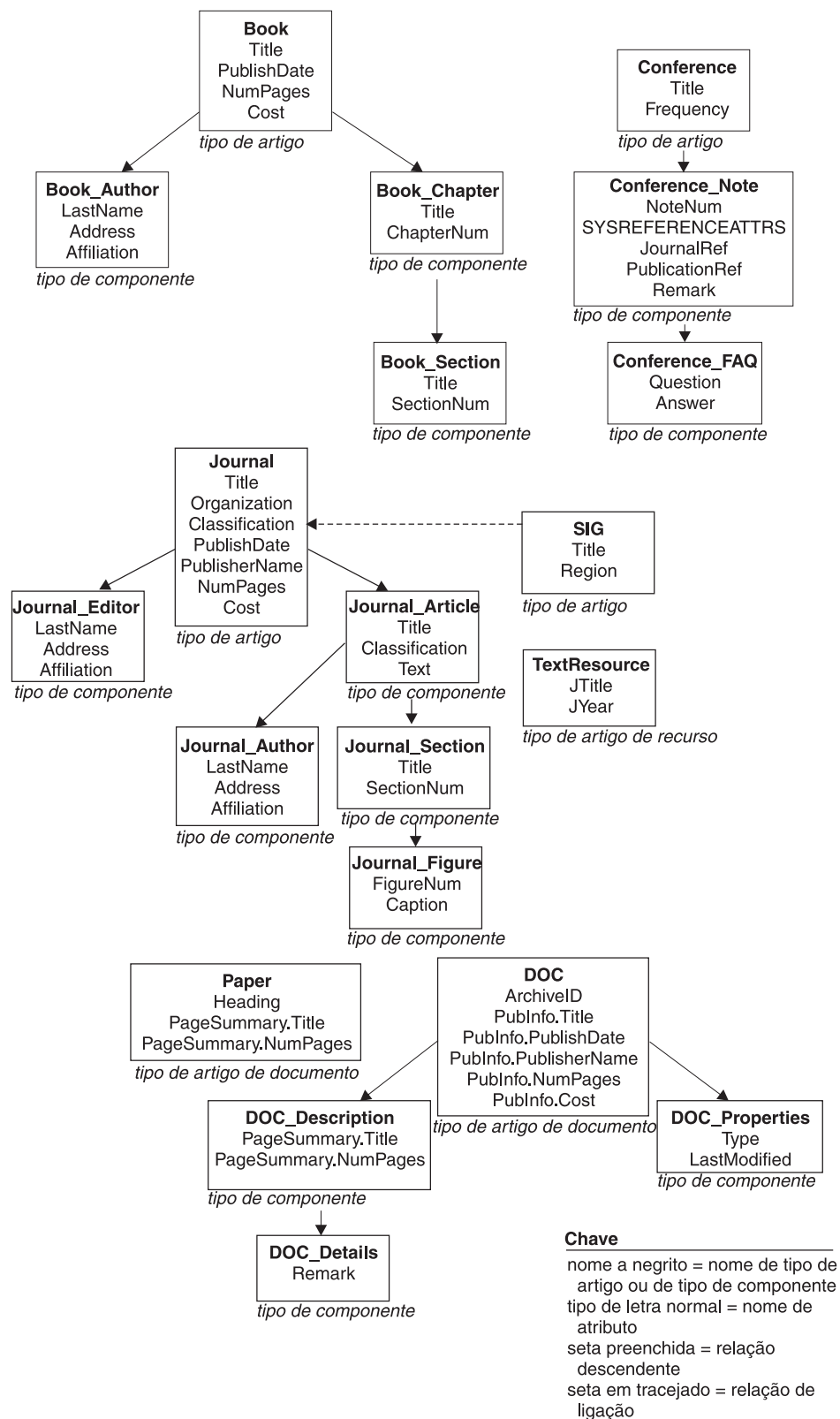


Figura 11. Modelo de dados de exemplos de consulta

Para obter sintaxe de consulta e exemplos adicionais com base num modelo de dados alternativo, consulte o exemplo SSearchICM.

O documento de XML seguinte é uma representação do modelo de dados da Figura 11 na página 197.

Representação XML do modelo de dados de exemplos de consulta:

```
<Journal (ID ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 INTEGER SEMANTICTYPE, Title, Organization, Classification,
 PublishDate, PublisherName, NumPages, Cost)>
 <Journal_Editor (STRING ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 LastName, Address, Affiliation)>
 </Journal_Editor>
 ... (repeating <Journal_Editor>)

 <Journal_Article (STRING ITEMID, STRING COMPONENTID,
 INTEGER VERSIONID, Title, Classification, Text)>
 <Journal_Section (STRING ITEMID, STRING COMPONENTID,
 INTEGER VERSIONID, Title, SectionNum)>
 <Journal_Figure (STRING ITEMID, STRING COMPONENTID,
 INTEGER VERSIONID, FigureNum, Caption)>
 </Journal_Figure>
 ... (repeating <Journal_Figure>)
 </Journal_Section>
 ... (repeating <Journal_Section>)

 <Journal_Author (STRING ITEMID, STRING COMPONENTID,
 INTEGER VERSIONID, LastName, Address, Affiliation)>
 </Journal_Author>
 ... (repeating <Journal_Author>)
</Journal_Article>
... (repeating <Journal_Article>)

<OUTBOUNDLINK (IDREF LINKITEMREF, IDREF TARGETITEMREF,
 STRING LINKTYPE) >
</OUTBOUNDLINK>
... (repeating <OUTBOUNDLINK>)

<INBOUNDLINK (IDREF LINKITEMREF, IDREF SOURCEITEMREF,
 STRING LINKTYPE) >
</INBOUNDLINK>
... (repeating <INBOUNDLINK>)

<REFERENCEDBY (IDREF REFERENCER)>
</REFERENCEDBY>
... (repeating <REFERENCEDBY>)
<ICMCHECKEDOUT (STRING ICMCHKOUTUSER, TIMESTAMP ICMCHKOUTTS)
</ICMCHECKEDOUT>
... (repeating <ICMCHECKEDOUT>)

</Journal>
... (repeating <Journal>)

<Book (ID ITEMID, STRING COMPONENTID, INTEGER VERSIONID, INTEGER
 SEMANTICTYPE, Title, PublishDate, NumPages, Cost)>
 <Book_Author (STRING ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 LastName, Address, Affiliation)>
 </Book_Author>
 ... (repeating <Book_Author>)

 <Book_Chapter (STRING ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 Title, ChapterNum)>
 <Book_Section (STRING ITEMID, STRING COMPONENTID,
 INTEGER VERSIONID, Title, SectionNum)>
 </Book_Section>
 ... (repeating <Book_Section>)
 </Book_Chapter>
 ... (repeating <Book_Chapter>)
```

```

 <OUTBOUNDLINK (IDREF LINKITEMREF, IDREF TARGETITEMREF,
 STRING LINKTYPE) >
 </OUTBOUNDLINK>
 ... (repeating <OUTBOUNDLINK>)

 <INBOUNDLINK (IDREF LINKITEMREF, IDREF SOURCEITEMREF,
 STRING LINKTYPE) >
 </INBOUNDLINK>
 ... (repeating <INBOUNDLINK>)

 <REFERENCEDBY (IDREF REFERENCER)>
 </REFERENCEDBY>
 ... (repeating <REFERENCEDBY>)
</Book>
... (repeating <Book>)

<SIG (ID ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 INTEGER SEMANTICTYPE, Title, Region)>
 <OUTBOUNDLINK (IDREF LINKITEMREF, IDREF TARGETITEMREF,
 STRING LINKTYPE) >
 </OUTBOUNDLINK>
 ... (repeating <OUTBOUNDLINK>)

 <INBOUNDLINK (IDREF LINKITEMREF, IDREF SOURCEITEMREF,
 STRING LINKTYPE) >
 </INBOUNDLINK>
 ... (repeating <INBOUNDLINK>)

 <REFERENCEDBY (IDREF REFERENCER)>
 </REFERENCEDBY>
 ... (repeating <REFERENCEDBY>)

 <ICMCKECKEDOUT (STRING ICMCHKOUTUSER, TIMESTAMP ICMCHKOUTTS)
 </ICMCKECKEDOUT>
 ... (repeating <ICMCKECKEDOUT>)

</SIG>
... (repeating <SIG>)

<TextResource (ID ITEMID, STRING COMPONENTID, INTEGER VERSIONID,
 INTEGER SEMANTICTYPE, JTitle, JYear)>
 <OUTBOUNDLINK (IDREF LINKITEMREF, IDREF TARGETITEMREF,
 STRING LINKTYPE) >
 </OUTBOUNDLINK>
 ... (repeating <OUTBOUNDLINK>)

 <INBOUNDLINK (IDREF LINKITEMREF, IDREF SOURCEITEMREF,
 STRING LINKTYPE)>
 </INBOUNDLINK>
 ... (repeating <INBOUNDLINK>)

 <REFERENCEDBY (IDREF REFERENCER)>
 </REFERENCEDBY>
 ... (repeating <REFERENCEDBY>)
</TextResource>
... (repeating <TextResource>)

```

Os resultados intermédios obtidos por INTERSECT/EXCEPT não podem ser combinados com operadores aritméticos (monádicos/binários) ou de comparação. Podem ser combinados por operadores de conjunto (UNION/INTERSECT/EXCEPT) ou apresentados isoladamente.

## Exemplos de utilização válida

Exemplos de utilização válida de UNION/INTERSECT/EXCEPT:

```
(//Journal/Journal_Article[@Title = "Content Management"] EXCEPT
//Journal_Article[@Classification = "Security"])/Journal_Section
```

Esta consulta é válida, uma vez que o resultado de EXCEPT é o resultado de toda a consulta - não é combinada utilizando operadores.

```
/Journal[(Journal_Editor/@LastName UNION .//Journal_Author/@LastName) =
"Davis"]
```

Esta consulta é válida, uma vez que não existe restrição sobre o operador UNION.

```
/Journal[Journal_Article[Journal_Section/@Title INTERSECT
./Journal_Figure/@Caption]/@Title = "Content Management"]
```

Esta consulta é válida, uma vez que o resultado do operador INTERSECT é combinado através e um operador de conjunto (UNION).

## Exemplos de utilização não válida

Exemplos de utilização não válida de INTERSECT/EXCEPT:

```
/Journal[(Journal_Editor/@LastName INTERSECT .//Journal_Author/@LastName) =
"Davis"]
```

Esta consulta não é válida, uma vez que o resultado do operador INTERSECT é combinado através de um operador de comparação (=).

```
/Journal[(./Journal_Section/@SectionNum EXCEPT .//Journal_Figure/
@FigureNum) + 5 = 10]
```

Esta consulta não é válida, uma vez que o resultado de EXCEPT é combinado utilizando um operador aritmético (+).

## Exemplos de consultas

Para começar a escrever consultas, o utilizador deve compreender os conceitos, sintaxe e gramática da linguagem de consulta. Estas consultas exemplo abrangem tipos de artigos, pesquisas de texto, operações aritméticas em condições, cruzamento de ligações e referências, artigos com saída dada e consulta por marca de hora.

Estas são algumas indicações para ajudar o utilizador a compreender os exemplos de consulta:

- Siga a cadeia da consulta como seguiria uma estrutura de directórios
- / (barra única) indica uma relação de descendência directa
- // (barra dupla) indica uma relação de descendência directa ou uma relação descendente
- . (PONTO) representa o componente actual da hierarquia
- .. (PONTO-PONTO) representa o ascendente do componente actual
- @ (sinal AT) indica um atributo
- [ ] (parêntesis rectos) indicam uma instrução condicional ou uma lista
- => (operador DEREFERENCE) representa uma acção de ligação ou referência
- O resultado da consulta deve ser um componente (por exemplo, um atributo não poderá ser o último elemento do caminho)

São facultados exemplos e documentação adicionais no exemplo SSearchICM .

**Importante:** Ao especificar atributos como condições de consulta, utilize o nome de atributo de 15 caracteres não traduzido. Não utilize o nome de apresentação para as condições de consulta.

Para obter mais informações sobre nomes de atributo e nomes de apresentação, consulte os exemplos educativos da API SAttributeDefinitionCreationICM e SSearchICM.

As consultas exemplo facultadas nesta secção têm por base o modelo de dados exemplo, Figura 11 na página 197, e o documento XML exemplo.

### Exemplo: Acesso a componentes

Esta consulta localiza todos os jornais.

```
/Journal
```

O carácter / é iniciado na raiz implícita do documento XML, que, neste caso, é a totalidade do servidor de bibliotecas. Cada tipo de artigo é um elemento sob esta raiz. Se SB.xml for o documento de XML que contém o modelo completo, tal como foi descrito anteriormente, a raiz de documento explícita será o documento (BS.xml).

### Exemplo: Acesso a atributos

Esta consulta localiza todos os artigos de jornal com um total de 50 páginas cada.

```
/Journal[@NumPages=50]
```

O predicado @NumPages = 50 é avaliado como true para todos os diários que possuem o atributo NumPages definido como 50.

### Exemplo: Vários tipos de artigo

Esta consulta localiza todos os livros ou jornais que possuem Williams como um dos autores e que possuem um título de secção começado por XML.

```
(/Book | /Journal)
[(./Journal_Author/@LastName = "Williams"
OR ./Book_Author/@LastName = "Williams")
AND (./Book_Section/@Title LIKE "XML%"
OR ./Journal_Section/@Title LIKE "XML%")]
```

OR

```
(/Book[./Book_Author/@LastName = "Williams"
AND ./Book_Section/@Title LIKE "XML%"])
| (/Journal[./Journal_Author/@LastName = "Williams"
AND ./Journal_Section/@Title LIKE "XML%"])
```

As duas consultas acima descritas produzem o mesmo resultado.

./Journal\_Author significa que um componente Journal\_Author deverá ser localizado directamente no componente actual no caminho (que, no primeiro caso, é um Book ou um Journal) ou mais abaixo na hierarquia. Tenha em atenção que o operador LIKE é utilizado em conjunção com um carácter global, neste caso %.

### Exemplo: Operações aritméticas em condições

Esta consulta localiza todos os jornais com um número de páginas entre 45 e 200.

```
/Journal[@NumPages BETWEEN 45-4 AND 2*100]
```



Tenha em atenção que pode executar operações aritméticas para calcular os valores resultantes que serão utilizados com o operador BETWEEN.

### Exemplo: Cruzamento de ligações no sentido progressivo

Esta consulta localiza todos os artigos de diários editados por Williams que estão incluídos em SIGs com o título SIGMOD.

```
/SIG[@Title = "SIGMOD"]/OUTBOUNDLINK
[@LINKTYPE = "contains"]/@TARGETITEMREF =>
Journal[Journal_Editor/@LastName = "Williams"]
/Journal_Article
```

Este é um exemplo de seguimento de ligações no sentido progressivo. O componente XML virtual OUTBOUNDLINK e o respectivo atributo TARGETITEMREF são utilizados para efectuar o cruzamento para todos os Journals e, finalmente, os Journal\_Articles subjacentes. O último componente do caminho consiste no que é devolvido como resultado da consulta. O resultado pode ser forçado cruzando apenas tipos de ligação específicos (contains, neste exemplo) com um tipo específico de artigos (Journal, neste exemplo). Como a representação conceptual de XML do servidor de bibliotecas considera as ligações de chegada e de partida como sendo partes de artigos, o operador de anulação de referência pode ser utilizado para que as aplicações não tenham de efectuar a escrita de junções explícitas.

### Exemplo: Cruzamento de ligações no sentido regressivo

Esta consulta localiza todos os artigos de qualquer tipo que possuam jornais que custem menos de cinco dólares, com artigos cujo autor seja Nelson.

```
/Journal[@Cost < 5
AND ../Journal_Author/@LastName = "Nelson"]
/INBOUNDLINK[@LINKTYPE = "contains"]
/@SOURCEITEMREF => *
```

Este é um exemplo de seguimento de ligações no sentido regressivo. O carácter global \*, que se segue ao operador de anulação de referência => garante que os artigos de tipo ANY são devolvidos como resultado.

### Pesquisa de texto avançada (funções contains-text e score)

Esta consulta localiza artigos de artigo cujo autor é Richardt e que contêm o texto Java e o texto XML. Os resultados são ordenados pelo resultado da pesquisa de texto. Este é um exemplo de execução de uma pesquisa de texto com a função contains-text-db2. Para obter a sintaxe suportada por esta função, consulte o DB2 Information Center.

**Importante:** A função contains-text-db2 deverá ser igual a 1 para ser true e a 0 para ser false. A função score utiliza as informações de ordenação devolvidas pelo DB2 UDB Net Search Extender, que é utilizando, neste caso, para ordenar os artigos de diário resultantes através de SORTBY.

### Exemplo: DB2

```
//Journal_Article[Journal_Author/@LastName = "Richardt"
AND contains-text-db2(@Text, " 'Java' & 'XML' ")=1]
SORTBY(score(@Text, " 'Java' & 'XML' "))
```

## Pesquisa de texto (função contains-text e ordenação de atributos)

Esta consulta localiza todos os jornais que possuem a palavra Design ou a palavra Index nos títulos e ordena os resultados por título, em ordem decrescente. Este é outro exemplo de execução de uma pesquisa de texto através da função contains-text. Neste caso, a ordenação utiliza o operador DESCENDING no atributo Title. O valor predefinido para SORTBY é ASCENDING.

### Exemplo: DB2

```
/Journal
[Journal_Article[contains-text-db2(@Title, " 'Design' |
 'Index' ")=1]]
SORTBY (@Title DESCENDING)
```

## Exemplo: Pesquisa de texto (funções contains-text-basic e score-basic)

Esta consulta localiza todos os artigos de jornal que contêm o texto Java e o texto JDK 1.3, mas não contêm o texto XML, utilizando a sintaxe de pesquisa de texto simplificada (básica) e ordena os resultados segundo o resultado da pesquisa de texto.

```
//Journal_Article
[contains-text-basic(@Title, " +Java -XML +'JDK 1.3'")=1]
SORTBY (score-basic(@Title, " +Java -XML +'JDK 1.3' "))
```

Este é um exemplo de execução de pesquisa de texto através de uma sintaxe de pesquisa de texto simplificada. Utilize um + para indicar as palavras ou expressões que devem estar presentes no atributo Title e, de forma idêntica, utilize um - para excluir outras palavras ou expressões. A função score-basic funciona de modo idêntico à função score no exemplo anterior, mas utiliza uma sintaxe simplificada.

## Pesquisa de texto em artigos de recurso

Esta consulta localiza recursos de texto num tipo de artigo de recurso de texto TextResource que contém o texto Java e o texto XML. Este é um exemplo de execução de pesquisa de texto nos recursos do gestor de recursos. Tenha em atenção que o atributo TIREF é utilizado como uma representação do recurso que é representado pelo artigo de tipo TextResource. Para obter a sintaxe suportada pela função contains-text-db2, consulte o manual *IBM DB2 Universal Database: Net Search Extender Administration and User's Guide* no Information Center do DB2 UDB.

### Exemplo: DB2

```
/TextResource[contains-text-db2(@TIREF, " 'Java' & 'XML'
")=1]
```

## Exemplo: Cruzamento de referências no sentido progressivo

Esta consulta localiza todas as perguntas frequentes em conferências, para as quais as notas de conferência referem livros com títulos que mencionam o DB2 Information Integrator for Content..

```
/Conference/Conference_Note [@PublicationRef =>
Book[@Title LIKE "%EIP%"]]
/Conference_FAQ
```

### Exemplo: Cruzamento de referências no sentido progressivo

Esta consulta localiza todos os capítulos de livros referenciados nas notas de conferências relacionadas com a Internet.

```
/Conference[@Title LIKE "%Internet%"]
/Conference_Note/@PublicationRef =>
*/Book_Chapter
```

### Exemplo: Cruzamento de referências no sentido inverso

Esta consulta localiza todos os componentes que possuem referências que indicam livros.

```
/Book/REFERENCEDBY/@REFERENCER => *
```

### Exemplo: Cruzamento de referências no sentido inverso

Esta consulta localiza todas as perguntas frequentes e notas de conferências que fazem referência a manuais sobre XML.

```
/Book[@Title LIKE "XML"]/REFERENCEDBY/@REFERENCER =>
Conference_Note/Conference_FAQ
```

Tenha em atenção que, uma vez que os atributos de referência têm origem no interior do componente `Conference_Note`, este é o componente que deverá ser apresentado como primeiro componente após o operador de anulação de referência. Esta consulta produz um conjunto de resultados vazio se, por exemplo, `Conference` for apresentado a seguir ao operador `=>`.

### Exemplo: Cruzamento de referências no sentido inverso

Esta consulta localiza todos os componentes que contêm XML nos comentários e que possuem referências que indicam livros.

```
/Book/REFERENCEDBY/@REFERENCER =>
*[@Remark LIKE "%XML%"]
```

### Exemplo: Função de última versão

Esta consulta localiza todos os jornais da última versão. Por predefinição, são devolvidas todas as versões da vista de tipo do componente indicado que correspondam à consulta. `VERSIONID` é um atributo definido pelo sistema que está incluído em todos os tipos de componentes.

```
/Journal[@VERSIONID = latest-version(.)]
```

### Exemplo: Função de última versão no destino da anulação de referência

Esta consulta localiza todos os livros da última versão que são referidos nas notas de quaisquer conferências.

```
/Conference/Conference_Note/@SYSREFERENCEATTRS =>
Book[@VERSIONID = latest-version(.)]
```

### Exemplo: Função de última versão em componentes de caracteres globais

Esta consulta localiza todos os componentes da última versão que possuem referências que indicam quaisquer livros.

```
/Book/REFERENCEDBY/@REFERENCER => *
[@VERSIONID = latest-version(.)]
```

### Exemplo: Atributos definidos pelo sistema

Esta consulta localiza todos os componentes raiz com um ID de artigo específico.

```
/*[@ITEMID =
"A1001001A01J09B00241C95000"]
```

### Pesquisa de texto no modelo de documentos

Esta consulta localiza todos os documentos que contêm a palavra XML em qualquer uma das suas partes. A linguagem de consulta faculta um componente virtual ICMPARTS que permite o acesso a todos os tipos de artigo ICM Parts contidos num tipo de artigo específico de Classificação de documentos.

### Exemplo: DB2

```
/Doc[contains-text-db2(.//ICMPARTS/@TIREF, " 'XML' ")=1]
```

### Exemplo: Modelo de documentos (acesso a Partes ICM)

Esta consulta localiza todas as partes do documento com o ID de armazenamento 555.

```
/Doc[@ArchiveID = 555]/ICMPARTS/
@SYSREFERENCEATTRS => *
```

### Exemplo: Modelo de documentos (acesso a Partes ICM)

Esta consulta localiza todas as partes de todos os documentos do sistema.

```
//ICMPARTS/@SYSREFERENCEATTRS => *
```

Uma vez que os tipos de artigo Doc e Paper foram definidos como Documentos no sistema, as Partes ICM de ambos os tipos de artigo são devolvidos no resultado.

### Exemplo: Existência de atributos

Esta consulta localiza todos os componentes raiz que possuem título.

```
/*[@Title]
```

Para eliminar a restrição que estipula apenas a devolução de componentes raiz, a consulta pode ser novamente escrita com duas barras.

```
/**[@Title]
```

### Exemplo: Lista de literais e expressões

Esta consulta localiza todos os diários que possuem um título igual ao título do artigo, ao título da secção ou a IBM Systems Journal.

```
/Journal[@Title = [Journal_Article/@Title,
./Journal_Section/@Title,"IBM Systems Journal"]]
```

### Exemplo: Lista de literais

Esta consulta localiza todos os livros que custam 10, 20 ou 30 dólares.

```
/Book[@Cost IN (10, 20, 30)]
```

Embora seja possível efectuar a mesma consulta com o operador de lista, tal como em `/Book[@Cost = [10, 20, 30]]` (resultados menos positivos), para um grande número de literais, esta abordagem pode induzir em erro, dado que o SQL gerado seria demasiado longo ou demasiado complexo. Se todos os elementos da lista forem literais, utilize sempre o operador `IN` para obter um melhor desempenho e o SQL mais curto. Pode utilizar o operador `IN` para literais de qualquer tipo incluindo tipos não numéricos.

### Exemplo: Lista de um resultado de consulta

Esta consulta localiza todos os jornais ou livros com o título Star Wars.

```
[/Journal, /Book[@Title = "Star Wars"]]
```

### Exemplo: Grupos de atributos

Esta consulta localiza todos os detalhes em documentos cuja descrição é de, pelo menos, 20 páginas.

```
/Doc[Doc_Description/@PageSummary.NumPages >= 20]//Doc_Details
```

Tenha em atenção que se um atributo (por exemplo, `NumPages`) estiver contido num grupo de atributos (por exemplo, `PageSummary`), é necessário fazer referência a esse atributo como `GroupName.AttrName` (por exemplo, `PageSummary.NumPages`). O atributo `@NumPages` não seria localizado em `Doc_Description`.

### Exemplo: Artigos com saída dada

Esta consulta localiza todos os artigos do tipo de artigo Journal actualmente com saída dada.

```
/Journal [ICMCHECKEDOUT]
```

O elemento de XML `ICMCHECKEDOUT` é um subelemento apenas dos componentes raiz e não dos componentes descendentes. Deste modo, se o elemento `ICMCHECKEDOUT` for escrito numa consulta como condição de um componente descendente (por exemplo, `//Journal_Author [ICMCHECKEDOUT]`), não são devolvidos quaisquer resultados.

Quando a um artigo é dada saída, também é dada saída a todas as versões desse artigo. Deste modo, quando um elemento `ICMCHECKEDOUT` é aplicado a um artigo com saída dada, todas as versões actualmente disponíveis serão devolvidas. Para obter uma versão específica, pode ainda utilizar a sintaxe de consulta `@VERSIONID` (por exemplo, `/Journal [ICMCHECKEDOUT AND @VERSIONID = 4]`). Para a versão mais recente, pode utilizar a função `latest-version()`.

### Exemplo: Artigos com saída dada por pessoa

Esta consulta localiza todos os artigos com saída dada por SMITH.

```
/Journal [ICMCHECKEDOUT/@ICMCHKOUTUSER = "SMITH"]
```

O valor de `ICMCHKOUTUSER` tem de ser introduzido em maiúsculas numa consulta. Uma vez que os servidores de conteúdos armazenam IDs de utilizador como maiúsculas, todas as consultas têm de pesquisar os IDs de utilizador utilizando maiúsculas. Todos os dados de atributos relativos a IDs de utilizador têm de os armazenar em letras maiúsculas.

## Exemplo: Artigos com saída da por marca de hora

Esta consulta localiza todos os artigos com saída dada após 2003-08-02-17.29.23.977001.

```
/Journal [ICMCHECKEDOUT/@ICMCHKOUTTS > "2003-08-02-17.29.23.977001"]
```

---

## Utilizar sequências de abandono nas suas consultas

O processamento correcto de caracteres especiais garante uma execução com êxito de consultas e obtém os resultados de consulta pretendidos.

Para suportar funções avançadas de linguagem de consulta (como, por exemplo, caracteres globais % ou \_ dentro de cadeias de texto), são utilizadas sequências de abandono para diferenciar situações em que os caracteres globais são considerados caracteres normais de situações em que lhes é atribuído o significado especial de caracteres globais. Para o utilizador, é importante saber quais os caracteres utilizados como globais, pois quando se pretende tratar os caracteres globais como caracteres normais, estes devem ser precedidos por um carácter de abandono. As sequências de abandono também são utilizadas para processar plicas e aspas.

Necessita de adicionar sequências de abandono quando as cadeias utilizadas em consultas contêm caracteres especiais (aspas, apóstrofes) ou caracteres globais (sinal de percentagem, traço de sublinhado, asterisco, ponto de interrogação) ou um carácter de abandono predefinido (uma barra invertida). Este processamento é o mais simples relativamente a cadeias utilizadas em condições de comparação e envolve-se mais no operador LIKE e nas funções de pesquisa de texto.

**Importante:** Utilize caracteres globais de forma regrada, uma vez a sua utilização nas consultas pode aumentar significativamente o tamanho da lista de resultados, o que pode reduzir o desempenho e devolver resultados de pesquisa inesperados.

## Utilizar sequências de abandono com o operadores de comparação

Nem sempre é necessário utilizar sequências de abandono com operadores de comparação.

### Exemplo: aspas "

Coloque dois caracteres de aspas seguidos.

```
//Journal_Article[@Title = "Analysis of ""The Time Machine"" by H. G. Wells himself"]
```

Uma vez que o título do artigo contém o nome do livro entre aspas, The Time Machine, estas aspas internas devem ser abandonadas.

### Exemplo: plica (apóstrofo) '

Neste caso, não necessita de efectuar o abandono.

```
/Book[@Title != "Uncle Tom's Cabin"]
```

## Utilizar sequências de abandono com o operador LIKE

As sequências de abandono nem sempre são necessárias com operador LIKE, mas normalmente são.

### Aspas "

Coloque dois caracteres de aspas seguidos.

```
//Journal_Article[@Title LIKE "Analysis of ""The Time Machine"" %"]
```

Uma vez que o título do artigo contém o nome do livro entre aspas, "The Time Machine", estas aspas internas devem ser abandonadas.

### Plica (apóstrofo) '

Neste caso, não é necessário efectuar o abandono. /Book[@Title LIKE "Uncle Tom's Cabin"]

### Caracteres globais ("%", "\_")

O sinal de percentagem % é um carácter global utilizado para representar um determinado número de caracteres arbitrários numa cadeia utilizada com o operador LIKE. O traço de sublinhado \_ é um carácter global utilizado para representar um único carácter arbitrário. Se pretender que estes caracteres globais sejam tratados como caracteres normais, deverá efectuar o seguinte:

1. Coloque um carácter de abandono antes do carácter global.
2. Adicione uma cláusula ESCAPE com o carácter de abandono após a expressão LIKE.

#### Exemplo A

```
/Book[@Title LIKE "Plato%S%S_mposium"]
```

Este exemplo apresenta a forma como os caracteres globais % e \_ são utilizados para localizar um livro cuja ortografia do título é incerta.

#### Exemplo B

```
//Journal_Article[@Title LIKE "Usage of underscore !_ in query"
ESCAPE "!"]
```

Uma vez que a cadeia de pesquisa neste exemplo contém o traço de sublinhado \_ como carácter normal (não como carácter global), pode abandonar o traço de sublinhado com um carácter de ponto de exclamação !. Qualquer carácter único pode ser utilizado como um carácter de abandono.

#### Exemplo C

```
//Journal_Article[@Title LIKE "_sage of underscore _ in%" ESCAPE
"\"]
```

Nesta consulta, os caracteres globais são utilizados como caracteres normais ( \_ abandonados por \) e como caracteres globais ( \_ ) para localizar versões em maiúsculas e minúsculas da palavra "Usage", bem como % para localizar várias terminações da cadeia.

#### Exemplo D

```
//Journal_Article[@Title LIKE "Usage of underscore !_ on Yahoo!!"
ESCAPE "!"]
```



Pode também utilizar um carácter de abandono como um carácter normal. Para esse efeito, coloque dois caracteres de abandono iguais seguidos, tal como no exemplo de pesquisa de "Yahoo!" apresentado em baixo.

## Utilizar sequências de abandono nas pesquisas de texto avançadas

As sequências de abandono são necessárias ao utilizar aspas e caracteres globais em pesquisas de texto.

### Aspas "

Coloque dois caracteres de aspas seguidos.

#### DB2

```
//Journal_Article[contains-text-db2 (@Title,
" 'Analysis of "'The Time Machine'" %' ")=1]
```

Uma vez que o título do artigo contém o nome do livro entre aspas, "The Time Machine", estas aspas internas devem ser abandonadas.

### Plica (apóstrofo) '

Coloque dois apóstrofos seguidos. Não é permitido apenas um apóstrofo na pesquisa de texto avançada, pois um conjunto de apóstrofos é utilizado para delimitar termos ou expressões. Se for apresentado um apóstrofo dentro de um termo, o apóstrofo deverá ser abandonado para o diferenciar do apóstrofo que termina o termo ou expressão.

#### DB2

```
/Book[contains-text-db2 (@Title,
" 'Uncle Tom''s Cabin' ")=1]SORTBY
(score-db2 (@Title, " 'Uncle Tom''s Cabin' "))
```

**Nota:** Tom''s tem dois apóstrofos.

#### DB2

```
/Book[contains-text-db2 (@Title,
" ('Greek' & 'Plato''s Symposium') & NOT ' Socrates' ")=1]
SORTBY (score-db2 (@Title, " ('Greek' & 'Plato''s Symposium')
& NOT ' Socrates' "))
```

**Nota:** Plato''s tem dos apóstrofos.

### Caracteres globais ("% , " \_")

Tal como o operador LIKE, a sintaxe avançada utiliza % e \_ como caracteres globais. O sinal de percentagem % é um carácter global utilizado para representar um determinado número de caracteres arbitrários. O traço de sublinhado \_ é um carácter global utilizado para representar um único carácter arbitrário. Se pretender que um carácter global seja tratado como um carácter normal, deverá efectuar o seguinte:

1. Coloque um carácter de abandono antes do carácter global.
2. Adicione uma cláusula ESCAPE após CADA termo em que utiliza o carácter de abandono.

### Exemplo A

```
/Book[contains-text-db2 (@Title,
" 'Usage of underscore !_ in query' ESCAPE '!' ")=1]SORTBY (score-db2 (@Title,
" 'Usage of underscore !_ in query' ESCAPE '!' ")
```

Neste exemplo, é utilizando um ponto de exclamação ! como carácter de abandono antes do traço de sublinhado.

### Exemplo B

```
/Book[contains-text-db2 (@Title,
" 'Usage of underscore !_ in query' ESCAPE '!'
| 'Yahoo! For Dummies'
| 'Usage of underscore !_ on Yahoo!!' ESCAPE '!' | 'War and Peace' ")=1]
```

Tenha em atenção que deve ser adicionada uma cláusula ESCAPE após cada termo da cadeia de pesquisa de texto em que efectua o escape dos caracteres globais, mesmo que o carácter de abandono seja igual em todos os termos.

## Utilizar sequências de abandono com pesquisa de texto básica (funções contains-text-basic e score-basic)

As sequências de abandono são necessárias ao utilizar aspas e caracteres globais em pesquisas de texto.

### Aspas "

Coloque dois caracteres de aspas seguidos.

```
//Journal_Article[contains-text-basic (@Title,
"Analysis of '"The Time Machine'" ")=1]
```

Uma vez que o título do artigo contém o nome do livro entre aspas, "The Time Machine", estas aspas internas devem ser abandonadas. O título do livro encontra-se entre apóstrofes para o distinguir como expressão.

### Plica (apóstrofo) '

Coloque dois apóstrofes seguidos. A sintaxe de pesquisa de texto básica permite termos entre plicas, de modo que um termo pode conter um espaço. Consequentemente, é necessário colocar dois apóstrofes seguidos para diferenciar o caso de um apóstrofo que surge dentro de um termo e o caso de um apóstrofo que inicia um termo novo.

### Exemplo A

```
/Book[contains-text-basic (@Title,
"Uncle Tom''s Cabin")=1]
SORTBY (score-basic (@Title, "Uncle Tom''s Cabin"))
```

Tom''s tem dois apóstrofes.

### Exemplo B

```
/Book[contains-text-basic (@Title,
" +Greek +'Plato''s Symposium' -Socrates ")=1]
SORTBY (score-basic (@Title, " +Greek +'Plato''s Symposium' -Socrates "))
```

Tenha em atenção que Plato''s possui dois apóstrofes e 'Plato's Symposium' se encontra entre plicas, uma vez que é uma expressão.

## Caracteres globais ("\*", "?" e "\")

Coloque uma barra invertida \ antes dos caracteres \*, ? e \, caso estes caracteres não devam ser considerados caracteres globais. O asterisco \* é um carácter global utilizado para representar um determinado numero de caracteres arbitrários numa pesquisa de texto básica para as funções contains-text-basic e score-basic. O ponto de interrogação ? é um carácter global utilizado para representar um único carácter arbitrário. Para a pesquisa de texto básica, a linguagem de consulta faculta um barra invertida \ de carácter de abandono que deve ser utilizada quando o termo a pesquisar contém o carácter global e o utilizador pretende que esse carácter global seja considerado um carácter normal.

### Exemplo A

```
/Book[contains-text-basic (@Title,
" +Greek +'Plato*s*S?mposium' -Socrates ")=1]
SORTBY (score-basic (@Title,
" +Greek +'Plato*s*S?mposium' -Socrates "))
```

Este exemplo mostra como utilizar a pesquisa de texto básica quando a ortografia de um termo não é certa. O \* e o ? devem, neste caso, ser globais e, por isso, não devem ser abandonados.

### Exemplo B

```
/Book[contains-text-basic (@Title,
"Why forgive\?")=1]SORTBY
(score-basic (@Title, "Why forgive\?"))
```

Neste exemplo, o título contém o ponto de interrogação ? como carácter normal e, por isso, este carácter pode ser abandonado com uma barra invertida.

### Exemplo C

```
//Journal_Section[contains-text-basic
(@Title, "C:\\OurWork\\IsNeverDone")=1]
SORTBY (score-basic (@Title, "C:\\OurWork\\IsNeverDone"))
```

Cada barra invertida que ocorre naturalmente no termo de pesquisa "C:\\OurWork\\IsNeverDone" deve ser abandonada com outra barra invertida.

## Utilizar sequências de abandono em Java e C++

Coloque uma barra invertida antes dos caracteres especiais (por exemplo, as aspas e a barra invertida).

As sequências de abandono de Java e C++ são processadas da seguinte forma:

Exemplo:

Consulta:

```
/Book[contains-text-basic (@Title, "Why forgive\?")=1]
```

### Exemplo: Java

```
String query = "/Book[contains-text-basic (@Title, \"Why forgive\\?\")=1]";
```

### Exemplo: C++

```
DKString query ("/Book[contains-text-basic (@Title,
\"Why forgive\\?\")=1]");
```

Tenha em atenção que as aspas internas e a barra invertida antes do ponto de interrogação são precedidas por uma barra invertida. Este processamento é inerente às linguagens de programação de Java e de C++. Para obter mais informações, consulte as especificações relativas a estas linguagens.

---

## Compreender os filtros de vistas com base em linhas nas consultas

Os filtros de vistas com base em linhas ajudam a limitar a quantidade de dados obtidos para uma determinada vista. Uma utilização inadequada de filtros de vistas com base em linhas pode resultar num aumento significativo do comprimento do SQL gerado e num decréscimo no desempenho da consulta.

Com os filtros de vistas com base em linhas, pode filtrar um componente com base nos conteúdos de um dos atributos do componente. Ao ter vistas diferentes ao mesmo tipo de artigo com condições de filtro diferentes, pode separar os dados de um tipo de artigo em blocos lógicos, permitindo aos utilizadores ver apenas determinados dados, dependendo da vista que é utilizada para aceder aos dados. Por conseguinte, numa consulta, os filtros de vistas com base em linhas ajudam a limitar automaticamente a quantidade de dados obtidos para uma determinada vista.

**Importante:** No sistema do DB2 Content Manager, a consulta é convertida para uma cadeia de consulta SQL que é executada nas tabelas de base de dados adjacentes. Como os sistemas de base de dados têm um limite para o comprimento da cadeia de consulta SQL, a utilização inadequada dos filtros pode fazer com que esta cadeia se torne tão longa que exceda os limites, obstando assim à boa execução da consulta. Deve rever a questão do rendimento antes de optar por utilizar esta função.

## Considerações importantes para utilização de filtros baseados em linhas

Se utilizar filtros baseados em linhas, existem algumas directrizes de utilização importantes que deverá ter em consideração para obter um melhor desempenho.

Para operações que podem devolver documentos ou itens de vários tipos de itens, tais como obter conteúdos de uma pasta ou lista de trabalho, dependendo da definição de ACL, a forma como o DB2 Content Manager selecciona os subconjuntos e aplica as condições do filtro baseado em linhas pode não ser clara e afectar os conjuntos de resultados.

Se existirem subconjuntos (também denominados de vistas) com condições de filtro baseado em linhas definidos nos tipos de itens do documento e se as ACLs estiverem definidas para permitir que determinados utilizadores possam ver os documentos ou itens de vários subconjuntos, o DB2 Content Manager pode seleccionar um subconjunto aleatoriamente e a acção de obtenção dos conteúdos de uma pasta ou lista de trabalho poderá não devolver todos os documentos que deveriam estar no resultado.

Estes problemas podem ocorrer nas seguintes operações:

- Ao obter documentos utilizando o Cliente do Windows ou eClient e quando o utilizador abre uma pasta ou lista de trabalho.
- Ao criar uma aplicação personalizada (Java ou C++) utilizando a DB2 Content Manager API que invoca a função `getWorkList` ou que obtém um item de pasta através do método `DKConstant.DK_CM_CONTENT_LINKS_OUTBOUND` ou

DKConstant.DK\_CM\_CONTENT\_ITEMTREE (no DB2 Content Manager V8.2 e todos os pacotes de correcções e no DB2 Content Manager V8.3, níveis de pacotes de correcções anteriores ao pacote de correcções 4) ou do método DKRetrieveOptionsICM::linksOutbound(true) (no DB2 Content Manager V8.3 pacote de correcções 4 e posterior).

Geralmente estes problemas podem ocorrer devido a duas causas principais:

- Uma das causas pode dever-se ao facto de, durante a acção de obtenção, o utilizador ter especificado uma vista activa na qual as condições do filtro filtram alguns documentos a serem obtidos, mesmo se esses documentos estiverem acessíveis em conformidade com a definição de ACL. Este problema surge porque a forma como o subconjunto é seleccionado pode não corresponder às expectativas do utilizador, por exemplo:
  - No Cliente do Windows ou noutra aplicação, os utilizadores podem seleccionar explicitamente o subconjunto para o tipo de item do documento.
  - No eClient e noutras aplicações que não permitem a selecção da vista activa, o DB2 Content Manager selecciona automaticamente uma vista com uma ACL que permite ao utilizador aceder ao documento. Se existirem várias vistas disponíveis, a aplicação selecciona uma vista aleatoriamente. Geralmente esta vista é seleccionada segundo a ordem pela qual o subconjunto foi criado.

Como resultado, a vista activa seleccionada pode corresponder a uma vista incorrecta com uma condição de filtro baseada em linhas, uma condição que irá filtrar os documentos.

- Outra causa pode dever-se ao facto de as operações de obtenção serem efectuadas em dois passos. No primeiro passo, a lista de IDs de documento é devolvida com base na definição de ACL do subconjunto. No segundo passo, o filtro baseado em linhas é aplicado quando cada ID de item é acedido. Como resultado, se o primeiro passo originar um grande número de IDs, o desempenho da operação de obtenção pode ficar afectado.

Se utilizar filtros baseados em linhas em subconjuntos, os utilizadores devem ter conhecimento da forma como estes subconjuntos podem afectar os conjuntos de resultado, conforme descrito acima. Conceba cuidadosamente os subconjuntos, condições de filtro e ACLs em conformidade.

**Sugestão:** Não utilize uma condição de filtro baseada em linhas, caso já esteja a utilizar a associação automática para colocar documentos em pastas diferentes. Em alternativa, utilize as ACLs para restringir o acesso às pastas, se necessário. Se possível, conceba cuidadosamente os subconjuntos e os grupos de utilizadores, de forma a que os utilizadores possam ver todos os documentos que é suposto verem através da selecção de um único subconjunto.

**Importante:** Se utilizar o filtro baseado em linhas para filtrar os conteúdos das pastas ou listas de trabalho, a utilização deste filtro pode originar problemas de desempenho. Isto acontece uma vez que a obtenção começa por listar todos os itens independentemente da condição de filtro e, em seguida, filtra os itens quando os atributos são obtidos.

## Exemplo de situação de utilização

Numa perspectiva de nível elevado, os filtros de vistas com base em linhas são utilizados no sistema do DB2 Content Manager.

De seguida, são apresentados os passos envolvidos nesta situação:

1. Defina um tipo de artigo denominado Journal.
2. Adicione alguns artigos ao tipo de artigo.
3. Defina uma vista de tipo de artigo denominado MyJournal com o seguinte filtro: @Organization = "IBM"
4. Execute uma consulta com a vista de tipo de artigo MyJournal.
5. Apresente os os resultados da consulta ao utilizador. Apenas são devolvidos diários da organização IBM.

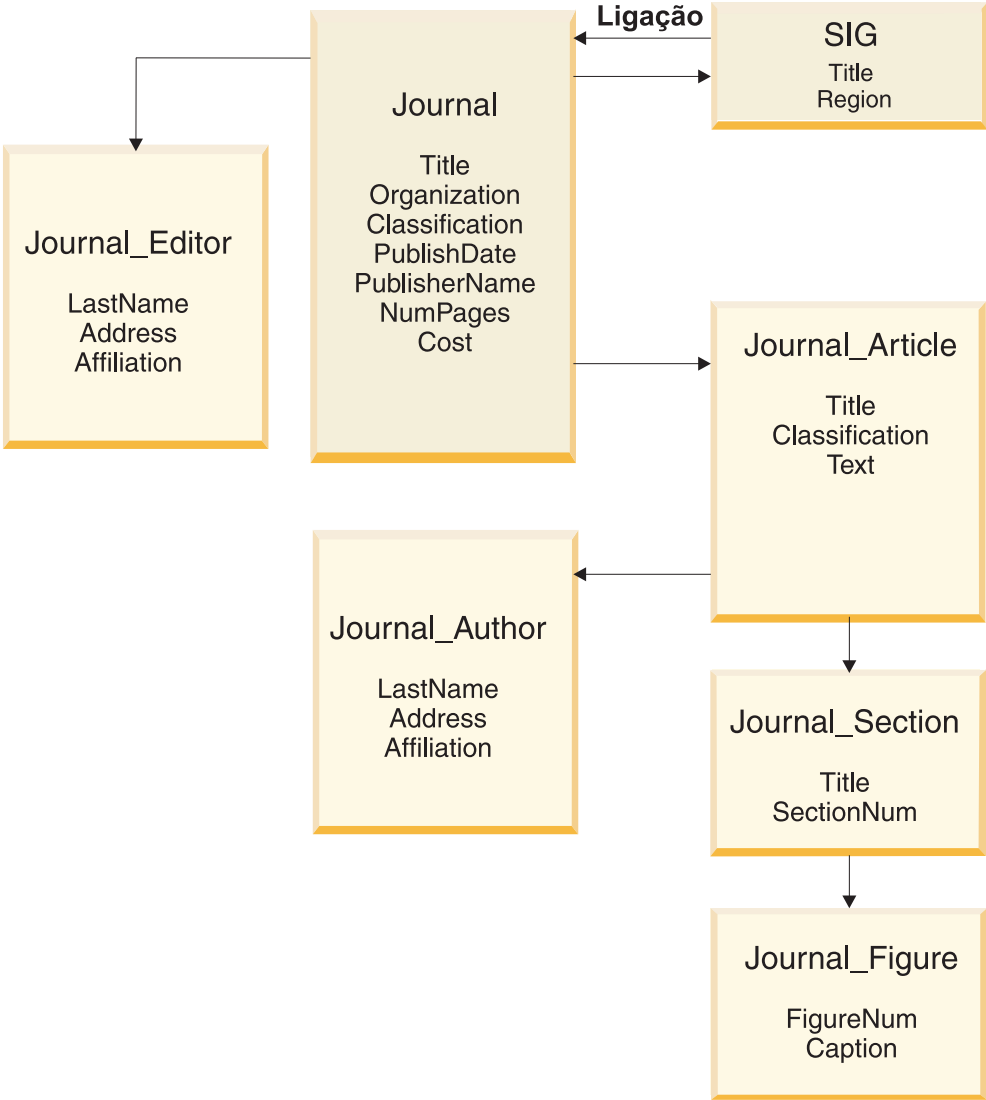


Figura 12. Exemplo de modelo de dados para consulta com filtros de vistas com base em linhas

Tabela 28. Exemplo de definição de vistas com filtros

| Vista de tipo de componente de utilizador | Tipo de componente base | Condição de filtro         |
|-------------------------------------------|-------------------------|----------------------------|
| MyJournal                                 | Journal                 | @Organization = "IBM"      |
| MyJournal_Article                         | Journal_Article         | @Classification = "Public" |
| MyJournal_Section                         | Journal_Section         | Nenhum                     |

Tabela 28. Exemplo de definição de vistas com filtros (continuação)

| Vista de tipo de componente de utilizador | Tipo de componente base | Condição de filtro       |
|-------------------------------------------|-------------------------|--------------------------|
| MyJournal_Figure                          | Journal_Figure          | @FigureNum = 5           |
| MyJournal_Author                          | Journal_Author          | @Affiliation = "Almaden" |
| MyJournal_Editor                          | Journal_Editor          | @Affiliation = "Almaden" |
| MySIG                                     | SIG                     | @Region = "USA"          |

## Descrição do comportamento de filtros

A forma como os filtros de vistas com base em linhas se comportam depende do tipo de consulta que pretende executar. Os filtros com base em linhas em consultas do DB2 Content Manager aplicam-se em vistas de tipos de componentes mencionadas de forma explícita na cadeia de consulta ou de forma implícita por um carácter global ou uma consulta num dos atributos de vista.

A lista que se segue descreve a forma como os filtros de vistas com base em linhas se comportam.

### Filtros de componente raiz

Um filtro na vista de tipo de componentes raiz pode ser utilizado para filtrar o conteúdo de todo o artigo se a vista de tipo de componente raiz for expressamente referida na consulta.

#### Exemplo

Existem 1.000.000 componentes do tipo de componente Journal no sistema. 1.000 destes componente têm "IBM" no atributo Organization. Execute a seguinte consulta para obter todos os diários associados à vista MyJournal:"/MyJournal"

#### Resultado

São apenas devolvidos 1.000 componentes da consulta, uma vez que a consulta é equivalente a `"/MyJournal [@Organization = "IBM"]`. Os resultados não incluem artigos para os quais o filtro com base em linhas em Organization não corresponde aos dados. Por exemplo, os artigos que têm Microsoft e Sun no atributo Organization não são devolvidos. A vantagem é que não há necessidade de especificar as condições de filtro expressamente, uma vez que são aplicadas automaticamente com base na vista consultada.

**Restrição:** Não é possível filtrar um componente com base em filtros dos respectivos descendentes. Os filtros não se aplicam na direcção descendente da hierarquia em árvore. Tenha em atenção que os filtros de MyJournal\_Editor ou de MyJournal\_Figure não se aplicam à consulta `"/MyJournal"` acima.

### Filtros de componente descendente

Quando é utilizada uma vista de tipo de componente descendente numa consulta, aplica-se o filtro específico para essa vista descendente.

#### Exemplo

Execute a seguinte consulta para obter todos os artigos de diário disponíveis para um indivíduo: `"//MyJournal_Article"`

#### Resultado

A condição de filtro `@Classification = "Public"` é aplicada em on



MyJournal\_Article. Por conseguinte, só são obtidos os artigos para consumo público, mas os ensaios e outros artigos que possam não ser aprovados para o público são ignorados.

**Restrição:** Não é possível filtrar um componente com base nos filtros do respectivo ascendente. Por exemplo, para a consulta `"/MyJournal_Article"`, o filtro apenas é aplicado em MyJournal\_Article. Não é aplicado um filtro na vista ascendente MyJournal.

Se pretende aplicar um filtro ascendente, pode seguir uma das seguintes abordagens:

- Escreva novamente a consulta para percorrer a raiz, tal como neste exemplo: `"/MyJournal/MyJournal_Article"`
- Adicione o mesmo atributo de filtro e condição de filtro, tal como adicionou à raiz quando concebeu o tipo de artigo, e preencha o valor para o atributo de filtro descendente com os mesmos dados que o atributo de filtro da raiz. Por exemplo, pode adicionar o atributo Organization a MyJournal\_Article com a seguinte condição de filtro: `@Organization = "IBM"`.

**Restrição:** Não é possível filtrar um componente com base nos filtros dos respectivos equivalentes ou parentes distantes. Por exemplo, para a consulta `"/MyJournal_Editor"`, o filtro apenas é aplicado em MyJournal\_Editor. Não se aplicam filtros em MyJournal\_Article ou em qualquer um dos respectivos descendentes (MyJournal\_Author e MyJournal\_Figure).

### Filtros em intermediários

Com a linguagem de consulta do DB2 Content Manager, é possível cruzar ligações e referência antes de chegar ao componente final. Os filtros aplicam-se a quaisquer componentes envolvidos nesses cruzamentos intermédios.

#### Exemplo

Execute a seguinte consulta para obter todos os grupos interesses especiais com ligações a publicações de um determinado editor: `"/MyJournal [@PublisherName = "Acme Publishing"]/INBOUNDLINK/@SOURCEITEMREF => MySIG"`

#### Resultado

Embora não sejam devolvidos diários como resultado desta consulta, é aplicada uma condição de filtro (`@Organization = "IBM"`) no tipo de componente Journal, filtrando, assim, quaisquer SIGs com ligações a diários de outras organizações.

### Filtros em caracteres universais

Como um determinado carácter global consulta todas as vistas de tipos de componentes activas para um determinado utilizador, as condições de filtro são aplicadas em qualquer uma destas vistas se tiverem os filtros definidos.

#### Exemplo

Execute a seguinte consulta para localizar todos os artigos ligados a grupos de interesses especiais denominados "XML": `"/MySIG [@Title = "XML"]/OUTBOUNDLINK/@TARGETITEMREF => *"`

#### Resultado

O carácter global (\*) é expandido para representar todas as vistas

de tipos de componentes, sendo adicionadas condições de filtro às vistas MyJournal e MySIG para garantir que apenas os diários que correspondam à IBM e apenas os SIGs nos E.U.A. são considerados resultados finais.

## Considerações de desempenho

Obter o melhor desempenho de consulta depende da forma como utiliza os filtros e os caracteres globais.

Sigas estas recomendações nas consultas.

### Tornar os filtros altamente restritivos

Um filtro restritivo só corresponde a uma pequena quantidade do número total de linhas num tipo de componentes. A utilização de filtros restritivos leva a melhores planos de execução para as consultas das bases de dados.

Exemplo: No exemplo anterior, o filtro no atributo MyJournal Organization é um filtro restritivo, uma vez que apenas 1.000 componentes, de um total de 1.000.000, têm Organization = "IBM".

### Minimizar a utilização de caracteres universais nas consultas

Minimizar a utilização de caracteres universais nas consultas é sempre boa ideia. Em particular, dado que a aplicação de cada filtro envolve normalmente a adição de mais condições e junções às consultas, a combinação desta complexidade extra pode ser significativa quando são usados caracteres universais. Como um carácter global consulta todas as vistas activas de um utilizador no sistema, a consulta vai aumentando com o aumento do número de vistas filtradas. Sempre que possível, utilize a vista de um tipo de componente específica em vez de um carácter global.

Exemplo: Se souber que, no SIG do modelo de dados, os artigos apenas têm ligação a Journals ou Books, utilize os nomes específicos destas vistas de tipo de componente na consulta em vez do carácter global para indicar o destino do cruzamento da ligação. Por exemplo, para localizar todos os artigos ligados a grupos de interesses especiais denominados "XML", escreva novamente a consulta da seguinte forma:

#### Sub-optimal

```
"/MySIG [@Title = "XML"]/OUTBOUNDLINK/@TARGETITEMREF => *
```

#### Optimal

```
"/MySIG [@Title = "XML"]/OUTBOUNDLINK/@TARGETITEMREF =>
MyJournal UNION /MySIG [@Title = "XML"]/OUTBOUNDLINK/
@TARGETITEMREF => MyBook"
```

### Evite definir demasiados filtros no sistema

Quanto mais vistas de filtros tiver o sistema, mais complexas se tornam as consultas finais de base de dados quando utiliza caracteres universais. Quando optar por utilizar filtros, defina estes filtros apenas nas vistas que realmente precisam de tais filtros e evite a utilização de caracteres universais.

Exemplo: A consulta simples `"/* [@ITEMID = "myItemID"]` para obter um artigo com um ITEMID específico pode resultar numa consulta de SQL complexa final caso tenha, por exemplo, 30 vistas de tipos de artigo raiz no sistema, tendo 25 destas filtros definidos. Para cada um destes filtros, é necessário adicionar mais condições e junções à consulta de base de dados.

Por exemplo, se souber que o artigo pertence a um tipo de artigo Journal ou SIG, escreva novamente a consulta da seguinte forma: "(Journal | SIG) [@ITEMID = "myItemID"]"

Como existe um limite ao comprimento da cadeia de consulta SQL para que a base de dados possa processá-la, algumas das consultas podem lançar exceções, caso tenha muitos filtros definidos e utilize caracteres universais.

## Índices de base de dados em cada atributo filtrado

Alguns índices de base de dados são definidos pelo sistema. No entanto, alguns índices podem ser definidos por um administrador do sistema para melhorar o desempenho.

Esta secção descreve a relação entre os índices de bases de dados e o rendimento de consultas com filtros de vistas com base em linhas.

### Os índices são automaticamente definidos pelo sistema do DB2 Content Manager

Quando um filtro com base em linhas é definido numa vista de tipo de componentes, o servidor de bibliotecas tenta automaticamente criar um índice da base de dados na coluna correspondente ao atributo do filtro. Este índice deve facilitar a melhoria do rendimento para consultas complexas escritas num modelo de dados que utilize filtros com base em linhas. O sistema da base de dados pode utilizar o índice para criar um plano melhor de acesso para executar uma consulta complexa.

Exemplo: Se o atributo Organization for especificado como um atributo de filtro durante a criação da vista de utilizador MyJournal no tipo de artigo Journal, é criado um índice de base de dados na coluna correspondente ao atributo Organization na tabela do tipo de componente base de Journal.

O índice é criado no atributo do filtro, independentemente da vista de tipo de componentes na qual existe este atributo ser uma vista de raiz ou uma das vistas descendentes. Além disso, o servidor de bibliotecas depende do sistema de base de dados para determinar se o índice pode ser criado. Nalguns casos, o índice pode não ser criado, por exemplo, se já existir um índice. Para verificar se um índice foi criado no atributo de filtro Organization para Journal, por exemplo, o administrador do sistema pode abrir o cliente de administração do sistema do DB2 Content Manager, **Data Modeling (Modelação de Dados) → Item Types (Tipos de artigo) → Journal (Diário) → Database Indexes (Índices de Base de Dados)**, e procurar uma lista com um índice no atributo Organization.

### Índices que o administrador do sistema pode definir para melhorar o rendimento.

Para melhor o rendimento de algumas consultas com caracteres universais no sistema do DB2 Content Manager com vistas de filtro, o administrador do sistema pode definir índices das seguintes colunas nas tabelas do sistema:

ICMSTITEMS001001.COMPONENTTYPEID  
ICMSTITEMVER001001.COMPONENTTYPEID  
ICMSTRI001001.SOURCECOMPTYPEID

Para tabelas de componentes de utilizador do DB2 Content Manager (ICMUT\* tables) com atributos de referência, pode criar índices na coluna RTARGETCOMPTYPEID (para o atributo de referência SYSREFERENCEATTRS) ou em qualquer coluna ATTRXXXXX00110 (para

os atributos de referência definidos pelo utilizador), de forma a que XXXXX seja o ID de grupo de atributos do atributo de referência, por exemplo, 01005.

## Implicações de Segurança

Embora seja possível utilizar filtros de vistas com base em linhas para imitar o mecanismo de segurança, existem algumas limitações que o utilizador deve considerar antes de utilizar esta função. É importante que compreenda exactamente como o filtro é aplicado em consultas e noutras partes do sistema.

Nas consultas, há casos em que os filtros não se aplicam. Além das consultas, o utilizador pode ter acesso a vistas de base de dados que permitem ver os dados filtrados. Uma aplicação pode também ser capaz de obter componentes descendentes directamente, utilizando outras partes da API se os PIDs dos componentes forem conhecidos previamente.

---

## Gramática da linguagem de consulta

Melhore a gramática de consulta com estas palavras-chave e literais na anotação Extended Backus-Naur Form (EBNF).

### (\* palavras-chave \*)

- **AND** = (a | A), (n | N), (d | D) ;
- **ASCENDING** = (a | A), (s | S), (c | C), (e | E), (n | N), (d | D), (i | I), (n | N), (g | G);
- **BETWEEN** = (b | B), (e | E), (t | T), (w | W), (e | E), (e | E), (n | N);
- **DESCENDING** = (d | D), (e | E), (s | S), (c | C), (e | E), (n | N), (d | D), (i | I), (n | N), (g | G);
- **DIV** = (d | D), (i | I), (v | V) ;
- **EXCEPT** = (e | E), (x | X), (c | C), (e | E), (p | P), (t | T);
- **INTERSECT** = (i | I), (n | N), (t | T), (e | E), (r | R), (s | S), (e | E), (c | C), (t | T);
- **LIKE** = (l | L), (i | I), (k | K), (e | E);
- **MOD** = (m | M), (o | O), (d | D);
- **NOT** = (n | N), (o | O), (t | T);
- **OR** = (o | O), (r | R);
- **SORTBY** = (s | S), (o | O), (r | R), (t | T), (b | B), (y | Y);
- **UNION** = (u | U), (n | N), (i | I), (o | O), (n | N);
- **IS** = (i | I), (s | S);
- **NULL** = (n | N), (u | U), (l | L), (l | L);
- **IN** = (i | I), (n | N);
- **ESCAPE\_KEYWORD** = (e | E), (s | S), (c | C), (a | A), (p | P), (e | E);
- **KEYWORD** = (AND | ASCENDING | BETWEEN | DESCENDING | DIV | EXCEPT) | INTERSECT | LIKE | MOD | NOT | OR | SORTBY | UNION | IS | NULL | IN);

### (\* literais \*)

- **DIGIT** = ( 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 );
- **NONZERO\_DIGIT** = ( 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 );
- **EXPONENT** = (e | E), [+ | -], DIGIT, { DIGIT };

- **INTEGER\_LITERAL**= 0 | NONZERO\_DIGIT), { DIGIT };
- **FLOAT\_LITERAL**= DIGIT, { DIGIT }, ., { DIGIT }, [EXPONENT] | [., DIGIT, { DIGIT }, [EXPONENT];
- (\***UNICODE\_CHARACTER** é o conjunto de todos os caracteres Unicode e sequências de abandono. A respectiva definição não está incluída neste documento \*) (\*Os literais de cadeia estão delimitados por aspas e podem conter quaisquer caracteres, excepto aspas. Para incluir aspas como parte do literal da cadeia, especifique duas aspas consecutivas, ou seja, um carácter de aspas é abandonado por outro carácter de aspas. Estes serão considerados como um carácter de aspas \*)
- **STRING\_LITERAL**= "", { (UNICODE\_CHARACTER - ""), | ("", "")), "";
- (\* A sequência de abandono é um único carácter delimitado por aspas. Para especificar as próprias aspas como carácter de escape, especifique duas aspas consecutivas, ou seja, um carácter de aspas é abandonado por outro carácter de aspas. Estes serão tratados como um carácter de aspas. Para obter uma explicação completa dos valores legais de **ESCAPE\_CHARACTER**, consulte a secção *IBM DB2 Universal Database: SQL Reference Volume 1* referente ao Predicado LIKE. \*)
- **ESCAPE\_LITERAL**= "", ((**ESCAPE\_CHARACTER** - ""), | ("", "")), "";
- **LETTER**= (a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | \_ | \$);
- (\* Um **IDENTIFIER** começa com uma letra (a-z, A-Z), um traço de sublinhado ou carácter de dólar, seguindo de zero ou mais letras, traços de sublinhado caracteres de dólar ou dígitos (0-9). Uma palavra-chave pode ser apenas um **IDENTIFIER** se este se encontrar entre plicas \*)
- **IDENTIFIER**= (LETTER, {LETTER | DIGIT }) - **KEYWORD** | "", LETTER, {LETTER, | DIGIT }, "";
- **ExpressionWithOptionalSortBy**= LogicalOrSetExpression, **SORTBY**, ( ,SortSpecList, ) | Expression;
- **Expression**= LogicalOrSetExpression;
- **SortSpecList**= SortSpec, {,, SortSpec};
- **SortSpec**= Expression, [ASCENDING | DESCENDING];
- **LogicalOrSetExpression** = LogicalOrSetTerm | LogicalOrSetExpression , (OR | UNION | | EXCEPT), LogicalOrSetTerm;
- **LogicalOrSetTerm**= LogicalOrSetPrimitive | LogicalOrSetTerm , (AND | INTERSECT), LogicalOrSetPrimitive;
- **LogicalOrSetPrimitive**= [NOT], SequencedValue;
- **SequencedValue**= ValueExpression;
- **ValueExpression**= Comparison;
- **Comparison**= ArithmeticExpression | Comparison , CompareOperator, ArithmeticExpression, **ESCAPE\_KEYWORD**, **ESCAPE\_LITERAL** | Comparison, CompareOperator, ArithmeticExpression | Comparison, [NOT], IN, (OptionalExpressionList, );
- **ArithmeticExpression**= ArithmeticTerm | ArithmeticExpression , (+ | -), ArithmeticTerm;
- **ArithmeticTerm**= ArithmeticFactor | ArithmeticTerm , (\* | DIV | MOD), ArithmeticFactor;
- **ArithmeticFactor**= ArithmeticPrimitive | (+ | -) ArithmeticFactor;

- **ArithmeticPrimitive**= BasicExpression , OptionalPredicateList  
| PathExpression;
- **PathExpression**= Path | (/ | //), Path | BasicExpression, OptionalPredicateList ,  
(/ | //), Path;
- **Path**= Step | Path, (/ | //), Step;
- **Step**= NodeGenerator OptionalPredicateList;
- **NodeGenerator**= NameTest | @, NameTest | @NameTest,=>, NameTest | ..;
- **OptionalPredicateList**= {Predicate} ;
- **Predicate**= [ ,Expression] ;
- **BasicExpression**= Literal | FunctionName, (, OptionalExpressionList, )  
| (Expression) | ListConstructor | .;
- **FunctionName**= QName ;
- **Literal**= STRING\_LITERAL | INTEGER\_LITERAL | FLOAT\_LITERAL;
- **OptionalExpressionList**= [ExpressionList ];
- **ExpressionList**= Expression, {,, Expression};
- **ListConstructor**= [, [ListContent], ];
- **ListContent**= Expression, {,, Expression};
- **NameTest**=QName | \*;
- **QName**=LocalPart;
- **LocalPart**=IDENTIFIER;
- **CompareOperator**= = | < | <= | > | >= | != | [NOT],LIKE ;





---

## Encaminhar um documento através de um processo

As APIs de encaminhamento de documentos permite-lhe construir novas aplicações utilizando encaminhamento de documentos, ou adicionar a funcionalidade de encaminhamento de documentos às aplicações do utilizador.

O DB2 Content Manager faculta um serviço de encaminhamento de documentos integrado para o auxiliar a encaminhar documentos através de um processo empresarial. O Encaminhar documentos faculta as seguintes funções:

- Sincronização de todos os artigos num processo de encaminhamento de documentos porque as funções de encaminhamento de documentos estão incluídas em transacções do DB2 Content Manager.
- Apresentação apenas do trabalho ao qual o utilizador tem acesso.
- Pista de auditoria única que inclui registos da criação, modificação e encaminhamento de documentos.

Para obter os conceitos básicos de encaminhamento de documentos, consulte o *Manual de Administração do Sistema*. Consulte também os exemplos para mais informações sobre encaminhamento de documentos.

---

## Compreender o processo do encaminhamento de documentos

O encaminhamento de documentos consiste em processos, nós de trabalho, listas de trabalho e pacotes de trabalho. O administrador do sistema cria os nós de trabalho, os processos e as listas de trabalho através do cliente de administração do sistema.

Um processo é formado por nós de trabalho. Cada nó de trabalho do processo consiste num passo diferente do processo. O utilizador pode criar um processo que se ramifique em várias direcções. O utilizador determina qual a ramificação para a qual segue o nó de trabalho. O utilizador pode escolher numa lista de selecções possíveis definida pelo administrador do sistema. Pode definir uma saída de servidor quando define um nó de trabalho. Pode definir saídas de servidor para entrar num nó de trabalho, sair de um nó de trabalho e para notificar o utilizador quando o limite de sobrecarga é atingido. Quando é iniciado um processo, é criado um pacote de trabalho. O pacote de trabalho é o elemento de encaminhamento e contém os atributos do trabalho. Os atributos do pacote de trabalho são formados pelo PID de artigo, pela prioridade, proprietário, etc.

Os pontos de recolha são nós de trabalho com função adicional. Um pacote de trabalho num nó de ponto de recolha prossegue para o nó de trabalho seguinte do processo, assim que existir o número de artigos especificado de um tipo de artigo especificado na pasta especificada. As listas de trabalho definem os pacotes de trabalho atribuídos a um utilizador. Pode ter uma ou mais listas de trabalho. Cada lista de trabalho pode incluir um ou mais nós de trabalho. Pode especificar a ordem dos pacotes de trabalho na lista de trabalho por prioridade ou data. Pode ainda definir a ordem dos nós de trabalho na lista de trabalho.

Ao obter listas de trabalho, pode filtrar os resultados para incluir ou excluir trabalho suspenso. Os pacotes de trabalho também podem encontrar-se em estado de notificação. O estado de notificação ocorre quando os pacotes de trabalho se encontram no nó por um período superior ao tempo especificado pelo

administrador. Tenha em mente que um nó de trabalho pode fazer parte de mais do que uma lista de trabalhos. O número de pacotes devolvidos numa lista de trabalho é definido pelo administrador do sistema.

Essas operações básicas que pode executar utilizando encaminhamento de documentos incluem:

- Iniciar um processo
- Terminar um processo
- Continuar um processo
- Suspender um processo
- Retomar um processo
- Obter trabalho de uma lista de trabalho
- Obter o artigo seguinte de uma lista de trabalho
- Definir, actualizar e eliminar um processo
- Definir, actualizar e eliminar um nó de trabalho
- Definir, actualizar e eliminar uma lista de trabalho

---

## Funções de encaminhamento de documentos

Para definir processos de encaminhamento de documentos no DB2 Content Manager, é necessário utilizar o construtor gráfico de fluxos de trabalho fornecido.

### Construtor gráfico

O DB2 Content Manager inclui um novo construtor de fluxos de trabalho gráfico e centrado em conteúdos. Pode utilizá-lo para modelar os fluxos de documentos dos processos empresariais. Antes que uma definição de processos de encaminhamento de documentos possa ser utilizada para encaminhar documentos, tem de ser verificada pelo construtor gráfico de fluxos de trabalho. Consulte o *System Administration Guide* para obter informações sobre a sua utilização.

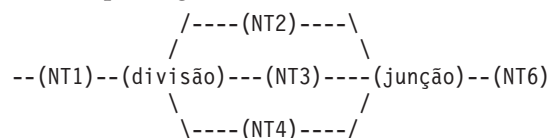
### XML de importação e exportação de processos (diagrama)

O construtor gráfico pode importar e exportar um diagrama de processo no seu formato XML. Esta função é útil quando pretende transportar um diagrama de um sistema para outro sistema. O cliente de administração do sistema também pode importar e exportar objectos de Gestão de Conteúdos no formato XML. A função é útil para transportar a definição de um processo, incluindo os objectos de definição relacionados (como, por exemplo, nós de trabalho) de um sistema para outro sistema.

### Encaminhamento paralelo

O encaminhamento paralelo é um esquema que pode replicar um pacote de trabalhos em várias cópias ligadas e enviá-los para encaminhamentos paralelos. O encaminhamento paralelo começa com um nó de divisão e termina com um nó de junção. As divisões e junções funcionam como parênteses numa expressão matemática. É uma correlação um para um. Pode ter encaminhamentos paralelos imbricados.

O exemplo seguinte ilustra um encaminhamento paralelo:



1. O pacote de trabalho deixa o nó (NT1). O nó de divisão encaminha o pacote para vários nós (NT2, NT3 e NT4).

2. Se o pacote de trabalhos for actualizado no NT2, NT3 ou NT4, as três cópias vão replicar a mesma actualização.
3. Continuando o processo, o pacote de trabalhos junta-se num único nó de junção antes de continuar para o nó de trabalho seguinte (NT6).

Nenhum encaminhamento pode violar a delimitação de um par divisão-junção. Por exemplo, a ligação a um encaminhamento entre NT4 e NT6 seria ilegal porque iria violar a delimitação entre a delimitação do par divisão-junção. No entanto, pode ligar um encaminhamento directo entre o NT2 e o NT3, já que isso não viola a regra de delimitação de encaminhamentos paralelos.

#### **Ponto de decisão**

O ponto de decisão é um nó de trabalho "sem funcionários" no processo que decide por qual dos encaminhamentos possíveis esse pacote de trabalhos pode continuar. Pode basear estas decisões em valores definidos pelo sistema, atributos de documento e variáveis do contentor de fluxos de trabalho. Sempre que um pacote de trabalho atinge um ponto de decisão, o DB2 Content Manager testa o pacote de trabalho comparando-o com expressões de decisão associadas aos encaminhamentos. Estes encaminhamentos são testados de acordo com a ordem de precedência que lhes foi atribuída, ou seja, 1, 2, 3. O pacote de trabalho prossegue até à primeira expressão que devolver um valor TRUE. É necessário um encaminhamento alternativo (precedência 0) para um ponto de decisão. O pacote de trabalhos não pode ser enviado em mais do que um encaminhamento.

Quando modela um nó de ponto de decisão, certifique-se de que definiu correctamente todos os nós antes do ponto de decisão. Caso contrário, pode ser tomada uma decisão nas variáveis do nó de trabalho definidas em cestos de trabalho, pontos de recolha e nós de subprocessos (sendo os cestos de trabalho e os pontos de recolha parte do processo), decisão essa que não pode ser alcançada antes do ponto de decisão.

Quando o utilizador apresenta a ramificação de decisão, enumera todas as variáveis do nó de trabalho para as quais o ponto de decisão pode ser definido. A lista acumula as variáveis à medida que o pacote de trabalho passa do nó de início para cada nó ligado antes do nó de ponto de decisão.

**Importante:** Pode ocorrer um erro se o nó do ponto de decisão não conseguir ler as variáveis do nó de trabalho de que necessita (por exemplo, de um cesto de trabalhos, ponto de decisão ou subprocesso). Vejamos duas situações onde isso pode ocorrer:

- Actualizar ou eliminar um cesto de trabalhos, ponto de recolha ou subprocesso sem actualizar ou eliminar a decisão correspondente no nó de ponto de decisão.
- Basear a decisão nas variáveis do nó de trabalho que ainda não tenham sido atribuídas. Por exemplo, se o pacote de trabalhos saltar determinados nós devido a encaminhamento ramificado ou paralelo.

#### **Nó da aplicação empresarial (nó LOB)**

Os nós de aplicação empresarial são fase de processamento em que o servidor de bibliotecas invoca um DLL de saída de utilizador para executar aplicações de áreas empresariais.

#### **Acção do fluxo de trabalho**

Com os objectos de acção do fluxo de trabalho, pode personalizar

aplicações do fluxo de trabalho para integrarem no DB2 Content Manager sistema externos. Por exemplo, o objecto de acção do fluxo de trabalho pode consultar uma classe DLL ou Java externa para obter e enviar por correio electrónico um documento. Cabe ao integrador da aplicação invocar, validar e implementar a classe DLL ou Java externa descrita pela acção do fluxo de trabalho. O DB2 Content Manager nunca actua em nenhuma das acções do fluxo de trabalho nos objectos. O DB2 Content Manager limita-se a manter acções de fluxo de trabalho nos objectos para que outra aplicação as leia.

#### **Lista das acções de fluxo de trabalho**

Uma `actionList` representa uma recolha de acções de fluxo de trabalho às quais pode associar nós de trabalho num processo. Quando um pacote de trabalhos atinge esse nó de trabalho, a aplicação cliente pode obter a `actionList` e apresentar todas as acções de fluxo de trabalho que o utilizador pode seleccionar.

#### **Variáveis do contentor de fluxos de trabalho**

As variáveis do contentor de fluxos de trabalho representam instâncias de pares de nome-valor (cadeias) nos pacotes de trabalho. As variáveis funcionam como objectos de ajudante que podem descrever tipos de dados e comportamento de tempo de execução esperado (por exemplo, nome, linha de comandos e apresentação ao utilizador). Estes objectos de ajudante podem formular expressões de decisão para um ponto de decisão. Podem também ser usados pelo cliente para interpretar o comportamento estimado para apresentar essas variáveis do contentor de fluxos de trabalho durante o tempo de execução. Tenha em atenção que o DB2 Content Manager não efectua qualquer verificação ou validação semânticas para os pares nome-valor em relação às correspondentes definições de variáveis do contentor.

#### **Subprocessos**

Os subprocessos são processos noutro processo. Após ter definido um processo, pode reutilizar esse mesmo processo com outra definição de processo. Os subprocessos são modelados através da utilização do nó de trabalho de um determinado tipo `DK_ICM_DR_SUB_PROCESS_NODE_TYPE`.

#### **Informações relacionadas**

-  Compreender a compatibilidade com versões anteriores à Versão 8.3
-  Compreender as classes do encaminhamento de documentos
-  Constantes de encaminhamento de documentos
-  Programar saídas de utilizador de encaminhamento de documentos

## **Compreender a compatibilidade com versões anteriores à Versão 8.3**

Se efectuar modificações a processos existentes da Versão 8.2 ou criar novos processos que tirem partido das funcionalidades da Versão 8.3 (ou seja, encaminhamento paralelo, nó de aplicações empresariais, ponto de decisão e subprocesso), tem de actualizar os clientes para a Versão 8.3. Os clientes da Versão 8.2 não funcionarão com o servidor de bibliotecas da Versão 8.3 que contém processos que tiram partido das funções de encaminhamento de documentos da Versão 8.3.

Depois de ter actualizado o servidor de bibliotecas Versão 8.2 com a Versão 8.3, pode utilizar o construtor gráfico para modificar os processos existentes que foram criados com APIs da Versão 8.2 ou com o cliente de administração do sistema Versão 8.2. O construtor gráfico apresenta graficamente os processos previamente definidos com a Versão 8.2 e dá-lhe a opção de reorganizar o esquema do diagrama. Após a reorganização do esquema do diagrama, é necessário voltar a verificar o diagrama antes de o guardar.

## Alterar a coluna DOCROUTINGFREQ na tabela ICMSTSYSCONTROL para DB2

Se utilizar comandos de base de dados, tem de alterar manualmente o valor para a verificação de frequência de actualização dos atributos do pacote de trabalho de encaminhamento do documento para o sinalizador de suspensão e de notificação.

Para definir o valor de verificação de frequência de encaminhamento de documentos na DB2, execute os seguintes passos:

1. Inicie sessão com um ID de utilizador com autoridade de administração de db2. Numa janela da linha de comandos DB2, introduza o seguinte: `db2 connect to dbname user userID using password` em que *userID* corresponde a um ID de utilizador DB2 válido.
2. Determine a definição de DOCROUTINGFREQ actual. Introduza o seguinte: `db2 select docroutingfreq from icmstsyscontrol`.  
A predefinição é 10, o que significa que o tempo mínimo de actualização dos sinalizadores de suspensão e de notificação do pacote de trabalho corresponde a 10 minutos.
3. Especifique a frequência. Introduza o seguinte: `db2 UPDATE icmadmin.ICMSTSysControl set DOCROUTINGFREQ = x` em que *LIBRARYSERVERID* = *y* em que *x* corresponde ao novo valor de verificação da frequência de encaminhamento de documentos e *y* ao valor do ID de servidor da biblioteca. O valor do ID é definido durante a instalação, sendo que o valor normal corresponde a 1.
4. Introduza o seguinte na linha de comandos: `db2 connect reset`

---

## Compreender as classes do encaminhamento de documentos

Existem 12 classes que pode utilizar para implementar a funcionalidade de encaminhamento de documentos na aplicação do utilizador.

Poderá encontrar mais informações relativas a estas classes e métodos no manual *Application Programming Reference*. As APIs de encaminhamento de documentos incluem:

### DKDocRoutingServiceICM

Esta classe faculta os métodos para encaminhar e aceder aos pacotes de trabalho e dados de contentor através de um processo. Por exemplo, `start`, `terminate`, `continue`, `suspend`, `resume`, `listWorkPackages` e `setWorkPackageContainerData`.

### DKDocRoutingServiceMgmtICM

Esta classe fornece os métodos para gerir as classes de definição de encaminhamento de documentos: `DKProcessICM`, `DKWorkNodeICM`, `DKWorkListICM`, `DKWorkflowActionICM` e `DKWorkflowActionListICM`.

A obtenção de uma cópia do objecto `DKDocRoutingServiceMgmtICM` a partir do objecto `DKDocRoutingServiceICM` é um processo mais eficaz do

que a criação de um objecto DKDocRoutingServiceMgmtICM próprio, uma vez que a API pode partilhar internamente a mesma cópia de dados de definição do fluxo de trabalho.

### **DKProcessICM**

Esta classe representa a definição de um processo que contém um recolha de encaminhamentos interligados que descrevem as fases e fluxos de um processo (além de outros atributos como, por exemplo, marcas de hora) e a descrição de um processo.

É possível criar definições de processos de encaminhamento de documentos da Versão 8.4 directamente com a API. Mas deve evitar fazê-lo tanto quanto possível. Uma vez que a criação de processos de encaminhamento de documentos com as APIs da Versão 8.4 acarreta o risco de comportamento inesperado (como, por exemplo, a criação de construções ilegais de encaminhamento paralelo) e danos no sistema, o construtor gráfico de fluxos de trabalho previne esse risco validando o processo antes de este ser guardado no servidor de bibliotecas.

A capacidade de guardar o processo depende da definição de três premissas:

#### **Estado do processo**

Indica se o processo foi verificado ou se ainda está em estado de rascunho. Pode obter este estado com o método getState em DKProcessICM. Este procedimento devolve DK\_ICM\_DR\_PROCESS\_DRAFT\_STATE (0) ou DK\_ICM\_DR\_PROCESS\_VERIFIED\_STATE (1). Embora possa alternar este estado com o método setState, deverá permitir que o construtor gráfico de fluxos de trabalho execute este procedimento para evitar danos no sistema.

#### **Definição de diagrama**

Define a aparência gráfica do processo na matriz de bytes . O comprimento máximo é 1 Mb. Os dois métodos DKProcessICM: setDiagramDefinition(byte[] diagram\_definition) e getDiagramDefinition(byte[] diagram\_definition) configuram e obtêm a definição do diagrama.

#### **Definição do encaminhamento**

Define o encaminhamento. O DKRouteListEntryICM expandido liga os seguintes tipos de nós de trabalho:

- Cesto de trabalho: DK\_ICM\_DR\_WB\_NODE\_TYPE (0)
- Nó de ponto de recolha: DK\_ICM\_DR\_CP\_NODE\_TYPE (1)
- Nó de aplicação empresarial: DK\_ICM\_DR\_BA\_NODE\_TYPE (2)

O DKRouteListEntryICM liga ainda os seguintes nós virtuais:

- Nó de divisão: DK\_ICM\_DR\_SPLIT\_NODE\_TYPE (3)
- Nó de junção: DK\_ICM\_DR\_JOIN\_NODE\_TYPE (4)
- Nó de ponto de decisão: DK\_ICM\_DR\_DP\_NODE\_TYPE (5)
- Nó de subprocesso: DK\_ICM\_DR\_SUB\_PROCESS\_NODE\_TYPE (6)
- Nó inicial: DK\_ICM\_DR\_START\_NODE\_TYPE (7)
- Nó final: DK\_ICM\_DR\_END\_NODE\_TYPE (8)

Os nós virtuais são utilizados pelo sistema para facilitar a navegação do processo. São invocados de nós virtuais porque as APIs de encaminhamento de documentos não devolvem pacotes de



trabalho para estes. No entanto, efectuar uma consulta para a localização dos pacotes de trabalho (no respectivo formato DDO em bruto).

Os métodos `DKRouteListEntryICM: setDecisionRuleExternal`, `getDecisionRuleExternal`, `setDecisionRuleInternal`, `setPrecedence` e `getPrecedence` definem a data para encaminhamentos com origem em pontos de decisão. Os pontos de `DecisionRuleExternal` são utilizados pelo construtor gráfico de fluxo de trabalhos para apresentar as regras de decisão aos utilizadores; os pontos `DecisionRuleInternal` são utilizados pelo servidor de bibliotecas para avaliar o resultado da regra de decisão durante o tempo de execução.

Os métodos `listProcessNames()` e `listProcesses()` no `DKDocRoutingServiceMgmtICM` apresentam apenas a lista dos processos verificados. Os processos em estado de rascunho não serão devolvidos.

Os métodos `add(DKProcessICM process)` e `update(DKProcessICM process)` apenas podem guardar processos que tenham passado pela verificação do construtor gráfico de fluxos de trabalho. por conseguinte, os métodos `add` e `update` só trabalham nas situações seguintes:

- O processo está no estado rascunho e a lista de encaminhamentos indefinida; mas a definição do diagrama está definida.
- O processo está no estado verificado; o diagrama está indefinido; e a lista de encaminhamentos está definida.
- O processo está no estado verificado e tanto o diagrama como a lista de encaminhamentos estão indefinidos.

Nestas situações, o construtor gráfico de fluxos de trabalho vai regular as devidas combinações dos atributos do processo antes de o guardar, para assegurar a correcção dos dados do sistema. O processo não pode ser guardado para mais nenhuma outra combinação. Consulte Tabela 29 para ver as combinações possíveis.

*Tabela 29. Combinações que os métodos `add(DKProcessICM process)` e `update(DKProcessICM process)` podem gravar*

| Estado do processo | Definição de diagrama | Definição do encaminhamento | Podem ser guardados |
|--------------------|-----------------------|-----------------------------|---------------------|
| Rascunho (0)       | Não                   | Não                         | Não                 |
| Rascunho (0)       | Não                   | Sim                         | Não                 |
| Rascunho (0)       | Sim                   | Não                         | Sim                 |
| Rascunho (0)       | Sim                   | Sim                         | Não                 |
| Verificado (1)     | Não                   | Não                         | Não                 |
| Verificado (1)     | Não                   | Sim                         | Sim                 |
| Verificado (1)     | Sim                   | Não                         | Não                 |
| Verificado (1)     | Sim                   | Sim                         | Sim                 |

## DKWorkNodeICM

Esta classe representa a definição de um nó de trabalho que apresenta os detalhes das tarefas esperadas e possíveis (como, por exemplo, `Workflow ActionList` e saída do servidor de bibliotecas) a efectuar numa determinada fase do processo.



Tenha em atenção que um subprocesso é registado com nó de trabalho do tipo DK\_ICM\_DR\_SUB\_PROCESS\_NODE\_TYPE. O subprocesso e o nó de trabalho partilham o mesmo nome.

#### **DKWorkNodeContainerDefICM**

Esta classe funciona como um ajudante que descreve o tipo de dados pretendidos e o comportamento de tempo de execução esperado (como, por exemplo, nome, "linha de comandos e apresentação ao utilizador) para dados do contentor de fluxos de trabalho encaminhado ao longo do pacote de trabalho.

Tenha em atenção que os próprios dados do contentor são instâncias de par nome-valor (cadeia de tipos). As instâncias de dados de conteúdo são componentes descendentes de instâncias de pacotes de trabalho e não instâncias desta classe DKWorkNodeContainerDefICM. O DB2 Content Manager não efectua qualquer verificação e validação semânticas para esses pares nome-valor em relação às correspondentes definições de variáveis do contentor.

#### **DKWorkFlowActionICM**

Esta classe regista detalhes (como, por exemplo, nome de ficheiro e nome de função) de uma classe DLL ou Java externa a executar no lado do cliente da API. O DB2 Content Manager não invoca automaticamente a saída especificada no objecto DKWorkFlowActionICM. A aplicação personalizada tem de invocar, validar e implementar a acção do fluxo de trabalhos.

O DB2 Content Manager predefine várias acções do fluxo de trabalhos. Estas acções ajudam, principalmente, clientes IBM (como o eClient e o pClient) a apresentar selecções de menu num nó de trabalho. Tenha em atenção que o DB2 Content Manager não executa nem cumpre nenhuma desta acções de fluxos de trabalho definidas pelo sistema.

#### **DKWorkFlowActionListICM**

Esta classe regista uma recolha de acções de fluxos de trabalho e o seu nome é devolvido com a obtenção do nó de trabalho. Isto permite que a aplicação cliente apresente as acções de fluxos de trabalho adequadas no nó de trabalho.

#### **DKWorkListICM**

Esta classe representa uma vista filtrada (com base em critérios como nó de trabalho e prioridade) para uma colecção de pacotes de trabalho que encaminha documentos aos quais o utilizador tem acesso.

#### **DKRouteListEntryICM**

Esta classe define o encaminhamento que o processo pode realizar (tal como, de ou para). Um objecto de processo (DKProcessICM) contém uma recolha de objectos de entrada de encaminhamento (DKRouteListEntryICM).

Alguns campos (regra de decisão e precedência) facilitam a construção de regras de decisão para encaminhamentos que deixam os nós de trabalho.

**Sugestão:** Pode escrever um programa que utilize as APIs para obter e examinar o valor definido pelo construtor gráfico. No entanto, o utilizador não deverá definir essas regras de decisão sozinho devido ao risco de criação de uma regra de decisão não válida.

### **DKCollectionResumeListEntryICM**

Esta classe representa uma entrada na lista de continuação para os nós de trabalho do ponto de recolha.

### **DKResumeListEntryICM**

Esta classe representa uma entrada da lista de continuação para definir quando um pacote de trabalho suspenso espera pela chegada de determinados documentos (não necessariamente no ponto de recolha) para continuar.

### **DKWorkPackageICM**

Esta classe representa um pacote de trabalho para uma tarefa de encaminhamento. Contém um identificador persistente para o documento a encaminhar e as informações sobre o estado de encaminhamento (como, por exemplo, nome do processo, nome do nó de trabalho hora de conclusão).

Quando um processo é iniciado, são criados um ou mais pacotes de trabalho em consequência da chamada da API.

## **Criar objectos de serviço de encaminhamento de documentos**

Na programação de Java e C++, a classe DKDocRoutingServiceICM faculta os serviços de encaminhamento nucleares como, por exemplo, iniciar, continuar, suspender e retomar um processo. O objecto DKDocRoutingServiceMgmtICM é uma classe auxiliar que faculta métodos para gerir DKProcessICM, DKWorkNodeICM e DKWorkListICM e os metadados requeridos para definir os objectos.

Os exemplos seguintes demonstram como criar um objecto de encaminhamento de documentos.

### **Exemplo: Java**

```
// The DKDocRoutingServiceICM class:
DKDocRoutingServiceICM routingService = new DKDocRoutingServiceICM(dsICM);

// Retrieving the DKDocRoutingServiceMgmtICM from DKDocRoutingServiceICM has
// the benefit of reducing cache footprint by sharing definition objects
// between these two classes.
DKDocRoutingServiceMgmtICM routingMgmt=
 routingService.getDocRoutingServiceMgmt();
```

### **Exemplo: C++**

```
// The DKDocRoutingServiceICM class:
DKDocRoutingServiceICM* routingService = new
 DKDocRoutingServiceICM(dsICM);

// Retrieving the DKDocRoutingServiceMgmtICM from DKDocRoutingServiceICM
// has the benefit of reducing cache footprint by sharing definition
// objects between these two classes.
//
// Deleting routingService object will also clean up memory allocated
// for routingMgmt
DKDocRoutingServiceMgmtICM* routingMgmt =
 routingService->getDocRoutingServiceMgmt();
```

Para obter um exemplo completo, consulte o exemplo SDocRoutingDefinitionCreationICM.

## Definir um novo nó de trabalho normal

Um nó de trabalho constitui uma fase na definição de um processo de encaminhamento de documentos.

Pode ser concluída para prosseguir para a fase seguinte pela aplicação cliente a qualquer momento. A aplicação é responsável pela validação dos seus próprios critérios de conclusão.

### Exemplo: Java

```
// Create new Work Node Object.
DKWorkNodeICM workNode1 = new DKWorkNodeICM();
//Choose a Name that is 15 characters or less length
workNode1.setName("S_fillClaim");
//Choose a Description for more information than the name.
workNode1.setDescription("Claimant Fills Out Claim");
// Sets the value of the maximum time that an item can spend at this
// work node (in minutes).
workNode1.setTimeLimit(100);
// Specify the threshold that activates overload user
// exit when the number of work packages at this node
// is greater than the threshold.
// Note that the number of workpackages at this worknode
// is not limited to the threshold.
workNode1.setOverloadLimit(200);
workNode1.setOverloadUserDll("C:\\USEREXIT\\exitOverload.dll");
workNode1.setOverloadUserFunction("callOverload");

// Set the type to be a regular work node
workNode1.setType(0);

//Add the new work node definition to the document routing
//managment object
routingMgmt.add(workNode1);
```

### Exemplo: C++

```
// Create new Work Node Object.
DKWorkNodeICM* workNode1 = new DKWorkNodeICM();
// Choose a Name that is 15 characters or less length
workNode1->setName("ValidateCreditCard");
// Choose a Description for more information than the name.
workNode1->setDescription("Buyer's credit card is validated with
the credit card agency");

// Sets the value of the maximum time that an item can spend at
//this work node (in minutes).
workNode1->setTimeLimit(100);

// Specify the threshold that activates overload user
// exit when the number of work packages at this node
// is greater than the threshold.
// Note that the number of workpackages at this worknode
// is not limited to the threshold.
workNode1->setOverloadLimit(200);
workNode1->setOverloadUserDll("C:\\USEREXIT\\exitOverload.dll");
workNode1->setOverloadUserFunction("callOverload");

// Set the type to be a regular work node
workNode1->setType(0);

//Add the new work node definition to the document routing
//management object
```

```
routingMgmt->add(workNode1);

// Free memory. This object will no longer be needed.
delete(workNode1);
```

Para obter um exemplo completo relativo à criação de nós de trabalho, consulte o exemplo `SDocRoutingDefinitionCreationICM`.

## Enumerar nós de trabalho

Se não for solicitado qualquer parâmetro de tipo de nó de trabalho, todos os nós de trabalho do tipo cesto de trabalhos, ponto de recolha e nó de aplicação empresarial são enumerados por predefinição.

O método `listWorkNodeNames` enumera os nomes dos nós de trabalho no servidor de bibliotecas.

O método `listWorkNodes` devolve uma recolha de objectos `DKWorkNodeICM` que representam nós de trabalho no servidor de bibliotecas.

### Exemplo: Java

```
// Obtain the document routing management object.
// Obtain the Routing Management object.
DKDocRoutingServiceMgmtICM routingMgmt = new
 DKDocRoutingServiceMgmtICM(dsICM);

// Obtain all Work Nodes in the System.
dkCollection workNodes = routingMgmt.listWorkNodes();
System.out.println("Work Nodes in System:
 (" + workNodes.cardinality() + ")");
dkIterator iter = workNodes.createIterator();
while(iter.more())
{
 DKWorkNodeICM workNode = (DKWorkNodeICM) iter.next();
 if(workNode.getType()==0)
 System.out.println(" Normal Node -
 " + workNode.getName() + ":
 " + workNode.getDescription());
 else
 System.out.println(" Collection Pt -
 " + workNode.getName() + ":
 " + workNode.getDescription());
}
```

### Exemplo: C++

```
// Obtain the document routing management object.
DKDocRoutingServiceMgmtICM* routingMgmt =
 new DKDocRoutingServiceMgmtICM(dsICM);

// Obtain the collection containing all the
//work nodes in the system.
dkCollection* workNodes = routingMgmt->listWorkNodes();
if (workNodes && (workNodes->cardinality()>0))
{
 cout << "Work Nodes in System:
 (" << workNodes->cardinality() << ")"
 << endl;
 dkIterator* iter = workNodes->createIterator();
 while(iter->more())
 {
 DKWorkNodeICM* workNode = (DKWorkNodeICM*)iter->next()
 ->value();
 if(workNode->getType()==0)
```

```

 {
 cout << " Normal Node - "
 << workNode->getName() << ": " <<
 workNode->getDescription() << endl;
 }
 else
 {
 cout << " Collection Pt - "
 << workNode->getName() << ": " <<
 workNode->getDescription() << endl;
 }
 delete(workNode);
 }
 delete(iter);
 delete(workNodes);
}
delete(routingMgmt);

```

Para obter mais informações relativas a enumerar nós de trabalho, consulte o exemplo SDocRoutingListingICM.

## Definir um novo ponto de recolha

Um ponto de recolha é um nó de trabalho que aplica a disponibilidade do documento como critérios de conclusão (especificamente aplicáveis a pastas de encaminhamento). Num ponto de recolha, pode especificar os requisitos a cumprir para que o processo possa continuar ou avançar de um local no sistema.

Em alternativa, pode invocar o `continueProcess` para forçar o pacote de trabalho a sair do ponto de recolha sem cumprir os requisitos de disponibilidade do documento.

### Exemplo: Java

```

// Create a new Work Node Object. This will be
//the collection point
DKWorkNodeICM collectionPoint = new DKWorkNodeICM();

// Choose a Name with 15 characters or less.
collectionPoint.setName("S_gatherAll");

// Choose a Description for more information than the name.
collectionPoint.setDescription("Gather Claim,Police
Report,Policy,& Photos");

// Sets the value of the maximum time that an
// item can spend at this work node (in minutes).
collectionPoint.setTimeLimit(100);
// Specify the threshold that activates overload user
// exit when the number of work packages at this node
// is greater than the threshold.
// Note that the number of workpackages at this worknode
// is not limited to the threshold.
collectionPoint.setOverloadLimit(200);
collectionPoint.setOverloadUserDll
("C:\\USEREXIT\\exitOverload.dll");
collectionPoint.setOverloadUserFunction("callOverload");
// Set the type of node to be a collection point.
collectionPoint.setType(1);
// Create the "Resume" List, which is the list of document types
//that the process must wait for before moving on to the next node.
//A list will be created to hold "resume entries"
//which are descriptions
//of requirements that must be met before the process can move on.
// Create a List/Collection to hold all Resume Entries.

```

```

dkCollection resumeList = new DKSequentialCollection();
// Create as many requirements, or "Resume List Entries", which
//specify what Item Types it must wait for. The process cannot
//pass this collection point unless the specified number of Item
//(DDO) of the specified Item Type reaches this collection point.
DKCollectionResumeListEntryICM resumeRequirement = new
 DKCollectionResumeListEntryICM();
// Set the Item Type Name Folder Item that is being routed.
resumeRequirement.setFolderItemTypeName("S_simple");
// Make the collection wait for an Item of
//the specified Item Type
//to be added to the folder before proceeding.
resumeRequirement.setRequiredItemTypeName("S_autoClaim");
// Specify the number of Items of the specified
//Item Type that it must wait for.
resumeRequirement.setQuantityNeeded(1);
// Add the requirement (Entry) to the List of
//Requirements (Resume List).
resumeList.addElement(resumeRequirement);
resumeRequirement = new DKCollectionResumeListEntryICM();
resumeRequirement.setFolderItemTypeName("S_simple");
resumeRequirement.setRequiredItemTypeName("S_policeReport");
resumeRequirement.setQuantityNeeded(1);
// When all requirements (resume list entries)
//have been added to the
//list of requirements (resume list),set the resume
//list in the collection point.
collectionPoint.setCollectionResumeList(resumeList);
// Add the new collection point definition to
// the document routing
// managment object
routingMgmt.add(collectionPoint);

```

## Exemplo: C++

```

// Create a new Work Node Object. This will be the collection point
DKWorkNodeICM* collectionPoint = new DKWorkNodeICM();
// Choose a Name with 15 characters or less.
collectionPoint->setName("GatherOrderDetails");
// Choose a Description for more information than the name.
collectionPoint->setDescription("Gather all the information
related to the order, shipping mechanism and shipping address");
// Sets the value of the maximum time that an item
//can spend at this work node (in minutes).
collectionPoint->setTimeLimit(100);

// Specify the threshold that activates overload user
// exit when the number of work packages at this node
// is greater than the threshold.
// Note that the number of workpackages at this worknode
// is not limited to the threshold.
collectionPoint->setOverloadLimit(200);
collectionPoint->setOverloadUserDll
("C:\\USEREXIT\\exitOverload.dll");
collectionPoint->setOverloadUserFunction("callOverload");

// Set the type of node to be a collection point.
collectionPoint->setType(1);
// Create the "Resume" List, which is the list of document types
//that the process must wait for before moving on to the next node.
//A list will be created to hold "resume entries" which are descriptions
//of requirements that must be met before the process may move on.
// Create a List / Collection to hold all Resume Entries.
dkCollection* resumeList = new DKSequentialCollection();
// Create as many requirements, or "Resume List Entries", which specify
//what Item Types it must wait for. The process cannot pass this
//collectionpoint unless the specified number of Item (DDO) of the

```

```

//specified Item Type reaches this collection point. DKCollectionResumeListEntryICM* resumeRequirement
 DKCollectionResumeListEntryICM();
// Set the Item Type Name Folder Item that is being routed.
resumeRequirement->setFolderItemTypeName("book");

// Make the collection wait for an Item of the specified Item Type to
//be added to the folder before proceeding.
resumeRequirement->setRequiredItemTypeName("AnItemType");

//Specify the number of Items of the specified Item Type that
//it must wait for.
resumeRequirement->setQuantityNeeded(1);

// Add the requirement (Entry) to List of Requirements (Resume List).
resumeList->addElement(resumeRequirement);
resumeRequirement = new DKCollectionResumeListEntryICM();
resumeRequirement->setFolderItemTypeName("book");
resumeRequirement->setRequiredItemTypeName("AnotherItemType");
resumeRequirement->setQuantityNeeded(1);
resumeList->addElement(resumeRequirement);

// continued...
// When all requirements (resume list entries) have been added to
//the list of requirements (resume list), set the resume list in the
//collection point.
collectionPoint->setCollectionResumeList(resumeList);
// Add the new collection point definition to the document routing
// managment object
routingMgmt->add(collectionPoint);

//Free the memory associated with this collection point
// Note that the resumeRequirement collection becomes
// a part of the collectionPoint object once the
// affinity between those two objects
// are established. Deleting the collectionPoint object cleans up
// memory allocated for resumeRequirement object.
delete(collectionPoint);

```

Para obter mais informações relativas à definição de pontos de recolha, consulte o exemplo `SDocRoutingDefinitionCreationICM`.

## Definir uma lista de trabalho

Uma lista de trabalho permite a um administrador ou a uma aplicação de utilizador alterar dinamicamente atribuições de trabalho sem contactar os utilizadores finais.

Uma lista de trabalho é composta por um ou mais nós de trabalho a partir dos quais um utilizador obtém uma lista de pacotes de trabalho ou o pacote de trabalho "seguinte". Um nó de trabalho pode fazer parte de mais do que uma lista de trabalho.

As listas de trabalho são definidas pelo cliente de administração do sistema ou pelas APIs. Ao utilizar o cliente de administração do sistema e as APIs, uma lista de trabalho tem de incluir um ou mais nós de trabalho. Ao definir uma lista de trabalho utilizando as APIs, se não existirem quaisquer nós de trabalho incluídos na lista de trabalho, é apresentada uma excepção `DK_ICM_MSG_EMPTY_WORK_LIST`. Ao definir uma lista de trabalho utilizando o cliente de administração do sistema, o utilizador não pode guardar a caixa de diálogo da lista de trabalho, salvo se existirem um ou mais nós de trabalho incluídos na lista de trabalho.



Contudo, o servidor da biblioteca suporta uma lista de trabalho sem quaisquer nós de trabalho. Isto pode ocorrer quando um nó de trabalho é eliminado. Quando um nó de trabalho é eliminado, o servidor da biblioteca remove o nó de trabalho de qualquer lista de trabalho na qual esteja definido. Se uma lista de trabalho não incluir qualquer nó de trabalho, o servidor da biblioteca devolve todos os pacotes de trabalho do sistema ao obter a lista de trabalho.

### Exemplo: Java

```
// Create a new worklist.
DKWorkListICM workList = new DKWorkListICM();
// Choose a name of 15 characters or less.
workList.setName("S_fillClaimWL");
workList.setDescription("worklist Covering Fill/Submit
 Claim Work Node.");
//Specify that work packages returned by the work
// list will be sorted by time
workList.setSelectionOrder
(DKConstantICM.DK_ICM_DR_SELECTION_ORDER_TIME);
//Specify that the work packages returned
//will be the one that are not in the suspend state
workList.setSelectionFilterOnSuspend
(DKConstantICM.DK_ICM_DR_SELECTION_FILTER_NO);

//Specify that work packages returned are
//not the ones in the notify state
workList.setSelectionFilterOnNotify
(DKConstantICM.DK_ICM_DR_SELECTION_FILTER_NO);
// Specify that at most 100 work packages should be listed in
// this worklist
workList.setMaxResult(100);
String[] wnNames = {"S_fillClaim"};
workList.setWorkNodeNames(wnNames);

//Add the new worklist definition to the document
//routing management object
routingMgmt.add(workList);
```

### Exemplo: C++

```
// Create a new worklist.
DKWorkListICM* workList = new DKWorkListICM();
//Choose a name of 15 characters or less.
workList->setName("ValidateWorkList");
workList->setDescription("worklist Covering the
 credit card validation work node.");

//Specify that work packages returned by the worklist
//will be sorted by time
workList->setSelectionOrder(DK_ICM_DR_SELECTION_ORDER_TIME);

//Specify that the work packages returned will
//be the one that are not in the suspend state
workList->setSelectionFilterOnSuspend
(DK_ICM_DR_SELECTION_FILTER_NO);
//Specify that work packages returned are
//not the ones in the notify state
workList->setSelectionFilterOnNotify
(DK_ICM_DR_SELECTION_FILTER_NO);
//Specify that at most 100 work packages
//should be listed in this worklist
workList->setMaxResult(100);

DKString* wnNames = new DKString[1];
wnNames[0] = "ValidateCreditCard";
//Add the work node to the worklist
```

```
workList->setWorkNodeNames(wnNames,1);

//Add the new worklist definition to the document
//routing management object
routingMgmt->add(workList);

//Free the memory associated with this worklist
delete(workList);
```

Para obter mais informações sobre a definição de listas de trabalho, consulte o exemplo SDocRoutingDefinitionCreationICM.

## Enumerar listas de trabalho

O método listWorkListNames enumera os nomes das listas de trabalho aos quais o utilizador que iniciou a sessão tem acesso. O método listWorkLists devolve uma recolha de objectos DKWorkListICM que representam listas de trabalho no servidor de bibliotecas.

### Exemplo: Java

```
// Obtain the document routing management object.
// Obtain the Routing Management object.
DKDocRoutingServiceMgmtICM routingMgmt =
 new DKDocRoutingServiceMgmtICM(dsICM);
// Obtain all Work Lists in the System.
dkCollection workLists = routingMgmt.listWorkLists();
System.out.println("Work Lists in System: (" +workLists.cardinality()+")");
dkIterator iter = workLists.createIterator();
while(iter.more())
{
 DKWorkListICM workList = (DKWorkListICM) iter.next();
 System.out.println(" - " +workList.getName()+":
 "+workList.getDescription());
}
```

### Exemplo: C++

```
dkCollection* workLists = routingMgmt->listWorkLists(); // Obtain all
// Work Lists in the System.

if (workLists && (workLists->cardinality()>0))
{
 cout<<"Work Lists in System: ("<< workLists->cardinality()<<)"<<endl;
 dkIterator* iter = workLists->createIterator();
 while(iter->more()){
 DKWorkListICM* workList = (DKWorkListICM*) iter->next()->value();
 cout << " - " << workList->getName() << ": "
 << workList->getDescription() << endl;
 delete(workList); // Free Memory
 }
 delete(iter); // Free Memory
 delete(workLists);
}
```

Para obter um exemplo completo, consulte o exemplo SDocRoutingListingICM.

## Definir um novo processo e encaminhamento associado

Embora possa criar definições de processos de encaminhamento de documentos com APIs, deverá evitar este procedimento devido ao risco de danos no sistema provocados por comportamentos inesperados (como, por exemplo, a criação de uma construção ilegal de encaminhamento paralelo). O construtor gráfico previne isso, avaliando um processo antes de o guardar no servidor de bibliotecas.

Um processo de encaminhamento de documentos consiste nos encaminhamentos definidos que um pacote de trabalhos segue. Vários processos de encaminhamento podem reutilizar os mesmos nós e o utilizador também pode utilizar vários encaminhamentos entre nós.

### Exemplo: Java

```
// Create a new Process Definition
DKProcessICM process = new DKProcessICM();
process.setName("S_claimProcess");
process.setDescription("Process for an Insurance Claim");

// Define all possible Routes.

// Create a list of all possible routes between nodes.
dkCollection routes = new DKSequentialCollection();

// Connect the Work Nodes using Route List Entries. A simple route
//between two work nodes is specified by associating a 'From' work
//node and a 'To' work node.
// A Route List Entry simply connects two nodes with an implied direction.
// Multiple routes may exist between nodes.
A specific route may be selected
// by a user-defined "selection" keyword.Examples might be "Continue", "Go",
// "Accept", "Reject", "Complete", etc.
// Create a new connection between two nodes.
DKRouteListEntryICM nodeRoute = new DKRouteListEntryICM();

// Every process must start with the start node.
nodeRoute.setFrom(DKConstantICM.DK_ICM_DR_START_NODE);
nodeRoute.setTo("S_fillClaim");
// Choose any user-defined name for an action that will make the
//transition from the 1st node to the second
nodeRoute.setSelection("Continue");

// Add the individual route to the collection of all possible routes.
routes.addElement(nodeRoute);
nodeRoute = new DKRouteListEntryICM();
nodeRoute.setFrom("S_fillClaim");
nodeRoute.setTo("S_gatherAll");
// Choose any user-defined name for an action that will make the
// transition take place.
nodeRoute.setSelection("Continue");

// Add the individual route to the collection of all possible routes. routes.addElement(nodeRoute);
nodeRoute = new DKRouteListEntryICM();
nodeRoute.setFrom("S_gatherAll");
nodeRoute.setTo(DKConstantICM.DK_ICM_DR_END_NODE);

// Choose any user-defined name for an action that will make
//the transition take place.
nodeRoute.setSelection("Complete");
// Add the individual route to the collection of all possible routes. routes.addElement(nodeRoute);
// Set the route in the process.
process.setRoute(routes);
// Add the process to the routing Management.
routingMgmt.add(process);
```

### Exemplo: C++

```
//A document routing process is the defined routes that a work
//package being routed will follow. Multiple routing processes may
//re-use the same nodes and multiple routes between nodes may be used.

// Create a new Process Definition
DKProcessICM* process = new DKProcessICM();
process->setName("Buy_Book");
```

```

process->setDescription("Purchase a book online");
// Define all possible Routes.

// Create a list of all possible routes between nodes.
dkCollection* routes = new DKSequentialCollection();

// Connect the Work Nodes using Route List Entries. A simple route
// between two work nodes is specified by associating a 'From' work node
// and a 'To' work node.
// A Route List Entry simply connects two nodes with an implied direction.
// Multiple routes may exist between nodes.
// A specific route may be selected by a user-defined "selection"
// keyword.Examples might be "Continue", "Go",
// "Accept", "Reject", "Complete", etc.

// Create a new connection between two nodes.
DKRouteListEntryICM* nodeRoute = new DKRouteListEntryICM();

// Every process must start with the start node.
nodeRoute->setFrom(DK_ICM_DR_START_NODE);
nodeRoute->setTo("ValidateCreditCard");
//Choose any user-defined name for an action that
// will make the transition from the 1st node to the 2nd
nodeRoute->setSelection("Continue");

// Add the individual route to the collection
// of all possible routes.
routes->addElement(nodeRoute);

nodeRoute = new DKRouteListEntryICM();
nodeRoute->setFrom("ValidateCreditCard");
nodeRoute->setTo("GatherShippingDetails");

// Choose any user-defined name for an action that
// will make the transition take place.
nodeRoute->setSelection("Continue");

// Add the individual route to the collection of all possible routes.
routes->addElement(nodeRoute);
nodeRoute = new DKRouteListEntryICM();
nodeRoute->setFrom("GatherOrderDetails");
nodeRoute->setTo(DK_ICM_DR_END_NODE);

// Choose any user-defined name for an action that
//will make the transition take place.
nodeRoute->setSelection("Complete");

// Add the individual route to the collection of all possible routes. routes->addElement(nodeRoute);
// Set the route in the process.
process->setRoute(routes);

// Add the process to the routing Management.
routingMgmt->add(process);

delete(process);

```

Para ver o exemplo completo, consulte o exemplo  
SDocRoutingDefinitionCreationICM.

## Iniciar um processo de encaminhamento de documentos

O método `startProcess` iniciar um processo com o nome, PID de artigo, prioridade e nome de proprietário especificado. Devolve um PID do pacote de trabalhos criado por este método.

O pacote de trabalhos começa no primeiro nó de trabalhos desse processo.

O exemplo seguinte mostra como iniciar um processo de encaminhamento de documentos.

### Exemplo: Java

```
//First create a document or folder that will be routed.
//An item type of name "s_simple" must be pre-defined before a
// DDO of that name can be created.

DKDDO ddoFolder = dsICM.createDDO("S_simple", DKConstant.DK_CM_FOLDER);

// Save the created folder to the persistent datastore.
ddoFolder.add();

//Create the core document routing service object.
DKDocRoutingServiceICM routingService = new DKDocRoutingServiceICM(dsICM);

//Start a process with the name "S_claimProcess" which must be pre-defined.
// The PID string of the work package as a result of the startProcess API
// call is returned.
// Note that the PID string may be an empty string if multiple
// workpackages are created by parallel routing.
String workPackagePidStr = routingService.startProcess
 ("S_claimProcess", ddoFolder.getPidObject().pidString(),1,"icmadmin");
```

### Exemplo: C++

```
//First create a document or folder that will be routed.
//An item type of name "book" must be pre-defined before a DDO of
// that name can be created.
DKDDO* ddoFolder = dsICM->createDDO("book", DK_CM_FOLDER);

// Save the created folder to the persistent datastore.
ddoFolder->add();

//Set the priority for this document routing process
int Priority = 1;

//Create the core document routing service object.
DKDocRoutingServiceICM* routingService = new DKDocRoutingServiceICM(dsICM);

//Start a process with the name "Buy_Book" (which must be pre-defined.
// The PID string of the work package as a result of the startProcess API
// call is returned.
// Note that the PID string may be an empty string if multiple
// workpackages are created by parallel routing.
DKString workPackagePidStr=routingService->startProcess("Buy_Book",
 ((DKPidICM*)
 ddoFolder->getPidObject())->pidString(), Priority, "icmadmin");
```

Para obter um exemplo completo, consulte o exemplo SDocRoutingProcessingICM.

## Terminar um processo

Pode terminar um processo explicitamente antes que este alcance o nó final, especificando o respectivo PID de pacote de trabalho no método `terminateProcess`. Quando termina um processo, todos os pacotes de trabalhos no processo incluindo pacotes de trabalho em encaminhamentos paralelos e de subprocessos) são removidos do sistema.

O método também verifica no artigo referenciado no pacote de trabalhos se foi dada saída do artigo.

### Exemplo: Java

```
routingService.terminateProcess(workPackagePidStr);
```

### Exemplo: C++

```
routingService->terminateProcess(workPackagePidStr);
```

Para obter mais informações relativas a terminar um processo, consulte o exemplo SDocRoutingProcessingICM.

## Continuar um processo

O método `continueProcess()` encaminha o artigo referenciado pelo PID de artigo no pacote de trabalho especificado do nó de trabalho actual para o nó de trabalho seguinte, que é determinado pela selecção.

O pacote de trabalho especificado é removido do servidor de bibliotecas, e é criado um novo pacote de trabalho para o proprietário especificado. O artigo referido pelo PID de artigo continua com saída dada se tiver sido dada saída do mesmo.

O PID do novo pacote de trabalho é devolvido. Na API de Java, é devolvida uma cadeia nula, se o processo tiver terminado ou continuado para um nó de divisão (encaminhamentos paralelos). Na API de C++, é devolvida uma cadeia vazia se o processo tiver terminado ou continuado para um nó de divisão (encaminhamentos paralelos).

No exemplo de código a seguir, o nome da selecção que irá causar a transição do nó de trabalho actual para o nó de trabalho seguinte é "Continue" ("Continuar"). Tenha em atenção que a cadeia de PID do pacote de trabalho do actual pacote de trabalho é especificada na chamada do método. A chamada do método devolve a cadeia de PID do novo pacote de trabalho ou a cadeia vazia.

### Exemplo: Java

```
workPackagePidStr = routingService.continueProcess
(workPackagePidStr, "Continue", "icmadmin");
```

### Exemplo: C++

```
char * userName = "icmadmin";

workPackagePidStr = routingService->continueProcess(workPackagePidStr,
"Continue", userName);
```

Para obter um exemplo completo, consulte o exemplo SDocRoutingProcessingICM.

## Suspender um processo

O método `suspendProcess` pode suspender uma instância de uma pacote de trabalhos de encaminhamento de documentos durante um determinado tempo (em minutos) por uma determinada lista de continuação. O método define para `true` o sinalizador de suspensão do pacote de trabalho especificado.

A duração especifica durante quanto tempo é mantido o sinalizador de suspensão como `true`. A lista de continuação consiste num conjunto de requisitos que instruem uma pasta a aguardar a chegada de determinados tipos e quantidades de artigos (especificados numa recolha sequencial de objectos DKResumeListEntryICM). Quando este período expira ou quando a lista de continuação é concluída, o sistema repõe o sinalizador `suspendState` do pacote de

trabalho de true (1) para false (0). Pode também forçar o pacote de trabalhos a passar prematuramente para o nó de trabalho seguinte, invocando a API `continueProcess`.

Suspender um pacote não afecta o estado de outros pacotes do mesmo processo.

A API `suspendProcess` não diz respeito a processos e módulos num ambiente de programação. O módulo ou processo no ambiente de tempo de execução de C++ não irá ser parado.

O campo `DOCROUTINGUPDATE` na tabela de controlo do sistema controla o tempo previsto para reavaliar o sinalizador de suspensão num pacote de trabalhos. O intervalo de tempo predefinido corresponde a 10 minutos.

### Exemplo: Java

```
dkCollection requirements = new DKSequentialCollection();
//Process will be suspended for 2 minutes.
routingService.suspendProcess(workPackagePidStr, 2, requirements);
```

### Exemplo: C++

```
dkCollection * requirements = new DKSequentialCollection();
//If no requirements are to be provided and the process is only to be
//suspended for a fixed period of time, the user can also pass in a
//NULL collection to this method.
//dkCollection * requirements = NULL;

//Process will be suspended for 2 minutes.
routingService->suspendProcess(workPackagePidStr, 2, requirements);
delete(requirements);
```

Para obter um exemplo completo, consulte o exemplo `SDocRoutingProcessingICM`.

## Retomar um processo

A continuação redefine o sinalizador do pacote de trabalhos como false e devolve o processo ao funcionamento normal. Não são efectuados quaisquer encaminhamentos ou retiradas do artigo de trabalho associado.

Um processo no estado suspenso pode prosseguir de três formas:

- Implicitamente depois do prazo especificado expirar.
- Implicitamente após o cumprimento dos requisitos definidos.
- Explicitamente através do método `resumeProcess`. Este método continua o pacote de trabalhos antes da duração especificada ter expirado ou antes da lista de continuação ter sido cumprida.

### Exemplo: Java

```
routingService.resumeProcess(workPackagePidStr);
```

### Exemplo: C++

```
routingService->resumeProcess(workPackagePidStr);
```

Para obter mais informações relativas à continuação de um processo, consulte o exemplo `SDocRoutingProcessingICM`.



## Enumerar cadeias de identificador persistente de pacotes de trabalho numa lista de trabalhos

Com base na definição de administração do sistema relativa à lista de trabalho, o campo de proprietário no método `listWorkPackagePidStrings` pode devolver os PIDs de pacotes de trabalho numa lista de trabalho específica.

*Tabela 30. Resultados da API `listWorkPackagePidStrings` com base no proprietário*

| Definição de administração do sistema para uma lista de trabalho | proprietário=vazio                                                                       | proprietário=ID de utilizador que iniciou sessão                                         | proprietário=ID de utilizador que não o que iniciou sessão                                                                                                     |
|------------------------------------------------------------------|------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| O filtro do proprietário não está seleccionado                   | Devolve todos os pacotes de trabalho na lista de trabalho.                               | Devolve todos os pacotes de trabalho na lista de trabalho.                               | Devolve todos os pacotes de trabalho na lista de trabalho.                                                                                                     |
| O filtro do proprietário está seleccionado                       | Devolve pacotes de trabalho do ID de utilizador que iniciou sessão na lista de trabalho. | Devolve pacotes de trabalho do ID de utilizador que iniciou sessão na lista de trabalho. | Devolve pacotes de trabalho do ID de utilizador especificado na lista de trabalho se o privilégio <code>ICM_PRIV_ITEM_GET_ASSIGN_WORK</code> estiver definido. |

### Exemplo: Java

O exemplo de código seguinte mostra como enumerar as cadeias de PID de todos os pacotes de trabalho numa lista de trabalhos especificada.

```
String[] workPackagePIDs =
 routingService.listWorkPackagePidStrings(workListName,processOwner);

// Print Work Package PIDs
System.out.println("Work Packages in worklist: (+workPackagePIDs.length+));
for(int i=0; i< workPackagePIDs.length; i++)
 System.out.println(" - PID: +workPackagePIDs[i]);
```

### Exemplo: C++

O exemplo de código seguinte mostra como enumerar as cadeias de PID de todos os pacotes de trabalho numa lista de trabalhos especificada.

```
long arraySize = -1; // Size to be set by the API.
DKString* workPackagePIDs = routingService->
 listWorkPackagePidStrings("workListName",processOwner,arraySize);

// Print Work Package PIDs
cout << "Work Packages in worklist: (" << arraySize << ")" << endl;
for(int i=0; i< arraySize; i++)
 cout << " - PID: " << workPackagePIDs[i] << endl;

delete[] workPackagePIDs; // Free Memory
```

Para obter um exemplo completo, consulte o exemplo `SDocRoutingListingICM`.

## Obter informações relativas ao pacote de trabalho

Um pacote de trabalho contém todas as informações necessárias sobre um processo e o artigo que o transporta. Quando uma instância de um processo de

encaminhamento de documentos se encontra a decorrer, um pacote de trabalho constitui o veículo através do qual um artigo (instância de um tipo de artigo) avança no processo de encaminhamento.

O pacote de trabalho corresponde ao objecto que uma aplicação utiliza e manipula como for necessário.

O método `retrieveWorkPackage` devolve o objecto `DKWorkPackageICM` referenciado pelo PID do pacote de trabalho especificado (`wpPidStringStr`).

### Exemplo: Java

```
//Use an established document routing service
//Specifying false in this method call makes sure that the work package
// is not checked out
DKWorkPackageICM workPackage =
 routingService.retrieveWorkPackage(workPackagePidStr,false);
System.out.println("-----");
System.out.println(" Work Package");
System.out.println("-----");
System.out.println(" Process Name: " + workPackage.getProcessName());
System.out.println(" work Node Name: " + workPackage.getWorkNodeName());
System.out.println(" Owner: " + workPackage.getOwner());
System.out.println(" Priority: " + workPackage.getPriority());
System.out.println(" User Last Moved: " + workPackage.getUserLastMoved());
System.out.println(" Time Last Moved: " + workPackage.getTimeLastMoved());
System.out.println(" Suspend State: " + workPackage.getSuspendState());
System.out.println(" Notify State: " + workPackage.getNotifyState());
System.out.println(" Notify Time: " + workPackage.getNotifyTime());
System.out.println(" Resume Time: " + workPackage.getResumeTime());
System.out.println("Work Package Pid: " + workPackage.getPidString());
System.out.println(" Item Pid: " + workPackage.getItemPidString());
```

### Exemplo: C++

```
cout << "-----" << endl;
cout << " Work Package" << endl;
cout << "-----" << endl;
cout << " Process Name: " << workPackage->getProcessName() << endl;
cout << " work Node Name: " << workPackage->getWorkNodeName() << endl;
cout << " Owner: " << workPackage->getOwner() << endl;
cout << " Priority: " << workPackage->getPriority() << endl;
cout << " User Last Moved: " << workPackage->getUserLastMoved() << endl;
cout << " Time Last Moved: " << workPackage->getTimeLastMoved() << endl;
cout << " Suspend State: " << workPackage->getSuspendState() << endl;
cout << " Notify State: " << workPackage->getNotifyState() << endl;
cout << " Notify Time: " << workPackage->getNotifyTime() << endl;
cout << " Resume Time: " << workPackage->getResumeTime() << endl;
cout << "Work Package Pid: " << workPackage->getPidString() << endl;
cout << " Item Pid: " << workPackage->getItemPidString() << endl;
```

Para obter um exemplo completo, consulte o exemplo `SDocRoutingProcessingICM`.

## Enumerar processos de encaminhamento de documentos

Os métodos `listProcessNames()` e `listProcesses()` no `DKDocRoutingServiceMgmtICM` apresentam apenas processos verificados por predefinição, excepto se as informações de estado do processo forem expressamente pedidas. Os processos em estado de rascunho não são devolvidos.

Os exemplos seguintes mostram como enumerar processos de encaminhamento de documentos.

## Exemplo: Java

```
//The listProcessNames method lists all process names in the
//Library Server, and the listProcesses method returns a collection
//of DKProcessICM objects representing a process in the Library Server.
// Obtain the document routing management object.
DKDocRoutingServiceMgmtICM routingMgmt =
 new DKDocRoutingServiceMgmtICM(dsICM);
// Obtain the list of all running document routing processes
// Obtain the Routing Management object.
DKDocRoutingServiceMgmtICM routingMgmt =
 new DKDocRoutingServiceMgmtICM(dsICM);
// Obtain list of all routing processes running.
dkCollection processes = routingMgmt.listProcesses();
System.out.println("Running Processes: (" +processes.cardinality()+")");

dkIterator iter = processes.createIterator();
while(iter.more())
{
 // Move pointer to next element and obtain that next element.
 DKProcessICM proc = (DKProcessICM) iter.next();
 System.out.println(" - "+proc.getName()+" : "+proc.getDescription());
}
```

## Exemplo: C++

```
// Obtain the document routing management object.
DKDocRoutingServiceMgmtICM* routingMgmt =
 new DKDocRoutingServiceMgmtICM(dsICM);

// Obtain the list of all running document routing processes
dkCollection* processes = routingMgmt->listProcesses();
if (processes && (processes->cardinality()>0))
{
 cout << "Running Processes: (" << processes->cardinality() << ")"
 << endl;
 dkIterator* iter = processes->createIterator();
 while(iter->more())
 {
 DKProcessICM* proc = (DKProcessICM*) iter->next()->value();
 cout << " - " << proc->getName() << ": "
 << proc->getDescription() << endl;
 delete(proc);
 }
 delete(iter);
 delete(processes);
}
delete(routingMgmt);
```

Uma função de impressão é facultada no exemplo SDocRoutingListingICM.

### Informações relacionadas



Compreender a compatibilidade com versões anteriores à Versão 8.3

## Implementar encaminhamento ad hoc

O cliente de administração do sistema pode ser utilizado para configurar os nós de trabalho, processos e listas de trabalho.

De seguida é apresentado um procedimento de exemplo de encaminhamento ad hoc.

1. Crie dois nós de trabalho, N1 e N2, por exemplo.
2. Crie dois processos de um nó, P1 e P2, de forma a que P1 possua um nó de trabalho, N1 e P2 tenham um nó de trabalho, N2.

**P1 terá o seguinte aspecto:**

| De:   | Acção:    | Para: |
|-------|-----------|-------|
| START | Continuar | N1    |
| N1    | Continuar | END   |

**P2 terá o seguinte aspecto:**

| De:   | Acção:    | Para: |
|-------|-----------|-------|
| START | Continuar | N2    |
| N2    | Continuar | END   |

3. Crie duas listas de trabalho, LT1 e LT2, de forma a que LT1 possua um nó de trabalho, N1 e LT2 tenham um nó de trabalho, N2.
4. No tempo de execução, inicie o processo P1 com um PID de documento (Exemplo, ABC). É criado um pacote de trabalho, o PT1. A lista de trabalhos LT1 apresenta o pacote de trabalhos PT1 no nó de trabalho N1.
5. Para mover o documento ABC do processo P1 para o processo P2, termine o pacote de trabalho PT1 e inicie o processo P2 com o mesmo documento (ABC). É criado o pacote de trabalho PT2.

A lista de trabalhos LT2 apresenta o pacote de trabalho PT2 no nó de trabalho N2.

Para ver exemplos adicionais, consulte o exemplo  
SDocRoutingDefinitionCreationICM.

## Consultas exemplo de encaminhamento de documentos

O aperfeiçoamento da pesquisa em consultas de documentos pode proporcionar as informações específicas de que necessita.

### Exemplo 1

Devolve documentos de automóveis cujos pacotes de trabalhos associados estão activos (e não suspensos).

```
/Car[@VERSIONID = latest-version(.)
AND @SEMANTICTYPE = 1 AND
REFERENCEDBY/@REFERENCER => WORKPACKAGE[@SUSPENDFLAG = 0]]
```

### Exemplo 2

Devolve documentos de automóveis cujos pacotes de trabalhos associados estão no processo AccidentInvestigation.

```
/Car[@VERSIONID = latest-version(.)
AND @SEMANTICTYPE = 1 AND
REFERENCEDBY/@REFERENCER => WORKPACKAGE[@PROCESSITEMID =
/ROUTINGPROCESS[@PROCESSNAME = "AccidentInvestigation"]/@ITEMID]]
```

### Exemplo 3

Devolve documentos de automóveis cujo o nome é Honda e os pacotes de trabalho associados aos documentos que se encontram no processo AccidentInvestigation.

```
/Car[@Name = "Honda" AND @VERSIONID = latest-version(.)
AND @SEMANTICTYPE = 1 AND REFERENCEDBY/@REFERENCER =>
WORKPACKAGE[@PROCESSITEMID = /ROUTINGPROCESS
[@PROCESSNAME = "AccidentInvestigation"]/@ITEMID]]
```

## Exemplo 4

Devolve documentos de automóveis cujos pacotes de trabalhos associados estão na fase UnderReview do processo AccidentInvestigation.

```
/Car[@VERSIONID = latest-version(.) AND @SEMANTICTYPE = 1
AND REFERENCEDBY/@REFERENCER => WORKPACKAGE[@PROCESSITEMID =
/ROUTINGPROCESS[@PROCESSNAME = "AccidentInvestigation"]/@ITEMID
AND ../@WORKNODENAME = "UnderReview"]]
```

## Exemplo 5

Devolve documentos de automóveis cujos pacotes de trabalhos associados se encontram suspensos na fase UnderReview do processo AccidentInvestigation.

```
/Car[@VERSIONID = latest-version(.) AND @SEMANTICTYPE = 1
AND REFERENCEDBY/@REFERENCER => WORKPACKAGE[@PROCESSITEMID =
/ROUTINGPROCESS[@PROCESSNAME = "AccidentInvestigation"]/@ITEMID
AND ../@WORKNODENAME = "UnderReview" AND @SUSPENDFLAG = 1]]
```

## Conceder privilégios para encaminhamento de documentos

Para executar operações de encaminhamento de documentos, é necessário dispor dos privilégios apropriados. Por predefinição, um ID de utilizador administrador, com todos os privilégios, pode criar, actualizar ou eliminar nós de trabalho ou listas de trabalho de processos de encaminhamento de documentos.

Nos processos de encaminhamento de documentos do DB2 Content Manager, as listas de trabalho e os nós de trabalho são geridos como artigos. Para criar, eliminar ou actualizar estes objectos de encaminhamento de documentos, é necessário dispor dos seguintes privilégios: ItemAdd, ItemSetUserAttrs ou ItemDelete. Por predefinição, um ID de utilizador administrador, com todos os privilégios, poderá criar, actualizar ou eliminar nós de trabalho ou listas de trabalho de processos de encaminhamento de documentos.

Cada processo de encaminhamento de documentos, nó de trabalho e lista de trabalho pertence a um tipo de artigo definido pelo sistema. Estes tipos de artigo definidos pelo sistema estão associados à ACL definida pelo sistema, DocRoutingACL. A DocRoutingACL não tem quaisquer regras, mas, por predefinição, todos os utilizadores com o privilégio ItemSuperAccess pertencem a esta ACL. Deste modo, apenas os utilizadores que têm o privilégio ItemSuperAccess juntamente com o ItemAdd, que é requerido no respectivo conjunto de privilégios, como, por exemplo, ICMADMIN (ou qualquer administrador do IBM DB2 Content Manager) podem criar um novo processo, nó de trabalho ou lista de trabalho.

Para permitir que um utilizador que não tem o privilégio ItemSuperAccess crie um novo processo de encaminhamento de documentos, nó de trabalho ou lista de trabalho, é necessário adicionar o utilizador à DocRoutingACL, com um conjunto de privilégios que contenha ItemAdd. Além disso, o utilizador tem de dispor de ItemAdd no respectivo conjunto de privilégios para criar um novo objecto de encaminhamento de documentos.

Para permitir que um utilizador sem o privilégio ItemSuperAccess actualize ou elimine objectos administrativos de encaminhamento de documentos, o utilizador tem de estar na ACL desse objecto específico com um conjunto de privilégios que contenha os privilégios correctos. Para actualizar um objecto, ItemSetSysAttrs e ItemSetUserAttrs têm de ser associados ao conjunto de privilégios. Para eliminar um objecto, ItemDelete tem de ser associado ao conjunto de privilégios.

Os privilégios associados ao encaminhamento de documentos encontram-se enumerados na tabela seguinte. Os privilégios gerais dos artigos são aplicáveis a processos, nós de trabalho e listas de trabalho.

*Tabela 31. Privilégios de encaminhamento de documentos*

| Privilégio                  | Descrição                                                                                                                                                                                                      | API relacionada                                                                                       |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| ICM_PRIV_ITEM_UPDATE_WORK   | Usado para ver se o utilizador está autorizado a fazer o seguinte para um pacote de trabalho:<br>definir a prioridade<br>definir o proprietário<br>definir a lista de retoma<br>definir a duração da suspensão | suspendProcess resumeProcess<br>setWorkPackagePriority<br>setWorkPackageOwner                         |
| ICM_PRIV_ITEM_ROUTE_START   | Utilizado para verificar se o utilizador possui autorização para iniciar um processo.                                                                                                                          | startProcess                                                                                          |
| ICM_PRIV_ITEM_ROUTE_END     | Utilizado para verificar se o utilizador possui autorização para terminar um processo.                                                                                                                         | terminateProcess                                                                                      |
| ICM_PRIV_ITEM_GET_WORKLIST  | Utilizado para ver de o utilizador está autorizado a obter a contagem ou os pacotes de trabalho de uma lista de trabalho.                                                                                      | getCount listWorkPackagePidStrings                                                                    |
| ICM_PRIV_ITEM_GET_WORK      | Utilizado para verificar se o utilizador possui autorização para obter um pacote de trabalho.                                                                                                                  | getNextWorkPackagePidString<br>getNextWorkPackage<br>checkOutItemInWorkPackage<br>retrieveWorkPackage |
| ICM_PRIV_ITEM_GET_ASGN_WORK | Utilizado para verificar se o utilizador possui autorização para obter um pacote de trabalho possuído por um utilizador diferente.                                                                             | getNextWorkPackagePidString<br>getNextWorkPackage<br>getCount<br>listWorkPackagePidStrings            |
| ICM_PRIV_ITEM_ROUTE         | Utilizado para verificar se o utilizador possui autorização para encaminhar um pacote de trabalho.                                                                                                             | continueProcess                                                                                       |

## Trabalhar com listas de controlo de acessos para encaminhamento de documentos

Ao definir uma ACL para uma entidade de encaminhamento de documentos como, por exemplo, um processo, nó de trabalho e lista de trabalho, existe impacto nas operações permitidas na entidade.

O efeito das ACLs na entidade de encaminhamento de documentos do DB2 Content Manager e privilégios associados encontram-se enumerados na seguinte tabela.

*Tabela 32. Listas de controlo de acessos e encaminhamento de documentos*

| Objectos | Métodos relacionados | Privilégios               |
|----------|----------------------|---------------------------|
| Processo | startProcess         | ICM_PRIV_ITEM_ROUTE_START |

Tabela 32. Listas de controlo de acessos e encaminhamento de documentos (continuação)

| Objectos           | Métodos relacionados                                                                                                 | Privilégios                                                                         |
|--------------------|----------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|
| Nó de trabalho     | continueProcess suspendProcess<br>resumeProcess terminateProcess<br>setWorkPackagePriority<br>setWorkPackageOwner    | ICM_PRIV_ITEM_ROUTE<br>ICM_PRIV_ITEM_ROUTE_END<br>ICM_PRIV_ITEM_UPDATE_WORK         |
| Lista de trabalhos | getNextWorkPackagePidString<br>getNextWorkPackage getCount<br>listWorkPackagePidStrings<br>checkOutItemInWorkPackage | ICM_PRIV_ITEM_GET_WORK<br>ICM_PRIV_ITEM_GET_ASGN_WORK<br>ICM_PRIV_ITEM_GET_WORKLIST |

## Programar saídas de utilizador de encaminhamento de documentos

A rotina de uma saída de utilizador de encaminhamento de documentos é uma aplicação de programação personalizada (ficheiro DLL) que pode criar especificamente num nó de trabalho.

Pode definir um nó de trabalho para invocar uma determinada função num determinado ficheiro DLL nas situações seguintes:

- Quando um pacote de trabalhos é criado e iniciado no processo.
- Quando um pacote de trabalhos passa para um nó de trabalho.
- Quando um pacote de trabalhos deixa o nó de trabalho.
- Quando o ponto de recolha atinge um determinado limite de sobrecarga.

Quando o DB2 Content Manager invoca uma rotina de saída de utilizador, o utilizador pode obter um pacote de trabalho da tabela de pacotes de trabalho utilizando o ComponentID na API myExit.

Para transferir dados de pacotes de trabalho (incluindo dados do contentor) de e para a rotina de saída de utilizador, utilize ICMUSERSTRUCT, conforme definido no ficheiro do cabeçalho exemplo *IBMCMROOT /samples/server/exit/wxv2tue.h*. Pode incluir este cabeçalho ao compilar a sua própria função num ficheiro DLL, por exemplo, *WXV2UserExitSample.dll*.

Como todas as rotinas de saída de utilizador de nó de trabalho são suportadas apenas como processos síncronos (como bibliotecas partilhadas), deve minimizar o tempo de execução da aplicação. Caso contrário, o servidor de bibliotecas caduca se a transacção de execução da saída demorar muito tempo. Em alternativa, para executar a rotina de saída de utilizador num processo assíncrono, pode programar o DLL de rotina de saída para detectar um processo que continue a transacção empresarial separadamente do servidor de bibliotecas.

Ao trabalhar com rotinas de saída de utilizador de encaminhamento de documentos em sistemas operativos UNIX, certifique-se de que o ID delimitador do DB2 UDB tem autorização de execução na rotina de saída de utilizador e no directório no qual está localizada a rotina de saída de utilizador.

Pode encontrar um ficheiro exemplo C, incluindo directrizes, exemplos de utilização e documentação para a rotina de saída de utilizador no directório *IBMCMROOT /samples/server/exit/icmdruext.c*. O nome da função tem de começar pela cadeia *WXV2* para se diferenciar das funções criadas em versões anteriores ao DB2 Content Manager 8.3.



## Constantes de encaminhamento de documentos

As constantes de encaminhamento de documentos estão incluídas (definidas) em `DKConstantICM`.

Parâmetros de filtro de lista de trabalhos:

- `public final static int DK_ICM_DR_SELECTION_FILTER_NO = 0;`
- `public final static int DK_ICM_DR_SELECTION_FILTER_YES = 1;`
- `public final static int DK_ICM_DR_SELECTION_FILTER_EITHER = 2;`
- `public final static int DK_ICM_DR_SELECTION_ORDER_PRIORITY = 0;`
- `public final static int DK_ICM_DR_SELECTION_ORDER_TIME = 1;`
- `public final static int DK_ICM_DR_MAX_RESULT_ALL = 0;`

Tipos de dados de variáveis de conteúdo de fluxos de trabalhos:

- `public final static short DK_ICM_DR_WNV_TYPE_CHARACTER = 0;`
- `public final static short DK_ICM_DR_WNV_TYPE_INTEGER = 1;`
- `public final static short DK_ICM_DR_WNV_TYPE_TIMESTAMP = 2;`

Tipos de nó de trabalho:

- `public final static short DK_ICM_DR_WB_NODE_TYPE = 0;`
- `public final static short DK_ICM_DR_CP_NODE_TYPE = 1;`
- `public final static short DK_ICM_DR_SPLIT_NODE_TYPE = 2;`
- `public final static short DK_ICM_DR_JOIN_NODE_TYPE = 3;`
- `public final static short DK_ICM_DR_DP_NODE_TYPE = 4;`
- `public final static short DK_ICM_DR_SUB_PROCESS_NODE_TYPE = 5;`
- `public final static short DK_ICM_DR_BA_NODE_TYPE = 6;`

Estados de processo:

- `public final static short DK_ICM_DR_PROCESS_VERIFIED_STATE = 0;`
- `public final static short DK_ICM_DR_PROCESS_DRAFT_STATE = 1;`



---

## Compreender a busca prévia no DB2 Content Manager for z/OS

>A função de busca prévia permite transportar objectos que estejam actualmente localizados em meios mais lentos, como suportes ópticos ou de banda, para meios mais rápidos, como dispositivos de armazenamento de acesso directo (DASD - direct access storage devices) ou vice versa.

A busca prévia faz com que os objectos fiquem imediatamente disponíveis para acesso dos utilizadores. Por exemplo, a XYZ Seguros acaba de saber que vai ser auditada e que tem de fornecer os registos dos últimos sete anos. A política da empresa é migrar as apólices dos clientes que passaram para outra companhia de seguros para um suporte óptico. Nesta situação, o administrador do sistema do DB2 Content Manager tem de efectuar uma busca prévia das apólices dos clientes anteriores e restaurar essas apólices no DASD. As apólices podem, assim, ser obtidas rapidamente, caso os auditores as queiram ver.

---

### Busca prévia de objectos

Para utilizar a função de busca prévia, pode invocar as APIs DKLobICM e DKDatastoreICM na sua aplicação.

Quando invocar as APIs, especifique que pretende um determinado artigo ou, em caso de operação por lotes, uma recolha de artigos, para a busca prévia. As APIs enviam um pedido ao gestor de recursos para efectuar a busca prévia do objecto na recolha onde está localizado e colocam-no na recolha especificada como a recolha de busca prévia em determinada altura. A Figura 13 na página 254 descreve o fluxo de uma transacção de busca prévia:

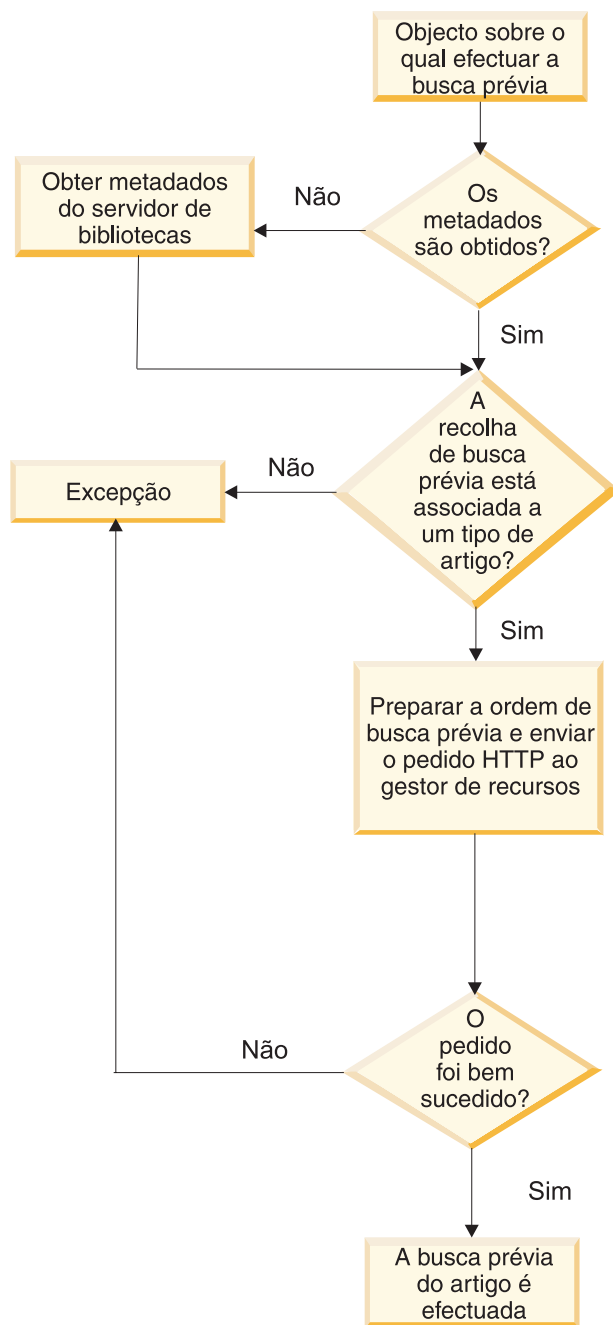


Figura 13. Busca prévia de objectos

A busca prévia no gestor de recursos é uma transacção assíncrona. Por conseguinte, a conclusão satisfatória da transacção de busca prévia consiste nas duas partes descritas abaixo.

1. A aplicação invoca o gestor de recursos com uma ordem de busca prévia. A ordem de busca prévia é processada pelo gestor de recursos, inserindo uma entrada na tabela ICMRMPREFETCH. Se esta inserção na tabela for bem sucedida, é enviado um código de devolução 0 para aquela aplicação de chamada. Nesta fase, o objecto só existe na localização original ou na recolha origem. O processo de actualização da tabela é o seguinte:
  - a. É inserida uma entrada na tabela ICMRMPREFETCH.

- b. O requesttimestamp e o statetimestamp são definidos como a marca de hora actual.
  - c. O prefetchstate é definido como INITIATED.
2. O processo em si, que se trata de copiar um objecto da origem para a recolha de busca prévia, é efectuado pelo processo assíncrono ICM MOSAP. Este processo efectua uma select na tabela ICM RMCONTROL para PREFETCHENABLED. Se este valor for definido como 1, o ICM MOSAP processa as entradas na tabela ICM RMPREFETCH, se as houver. Este processo desenrola-se da seguinte forma:
  - a. O prefetchstate é actualizado para WORKING e a statetimestamp é actualizada com a marca de hora actual.
  - b. É efectuada uma OAM\_Query para verificar se o objecto existe no OAM, pesquisando no respectivo extobjname e nome de recolha origem.
  - c. Se o objecto existir em OAM, é efectuada uma operação OAM\_Retrieve .
  - d. Uma vez obtido, é efectuado um OAM\_Store para o extobjname existente em OAM no nome da recolha de busca prévia.
  - e. Se o OAM\_Store for bem sucedido, o prefetchstate é actualizado para COMPLETE e o statetimestamp é actualizado com a marca de hora actual.
  - f. Se ocorrer uma falha, o Oamreturncode e o oamreasoncode são actualizados com códigos de erro do OAM e o prefetchstate é actualizado para FAILED.

O suporte para as APIs de busca prévia está limitado só para APIs de Java. As APIs abaixo são as APIs com que trabalha para a busca prévia de um objecto:

- DKDatastoreICM public dkCollection prefetchObjects(dkCollection prefetchColl,DKNVPair[] nvPairs) throws DKException,Exception
- DKLobICM public boolean prefetchContent(DKNVPair[] nvPairs) throws DKException, java.lang.Exception public boolean prefetchContent(DKNVPair[] nvPairs) throws DKException, java.lang.Exception

Para obter mais informações relativas a APIs, consulte Application Programming Reference.

## Definições de tabela relativas à busca prévia

Esta secção apresenta as definições de tabela relativas à busca prévia.

*Tabela 33. ICM RMCONTROL*

|                 |          |           |
|-----------------|----------|-----------|
| PREFETCHENABLED | SMALLINT | NOT NULL, |
|-----------------|----------|-----------|

*Tabela 34. ICM RMPREFETCH*

|                  |               |          |
|------------------|---------------|----------|
| ITEMID           | CHAR(26)      | NOT NULL |
| VERSIONID        | SMALLINT      |          |
| VERSIONID        | SMALLINT      |          |
| EXTOBJNAME       | CHAR(44)      | NOT NULL |
| SOURCECOLLNAME   | CHAR(44)      | NOT NULL |
| PREFETCHCOLL     | NAME CHAR(44) | NOT NULL |
| REQUESTTIMESTAMP | TIMESTAMP     | NOT NULL |
| STATETIMESTAMP   | TIMESTAMP     | NOT NULL |
| OAMRETURNCODE    | SMALLINT      |          |

Tabela 34. ICMRMPREFETCH (continuação)

| ITEMID                   | CHAR(26)     | NOT NULL |
|--------------------------|--------------|----------|
| OAMREASONCODE            | CHAR(8)      |          |
| VOLSER                   | CHAR(6)      |          |
| OPERATION                | VARCHAR(128) | NOT NULL |
| PREFETCHSTATE            | VARCHAR(128) | NOT NULL |
| PRIMARY KEY (EXTOBJNAME) |              |          |

#### ITEMID

Gerado pelo servidor de bibliotecas durante a transacção de armazenamento do objecto. Exemplo ITEMID: A1001001A03L09B64813192553

#### VERSIONID

Quando Criação de versões está activada, este valor apresenta uma determinada instância de um objecto. Exemplo VERSIONID: 1

#### EXTOBJNAME

EXTOBJNAME é a combinação de ITEMID e VERSIONID delimitados por "." e representa o único localizador de objectos do DB2 Content Manager no Object Access Method (OAM). Exemplo EXTOBJNAME: A1001001.A03L09.B64813.I92553.V001

#### SOURCECOLLNAME

É o nome da recolha em que o objecto foi originalmente armazenado e a origem na operação da busca prévia. Exemplo SOURCECOLLNAME: CLLCT001

#### PREFETCHCOLLNAME

Representa de forma geral uma recolha suportada num meio de acesso rápido, como o DASD, e é o destino da operação de busca prévia. Exemplo PREFETCHCOLLNAME: CLLCT002

#### REQUESTTIMESTAMP

É a hora a que é efectuado o pedido de busca prévia de um objecto. Exemplo REQUESTTIMESTAMP: 2003-12-09 21:48:22.778167000

#### STATETIMESTAMP

Reflecte a hora a que o PREFETCHSTATE foi modificado. Por exemplo, quando é efectuado inicialmente o pedido de busca prévia de um objecto, a REQUESTTIMESTAMP e a STATETIMESTAMP são iguais e o prefetchstate é INITIATED. Assim que começa o processo do pedido de busca prévia, o PREFETCHSTATE é actualizado para WORKING. A STATETIMESTAMP também é actualizada para reflectir a hora a que começou o processo de busca prévia. Se o processo de busca prévia for bem sucedido, o PREFETCHSTATE é actualizado para COMPLETE. Se ocorrer uma falha, o PREFETCHSTATE é actualizado para FAILED. Em qualquer dos casos a STATETIMESTAMP também é actualizada para a marca de hora actual. Exemplo STATETIMESTAMP: 2003-12-09 21:48:22.778167000

#### OAMRETURNCODE

É um valor inteiro que representa um código de devolução de uma operação OAM. A busca prévia de um objecto necessita de várias chamadas à interface OAM. Por conseguinte, o código devolvido pelo OAM indica o êxito ou o fracasso de uma determinada operação de busca prévia. Uma vez que a função de busca prévia é actualmente executada de forma assíncrona, a aplicação de chamada não tem qualquer forma de saber o estado de uma determinada operação de busca prévia, daí

que seja importante armazenar estas informações de estado de forma persistente para a posterior consulta de estados. Exemplo OAMRETURNCODE: 8

#### **OAMREASONCODE**

Faculta informações detalhadas relativas à falha no OAMRETURNCODE e é a representação de caracteres do valor hexadecimal utilizado para localizar a razão da descrição do código no *DFSMSdfp Diagnosis Reference Guide* (GY27-7618-03). Exemplo OAMREASONCODE: 2C040100

#### **VOLSER (para utilização futura)**

Representa o volume do suporte de banda onde está armazenado um determinado objecto. Este valor é incluído para efeitos de agrupamento dos pedidos de busca prévia de acordo com o volume do suporte de banda onde estão alojados para evitar “estragos” das unidades de banda.

#### **OPERATION**

Este valor indica a operação que está a ser efectuada. Actualmente, o único valor válido para a coluna é PREFETCH. A coluna da operação é para a expansão futura e, por conseguinte, irá conter outros valores no futuro. Exemplo OPERATION: PREFETCH

#### **PREFETCHSTATE**

Indica o progresso da transacção de busca prévia. A lista abaixo fornece os valores válidos para esta coluna:

##### **INITIATED**

O estado inicial quando um pedido de busca prévia é inserido na tabela ICMRMPREFETCH.

##### **WORKING**

O estado actualizado que indica que o processo de busca prévia iniciou o processamento do pedido de busca prévia de um determinado objecto.

##### **COMPLETE**

O estado actualizado que indica que o processo de busca prévia foi terminado com sucesso.

##### **FAILED**

O estado actualizado que indica que o processo de busca prévia devolveu uma condição de erro que não é zero.





---

## Trabalhar com outros servidores de conteúdos

Utilize as classes do `dkDatastore` para definir um servidor de conteúdos apropriado para os servidores de conteúdos da sua aplicação.

O servidor de conteúdos é a interface primária do DB2 Information Integrator for Content. Cada servidor de conteúdos possui uma classe de servidor de conteúdos diferente.

Para criar um servidor de conteúdos, utilize as classes `DKDatastorexx`, em que `xx` identifica o servidor de conteúdos específico. A Tabela 4 na página 33 apresenta estas classes.

*Tabela 35. Tipo de servidor e terminologia do nome da classe*

| Servidor de conteúdos                                   | Nome da classe              |
|---------------------------------------------------------|-----------------------------|
| DB2 Content Manager Versão 8.4                          | <code>DKDatastoreICM</code> |
| OnDemand                                                | <code>DKDatastoreOD</code>  |
| DB2 Content Manager for AS/400 (VisualInfo para AS/400) | <code>DKDatastoreV4</code>  |
| ImagePlus for OS/390                                    | <code>DKDatastoreIP</code>  |

Ao criar um servidor de conteúdos para um servidor de conteúdos, implemente cada uma das seguintes classes e interfaces:

### **dkDatastore**

Representa o servidor de conteúdos e gere a ligação, as comunicações e a execução de comandos do servidor de conteúdos. `dkDatastore` é uma versão abstracta da classe do gestor de consultas. Suporta o método `avaliar`.

### **dkDatastoreDef**

Utiliza os métodos para aceder a artigos armazenados no servidor de conteúdos. Também cria, enumera e elimina as suas entidades. Mantém uma recolha de `dkEntityDefs`. Um exemplo de classes concretas para esta interface é `DKDatastoreDefICM`.

### **dkEntityDef**

Utilize os métodos para aceder a informações da entidade. Também cria e elimina entidades e atributos. Os métodos desta classe suportam aceder a entidades de múltiplos níveis. Se um servidor de conteúdos não suportar subentidades, estas geram objectos `DKUsageError`. Se um servidor de conteúdos suportar entidades de vários níveis, o utilizador deve implementar métodos para sobrepor as excepções de subclasses destes servidores de conteúdos. Um exemplo de classes concretas para a interface `dkEntityDef` é `DKItemTypeDefICM`.

### **dkAttrDef**

Define métodos para aceder a informações de atributos e para criar e eliminar atributos. Um exemplo de classes concretas para `dkAttrDef` é `DKAttrDefICM`.

### dkServerDef

Define métodos de acesso a informações do servidor. Um exemplo de classes concretas para dkServerDef é DKServerDefICM:

### dkResultSetCursor

Cria um cursor de servidor de conteúdos que gere uma recolha de objectos DDO.

### dkResource

declara uma interface pública comum para dados de objectos binários de grandes dimensões (BLOB). Um exemplo de classes concretas para dkResource é DKLobICM.

As classes de definição de dados e a sua hierarquia de classes estão representadas em Figura 14:

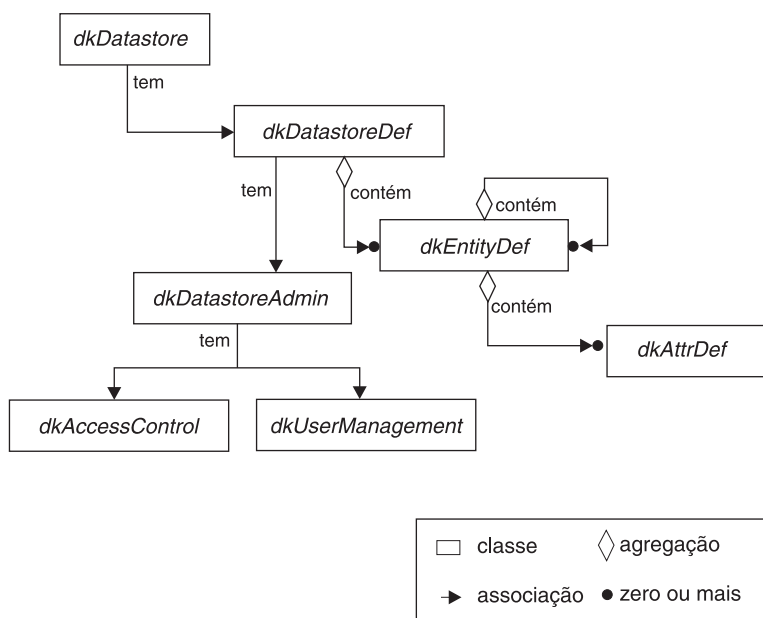


Figura 14. Hierarquia de classes de definição de dados

## Trabalhar com o OnDemand

O DB2 Information Integrator for Content fornece um conector e classes relacionadas para aceder ao conteúdo nos servidores OnDemand.

Esta acção está obsoleta no DB2 Content Manager Versão 8.4.

**Importante:** O OnDemand não suporta Net Search Extender nem Pesquisa combinada.

## Representar servidores e documentos do OnDemand

Pode representar um servidor de conteúdos do OnDemand utilizando um DKDatastoreOD numa aplicação do DB2 Information Integrator for Content e representar um documentos do OnDemand como DDO utilizando um DKDDO.

Os DDOs OnDemand contêm as seguintes informações:

- Nomes dos atributos dos documentos e os seus valores

- Dados e anotações de documentos (representados como DKParts)
- Recolha de vistas lógicas de um documento
- Dados de grupo de recurso

Os atributos de um documento OnDemand são armazenados num DKDDO como propriedades. Os segmentos e notas de um documento OnDemand são armazenados como DKParts.

Todos os outros dados de documentos (grupo e vistas de recurso, tanto fixos como lógicos), são armazenados como propriedades especiais num DDO OnDemand, em que os seguintes nomes de propriedades estão reservados para o OnDemand:

#### **DKViews**

Uma recolha de vistas lógicas

#### **DKFixedView**

Contém informações de vista fixa

#### **DKResource**

Contém dados do grupo de recurso

#### **Importante:**

1. Um administrador do DB2 Information Integrator for Content deve definir adequadamente o conector OnDemand especificando a cadeia no campo **Parâmetros adicionais** na página Parâmetros de iniciação. A cadeia deverá ter o aspecto seguinte:  
ENTITY\_TYPE=TEMPLATES;; Certifique-se de que coloca ponto e vírgula duas vezes.
2. No DB2 Information Integrator for Content Versão 8.3, DKViews, DKFixedView e DKResource substituem DKViewDataOD, DKFixedViewDataOD e DKResouceGrpOD, respectivamente.

## **Estabelecer e anular ligações ao servidor do OnDemand**

Esta secção descreve o modo como estabelecer e anular ligações ao servidor do OnDemand.

Para iniciar sessão num servidor de conteúdos do OnDemand, transmite o *nome do servidor* (por exemplo, ODServer.mycompany.com), o *ID de utilizador* e a *palavra-passe* através do método connect.

### **Exemplo: Java**

```
DKDatastoreOD dsOD = new DKDatastoreOD();
System.out.println("connecting to datastore ...");
dsOD.connect(ODServer, UserID, Password, "");
```

### **Exemplo: C++**

```
DKDatastoreOD* dsOD = new DKDatastoreOD();
cout << "connecting to datastore ..." << endl;
dsOD->connect(ODServer, UserID, Password, "");
```

Utilize o método de anulação de ligação para terminar uma sessão no servidor de OnDemand.

### Exemplo: Java

```
System.out.println("disconnecting from the datastore ...");
dsOD.disconnect();
dsOD.destroy(); // Finished with the datastore
```

### Exemplo: C++

```
cout << "disconnecting from the datastore ..." << endl;
dsOD->disconnect();
delete dsOD;
```

## Enumerar informações no OnDemand

O utilizador poderá enumerar grupos e pastas de aplicações para servidores OnDemand.

### Enumerar grupos de aplicações

Pode enumerar grupos de aplicações no OnDemand utilizando o método `listEntities()` do `DKDatastoreOD`.

O exemplo seguinte ilustra como utilizar este método.

### Exemplo: Java

```
...
pCol = (DKSequentialCollection) dsOD.listEntities();
//get application groups
pIter = pCol.createIterator();
i = 0;
while (pIter.more() == true)
{
 i++;
 agDef = (DKAppGrpDefOD)pIter.next();
 strAppGrp = agDef.getName();
 System.out.println(" app grp name[" + i + "]: " + strAppGrp);
 System.out.println(" show attributes for " + strAppGrp + " app grp - ");
 ...
}
```

### Exemplo: C++

```
// ----- Show the application groups
// ----- First get the groups
pCol = (DKSequentialCollection*)dsOD.listEntities();
pIter = pCol->createIterator();
int i = 0;
// ---- Process the list
while (pIter->more() == TRUE)
{
 i++;
 agDef = (DKAppGrpDefOD*)((void*)(*pIter->next()));
 strAppGrp = agDef->getName();
 cout << " app group name[" << i << "]: ==>" << strAppGrp << endl;
 . . .
}
```

O exemplo seguinte ilustra a obtenção de informações de atributos para cada grupo de aplicações:

### Exemplo: Java

```
...
pCol2 = (DKSequentialCollection)
dsOD.listEntityAttrs(strAppGrp);
```

```

pIter2 = pCol2.createIterator();
j = 0;

while (pIter2.more() == true)
{
 j++;
 attrDef = (DKFieldDefOD)pIter2.next();
 System.out.println(" Attribute name[" + j + "]:
+ attrDef.getName());
 System.out.println(" datastoreType: "
+ attrDef.datastoreType());
 System.out.println(" attributeOf: "
+ attrDef.getEntityName());
 System.out.println(" type: "
+ attrDef.getType());
 System.out.println(" size: "
+ attrDef.getSize());
 System.out.println(" id: "
+ attrDef.getId());
 System.out.println(" nullable: "
+ attrDef.isNullable());
 System.out.println(" precision: "
+ attrDef.getPrecision());
 System.out.println(" scale: "
+ attrDef.getScale());
 System.out.println(" stringType: "
+ attrDef.getStringType());
}

System.out.println(" " + j + "
attribute(s) listed for " +
 strAppGrp + " app grp\n");
...

```

### Exemplo: C++

```

//- Get the attributes for each of
the entities(application groups)
pCol2 = (DKSequentialCollection*)dsOD.listEntityAttrs(strAppGrp);
pIter2 = pCol2->createIterator();
int j = 0;
// ----- List the attributes
while (pIter2->more() == TRUE)
{
 j++;
 attrDef = (DKFieldDefOD*)(void*)(*pIter2->next());
 cout << "attribute name[" << j << "]: ==>"
<< attrDef->getName() << endl;
 cout << " datastore type: " <<
attrDef->datastoreType() << endl;
 cout << " attribute of: "
<< attrDef->getEntityName() << endl;
 cout << " type: " <<
attrDef->getType() << endl;
 cout << " size: " <<
attrDef->getSize() << endl;
 cout << " ID: " <<
attrDef->getId() << endl;
 cout << " precision: " <<
attrDef->getPrecision() << endl;
 cout << " scale: " <<
attrDef->getScale() << endl;
 cout << " stringType: " <<
attrDef->getStringType() << endl;
 cout << " nULLable: " <<
attrDef->isNullable() << endl;
 cout << " queryable: " <<

```

```

attrDef->isQueryable() << endl;
 cout << " updatable: " <<
attrDef->isUpdatable() << endl;
 // ----- Clean up the attribute
 delete attrDef;
}
cout << " " << j << " attribute(s)
listed for the " << strAppGrp
 << " app group\n" << endl;
// ----- Clean up the iterators and collections
if (pIter2)
 delete pIter2;
if (pCol2)
 delete pCol2;
. . .

```

## Enumerar pastas do OnDemand

Para obter uma lista de pasta num servidor de conteúdos do OnDemand, utilize a função `listSearchTemplates()`.

### Exemplo: Java

```

...
dsDef = (DKDatastoreDefOD)dsOD.datastoreDef();
pCol = (DKSequentialCollection) dsDef.listSearchTemplates();
pIter = pCol.createIterator();
i = 0;
while (pIter.more() == true)
{
 i++;
 folderName = (String)pIter.next();
 // Process the folder as appropriate
}
dsOD.disconnect();
dsOD.destroy();

```

### Exemplo: C++

```

. . .
// ----- List the folders
dsDef = (DKDatastoreDefOD*)dsOD.datastoreDef();
pCol = (DKSequentialCollection*)dsDef->listSearchTemplates();
pIter = pCol->createIterator();
i = 0;
// ----- Process the list of folders
while (pIter->more() == TRUE)
{
 i++;
 folderName = (DKString)(*pIter->next());
 cout << "folder name [" << i << "] - " << folderName << endl;
}
// ----- Disconnect
dsOD.disconnect();
. . .

```

## Obter um documento OnDemand

Num servidor de OnDemand, pode obter documentos. Também pode apresentar documentos com as respectivas partes e atributos.

### Pesquisar um documento específico

O exemplo seguinte pesquisa um grupo de aplicações (OnDemand Publications) no servidor de conteúdos de OnDemand.

### Exemplo: Java

```
DKDatastoreOD dsOD = new DKDatastoreOD();
String appgrp = "OnDemand Publications";
String SQLcmd = "where bookname LIKE 'A

DKNVPair[] parms = new DKNVPair[3];
parms[0] = new DKNVPair("APPL_GROUP", appgrp);
parms[1] = new DKNVPair("MAX_RESULTS", new String(Integer.toString(5)));
parms[2] = new DKNVPair("CONTENT", new String("ATTRONLY"));

System.out.println("executing query");
dkResultSetCursor pCur = dsOD.execute(SQLcmd,DK_CM_SQL_QL_TYPE,parms);
System.out.println("datastore executed query");
```

### Exemplo: C++

```
DKDatastoreOD* dsOD = new DKDatastoreOD();

cout << "connecting to datastore ..." << endl;
dsOD->connect(ODServer, UserID, Password, "");

DKString appgrp = "OnDemand Publications";
DKString SQLcmd = "where bookname LIKE 'A

DKNVPair parms[4];
parms[0] = DKNVPair("APPL_GROUP", appgrp);
parms[1] = DKNVPair("MAX_RESULTS", DKString(5));
parms[2] = DKNVPair("CONTENT", DKString("ATTRONLY"));
parms[3] = DKNVPair(DK_CM_PARM_END, DAny((long)0));

cout << "executing query" << endl;
dkResultSetCursor* pCur = dsOD->execute(SQLcmd,DK_CM_SQL_QL_TYPE,parms);
cout << "datastore executed query" << endl;
if (pCur != 0)
 delete pCur;
```

O exemplo seguinte pesquisa um grupo de aplicações (OnDemand Publications) e obtém os documentos devolvidos.

### Exemplo: Java

```
DKDatastoreOD dsOD = new DKDatastoreOD();
String appgrp = "OnDemand Publications";
String SQLcmd = "where bookname LIKE 'A

DKNVPair[] parms = new DKNVPair[3];
parms[0] = new DKNVPair("APPL_GROUP", appgrp);
parms[1] = new DKNVPair("MAX_RESULTS", new String(Integer.toString(5)));
parms[2] = new DKNVPair("CONTENT", new String("ATTRONLY"));

System.out.println("executing query");
dkResultSetCursor pCur = dsOD.execute(SQLcmd,DK_CM_SQL_QL_TYPE,parms);
System.out.println("datastore executed query");

while (pCur.isValid())
{
 DKDDO p = pCur.fetchNext();
 if (p != null)
 {
 String idstr = ((DKPid)p.getPidObject()).pidString();
 System.out.println(" pidString : " + idstr);
 DKPid pid = new DKPid (idstr);
 DKDDO ddoold = p;
 short id, docType = 0;
 if ((id = ddoold.propertyId(DK_CM_PROPERTY_ITEM_TYPE)) > 0)
 docType = ((Short)ddoold.getProperty(id)).shortValue();
```



```

if (docType == DK_CM_DOCUMENT)
{
 System.out.println("create a new DDO with a cloned pid to retrieve!");
 p = dsOD.createDDO(ddoold.getObjectType(), DK_CM_DOCUMENT);
 p.setPidObject(pid);
 try
 {
 dsOD.retrieveObject((dkDataObject)p);
 }
 catch (DKException exc)
 {
 System.out.println("Exception name " + exc.name());
 System.out.println("Exception message " + exc.getMessage());
 System.out.println("Exception error code " + exc.errorCode());
 System.out.println("Exception error state " + exc.errorState());
 exc.printStackTrace();
 }
}
}
}
pCur.destroy(); // Finished with the cursor

```

### Exemplo: C++

```

DKDatastoreOD* dsOD = new DKDatastoreOD();
DKString appgrp = "OnDemand Publications";
DKString SQLcmd = "where bookname LIKE 'A'

DKNVPair parms[4];
parms[0] = DKNVPair("APPL_GROUP", appgrp);
parms[1] = DKNVPair("MAX_RESULTS", DKString(5));
parms[2] = DKNVPair("CONTENT", DKString("ATTRONLY"));
parms[3] = DKNVPair(DK_CM_PARM_END, DKAny((long)0));

cout << "executing query" << endl;
dkResultSetCursor* pCur = dsOD->execute(SQLcmd,DK_CM_SQL_QL_TYPE,parms);
cout << "datastore executed query" << endl;

if (pCur != 0)
{
 while (pCur->isValid())
 {
 DKDDO* p = pCur->fetchNext();
 DKDDO* ddoold = p;
 DKString pidStr = ((DKPid*)ddoold->getPidObject())->pidString();
 DKPid* pid = new DKPid(pidStr);
 short id, docType = 0;
 DKAny a;
 if ((id = ddoold->propertyId(DK_CM_PROPERTY_ITEM_TYPE)) > 0)
 {
 a = ddoold->getProperty(id);
 if (a.typeCode() == DKAny::tc_ushort)
 docType = (short)(USHORT)a;
 else
 docType = a;
 }
 if (docType == DK_CM_DOCUMENT)
 {
 cout << "create the DDO from the pidstring..." << endl;
 p = dsOD->createDDO(ddoold->getObjectType(), DK_CM_DOCUMENT);
 p->setPidObject(pid);

 dsOD->retrieveObject((dkDataObject*)p);
 }
 delete pid;
 }
}

```

```

 delete ddoold;
 }
 delete pCur;
}

```

## Apresentar documentos e os respectivos e atributos

O exemplo que se segue apresenta o documento localizado pela consulta com as respectivas partes e atributos.

### Exemplo: Java

```

//-----For each data item, get the attributes
//-----numDataItems is the number of data items
for (j = 1; j <= numDataItems; j++)
{
 a = p.getData(j)
 strDataName = p.getDataName(j);
 System.out.println(" " + j + ". Attribute Name: " + strDataName);
 System.out.println(" type: " + p.getDataPropertyByName
 (j,DK_PROPERTY_TYPE));

 System.out.println(" nullable: " +
 p.getDataPropertyByName (j,DK_PROPERTY_NULLABLE));
 if (strDataName.equals(DKPARTS) == false &&
 strDataName.equals("DKResource") == false &&
 strDataName.equals("DKViews") == false &&
 strDataName.equals("DKLargeObject") == false &&
 strDataName.equals("DKFixedView") == false &&
 strDataName.equals("DKAnnotations") == false)
 {
 System.out.println(" attribute id: " +
 p.getDataPropertyByName(j,DK_PROPERTY_ATTRIBUTE_ID));
 }
 //-----Check for the type of the attribute
 if (a instanceof String)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof Integer)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof Short)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof DKDate)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof DKTime)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof DKTimestamp)
 {
 System.out.println(" Attribute Value: " + a);
 }
 else if (a instanceof dkCollection)
 {
 System.out.println(" Attribute Value is collection");
 pCol = (dkCollection)a;
 pIter = pCol.createIterator();
 i = 0;
 while (pIter.more() == true)
 {
 i++;

```

```

a = pIter.next();
pD0 = (dkDataObjectBase)a;

// continued...
if (pD0.protocol() == DK_XD0)
{
 System.out.println(" dkXD0 object " + i + " in collection");
 pXD0 = (dkXD0)pD0;
 DKPidXD0 pid2 = pXD0.getPidObject();
 System.out.println(" XD0 pid string: " +
 pid2.pidString());
 //----- Retrieve and open instance handler for an XD0
 pXD0.retrieve();
 // pXD0.open();
}
}
else if (a != null)
{
 System.out.println(" Attribute Value: " + a.toString());
 if (strDataName.equals("DKResource") ||
 strDataName.equals("DKFixedView") ||
 strDataName.equals("DKLargeObject"))
 {
 pD0 = (dkDataObjectBase)a;

 if (pD0.protocol() == DK_XD0)
 {
 System.out.println(" dkXD0 object ");
 pXD0 = (dkXD0)pD0;
 DKPidXD0 pid2 = pXD0.getPidObject();
 System.out.println(" XD0 pid string: " +
 pid2.pidString());
 // Retrieve and open instance handler for an XD0
 pXD0.retrieve();
 // pXD0.open();
 }
 }
}
}

```

### Exemplo: C++

```

DKDDO *p = 0;
DKAny a;

for (j = 1; j <= numDataItems; j++)
{
 a = p->getData(j);
 strDataName = p->getDataName(j);

 cout << " " << j << ". Attribute Name: " << strDataName << endl;
 cout << "type: " << p->getDataPropertyByName(j,DK_PROPERTY_TYPE)<<endl;
 cout << "nullable: "
 << p->getDataPropertyByName(j,DK_PROPERTY_NULLABLE) << endl;

 if (strDataName != DK_CM_DKPARTS &&
 strDataName != "DKResource" &&
 strDataName != "DKViews" &&
 strDataName != "DKLargeObject" &&
 strDataName != "DKPermissions" &&
 strDataName != "DKFixedView" &&
 strDataName != "DKAnnotations")
 {
 cout << " attribute ID: "
 << p->getDataPropertyByName(j,DK_PROPERTY_ATTRIBUTE_ID) << endl;
 }

 if (a.typeCode() == DKAny::tc_string)
 {
 DKString astring = a;
 }
}

```

```

 cout << " attribute Value (string): " << astring << endl;
 }
 else if . . .
 {
 // ----- Handle each of the other types
 }
 else if (a.typeCode() != DKAny::tc_null)
 {
 cout << " Attribute Value (non NULL): " << a << endl;
 if (strDataName == "DKResource" ||
 strDataName == "DKFixedView" ||
 strDataName == "DKLargeObject")
 {
 pDO = (dkDataObjectBase*)a;
 if (pDO->protocol() == DK_XDO)
 {
 cout << " dkXDO object " << endl;
 pXDO = (dkXDO*)pDO;
 pidXDO = (DKPidXDOOD*)pXDO->getPid();
 cout << " XDO PID string: " << pidXDO->pidString() << endl;
 // ----- Obter e abrir tratador de instância para um XDO
 pXDO->retrieve();
 }
 }
 }
 else cout << " Attribute Value is NULL" << endl;

```

Para obter a aplicação completa, consulte o ficheiro TRetrieveOD.cpp no directório samples.

## Activar o modo de pastas OnDemand

Para activar o modo de pastas do OnDemand, a cadeia ENTITY\_TYPE=TEMPLATES tem de ser transmitida para o conector OnDemand como parte da cadeia de ligação ou da cadeia de configuração.

Uma cadeia de configuração exemplo teria o seguinte aspecto:

### Exemplo: Java

```
DKDatastoreOD dsOD = new DKDatastoreOD("ENTITY_TYPE=TEMPLATES");
```

### Exemplo: C++

```
DKDatastoreOD* dsOD = new DKDatastoreOD("ENTITY_TYPE=TEMPLATES");
```

Uma cadeia de ligação exemplo teria o seguinte aspecto:

### Exemplo: Java

```
DKDatastoreOD dsOD = new DKDatastoreOD();
dsOD.connect(hostname, userid, password, "ENTITY_TYPE=TEMPLATES");
```

### Exemplo: C++

```
DKDatastoreOD* dsOD = new DKDatastoreOD("ENTITY_TYPE=TEMPLATES");
dsOD->connect(hostname, userid, password, "ENTITY_TYPE=TEMPLATES");
```

## Pesquisa assíncrona

O conector OnDemand suporta pesquisas federadas e assíncronas directas.

Uma pesquisa assíncrona não ocupa o módulo principal e permite o cancelamento da pesquisa em qualquer altura. Faça clique em **Stop Search (Parar Pesquisa)** no Search Template Viewer (Visualizador do Modelo de Pesquisa) para concluir a

pesquisa. A propriedade max hits no Search Template Viewer (Visualizador do Modelo de Pesquisa) limita o número máximo de resultados devolvidos.

O conector OnDemand também suporta pesquisas síncronas e assíncronas no modo do grupo de aplicações a partir do cliente de AIX.

Se utilizar um critério de pesquisa como WHERE userid LIKE ', o número resultante de documentos devolvidos ao cliente pode consumir toda a memória disponível na máquina cliente. Ao emitir uma pesquisa assíncrona utilizando o método executeWithCallback(), pode definir o valor para o número máximo de documentos devolvidos e cancelar a pesquisa em qualquer momento.

Provavelmente também terá de aumentar o tamanho da pilha predefinido para a Java Virtual Machine (JVM) se o seu conjunto de resultados for demasiado extenso. O tamanho de pilha predefinido para cada módulo de Java é de 400k, que permite 3920 artigos devolvidos antes que a pilha ultrapasse a sua capacidade. Aumentar o tamanho da pilha da JVM para 800k duplica a capacidade para 7840 artigos. Caso seja necessário, poderá depois aumentar o tamanho da pilha da JVM.

Para aumentar o tamanho da pilha de JVM, utilize a opção de linha de comandos de Java -oss seguida de *nnnK* ou *nnM*, em que *K* significa Kilobytes e *M* significa Megabytes.

Para obter exemplo de utilização da pesquisa assíncrona, consulte os programas exemplo TRetrieveWithCallbackOD, TRetrieveFolderWithCallbackOD e TCallbackOD.

## Pastas OnDemand como modelos de pesquisa

Utilize beans visuais para pesquisar directas e federadas.

Três dos JavaBeans visuais do DB2 Information Integrator for Content, CMBSearchTemplateList, CMBSearchTemplateViewer e CMBSearchResultsView, utilizam modelos de pesquisa do DB2 Information Integrator for Content. Pode utilizar estes beans para pesquisas federadas definindo o CMBCConnection dsType como Fed.

Poderá utilizar estes beans também para pesquisas directas em servidores OnDemand. Defina as propriedades seguintes antes de iniciar sessão:

```
connection.setDsType("OD");
connection.setServerName(<odserver>);
connection.setConnectString("ENTITY_TYPE=TEMPLATES");
```

## Pastas OnDemand como entidades nativas

O conector de OnDemand também pode correlacionar pastas OnDemand como entidades nativas especificando a cadeia de ligação "ENTITY\_TYPE=TEMPLATES".

Utilizar as pastas OnDemand como entidades pode ser útil em pesquisas federadas, onde as definições da pasta podem ser mais fáceis de trabalhar do que em grupos de aplicações OnDemand, a entidade nativa predefinida para OnDemand.

Para pesquisas federadas, especifique a definição do servidor na administração do DB2 Information Integrator for Content. Depois já pode definir e correlacionar as entidades federadas para as pastas no servidor OnDemand.

## Criar e modificar anotações

Ao utilizar o visualizador OnDemand, que é iniciado pelo bean CMBDocumentViewer, pode criar, modificar e eliminar anotações de documentos OnDemand utilizando o bean CMBDataManagement e a classe CMBAnnotation associada.

## Rastreio

Pode rastrear eventos com o conector de OnDemand.

Para activar o rastreio de API de Java do conector, coloque o ficheiro INI de rastreio (cmbodtrace.ini para Java ou cmbodCtrace.ini para C++), na raiz da unidade C. Em AIX, coloque este ficheiro em: /opt/IBM/db2cmv8. Em Solaris, coloque este ficheiro em /opt/IBMcmb/cmgmt. Em Linux, coloque também este ficheiro em /opt/IBM/db2cmv8.

O directório de saída de dados predefinido para ficheiros de rastreio é C:\Ctrace. Para escrever informações de rastreio noutra local, edite o ficheiro INI de rastreio. Em AIX, tenha em atenção que os nomes dos ficheiros devem estar em letra minúscula.

Certifique-se de que o nome de caminho especificado no ficheiro de rastreio é válido e de que a linha que contém CMBODTRACEDIR não é antecedida por um sinal de #. Os ficheiros INI de rastreio exemplo são:

### Exemplo: Java

```
#=====
This is a java property file - not a real INI file!
*****#
For windows systems, make sure to use TWO BACK SLASH CHARACTERS (\\)
to separate the directory names!!! =====
*****#
#
***** On windows systems, this file must be located in c:\ *****
#
OD Trace File Directory Name property - CMBODTRACEDIR
#
The CMBODTRACEDIR property defines the directory where the trace files
will be written to. If the directory name does not exist, it will be
created.
#
Please make sure that the directory names are separated by two back slash
characters to avoid undesirable results.
#
Please make sure the path name does not point to an existing file name.
Otherwise, no trace files will be created.
#
The trace output directory name can be changed to point to a drive
where more space is available. But it is recommended not to change the
trace output directory name in the middle of an active trace session.
#
CMBODTRACESCOPE controls how much trace information to generate.
#
CMBODTRACESCOPE=ENTRY_EXIT_JNI_ONLY
Trace the entry & exit points in JNI only. Produce the least amount
of trace.
#
CMBODTRACESCOPE=ENTRY_EXIT_ONLY
Trace the entry and exit points in Java methods and JNI functions.
#
CMBODTRACESCOPE=JNI_ONLY
Full trace for the JNI functions only.
```

```
#
If CMBODTRACESCOPE is missing, or set to anything else,
a full trace will be taken.
#
To disable the trace, add a leading # character in column 1.
#
AIX: change the following line to CMBODTRACEDIR=/usr/lpp/cmb/cmgmt/trace
Sun: change the following line to CMBODTRACEDIR=/opt/IBMcmb/cmgmt/trace
CMBODTRACEDIR=c:\\trace
```

### Exemplo: C++

```
=====
OnDemand Trace INI file
#
OnDemand Trace File Directory Name key - CMBODTRACEDIR
#
The CMBODTRACEDIR key defines the directory where the trace files will
be written to. If the directory name does not exist, it will be created.
#
Please make sure the path name does not point to an existing file name.
Otherwise, no trace files will be created.
#
The trace output directory name can be changed to point to a drive
where more space is available. But it is recommended not to change
the trace output directory name in the middle of an active trace
session.
#
CMBODTRACESCOPE controls how much trace information to generate.
#
CMBODTRACESCOPE=ENTRY_EXIT_ONLY
Trace only the entry and exit of all C++ methods and functions.
#
If CMBODTRACESCOPE is missing, or set to anything else, a full trace
is taken.
#
To disable the trace, add a leading # character in column 1 on
the CMBODTRACEDIR line.
#
[ODCTRACE]
For AIX: change next line to CMBODTRACEDIR=/usr/lpp/cmb/cmgmt/ctrace
CMBODTRACEDIR=D:\\Ctrace
#CMBODTRACESCOPE=ENTRY_EXIT_ONLY
```

---

## Trabalhar com o ImagePlus for OS/390

As APIs do DB2 Information Integrator for Content suportam um grande número de funções ao utilizar servidores de conteúdos do ImagePlus for OS/390.

As APIs do DB2 Information Integrator for Content suportam as seguintes funções ao trabalhar com os servidores de conteúdos do ImagePlus for OS/390:

- Estabelecer e anular ligações a um ou mais servidores de ImagePlus for OS/390.
- Obter categorias.
- Obter campos de atributo.
- Obter pastas.
- Obter documentos.

Represente um servidor de conteúdos do ImagePlus for OS/390 utilizando o DKDatastoreIP na aplicação.

**Restrição:** O ImagePlus for OS/390 não suporta:

- Net Search Extender

- Consulta combinada
- Cesto de trabalho e fluxo de trabalho

## Enumerar entidades e atributos

Após a criação e ligação de um servidor de conteúdos para o servidor de conteúdos do ImagePlus for OS/390 como objecto de DKDatastoreIP, pode verificar as entidades e atributos para o servidor de conteúdos.

O exemplo seguinte enumera todas as entidades de um servidor de conteúdos de ImagePlus for OS/390:

### Exemplo: Java

```
//--- After creating a datastore and connecting
// dsIP is a DKDatastoreIP object
DKEntityDefIP entDef = null;
DKAttrDefIP attrDef = null;

DKSequentialCollection pCol =
(DKSequentialCollection)dsIP.listEntities();
dkIterator pIter = null;

if (pCol == null)
{
 //-- Handle if the collection of entities is null
}
else
{
 ... // ----- Process as appropriate
```

A aplicação exemplo completa da qual este exemplo foi retirado (TListCatalogIP.java) está disponível no directório samples.

### Exemplo: C++

```
// List entities...
DKEntityDefIP* docDef = 0;
DKAttrDefIP* attrDef = 0;

cout << "---List entities---" << endl;
DKSequentialCollection* pCol =
(DKSequentialCollection*)(dsIP.listEntities());
dkIterator* pIter = 0;

if (pCol == 0)
{
 cout << "collection of entities is null!" << endl;
}
else
{
 ...
```

A aplicação exemplo completa da qual este exemplo foi retirado (TListCatalogIP.cpp) está disponível no directório samples.

O exemplo que se segue enumera todos os atributos associados a cada entidade, utilizando as funções getAttr e listAttrNames do DKEntityDefIP.

### Exemplo: Java

```
//-- List attributes using listAttrNames and getAttr methods

pIter = pCol.createIterator();
```



```

while (pIter.more())
{
 // ----- Iterate over the each entity
 entDef = (DKEntityDefIP)pIter.next();
 System.out.println(" Entity type : " + entDef.getType());
 System.out.println(" Entity type name: " + entDef.getName());

 // ----- Get a list of attributes for the entity
 String[] attrNames = entDef.listAttrNames();
 int count = attrNames.length;
 for (int i = 0; i < count; i++)
 {
 attrDef = (DKAttrDefIP)entDef.getAttr(attrNames[i]);
 System.out.println(" Attr name
: " + attrDef.getName());
 System.out.println(" Attr id
: " + attrDef.getId());
 System.out.println(" Entity name
: " + attrDef.getEntityName());
 System.out.println(" Datastore name
: " + attrDef.datastoreName());
 System.out.println(" Attr type
: " + attrDef.getType());
 System.out.println(" Attr restrict
: " + attrDef.getStringType());
 System.out.println(" Attr min val
: " + attrDef.getMin());
 System.out.println(" Attr max val
: " + attrDef.getMax());
 System.out.println(" Attr display
: " + attrDef.getSize());
 System.out.println(" Attr precision
: " + attrDef.getPrecision());
 System.out.println(" Attr scale
: " + attrDef.getScale());
 System.out.println(" Attr update ? "
+ attrDef.isUpdatable());
 System.out.println(" Attr nullable ? "
+ attrDef.isNullable());
 System.out.println(" Attr queryable? "
+ attrDef.isQueryable());
 System.out.println(" ");
 }
}

```

## Exemplo: C++

```

// Method 1:
cout << "List attributes using listAttrNames
and getAttr functions" << endl;

pIter = pCol->createIterator();
while (pIter->more())
{
 docDef = (DKEntityDefIP*)(pIter->next()->value());
 cout << " Document type : "
<< docDef->getType() << endl;
 cout << " Document type name: "
<< docDef->getName() << endl;

 long tmpCount;
 DKString* attrNames;

 // Upon return, tmpCount contains the number
 // of elements in the list.
 attrNames = docDef->listAttrNames(tmpCount);
 for (int i=0; i<tmpcount; i++)

```

```

 {
 cout << " Attr name before lookup "
 << attrNames[i] << endl;
 attrDef = (DKAttrDefIP*)(docDef->
getAttr(attrNames[i]));
 cout << " Attr name [" << i << "]"
: " << attrDef->getName() << endl;
 cout << " Attr id
: " << attrDef->getId() << endl;
 cout << " Entity name
: " << attrDef->getEntityName() << endl;
 cout << " Datastore name
: " << attrDef->datastoreName() << endl;
 cout << " Attr type
: " << attrDef->getType() << endl;
 cout << " Attr restrict
: " << attrDef->getStringType() << endl;
 cout << " Attr min val
: " << attrDef->getMin() << endl;
 cout << " Attr max val
: " << attrDef->getMax() << endl;
 cout << " Attr display
: " << attrDef->getSize() << endl;
 cout << " Attr precision
: " << attrDef->getPrecision() << endl;
 cout << " Attr scale
: " << attrDef->getScale() << endl;
 cout << " Attr update ? "
<< attrDef->isUpdatable() << endl;
 cout << " Attr nullable ? "
<< attrDef->isNullable() << endl;
 cout << " Attr queryable? "
<< attrDef->isQueryable() << endl;
 cout << "" << endl;
 delete attrDef;
 } // end for

 delete [] attrNames;

} // end while
delete pIter;

```

O exemplo seguinte demonstra uma forma alternativa de enumerar os atributos associados a cada entidade utilizando o método `listEntityAttrs` do `DKDatastoreIP`.

### Exemplo: Java

```

// - List attributes using listEntityAttrs method
pIter = pCol.createIterator();
while (pIter.more())
{
 entDef = (DKEntityDefIP)pIter.next();
 System.out.println(" Entity type :
" + entDef.getType());
 System.out.println(" Entity type name:
" + entDef.getName());

 DKSequentialCollection pAttrCol =
 (DKSequentialCollection)dsIP.listEntityAttrs
(entDef.getName());
 if (pAttrCol == null)
 {
 // -- Handle if the collection of attributes is null
 }
 else
 {
 dkIterator pAttrIter = pAttrCol.createIterator();

```

```

while (pAttrIter.more())
{
 attrDef = (DKAttrDefIP)pAttrIter.next();
 System.out.println(" Attr name :
" + attrDef.getName());
 System.out.println(" Attr id :
" + attrDef.getId());
 System.out.println(" Entity name :
" + attrDef.getEntityName());
 System.out.println(" Datastore name:
" + attrDef.datastoreName());
 System.out.println(" Attr type :
" + attrDef.getType());
 System.out.println(" Attr restrict :
" + attrDef.getStringType());
 System.out.println(" Attr min val :
" + attrDef.getMin());
 System.out.println(" Attr max val :
" + attrDef.getMax());
 System.out.println(" Attr display :
" + attrDef.getSize());
 System.out.println(" Attr precision:
" + attrDef.getPrecision());
 System.out.println(" Attr scale :
" + attrDef.getScale());
 System.out.println(" Attr update ?
" + attrDef.isUpdatable());
 System.out.println(" Attr nullable ?
" + attrDef.isNullable());
 System.out.println(" Attr queryable?
" + attrDef.isQueryable());
 System.out.println("");
}
}
}

```

### Exemplo: C++

```

// Method 2:
cout << "---List attributes using listEntityAttrs
function---" << endl;

pIter = pCol->createIterator();
while (pIter->more())
{
 docDef = (DKEntityDefIP*)(pIter->next()->value());
 //iterator returns DKAny*
 cout << " Document type : " << docDef->getType()
<< endl;
 cout << " Document type name: " << docDef->getName()
<< endl;
 DKSequentialCollection* pAttrCol =
(DKSequentialCollection *)
(dsIP.listEntityAttrs(docDef->getName()));
 if (pAttrCol == 0)
 {
 cout << "collection of entity attrs is null for entity "
<< docDef->getName()
<< endl;
 }
 else
 {
 int i=0;
 dkIterator* pAttrIter = pAttrCol->createIterator();
 while (pAttrIter->more())
 {
 i++;

```

```

// ----- The iterator returns a pointer to DKAny
attrDef = (DKAttrDefIP*)(pAttrIter->next()->value());
cout << " Attr name [" << i << "]" :
" << attrDef->getName() << endl;
cout << " Attr id :
" << attrDef->getId() << endl;
cout << " Entity name :
" << attrDef->getEntityName() << endl;
cout << " Datastore name:
" << attrDef->datastoreName() << endl;
cout << " Attr type :
" << attrDef->getType() << endl;
cout << " Attr restrict :
" << attrDef->getStringType() << endl;
cout << " Attr min val :
" << attrDef->getMin() << endl;
cout << " Attr max val :
" << attrDef->getMax() << endl;
cout << " Attr display :
" << attrDef->getSize() << endl;
cout << " Attr precision:
" << attrDef->getPrecision() << endl;
cout << " Attr scale :
" << attrDef->getScale() << endl;
cout << " Attr update ?
" << attrDef->isUpdatable() << endl;
cout << " Attr nullable ?
" << attrDef->isNullable() << endl;
cout << " Attr queryable?
" << attrDef->isQueryable() << endl;
cout << "" << endl;
delete attrDef;
} // end while
delete pAttrIter;
}
delete pAttrCol;
delete docDef;
} // end while
delete pIter;
}
delete pCol;

```

## Sintaxe da consulta do ImagePlus for OS/390

O exemplo seguinte apresenta a sintaxe de consulta para o ImagePlus for OS/390.

### Exemplo: Java

```

SEARCH = (COND=(search_expression),
ENTITY={entity_name | mapped_entity_name}
[,MAX_RESULTS=maximum_results]);
[OPTION=([CONTENT={YES | ATTRONLY |
NO};] [PENDING={YES | NO};])]

```

### Exemplo: C++

```

SEARCH=(COND=(search_expression),
ENTITY={entity_name | mapped_entity_name}
[,MAX_RESULTS=maximum_results]);
[OPTION=([CONTENT={YES | ATTRONLY | NO};] [PENDING={YES | NO};])]

```

A consulta utiliza os seguintes parâmetros:

#### search\_expression

Cada expressão de pesquisa consiste num ou mais critérios de pesquisa. Só pode utilizar o operador booleano AND entre critérios de pesquisa.

Os critérios de pesquisa possuem a seguinte forma:

```
{attr_name | mapped_attr_name
} operator literal
```

em que:

**attr\_name**

É o nome do atributo da entidade no qual se deve basear a pesquisa.

**mapped\_attr\_name**

É o nome de atributo correlacionado com o atributo no qual se deve basear a pesquisa.

**operador**

Todos os atributos suportam sinais de igual (==). Para atributos de tipo DATE, pode utilizar os seguintes operadores adicionais:

> maior que  
< menor que  
>= maior que ou igual a  
<= menor que ou igual a

**literal**

Um literal. Em atributos numéricos, não utilize aspas ("), por exemplo:

```
FolderType == 9
```

Para atributos de data, hora e marca da hora, não são necessários aspas nem apóstrofes ('), mas são tolerados, por exemplo:

```
ReceiveDate == 1999-03-08
ReceiveDate == '1999-03-08'
```

Para atributos de cadeia, não são necessários aspas nem apóstrofes ('), mas são tolerados. Se a cadeia contiver um apóstrofo ('), a cadeia terá de ser especificada através de dois apóstrofes, por exemplo para um valor de Folder'1:

```
FolderId == 'Folder''1'
```

**entity\_name**

Nome da entidade a ser pesquisada.

**mapped\_entity\_name**

Nome da entidade correlacionada com entidade a ser pesquisada.

**maximum\_results**

Número máximo de resultados a devolver.

As palavras chave opcionais são:

**CONTENT**

Controla a quantidade de informação devolvida nos resultados

**YES (predefinição)**

Define os PIDs, os atributos e os seus valores de um documento ou pasta. São definidos os PIDs de XDO, caso existam partes num documento. Se existirem documentos numa pasta, são definidos os PIDs dos documentos.

**NO** Só define os PIDs do documento ou da pasta.

#### ATTRONLY

Só define os PIDs, os atributos e os seus valores de um documento ou pasta.

#### PENDING

Controla se deve ou não incluir documentos pendentes que não tem quaisquer partes. Esta opção aplica-se apenas quando ENTITY é definido como DOCUMENT ou como uma entidade correlacionada com o DOCUMENT .

**YES** Inclui documentos pendentes nos resultados.

#### NO (predefinição)

Não inclui documentos pendentes nos resultados.

---

## Trabalhar com o DB2 Content Manager for AS/400

As classes de API facultadas para o DB2 Content Manager for AS/400 (VisualInfo for AS/400) são idênticas às facultadas para o DB2 Content Manager.

**Restrição:** O DB2 Content Manager for AS/400 não suporta:

- Net Search Extender
- Consulta combinada
- Cesto de trabalho e fluxo de trabalho

## Enumerar entidades (classes de índices) e atributos

Pode representar um servidor de conteúdos do DB2 Content Manager for AS/400 como um DKDatastoreV4.

Após ter criado o servidor de conteúdos e de ter estabelecido ligação ao mesmo, pode enumerar as entidades (classes de índice) e atributos do servidor DB2 Content Manager for AS/400 (consultar o exemplo).

### Exemplo: Java

```
//-- After creating a datastore (dsV4)
//and connecting, get index classes
pCol = (DKSequentialCollection) dsV4.listEntities();
pIter = pCol.createIterator();
i = 0;
while (pIter.more() == true)
{
 i++;
 icDef = (DKIndexClassDefV4)pIter.next();
 strIndexClass = icDef.getName();
 ... //-- Process the index classes as appropriate
 //-- Get the attributes
 pCol2 = (DKSequentialCollection)
dsV4.listEntityAttrs(strIndexClass);
 pIter2 = pCol2.createIterator();
 j = 0;

 while (pIter2.more() == true)
 {
 j++;
 attrDef = (DKAttrDefV4)pIter2.next();
 ... //-- Process the attributes
 }
}

dsV4.disconnect();
dsV4.destroy();
```

A aplicação exemplo completa da qual este exemplo foi retirado (TListCatalogV4.java) está disponível no directório samples.

### Exemplo: C++

```
cout << "list index class(es)..." << endl;
pCol = (DKSequentialCollection*)
((dkCollection*)dsV4.listSchema());
pIter = pCol->createIterator();
i = 0;

while (pIter->more() == TRUE)
{
 i++;
 a = (*pIter->next());
 strIndexClass = a;
 cout << "index class name [" << i << "] - "
 << strIndexClass << endl;
 cout << " list attribute(s) for " << strIndexClass
 << " index class:" << endl;
 pCol2 =

 (DKSequentialCollection*)((dkCollection*)
 dsV4.listSchemaAttributes(strIndexClass));
 pIter2 = pCol2->createIterator();
 j = 0;

 while (pIter2->more() == TRUE)
 {
 j++;
 pA = pIter2->next();
 pDef = (DKAttributeDef*) pA->value();
 cout << " Attribute name [" << j << "] - "
 << pDef->name << endl;
 cout << " datastoreType - " << pDef->datastoreType
 << endl;
 cout << " attributeOf - " << pDef->attributeOf
 << endl;
 cout << " type - " << pDef->type << endl;
 cout << " size - " << pDef->size << endl;
 cout << " id - " << pDef->id << endl;
 cout << " nullable - " << pDef->nullable
 << endl;
 cout << " precision - " << pDef->precision
 << endl;
 cout << " scale - " << pDef->scale << endl;
 cout << " string type - " << pDef->stringType
 << endl;
 }

 cout << " " << j << " attribute(s) listed for "
 << strIndexClass << " index class" << endl;
 pCol2->apply(deleteDKAttributeDef);
 delete pIter2;
 delete pCol2;
}

delete pIter;
delete pCol;
cout << i << " index class(es) listed" << endl;
dsV4.disconnect();
cout << "datastore disconnected" << endl;
```

A aplicação exemplo completa da qual esta aplicação foi retirada (TListCatalogV4.cpp) está disponível no directório samples.

## Executar uma consulta

O exemplo seguinte executa uma consulta em DB2 Content Manager for AS/400 e processa os resultados.

### Exemplo: Java

```
// ----- After creating a datastore
// (dsV4) and connecting, build the
// query and parameters and execute it
pCur = dsV4.execute(cmd,DK_CM_PARAMETRIC_QL_TYPE,parms);
...

if (pCur == null)
{
// ---- Handle if the cursor is null
}

while (pCur.isValid())
{
 p = pCur.fetchNext();
 if (p != null)
 {
 cnt++;
 i = pCur.getPosition();
 System.out.println("\n====> Item " + i + "
<====");
 numDataItems = p.dataCount();
 DKPid pid = p.getPid();
 System.out.println(" pid string: " +
pid.pidString());
 k = p.propertyId(DK_CM_PROPERTY_ITEM_TYPE);

 if (k > 0)
 {
 Short sVal = (Short)p.getProperty(k);
 j = sVal.shortValue();
 switch (j)
 {
 case DK_CM_DOCUMENT :
 {
... // Handle if the item is a document ");
 break;
 }
 case DK_CM_FOLDER :
 {
 ... // Handle if the item is a folder
 break;
 }
 }
 }
 }
}

for (j = 1; j <= numDataItems; j++)
{
 a = p.getData(j);
 strDataName = p.getDataName(j);
 ... // Process the attributes as appropriate
 if (strDataName.equals(DKPARTS) == false
 && strDataName.equals(DKFOLDER) == false)
 {
 System.out.println(" attribute id: "
+ p.getDataPropertyByName(j,DK_CM_PROPERTY_ATTRIBUTE_ID));
 }
 // continued...
 if (a instanceof String)
 {
 System.out.println(" Attribute Value: " + a);
 }
}
```



```

 }
 else if (a instanceof Integer)
... // ---- Handle each type for attribute {
 else if (a instanceof dkCollection)
 {
//-- Handle if attribute value is a collection
 pCol = (dkCollection)a;
 pIter = pCol.createIterator();
 i = 0;
 while (pIter.more() == true)
 {
 i++;
 a = pIter.next();
 pDO = (dkDataObjectBase)a;

 if (pDO.protocol() == DK_CM_PDDO)
 {
 // Processar um DDO
 pDDO = (DKDDO)pDO;
 ...
 }
 else if (pDO.protocol() == DK_CM_XDO)
 {
 // Process an XDO
 pXDO = (dkXDO)pDO;
 DKPidXDO pid2 = pXDO.getPid();
 ...
 }
 }
 }
 else if (a != null)
 {
 // Process the attribute
 }
 else ... // Handle if the attribute is null
 }
}
pCur.destroy(); // Delete the cursor when you're done

```

A aplicação exemplo completa a partir da qual este exemplo foi retirado (TExecuteV4.java) está disponível no directório samples.

### Exemplo: C++

```

cout << "executing query..." << endl;
...
pCur = dsV4.execute(cmd);
cout << " query executed" << endl;
...
cout << "\n..... Displaying query results \n\n";

...
while (pCur->isValid())
{
 p = pCur->fetchNext();

 if (p != 0)
 {
 cout << "=====> " << "Item " << cnt << " <=====" << endl;
 numDataItems = p->dataCount();
 pid = p->getPid();
 cout << " Pid String: " << pid.pidString() << endl;
 k = p->propertyId(DK_CM_PROPERTY_ITEM_TYPE);

 if (k > 0)
 {

```

```

a = p->getProperty(k);
val = a;
cout << " *****" << endl;

 switch (val)
 {
 case DK_CM_DOCUMENT :
 {
 cout << " Item is a document " << endl;
 break;
 }
 case DK_CM_FOLDER :
 {
 cout << " Item is a folder " << endl;
 break;
 }
 }

cout << " *****" << endl;
}

cout << " Number of Data Items: "
<< numDataItems << endl;

for (j = 1; j <= numDataItems; j++)
{
 a = p->getData(j);
 strDataName = p->getDataName(j);

 switch (a.typeCode())
 {
 case DKAny::tc_string :
 {
 strData = a;
 cout << " attribute name: " << strDataName
 << ", value: " << strData << endl;
 break;
 }
 // continued...
 case DKAny::tc_long :
 {
 lVal = a;
 cout << " attribute name: " << strDataName
 << ", value: " << lVal << endl;
 break;
 }

 case DKAny::tc_null :
 {
 cout<<" attribute name: "<<strDataName<<",
 value: NULL "<< endl;
 break;
 }

 case DKAny::tc_collection :
 {
 pdCol = a;
 cout<<strDataName<<" collection name:
 "<<strDataName << endl;
 cout<<"-----"<<endl;
 pdIter = pdCol->createIterator();
 ushort b = 0;

 while (pdIter->more() == TRUE)
 {

```

```

 b++;
 cout << " -----" << endl;
 a = *(pdIter->next());
 pDDBase = a;

 if (pDDBase->protocol() == DK_PDDO)
 {
 pDDO = (DKDDO*)pDDBase;
 cout << " DKDDO object " << b
<< " in " << strDataName
 << " collection " << endl;
 k = pDDO->propertyId
(DK_CM_PROPERTY_ITEM_TYPE);

 if (k > 0)
 {
 a = pDDO->getProperty(k);
 val = a;
 }
 cout << " *****" << endl;

 switch (val)
 {
 case DK_CM_DOCUMENT :

 cout << " Item is a document " << endl;
 break;
 }
 case DK_CM_FOLDER :
 {
 cout << " Item is a folder " << endl;
 break;
 }
 }
 cout << " *****" << endl;
 }
 // continued...
 else if (pDDBase->protocol() == DK_XDO)
 {
 pXDO = (dkXDO*)pDDBase;
 cout << " dkXDO object " << b << " in "
<< strDataName
<< " collection " << endl;

 }
 }

 if (pdIter != 0)
 {
 delete pdIter;
 }

 if (b == 0)
 {
 cout << strDataName << " collection has
 no elements " << endl;
 }

 cout << " -----" << endl;
 break;
 }

 default:
 cout << "Type is not supported\n";
 }
}

```

```

 cout<<"type: "<< p->getDataPropertyByName
(j,DK_CM_PROPERTY_TYPE)<<endl;
 cout<<"nullable: "<< p->getDataPropertyByName
(j,DK_CM_PROPERTY_NULLABLE)
 << endl;

 if (strDataName != DKPARTS && strDataName
!= DKFOLDER)
 {
 cout << " attribute id: "
<< p->getDataPropertyByName
(j,DK_PROPERTY_ATTRIBUTE_ID) << endl;
 }
 }
 cnt++;
 delete p;
}
}
cout << "Total Item count is " << cnt-1 << endl;

if (pCur != 0)
 delete pCur;

```

A aplicação exemplo completa a partir da qual esta aplicação foi retirada (TExecuteV4.cpp) está disponível no directório samples.

## Executar uma consulta paramétrica

O exemplo seguinte executa uma consulta paramétrica.

### Exemplo: Java

```

// ----- Create the query string and the query object
String cmd = "SEARCH=(INDEX_CLASS=V4DEMO)";
pQry = dsV4.createQuery(cmd, DK_CM_PARAMETRIC_QL_TYPE, parms);
// ----- Run the query
pQry.execute(parms);

System.out.println("number of query results = " + pQry.numberOfResults());

// ----- Processing the query results
pResults = (DKResults)pQry.result();
processResults((dkCollection)pResults);
...

```

A aplicação exemplo completa a partir da qual este exemplo foi retirado (TSamplePQryV4.java) está disponível no directório samples.

### Exemplo: C++

```

cout << "query string: " << cmd << endl;
cout << "creating query..." << endl;
pQry = dsV4.createQuery(cmd);
cout << "executing query..." << endl;
pQry->execute();
cout << "query executed" << endl;
cout << "getting query results..." << endl;
any = pQry->result();
pResults = (DKResults*)((dkCollection*) any);

processResults(pResults);

dsV4.disconnect();
cout << "datastore disconnected" << endl;
delete pQry;
delete pResults;

```

A aplicação exemplo completa a partir da qual esta aplicação foi retirada (TSamplePQryV4.cpp) está disponível no directório samples.

---

## Construir aplicações de fluxo de trabalho do DB2 Information Integrator for Content (obsoleto)

Ao utilizar classes e APIs de DB2 Information Integrator for Content, pode criar ou expandir as suas próprias aplicações para utilizarem o suporte de fluxo de trabalho de DB2 Information Integrator for Content.

Normalmente, é efectuada uma pesquisa federada e o fluxo de trabalho é iniciado com o resultado da pesquisa (um artigo de conteúdo ou uma pasta de vários artigos de conteúdo). Irá utilizar as APIs para aceder a uma lista de trabalho e depois apresentar os conteúdos da lista de trabalho. À medida que cada actividade é concluída, o fluxo de trabalho avança para a actividade seguinte do fluxo de trabalho.

Esta acção está obsoleta no DB2 Content Manager Versão 8.4.

---

### Ligação a serviços de fluxo de trabalho

Para utilizar o fluxo de trabalho do DB2 Information Integrator for Content nas aplicações, comece por criar uma instância de DKWorkflowServicesFed e, em seguida, ligue-a. Quando tiver terminado a utilização do serviço de fluxo de trabalho, terá de anular a ligação invocando as funções disconnect() e delete().

O exemplo seguinte inicia os serviços do fluxo de trabalho:

#### Exemplo: Java

```
// ----- Create the strings for the name of the
//service, user ID and Password
String wfsrv = "icmnlbdb";
String userid = "icmadmin";
String pw = "password";
// ----- Create a federated datastore
DKDatastoreFed dsFed = new DKDatastoreFed();
dsFed.connect(wfsrv, userid, pw,"");
//----- Create the workflow service
DKWorkflowServiceFed svWF =new DKWorkflowServiceFed ();
// ----- Set the datastore in the workflow service
svWF.setDatastore(dsFed);
// ----- Connect to the service
svWF.connect (wfsrv, userid, pw,"");
```

#### Exemplo: C++

```
// ----- Create the strings for the name of the service, user ID
// ----- and Password
DKString wfsrv = "icmnlbdb";
DKString userid = "icmadmin";
DKString pw = "password";
// ----- Create a federated datastore
DKDatastoreFed* dsFed = new DKDatastoreFed();
dsFed->connect(wfsrv, userid, pw,"");
//----- Create the workflow service
DKWorkflowServiceFed* svWF =new DKWorkflowServiceFed ();
// ----- Set the datastore in the workflow service
svWF->setDatastore(dsFed);
// ----- Connect to the service
svWF->connect (wfsrv, userid, pw,"");
```

O exemplo seguinte desliga os serviços de fluxo de trabalho:

### Exemplo: Java

```
svWF.disconnect();
dsFed.disconnect();
svWF.destroy();
dsFed.destroy();
```

### Exemplo: C++

```
svWF->disconnect();
dsFed->disconnect();
delete svWF;
delete dsFed;
```

---

## Iniciar um fluxo de trabalho

Após criar o fluxo de trabalho, deve iniciá-lo.

Para iniciar um fluxo de trabalho:

1. Crie um objecto DKWorkflowFed e defina o nome do fluxo de trabalho.
2. Crie uma instância de fluxo de trabalho através de um modelo de fluxo de trabalho válido, que é uma definição do fluxo de trabalho concebida no construtor do fluxo de trabalho do DB2 Information Integrator for Content.
3. Defina o PID e a prioridade no contentor.
4. Inicie o fluxo de trabalho.

### Exemplo: Iniciar um fluxo de trabalho

O exemplo seguinte utiliza estes passos para iniciar um fluxo de trabalho:

#### Java

```
// ----- Create the DKWorkflowFed object and set the name
DKWorkflowFed WF = new DKWorkflowFed(svWF);
WF.setName("wf1");
// ----- Create an instance of a workflow with the workflow template name
WF.add("WD1");
// ----- Refresh the workflow object
WF.retrieve();
// ----- Construct the container object for the workflow
DKWorkflowContainerFed con = WF.inContainer();
// ----- Retrieve the container data
con.retrieve();
// ----- Add a PID string referring to an Extended Search document
con.setPersistentID("45 3 DES4ross10 Notes Help18 15 Help|23fa");
con.setPriority(100);
// ----- Update the container
con.update();
// ----- Start the workflow
WF.start(con);
```

#### C++

```
// - Create the DKWorkflowFed object and set the name
DKWorkflowFed* WF = new DKWorkflowFed(svWF);
WF->setName("wf1");
// Create an instance of a workflow with the workflow template name
WF->add("WD1");
// ----- Refresh the workflow object
WF->retrieve();
// ----- Construct the container object for the workflow
DKWorkflowContainerFed* con = WF.inContainer();
// ----- Retrieve the container data
```

```

con->retrieve();
// Add a PID string referring to a content item from Extended Search
con->setPersistentID("45 3 DES4ross10 Notes Help18 15 Help|23fa");
// ----- Assign a priority of 100
con->setPriority(100);
// ----- Update the container
con->update();
// ----- Start the workflow
WF->start(con);

. . .
// When you are done, clean up by deleting the container and workflow
delete con;
delete WF;

```

---

## Terminar um fluxo de trabalho

Pode terminar um fluxo de trabalho invocando a função `terminate()` ou `del()`.

### Exemplo: Java

```

//-----Retrieve the status of the workflow named WF
WF.retrieve();
int state = WF.state();
//-----Check the status and either terminate or delete
if (state ==DKConstantFed.DK_FED_FMC_PS_RUNNING ||
 state ==DKConstantFed.DK_FED_FMC_PS_SUSPENDED ||
 state ==DKConstantFed.DK_FED_FMC_PS_SUSPENDING)
{
 WF.terminate();
}
if (state == DKConstantFed.DK_FED_FMC_PS_READY ||
 state ==DKConstantFed.DK_FED_FMC_PS_FINISHED||
 state ==DKConstantFed.DK_FED_FMC_PS_TERMINATED)
{
 WF.del();
}

```

### Exemplo: C++

```

/---Construct a DKWorkFlowFed instance
DKWorkFlowFed* WF = new DKWorkFlowFed(svWF, "Test");
//-----Retrieve the status of the workflow named WF
WF->retrieve();
int state = WF->state();
//---Check the status and either terminate or delete
if (state == DK_FED_FMC_PS_RUNNING ||
 state == DK_FED_FMC_PS_SUSPENDED ||
 state == DK_FED_FMC_PS_SUSPENDING)
{
 WF->terminate();
}
if (state == DK_FED_FMC_PS_READY ||
 state == DK_FED_FMC_PS_FINISHED ||
 state == DK_FED_FMC_PS_TERMINATED)
{
 WF->del();
}
delete WF;

```

---

## Enumerar todos os fluxos de trabalho

Pode enumerar todos os fluxos de trabalho num serviço de fluxo de trabalho utilizando a função `listWorkFlows()`.



O exemplo seguinte enumera o nome e a descrição de todos os fluxos de trabalho num serviço de fluxo de trabalho referenciados pelo objecto svWF do DKWorkflowServiceFed.

### Exemplo: Java

```
// ----- Call the listWorkFlows method
DKSequentialCollection collWF =
(DKSequentialCollection)svWF.listWorkFlows();
DKWorkflowFed WF = null;
if (collWF != null)
{
 dkIterator iterWF = collWF.createIterator();
 while (iterWF.more() == true)
 {
 WF = (DKWorkflowFed)iterWF.next();
 WF.retrieve();
 System.out.println("name = " + WF.getName()
+ " description = "
+ WF.getDescription());
 }
 iterWF = null;
}
```

### Exemplo: C++

```
// ----- Call the listWorkFlows function
DKSequentialCollection *collWF =
(DKSequentialCollection*)svWF.listWorkFlows();
DKWorkflowFed *WF = NULL;
if (collWF != NULL)
{
 dkIterator *iterWF = collWF->createIterator();
 while (iterWF->more())
 {
 WF = (DKWorkflowFed*)(void*)(*iterWF->next());
 WF->retrieve();
 cout << "name = " + WF->getName()
<< " description = " << WF->getDescription() << endl;
 delete WF;
 }
 delete iterWF;
}
elimine collWF;
```

---

## Suspender um fluxo de trabalho

Pode suspender um fluxo de trabalho em execução indefinidamente ou com uma hora específica. Se facultar um DKTimestamp nulo, o DB2 Information Integrator for Content suspende o fluxo de trabalho indefinidamente.

O exemplo seguinte demonstra como suspender um fluxo de trabalho até determinada hora.

### Exemplo: Java

```
// ----- Construct a DKWorkflowFed object
DKWorkflowFed WF = new DKWorkflowFed(svWF, "Test");
WF.retrieve();
// ----- Call the suspend method if the workflow is in the running state
if (WF.state() == DKConstantFed.DK_FED_FMC_PS_RUNNING)
{
 // ----- Suspended until 2000-07-27-16.30.00.000000
 // ----- The timestamp uses the base year 1900; months are
```

```
// ----- numbered 0 to 11
DKTimestamp suspension = new DKTimestamp(100, 6, 27, 16, 30, 0, 0);
WF.suspend(suspension);
}
```

### Exemplo: C++

```
// ----- Construct a DKWorkflowFed instance
DKWorkflowFed* WF = new DKWorkflowFed(svWF, "Test");
WF->retrieve();
// ----- Call the suspend function if the workflow is in
the running state
if (WF->state() == DK_FED_FMC_PS_RUNNING)
{
 // ----- Suspended until 2000-07-27-16.30.00.000000
 DKTimestamp* suspension = new DKTimestamp(2000, 7,
 27, 16, 30, 0, 0);
 WF->suspend(suspension);
 delete suspension;
}
delete WF;
```

---

## Retomar um fluxo de trabalho

Pode retomar um fluxo de trabalho suspenso invocando a função `resume()`.

O exemplo seguinte retoma um fluxo de trabalho suspenso.

### Exemplo: Java

```
// ----- Construct a DKWorkflowFed object
DKWorkflowFed WF = new DKWorkflowFed(svWF, "Test");
WF.retrieve();
// ----- Call resume() if the workflow is in the suspended state
if (WF.state() == DKConstantFed.DK_FED_FMC_PS_SUSPENDED)
{
 WF.resume();
}
```

### Exemplo: C++

```
// ----- Construct a DKWorkflowFed instance
DKWorkflowFed* WF = new DKWorkflowFed(svWF, "Test");
WF->retrieve();
// ----- Check whether the workflow is suspended and call resume
if (WF->state() == DK_FED_FMC_PS_SUSPENDED)
{
 WF->resume();
}
delete WF;
```

---

## Enumerar todas as listas de trabalho

Pode enumerar todas as listas de trabalho num serviço de fluxo de trabalho invocando a função `listWorkLists()` no serviço de fluxo de trabalho.

O exemplo que se segue enumera o nome e a descrição de todas as listas de trabalho num serviço de fluxo de trabalho referenciado pela instância `svWF` do `DKWorkflowServiceFed`.

### Exemplo: Java

```
// ----- Call the listWorkLists method
DKSequentialCollection collWL =
(DKSequentialCollection)svWF.listWorkLists();
```

```

DKWorkListFed WL = null;
if (collWL != null)
{
 dkIterator iterWL = collWL.createIterator();
 while (iterWL.more() == true)
 {
 WL = (DKWorkListFed)iterWL.next();
 WL.retrieve();
 System.out.println("name = " + WL.getName()
+ " description = "
+ WL.getDescription());
 }
 iterWL = null;
}

```

### Exemplo: C++

```

// ----- Call the listWorkLists function
DKSequentialCollection *collWL =
 (DKSequentialCollection*)svWF.listWorkLists();
DKWorkListFed *WL = NULL;
if (collWL != NULL)
{
 dkIterator *iterWL = collWL->createIterator();
 while (iterWL->more())
 {
 WL = (DKWorkListFed*)(void*)(*iterWL->next());
 WL->retrieve();
 cout << "name = " << WL->getName()
<< " description = "
<< WL->getDescription() << endl;
 cout << "Threshold = "
<< WL->getThreshold() << endl;
 delete WL;
 }
 delete iterWL;
}
delete collWL;

```

---

## Aceder a uma lista de trabalhos

Pode aceder a uma lista de trabalho criando uma instância de DKWorkListFed com referência à lista de trabalho que criou utilizando o cliente de administração do sistema.

O exemplo seguinte acede a uma lista de trabalhos denominada WL0712 e apresenta as informações contidas nessa lista de trabalhos.

### Exemplo: Java

```

// ----- Create the DKWorkListFed
DKWorkListFed WL = new DKWorkListFed(svWF, "WL0712");
WL.retrieve();
// ----- Display information about the worklist
System.out.println ("worklist name = " + WL.getName());
System.out.println ("description = " + WL.getDescription() +
 " owner = " + WL.getOwner() +
 " filter = " + WL.getFilter() +
 " threshold = " + WL.getThreshold() +
 " sort criteria = " + WL.getSortCriteria());

```

### Exemplo: C++

```

// ----- Create the DKWorkListFed
DKWorkListFed* WL = new DKWorkListFed(svWF, "WL0712");
WL->retrieve();

```

```
// ----- Display information about the worklist
cout << "worklist name = " << WL->getName() << endl;
cout << "description = " << WL->getDescription() <<
 " owner = " << WL->getOwner() <<
 " filter = " << WL->getFilter() <<
 " threshold = " << WL->getThreshold() <<
 " sort criteria = " << WL->getSortCriteria() << endl;
// ----- Delete the worklist when you are done
delete WL;
```

---

## Aceder a artigos de trabalho

Após a criação do DKWorkListFed, pode obter os artigos de trabalho como recolha.

O exemplo seguinte obtém os artigos de trabalho.

### Exemplo: Java

```
// ----- Create a collection and an iterator
DKSequentialCollection coll = (DKSequentialCollection)WL.listWorkItems();
dkIterator iter = (DKSequentialIterator) coll.createIterator ();
Object a;
DKWorkItemFed item;
String nodename;
String workflowname;

// ----- Step through the collections
while (iter.more ())
{
 a = iter.next ();
 item = (DKWorkItemFed) a;
 if (item != null)
 {
 item.retrieve ();
 nodename = item.name ();
 workflowname = item.workflowName ();
 System.out.println ("workitem node = " + nodename +
 " workflow name = " + workflowname);
 }
}
iter = null;
```

### Exemplo: C++

```
DKSequentialCollection *coll;
dkIterator *iter;
DKWorkItemFed* item;
DKString nodename;
DKString workflowname;
// ----- Create a collection and an iterator
coll = (DKSequentialCollection*)WL->listWorkItems();

if (coll != NULL)
{
 iter = coll->createIterator();
 cout << "listWorkItems" << endl;
 // ----- Step through the collections
 while (iter->more ())
 {
 item = (DKWorkItemFed*)((void*)(*iter->next()));

 if (item != NULL)
 {
 //item.retrieve ();
 nodename = item->name();
 workflowname = item->workflowName();
 }
 }
}
```

```

 cout << "workitem node = " << nodename
 << " workflow name = " << workflowname << endl;
 delete item;
 }
}
delete iter;
delete coll;
}

```

---

## Mover artigos no fluxo de trabalho

À medida que um fluxo de trabalho avança, o utilizador move artigos de uma actividade para a actividade seguinte utilizando as funções `checkOut()` e `checkIn()`.

O exemplo seguinte demonstra como mover os artigos de trabalho. Tenha em atenção que apenas o utilizador do fluxo de trabalho actualmente nomeado para executar o artigo de trabalho pode dar entrada e dar saída do artigo de trabalho.

### Exemplo: Java

```

DKWorkItemFed item =new DKWorkItemFed(svWF, "wf1", "node1", wfuser);
item.retrieve();
// ----- Call the checkOut method to lock the workitem
item.checkOut();
// ----- Call the checkIn method
item.checkIn(null);

```

### Exemplo: C++

```

DKWorkItemFed* item =new DKWorkItemFed(svWF, "wf1", "node1", wfuser);
item->retrieve();
// ----- Call the checkOut method to lock the workitem
item->checkOut();
// ----- Call the checkIn method
item->checkIn(NULL);
delete item;

```

---

## Enumerar todos os modelos de fluxo de trabalho

Pode enumerar todos os modelos de fluxo de trabalho num serviço de fluxo de trabalho invocando a função `listWorkFlowTemplates()`.

O exemplo que se segue enumera o nome e a descrição de todos os modelos de fluxo de trabalho num serviço de fluxo de trabalho referenciado pelo objecto `svWF` do `DKWorkFlowServiceFed`.

### Exemplo: Java

```

// ----- Call the listWorkFlowTemplates method
DKSequentialCollection collWT =
 (DKSequentialCollection)svWF.listWorkFlowTemplates();
DKWorkFlowTemplateFed WT = null;
if (collWT != null)
{
 dkIterator iterWT = collWT.createIterator();
 while (iterWT.more() == true)
 {
 WT = (DKWorkFlowTemplateFed)iterWT.next();
 WT.retrieve();
 System.out.println("name = " + WT.name() + " description = "
 + WT.description());
 }
 iterWT = null;
}

```

## Exemplo: C++

```
// ----- Call the listWorkflowTemplates function
DKSequentialCollection *collWT =
 (DKSequentialCollection*)svWF.listWorkflowTemplates();
DKWorkflowTemplateFed *WT = NULL;
if (collWT != NULL)
{
 dkIterator* iterWT = collWT->createIterator();
 while (iterWT->more())
 {
 WT = (DKWorkflowTemplateFed*)(void*)(*iterWT->next());
 WT->retrieve();
 cout << "name = " << WT->name() << " description = "
 << WT->description() << endl;
 delete WT;
 }
 delete iterWT;
}
delete collWT;
```

---

## Criar acções próprias (apenas para Java)

Pode criar acções para utilizar num fluxo de trabalho utilizando objectos de acção (objectos `DKWorkflowActionFed`). Defina as acções e adicione-as a listas de acções na Administração do DB2 Information Integrator for Content.

Um objecto de acção é um contentor de metadados que regista instruções detalhadas relativas à forma como se pretende que uma tarefa específica seja executada no nó cliente. Os objectos de acção (contentores de metadados) só registam instruções; não iniciam a invocação das tarefas descritas nos metadados da acção.

As acções podem ser agrupadas numa lista de acções (`DKWorkflowActionListFed`). Um contentor de fluxo de trabalho contém o nome da lista de acções (não contém os conteúdos da lista de acções) na qual um conjunto de acções relacionadas com o artigo de trabalho está associado. Um cliente deve obter a lista de acções e, de seguida, iterar através das entradas (acções) da lista e reagir de acordo com as mesmas.

## Exemplo

O exemplo seguinte mostra como trabalhar com acções e listas de acções. O exemplo completa as seguintes tarefas:

1. Obtém o artigo de trabalho.
2. Obtém o contentor que é encaminhado juntamente com o artigo de trabalho.
3. Obtém a lista de acções do contentor.
4. Obtém a lista de acções da lista de acções.
5. Inicia as acções de acordo com as mesmas.

```
wit.retrieve(); // wit is a DKWorkItemFed object
DKWorkflowContainerFed wcn = wit.inContainer();
wcn.retrieve();
String alName = wcn.getActionList();
DKWorkflowActionListFed wal = new DKWorkflowActionListFed(dsFed);
wal.setName(alName);
wal.retrieve();
dkIterator iter = null;
if ((coll!=null) && (coll.cardinality()>0))
{
 iter = coll.createIterator();
```

```

while (iter.more ())
{
DKWorkflowActionFed act = (DKWorkflowActionFed) iter.next();
System.out.println("ACTION = " + act.getCommand());
Runtime.getRuntime().exec(act.getCommand());
}
}
else
System.out.println("NO ACTION DEFINED");

```

---

## Trabalhar com os JavaBeans do fluxo de trabalho do DB2 Information Integrator for Content

Os beans fornecem a funcionalidade de construção de aplicações de fluxo de trabalho. Por exemplo, pode utilizar JavaBeans nas aplicações para iniciar processos de fluxo de trabalho que seguem um caminho de execução predefinido previamente estabelecido pelo gestor de processos definido no servidor do Content Manager.

Esta secção descreve os JavaBeans do fluxo de trabalho do DB2 Information Integrator for Content que pode utilizar para ligar ao servidor do Content Manager Versão 8. Os beans e as excepções estão incluídos no pacote `com.ibm.mm.beans.wcm`. O utilizador pode utilizar os JavaBeans em construtores que os suportem (consulte a documentação do produto para o seu construtor para informações adicionais).

Um exemplo de uma utilização prática do JavaBeans é automatizar um processo de uma companhia de seguros online, como, por exemplo, emitir uma participação de sinistro. Ao utilizar o JavaBeans, pode automatizar todo o processo desde quando a participação é submetida até quando é processada.

### Pré-requisitos

É necessário dispor destes componentes instalados e configurados para poder trabalhar com os JavaBeans do Fluxo de trabalho do DB2 Information Integrator for Content.

- IBM WebSphere Application Server 4.0.4
- IBM DB2 Information Integrator for Content V 8.1, Fix Pack 1 e pré-requisitos
 

Tem de ter o DB2 Information Integrator for Content instalado com a opção de fluxo de trabalho e um servidor de LDAP para activar importação de utilizadores e autenticação de utilizadores.
- IBM Content Manager 8.1, Fix Pack 1 e pré-requisitos
 

Consulte o manual *Planeamento e Instalação do Sistema de Gestão de Conteúdos* para obter uma lista de servidores de LDAP e para obter instruções completas sobre a configuração do seu sistema para interagir com um dos servidores de LDAP.

O ID de utilizador seleccionado como o ID de utilizador de ligação do DB2 UDB (a predefinição é ICMCONCT) tem de ser devidamente configurado ao nível do sistema operativo e tem de ter o privilégio UserDB2TrustedConnect definido no seu sistema IBM Content Manager.
- O servidor de LDAP suportado pelo IBM Content Manager e WebSphere Application Server

### Configurar o modelo de dados exemplo

É necessário executar 9 passos para configurar o modelo de dados.

Os passos necessários para configurar o modelo de dados exemplo abaixo são os mesmos passos gerais que vai seguir para configurar o seu modelo de dados. Tem de completar os passos seguintes utilizando o cliente de administração do sistema do DB2 Content Manager.

1. Crie um privilégio intitulado `WCMPrivilegeSet` e adicione os privilégios seguintes:

- `AllowConnectToLogon`
- `ItemSQLSelect`
- `ItemTypeQuery`
- `ItemQuery`
- `ItemAdd`
- `ItemSetUserAttr`
- `ItemSetSysAttr`
- `ItemDelete`
- `ItemMove`
- `ItemLinkTo`
- `ItemLinked`
- `ItemAddLink`
- `ItemRemoveLink`
- `ItemCheckInOut`
- `ItemAddToDomain`
- `ItemGetWorkList`
- `ItemGetWork`
- `ItemRoute`
- `ItemRouteStart`
- `ItemRouteEnd`
-



- ItemUpdateWork
- 
- ItemGetAssignedWork
- 
- SystemDomainAdmin
- 
- SystemDomainQuery
- 
- SystemDefineUser
- 
- SystemGrantUserPrivs
- 
- SystemQueryUserPrivs
- 
- SystemDefineGroup
- 
- SystemQueryGroup
- 
- SystemDefinePrivs
- 
- SystemDefineDomain
- 
- SystemDefineACL
- 
- SystemDefineSemanticType
- 
- SystemSetACL
- 
- SystemDefineRM
- 
- SystemDefineXdoObject
- 
- SystemDefineSMSColl
- 
- SystemSetReplicaRule
- 
- SystemSetCtrlParm
- 
- SystemQueryOtherDomains
- 
- SystemDefineNLSLang
- 
- SystemBatchCompileACL
- 
- SystemDefineMimeType
-

- SystemManageKey
- 
- SystemDefineAttrs
- 
- SystemGetKey
- 
- SystemDefineItemType
- 
- SystemDefineNewKywdClass
- 
- SystemQueryAllKywdClass
- 
- SystemDefineLinkType
- 
- SystemQueryItemType
- 
- IKFAllPermissions
- 
- IKFCreateCatalog
- 
- IKFDeleteCatalog
- 
- IKFRetrieveCatalog
- 
- IKFUpdateCatalog
- 
- IKFCreateCategory
- 
- IKFRetrieveCategory
- 
- IKFUpdateCategory
- 
- IKFDeleteCategory
- 
- IKFCreateTrainingDoc
- 
- IKFRetrieveTrainingDoc
- 
- IKFUpdateTrainingDoc
- 
- IKFDeleteTrainingDoc
- 
- IKFCreateRecord
- 
- IKFRetrieveRecord
-

- IKFUpdateRecordIKFDeleteRecord
- 
- IKFRunServerTask
- 
- IKFRunAnalysisFunc
- 
- ClientScan
- 
- ClientPrint
- 
- ClientImport
- 
- ClientExport
- 
- ClientSendMail
- 
- ClientReceiveMail
- 
- ClientReadBasePart
- 
- ClientModifyBasePart
- 
- ClientAddNewBasePart
- 
- ClientDeleteBasePart
- 
- ClientReadAnnotation
- 
- ClientModifyAnnotation
- 
- ClientReadNoteLog
- 
- ClientAddToNoteLog
- 
- ClientModifyNoteLog
- 
- ClientReadHistory
- 
- ClientAdvancedSearch
- 
- ClientReadFolderContents
- 
- WFWorklist
- 
- EIPAdminServer
-

- EIPAdminEntity
  - EIPAdminTextEntity
  - EIPAdminTemplate
  - EIPAdminInfoMining
2. Crie os grupos de utilizadores na lista abaixo.
    - Participantes do fluxo de trabalho
    - Editor de conteúdos
    - Líder do projecto
    - Fornecedor de conteúdos
    - Perito dos domínios

**Nota:** Por agora, não adicione utilizadores aos grupos. Pode adicionar utilizadores aos grupos em passos posteriores.

3. Criar listas de controlo de acessos (ACLs) na tabela abaixo. Repare que deve criar uma ACL diferente para cada grupo de utilizadores que necessite dos mesmos privilégios de acesso.

*Tabela 36. Listas de Controlo de Acessos*

| nome da ACL            | Grupo                              | Conjunto de privilégios |
|------------------------|------------------------------------|-------------------------|
| WCMACL                 | Participantes do fluxo de trabalho | WCMPrivilegeSet         |
| Content PublisherACL   | Editor de conteúdos                | WCMPrivilegeSet         |
| Content ContributorACL | Fornecedor de conteúdos            | WCMPrivilegeSet         |
| Project LeadACL        | Líder do projecto                  | WCMPrivilegeSet         |

4. Criar os atributos, com as características apresentadas, na tabela abaixo. Deixe os valores predefinidos caso não estejam especificados abaixo.

*Tabela 37. Atributos*

| Nome (sensível a maiúsculas e minúsculas) | Tipo de atributo  | Mínimo | Máximo |
|-------------------------------------------|-------------------|--------|--------|
| WCM_Fields                                | Carácter Variável | 0      | 4,096  |
| WorkPackageACL                            | Carácter Variável | 0      | 500    |

Crie os tipos de artigos, com as características apresentadas, na tabela abaixo. Deixe os valores predefinidos caso não estejam especificados abaixo.

*Tabela 38. Tipos de artigos*

| Nome (sensível a maiúsculas e minúsculas) | Classificação do tipo de artigo | Controlo de Acessos | Atributos      |
|-------------------------------------------|---------------------------------|---------------------|----------------|
| WCM_Document                              | Artigo                          | WCMACL              | WCM_Fields     |
| WCM_Folder                                | Artigo                          | WCMACL              | WorkPackageACL |

**Notas:** WCMACL representa uma ACL que contém dos os utilizadores do fluxo de trabalho.

5. Crie um gestor de processos. Para criar um gestor de projectos, tem de designar nós e acções de trabalho que definam um novo processo. Um nó de trabalho é um passo num processo em que os artigos aguardam por acções que vão ser

tomadas por utilizadores finais ou aplicações, ou através do qual os artigos se movem automaticamente. Pode definir um processo de um passo, ou pode criar um processo com vários passos nele integrados.

**Restrição:** É necessário ter pelo menos um nó de trabalho definido para criar o processo.

a. Criar nós de trabalho

Crie os nós de trabalho, com as características apresentadas, na tabela abaixo. Deixe os valores predefinidos caso não estejam especificados na tabela.

*Tabela 39. Nós de trabalho*

| Nome (sensível a maiúsculas e minúsculas) | Descrição                     | Lista de Controlo de Acessos |
|-------------------------------------------|-------------------------------|------------------------------|
| Pedir alteração                           | Iniciar o pedido de alteração | WCMACL                       |
| Efectuar alteração                        | Alterar o conteúdo            | Content ContributorACL       |
| Rever Pedido                              | Rever o pedido de alteração   | Project LeadACL              |
| Rever Alteração                           | Aprovar a alteração           | Domain ExpertACL             |

b. Criar um gestor de processos

Crie o gestor de processos listado abaixo com as características demonstradas. Deixe os valores predefinidos onde não estejam especificados.

Nome: Processo de Alteração Simples Descrição: Modelo de processo para processo de alteração simples Lista de controlo de acessos: WCMACL

*Tabela 40. Gestor de processos*

| Do nó              | Seleção   | Para o nó          |
|--------------------|-----------|--------------------|
| START              | Continuar | Pedir alteração    |
| Pedir alteração    | Continuar | Rever Pedido       |
| Rever Pedido       | Aceitar   | Efectuar alteração |
| Rever Pedido       | Rejeitar  | END                |
| Efectuar alteração | Continuar | Rever Alteração    |
| Rever Alteração    | Aceitar   | END                |
| Rever Alteração    | Rejeitar  | Efectuar alteração |

6. Crie utilizadores. Deverá criar utilizadores num servidor LDAP, tal como o IBM Directory Server e configurar o IBM WebSphere Application Server para utilizar o servidor LDAP como mecanismo de autenticação no Security Center. Deverá definir os utilizadores no servidor Content Manager.

Crie os seguintes utilizadores no seu servidor LDAP e de seguida importe-os para o Content Manager:

*Tabela 41. Utilizadores para exemplo*

| Utilizador | Grupo                                                       |
|------------|-------------------------------------------------------------|
| "Cristina" | "Participantes de fluxo de trabalho", "Editor de conteúdos" |
| "Rui"      | "Participantes de fluxo de trabalho", "Líder do projecto"   |

Tabela 41. Utilizadores para exemplo (continuação)

| Utilizador  | Grupo                                                           |
|-------------|-----------------------------------------------------------------|
| "Guilherme" | "Participantes de fluxo de trabalho", "Fornecedor de conteúdos" |
| "David"     | "Participantes de fluxo de trabalho", "Fornecedor de conteúdos" |

Consulte a secção Creating users e siga as instruções opcionais de obtenção a partir de LDAP. O WebSphere Application Server utiliza o atributo de LDAP uid para autenticação. Por predefinição, o Content Manager utiliza o atributo de utilizador cn. Para fins de compatibilidade, deverá alterar a configuração de LDAP do Content Manager para utilizar uid como atributo de utilizador. Pode ainda definir o cn e o uid para o mesmo valor ao criar s utilizadores no LDAP.

7. (Opcional) Adicione atributos personalizados definidos pelo utilizador ao gestor de processos de fluxo de trabalho

Em seguida, é apresentado um código exemplo que demonstra como adicionar atributos personalizados ao gestor de processos de fluxo de trabalho:

```
java CustomAttributeTool -d datastore -u user -p password -w <worknode>
-n <attribute name> -v <attribute value> [...-n <attribute name> -v <attribute
valor>]
```

Onde

worknode - nome do nó de trabalho ao qual são adicionados os atributos personalizados. O nó de trabalho deve existir.

attribute name - nome definido do utilizador (este atributo não tem necessariamente de existir)

attribute value - o valor de Cadeia a associar ao nome de atributo.

Exemplo: java CustomAttributeTool -w Review Change -n WCM.Promote -v yes -n WCM.Publish -v WCM Publish

java CustomAttributeTool -h para mais opções.

O utilizador pode descarregar esta ferramenta no sítio de suporte da IBM na Web.

8. (Opcional) Adicione etiquetas de decisão ao gestor de processos de fluxo de trabalho

Abaixo encontra-se um código exemplo que demonstra como adicionar etiquetas de decisão ao seu gestor de processos de fluxo de trabalho.

```
java DecisionLabelTool -d datastore -u user -p password -w <worknode>
-l <decision label>
```

Onde

worknode - nome do nó de trabalho ao qual se adiciona a etiqueta de decisão. O nó de trabalho deve existir.

decision label - Cadeia de texto a ser mostrada como etiqueta de decisão.

Exemplo:

```
java DecisionLabelTool -w Review Change -l Was this change
implemented correctly?
```

```
java DecisionLabelTool -h for more options.
```

O utilizador pode descarregar esta ferramenta no sítio de suporte da IBM na Web.

9. Crie uma lista de trabalho. Crie a lista de trabalho abaixo apresentada com as características indicadas. Deixe os valores predefinidos caso não estejam especificados.

Nome: WCM\_WL (dependente de maiúsculas e minúsculas)

Lista de controlo de acessos: WCMACL

Nós: Adicionar todos os nós que estejam listados nos encaminhamentos do gestor de processos (Tabela 40 na página 302).

Configuração opcional exemplo: Adicione os seguintes nós de trabalho a partir do separador **Node (Nó): Request Change (Pedir alteração), Review Request (Rever pedido), Make Change (Efectuar alteração), Review Change (Rever alteração)**

Encaminhamento de documentos

Lista de trabalhos

Atributos

Criar utilizadores

Criar grupos de utilizadores

Criar conjuntos de privilégios

Criar listas de controlo de acesso

Definir um ponto de recolha

Tipos de artigos

## Utilizar JavaBeans de fluxo de trabalho na aplicação

Antes de invocar os beans, o utilizador deve configurar o nome do arquivo de dados na sessão.

Comece por utilizar os beans invocando-os de um servlet ou JSP que esteja em execução no Websphere Application Server. Tenha em atenção que existe um objecto `HttpServletRequest` (esta variável é denominada 'request' no código exemplo abaixo) associado à servlet.

Os JavaBeans do fluxo de trabalho do DB2 Information Integrator for Content iniciam sessão automaticamente no servidor do DB2 Content Manager quando o utilizador inicia qualquer dos beans de fluxo de trabalho ao invocar o construtor. O DB2 Content Manager utiliza o ID do utilizador e a respectiva palavra-passe do `LtpaToken` definidos pelos WebSphere Application Server. A sessão é reutilizada se o utilizador utilizar os beans de classes diferentes. A sua sessão será desligada automaticamente quando terminar sessão ou a sessão expirar.

### Exemplo

```
HttpSession session = request.getSession();
session.setAttribute("com.ibm.mm.beans.wcm.ICMServerName", "ICMNLADB");
```

No exemplo acima, `ICMNLADB` é o nome do arquivo de dados.

### Fragmentos de código exemplo

Existem tarefas comuns que pode executar utilizando os JavaBeans do fluxo de trabalho do DB2 Information Integrator for Content.

Estes fragmentos de código destinam-se apenas a referência, podendo não funcionar na sua aplicação caso seja utilizados tal como são apresentados abaixo.

### Criar um processo de fluxo de trabalho

Este fragmento de código de servlet cria e inicia um processo de fluxo de trabalho com base no gestor de processos 'Simple Change Process' com um parâmetros definido pelo utilizador denominado 'subject'.

```
// the request variable has the com.ibm.mm.beans.wcm.ICMServerName
//set to ICMNLADB
WorklistHandlerAccessBean wb = new WorklistHandlerAccessBean(
 new WorklistHandlerKey(),request);
```

```

java.util.Hashtable myFields = new java.util.Hashtable();
myFields.put("subject","my subject");
String processName="My test job";
ExtendedActivityAccessBean activity =
 wb.createAndStartProcessAndClaimFirstActivity("Simple Change
Process", processName, myFields);

```

### Listar os processos de fluxo de trabalho

```

//list all activities
Vector results = wb.getActivities("SELECT ACTIVITY, NAME, DESCRIPTION,
STATE, OWNER, starttime from allactivities", -1);
//list only activities I can claim
Vector results = wb.getActivities("SELECT ACTIVITY, NAME, DESCRIPTION,
STATE, OWNER, starttime FROM canclaim", -1);

```

### Listar informações acerca de um processo de fluxo de trabalho

```

String activityId = "90 3 ICM8 ICMNLSDB11 WORKPACKAGE58
26 A1001001A02K06B34535J3113018 A02K21B14457G092121 13 204";
ExtendedActivityAccessBean activity = new ExtendedActivityAccessBean(new
 ExecutionObjectKey(activityId),null);
System.out.println("start date: "+ activity.getStartDate());
System.out.println("unique job id: " +
 activity.getContainer().getJobId());
System.out.println("state: "+ activity.getState());
System.out.println("Process Model: " + activity.getContainer().
 getManager().getName());
System.out.println("Process name: " +
 activity.getContainer().getName());
ExtendedDocumentAccessBean document =
 activity.getBinder().getMainDocument();
System.out.println("subject:" + document.getField("subject"));
System.out.println("potential owners: ");
java.util.Vector potentialOwners = activity.getPotentialOwners();
java.util.Enumeration e = potentialOwners.elements();
while (e.hasMoreElements()){
 String s = e.nextElement().toString();
 System.out.println(s);
}

```

### Participar e avançar um processo de fluxo de trabalho

```

// the user running this must have authority to claim at this worknode
String activityId = "90 3 ICM8 ICMNLSDB11 WORKPACKAGE58 26
A1001001A02K06B34535J3113018 A02K21B14457G092121 13 204";
ExtendedActivityAccessBean activity = new ExtendedActivityAccessBean(new
 ExecutionObjectKey(activityId),null);
activity.claim();
// completing the activity advances to the next work node in the routing
// process list decision choices

System.out.println("decisionChoices: ");
java.util.Vector v = activity.getDecisionChoices("Decision");
e = v.elements();
while (e.hasMoreElements()){
 String s = e.nextElement().toString();
 System.out.println(s);
}

// (assuming node has more than one decision choice for this example)
// make a decision
activity.setDecision("Decision","Accept");
activity.complete();

```



## Configurar e obter campos num processo de fluxo de trabalho

```
Setting and getting fields in a workflow process
String activityId = "90 3 ICM8 icmnlbdb11 WORKPACKAGE58 26
A1001001A02I30B32323A1048018 A02J04A60100A986431 13 204";
ExtendedActivityAccessBean activity = new ExtendedActivityAccessBean(new
 ExecutionObjectKey(activityId),null);
BinderAccessBean binder = activity.getBinder();

 // getMainDocument()
ExtendedDocumentAccessBean document = binder.getMainDocument();
document.updateField("my field","my field test");
System.out.println("Successfully updated.");

// getField(String fieldName)
String fieldValue = document.getField("my field");
System.out.println("my field =" + fieldValue);
```

---

## Construir aplicações com JavaBeans visuais e não visuais

Pode utilizar os JavaBeans visuais e não visuais no DB2 Information Integrator for Content para construir aplicações com base em Java ou de cliente da Web.

Os JavaBeans do DB2 Information Integrator for Content podem ser divididos nas seguintes categorias:

### Beans não visuais

Pode utilizar os beans não visuais para construir aplicações de clientes Java e da Web que necessitem de uma interface de utilizador personalizada. Os beans não visuais suportam o modelo de programação de bean padrão facultando construtores, propriedades, eventos e uma interface seriável predefinidos. Pode utilizar os beans não visuais em ferramentas de construtor que suportam análise introspectiva.

### Beans visuais

Os beans visuais são componentes de interface gráfica de utilizador com base em Swing, que podem ser personalizados. Utilize os beans visuais para construir aplicações de Java para Windows. Pode colocá-los em janelas e caixas de diálogo de aplicações com base em Java. Devido ao facto dos beans visuais serem construídos recorrendo à utilização de beans não visuais (como, por exemplo, um modelo de dados) o utilizador deve utilizá-los juntamente com os beans não visuais ao construir uma aplicação.

---

## Compreender conceitos básicos de beans

Os beans são classes de Java que obedecem a convenções específicas relativas a definições de propriedades e interfaces de eventos. Cumprindo estas convenções, o utilizador pode transformar quase todos os componentes de programação ou classes de Java existentes num bean.

Os JavaBeans (adiante referidos como beans) são componentes de software reutilizáveis que são escritos na linguagem de programação de Java e que podem ser manipulados utilizando ferramentas de construtor com conhecimento de Beans. Devido ao facto dos beans poderem ser reutilizados, poderá utilizá-los para construir mais componentes complexos, construir novas aplicações ou adicionar funcionalidade às aplicações existentes. Poderá efectuar todos estes procedimentos visualmente, utilizando um construtor, ou manualmente, invocando os métodos de beans de um programa.

Os beans são classes de Java que obedecem a convenções específicas relativas a definições de propriedades e interfaces de eventos. Cumprindo estas convenções, o utilizador pode transformar quase todos os componentes de programação ou classes de Java existentes num bean.

Os beans definem uma interface na altura da concepção que permite a ferramentas de autor, ou de construtor, de aplicações, consultar componentes para determinar os tipos de propriedades que estes componentes definem e os tipos de eventos que geram ou aos quais respondem. Não tem de utilizar ferramentas de análise introspectiva e de construção especiais ao trabalhar com beans. As assinaturas padrão estão bem definidas e podem ser facilmente reconhecidas e compreendidas mediante observação visual.

Os beans possuem as seguintes características:

#### **Introspecção**

A análise introspectiva é o processo através do qual uma ferramenta de construtor determina e analisa como um bean funciona na altura da concepção e da execução. Devido ao facto dos beans serem codificados com padrões predeterminados para as suas assinaturas de métodos e definições de classe, as ferramentas que reconhecem estes padrões têm a capacidade de "ver o interior de" um bean e determinar as suas propriedades e comportamento. Cada bean possui uma classe de informações relacionadas com beans, que faculta informações relativas a propriedades, métodos e eventos sobre o próprio bean. Cada classe de informações de beans implementa uma interface **BeanInfo**, que lista explicitamente as funções de beans que serão expostas a ferramentas de construtor de aplicações.

#### **Propriedades**

As propriedades controlam a aparência e comportamento de um bean. As ferramentas de construtor analisam introspectivamente um bean de forma a descobrir as suas propriedades e a expor essas propriedades para manipulação. Isto permite ao utilizador alterar a propriedade de um bean na altura da concepção.

#### **Personalização**

As propriedades expostas de um bean podem ser personalizadas na altura da concepção. A personalização permite alterar a aparência e o comportamento de um bean. Os Beans suportam a personalização através da utilização de editores de propriedades ou utilizando personalizadores especiais e sofisticados de Beans.

#### **Eventos**

Os beans utilizam modelos de eventos de Java para comunicar com outros beans. Os beans podem desencadear eventos. Quando um bean desencadeia um evento, este é considerado um bean origem. Um bean pode também receber um evento, sendo neste caso considerado um bean receptor. Um bean receptor regista o seu interesse no evento junto do bean origem. As ferramentas de construtor utilizam a análise introspectiva para determinar os eventos que um bean envia e os eventos que recebe.

#### **Persistência**

Os beans utilizam serialização de objectos Java implementando a interface `java.io.Serializable` para guardar e repor estados que poderão ter sido alterados devido a uma personalização. Por exemplo, o estado é guardado quando uma aplicação personaliza um bean num construtor de aplicações, de modo a que as propriedades alteradas possam ser repostas numa altura posterior.

#### **Métodos**

Todos os métodos de bean são idênticos a métodos de outras classes de Java. Os métodos de bean podem ser invocados por outros beans ou através de linguagens de realização de scripts.

---

## **Utilizar JavaBeans em construtores**

Utilize JavaBeans no IBM Websphere Studio Application Developer e noutros construtores.

Para utilizar construtores que não o IBM Websphere StudioApplication Developer, certifique-se que o construtor suporta o nível de JDK requerido pelos beans.

Siga as instruções do construtor para adicionar novos ficheiros jar, de modo a adicionar os jars especificados nas instruções que se seguem. Em seguida, siga as instruções do construtor para adicionar beans a partir de um jar, de modo a adicionar os beans do DB2 Information Integrator for Content, que se encontram no ficheiro cmb81.jar.

**Nota:** Para utilizar os beans, é necessário ter o Java JDK 1.5.

O directório `%IBMCMROOT%\samples\java\beans` contém exemplos de código dos beans não visuais, enquanto directório `%IBMCMROOT%\samples\java\beans\gui` contém uma aplicação exemplo escrita através da utilização dos beans visuais e não visuais.

## Utilizar o IBM Websphere Studio Application Developer

Utilize beans não visuais para construir aplicações no WebSphere Studio Application Developer.

Para construir servlets e páginas de JSP através da utilização de beans não visuais no WebSphere Studio Application Developer, execute os seguintes passos:

1. Criar um projecto na Web para a aplicação da Web.
2. Nas propriedades do projecto, em Java Build Path | Libraries, especifique os seguintes ficheiros JAR:

```
%IBMCMROOT%\lib\cmb81.jar
%IBMCMROOT%\lib\cmbview81.jar
%IBMCMROOT%\lib\cmbsdk81.jar
\SQLLIB\java\db2jcc.jar
\SQLLIB\java\db2jcc_license_cu.jar
```

3. Se pretende utilizar os servlets do DB2 Information Integrator for Content e o taglib de JSP, deve também especificar os seguintes ficheiros:

```
%IBMCMROOT%\lib\cmbervlet81.jar
%IBMCMROOT%\lib\cmbtag81.jar
```

Para a biblioteca de identificadores, também será necessário importar o ficheiro taglib.tld para a biblioteca de identificadores de JSP do DB2 Information Integrator for Content para a aplicação da Web:

- Copie `%IBMCMROOT%\lib\taglib.tld` para o directório `webApplication\WEB-INF` na aplicação da Web.
- Configure a biblioteca de identificadores do ficheiro `webApplication\WEB-INF\web.xml` na aplicação da Web adicionando o seguinte:

```
<taglib>
<taglib-uri>cmb</taglib-uri>
<taglib-location>/WEB-INF/taglib.tld</taglib-location>
</taglib>
```

4. Uma vez que os JARs acima enumerados contêm classes de J2EE, o utilizador terá de incluir o JAR de J2EE, que normalmente está localizado no directório:  
`\Program Files\IBM\Application Developer\plugins\com.ibm.etools.websphere.runtime\lib\j2ee.jar`

---

## Invocar os JavaBeans do DB2 Information Integrator for Content

Pode invocar JavaBeans do DB2 Information Integrator for Content invocando-os directamente ou ligando instâncias de beans a outros beans.

Os beans no nível do DB2 Information Integrator for Content podem ser invocados de duas formas. O utilizador pode invocá-los directamente utilizando as suas interfaces públicas (métodos públicos). Neste caso, as excepções explícitas de Java são devolvidas para indicar eventos de erro.

Outro método através do qual pode invocar a funcionalidade nos beans a nível de sessão é ligar quaisquer instâncias deste tipo de bean a outros beans do DB2 Information Integrator for Content utilizando eventos de pedido e resposta. Ao utilizar este método, tenha em consideração o seguinte:

- O bean `CMBConnection` recebe eventos de pedido de ligação e responde transmitindo eventos de resposta de ligação.
- O bean `CMBDataManagement` recebe eventos de pedido de dados e transmite de volta eventos de resposta de dados.
- O bean `CMBSchemaManagement` recebe eventos de pedido de esquema e transmite de volta eventos de resposta de esquema.
- O bean `CMBQueryService` recebe eventos de pedido de pesquisa e transmite de volta eventos de resposta de pesquisa.
- O bean `CMBWorkflowDataManagement` recebe eventos de pedido de dados de fluxo de trabalho e transmite de volta eventos de resposta de dados de fluxo de trabalho.
- O bean `CMBWorkflowQueryService` recebe eventos de pedido de lista de trabalho e transmite de volta eventos de resposta de lista de trabalho.

---

## Trabalhar com os beans não visuais

O DB2 Information Integrator for Content faculta um conjunto de JavaBeans não visuais que pode utilizar para construir aplicações de Java.

Os beans não visuais são um conjunto de classes de Java que se regem pelas convenções de beans. Estes beans são construídos utilizando as classes de conector de Java do DB2 Information Integrator for Content e do DB2 Content Manager. Pode utilizar os beans para construir Servlets ou Java Server Pages (JSP), embora os beans também possam ser usados numa linha de comandos ou em aplicações Windows.

De seguida é apresentada uma lista das vantagens da utilização dos beans não visuais:

- Fornece um mecanismo de acesso federado e um modelo de programação comum para muitos dos conectores diferentes em comunicação com DB2 Information Integrator for Content.
- Permitem ao utilizador programar a um nível superior de abstracção.
- Ocultam a complexidade e detalhes de conectores individuais.
- Permitem-lhe potenciar o suporte de beans incorporado na maioria dos ambientes de desenvolvimento comercial.
- Facilitam a aplicação para utilizar vários conectores ou para migrar, sem ter de efectuar grandes alterações a essa aplicação.

A utilização dos beans facilita a construção de aplicações básicas. No entanto, existem algumas limitações na utilização dos beans. De seguida é apresentada uma lista das funcionalidades que os beans não facultam:

- Funcionalidade administrativa e de configuração.

- Importação de lotes: Os beans suportam apenas capacidades de importação de artigo único. Se precisa dos processos de lote para importar e exportar grandes quantidades de dados, tem de utilizar as interfaces de conector.
- Funcionalidade específica de servidor total para todos os conectores. A maioria dos beans tem de trabalhar em todos os tipos de servidor e, embora os beans incluam algumas funcionalidades específicas de servidor para determinados servidores, os beans não abrangem a funcionalidade total de cada servidor disponível a partir das APIs de Java.

As limitações acima enumeradas podem, em alguns casos, ser ultrapassadas utilizando métodos de acesso que permitem o acesso às APIs de Java de base.

**Importante:** Os beans do DB2 Information Integrator for Content não são Enterprise JavaBeans (EJB). Por conseguinte, não podem ser directamente detidos no ambiente gerido, facultado por contentores como o IBM Websphere. Contudo, podem ser utilizados a partir dos EJBs como mecanismo de ligação base para repositórios de dados não estruturados.

## Configurações de beans não visuais

Os beans não visuais têm configurações locais, remotas e dinâmicas.

No entanto, a configuração de beans não visuais é concluída ao nível do conector e os beans são transferidos através destas definições para o conector.

**locais** Liga directamente ao servidor de conteúdos.

**remotas**

Liga directamente ao servidor de conteúdos utilizando um servidor RMI.

**dinâmicas**

Activa uma aplicação que executa uma comutação dinâmica entre local e remoto com base no ficheiro cmbcs.ini. O ficheiro cmbcs.ini especifica se o servidor de conteúdos é local ou remoto.

## Compreender as funções dos beans não visuais

Os beans não visuais fornecem um conjunto de funções que permitem a utilização de beans em JSP.

Pode utilizar os beans do DB2 Information Integrator for Content em JSP, uma vez que as suas propriedades são, normalmente, tipos simples como, por exemplo, cadeias e matrizes. No fundo, funcionam como componente modelo de aplicações Web, pois são modelados utilizando o padrão de concepção do Model View Controller (MVC).

**Nota:** Tenha em atenção que o componente de visualização normalmente é formado por JSP e pelo componente controlador de servlets (como os do toolkit de servlet de EJB).

De seguida é apresentada uma lista das características dos beans não visuais do DB2 Information Integrator for Content:

- Facultam acesso às definições do esquema no servidor de bibliotecas.
- Facultam métodos CRUD (criar, obter, actualizar, eliminar) para documentos, artigos simples (que não são de recurso) e artigos de recurso em todos os repositórios suportados pelo DB2 Information Integrator for Content.

- Facultam funcionalidade para pesquisar e obter documentos, artigos simples e artigos de recurso em todos os repositórios suportados pelo DB2 Information Integrator for Content.
- Suportam a conversão de tipos de dados em formatos visualizáveis.
- Funcionam como um nível federado que aplica um conjunto coerente de semântica em todos os repositórios de gestão de conteúdos suportados pelo DB2 Information Integrator for Content.
- Integra e expõe a funcionalidade facultada nos serviços de fluxo de trabalho do DB2 Information Integrator for Content.
- Facultam serviços de extracção e conversão de documentos, bem como suporte para a gestão de anotações de documentos.
- Facultam a funcionalidade de ordenação e conversão.
- Facultam eventos que são transmitidos para acções chave nos beans membros como, por exemplo, eventos de ligação e de anulação de ligação, eventos de notificação de resultados de pesquisa e eventos de notificação de alteração de conteúdo.

## Categorias de beans não visuais

As categorias de beans não visuais incluem: de arquivo de dados, auxiliares, acessórios, beans de serviços XML e classes de ajudante, de fluxo de trabalho, de serviços de documento, entre outros.

Os JavaBeans não visuais podem ser divididos nas seguintes categorias:

### Beans de arquivo de dados

Estes beans existem em sessões de utilizador normais e facultam ao utilizador serviços especializados. Os beans a nível da sessão incluem:

#### **CMBConnection**

Este bean mantém a ligação a um servidor de fundo, que poderá ser um servidor de conteúdos nativo ou um servidor federado. Este bean é necessário para utilizar um JavaBeans.

#### **CMBSchemaManagement**

Utilizado para trabalhar com metadados de repositórios.

#### **CMBDataManagement**

Utilizado para trabalhar com dados de repositórios.

#### **CMBQueryService e CMBSearchResults**

Utilizado para executar consultas e trabalhar com os resultados de consultas.

#### **CMBWorkflowDataManagement e CMBWorkflowQueryService**

Utilizado para trabalhar com processos de fluxo de trabalho avançados do DB2 Information Integrator for Content.

#### **CMBDocRoutingDataManagement e CMBDocRoutingQueryService**

Utilizado para trabalhar com processos de encaminhamento de documentos do DB2 Content Manager Versão 8.

#### **CMBDocumentServices**

Utilizar para facultar conversão de documentos, manipulação de documentos e serviços de anotação.

### Beans auxiliares

Os beans auxiliares existem no contexto de um ou mais do que um bean a



nível da sessão e são primariamente utilizados para a encapsulamento de valores de dados e para facultar serviços aos beans a nível da sessão. Os beans auxiliares disponíveis incluem:

#### **CMBEntity**

Representa definições de artigos de dados disponíveis em repositórios de gestão de conteúdos. Por exemplo, para um repositório do DB2 Content Manager Versão 8, uma CMBEntity representa as definições de tipos de artigo e de componentes descendentes, enquanto que para um repositório do OnDemand, uma CMBEntity representa um grupo de aplicação. A CMBEntity é uma classe auxiliar do CMBSchemaManagement.

#### **CMBAttribute**

Representa definições de atributos nos repositórios. A CMBAttribute é uma classe auxiliar do CMBSchemaManagement.

#### **CMBSearchTemplate**

Representa um modelo de pesquisa federada. A CMBSearchTemplate é uma classe auxiliar do CMBSchemaManagement.

#### **CMBSTCriterion**

Representa um critério de dados que faz parte de um modelo de pesquisa federada. A CMBSTCriterion é uma classe auxiliar do CMBSchemaManagement.

#### **CMBItem**

Representa instâncias de documentos, artigos de recurso e artigos que não são de recurso. A CMBItem é uma classe auxiliar do CMBDDataManagement.

#### **CMBOject**

Representa instâncias de artigos de recurso, partes de base e partes de diário de notas. A CMBOject também é utilizada para representar atributos BLOB. A CMBOject é uma classe auxiliar do CMBDDataManagement.

#### **CMBAnnotation**

Representa instâncias de partes de anotação para repositórios do DB2 Content Manager Versão 8 e notas para repositórios do OnDemand.

#### **CMBPrivilege**

Faculta a funcionalidade necessária para obter informações relacionadas com privilégios de um repositório suportado por CM ou por DB2 Information Integrator for Content.

#### **Beans acessórios**

Os beans acessórios não são essenciais nas aplicações, mas podem ser úteis para aperfeiçoar a funcionalidade. De seguida é apresentada uma lista de beans acessórios.

#### **CMBConnectionPool**

Utilizado para agrupar beans CMBConnection. Os beans CMBConnection efectuem a translineação de instâncias do DKDatastore, cuja criação pode ser dispendiosa. A utilização do CMBConnectionPool permite a reutilização de ocorrências do bean CMBConnection e das ocorrências do DKDatastore das quais efectuem a translineação.



**CMBUserManagement**

Utilizado em ligações a repositórios federados para gerir as correlações de utilizadores federados a utilizadores de servidores de nativos.

**CMBExceptionSupport**

Faculta uma estrutura para o tratamento comum de eventos de excepção.

**CMBTraceLog**

Faculta uma estrutura para o tratamento comum de eventos de rastreio e faculta capacidades de recepção de eventos de rastreio transmitidos por outros beans.

**Bean de serviços XML e classes de ajudante**

O XML recebe mensagens e responde no formato de pedidos XML para executar várias operações no DB2 Content Manager, incluindo pesquisa, criação, actualização, colocação em lotes, exportação de artigos para o formato XML, exportação de definições de tipo de artigo como esquemas XML e execução de operações de encaminhamento de documentos. O bean de XML aumenta as capacidades do suporte XML na API para importação de artigos, exportação de artigos e operações de exportação de esquemas.

**CMBXMLServices**

Este bean é o componente base dos serviços da Web do Content Manager e partilha o respectivo formato de mensagens com os serviços da Web. O bean CMBXMLServices lê a configuração a partir do ficheiro cmbxmlservices.properties, em *%IBMCMROOT%/cmgmt*. Os pedidos XML enviados para este bean terão de estar em conformidade com o esquema definido no ficheiro cmbmessages.xsd, em *%IBMCMROOT%/config*.

**CMBXMLMessage**

Utilizado como classe de cápsulas para um documento XML para descrever um pedido ou responder aos beans. Contém tanto a origem XML do documento de pedido, como o conjunto de anexos associados à mensagem. O documento XML pode ser um ficheiro, uma cadeia, uma sequência de entradas ou um modelo de objecto de documento (DOM).

**CMBXMLAttachment**

Utilizado juntamente com a classe CMBXMLMessage para representar um anexo na mensagem XML. Este objecto inclui conteúdo de recurso para um objecto de recurso ou parte de documento necessária para cumprir a operação de pedido.

**Beans de fluxo de trabalho**

Os beans de fluxo de trabalho facultam serviços de fluxo de trabalho. Os serviços de fluxo de trabalho fornecidos pelo nível de beans de EIP suportam dois tipos de sistemas de fluxo de trabalho: Os serviços de fluxo de trabalho fornecido pelo nível de beans do DB2 Information Integrator for Content suportam dois tipos de sistemas de fluxo de trabalho: fluxo de trabalho avançado e encaminhamento de documentos. A funcionalidade de fluxo de trabalho avançado utiliza o MQSeries Workflow, enquanto o encaminhamento de documentos consiste num sistema de fluxo de trabalho integrado no produto ~DB2 Content Manager Versão 8 e no conjunto de APIs. O nível de beans faculta o conjunto completo de objectos necessários à criação de definições de fluxo de trabalho, à execução de instâncias de fluxo de trabalho com base nas definições criadas e à gestão

de instâncias de fluxos de trabalhos em execução. Os beans seguintes constituem os principais componentes do suporte de fluxo de trabalho avançado:

#### **CMBWorkflowDataManagement**

Este bean é utilizado para criar e trabalhar com instâncias de fluxo de trabalho avançado. Pode ser obtida uma instância deste bean a partir do objecto CMBCConnection. Este bean faculta as seguintes funcionalidades:

- Início, paragem, suspensão e retoma de uma instância de fluxo de trabalho.
- Transferência de artigos e notificações de trabalho de um utilizador para outro.
- Cancelamento de notificações de trabalho.

#### **CMBWorkflowQueryService**

O CMBWorkflowQueryService fornece uma interface para consultar informações relacionadas com fluxos de trabalho avançados. Pode ser obtida uma instância deste bean a partir do objecto CMBCConnection. Este bean faculta suporte para a obtenção das seguintes informações:

- Informações relativas a fluxos de trabalho no sistema.
- Informações relativas a artigos de trabalho que se deslocam no sistema como parte de fluxos de trabalho activos.
- Informações relacionadas com listas de trabalho.
- Informações relativas a todas as notificações de trabalho registadas.

Os beans seguintes constituem os principais componentes do suporte de encaminhamento de documentos.

#### **CMBDocRoutingManagementICM**

Este bean é utilizado para criar e gerir processos de encaminhamento de documentos. Pode obter uma instância deste bean a partir de uma instância do objecto CMBCConnection. Este bean faculta suporte para as seguintes funções:

- Início, paragem, suspensão e retoma de uma instância de encaminhamento de documentos.
- Dar saída de um artigo de CM contido num pacote de trabalhos.
- Definição de propriedades de pacotes de trabalho a serem enviados por uma instância de encaminhamento de documentos.,

#### **CMBDocRoutingQueryServiceICM**

Este bean faculta uma interface para consultar informações facultadas por processos de encaminhamento de documentos. Pode obter uma instância deste bean a partir de uma instância do objecto CMBCConnection. Este bean faculta suporte para a obtenção das seguintes informações:

- Informações relacionadas com os pacotes de trabalho a serem encaminhados através do sistema de encaminhamento de documentos.
- Informações relacionadas com os processos que actualmente se encontram activos no sistema.

- Informações relativas a todas as listas de trabalho que se encontram no sistema.
- Todos os nós de trabalho que fazem parte do sistema.

### **beans de Serviços de Documento**

Os beans de serviços de documento facultam serviços de emissão e anotação de documentos. Os seguintes beans e classes são oriundos do toolkit do visualizador de Java.

#### **CMBDocumentServices**

O bean CMBDocumentServices fornece serviços necessários ao trabalhar com documentos, incluindo a funcionalidade necessária produzir, converter e reconstituir as páginas de um ou mais documentos.

#### **CMBDocument**

Esta classe representa a entidade criada ao carregar um documento utilizando CMBDocumentServices. Essencialmente, o CMBDocument é um contentor para as páginas do documento. O CMBDocument permite também consultar e definir propriedades que controlam as características de um documento.

#### **CMBPage**

Esta classe faculta uma representação de uma página específica num documento. A funcionalidade desta classe permite especificar e controlar as propriedades de um conjunto de imagens apresentáveis que pode ser gerado para a página.

#### **CMBPageAnnotation**

Esta classe modela uma anotação que pode ser associada a uma página de um documento. Todas as notações suportadas são modeladas por subclasses deste bean. Além disso, o CMBPageAnnotation contém propriedades como, por exemplo, a página em que se encontra a anotação e o tipo de anotação. As subclasses incluem:

- CMBArrowAnnotation
- CMBCircleAnnotation
- CMBHighlightAnnotation
- CMBLineAnnotation
- CMBNoteAnnotation
- CMBPenAnnotation
- CMBRectAnnotation
- CMBStampAnnotation
- CMBTextAnnotation

### **Outras classes de beans**

As classes de beans enumeradas nesta secção facultam uma variedade de funcionalidades, que são descritas de seguida.

#### **Classes de BeanInfo**

Permitem a exposição explícita das funções dos beans do DB2 Information Integrator for Content numa classe separada e associada que implementa a interface BeanInfo.

#### **Classes de excepção**

Utilizadas para encapsular excepções no nível de beans. Todas as classes de excepção herdam da classe base CMBException. Cada

uma das subclasses do `CMBException` indica uma condição de erro específica. Em cada caso, as propriedades do objecto de excepção podem ser utilizadas para obter informações de erro detalhadas relativas à condição de erro que conduziu à excepção. Quando os beans são utilizados de um modo direccionado a eventos, as excepções são enviadas como eventos.

#### **Classes de eventos e de receptor**

Implementam o modelo padrão de receptor de eventos de beans. As classes têm de pedir explicitamente um evento, implementando a interface receptora associada ao evento e registando essa interface receptora junto do objecto que gera um evento. O nível de beans de DB2 Information Integrator for Content faculta pares de eventos e de receptores para operações de acesso a esquema, operações de acesso a dados, operações de fluxo de trabalho e operações de pesquisa.

#### **Receptores de sessão**

Estas são as classes de receptores que existem a nível da sessão. Nos beans de DB2 Information Integrator for Content, existem receptores de sessão actuais que identificam pedidos de ligação e eventos de resposta de ligação.

## **Considerações a ter ao utilizar beans não visuais**

Pode utilizar os beans não visuais para activar aplicações com fins gerais com a funcionalidade necessária para aceder a repositórios de gestão de conteúdos suportados pelo DB2 Information Integrator for Content.

Tenha em consideração as seguintes sugestões sobre padrões específicos de utilização nos beans:

#### **Singletons nos beans**

O `CMBConnection` possui métodos para obter acesso a instâncias dos outros beans do DB2 Information Integrator for Content a nível da sessão. Quando se obtêm desta forma os beans a nível da sessão como, por exemplo, `CMBSchemaManagement` e `CMBDataManagement`, estes já se encontram ligados ao bean `CMBConnection` (a partir do qual foram obtidos) para serem informados de uma ligação ou anulação de ligação e para partilhar operadores de eventos de rastreio e excepção. É criada apenas uma instância de cada um dos beans a nível de sessão. Se estes métodos forem repetidamente invocados, é devolvida a mesma instância (padrão de concepção singleton). Se forem criados beans a nível de sessão na aplicação e não pelo bean `CMBConnection`, estes terão de ser ligados ao bean `CMBConnection` para serem utilizados.

#### **Considerações relativas à modulação nos beans**

Uma instância única do bean `CMBConnection` apenas pode ser utilizada num módulo único em qualquer altura. Esta restrição estende-se a todos os outros beans associados a um bean `CMBConnection` (através da propriedade de ligação do bean associado). Isso significa que deve criar ligações diferentes para cada módulo. Opcionalmente, vários módulos podem obter e libertar ligações utilizando o bean `CMBConnectionPool`. Consequentemente, cada módulo deve obter, utilizar e libertar uma ligação.

Todos os beans a nível de sessão possuem afinidades com uma instância do `CMBConnection` a partir da qual foram obtidos ou à qual foram associados após a criação. Isto implica que uma instância dos beans a nível

de sessão como, por exemplo, o CMBSchemaManagement, apenas pode ser utilizada por um único módulo em qualquer altura. Se a instância do bean a nível de sessão for utilizada por vários módulos, deve realizar uma sincronização explícita na sua aplicação, de modo a garantir que apenas um módulo utiliza activamente a instância de bean a nível de sessão em qualquer altura.

Todos os beans a nível de sessão recebem também eventos de resposta de ligação gerados pelos beans CMBConnection. Isto permite-lhes reconhecer que o repositório de conteúdos de base ao qual a ocorrência do bean CMBConnection está associada foi ligado ou desligado, de modo a que os beans possam aplicar a acção adequada.

**Importante:** Os eventos de resposta de ligação apenas são enviados quando o arquivo de dados é ligado ou desligado como resultado de um método invocado na ocorrência do bean CMBConnection.

Ao contrário do bean CMBConnection, o bean CMBConnectionPool destina-se a ser utilizado por vários módulos. Vários módulos podem invocar simultaneamente os métodos relacionados com a obtenção e libertação de objectos de ligação. Todas as ligações obtidas a partir do conjunto são instâncias do CMBConnection e só podem ser acedidas por um único módulo. Todas as ligações obtidas a partir do bean de conjunto de ligações devem ser devolvidas ao conjunto o mais depressa possível após a sua utilização, de modo a que sejam disponibilizadas a outros módulos que possam estar a pedir ligações do conjunto.

## O bean CMBConnectionPool

O bean CMBConnectionPool mantém um conjunto de instâncias activas e ligadas do CMBConnection.

Os beans CMBConnection efectuem a translineação de instâncias do DKDatastore. Deste modo, o conjunto é, com efeito, um conjunto de objectos do DKDatastore. Uma vez que as instâncias do arquivo de dados suportadas pelo conjunto são ligadas e permanecem ligadas enquanto são geridas pelo conjunto, cada instância é associada a um ID de utilizador e palavra-passe específicos que tenham sido utilizados para criar a instância do arquivo de dados. O conjunto de ligações apenas devolverá um objecto de ligação pré-existente no conjunto se o novo pedido de ligação corresponder exactamente aos dados do ID de utilizador e palavra-passe. Se não existir nenhuma ligação disponível, o conjunto cria uma nova ligação para esse ID de utilizador. O conjunto não tenta desligar ou reutilizar uma das ligações inactivas do conjunto que estejam ligadas a credenciais de outro utilizador.

### Utilização do conjunto de ligações

Pode utilizar o conjunto de ligações se a ligação tiver sido concebida de modo a que durante a respectiva utilização, o mesmo ID de utilizador efectue vários pedidos de um objecto de ligação e devolva a ligação ao conjunto após a execução de algumas tarefas com a ligação.

Se a aplicação tiver sido concebida de forma a que um único ID de utilizador nunca efectue um pedidos de ligação mais do que uma vez por sessão, não existe qualquer vantagem em utilizar um conjunto de ligações.

## Colocação de módulos em fila de espera

O bean `CMBConnectionPool` tem duas opções quando não é encontrada qualquer ligação correspondente no conjunto para um pedido de ligação e não existe espaço para criar um novo objecto de ligação. Este comportamento é controlado pela propriedade `maxConnectionsBehavior`. O valor predefinido para esta propriedade é `CMBConnectionPool.CMB_MAX_CONNECTIONS_ERROR`. Quando não é encontrada qualquer ligação no conjunto que corresponda às informações de credencial fornecidas num pedido `getConnection` e se não existir espaço suficiente para criar uma nova ligação para o utilizador, o `CMBConnectionPool` devolve um erro. A outra definição possível para esta propriedade é `CMBConnectionPool.CMB_MAX_CONNECTIONS_QUEUE`, que activa a colocação de módulos em fila de espera. Quando não é encontrada qualquer ligação e não existe espaço suficiente no conjunto para criar uma ligação, o módulo que está a efectuar o pedido de ligação é colocado em estado de espera. O módulo aguarda até que uma das seguintes situações ocorra:

- Uma ligação com credenciais correspondentes seja libertada no conjunto por outro módulo. Neste caso o primeiro módulo em espera com credenciais correspondentes à ligação libertada é activado e a ligação libertada é devolvida a esse módulo.
- O tempo de espera do módulo seja excedido. Pode especificar o valor máximo de tempo que um módulo aguardará por uma ligação disponível definindo a propriedade `maxThreadWaitTime` no `CMBConnectionPool`. Se o módulo estiver em espera por um período superior ao tempo de espera e continuar a não existir nenhuma ligação disponível para atender ao pedido `getConnection`, o pedido será devolvido com uma excepção.
- Uma ligação que está a ser libertada no conjunto tenha de ser destruída. Quando a recolocação de uma ligação no conjunto viola um dos três parâmetros de optimização de ligações livres, essa ligação é destruída em vez de ser devolvida ao conjunto. Se existirem módulos em espera, o primeiro módulo em espera é activado e é criada uma nova ligação para esse módulo.
- Uma das ligações no conjunto livre esgote o tempo de espera e tenha de ser destruída. Pode especificar o período máximo de tempo durante o qual uma ligação pode permanecer inactiva no conjunto livre definindo a propriedade `maxWaitTime` no `CMBConnectionPool`. Qualquer ligações que permaneça inactiva no conjunto livre durante um período superior ao `maxWaitTime` é destruída. No momento em que a ligação expirada é destruída, se existirem módulos em espera, é criada uma nova ligação para o primeiro módulo em espera.

Dependendo da utilização da aplicação e das definições de optimização no conjunto, é possível que um módulo aguarde indefinidamente por uma ligação sem nunca receber uma nova ligação. Esta situação não representa necessariamente um erro no conjunto mas poderá indicar uma situação na qual os parâmetros de optimização definidos no conjunto não garantem que todos os utilizadores possam obter uma ligação num período de tempo razoável. Por exemplo, se definir a propriedade `maxConnections` como 10 e 10 utilizadores obtiverem ligações do conjunto sem nunca as devolverem, todos os pedidos subsequentes efectuados ao conjunto irão aguardar até esgotarem o tempo limite, uma vez que não existem ligações a serem libertadas nem existe espaço no conjunto para criar mais ligações.

## Opções de optimização do `CMBConnectionPool`

A manutenção de um conjunto de `DKDatastores` pode utilizar uma grande quantidade de memória. O bean `CMBConnectionPool` fornece várias propriedades



para ajudar os utilizadores a otimizar a utilização de memória de forma a cumprir da melhor forma os requisitos específicos da aplicação. As definições ideais para estes parâmetros variam de acordo com o ambiente específico e a utilização da aplicação. A forma mais rápida de obter uma ligação do conjunto é localizar uma ligação existente no conjunto livre que possa ser reutilizada para o pedido `getConnection`. Os utilizadores terão de esforçar-se para definir os parâmetros de optimização de modo a encorajar a reutilização máxima de ligações e para minimizar a necessidade de destruir ligações e criar novas ligações para atender ao pedido `getConnection`.

As seguintes propriedades do conjunto de ligações são utilizadas para controlar a marca de memória geral do conjunto e para gerir a quantidade de tempo que cada módulo deverá aguardar uma ligação.

#### **maxConnections**

A propriedade `maxConnections` controla o número total de ligações geridas pelo conjunto. Este é o total de todas as ligações em utilização e de todas as ligações disponíveis que se encontram no conjunto livre. Esta definição pode ser utilizada para impedir que o conjunto se torne demasiado grande para os limites de memória do sistema. Se não existir um valor máximo de número total de ligações geridas pelo conjunto, pode definir o valor como `CMBConnectionPool.CMB_NO_MAX`. O valor predefinido é 100. Defina o valor `maxConnections` como, no mínimo, 2 a 3 vezes o número de IDs de utilizador diferentes que irão aceder ao conjunto. Se tiver apenas um ID de utilizador partilhado por vários utilizadores, defina o valor `maxConnections` como 2 a 3 o número de utilizadores diferentes que partilham o ID de utilizador. A definição de um valor demasiado baixo para a propriedade `maxConnections` resultará num desempenho fraco, num aumento das falhas em pedidos `getConnection` e, possivelmente, em longas esperas dos utilizadores quando a colocação de módulos em fila de espera está activada.

#### **maxConnectionsPerUserid**

A propriedade `maxConnectionsPerUserid` controla o número total de ligações por ID de utilizador gerido pelo conjunto. O valor predefinido é `CMBConnectionPool.CMB_NO_MAX`. Este corresponde ao total de ligações associadas a esse ID de utilizador que estão a ser utilizadas actualmente mais as ligações disponíveis no conjunto livre associadas a esse ID de utilizador. Esta definição é utilizada para impedir que um ID de utilizador preencha o conjunto com ligações para esse ID de utilizador, deixando pouco espaço para pedidos de outros IDs de utilizador. Quando a utilização do conjunto é extremamente desequilibrada, como, por exemplo, 80 por cento de pedidos de um ID de utilizador e os outros 20 de uma variedade de IDs de utilizador, os outros IDs de utilizador poderão deparar-se com mais falhas na obtenção de ligações ou longas esperas por ligações se a colocação dos módulos em fila de espera estiver activada. A limitação do número de ligações que um ID de utilizador pode criar é uma forma de garantir melhores tempos de resposta para os outros IDs de utilizador. No entanto, uma utilização tão desequilibrada do conjunto não é comum, e, por isso, este parâmetro não é, normalmente, útil.

**Importante:** A definição do valor `maxConnectionsPerUserid` como algo diferente de `CMB_NO_MAX` terá um impacto negativo no desempenho, uma vez que as despesas gerais de pesquisa do conjunto para fazer cumprir este máximo são significativas. Deverá utilizar esta definição apenas se o teste ao sistema indicar que as vantagens de impedir que um ID de utilizador

preencha o conjunto superam o impacto negativo no desempenho que advém do cumprimento desta definição.

#### **maxFreeConnections**

A propriedade `maxFreeConnections` controla o número total de ligações permitidas no conjunto livre, que corresponde ao conjunto de ligações que são devolvidas ao `CMBConnectionPool` ao invocar o método `freeConnection`. Esta definição é utilizada para controlar a quantidade de memória absorvida pelo conjunto quando este se encontra inactivo. Após o limite no conjunto livre ter sido alcançado, a libertação de ligações adicionais resulta na destruição de ligações no conjunto livre para evitar que o limite seja ultrapassado. A definição de um valor demasiado baixo pode invalidar a utilidade do conjunto, causando a destruição e reutilização de ligações excessiva e impedindo a reutilização de ligações. Este parâmetro pode ser útil para limitar a quantidade de memória que pode ser utilizada pelo conjunto quando este está inactivo. Poderá optar por definir um limite bastante elevado ou não definir qualquer limite no número máximo de ligações, de modo a que todos os utilizadores possa ser servidos durante picos de utilização. No entanto, após o pico de utilização, o conjunto manterá um grande número de ligações na memória. Este parâmetro pode ajudar a controlar o número de ligações não utilizadas que o conjunto mantém, permitindo a redução da marca de memória do conjunto. Não defina o limite como um valor demasiado baixo de modo a que o conjunto não seja negativamente afectado.

#### **maxThreadWaitTime**

A propriedade `maxThreadWaitTime` aplica-se apenas quando a colocação de modos em fila de espera está activada. Esta propriedade controla o período máximo de tempo que qualquer módulo irá aguardar para que seja disponibilizada uma ligação na fila. Após o fim do tempo de espera, se ainda não existirem ligações disponíveis, o pedido `getConnection` para esse módulo será devolvido com uma `CMBGetConnectionFailedException` e um ID de erro com indicação de que tempo limite foi esgotado. O valor predefinido é `CMBConnectionPool.CMB_NO_TIMEOUT`, o que significa que, por predefinição, os módulos aguardam indefinidamente que seja disponibilizada uma ligação.

#### **timeOut**

A propriedade `timeOut` controla o período de tempo durante o qual uma ligação pode permanecer inactiva no conjunto livre antes de ser desligada e destruída. Por predefinição, o valor é `CMBConnectionPool.CMB_NO_TIMEOUT`, o que significa que as ligações podem permanecer indefinidamente no conjunto livre. A definição é reforçada por um módulo de tempo limite que é emitido em intervalos regulares para controlar o tempo que cada ligação livre esteve inactiva. A propriedade `timeOut` é bastante útil quando definida de modo a coincidir com o período de tempo que a ligação de base de dados subjacente pode permanecer inactiva antes de se tornar inválida. Se estiver a estabelecer ligação a um arquivo de dados que esteja implementado de tal forma que não exista qualquer tipo de preocupação relativamente a que ligação de base de dados subjacente se torne inválida, não é necessário utilizar esta definição. Se estiver a agrupar ligações do Content Manager V8 e tiver o agrupamento de ligações do Websphere Application Server activado para este arquivo de dados, não deverá definir esta propriedade para qualquer valor diferente de `CMBConnectionPool.CMB_NO_TIMEOUT`. A definição da propriedade como qualquer outro valor causa problemas de desempenho e não tem qualquer



benefícios, uma vez que o agrupamento de ligações do Websphere Application Server no arquivo de dados elimina o tempo limite de ligações de base de dados subjacentes.

## Nomes de apresentação

No conector do DB2 Content Manager Versão 8.4, pode traduzir os nomes de apresentação para um conjunto completo de idiomas.

Os nomes de apresentação são nomes alternativos que pode atribuir às entidades e aos atributos para os tornar mais descritivos. Por exemplo, pode atribuir um nome de apresentação de Insurance documents a um tipo de artigo denominado Doc 26 (Doc 26 será considerado o nome que não é para apresentação).

Para especificar qual o idioma de cujos nomes de apresentação pretende visualizar na aplicação, invoque `CMBConnection.setLanguageCode(String languageCode)`. Os nomes de apresentação devolvidos pelos beans serão traduzidos para o idioma que especificar nesta propriedade `languageCode`.

Tanto `CMBEntity` como `CMBAttribute` contêm três métodos para obter o nome e o nome de apresentação:

### **`getNonDisplayName()`**

Devolve sempre o nome real que não se destina a apresentação.

### **`getDisplayName()`**

Devolve sempre o nome de apresentação traduzido no idioma especificado pela propriedade `languageCode` do bean `CMBConnection`.

### **`getName()`**

Se `displayNamesEnabled` estiver definida como `false` (que é a predefinição), `getName()` devolve o nome que não se destina a apresentação. Se `displayNamesEnabled` estiver definido como `true`, `getName()` devolve o nome de apresentação (traduzido no idioma especificado pela propriedade `languageCode` do bean `CMBConnection`).

## Rastrear e iniciar sessão nos beans

Pode activar o rastreio em todos os beans a nível de sessão no nível de beans de DB2 Information Integrator for Content.

Activar o rastreio numa instância do bean `CMBConnection` permite também o rastreio em qualquer bean do DB2 Information Integrator for Content obtido a partir deste bean de ligação, incluindo beans de gestão de esquemas, gestão de dados, serviço de consulta e fluxo de trabalho. Para que o rastreio em beans produza efeito, é necessário activar o rastreio de beans no ficheiro `cmblogconfig.properties`, em `%IBMCMROOT%\cmgmt\connectors`. As definições de rastreio de beans estão localizadas na secção 6 deste ficheiro.

Após a activação do rastreio, são transmitidos eventos de rastreio. O `CMBTraceLog`, um bean de utilitário, pode receber eventos de rastreio e escrever registos de rastreio num registo, `stdout`, `stderr` ou janela definidos (quando utilizado com os beans visuais).

Todos os beans a nível de sessão também recebem eventos de rastreio. A funcionalidade de rastreio nos beans escreve informações num registo de rastreio

de beans, localizado, por predefinição, em `%IBMCMROOT%\log\beans`. A localização e o nome deste registo podem ser modificados alterando as definições no ficheiro `cmblogconfig.properties`.

## Compreender propriedades e eventos de beans não visuais

Cada bean não visual fornece um determinado número de propriedades e eventos.

Cada bean não visual fornece o seguinte

### Propriedades Importadas

O valor da propriedade é determinado por outros beans na altura do arranque por eventos `PropertyChange` ou `VetoableChange`. Os beans que possuem propriedades de importação terão de detectar eventos `PropertyChange` ou `VetoableChange`.

### Propriedades exportadas, proibidas ou não

Um bean não visual terá de possuir uma propriedade reduzida e outros beans poderão ter interesse no seu valor. Sempre que o valor é alterado, o bean é responsável pela geração de um evento `PropertyChange` ou `VetoableChange`.

### Propriedades autónomas

Estas propriedades são definidas e lidas apenas através dos métodos `setter` e `getter` nos beans. Não são accionados quaisquer eventos na definição destas propriedades.

Cada bean não visual fornece o seguinte

- Notifica os receptores de eventos gerados por este bean. Por exemplo, o `CMBConnectionBean` acciona automaticamente `CMBConnectionReplyEvents` para todos os receptores registados quando os métodos `connect()` ou `disconnect()` são invocados no bean.
- A capacidade de receber eventos nos quais este bean está interessado. Por exemplo, o bean `CMBSearchResults` possui um receptor incorporado para `CMBSearchReplyEvents`, pelo que quando é adicionado como receptor de uma instância do `CMBQueryServiceBean`, que é a origem dos `CMBSearchReplyEvents`, pode processar automaticamente os `searchReplyEvents` accionados por esse bean.

## Construir uma aplicação através de beans não visuais

Pode utilizar beans não visuais para construir uma aplicação.

### Aplicação não de Interface Gráfica de Utilizador (GUI) exemplo

O exemplo nesta secção utiliza beans não visuais para criar um exemplo de aplicação que não é GUI.

A aplicação exemplo inclui todos os beans excepto os beans `CMBUserManagement` e `CMBXMLServices`. A aplicação exemplo completa da qual foi retirado este exemplo (`DemoSimpleAppl.java`) está disponível no directório `Samples`. A aplicação exemplificativa demonstra como:

1. Ligar ao servidor (federado) do DB2 Information Integrator for Content
2. Obter uma lista dos nomes de modelos de pesquisas
3. Utilizar o nome dos modelos de pesquisas para obter uma lista de nomes de critérios de pesquisa
4. Seleccionar um modelo de pesquisa e obter os critérios de pesquisa

5. Concluir os valores de pesquisa e submeter uma consulta
6. Imprimir o resultado através do bean de resultados da pesquisa
7. Seleccionar uma fila de resultados e apresentá-la
8. Desligar do servidor

---

## Trabalhar com beans visuais

Os beans visuais permitem ao utilizador integrar a funcionalidade do DB2 Information Integrator for Content ou outros servidores de conteúdos nas aplicações de Java baseadas no Swing.

Os beans visuais executam tarefas básicas que são comuns a muitas aplicações, tais como iniciar sessão, pesquisar, apresentar e visualizar resultados, actualizar documentos e visualizar informações sobre a versão.

Cada bean visual tem uma propriedade Connection. Esta propriedade tem de referenciar uma instância do CMBCConnection, o bean não visual que mantém a ligação aos servidores de conteúdos. Uma aplicação construída com os beans visuais do DB2 Information Integrator for Content tem de conter uma instância do bean não visual CMBCConnection.

### Bean CMBLogonPanel

Este bean apresenta um painel para iniciar sessão no DB2 Information Integrator for Content ou em servidores de conteúdos, tais como o DB2 Content Manager Versão 8.4. Fornece ainda a janela na qual os utilizadores federados podem modificar os IDs de utilizador e as palavras-passe nos servidores de conteúdos.

O bean CMBLogonPanel (consultar Figura 15) apresenta uma janela que permite aos utilizadores iniciar sessão num servidor de conteúdos, actualizar correlações de utilizadores e alterar uma palavra-passe.



*Figura 15. Janela do bean CMBLogonPanel*

No bean CMBLogonPanel, quando um utilizador faz clique em **Change (Alterar)**, a janela Change Password (Alterar Palavra-passe) é apresentada (consultar Figura 16 na página 325). O utilizador insere a palavra-passe antiga e insere duas vezes a nova palavra-passe.



Figura 16. Janela Alterar Palavra Passe

No bean CMBLogonPanel, quando um utilizador faz clique em **Update Mapping (Actualizar Correlação)** na janela Logon (Iniciar Sessão), a janela Update Userid Mapping (Actualizar Correlação de ID de Utilizador) é apresentada (consultar Figura 17). Ao fazer clique em **Update Mapping (Actualizar Correlação)**, está a actualizar o ID de utilizador e palavra-passe especificada para um servidor. Esta função está disponível apenas quando inicia sessão na base de dados associada do DB2 Information Integrator for Content.

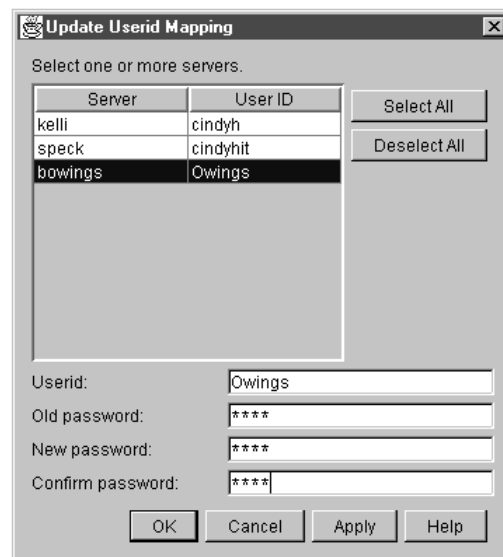


Figura 17. Janela de Correlação de ID de Utilizador

No topo da janela está uma lista de todos os servidores e um ID de utilizador correspondente. Os utilizadores podem seleccionar um ou mais servidores da lista. Faça clique em **Select All (Seleccionar Todos)** para seleccionar todos os servidores. Os utilizadores podem especificar um novo ID de utilizador e (opcionalmente) uma nova palavra-passe após a selecção de um ou mais servidores. Caso seleccione um servidor, o ID de utilizador surge no campo **Userid (ID de Utilizador)**. Se os utilizadores seleccionarem mais do que um ID de utilizador, o campo **Userid (ID de Utilizador)** fica em branco.

#### Anular todas as selecções

Remove todas as selecções do servidor.

**Aplicar**

Faça clique para aplicar as alterações de correlações e de palavra-passe sem fechar a janela.

**OK**

Faça clique para aceitar as alterações e fechar a janela.

**Cancelar**

Faça clique para fechar a janela sem efectuar alterações.

## Bean CMBSearchTemplateList

Para servidores que suportem modelos de pesquisa, este bean apresenta uma lista dos modelos de pesquisa disponíveis e permite a selecção de um modelo.

O bean CMBSearchTemplateList tem três estilos. O estilo da imagem, demonstrado na Figura 18, utiliza uma imagem para os segundos planos dos artigos seleccionados e outra para os artigos não seleccionados. A Figura 19 demonstra o estilo da lista de modelos simples. A Figura 20 demonstra o estilo dos modelos da lista pendente.

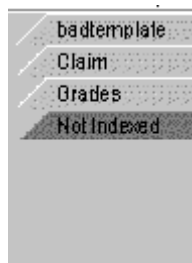


Figura 18. Estilo da lista dos modelos de imagens

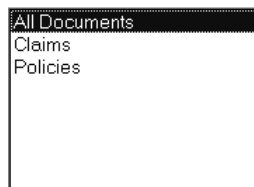


Figura 19. Estilo da lista dos modelos simples

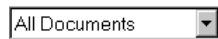


Figura 20. Estilo dos modelos da lista pendente

## Bean CMBSearchTemplateViewer

Para servidores que suportam esses modelos de pesquisa, este bean apresenta um modelo de pesquisa e fornece campos para os utilizadores inserirem os critérios de pesquisa. Executa uma pesquisa baseada nesses critérios.

O bean CMBSearchTemplateViewer (consultar Figura 21 na página 327) apresenta uma janela na qual os utilizadores podem especificar critérios de pesquisa de acordo com o modelo de pesquisa definido pelo administrador do sistema. O bean CMBSearchTemplateViewer inicia uma pesquisa e gera o evento CMBSearchResults para devolver os resultados de pesquisa.

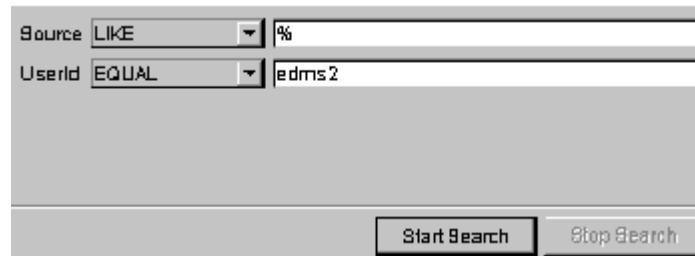


Figura 21. Bean CMBSearchTemplateViewer

O bean CMBSearchTemplateViewer apresenta uma lista de critérios de pesquisa como Source ou UserID. Cada critério de pesquisa tem um identificador, uma caixa pendente do operador e um campo de texto. O ecrã do operador BETWEEN ou NOTBETWEEN apresenta dois campos de texto. Os operadores IN ou NOTIN possuem uma área de texto multilinha. Cada valor deve ser inserido numa linha em separado.

### Áreas de pesquisa de texto

O bean CMBSearchTemplateViewer pode conter áreas que permitam aos utilizadores executar uma pesquisa em atributos totais de texto ou de índice. Uma área de pesquisa de texto total no modelo pode ser tão simples como um campo de texto com uma etiqueta.

Os utilizadores terão de fazer a correspondência entre a sintaxe de consulta e uma pesquisa de texto livre ou booleano ao inserir a cadeia de consulta no campo de texto (consulte a classe DKDatastoreTS). Consulte a Application Programming Reference para obter mais pormenores.

## Validar ou editar campos de CMBSearchTemplateViewer

Pode fornecer lógica de validação ao bean CMBSearchTemplateViewer para modificar critérios de pesquisa inseridos pelo utilizador.

Forneça lógica de validação fornecendo um operador para o CMBTemplateFieldChangedEvent. Os valores actuais dos critérios de pesquisa são armazenados no CMBTemplate devolvido pelo método getTemplate antes da chamada deste evento. O utilizador pode examinar e alterar os critérios. Após a conclusão da manipulação do evento, surgem os novos valores.

## Bean CMBSearchPanel

Para todos os servidores, o painel de pesquisa apresenta uma lista das entidades disponíveis e fornece campos para os utilizadores inserirem os critérios de pesquisa. Executa uma pesquisa baseada nesses critérios. O CMBSearchPanel é útil para executar pesquisas em servidores de conteúdos que não suportem modelos de pesquisa.

O bean CMBSearchPanel apresenta uma janela na qual os utilizadores podem especificar critérios de pesquisa de acordo com as entidades disponíveis no servidor de conteúdos actual. O bean CMBSearchPanel inicia uma pesquisa e gera o CMBSearchResultsEvent para devolver os resultados da pesquisa. O CMBSearchPanel apresenta uma lista de todas as entidades disponíveis na lista pendente na parte superior da janela. Quando é seleccionada uma entidade, o CMBSearchPanel apresenta os atributos da entidade. Cada atributo tem um identificador, uma caixa pendente do operador e um campo de texto. Uma

visualização do operador de intervalo, tal como BETWEEN ou NOTBETWEEN, tem dois campos de texto. Um operador que permita valores múltiplos, tal como o operador IN ou NOTIN, tem uma área de texto de multi-linhas. Cada valor deve ser inserido numa linha em separado na área de texto de multi-linhas.

Figura 22. Bean CMBSearchPanel

## Bean CMBSearchResultsViewer

Este bean apresenta os resultados da pesquisa. Quando o resultado da pesquisa devolve pastas, utilize o bean CMBSearchResultsViewer para entrar na pasta e consultar o respectivo conteúdo. Os artigos nos resultados da pesquisa ou pastas podem ser seleccionados e abertos para visualização numa janela do tipo Windows Explorer.

O bean CMBSearchResultsViewer apresenta resultados de pesquisa numa janela com uma área de janela em árvore e uma área de janela de detalhes. Os utilizadores podem redimensionar a janela fazendo clique em e arrastando a linha que separa as áreas de janela.

A Figura 23 mostra o bean CMBSearchResultsViewer com a pasta **Search Results (Resultados da Pesquisa)** seleccionada.

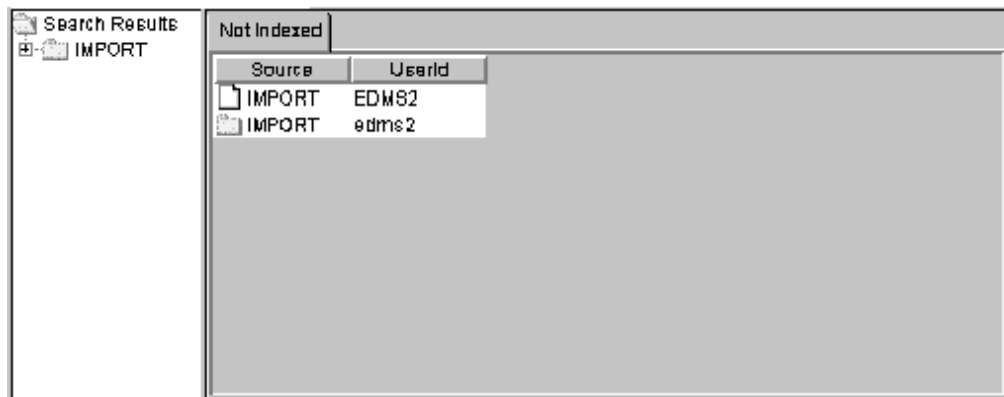


Figura 23. Bean CMBSearchResultsViewer

### Área de janela em árvore de CMBSearchResultsViewer

A área de janela (à esquerda) contém uma pasta principal com a etiqueta **Search Results (Resultados da Pesquisa)**. Por baixo dessa pasta estão todas as pastas localizadas na pesquisa. A área de janela em árvore é opcional. Remova-a definindo a propriedade TreePaneVisible: `setTreePaneVisible(false)`.

### Área de janela Pormenores de CMBSearchResultsViewer

A área de janela de pormenores apresenta o conteúdo de cada pasta seleccionada na área de janela em árvore. Quando os utilizadores seleccionam a pasta **Search Results (Resultados da Pesquisa)**, surge um separador no bloco de notas que contém o nome do modelo da pesquisa. Quando os utilizadores seleccionam uma pasta dentro de **Search Results (Resultados da Pesquisa)**, surge um ou mais separadores: um por cada entidade (tipo de artigo) na pasta. Os nomes dos separadores têm o formato:

classe de índices @ servidor

onde *classe de índice* é a classe de índice ou nome do tipo de artigo e *servidor* é o nome do servidor de conteúdos. As colunas da tabela são alteradas para visualizar os atributos de acordo com a classe de índice ou o tipo de artigo. É suportada selecção múltipla na área de janela de pormenores. Desactive a Selecção múltipla definindo a propriedade `MultiSelectEnabled`: `setMultiSelectEnabled(false)`. Se um tipo de artigo é hierárquico, os valores do atributo dos descendentes são visualizados na tabela com os cabeçalhos de coluna do formulário: o nome do componente descendente/nome do atributo, onde o nome do componente descendente é o nome do componente descendente e o nome do atributo é o nome do atributo do componente descendente. Por exemplo, se um tipo de artigo denominado Journal tiver um componente descendente denominado Author e o componente descendente Author tiver um atributo denominado Last Name, o cabeçalho da coluna será: Author/Last Name.

### Menus emergentes

Surge um menu emergente a disponibilizar opções para Ordenar quando o utilizador faz clique com o botão direito do rato num título da coluna da tabela. Os utilizadores fazem clique em **Sort Ascending (Ordem Ascendente)** para ordenar os artigos na tabela por ordem ascendente. Os utilizadores fazem clique em **Sort Descending (Ordem Descendente)** para ordenar os artigos por ordem descendente. Surge outro menu emergente quando um utilizador faz clique com o botão direito do rato sobre uma pasta que não seja a pasta **Search Results (Resultados da Pesquisa)** na área de janela em árvore, ou faz clique com o botão direito do rato sobre um documento ou pasta na área de janela de pormenores. O menu emergente permite aos utilizadores visualizar detalhes das pastas na área de janela ou editar atributos de pastas.

Opcional: Utilize o `CMBViewFolderEvent` em vez de mostrar os detalhes da pasta no bean `CMBSearchResultsViewer`. Utilize o evento para que o bean `CMBFolderViewer` apresente os conteúdos da pasta seleccionada.

### Acção de duplo clique

Fazer duplo clique em uma pasta numa área de janela em árvore ou num artigo na área de janela de pormenores executa a mesma acção que fazer clique em o artigo do menu emergente **Ver**. Se suprimir o menu emergente do artigo predefinido, irá ocorrer um `CMBItemActionEvent`.

## Substituir menus emergentes

Pode substituir os menus emergentes no `CMBSearchResultsViewer` e no `CMBFolderViewer` por outro menu emergente ou por nenhum menu emergente.

Para desactivar os menus predefinidos, utilize `setDefaultPopupMenu(false)`.



Ao fazer clique com o botão direito do rato numa pasta na área de janela, é gerado um CMBFolderPopupEvent. Ao fazer clique com o botão direito do rato num artigo na área de janela de detalhes, é gerado um CMBItemPopupEvent. Poderá utilizar uma rotina de tratamento para fornecer um menu emergente diferente.

## Bean CMBFolderViewer

Apresenta os conteúdos de uma ou mais pastas numa janela de estilo Windows Explorer.

O bean CMBFolderViewer apresenta uma área de janela em árvore semelhante ao bean CMBSearchResultsViewer. Não existe uma pasta principal de **Search Results (Resultados da Pesquisa)**. A Figura 24 mostra as áreas de janela em árvore e de detalhes do bean CMBFolderViewer.

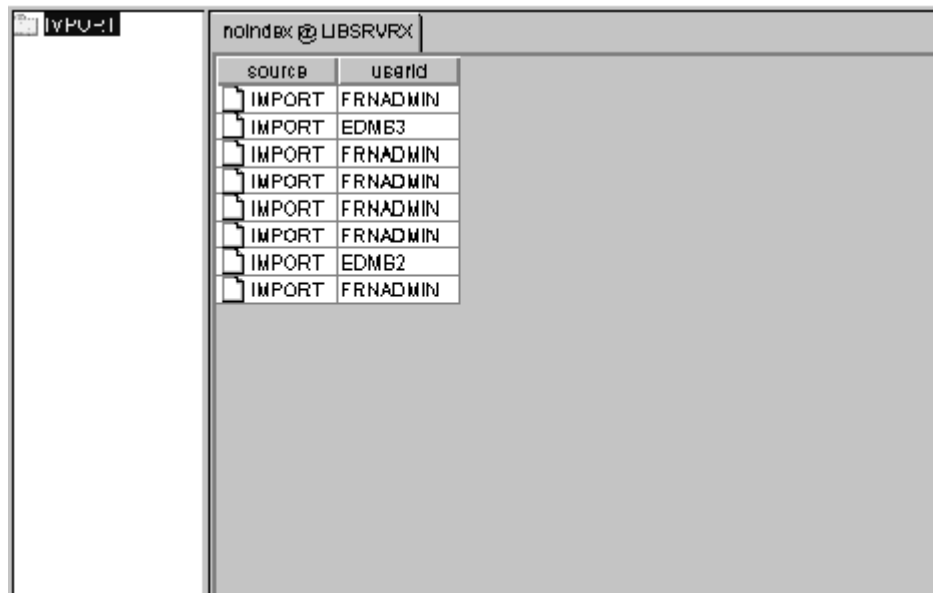


Figura 24. Bean CMBFolderViewer

O bean CMBFolderViewer apresenta uma árvore de pastas na área de janela à esquerda. A área de janela à direita apresenta um bloco de notas de tabelas dos documentos que se encontram na pasta seleccionada na área de janela em árvore. Um repartidor redimensionável separa as áreas de janela em árvore e de bloco de notas.

### Área de janela em árvore de CMBFolderViewer

A área de janela em árvore contém pastas. Surgem pastas imbricadas por baixo de cada pasta.

### Área de janela Pormenores de CMBFolderViewer

A área de janela de pormenores apresenta o conteúdo da pasta que é seleccionada na área de janela em árvore. Os conteúdos são apresentados num bloco de notas com um separador para cada entidade (classe de índice, tipo de artigo ou outro) e servidor sob os quais os artigos da tabela estão indexados. Os nomes dos separadores possuem a seguinte forma: classe de índice @ servidor em que *classe de índice* corresponde ao nome da classe de índice e *servidor* corresponde ao nome do servidor. Em cada página do bloco de notas está uma tabela que apresenta os documentos e pastas que integram a pasta seleccionada. As colunas da tabela mudam de forma a apresentar os atributos segundo a classe de índices.

**Menus emergentes**

O comportamento dos menus emergentes no visualizador de pastas é idêntico ao do visualizador de resultados da pesquisa.

**Ação de duplo clique**

Fazer duplo clique em o visualizador da pasta é idêntico a executar a mesma ação no visualizador de resultados da pesquisa.

## **Bean CMBDocumentViewer**

Apresenta um ou mais documentos com o visualizador de Java ou lançando uma outra aplicação de visualizador.

O bean CMBDocumentViewer fornece capacidades de visualização de documentos através da iniciação ou incorporação de visualizadores de documentos específicos de tipo de conteúdo. Existem dois tipos de visualizadores suportados:

**Visualizadores baseados em Java**

Estes visualizadores têm de expandir a classe CMBJavaDocumentViewer.

**Visualizadores não Java**

Pode ser iniciado um qualquer executável como visualizador para um tipo de conteúdo em particular.

Se a propriedade Visible estiver definida como false, o visualizador é sempre apresentado numa janela separada. Se a propriedade Visible estiver definida como true, o visualizador será apresentado na área de apresentação do bean CMBDocumentViewer, se tal for possível. (Actualmente, esta ação só é possível para visualizadores com base em Java.

CMBJavaDocumentViewer é uma classe abstracta expandida por fornecedores de visualizadores de documentos baseados em Java que se ligam ao bean CMBDocumentViewer. Estes visualizadores podem apresentar os documentos no espaço visível do bean CMBDocumentViewer ou numa janela separada no ecrã.

Uma chamada para CMBDocumentViewer terminate() aguarda até que todos os eventos encerrados do documento sejam processados. Se invocar terminate() a partir do operador de eventos encerrados do documento, poderá ocorrer um bloqueio e o documento irá parar. Para evitar este problema, ao invocar terminate() a partir do operador de eventos onDocumentClosed(CMBDocumentClosedEvent), invoque o método CMBDocumentViewer.terminate() utilizando SwingUtilities.invokeLater(Runnable). Este procedimento adiciona a chamada de terminate() ao final da fila de eventos e prossegue com os outros eventos na fila (tais como o processamento de outros eventos encerrados do documento) antes de invocar o método de terminação.

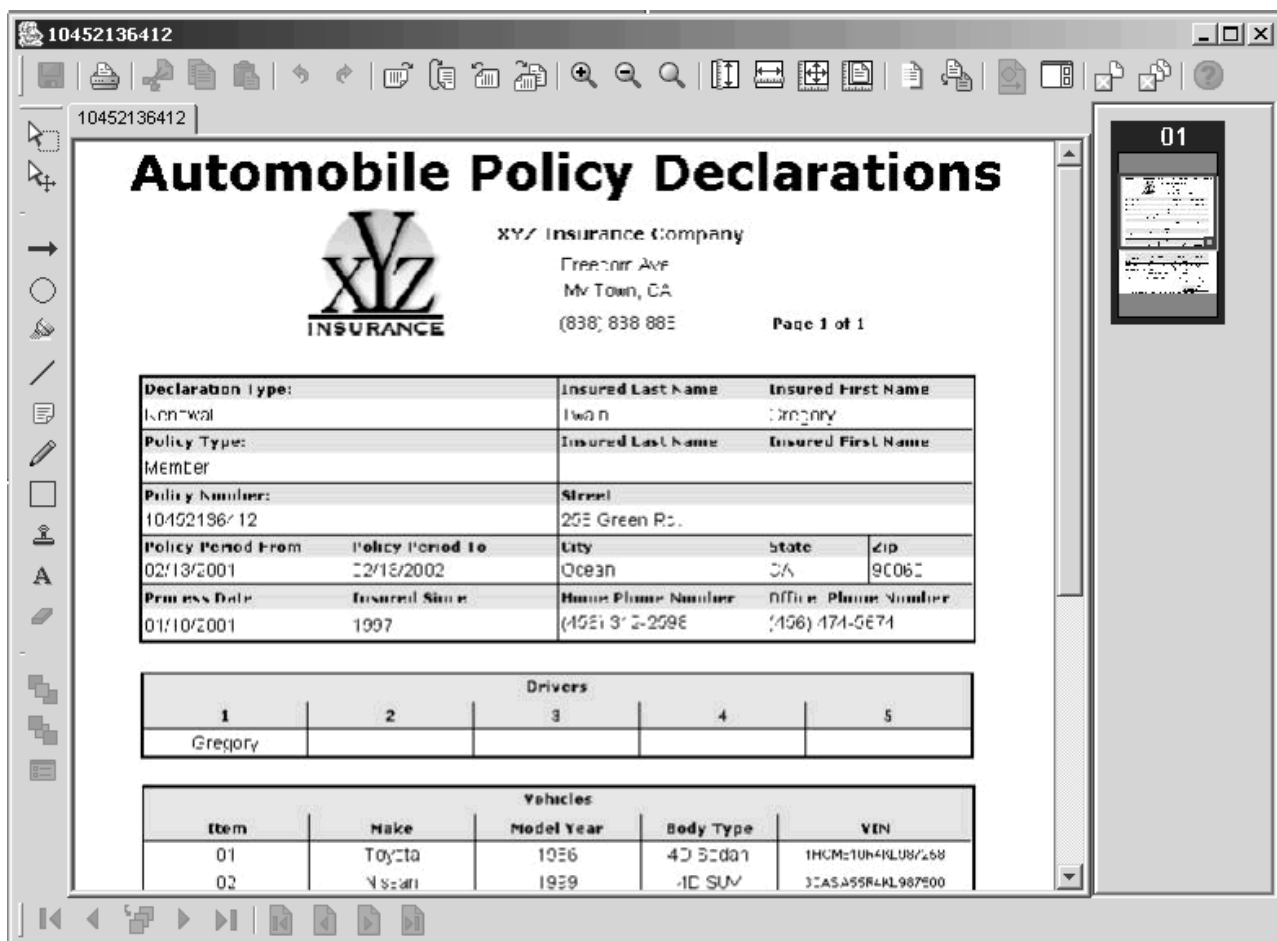


Figura 25. Bean CMBDocumentViewer

## Especificações do visualizador

Os visualizadores terão de ser especificados antes da construção de uma aplicação.

Existem duas formas de especificar visualizadores:

- No cliente de administração do sistema, especifique os visualizadores que utilizam o MIME Type to Application Association Editor (Editor de Associação de Tipos MIME a Aplicações), seleccionando o MIME to Appl. Apl. (Editor MIME para Apl.) no menu **Tools (Ferramentas)**. Para visualizadores com base em Java, o nome da aplicação deverá ser o nome da classe de Java, incluindo o sufixo de classe. Para executáveis, o nome da aplicação deverá ser o nome do executável.
- Utilizando a propriedade Mime2App no CMBDocumentViewer. Esta propriedade pode ser definida como uma instância de um objecto de Propriedades que correlaciona os tipos MIME com nomes de aplicações.

Nos casos em que é especificado um visualizador para um tipo MIME na Administração do DB2 Information Integrator for Content e através da utilização da propriedade Mime2App, terá prioridade a especificação de utilizar Mime2App.

## Visualizadores predefinidos

Se não for especificado nenhum visualizador para um tipo de conteúdo em particular, será iniciado um visualizador predefinido.

Para documentos do OnDemand, é instalado o cliente do OnDemand (em modo de só visualização), se instalado. Os documentos de todos os outros servidores de conteúdos serão visualizados utilizando o CMBGenericDocViewer do toolkit de visualizador de Java. Para editar anotações, seleccione **Edit Document (Editar documento)** no menu **File (Ficheiro)** do visualizador.

## Iniciar visualizadores externos

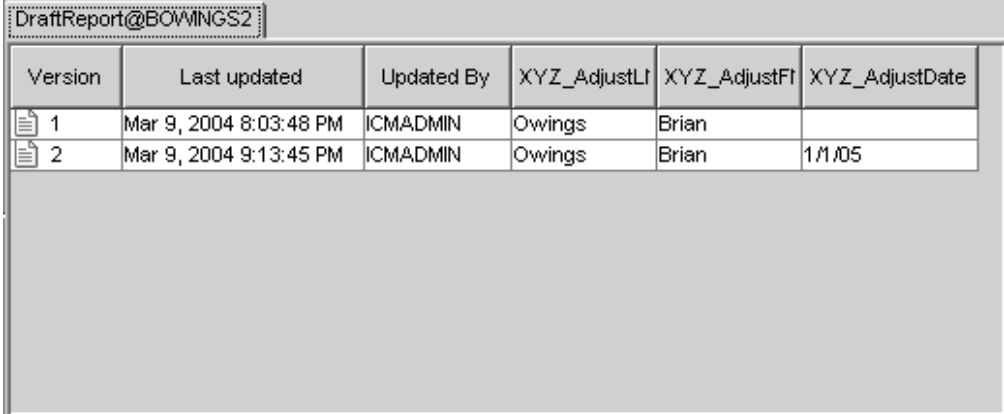
Pode iniciar visualizadores externos utilizando a propriedade Mime2App.

Utilize a propriedade Mime2App do CMBDocumentViewer para especificar quais as aplicações que devem ser iniciadas como visualizadores de documentos de determinados tipos de MIME. Utilize o setMime2App com um objecto de propriedades como argumento que tenha nomes de tipos de MIME que correlacionem valores que são nomes executáveis.

## Bean CMBItemAttributesEditor

Apresenta uma janela na qual os utilizadores podem actualizar o índice de classes e atributos de indexação para um artigo.

O bean CMBItemAttributesEditor (consulte a Figura 26) apresenta uma janela para visualização e modificação da entidade (tipo de artigo) e dos atributos de uma pasta ou documento.



The screenshot shows a window titled "DraftReport@BOWMINGS2". Inside, there is a table with the following data:

| Version | Last updated           | Updated By | XYZ_AdjustL | XYZ_AdjustF | XYZ_AdjustDate |
|---------|------------------------|------------|-------------|-------------|----------------|
| 1       | Mar 9, 2004 8:03:48 PM | ICMADMIN   | Owings      | Brian       |                |
| 2       | Mar 9, 2004 9:13:45 PM | ICMADMIN   | Owings      | Brian       | 1/1/05         |

Figura 26. Bean CMBItemAttributesEditor

No topo da janela surge uma lista que contém todas as entidades disponíveis. A entidade actual é seleccionada por predefinição. Sob a entidade surge uma lista de atributos para essa entidade. Os campos de texto (nome próprio, apelido, e assim por diante) contém inicialmente os valores actuais para o artigo.

Caso os utilizadores seleccionem uma nova entidade, quaisquer atributos com o mesmo nomes que a entidade seleccionada previamente terão os seus valores propagados para os atributos com nomes semelhantes na nova entidade.

Se fizer clique em **Reset (Repor)** irá devolver os valores originais da entidade e dos atributos.

Ao fazer clique em **OK** irá actualizar a entidade e os atributos, activando eventos antes e depois da actualização. Poderá utilizar o evento antes da actualização para validar campos ou completar campos em falta antes da execução da actualização. Este evento pode impedir a actualização especificada.

## Proibir alterações no CMBItemAttributesEditor

Pode fornecer lógica de validação adicional ao CMBItemAttributesEditor, que verifica valores de atributos inseridos pelo utilizador e modifica-os, ou rejeita uma actualização, caso os valores não sejam válidos.

A lógica de validação para proibir alterações no CMBItemAttributesEditor pode ser fornecida pelo operador do CMBEditRequestedEvent.

## Bean CMBVersionsViewer

Apresenta as informações sobre a versão para um documento, se estiver activada a atribuição de versão.

O bean CMBVersionsViewer apresenta uma tabela de atributos de atribuição de versão para um único documento ou artigo. Os atributos de criação de versões apresentados são: o número da versão, o ID de utilizador do criador, a marca de hora de quando a versão foi criada, o ID de utilizador do maior actualizador e a marca de hora da última actualização. A partir do visualizador das versões, o utilizador pode visualizar as diferentes versões de um artigo ou pode actualizar os atributos de um artigo.

| DraftReport@BOWINGS2                                                                  |                        |            |              |              |                |
|---------------------------------------------------------------------------------------|------------------------|------------|--------------|--------------|----------------|
| Version                                                                               | Last updated           | Updated By | XYZ_AdjustLt | XYZ_AdjustFl | XYZ_AdjustDate |
|  1 | Mar 9, 2004 8:03:48 PM | ICMADMIN   | Owings       | Brian        |                |
|  2 | Mar 9, 2004 9:13:45 PM | ICMADMIN   | Owings       | Brian        | 1/1/05         |

Figura 27. Bean Visualizador CMBVersions

## Comportamentos gerais de beans visuais

As secções seguintes descrevem propriedades e comportamentos que são comuns nos beans visuais.

### Propriedades

Esta secção descreve três propriedades partilhadas pelos beans visuais.

Os beans visuais possuem as três seguintes propriedades:

### Ligação

Cada bean tem uma propriedade Connection, que remete para uma instância do bean não visual CMConnection. Terá de definir a propriedade Connection para que o bean visual funcione correctamente.

### CollationStrength

Todos os beans que executam ordenação têm uma propriedade CollationStrength. Os valores definidos para a propriedade CollationStrength são os mesmos valores definidos para a classe java.text.Collator de Java.

### Ocultar/Mostrar botões

Pode ocultar ou apresentar os botões que são apresentados em todos os beans visuais. Utilize a propriedade setNameButtonVisible, em que **Name (Nome)** corresponde ao nome do botão.

### Guardar/restaurar configuração

Para guardar e restaurar campos, utilize os métodos loadConfiguration e saveConfiguration.

O CMBSearchTemplateViewer, o CMBSearchResultsViewer e o CMBFolderViewer têm dois métodos - loadConfiguration e saveConfiguration - que podem ser utilizados para guardar e restaurar valores de campos e tamanhos de colunas entre sessões de aplicação. Um objecto de propriedades é um argumento para todos estes métodos. Poderá utilizar o mesmo objecto de propriedades para os três beans. Os nomes das propriedades guardadas são únicos em todos os beans.

### Suporte para atributos de componente descendente

Os beans visuais suportam um único nível (não imbricado) de componentes descendentes.

Este comportamento é semelhante ao cliente Windows. Para obter informações mais detalhadas, consulte a documentação do cliente Windows.

### Eventos de Ajuda

Cada bean visual gera um evento CMBHelp quando o utilizar solicita ajuda através do botão **Help (Ajuda)** ou da tecla F1.

Alguns beans geram os seguintes eventos relacionados com a ajuda, assim que os utilizadores primem F1 ou Ajuda em janelas secundárias:

#### CMBChangePasswordHelpEvent

Ao fazer clique em **Help (Ajuda)** na janela Change Password (Alterar palavra-passe)

#### CMBUpdateMappingHelpEvent

Ao fazer clique em **Help (Ajuda)** na janela Update Mapping (Actualizar correlação)

#### CMBLoginFailedHelpEvent

Ao fazer clique em **Help (Ajuda)** na janela Server Logon Failed (O Início de Sessão no Servidor Falhou)

#### CMBServerUnavailableHelpEvent

Ao fazer clique em **Help (Ajuda)** na janela Server Unavailable (O Servidor não está disponível)

### Sugestão:

Um método possível para funcionar com a ajuda a partir de todas estas origens é criar uma única classe que implemente os receptores de todos estes eventos.

No método `onHelp`, poderá ser necessária lógica adicional para determinar qual o bean que originou o evento e apresentar o texto de ajuda apropriado para esse bean.

## Substituir um bean visual

É possível substituir um dos beans visuais por outro bean ou por componentes Swing.

Para substituir um bean visual ou um componente Swing, o novo bean deverá implementar os operadores para os eventos do bean visual que está a substituir. Também deverá gerar pelo menos os eventos chave do bean que está a substituir. Os eventos-chave estão descritos na Tabela 42.

*Tabela 42. Beans visuais e eventos chave*

| Bean visual             | Eventos chave                                                      |
|-------------------------|--------------------------------------------------------------------|
| CMBSearchTemplateList   | CMBTemplateSelectedEvent                                           |
| CMBSearchTemplateViewer | CMBSearchStartedEventCMBSearchResultsEvent                         |
| CMBSearchResultsViewer  | CMBViewDocumentEvent CMBViewFolderEvent-CMBEditItemAttributesEvent |
| CMBFolderViewer         | CMBViewDocumentEvent CMBEditItem AttributesEvent                   |
| CMBDocumentViewer       | CMBDocumentOpenedEvent CMBDocument Closed Event                    |
| CMBItemAttributesEditor | nenhum                                                             |

Todos os dados necessários para implementar a função do bean estão disponíveis nos eventos que o bean está a processar ou no bean não visual `CMBConnection`.

## Construir uma aplicação através de beans visuais

As secções seguintes demonstram como os beans visuais se ajustam entre si quando se constrói uma aplicação.

É facultado ao utilizador uma aplicação cliente exemplo que foi escrita utilizando os beans visuais. Os ficheiros origem para o exemplo encontram-se em: `%IBMCMROOT%/samples/java/beans/gui..` Deverá também consultar o ficheiro `readme.html` neste directório para obter detalhes sobre os requisitos de cliente e configuração exemplo.

### Ligar os beans visuais

Esta secção demonstra uma situação exemplificativa para a ligação de beans visuais, de forma a criar uma aplicação simples.

Exceptuando o botão Pesquisar, todos os beans são ligados adicionando o bean destino como receptor do evento indicado do bean origem. Por exemplo, para ligar o `SearchTemplateList` ao `SearchTemplateViewer`, apenas é necessária uma linha de código. Para adicionar um botão para iniciar pesquisas, utilize um **JButton** padrão. Crie uma classe interna para que o evento de acções do botão invoque o método adequado.



Na Figura 28, as linhas que vão de cada um dos beans para o bean de ligação indicam que o bean contém uma referência ao bean de ligação. Esta situação é criada através da definição da propriedade connection de cada bean. Por exemplo, para criar uma referência a partir do bean de painel de início de sessão para o bean de ligação, é necessária uma linha de código.

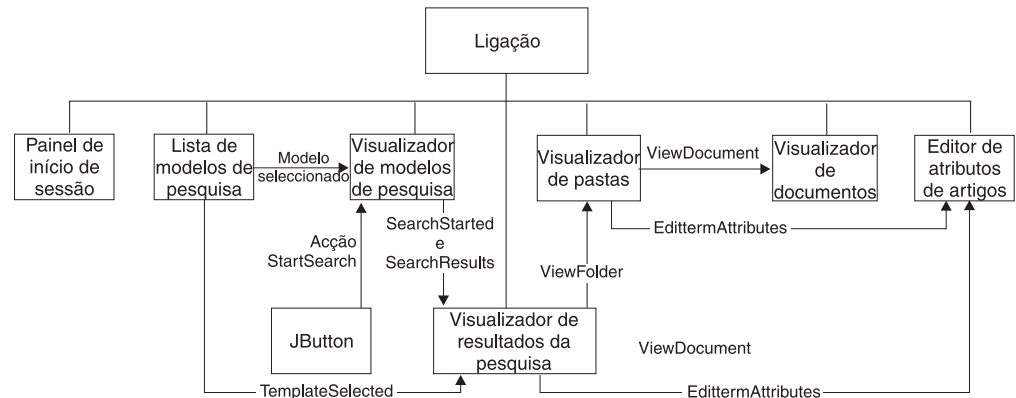


Figura 28. Ligações de bean visual

A Figura 28 mostra nove beans. Um JFrame ou outro bean de conteúdo seria o ascendente de todos estes beans. Uma ordem possível de eventos durante o tempo de execução poderá ser:

1. Introduza um ID de utilizador e palavra-passe na janela de início de sessão e faça clique em **OK**. O bean CMBLogonPanel invoca o método connect do bean CMBCConnection para estabelecer a ligação ao servidor. O bean de ligação estabelece a ligação. O bean CMBSearchTemplateList obtém e apresenta a lista de modelos de pesquisa para esse ID de utilizador. (Não é necessário invocar métodos para originar esta operação. O bean CMBSearchTemplateList está a receber os eventos apropriados do bean CMBCConnection. O CMBSearchTemplateList configura os receptores quando um bean CMBCConnection se associou através do método setConnection.)
2. Selecciona um modelo de pesquisa na lista. O bean CMBSearchTemplateList gera um CMBTemplateSelectedEvent. Quer o CMBSearchTemplateViewer, quer o CMBSearchResultsViewer estão a aguardar o evento. O CMBSearchTemplateViewer apresenta o modelo apropriado. O CMBSearchResultsViewer limpa e apresenta colunas na área de janela de detalhes, conforme definido pelo modelo.
3. Preencha o modelo e premir **Enter** ou fazer clique em **Search (Pesquisa)**. Se o utilizador fizer clique em **Search (Pesquisa)**, o operador do evento da acção invoca o método startSearch. Se o utilizador premir Enter, o método startSearch é implicitamente invocado. O bean CMBSearchTemplateViewer valida os campos do modelo para determinar se é possível iniciar uma pesquisa. Caso se possa iniciar a pesquisa, é gerado um CMBSearchStartedEvent. O CMBSearchResultsViewer aguarda um CMBSearchStartedEvent e limpa os resultados como preparação para novos resultados de pesquisa. À medida que a pesquisa progride, os CMBSearchResultsEvents são gerados para fornecer resultados de pesquisa parciais ao CMBSearchResultsViewer. (Quando a pesquisa é concluída, é gerado o evento CMBSearchCompleted.)
4. Utilize o evento CMBSearchCompleted para activar novamente o botão **Search (Pesquisa)**, caso tenha sido desactivado no início da pesquisa.
5. Expanda pastas na janela Search Results (Resultados da Pesquisa) e, em seguida, seleccionar um documento ou pasta para visualização. É gerado um



CMBViewFolderEvent ou CMBViewDocumentEvent. Os beans CMBFolderViewer e CMBDocumentViewer recebem os respectivos eventos e apresentam a pasta ou documento.

6. Selecciona um documento para visualização no CMBFolderViewer. Ao seleccionar um documento para visualização, irá gerar um CMBViewDocumentEvent. O CMBDocumentViewer recebe este evento e apresenta o documento no visualizador apropriado.
7. Para efectuar a actualização, selecciona os atributos de um documento ou pasta no CMBSearchResultsViewer ou no CMBFolderViewer. A selecção de um documento gera um CMBEditItemAttributesEvent. O bean CMBItemAttributesEditor recebe um CMBEditItemAttributesEvent. Este apresenta a entidade e os atributos para o artigo.
8. Altera a entidade e os atributos e, em seguida, faça clique em **OK** para aplicar as alterações.

### Utilizar beans em mais do que uma janela ou diálogo

O utilizador terá de fornecer códigos adicionais para passar um evento de um bean numa janela para um bean noutra janela.

Normalmente, a constatação de que foi enviado um evento é o motivo para surgir uma janela. A janela EditAttributesDialog contém o ItemAttributesEditor. O SearchFrame cria a janela quando é iniciado um CMBEditItemAttributesEvent:

```
// Invoke a secondary dialog for edit attributes
searchResultsViewer.addEditItemAttributesListener(new
CMBEditItemAttributesListener() {
 public void onEditItemAttributes(CMBEditItemAttributesEvent event) {
 EditAttributesDialog editAttributesDialog = new
 EditAttributesDialog(SearchFrame.this,connection,event.getItem());
 editAttributesDialog.setVisible(true);
 }
});
```

As informações normalmente transferidas para o bean CMBItemAttributesEditor são, em alternativa, transferidas como argumentos para o construtor da janela. No construtor, as informações são transferidas para o bean CMBItemAttributesEditor definindo as seguintes propriedades:

```
itemAttributesEditor.setConnection(connection);
```

```
itemAttributesEditor.setItem(item);
```

---

## Trabalhar com serviços XML (apenas para Java)

O conector do DB2 Content Manager faculta APIs no cmbxmbservice.jar que podem converter três tipos de dados de administração do sistema em XML:

- Objectos de metadados administrativos, que podem incluir utilizadores, grupos de utilizadores, configuração de gestores de recursos, definições de encaminhamento de documentos, privilégios ACL, conjuntos de privilégios, configuração de servidor de bibliotecas e modelos de pesquisa e configuração de servidores do DB2 Information Integrator for Content. Utiliza um esquema de XML denominado cmadmin.xsd (localizado em *IBMCMROOT/config/*) para representar os objectos de administração do DB2 Content Manager Versão 8 num ficheiro XML.
- Objectos de metadados de modelos de dados, que definem a estrutura para tipos de artigos, tipos de componentes e entidades de DB2 Information Integrator for Content. Estes objectos são armazenados como ficheiros XSD (esquemas de armazenamento) e todos eles consultam as definições de elementos e tipos definidos no ficheiro cmdatamodel.xsd (localizado em *IBMCMROOT/config/*).
- Objectos de instâncias de dados, que podem incluir artigos, atributos e artigos de recurso (anexos binários). Estes objectos são armazenados como ficheiros XML que se regem pelos esquemas de armazenamento.

Ao representar os objectos como XML, pode importar, armazenar, actualizar, obter e exportar uma larga variedade de objectos — como documentos ou dados administrativos — entre servidores de DB2 Content Manager sem desenvolver diferentes interfaces para cada sistema.

Para ver um exemplo da funcionalidade de importação e exportação de XML, consulte TXMLExport.java, TXXMLImport.java e TXXMLList.java no directório *IBMCMROOT/samples/java/xml*.

---

## Funcionamento dos serviços XML com outros níveis de programação do DB2 Content Manager

Alguns níveis de programação precisam de objectos XML convertidos para trabalhar com o conector do DB2 Content Manager.

Os níveis incluem:

### Serviços da Web

Interface HTTP do DB2 Content Manager que recebe as mensagens XML (definidas pelo cmbmessages.xsd) num envelope SOAP para efectuar operações de tempo de execução como, por exemplo, importar, exportar, pesquisar, criar, actualizar, obter, eliminar e encaminhar objectos. Os serviços da Web translineiam e extraem automaticamente as mensagens XML na mensagem SOAP e enviam-nas para o JavaBean de envio de mensagens XML.

### JavaBeans (XML)

Classes de Java reutilizáveis que têm por base as APIs de XML do conector do DB2 Content Manager e os JavaBeans do DB2 Information Integrator for Content. Os JavaBeans XML efectuem operações de tempo de execução como, por exemplo, importar, exportar, pesquisar, criar, actualizar, obter,

eliminar e encaminhar objectos. Por exemplo, o bean CMBXMLMessage interpreta todas as mensagens XML baseadas no esquema cmbmessages.xsd.

#### **Utilitário de correlação de esquemas (XML)**

Ferramenta de conversão XML que pode converter um esquema definido pelo utilizador no esquema de armazenamento suportado pelo DB2 Content Manager.

#### **Conector (XML) do DB2 Content Manager**

Interfaces de programação de aplicações XML que podem importar e exportar objectos metadados de modelos de dados, objectos metadados administrativos e objectos de instâncias de dados.

A Figura 29 ilustra como estes níveis de XML se relacionam com o conector do DB2 Content Manager.

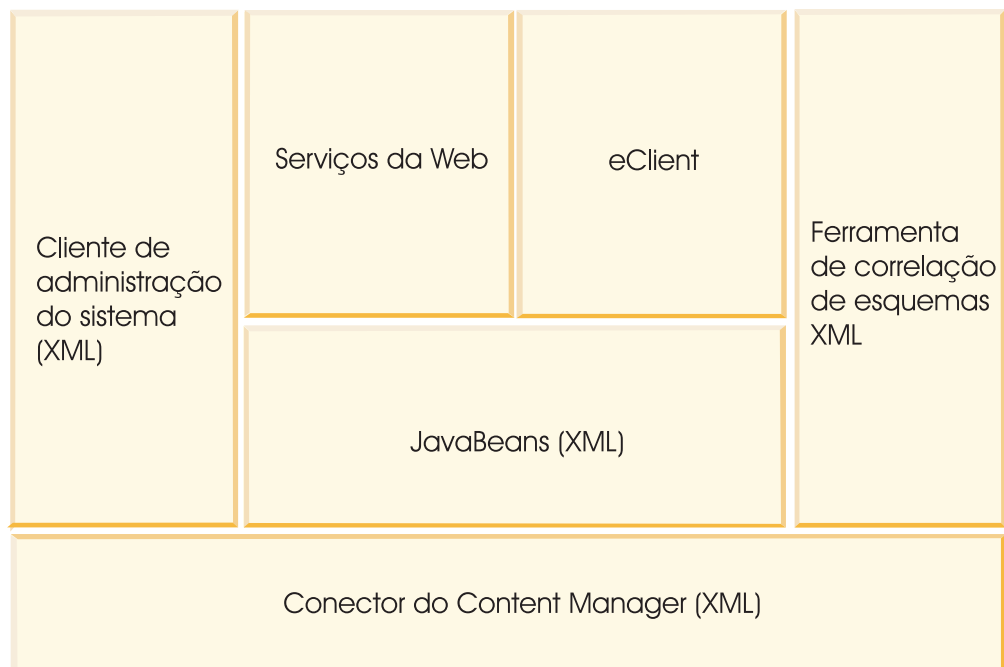


*Figura 29. Níveis de programação de serviços XML*

Figura 30 na página 341 descreve uma situação real para comunicar com o DB2 Content Manager através da sua interface XML:

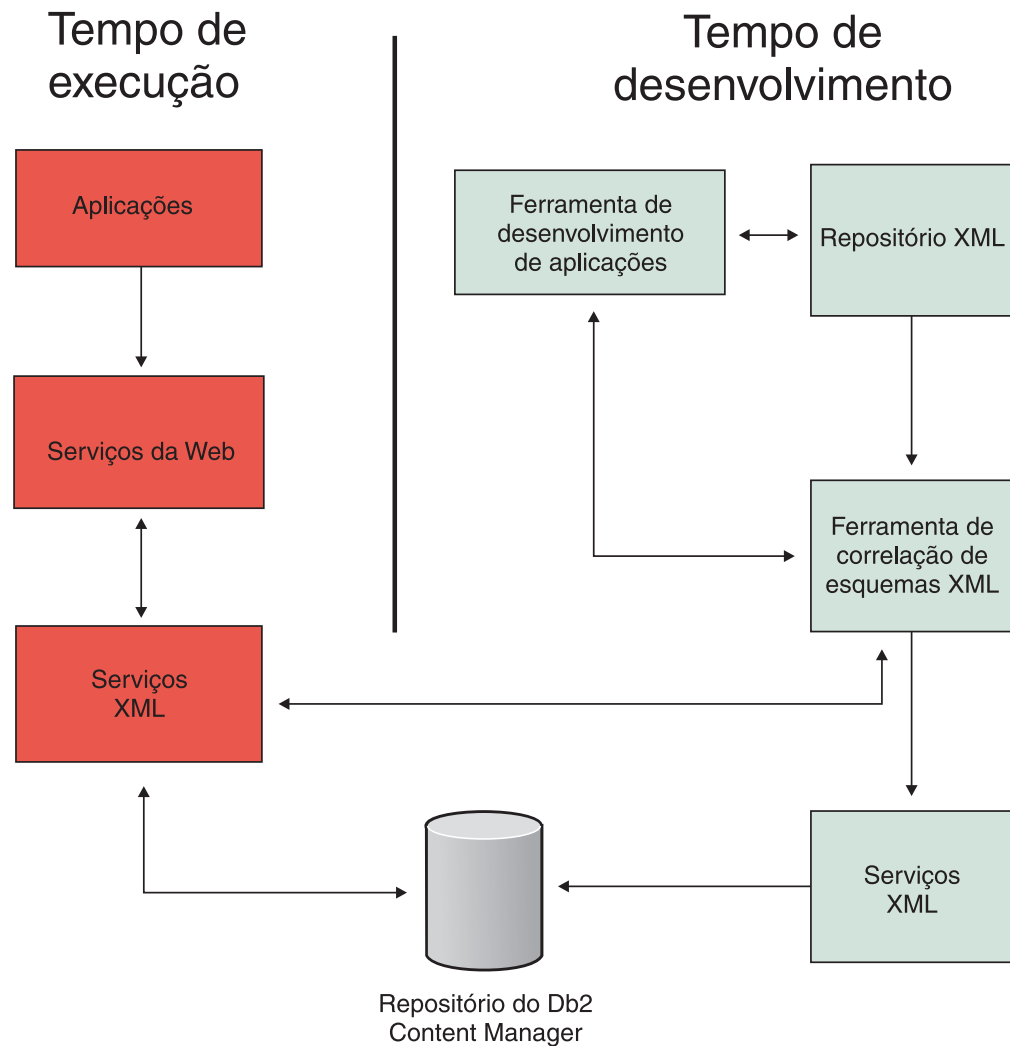


Figura 30. Gráfico de comunicação dos serviços XML

1. A companhia de seguros XYZ define esquemas para manter os seus dados, utilizando ferramentas de programação de aplicações como o WebSphere Application Developer (WSAD) para a sua criação. Esta companhia também opta por armazenar os dados no DB2 Content Manager como artigos.
2. Numa tentativa de migrar todos os dados para o DB2 Content Manager, a companhia de seguros XYZ utiliza a ferramenta de correlação de esquemas XML do DB2 Content Manager para converter o seu esquema em esquemas que o DB2 Content Manager possa interpretar, por exemplo, o esquema de armazenamento.
3. A companhia de seguros XYZ, através da ferramenta de correlação de esquemas XML, cria uma definição de modelo de dados que corresponde ao esquema de armazenamento; de seguida, armazena a correlação de esquemas no DB2 Content Manager. A ferramenta de correlação de esquemas XML invoca internamente a API ingest() dos serviços XML para criar a definição de modelo de dados.
4. A companhia de seguros XYZ escreve uma aplicação personalizada que utiliza transacções de serviços da Web para criar, obter, actualizar, eliminar e encaminhar dados no DB2 Content Manager. O nível de serviços da Web obtém a correlação de esquemas e transforma os dados; de seguida, invoca os beans XML para efectuarem a operação que gere os dados no DB2 Content Manager.

### Informações relacionadas



Programar operações de execução através de XML JavaBeans



Correlacionar um esquema definido pelo utilizador com um esquema de armazenamento através de uma ferramenta de correlação de esquemas XML

---

## Importar e exportar metadados do DB2 Content Manager com serviços XML

Pode importar e exportar dois tipos de metadados utilizando as APIs do DB2 Content Manager.

Os dois tipos de metadados que pode importar e exportar utilizando as APIs do DB2 Content Manager são:

### Objectos administrativos

Utilizadores, grupos de utilizadores, configuração de gestores de recursos, definições de encaminhamento de documentos, privilégios ACL e modelos de pesquisa e configuração de servidores do DB2 Information Integrator for Content. Utilizam um esquema de XML denominado *cmadmin.xsd* (localizado no *IBMCMROOT/config/*) para definir ficheiros XML que contêm objectos de administração do DB2 Content Manager Versão 8.

### Objectos de modelos de dados

Estrutura de tipos de artigos, tipos de componentes e entidades do DB2 Information Integrator for Content. Estes objectos são armazenados como ficheiros XSD, também conhecidos como esquemas de armazenamento. Cada esquema de armazenamento importa um ficheiro comum denominado *cmdatamodel.xsd* (localizado em *IBMCMROOT/config*).

Ao representar metadados do DB2 Content Manager como ficheiros intermédios, pode programar uma série de situações:

- Personalizar uma aplicação para administrar e actualizar dados no DB2 Content Manager através da interface XML.
- Transferir metadados de um sistema do DB2 Content Manager para outro sistema de DB2 Content Manager (tendo em conta quais conflitos de dados).
- Transferir entidades de um sistema do DB2 Information Integrator for Content para outro sistema do DB2 Information Integrator for Content (tendo em conta quais conflitos de dados).

Estas situações ganham importância em situações empresariais normais:

- Durante o desenvolvimento da aplicação de gestão de conteúdos, transferindo metadados de um sistema de teste para um sistema de produção.
- Durante o alargamento de uma aplicação ou adição de um novo sistema de produção do DB2 Content Manager, transferindo objectos específicos entre os sistema de desenvolvimento, teste e produção. Neste caso, os dados existentes no sistema destino são actualizados.
- Durante a detecção e correcção de problemas de um sistema de produção, transferindo objectos específicos de um sistema de produção para um sistema de teste, por forma a diagnosticar o problema.

A classe de serviços de instâncias XML, *DKXMLSysAdminService*, contém três novos métodos Versão 8 Edição 3 para importar e exportar metadados do DB2 Content Manager: *list()*, *ingest()* e *extract()*. Estes dois últimos métodos importam e

exportam esquemas de armazenamento (ficheiros XSD) para objectos de modelos de dados e ficheiros XML para objectos de administração (utilizando o esquema predefinido cmadmin.xsd).

Os métodos ingest() e extract() suportam os seguintes formatos:

#### **Formatos de entradas XML**

**DKXMLStreamObjectDefs**  
sequência de entradas

**DKXMLDOMObjectDefs**  
DOM

#### **Formatos de saída de dados XML**

**DKXMLDOMObjectDefs**  
DOM

DKXMLDOMObjectDefs contém um método para converter o objecto DOM numa sequência de saída de dados.

Adicionalmente, DKXMLSysAdminService suporta as seguintes opções ingest():

#### **DK\_CM\_XML\_IMPORT\_CONTINUE\_ON\_ERROR**

Se definir como OR esta constante, o método ingest() vai importar tantas definições de objectos quanto possível. Se nada especificar nem especificar OR, o processo de importação aborta sob qualquer erro.

#### **DK\_CM\_XML\_IMPORT\_CREATE\_UPDATE**

Se especificar esta constante, o método ingest() substitui objectos que já existem. Se nada especificar nesta constante, a importação falha com um erro para qualquer definição de objecto que já exista.

A classe DKXMLDOMObjectDefs faculta dois métodos, getSysAdminDefs() e getDataModelDefs(), para obter os objectos de modelo de dados exportados (no formato de Esquema XML) e objectos administrativos (no formato XML) em separado. A classe DKXMLExportList pode especificar quais os objectos XML a exportar.

#### **Informações relacionadas**



Importar e exportar dependências de objectos XML

## **Importar e exportar objectos de administração como XML**

A classe de serviços de instâncias XML, DKXMLSysAdminService, contém dois métodos para importar e exportar metadados do DB2 Content Manager: ingest() e extract(). Estes métodos importam e exportam ficheiros XML para objectos de administração que se regem pelo esquema cmadmin.xsd.

A classe DKXMLDOMObjectDefs faculta dois métodos, getSysAdminDefs() e getDataModelDefs(), para obter os objectos de modelo de dados exportados (no formato de Esquema XML) e objectos administrativos (no formato XML) em separado. A classe DKXMLExportList pode especificar quais os objectos XML a exportar.

Os seguintes objectos de administração do sistema do DB2 Content Manager (representados pelas constantes na classe com.ibm.mm.sdk.common.DKConstant) podem ser convertidos de e para XML:

- Domínio administrativo

- Definições de privilégios
- Grupos de privilégios
- Configurações de de privilégios
- Utilizadores
- Grupos de utilizadores
- ACLs
- Configuração de servidor de bibliotecas
- Definição da linguagem do servidor de bibliotecas
- Tipo de ligação
- Tipo MIME
- Tipo semântico
- Classe XDO
- Objectos do gestor de recursos
- Objectos de encaminhamento de documentos

Os seguintes objectos de administração do sistema do DB2 Information Integrator for Content podem ser convertidos de e para XML:

- Definição de servidores
- Entidades federadas
- Modelos de pesquisa

O esquema exportado relativo a um determinado objecto de modelo de dados é semanticamente equivalente ao esquema importado que o cria. Isto quer dizer que, ao exportar e importar um objecto de um sistema para outro sistema, todas as propriedades do objecto devem ser iguais às do objecto original exportado. No entanto, o documento do esquema exportado e o documento do esquema importado podem ser diferentes na sintaxe. Isto devido às muitas formas de representar as mesmas informações num esquema XML.

**Restrição:** Os serviços XML em z/OS não suportam a importação ou exportação de objectos do gestor de recursos.

## Exemplo de XML

O exemplo seguinte cria um novo utilizador (Joshua) e um novo grupo (XMLDev) no servidor do DB2 Content Manager:

```
<?xml version="1.0" encoding="UTF-8"?>
<CMSAdminDefinitions
 xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <user name="JOSHUA" adminDomainName="SuperDomain"
 defaultRM="RMDB"
 defaultSMSColl="CBR.CLLCT001" description="Regular user"
 passwordExpiration="0" userACL="PublicReadACL"
 userGrantPrivilegeSet="ClientUserReadOnly"
 userPrivilegeSet="AllPrivs">
 <userGroup name="XMLDEV"/>
 </user>
 <userGroup adminDomainName="PublicDomain"
 description="XML Development" name="XMLDEV"/>
 <groupData groupName="XMLDEV">
 <user userName="JOSHUA"/>
 </groupData>
</CMSAdminDefinitions>
```

## Importar e exportar objectos de modelos de dados do DB2 Content Manager como ficheiros de esquema XML (XSD)

A classe de serviços de instâncias XML, `DKXMLSysAdminService`, contém dois métodos para importar e exportar metadados do DB2 Content Manager: `ingest()` e `extract()`. Estes métodos importam e exportam esquemas de armazenamento (ficheiros XSD) para objectos de modelos de dados.

A classe `DKXMLDOMObjectDefs` faculta dois métodos, `getSysAdminDefs()` e `getDataModelDefs()`, para obter os objectos de modelo de dados exportados (no formato de Esquema XML) e objectos administrativos (no formato XML) em separado. A classe `DKXMLExportList` pode especificar quais os objectos XML a exportar.

De forma geral, as regras seguintes aplicam-se quando o esquema de armazenamento é importado para o DB2 Content Manager:

- A declaração de elementos raiz (por exemplo, uma apólice de seguros) é correlacionada com um tipo de artigo `XYZ_InsPolicy`.  
`<xsd:element name="XYZ_InsPolicy">`
- A declaração de elementos descendentes (por exemplo, o número de identificação de um veículo) é correlacionada com um tipo de componente `XYZ_VIN` no tipo de componente ascendente correspondente (neste exemplo, o tipo de componente raiz `XYZ_InsPolicy`).  
`<xsd:element maxOccurs="unbounded" minOccurs="0" name="XYZ_VIN">`
- Um atributo dentro de uma declaração de elementos é correlacionado com um atributo no componente correspondente (por exemplo, o atributo do número de ID de uma apólice é correlacionado com o atributo `XYZ_PolicyNum` no tipo de artigo de apólice).  
`<xsd:attribute name="XYZ_PolicyNum">`
- De acordo o padrão SQL/XML para correlacionar identificadores SQL com nomes XML, o conversor de esquemas XML abandona automaticamente caracteres especiais com equivalente Unicode (no formato `_xYYYY`, em que `YYYY` representa a cadeia Unicode). Por exemplo:
  - Os nomes de elementos e de atributos não podem começar pelas letras XML. Por conseguinte, se um tipo de artigo for denominado `XMLDocument`, o seu nome passa a ser: `_x0058_MLDocument`.
  - Os nomes de elementos e de atributos não podem conter espaços. Por conseguinte, se uma entidade federada for denominada `Project Entity`, o seu nome passa a ser: `Project_x0020_Entity`.

Para obter a lista completa de caracteres válidos em XML, consulte o padrão XML em <http://www.w3.org/TR/REC-xml#NT-Name>.

O esquema exportado relativo a um determinado objecto de modelo de dados é semanticamente equivalente ao esquema importado que o cria. Isto quer dizer que, ao exportar e importar um objecto de um sistema para outro sistema, todas as propriedades do objecto importado mantêm-se iguais às do objecto original exportado. No entanto, o documento do esquema exportado e o documento do esquema importado podem ser diferentes na sintaxe, devido às muitas formas de representar as mesmas informações num esquema XML.

O DB2 Content Manager define o esquema de armazenamento efectuando os passos seguintes:



1. O DB2 Content Manager tenta correlacionar qualquer construção ou função no esquema XML com um conceito de modelo de dados do DB2 Content Manager, tal como apresentado em Tabela 43.
2. Se nenhum dos conceitos do DB2 Content Manager corresponder a um esquema XML, o conceito passa a ser representado por um comentário (também conhecido como uma anotação XML), tal como apresentado em Tabela 44 na página 354.
3. As instâncias do DB2 Content Manager (por exemplo, ACLs de nível de artigo e tipos semânticos) são representadas como elementos XML (no espaço de nome do DB2 Content Manager) importados do ficheiro cmdatamodel.xsd. Estas estão apresentadas em Tabela 44 na página 354.

*Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento*

| Objecto                       | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | Comentários                                                          |
|-------------------------------|---------------|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| Atributos (global)            |               |            | Representa um atributo XML num esquema XML. Por exemplo: <code>&lt;attribute name=".." type=".." /&gt;</code> É uma declaração de atributos global. Os atributos locais estão descritos na secção de tipos de componentes.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                                                      |
| -Nome                         | cadeia        | Sim        | Representa um nome de atributo numa declaração de atributos. Por exemplo: <code>&lt;attribute name="XX" type=".." /&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                                      |
| -Tipo (adiante tipo possível) | curto         | Sim        | É correlacionado com tipos primitivos incorporados e derivados, ou com anotação.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | O tipo de atributo é carácter, carácter variável, curto, longo, etc. |
| --Carácter:                   |               |            | Representa a definição do tipo simples de cadeia derivada com as restrições de minLength e maxLength. Por exemplo: <code>&lt;simpleType name="char_100"&gt; &lt;restriction base="string"&gt; &lt;length value="100" /&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code> Tenha em atenção que minLength e maxLength têm de ser iguais. Existe uma definição de tipo simples para cada instância de tipo de atributo com comprimento diferente. O simpleType está ligado à declaração de atributo no ficheiro de exportação. Durante a importação a definição simpleType pode ser declarada globalmente ou dentro da declaração do atributo. A mesma definição pode ser reutilizada. Como o mesmo atributo pode existir em vários artigos ou componentes com propriedades diferentes (como, por exemplo, length, nullable), pode não ser possível referenciar a declaração de atributo global. Como tal, o componente tem a sua própria declaração de atributos local. Durante a importação, a API de serviços XML compara a definição e verifica se há conflitos. | O comprimento pode ser especificado.                                 |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                  | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                            | Comentários                                           |
|--------------------------|---------------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| --alfabético             |               |            | Pode utilizar o elemento padrão na restrição para definir o tipo. Por exemplo: <code>&lt;simpleType name="alpha_char_100"&gt; &lt;restriction base='string'&gt; &lt;pattern value="[a-zA-Z]*" /&gt; &lt;length value="100" /&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code>                                 |                                                       |
| --numérico               |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --alfanumérico           |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --Alfanumérico expandido |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --outros                 |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| -Carácter variável       |               |            | Representa a definição do tipo simples de cadeia derivada com as restrições de minLength e maxLength. Por exemplo, <code>&lt;simpleType name="varchar_0,256"&gt; &lt;restriction base='string'&gt; &lt;minLength value="0" /&gt; &lt;maxLength value="256" /&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code> | O comprimento máximo e mínimo pode ser especificado.  |
| --alfabético             |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --numérico               |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --alfanumérico           |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --Alfanumérico expandido |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| --outros                 |               |            |                                                                                                                                                                                                                                                                                                                  |                                                       |
| -Inteiro curto           |               |            | Representa o tipo derivado incorporado curto.                                                                                                                                                                                                                                                                    | Este valor máximo e mínimo é específico.              |
| -Inteiro longo           |               |            | Representa o tipo derivado incorporado do número inteiro.                                                                                                                                                                                                                                                        | Este valor máximo e mínimo é específico.              |
| -Duplo                   |               |            | Representa o tipo primitivo incorporado duplo.                                                                                                                                                                                                                                                                   |                                                       |
| -Decimal                 |               |            | Representa o tipo primitivo incorporado decimal.                                                                                                                                                                                                                                                                 | Este comprimento é específico e os lugares são fixos. |
| -Data                    |               |            | Representa o tipo de data primitivo incorporado com conversão de valor facial (conversão para o formato facial que é compreendido pelo Content Manager e pelo DB2).                                                                                                                                              |                                                       |
| -Hora                    |               |            | Representa o tipo de hora primitivo incorporado                                                                                                                                                                                                                                                                  |                                                       |
| -Marca de hora           |               |            | Representa o tipo dateTime primitivo incorporado com conversão de valor facial.                                                                                                                                                                                                                                  |                                                       |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                      | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                         | Comentários                          |
|------------------------------|---------------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------|
| -BLOB                        |               |            | Tipo simples de cadeia regular com um espaço de nome de meta-atributos de CM. Por exemplo: <code>&lt;attribute name=".." type="base64Binary"&gt; &lt;annotation&gt;&lt;appinfo&gt; &lt;CM:dataType value="BLOB"/&gt; &lt;/appinfo&gt; &lt;/annotation&gt; &lt;/attribute&gt;</code>                                           | O comprimento pode ser especificado. |
| -CLOB                        |               |            | É correlacionado com um tipo simples de cadeia regular com um espaço de nome de meta-atributos CM. Por exemplo: <code>&lt;attribute name=".." type="string"&gt; &lt;annotation&gt;&lt;appinfo&gt; &lt;CM:datatype value="CLOB"/&gt; &lt;/appinfo&gt;&lt;/annotation&gt; &lt;/attribute&gt;</code>                             | O comprimento pode ser especificado. |
| -máx                         | inteiro       |            | Elemento maxInclusive na definição de tipo simples derivado. Por exemplo: <code>&lt;simpleType name='integer_max100'&gt; &lt;restriction base='integer'&gt; &lt;maxInclusive value="100"/&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code> Criará um novo tipo. O DB2 Content Manager associa uma propriedade ao atributo. | Apenas para curto e número inteiro.  |
| -mín                         | inteiro       |            | Elemento minInclusive na definição de tipo simples derivado. Por exemplo: <code>&lt;simpleType name='integer_min10'&gt; &lt;restriction base='integer'&gt; &lt;minInclusive value="10"/&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code>                                                                                   | Apenas para curto e número inteiro.  |
| -comprimento                 | inteiro       |            | Elemento comprimento na definição de tipo simples derivado. Por exemplo: <code>&lt;simpleType name='char_256'&gt; &lt;restriction base="string"&gt; &lt;length value="256"/&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code>                                                                                               |                                      |
| -escala/precisão             | inteiro       |            | Elementos fractionDigits e totalDigits na definição de tipo simples derivado. Por exemplo: <code>&lt;simpleType name='decimal_8,3'&gt; &lt;restriction base='decimal'&gt; &lt;fractionDigits value="3"/&gt; &lt;totalDigits value="8"/&gt; &lt;/restriction&gt; &lt;/simpleType&gt;</code>                                    | Apenas para decimais.                |
| Grupos de atributos (global) |               |            | Elemento attributeGroup no XSD. Por exemplo: <code>&lt;xs:attributeGroup name="myAttrGrp"&gt; &lt;xs:attribute .../&gt; ... &lt;/xs:attributeGroup&gt;</code>                                                                                                                                                                 |                                      |
| -Nome                        | cadeia        | Sim        | Atributo de nome no elemento attributeGroup. Por exemplo: <code>&lt;xs:attributeGroup name="myAttrGrp"&gt; &lt;xs:attribute .../&gt; ... &lt;/xs:attributeGroup&gt;</code>                                                                                                                                                    |                                      |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                   | Tipo de dados     | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                                                                                                                                                                               | Comentários                                                                                                                                                                     |
|---------------------------|-------------------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -Atributos                | matriz de cadeias | sim        | Elementos descendentes no elemento attributeGroup. Por exemplo, <code>&lt;xs:attributeGroup name="myAttrGrp"&gt;<br/>&lt;xs:attribute .../&gt; ... &lt;/xs:attributeGroup&gt;</code> O nome do atributo terá o nome do grupo de atributos como prefixo. É semelhante à forma como é efectuado no CM. Para a instância de dados, o nome de atributo corresponde à combinação do nome do grupo de atributos e do nome de atributo. Por exemplo, <code>myAttrGrp.myStringAttr</code> . | Esta é a matriz de atributos.                                                                                                                                                   |
| Atributo de referência    |                   |            | É correlacionado com o atributo de cadeia com uma anotação de esquema XML específica. No modelo de dados do CM, o atributo de referência é correlacionado com uma definição de grupo de atributos, uma vez que se apresenta com alguns atributos internos; no entanto, não corresponde ao modelo de dados. Por isso, em vez de modelá-lo com um attributeGroup no XSD, é correlacionado com um atributo regular.                                                                    |                                                                                                                                                                                 |
| -Nome                     | cadeia            | sim        | É correlacionado com um nome de atributo.                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                 |
| Tipo de artigo: definição |                   |            | Definição global de elemento, com elemento <code>xs:schema</code> como ascendente. Por exemplo: <code>&lt;xs:schema ....&gt; &lt;xs:element name="elem1"&gt;<br/>.... &lt;/xs:element&gt; &lt;/xs:schema&gt;</code> Também tem uma estrutura de Esquema XML semelhante à da "Component Type Definition" ("Definição de Tipo de Componente") dado que o tipo de artigo é baseado no modelo de dados do DB2 Content Manager.                                                          |                                                                                                                                                                                 |
| -Nome                     | cadeia            | sim        | Atributo de nome na declaração <code>xs:element</code> . Por exemplo: <code>&lt;xs:schema ....&gt; &lt;xs:element name="elem1"&gt; .... &lt;/xs:element&gt;<br/>&lt;/xs:schema&gt;</code>                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                 |
| -Propriedades             |                   |            | Adiciona a definição de um elemento opcional imediatamente por baixo do elemento raiz para descrever qualquer propriedade de nível de instância. Irá remeter para o elemento definido no ficheiro de esquemas <code>cmdatamodel.xsd</code> . Por exemplo: <code>&lt;element ref="cm:properties" minOccurs="0" maxOccurs="unbounded"&gt;</code>                                                                                                                                      | Utilizado na instância do artigo para descrever quaisquer propriedades de nível de instância associadas ao artigo como, por exemplo, tipo semântico, ACL, hora de criação, etc. |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                                                                                                             | Tipo de dados  | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                                                                                                                                                                          | Comentários                                                                                                                                          |
|---------------------------------------------------------------------------------------------------------------------|----------------|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| -Objecto de Recurso                                                                                                 |                |            | Adiciona a definição de um elemento opcional imediatamente por baixo do elemento raiz para descrever qualquer conteúdo de recurso associado ao artigo. Irá remeter para o elemento definido no ficheiro de esquemas cmdatamodel.xsd. Esta referência de elemento só existe se a definição do elemento descrever um tipo de artigo de recurso ou tipo de artigo parte. <element ref="cm:resourceObj" minOccurs="0" maxOccurs="unbounded">                                       | Utilizado na instância de artigo para descrever quaisquer conteúdos de recurso associados ao artigo.                                                 |
| -Objecto de ligação Incorporada                                                                                     |                |            | Adiciona a definição de um elemento opcional imediatamente por baixo do elemento raiz para descrever qualquer ligação de chegada ou de partida no nível da instância. Irá remeter para o elemento definido no ficheiro de esquemas cmdatamodel.xsd. Esta referência do elemento só existe se a definição do elemento descrever um tipo de artigo, um tipo de artigo de recurso ou um tipo de artigo de documento. <element ref="cm:links" minOccurs="0" maxOccurs="unbounded"> | Utilizado na instância de artigo para descrever todas as ligações de entrada e de saída.                                                             |
| Classificação do tipo de artigo<br>• Artigo que não de recurso<br>• Artigo de recurso<br>• Tipo de artigo documento | short          |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Representa artigos, artigos de recurso, documentos ou partes de documentos. Utiliza o ItemTypeRelation DefICM para representar relações com a parte. |
| -Nome de tipo parte                                                                                                 | cadeia,inteiro |            | Adiciona um elemento opcional por baixo do elemento de raiz para referir a declaração do elemento PART definido. Por exemplo: <element ref="cm:ICMBASE" minOccurs="0" maxOccurs="unbounded"> A definição do elemento de parte terá o elemento de anotação <cm:referencedOnly> se o tipo de artigo de parte não fizer parte da lista de exportação.                                                                                                                             |                                                                                                                                                      |
| -Pesquisável por texto/opção                                                                                        |                |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                      |
| -Nome da ACL                                                                                                        | cadeia,inteiro |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                      |
| -Nome do GR                                                                                                         | cadeia, curto  |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                      |
| -Nome da recolha SMS                                                                                                | cadeia, curto  |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                      |
| -Política de novas versões                                                                                          | curto          |            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Nunca criar, criar sempre ou o utilizador escolhe                                                                                                    |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                    | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | Comentários |
|----------------------------|---------------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 4. Tipo de parte documento |               |            | Representa uma declaração do elemento de raiz (um tipo de artigo). Esse elemento de raiz só pode ser consultado por um tipo de documento. Nenhuma instância de XML pode basear-se neste tipo de documento.                                                                                                                                                                                                                                                                                                                                                                                                                   |             |
| Tipo de componente:        |               |            | Correlaciona-se com uma declaração de elemento no esquema XML. Se a definição existir de forma autónoma (ou seja, se não estiver associada a nenhum itemType), correlaciona-se com uma declaração de elemento de raiz num esquema de XML com um elemento cm:entityType com o valor value="component". Normalmente, um elemento que não de raiz - descendentes de outro elemento. Será elemento de raiz se não houver definição de tipo de artigo correspondente. Apenas aceita "sequence", e não aceita "all", "choice" nem partícula imbricada. O modelo de dados do DB2 Content Manager não detecta a ordem de instâncias. |             |
| -Grupos de atributos       |               |            | É correlacionado com a definição de grupo de atributos XSD com "ref=" para a definição de grupos de atributos global. A definição de grupo de atributos global toma o nome: <component type name>.<attribute group name>. A definição de grupo de atributos global tem o elemento de anotação <cm:referencedOnly> para o diferenciar do grupo de atributos global exportados.                                                                                                                                                                                                                                                |             |
| -Atributos de referência   |               |            | É correlacionado com a utilização de atributos XSD. A definição local é a mesma da definição de atributo de referência global.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |             |
| -Atributos                 |               |            | É correlacionado com a utilização de atributos XSD. O atributo local tem a sua própria definição e anotação. Utiliza todas as definições e anotações de tipo descritas no atributo global que se aplicam ao local.                                                                                                                                                                                                                                                                                                                                                                                                           |             |
| -É necessário              | booleano      |            | Representa o nome de atributo utilizado na declaração de atributos. Por exemplo: <xsd:attribute name=".." use="required" .../> É assim que o valor de atributo anulável é correlacionado com os atributos "use" e "default". "nullable --> use="optional" "non-nullable w/o default --> use="required" "non-nullable w/ default --> use="optional" default="X"                                                                                                                                                                                                                                                               |             |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                                   | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                                                                                                 | Comentários                           |
|-------------------------------------------|---------------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------|
| -Valor Máx                                | inteiro       |            | Define um novo simpleType com um novo valor máximo, como o elemento maxInclusive na definição de tipo simples derivado. Por exemplo: <simpleType name='integer_max100'> <restriction base='integer'> <maxInclusive value="100" /> </restriction> </simpleType>                                                                                        |                                       |
| -Valor Mín                                | inteiro       |            | Representa o elemento minInclusive na definição de tipo simples derivado. Por exemplo: <simpleType name='integer_min10'> <restriction base='integer'> <minInclusive value="10" /> </restriction> </simpleType>                                                                                                                                        |                                       |
| -Valor predefinido                        | cadeia        |            | Representa o atributo predefinido na declaração de atributos. Por exemplo: <xsd:attribute name=".." default="10" ... />                                                                                                                                                                                                                               |                                       |
| Subcomponentes / componentes descendentes |               |            | É correlacionado com uma declaração de elemento descendente (utilizando sequências), sendo a ocorrência mínima e máxima do componente descendente correlacionado com o minOccurs e maxOccurs da declaração de elemento.                                                                                                                               |                                       |
| Vista do tipo de artigo:                  |               |            | É correlacionado com uma definição de elemento de raiz. A definição de vista de tipo de artigo partilha a mesma definição que a definição de tipo de artigo, apenas com as anotações definidas com o âmbito = "VIEW". Também recebe a definição de vista de tipo de componente, dado que a vista de tipo de artigo é uma vista de tipo de componente. | Efectua a projecção e não a selecção. |
| Vista do tipo de componente:              |               |            | É correlacionado com uma definição de elemento descendente de uma definição de vista de tipo de artigo. Terá a mesma estrutura de hierarquia que a definição de tipo de componente. Por exemplo, contém atributos, grupos de atributos e subcomponentes.                                                                                              | Efectua a projecção e não a selecção. |
| -Atributos                                |               |            | Utiliza a mesma representação que a definição do atributo regular no tipo de componente.                                                                                                                                                                                                                                                              |                                       |
| Entidade federada                         |               |            | É correlacionado com uma definição de elemento de raiz. Como é uma entidade federada, não pode conter qualquer subelemento. Estas definições de entidade só podem ser criadas com o conector Federado.                                                                                                                                                |                                       |
| -nome                                     | cadeia        | sim        | Atributo de nome na declaração xs:element. Por exemplo: <xs:schema ....> <xs:element name="elem1"> <xs:annotation><xs:appinfo> <xs:entityType value="federated" /> </xs:appinfo></xs:annotation> .... </xs:element> </xs:schema>                                                                                                                      |                                       |
| -Pesquisável por texto                    | booleano      |            |                                                                                                                                                                                                                                                                                                                                                       |                                       |

Tabela 43. Como se correlacionam os objectos de modelos de dados DB2 Content Manager e as construções de esquemas de armazenamento (continuação)

| Objecto                                                                                                                                          | Tipo de dados | Necessário | Construção de esquema de armazenamento correspondente                                                                                                                                                                                                                              | Comentários |
|--------------------------------------------------------------------------------------------------------------------------------------------------|---------------|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| -Activado com a criação da pasta federada nativa.                                                                                                | booleano      |            |                                                                                                                                                                                                                                                                                    |             |
| -Atributo:                                                                                                                                       |               |            | É correlacionado com a declaração de atributos XSD dentro do elemento.                                                                                                                                                                                                             |             |
| --nome                                                                                                                                           | cadeia        |            | Atributo de nome na declaração xs:attribute.<br>Por exemplo: <xs:schema ....> <xs:element name="elem1"> <xs:annotation><xs:appinfo><xs:entityType value="federated" /></xs:appinfo></xs:annotation><xs:complexType> <xs:attribute name="attr1"....> .... </xs:element></xs:schema> |             |
| --tipo de dados e outras informações de atributos como, por exemplo, comprimento, precisão, escala, max, min, anulável, consultável e de escrita |               |            | Utiliza a mesma definição que a categoria de atributos, mas algumas representação não constam (como as informações pesquisáveis por texto).                                                                                                                                        |             |
| -Nome da entidade federada                                                                                                                       | cadeia        |            |                                                                                                                                                                                                                                                                                    |             |
| -Nome do atributo federado                                                                                                                       | cadeia        |            |                                                                                                                                                                                                                                                                                    |             |
| -Nome do servidor nativo                                                                                                                         | cadeia        |            |                                                                                                                                                                                                                                                                                    |             |
| -Tipo de servidor nativo                                                                                                                         | cadeia        |            |                                                                                                                                                                                                                                                                                    |             |
| -Nome da entidade nativa                                                                                                                         | cadeia        |            |                                                                                                                                                                                                                                                                                    |             |

Para a Tabela 44 na página 354, o Âmbito da anotação pode ter as seguintes definições:

#### GLOBAL

A anotação aplica-se se a propriedade pertencer a uma definição global ou a uma declaração global.

#### LOCAL

A anotação aplica-se se a propriedade pertencer a uma declaração local.

#### VIEW

A anotação aplica-se se a propriedade pertencer a uma declaração que faça parte da ITEM TYPE VIEW.



Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento

| Objecto                       | Informações especificadas em anotações                                                                                                                                                                                                         | Âmbito da anotação        | Informação de nível de instância |
|-------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|----------------------------------|
| Atributos (global)            |                                                                                                                                                                                                                                                |                           |                                  |
| -Tipo (adiante tipo possível) |                                                                                                                                                                                                                                                |                           |                                  |
| --Carácter:                   |                                                                                                                                                                                                                                                |                           |                                  |
| --alfabético                  | Utilize o elemento cm:stringType com um atributo de valor no tipo de enumeração {ALPHA, NUMERIC, ALPHANUMERIC, ALPHANUMERIC_EXT, OTHER}, para representar o valor. O elemento de padrão é opcional. Neste caso, <cm:stringType value="ALPHA"/> | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --numérico                    | <cm:stringType value="NUMERIC"/>                                                                                                                                                                                                               | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --alfanumérico                | <cm:stringType value="ALPHANUMERIC"/>                                                                                                                                                                                                          | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --Alfanumérico expandido      | <cm:stringType value="ALPHANUMERIC_EXT"/>                                                                                                                                                                                                      | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --outros                      | <cm:stringType value="OTHER"/>                                                                                                                                                                                                                 | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Carácter variável            |                                                                                                                                                                                                                                                |                           |                                  |
| --alfabético                  | Tal como no caso Carácter <cm:stringType value="ALPHA"/>                                                                                                                                                                                       | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --numérico                    | <cm:stringType value="NUMERIC"/>                                                                                                                                                                                                               | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --alfanumérico                | <cm:stringType value="ALPHANUMERIC"/>                                                                                                                                                                                                          | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --Alfanumérico expandido      | <cm:stringType value="ALPHANUMERIC_EXT"/>                                                                                                                                                                                                      | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| --outros                      | <cm:stringType value="OTHER"/>                                                                                                                                                                                                                 | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Inteiro curto                |                                                                                                                                                                                                                                                |                           |                                  |
| -Inteiro longo                |                                                                                                                                                                                                                                                |                           |                                  |
| -Duplo                        |                                                                                                                                                                                                                                                |                           |                                  |
| -Decimal                      |                                                                                                                                                                                                                                                |                           |                                  |
| -Data                         |                                                                                                                                                                                                                                                |                           |                                  |
| -Hora                         |                                                                                                                                                                                                                                                |                           |                                  |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                             | Informações especificadas em anotações                                                                         | Âmbito da anotação        | Informação de nível de instância |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------|---------------------------|----------------------------------|
| -Marca de hora                      |                                                                                                                |                           |                                  |
| -BLOB                               | <CM:dataType value="BLOB" />                                                                                   | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -CLOB                               | <CM:dataType value="CLOB" />                                                                                   | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -máx                                |                                                                                                                |                           |                                  |
| -mín                                |                                                                                                                |                           |                                  |
| -comprimento                        |                                                                                                                |                           |                                  |
| -escala/precisão                    |                                                                                                                |                           |                                  |
| -Descrição com códigos de idioma    | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence).  | GLOBAL                    |                                  |
| Grupos de atributos (global)        |                                                                                                                |                           |                                  |
| -Nome                               |                                                                                                                |                           |                                  |
| -Atributos                          |                                                                                                                |                           |                                  |
| -Descrição com códigos de idioma    | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence).  | GLOBAL                    |                                  |
| Atributo de referência              | Indicado por um elemento de anotação, cm:referenceAttribute com os seguintes atributos:                        | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Nome                               |                                                                                                                |                           |                                  |
| -Regra de eliminação de referências | {cm:referenceAttribute}<br>cm:deleteRule=<string> in enumeration type {NO_ACTION, SET_NULL, CASCADE, RESTRICT. | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Número de sequência de referências | {cm:referenceAttribute}<br>cm:sequenceNumber=<short>                                                           | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Descrição com códigos de idioma    | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence).  | GLOBAL                    |                                  |
| Tipo de artigo: definição           |                                                                                                                |                           |                                  |
| -Nome                               |                                                                                                                |                           |                                  |
| -Descrição com códigos de idioma    | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence).  | GLOBAL                    |                                  |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                                | Informações especificadas em anotações                                                                                                                                                              | Âmbito da anotação | Informação de nível de instância                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -Política de novas versões                             | elemento cm:versionPolicy com um atributo de valor no tipo de enumeração {NEVER, ALWAYS, BY_APPLICATION}.                                                                                           | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| -Número máximo de versões                              | elemento cm:maximumVersions com um atributo de valor.                                                                                                                                               | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| -Propriedades                                          |                                                                                                                                                                                                     |                    | O ficheiro cmdatamodel.xsd contém a definição de propriedades.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| -Objecto de Recurso                                    |                                                                                                                                                                                                     |                    | O ficheiro cmdatamodel.xsd3 contém a definição de resourceObj.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| -Objecto de ligação Incorporada                        |                                                                                                                                                                                                     |                    | O ficheiro cmdatamodel.xsd contém a definição das ligações, que é <element name="links"> <complexType> <sequence> <element name="outbound" minOccurs="0" maxOccurs="unbounded"> <complexType> <attribute name="toitem" type="string"/> <attribute name="linktype" type="string"/> <attribute name="linkitem" type="string"/> </complexType> </element> <element name="inbound" minOccurs="0" maxOccurs="unbounded"> <complexType> <attribute name="fromitem" type="string"/> <attribute name="linktype" type="string"/> <attribute name="linkinfoitem" type="string"/> </complexType> </element> </complexType> </element> Tenha em atenção que fará parte do ficheiro de esquemas predefinido de DB2 Content Manager, cmdatamodel.xsd, a importar. |
| Classificação do tipo de artigo                        | Implícita pelos elementos seguintes:                                                                                                                                                                |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 1. Artigo que não de recurso                           | Se o cm:entityType não existir ou cm:entityType="ITEM".                                                                                                                                             | GLOBAL, VIEW       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 2. Artigo de recurso (inclui as seguintes informações) | Elemento cm:entityType com um value="RESOURCEITEM" para indicar que este é um tipo de artigo de recurso. Também deve existir um elementos cm:resourceItemInfo com os seguintes atributos/elementos. | GLOBAL, VIEW       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| -Nome da classe XDO                                    | {cm:resourceItemInfo} atributo cm:name no elemento descendente cm:XDOClass do elemento cm:resourceItemInfo element.                                                                                 | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| -Pesquisável por texto                                 | {cm:resourceItemInfo} elemento cm:textSearchable, com um atributo de valor no tipo booleano.                                                                                                        | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                                          | Informações especificadas em anotações                                                                                                                                                                                                               | Âmbito da anotação | Informação de nível de instância                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -Informações de Índice de Texto                                  | {cm:resourceItemInfo} conjunto de atributos no elemento cm:textIndexInfo, igual às informações de texto na declaração do atributo. Mas, neste caso, será ao nível do tipo de artigo e o elemento para a elemento descendente do cm:resourceItemInfo. | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| -Nome do GR                                                      | {cm:resourceItemInfo} elemento descendente cm:RM do elemento cm:resourceItemInfo com nome e atributos.                                                                                                                                               | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| -Nome da recolha SMS                                             | {cm:resourceItemInfo} elemento descendente cm:SMSCollection do elemento cm:resourceItemInfo com atributos de nome.                                                                                                                                   | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| -Nome da coluna de Busca prévia                                  | {cm:resourceItemInfo} elemento cm:prefetchCollection com nome e atributos.                                                                                                                                                                           | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| -objecto binário incorporado (utilizado no nível da instância)   |                                                                                                                                                                                                                                                      |                    | O cmdatamodel.xsd contém a definição de resourceObject, que é <element name="cm:resourceObject"> <complexType> <choice> <element name="content"> <complexType> <attribute name="value" type="base64Binary"/> </complexType> </element> <element name="url"> <complexType> <attribute name="value" type="anyURI"/> </complexType> </element> </complexType> </element> Tenha em atenção que o tipo seria "base64Binary" ou "hexBinary". Faz parte do ficheiro de esquemas do DB2 Content Manager predefinido, cmdatamodel.xsd, a importar. |
| 3. Tipo de artigo de documento (inclui as seguintes informações) | Elemento cm:entityType com o value="DOCUMENT" para indicar que esta declaração de elemento é um tipo de documento.                                                                                                                                   | GLOBAL, VIEW       | Existem muitas ICMPARTs predefinidas (ICMANNOTATION, ICMBASE, ICMBASETEXT, ICMBASESTREAM, ICMNOTELOG). São tratadas ou representadas de forma semelhante às partes de utilizadores regulares, que farão parte do ficheiro de esquemas.                                                                                                                                                                                                                                                                                                    |
| -Nome de tipo parte                                              |                                                                                                                                                                                                                                                      |                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| -Pesquisável por texto/opção                                     | Caso semelhante ao da definição de tipo de artigo de recurso, com os mesmo elementos cm:textsearchable e cm:textIndexInfo no elemento cm:resourceItemInfo.                                                                                           | GLOBAL             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                          | Informações especificadas em anotações                                                                                                                                    | Âmbito da anotação | Informação de nível de instância                                             |
|--------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|------------------------------------------------------------------------------|
| -Nome da ACL                                     | Elemento cm:ACL com nome e atributos no elemento de referência.<br>Por exemplo: <element ref="cm:ICMBASE" ...><br><xsd:annotation> <xsd:appinfo><br><cm:ACL name="..." /> | GLOBAL             |                                                                              |
| -Nome do GR                                      | Elemento cm:RM com nome e atributos no nome do elemento de referência.                                                                                                    | GLOBAL             |                                                                              |
| -Nome da recolha SMS                             | Elemento cm:SMSCollection com nome e atributos no nome do elemento de referência.                                                                                         | GLOBAL             |                                                                              |
| -Política de novas versões                       | Nome do elemento cm:versionPolicy com um atributo de valor no tipo de enumeração {NEVER, ALWAYS, BY_APPLICATION}.                                                         | GLOBAL             |                                                                              |
| 4. Tipo de parte documento                       | Elemento cm:entityType com o value="PART" para indicar que este elemento um tipo de parte de documento.                                                                   | GLOBAL, VIEW       | Se for um tipo PART predefinido do CM, constará do ficheiro cmdatamodel.xsd. |
| -Nome da classe XDO                              | elemento cm:XDOClass com um atributo de valor.                                                                                                                            | GLOBAL             |                                                                              |
| -Pesquisável por texto/opção                     | Caso semelhante ao da definição de tipo de artigo de recurso, com os mesmos elementos cm:textsearchable e cm:textIndexInfo no elemento cm:resourceItemInfo.               | GLOBAL             |                                                                              |
| Período de retenção do artigo                    | elemento cm:itemRetention com atributos de valor e de unidade O atributo de unidade está no tipo de enumeração {YEAR, MONTH, WEEK, DAY}.                                  | GLOBAL             |                                                                              |
| Iniciar artigo no processo                       | elemento cm:startProcess com um atributo de nome.                                                                                                                         | GLOBAL             |                                                                              |
| Prioridade predefinida                           | elemento cm:defaultPriority com um atributo de valor.                                                                                                                     | GLOBAL             |                                                                              |
| Descrição                                        | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence).                                                             | GLOBAL             |                                                                              |
| Nome ACL do nível de tipo de artigo              | elemento cm:ACL com nome e atributos.                                                                                                                                     | GLOBAL             |                                                                              |
| Sinalizador de Associação ACL ao Nível do Artigo | elemento cm:itemACLBinding com um atributo sinalizador.                                                                                                                   | GLOBAL             |                                                                              |
| Ligação Automática                               | elemento cm:autoLinkEnable com os seguintes elementos descendentes.                                                                                                       | GLOBAL             |                                                                              |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                        | Informações especificadas em anotações                                             | Âmbito da anotação | Informação de nível de instância |
|------------------------------------------------|------------------------------------------------------------------------------------|--------------------|----------------------------------|
| -Regras de Ligação Automática                  | um elemento cm:autoLinkAttributes tem os seguintes atributos:                      |                    |                                  |
| -Tipo de artigo actual                         | {cm:autoLinkAttributes}<br>sourceItemType=<string>                                 | GLOBAL             |                                  |
| -Tipo de artigo a ligar                        | {cm:autoLinkAttributes}<br>cm:targetItemType=<string>                              | GLOBAL             |                                  |
| -Atributo de Ligação Automática                | {cm:autoLinkAttributes}<br>cm:attributeName=<string>                               | GLOBAL             |                                  |
| -Atributo de Ligação Automática                | {cm:autoLinkAttributes}<br>cm:attributeGroupName=<string>                          | GLOBAL             |                                  |
| -Tipo de ligação                               | {cm:autoLinkAttributes}<br>cm:autoLinkType=<string>                                | GLOBAL             |                                  |
| -SMS de Ligação Automática                     | elemento cm:autoLinkingSMSRule com um atributo de valor.                           | GLOBAL             |                                  |
| iniciar sessão                                 |                                                                                    |                    |                                  |
| -Eventos do tipo de artigo para iniciar sessão | elemento cm:itemEventFlag com um atributo de valor.                                | GLOBAL             |                                  |
| Chaves remotas                                 | um elemento cm:foreignKey tem os seguintes atributos:                              | GLOBAL             |                                  |
| -Nome da restrição                             | {cm:foreignKey}<br>cm:constraintName=<string>                                      | GLOBAL             |                                  |
| -Regra de Actualização                         | {cm:foreignKey}<br>cm:updateRule={RESTRICT, NO_ACTION}                             | GLOBAL             |                                  |
| -Regra de Eliminação                           | {cm:foreignKey}<br>cm:deleteRule={RESTRICT, CASCADE, NO_ACTION, SET_NULL}          | GLOBAL             |                                  |
| -Componente origem                             | {cm:foreignKey}<br>cm:sourceComponent=<string>                                     | GLOBAL             |                                  |
| -Tipo de artigo destino                        | {cm:foreignKey}<br>cm:targetItemType=<string>                                      | GLOBAL             |                                  |
| -Tabela destino externa                        | {cm:foreignKey}<br>cm:targetTable=<string>                                         | GLOBAL             |                                  |
| -Pares de Atributos                            | Um elemento cm:attributePair no elemento cm:foreignKey tem os seguintes atributos: | GLOBAL             |                                  |
| -Grupo de atributos origem                     | {cm:attributePair}<br>cm:sourceAttributeGroup=<string>                             | GLOBAL             |                                  |
| -Atributos origem                              | {cm:attributePair}<br>cm:sourceAttribute=<string>                                  | GLOBAL             |                                  |
| -Grupo de atributos destino                    | {cm:attributePair}<br>cm:targetAttributeGroup=<string>                             | GLOBAL             |                                  |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                           | Informações especificadas em anotações                                 | Âmbito da anotação        | Informação de nível de instância |
|-----------------------------------|------------------------------------------------------------------------|---------------------------|----------------------------------|
| -Atributos destino                | {cm:attributePair}<br>cm:targetAttribute=<string>                      | GLOBAL                    |                                  |
| -Nome da coluna da tabela externa | {cm:attributePair}<br>cm:targetTableColumn=<string>                    | GLOBAL                    |                                  |
| O utilizador existe               | elemento cm:userExit com os seguintes atributos:                       | GLOBAL,<br>VIEW           |                                  |
| -Nome de saída                    | {cm:userExit} cm:name=<string>                                         | GLOBAL,<br>VIEW           |                                  |
| -Nome da função                   | {cm:userExit}<br>cm:functionName=<string>                              | GLOBAL,<br>VIEW           |                                  |
| -Nome DLL                         | {cm:userExit} cm:DLLName=<string>                                      | GLOBAL,<br>VIEW           |                                  |
| Módulo de acesso                  | um elemento cm:accessModule tem os seguintes atributos:                | GLOBAL                    |                                  |
| -nome                             | {cm:accessModule}<br>cm:name=<string>                                  | GLOBAL                    |                                  |
| -resultado                        | {cm:accessModule}<br>cm:result=<integer>                               | GLOBAL                    |                                  |
| -estado                           | {cm:accessModule}<br>cm:status=<short>                                 | GLOBAL                    |                                  |
| -versão                           | {cm:accessModule}<br>cm:version=<short>                                | GLOBAL                    |                                  |
| Módulo de Acesso anterior         | elemento cm:previousAccessModule com um atributo de versão.            | GLOBAL                    |                                  |
| Tipo de componente:               |                                                                        |                           |                                  |
| -Grupos de atributos              |                                                                        |                           |                                  |
| -Atributos de referência          |                                                                        |                           |                                  |
| -Atributos                        |                                                                        |                           |                                  |
| -Pesquisável por texto            | elemento cm:textSearchable, com um atributo de valor no tipo booleano. | GLOBAL,<br>LOCAL,<br>VIEW |                                  |
| -Informações de Índice de Texto   | elemento cm:textIndexInfo, com os seguintes atributos:                 | GLOBAL,<br>LOCAL          |                                  |
| -Consolidar contagem              | {cm:textIndexInfo }<br>cm:commitCount=<integer>                        | GLOBAL,<br>LOCAL          |                                  |
| -Formato                          | {cm:textIndexInfo }<br>cm:format=<integer>                             | GLOBAL,<br>LOCAL          |                                  |
| -CCSID de índice                  | {cm:textIndexInfo }<br>cm:CCSID=<integer>                              | GLOBAL,<br>LOCAL          |                                  |
| -Directório de índice             | {cm:textIndexInfo }<br>cm:directory=<string>                           | GLOBAL,<br>LOCAL          |                                  |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                | Informações especificadas em anotações                                                                  | Âmbito da anotação | Informação de nível de instância |
|----------------------------------------|---------------------------------------------------------------------------------------------------------|--------------------|----------------------------------|
| -Código de Idioma de índice            | {cm:textIndexInfo }<br>cm:langCode=<string>                                                             | GLOBAL,<br>LOCAL   |                                  |
| -Alterações mínimas                    | {cm:textIndexInfo }<br>cm:minChanges=<integer>                                                          | GLOBAL,<br>LOCAL   |                                  |
| -Ficheiro de Modelo                    | {cm:textIndexInfo }<br>cm:modelFile=<string>                                                            | GLOBAL,<br>LOCAL   |                                  |
| -Nome do Modelo                        | {cm:textIndexInfo }<br>cm:modelName=<string>                                                            | GLOBAL,<br>LOCAL   |                                  |
| -CCSID do Modelo                       | {cm:textIndexInfo }<br>cm:modelCCSID=<integer>                                                          | GLOBAL,<br>LOCAL   |                                  |
| -Nome UDF                              | {cm:textIndexInfo }<br>cm:UDFName=<string>                                                              | GLOBAL,<br>LOCAL   |                                  |
| -Esquema UDF                           | {cm:textIndexInfo }<br>cm:UDFSchema=<integer>                                                           | GLOBAL,<br>LOCAL   |                                  |
| -Frequência de Actualização            | {cm:textIndexInfo }<br>cm:updateFrequency=<string>                                                      | GLOBAL,<br>LOCAL   |                                  |
| -Unidade de Frequência de Actualização | {cm:textIndexInfo }<br>cm:updateFrequencyUnit=<{MINUTE, HOUR}>                                          | GLOBAL,<br>LOCAL   |                                  |
| -Directório de Trabalho                | {cm:textIndexInfo }<br>cm:workingDir=<string>                                                           | GLOBAL,<br>LOCAL   |                                  |
| -representa o artigo                   | elemento cm:representative com um atributo de valor no tipo booleano.                                   | LOCAL              |                                  |
| -é único                               | elemento cm:unique com um atributo de valor no tipo booleano.                                           | LOCAL              |                                  |
| -É necessário                          |                                                                                                         |                    |                                  |
| -Valor Máx                             |                                                                                                         |                    |                                  |
| -Valor Mín                             |                                                                                                         |                    |                                  |
| -Valor predefinido                     |                                                                                                         |                    |                                  |
| -Atributo de Gestor de Recursos        | elemento cm:isResourceManagerAttr com um atributo de valor no tipo booleano.                            | LOCAL              |                                  |
| -Índices de Bases de Dados             | elemento cm:databaseIndexInfo, com os seguintes atributos:                                              | GLOBAL,<br>LOCAL   |                                  |
| --Nome                                 | {cm:databaseIndexInfo }<br>cm:name=<string>                                                             | GLOBAL,<br>LOCAL   |                                  |
| --Único                                | {cm:databaseIndexInfo }<br>cm:unique=<boolean>                                                          | GLOBAL,<br>LOCAL   |                                  |
| --Esquema de Índice                    | {cm:databaseIndexInfo }<br>cm:indexSchema=<string>                                                      | GLOBAL,<br>LOCAL   |                                  |
| --Atributos                            | Um subelemento, cm:indexedAttribute, no mesmo elemento cm:databaseIndexInfo com os seguintes atributos: | GLOBAL,<br>LOCAL   |                                  |



Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                         | Informações especificadas em anotações                                                                                                                                                                     | Âmbito da anotação | Informação de nível de instância |
|-------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|----------------------------------|
| ---Nome                                         | {cm:indexAttribute }<br>cm:name=<string>                                                                                                                                                                   | GLOBAL,<br>LOCAL   |                                  |
| ---Ordem de Índice                              | {cm:indexAttribute }<br>cm:indexOrder={ASCENDING,<br>DESCENDING}                                                                                                                                           | GLOBAL,<br>LOCAL   |                                  |
| Regra de Eliminação                             | elemento cm:deleteRule com um atributo de valor {RESTRICT, CASCADE}.                                                                                                                                       | LOCAL              |                                  |
| Subcomponentes /<br>componentes<br>descendentes |                                                                                                                                                                                                            |                    |                                  |
| Vista do tipo de artigo:                        | elemento cm:entityType com o value="ITEM" e elemento cm:entityView com o atributo cm:baseEntityType para indicar a que tipo de artigo se refere.                                                           | VIEW               |                                  |
| Vista do tipo de componente:                    |                                                                                                                                                                                                            |                    |                                  |
| -Atributos                                      | elemento cm:representative com um atributo de valor no tipo booleano                                                                                                                                       | VIEW               |                                  |
| --legível                                       | elemento cm:readable com um atributo de valor no tipo booleano. Este conceito não existe no esquema XML. Apenas tem "fixed" em termos de valor ou "prohibited" em termos de definição do nome do atributo. | VIEW               |                                  |
| --gravável                                      | elemento cm:writeable com um atributo de valor no tipo booleano                                                                                                                                            | VIEW               |                                  |
| --consultável                                   | elemento cm:queryable com um atributo de valor no tipo booleano                                                                                                                                            | VIEW               |                                  |
| --excludeRow                                    | elemento cm:excludeRow com um atributo de valor no tipo booleano                                                                                                                                           | VIEW               |                                  |
| --Valor de comparação de vistas                 | elemento cm:viewCompareValue com um atributo de valor                                                                                                                                                      | VIEW               |                                  |
| --Operador de vistas                            | elemento cm:viewOperator com um atributo de valor no tipo de enumeração {OPCODE_EQ}                                                                                                                        | VIEW               |                                  |
| --SequenceNumber de vistas                      | elemento cm:viewSequenceNo com um atributo de valor                                                                                                                                                        | VIEW               |                                  |
| Entidade federada                               | Elemento cm:entityType com o value="federated"                                                                                                                                                             |                    |                                  |
| -nome                                           |                                                                                                                                                                                                            |                    |                                  |
| -descrição                                      |                                                                                                                                                                                                            |                    |                                  |
| -Pesquisável por texto                          |                                                                                                                                                                                                            |                    |                                  |

Tabela 44. Como se correlacionam os objectos de modelos de dados DB2 Content Manager com as anotações em esquemas de armazenamento (continuação)

| Objecto                                                                                                                                          | Informações especificadas em anotações                                                                       | Âmbito da anotação | Informação de nível de instância |
|--------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|--------------------|----------------------------------|
| -Activado com a criação da pasta federada nativa.                                                                                                |                                                                                                              |                    |                                  |
| -Atributo:                                                                                                                                       |                                                                                                              |                    |                                  |
| --nome                                                                                                                                           |                                                                                                              |                    |                                  |
| --descrição                                                                                                                                      | elemento cm:description com um valor de atributo e um atributo xsi:lang (para indicar a que idioma pertence) | GLOBAL             |                                  |
| --tipo de dados e outras informações de atributos como, por exemplo, comprimento, precisão, escala, max, min, anulável, consultável e de escrita |                                                                                                              |                    |                                  |
| Correlação de esquemas:                                                                                                                          | Um ou mais elementos cm:schemaMapping com o seguintes atributos:                                             | GLOBAL             |                                  |
| -Nome da entidade federada                                                                                                                       | cm:fedEntityName=<string>                                                                                    | GLOBAL             |                                  |
| -Nome do atributo federado                                                                                                                       | cm:fedAttrName=<string>                                                                                      | GLOBAL             |                                  |
| -Nome do servidor nativo                                                                                                                         | cm:serverName=<string>                                                                                       | GLOBAL             |                                  |
| -Tipo de servidor nativo                                                                                                                         | cm:serverType=<string>                                                                                       | GLOBAL             |                                  |
| -Nome da entidade nativa                                                                                                                         | cm:nativeEntityName=<string>                                                                                 | GLOBAL             |                                  |
| -Nome do atributo nativo                                                                                                                         | cm:nativeAttrName=<string>                                                                                   | GLOBAL             |                                  |

## Exemplo: esquema XML

O exemplo seguinte mostra um fragmento exemplo de esquema de armazenamento (XSD) para o tipo de artigo de apólice de seguros XYZ:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <xsd:import namespace="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 schemaLocation="cmdatamodel.xsd"/>
 <xsd:attribute name="XYZ_VIN"><xsd:annotation><xsd:appinfo>
 <cm:description value="Vehicle Identification Number (Content
 Manager Sample Attribute)" xsi:lang="ENU"/><cm:stringType
 value="OTHER"/></xsd:appinfo></xsd:annotation><xsd:simpleType>
 <xsd:restriction base="xsd:string"><xsd:length value="17"/>
```

```

</xsd:restriction></xsd:simpleType></xsd:attribute>
<xsd:attribute name="XYZ_InsrdLName">...</xsd:attribute>
<xsd:attribute name="XYZ_InsrdFName">...</xsd:attribute>
<xsd:attribute name="XYZ_ZIPCode">...</xsd:attribute>
<xsd:attribute name="XYZ_City">...</xsd:attribute>
<xsd:attribute name="XYZ_State">...</xsd:attribute>
<xsd:attribute name="XYZ_Street">...</xsd:attribute>
<xsd:attribute name="XYZ_PolicyNum">...</xsd:attribute>
<xsd:element name="XYZ_InsPolicy"><xsd:annotation><xsd:appinfo>
 <cm:description value="Insurance Policy (Content Manager Sample
 Item Type)" xsi:lang="ENU"/><cm:ACL name="XYZInsurancePolicyACL"/>
 <cm:versionPolicy value="ALWAYS"/><cm:maximumVersions value="10"/>
 <cm:entityType value="DOCUMENT"/><cm:itemRetention unit="YEAR"
 value="0"/><cm:itemACLBinding flag="0"/><cm:itemEventFlag value=
 "0"/><cm:accessModule name="DUMMY" status="0" version="0"/><cm:
 previousAccessModule value="DUMMY"/></xsd:appinfo></xsd:annotation>
 <xsd:complexType><xsd:sequence>
 <xsd:element maxOccurs="1" minOccurs="0" ref="cm:properties"/>
 <xsd:element maxOccurs="1" minOccurs="0" ref="cm:links"/>
 <xsd:element maxOccurs="unbounded" minOccurs="0" name="XYZ_Insured">
 ...</xsd:element><xsd:element maxOccurs="unbounded" minOccurs="0"
 name="XYZ_VIN">...</xsd:element><xsd:element maxOccurs="unbounded"
 minOccurs="0" ref="ICMBASE">...</xsd:element><xsd:element
 maxOccurs="unbounded" minOccurs="0" ref="ICMBASETEXT">...
 </xsd:element><xsd:element maxOccurs="unbounded" minOccurs="0" ref=
 "ICMNOTELOG">...</xsd:element></xsd:sequence>...</xsd:complexType>
 </xsd:element>
 <xsd:element name="ICMBASETEXT">...</xsd:element>
 <xsd:element name="ICMNOTELOG">...</xsd:element>
</xsd:schema>

```

## Tipos de XML não suportados nos esquemas de armazenamento do DB2 Content Manager

Os esquema de armazenamento do DB2 Content Manager não suporta os seguintes tipos de dados primitivos:

- cadeia (tem de ser associada com uma das propriedades de comprimento)
- booleano
- duração
- gYearMonth
- gYear
- gMonthDay
- gDay
- gMonth
- hexBinary (only support base64Binary)
- QName
- NOTATION

Os esquemas de armazenamento do DB2 Content Manager não suportam os seguintes tipos de dados derivados:

- normalizedString
- token
- linguagem
- NMTOKEN
- NMTOKENS
- Nome
- NCName

- ID
- IDREF
- IDREFS
- ENTITY
- ENTITIES
- nonPositiveInteger
- negativeInteger
- long
- byte
- nonNegativeInteger
- unsignedLong
- unsignedInt
- unsignedShort
- unsignedByte
- positiveInteger

## Restrições à conversão de esquemas de armazenamento do DB2 Content Manager

As restrições seguintes também se aplicam à conversão de esquemas de armazenamento do DB2 Content Manager:

- Os valores de atributo minLength e maxLength (quando especificados) têm de ser iguais, o que descreve o tipo de dados DK\_CM\_CHAR.
- Nome de elemento, que se correlaciona com o nome de componente, não pode ser utilizado num espaço de símbolos diferentes.
- Atributos com o mesmo nome têm de ter o mesmo tipo de base. Excepção: Algumas propriedades dos atributos com o mesmo nome podem ser diferentes como, por exemplo, maxInclusive e minInclusive.
- Os atributos xsi:type e xsi:nil não são suportados no documento de instância.
- Não é permitida a definição de tipo recorrente. Por exemplo, as definições seguintes não são permitidas:

```
<xs:element name="Section">
 <xs:complexType>
 <xs:sequence>
 <xs:element ref="Section" minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute ref="title" use="required"/>
 <xs:attribute ref="content" use="required"/>
 </xs:complexType>
</xs:element>
```

---

## Importar e exportar objectos de instâncias de dados do DB2 Content Manager como XML

A classe de serviços de instâncias XML, DKXMLDataInstanceService, contém dois métodos para importar e exportar artigos XML: ingest() e extract().

Este métodos ingest() e extract() tomam o ficheiro XML que está em conformidade com o esquema de armazenamento (que pode ser exportado através da API de extracção ou do cliente de administração do sistema, em qualquer tipo de artigo) para estruturar os objectos de instâncias de dados (como, por exemplo, artigos e

documentos com partes). Os respectivos parâmetros de entrada e de devolução funcionam de modo semelhante aos antigos métodos da Versão 8, `toXML()` e `fromXML()`.

Os métodos `ingest()` e `extract()` suportam os seguintes formatos:

#### Formatos de entrada de artigos XML

##### **DKXMLDOMItem**

modelo de objecto de documentos (DOM)

##### **DKXMLStreamItem**

sequência de entrada de dados (processada com SAX)

##### **DKXMLStringItem**

cadeia

#### Formatos de saída de dados de artigos XML

##### **DKXMLDOMItem**

DOM (predefinição)

##### **DKXMLStringItem**

cadeia

`DKXMLDOMItem` inclui um método que pode converter um artigo XML de formato DOM em formato de sequência de entrada de dados.

### Exemplo: instância de um artigo XML

O exemplo seguinte ilustra uma instância de artigo exemplo que se rege pelo esquema de armazenamento de apólice de seguros da XYZ:

```
<Item><ItemXML>
<XYZ_InsPolicy XYZ_Street="532 Camino Viejo"
 XYZ_City="Marina" XYZ_State="CA" XYZ_ZIPCode="90546"
 XYZ_PolicyNum="57904965371" xmlns="">
 <XYZ_Insured XYZ_InsrdFName="Edward" XYZ_InsrdLName="Smith" />
 <XYZ_Insured XYZ_InsrdFName="Jennifer" XYZ_InsrdLName="Smith" />
 <XYZ_VIN XYZ_VIN="ICLA44P5KL9876543" />
 <ICMBASE><resourceObject MIMEType="image/tiff"
 xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"><label
 name="policyForm" /></resourceObject></ICMBASE>
</XYZ_InsPolicy>
</ItemXML></Item>
```

#### Informações relacionadas



Importar e exportar objectos de modelo de dados do DB2 Content Manager como ficheiros de esquema XML (XSD)

## Exportar artigos DDO do DB2 Content Manager como artigos XML

O método `extract()` no `DKXMLDataService` converte um DDO (incluindo todos os DDOs, ligações e referências de componentes descendentes) num objecto `DKXMLItem`.

Este objecto `DKXMLItem` contém os seguintes dados:

- Documento XML que representa a versão do artigo, incluindo todos os componentes descendentes.
- Propriedades, incluindo atributos do sistema, atributos de recurso e informações de ligação.

- Qualquer conteúdo de partes de recurso binárias.

**Importante:** Passe `DKConstant.DK_CM_XML_EMBED_UNIQUE_IDENTIFIER` para `TRUE` de modo a incluir o part number do conteúdo do recurso.

O método `extract()` aceita o DDO (e várias opções) como parâmetros de entrada. As diferentes opções incluem:

- Qual o formato XML para exportar o artigo e suas propriedades:

**DKXMLDOMItem**

`DKConstant.DK_CM_XML_DOM_FORMAT` (é a predefinição)

**DKXMLStreamItem**

`DKConstant.DK_CM_XML_RESOURCE_STREAM_FORMAT`

**DKXMLStringItem**

`DKConstant.DK_CM_XML_DOM_FORMAT`

- Qual o formato de saída para exportar o conteúdo do recurso como (URL ou sequência de entradas). A predefinição é URL. Caso selecione sequência de entradas, o sistema vai gerar etiquetas únicas para identificar cada recurso. Estas etiquetas podem ser encontradas nas propriedades do recurso.
- Se se deve incluir o PID e o part number no artigo XML.
- Se se deve exportar as propriedades do artigo como um documento XML separado. Use esta opção para excluir todas as informações que pertencem ao DB2 Content Manager do artigo XML.

## Exemplo: Java

O seguinte exemplo apresenta a entrada de um artigo ddo e a devolução do mesmo como um documento XML `xmlObj` (em formato DOM com o PID incorporado), a devolução das propriedades do sistema e do recurso num ficheiro em separado e a devolução do conteúdo do recurso como uma sequência de entradas.

```
//create an instance of DKXMLDataInstanceService
DKXMLDataInstanceService instService = new DKXMLDataInstanceService(dsICM);
DKXMLDOMItem xmlObj =
(DKXMLDOMItem) instService.extract(ddo,DKConstant.DK_CM_XML_DOM_FORMAT +
DKConstant.DK_CM_XML_SYSTEM_PROPERTY_REFERENCE +
DKConstant.DK_CM_XML_RESOURCE_PROPERTY_REFERENCE +
DKConstant.DK_CM_XML_RESOURCE_STREAM_FORMAT +
DKConstant.DK_CM_XML_EMBED_UNIQUE_IDENTIFIER);
//get the XML document representing item version
Document xmlDocument = xmlObj.getXMLItem();
//get the XML document with properties
Document propertyDocument = xmlObj.getItemProperties();
//get content labels
Set resLabels = xmlObj.getContentLabels();
//create an iterator
Iterator iter = resLabels.iterator();
//iterate over the set to get labels and resource contents
while (iter.hasNext()) {
 //get the label from the iterator
 String label = (String)iter.next();
 // get resource content as input stream from xml object
 BufferedInputStream inStream = new
 BufferedInputStream(xmlObj.getContentAsStream(label));
}
```

## Importar artigos XML como artigos DDO do DB2 Content Manager

O método `ingest()` no `DKXMLDataInstanceService` converte um objecto `DKXMLItem` num DDO no servidor do DB2 Content Manager. Estes construtores extraem o conteúdo de um documento XML, criam um `DKDDO` correspondente e qualquer `dkXDO` a ele associado. Em seguida, pode invocar o método `add` no DDO para adicionar o objecto ao DB2 Content Manager.

O novo DDO pertence a um tipo de artigo do DB2 Content Manager Versão 8 ou a uma classe de índice de uma versão anterior do Content Manager e só pode ser armazenado no DB2 Content Manager.

A importação de um ficheiro XML permite armazenar o ficheiro XML original como um XDO, ou seja, o utilizador não perde o ficheiro XML no processo de importação, tornando o próprio ficheiro XML disponível para uma futura utilização.

Quando importa o conteúdo XML, tenha em atenção os seguintes factos:

- Apenas pode importar para o DB2 Content Manager ou uma versão anterior do DB2 Content Manager.
- Os ficheiros XML com conteúdo para importação têm de estar em conformidade com o esquema de armazenamento do tipo de artigo correspondente, que pode ser exportado através da API ou através do cliente de administração do sistema.
- A importação e exportação de XML são suportadas só pelas APIs de Java.

O método `input()` aceita os seguintes parâmetros de entrada de dados:

- Documento XML num `DKXMLDOMItem`, `DKXMLStreamItem` ou `DKXMLStringItem`. Se introduzir `DKXMLStreamItem`, o operador SAX converte a sequência de entrada de dados num objecto DDO (não DOM).
- DDO já existente para preencher os dados XML (facultativo).
- Conteúdo de recursos como um objecto `DKXMLItem` num formato de sequência de entrada de dados. Utilizando o método `DKXMLItem.setContentAsStream()`, pode criar etiquetas exclusivas para as propriedades do recurso para que `ingest()` as interprete.
- Propriedades como, por exemplo, atributos do sistema, atributos de recurso e informações de ligação. Pode incorporá-las no documento XML original ou importá-las como um documento XML independente do documento XML original que descreve o artigo. Pode facultar este documento através do método `setItemProperties()` ou no construtor.

### Exemplo: Java

O exemplo seguinte introduz um artigo XML `XMLFile` e as respectivas propriedades (do sistema e recursos num ficheiro `XMLProperties` em separado) como sequências de entrada de dados; e devolve um ddo do DB2 Content Manager.

```
//create file stream for XML document representing artigo version
FileInputStream xmlDocument = new FileInputStream(XMLFile);
//create file stream for XML document representing properties
//properties include system properties, resource properties
FileInputStream properties = new FileInputStream(XMLProperties);
//Create an instance of DKXMLStreamItem
DKXMLStreamItem xmlItem = new DKXMLStreamItem(XMLFile, XMLProperties);
//set value for resource content label
String contentLabel = "AAA";
```

```
//Set resource content into xmlItem
xmlItem.setContentAsStream(contentLabel, contentStream);
//create an instance of DKXMLDataInstanceService
DKXMLDataInstanceService instService = new DKXMLDataInstanceService(dsICM);
//call ingest on instance service
DKDDO ddo = (DKDDO) instService.ingest(xmlItem, options);
ddo.add()
```

#### Informações relacionadas



Importar e exportar objectos de instâncias de dados do DB2 Content Manager como XML

---

## Importar e exportar dependências de objectos XML

Podem ocorrer situações em que os objectos de modelo de dados e os objectos administrativos requeiram a existência de outras definições (objectos de dependência) no servidor. Por exemplo, é necessário ter um utilizador definido antes de poder definir um grupo de utilizadores para o mesmo.

Por predefinição, o método `extract()` só exporta o objecto, sem objectos de dependência. Para evitar problemas de falta de dependências, pode especificar uma das seguintes opções de exportação de objectos para XML:

#### **DK\_CM\_XML\_EXPORT\_PREREQUISITE**

Exporta todos os objectos de dependência com o objecto.

#### **DK\_CM\_XML\_EXPORT\_DM\_ONLY\_PREREQUISITE**

Exporta apenas objectos de dependência de modelos de dados.

#### **DK\_CM\_XML\_EXPORT\_SA\_ONLY\_PREREQUISITE**

Exporta apenas objectos de dependência administrativa (incluindo autorização, autenticação e configuração do servidor de bibliotecas).

#### **DK\_CM\_XML\_EXPORT\_RM\_ONLY\_PREREQUISITE**

Exporta apenas objectos de dependência de configuração do gestor de recursos (do lado do servidor de bibliotecas).

#### **DK\_CM\_XML\_EXPORT\_DR\_ONLY\_PREREQUISITE**

Exporta apenas objectos de dependência de encaminhamento de documentos.

Durante uma importação, se os objectos de dependência não existirem no sistema destino, é registada uma excepção ou o processo é cancelado (consoante a opção de processamento definida).

---

## Extrair conteúdo de diferentes origens XML

Os métodos DKDDO podem extrair conteúdos de uma variedade de origens XML, incluindo entrada de dados padrão, ficheiros, memórias tampão e endereços da Web (URLs).

Invoke os métodos DKDDO para extrair conteúdos da sua origem XML e para iniciar o processo de importação.

Aqui estão alguns exemplos de cada origem XML:

### XML de um ficheiro

#### Java

```
xmlSource = new DKNVPair("FILE", "dlsamp01.xml");
```



## XML de uma memória tampão

Java

```
File file = new File("dlsamp01.xml");
int fileSize = (int) file.length();
byte[] data = new byte[fileSize];
DataInputStream dis = new DataInputStream
 (new FileInputStream(file));
dis.readFully(data);
String strBuffer = new String(data);
DKNVPair xmlSource = new DKNVPair("BUFFER",
 strBuffer);
int importOptions=DK_CM_XML_VALIDATION;
```

## XML de um endereço da Web (URL)

Java

```
xmlSource = new DKNVPair("URL",
 "file:///d://myxml//dlsamp01.xml");
// replace file:///d:// with
http://www.webaddress.com/ for URL
Int importOptions=0;
```

---

## Correlacionar um esquema definido pelo utilizador com um esquema de armazenamento através de uma ferramenta de correlação de esquemas XML

O DB2 Content Manager faculta a interface gráfica e APIs para converter um esquema definido pelo utilizador num esquema de armazenamento que possa ser importado para o sistema.

A ferramenta também pode gerar um script de consulta XSLT que pode ser guardado como parte de uma correlação num repositório. Com o script, pode programar uma aplicação que se converta automaticamente em documentos XML do esquema definido pelo utilizador para o esquema de armazenamento.

A ferramenta de correlação de esquemas XML suporta as seguintes situações quando desenvolve uma correlação de esquemas:

### **Criar correlações de esquemas com esquema de armazenamento totalmente novo**

Pode converter o esquema definido pelo utilizador num esquema de armazenamento totalmente novo e pode modificar tanto o esquema de armazenamento como as correlações. Pode, assim, criar um novo tipo de artigo a partir do esquema de armazenamento, atribuir um nome à correlação e guardar a correlação num repositório.

### **Criar correlações de esquemas com esquema de armazenamento já existente**

Pode converter o esquema definido pelo utilizador num esquema de armazenamento previamente criado, correlacionado manualmente o esquema do utilizador com o esquema de armazenamento. Pode, assim, invocar a função da ferramenta que vai gerar o novo script de consulta XSLT. Pode, então, atribuir um nome à correlação e guardar a correlação num repositório.

### **Rever correlações de esquemas existentes**

Pode reabrir uma correlação previamente criada (com o nome de correlação) e modificar tanto esta correlação como o esquema de

armazenamento. Pode, de seguida, guardar o esquema de armazenamento modificado, o esquema definido pelo utilizador, e o novo script da consulta XSLT novamente no repositório.

Como primeiro passo da utilização de APIs, deve utilizar os métodos na classe `DKSchemaConverter` para converter o esquema XML. Os métodos efectuem as seguintes tarefas:

**convert()**

Converte o esquema definido pelo utilizador num esquema de armazenamento e, opcionalmente, guarda a correlação como um script de consulta XSLT num repositório.

**getStorageSchema()**

Obtém o esquema de armazenamento convertido.

**getXSLTQuery()**

Obtém o script da consulta XSLT que pode ser automaticamente convertido em documentos XML do esquema definido pelo utilizador para o esquema de armazenamento.

Como segundo passo de utilização de APIs, deve utilizar os métodos na classe `DKDocumentConverter` para converter os documentos XML. Os métodos efectuem as seguintes tarefas:

**getSchemaMappingNames()**

Obtém os nomes das correlações de esquema a partir do repositório.

**getXSLTQuery()**

Obtém o script da consulta XSLT que pode ser automaticamente convertido em documentos XML do esquema definido pelo utilizador para o esquema de armazenamento.

**transformXMLDocument()**

Transforma um documentos XML utilizando o script de consulta do XSLT que o utilizador obteve.

**deleteSchemaMapping()**

Elimina as correlações de esquema do repositório.

## **Suporte de serviços XML em z/OS**

As APIs da ferramenta de correlação de esquemas XML não são suportadas nativamente em z/OS, o que significa que não é possível executar as APIs, o que inclui as classes `DKSchemaConverter`, `DKDocumentConverter` e `DKMapperException` (ou qualquer classe com `ibm.mm.sdk.xml.schema.*`) em z/OS para a transformação do esquema. Contudo, a transformação tem de ser executada utilizando a ferramenta de correlação de esquemas XML numa plataforma não z/OS (como, por exemplo, Windows) em primeiro lugar e o ficheiro XSD ou XML resultante pode ser importado para o DB2 Content Manager em z/OS. A importação pode ser executada de uma de três formas:

- Utilizando a interface GUI da ferramenta de correlação de esquemas XML directamente (estabelecendo ligação ao servidor do DB2 Content Manager em z/OS).
- O ficheiro XSD/XML pode ser descarregado para z/OS e importado utilizando a API de serviços XML em z/OS.
- Utilizando as APIs XML numa plataforma não z/OS para importar o ficheiro XSD/XML transformado para o DB2 Content Manager em z/OS.

## Exemplo: Converter esquemas XML

Java

```
import com.ibm.mm.sdk.common.DKException;
import com.ibm.mm.sdk.cs.DKDatastoreICM;
import com.ibm.mm.sdk.xml.schema.DKDocumentConverter;
import com.ibm.mm.sdk.xml.schema.DKMapperException;
...
DKDatastoreICM cmDatastore = new DKDatastoreICM();
cmDatastore.connect(cmDatabase, cmUser, cmPassword, "");
System.out.println("Connected.");
File inputSchema = new File (inputUserSchema);
DKSchemaConverter converter = new DKSchemaConverter
(cmDatastore);
if (mapName == null) {
 if (converter.convert(inputSchema.toURL(),
rootElementName)==false)
 {
 System.err.println("dkConvert returned null.");
 }
} else {
 if (converter.convert(inputSchema.toURL(),
rootElementName,
mapName) == false)
 {
 System.err.println("dkConvert returned null.");
 }
}

System.out.println("STORAGE SCHEMA:");
System.out.println(converter.getStorageSchema());
System.out.println("XSLT Scripts");
String scripts[] = converter.getXSLTQuery();
System.out.println(scripts[0]);
System.out.println("-----");
System.out.println(scripts[1]);
```

## Exemplo: Converter documentos XML

Java

```
import com.ibm.mm.sdk.common.DKException;
import com.ibm.mm.sdk.cs.DKDatastoreICM;
import com.ibm.mm.sdk.xml.schema.DKDocumentConverter;
import com.ibm.mm.sdk.xml.schema.DKMapperException;
...
DKDatastoreICM cmDatastore = new DKDatastoreICM();
cmDatastore.connect(cmDatabase, cmUser, cmPassword, "");
System.out.println("MAPPING NAMES:");
Collection names=DKDocumentConverter.
 getSchemaMappingNames(cmDatastore);
System.out.println(names);
if (mapName == null)
 return;
String[] query=DKDocumentConverter.
getXSLTQuery(cmDatastore, mapName);
System.out.println("XSLT Scripts for " + mapName);
if (query == null)
 System.out.println("NONE.");
else {
 for (int i = 0; i < query.length; i++) {
 if (i > 0)
 System.out.println("-----");
 System.out.println(query[i]);
 }
}
```

```

 if (inputXMLDoc == null)
 return;
 File inputFile = new File(inputXMLDoc);
 File outputFile = new File("APIoutput.xml");
 DKDocumentConverter.transformXMLDocument
 (inputFile.toURL(),
 query, outputFile);
 System.out.println("Output in APIoutput.xml");

```

#### Informações relacionadas



Correlacionar e importar esquemas XML

## Programar operações de tempo de execução através dos JavaBeans XML

Os JavaBeans XML são classes de Java que fornecem as interfaces adequadas para as APIs XML do conector do DB2 Content Manager e os JavaBeans do DB2 Information Integrator for Content. Também funcionam como o nível de comunicação entre os serviços da Web e as APIs do conector.

Os JavaBeans XML podem efectuar operações de tempo de execução como, por exemplo, importar, exportar, pesquisar, criar, actualizar, obter, eliminar e encaminhar objectos. Não suportam funções de administração do sistema.

Se optar por programar aplicações que comuniquem com o DB2 Content Manager directamente através de JavaBeans XML, pode dirigir os pedidos XML directamente para o bean CMBXMLServices (de forma semelhante ao que os serviços da Web fazem). Estes pedidos XML têm de se reger pela estrutura descrita no esquema cmbmessages.xsd.

Ao efectuar pedidos aos beans de Java XML, poderá ser necessário especificar o atributo URI de um artigo ou de uma pasta com os quais pretenda trabalhar. O URI é o identificador de um recurso exclusivo e pode assumir uma das duas formas. O PID do artigo conforme está armazenado no DB2 Content Manager é um URI válido que identifica de forma exclusiva o artigo. O PID é especificado como o URI de quaisquer artigos ou pastas em todos os pedidos aos serviços da Web e beans XML. Quando recebe respostas dos serviços da Web, o URI é devolvido sob a forma de um URL.

```

http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
92 3 ICM8 icmnlbdb13 XYZ InsPolicy59 26 A1001001A04I10B45558I8302118
93 A04I10B45558I830211 14 1029&server=icmnlbdb&dsType=ICM

```

Também pode utilizar o URL no exemplo anterior como o valor para o atributo URI em todos os pedidos aos serviços da Web ou aos beans XML.

O seguinte exemplo de JavaBean define um bean CMBXMLMessage para enviar um pedido de pesquisa XML directamente para este:

#### Exemplo: Java

```

public class TXMLSearch2 {
 public static void main(String[] args) throws Exception {
 String dstype = args[0];
 String server = args[1];
 String userid = args[2];
 String password = args[3];
 String entity = args[4];
 String condition = "";
 // Create beans

```

```

CMBXMLServices xmlServices = new CMBXMLServices();
// Create the search request message and get the reply message
CMBXMLMessage reply = search(xmlServices, dstype, server,
 userid, password, entity, condition);
System.out.println("Search reply: " + reply.getAsString());
}
static public CMBXMLMessage search(CMBXMLServices xmlServices,
String dstype, String server, String userid, String password,
String entity, String condition)
 throws CMBException, Exception
{
 return search(xmlServices, dstype, server, userid, password,
 entity, condition, null);
}
static public CMBXMLMessage search(CMBXMLServices xmlServices,
String dstype, String server, String userid, String password,
String entity, String condition, String maxResults)
 throws CMBException, Exception
{
 // Create the query string
 int queryType = CMBBaseConstant.CMB_QS_TYPE_XPATH;
 String queryString = "/" + entity;
 queryString += "[" + condition + "]";
 String connectString = "";
 // If the server name is followed by a parenthesized string,
 // use that string for the connect string.
 // e.g. ICMNLSDB(SCHEMA=ICMADMIN)
 if (server.indexOf("(") > 0) {
 connectString = server.substring(server.indexOf("(") + 1);
 server = server.substring(0, server.indexOf("("));
 if (connectString.endsWith("(")) {
 connectString = connectString.substring(0,
 connectString.length() - 1);
 }
 }
}
// continued...

```

Em seguida, para enviar um pedido de pesquisa XML utilizando o exemplo anterior, pode transmitir o *nome do servidor* (ICMNLSDB no exemplo) e *connectString* como SCHEMA=ICMADMIN:

### Exemplo: Java

```

StringBuffer XMLBuffer = new StringBuffer();
String maxResString = "";
if (maxResults != null) maxResString="maxResults=\\\" +
maxResults + "\\\"";
XMLBuffer.append("<?xml version='1.0' encoding='UTF-8' ?>");
XMLBuffer.append("<RunQueryRequest " + maxResString +
" version='\" + CMBXMLConstant.CMB_LATEST_VERSION +
"\" retrieveOption='\" + CMBXMLConstant.CMB_RETRIEVE_CONTENT +
"\" contentOption='\" + CMBXMLConstant.CMB_CONTENT_ATTACHMENTS +
"\" xmlns='\" + CMBXMLConstant.namespaceURI +
"\" xmlns:cm='\" + CMBXMLConstant.cmNamespaceURI + "\">");
XMLBuffer.append("<AuthenticationData connectString='\" +
connectString + "\" configString='\">");
 XMLBuffer.append("<ServerDef>");
 XMLBuffer.append(" <ServerType>" + dstype + "</ServerType>");
 XMLBuffer.append(" <ServerName>" + server + "</ServerName>");
XMLBuffer.append("</ServerDef>");
XMLBuffer.append("<LoginData>");
XMLBuffer.append(" <UserID>" + userid + "</UserID>");
XMLBuffer.append(" <Password>" + password + "</Password>");
XMLBuffer.append("</LoginData>");
 XMLBuffer.append("</AuthenticationData>");
XMLBuffer.append("<QueryCriteria>");
 XMLBuffer.append("<QueryString>" + queryString + "</QueryString>");

```

```

XMLBuffer.append("</QueryCriteria>");
XMLBuffer.append("</RunQueryRequest>");
System.out.println("XMLRequest: \n\n" + XMLBuffer.toString() + "\n\n");
CMBXMLMessage doc = new CMBXMLMessage(XMLBuffer.toString(), null);
// Search using the makeRequest method on the CMBXMLServices bean
System.out.println("Performing search");
System.out.println(XMLBuffer.toString());
CMBXMLMessage reply = xmlServices.makeRequest(doc);
System.out.println("Search reply");
System.out.println(reply.getAsString());
if (reply.getAttachments() != null) System.out.
 println("There are " + reply.getAttachments().length + " attachments");
return reply;
}
}

```

## Enumerar servidores do DB2 Content Manager com ListServerRequest

Criar um <ListServerRequest> para enumerar todos os servidores do DB2 Content Manager disponíveis.

Para enumerar todos os servidores do DB2 Content Manager disponíveis, crie um <ListServerRequest> que identifique as informações seguintes (o texto entre parênteses rectos é facultativo):

```

<ListServerRequest>
 [<ServerType>ICM</ServerType>]
</ListServerRequest>

```

### Elementos <ListServerRequest>

#### <ServerType> (facultativo)

Identifica os tipos de definições de servidor a enumerar. Se não especificar um <ServerType>, o pedido devolve definições de todos os servidores do DB2 Content Manager Versão 8 que o sistema conhece.

### Exemplo: pedido XML

O exemplo seguinte apresenta a lista de todos os servidores de tipo ICM:

```

<ListServersRequest
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <ServerType>ICM</ServerType>
</ListServersRequest>

```

Em resultado, o DB2 Content Manager devolve um XML <ListServerReply> que devolve objectos <ServerDef> para todos os servidores disponíveis.

### Exemplo: resposta XML

```

<ListServersReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <ServerDef>
 <ServerType>ICM</ServerType>
 <ServerName>icm1sdb</ServerName>
 </ServerDef>
 <ServerDef>
 <ServerType>ICM</ServerType>
 <ServerName>CMA30</ServerName>
 </ServerDef>
</ListServersReply>

```

```

 <ServerType>ICM</ServerType>
 <ServerName>cma15</ServerName>
 </ServerDef>
</ListServersReply>

```

## Autenticar pedidos de serviços da Web para segurança

Cada vez que efectua um pedido aos serviços da Web, tem de transmitir um ID de utilizador do DB2 Content Manager e respectiva palavra-passe, ou um token credencial do WebSphere associado ao utilizador do DB2 Content Manager.

Se o utilizador não tiver o privilégio para efectuar esse pedido, o pedido não é processado e é devolvido um erro da resposta SOAP. Por exemplo, se o utilizador quiser efectuar uma alteração à política de segurança, mas só tiver privilégios de vista, o utilizador não pode efectuar qualquer alteração à política.

**Importante:** Por predefinição, o ID de utilizador e respectiva palavra-passe transmitidos no pedido de serviços da Web não são encriptados. Não será significativo se todos os pedidos de serviço da Web forem processados dentro de uma firewall. No entanto, se o cliente estiver fora de uma firewall, deve utilizar SSL para os pedidos SOAP.

Para autenticar os pedidos de serviço da Web, crie um objecto <AuthenticationData>. Tem de incluir este objecto em cada pedido.

```

<AuthenticationData connectString="string"
 configString="string"
 [connectToWorkflow="boolean"] >
 <ServerDef>
 <ServerName>string</ServerName>
 [<ServerType>ICM</ServerType>]
 </ServerDef>
 <!-- You can specify either a user ID/password or a WebSphere SSO credential: -->
 <LoginData>
 <UserID>string</UserID>
 <Password>string</Password>
 <!-- or --> <Credential>base64Binary</Credential>
 </LoginData>
</AuthenticationData>

```

### Elementos <AuthenticationData>

#### <ServerDef> (necessário)

Identifica o <ServerName> do servidor de conteúdos (por exemplo, concord) e um <ServerType> opcional (a predefinição é ICM).

#### <LoginData> (necessário)

Autentica o utilizador pelo ID de utilizador/palavra-passe ou uma credencial SSO do WebSphere.

### Atributos <AuthenticationData>

#### connectString (facultativo)

Transmite uma propriedade específica do servidor (além de ID de utilizador e palavra-passe) para estabelecer uma ligação ao servidor. Ao enviar pedidos para o bean CMBXMLServices directamente, especifique sempre SCHEMA=userid, em que userid corresponde ao esquema da base de dados do DB2 Content Manager V8 à qual está a estabelecer ligação. Normalmente, é ICMADMIN.



**configString (facultativo)**

Transmite um valor de propriedade CMDBConnection para ajudar a construir uma instância dkDatastore.

**connectToWorkflow (facultativo)**

Comuta uma opção para ligar ao servidor do fluxo de trabalho. A predefinição é TRUE.

**Informações relacionadas**

Enumerar servidores do DB2 Content Manager com ListServerRequest

## Alterar a palavra-passe nos pedidos XML

<ChangePasswordRequest> altera a palavra-passe no DB2 Content Manager.

Para alterar a palavra-passe no DB2 Content Manager, crie um <ChangePasswordRequest > que identifique as informações de esquema XML seguintes:

```
<ChangePasswordRequest>
 <AuthenticationData> ... </AuthenticationData>
 <NewPassword>string</NewPassword>
</ChangePasswordRequest>
```

### Elementos <ChangePasswordRequest>

**<AuthenticationData> (necessário)**

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

**<NewPassword> (necessário)**

Especifica a nova palavra-passe para substituir a anterior no <AuthenticationData>.

### Exemplo: pedido XML

O seguinte exemplo altera a palavra-passe do ID de utilizador testuser para passw0rd2:

```
<ChangePasswordRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>testuser</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <NewPassword>passw0rd2</NewPassword>
</ChangePasswordRequest>
```

Como resultado, o DB2 Content Manager devolve um <ChangePasswordReply > XML que indica o sucesso:

### Exemplo: resposta XML

```
<ChangePasswordReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</ChangePasswordReply>
```

**Informações relacionadas**

Autenticar pedidos de serviços da Web para segurança



## Enumerar entidades do DB2 Content Manager com ListSchemaRequest

<ListSchemaRequest> ajuda a enumerar todas as entidades onde são necessários esquemas.

Para enumerar todas as entidades do DB2 Content Manager que necessitam de esquemas, crie um <ListSchemaRequest > que identifique as informações seguintes (os valores de atributos entre parênteses rectos são facultativos):

```
<ListSchemaRequest>
 <AuthenticationData> ... </AuthenticationData>
 <EntityList [all="boolean"]>
 [<Entity [name="string" />]
 </EntityList>
</ListSchemaRequest>
```

### Elementos <ListSchemaRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <EntityList> (necessário)

Especifica quais as entidades (que necessitam de esquemas) a devolver. Pode definir o atributo all como TRUE para devolver todas as entidades que necessitam de esquemas ou pode especificar os objectos <Entity [ name="string" /> exactos a devolver.

### Exemplo: pedido XML

A seguinte cadeia de exemplo enumera todas as entidade num servidor que necessitam de esquema e ainda devolveriam um anexo do esquema completo:

```
<ListSchemaRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <EntityList all="true"></EntityList>
</ListSchemaRequest>
```

Como resultado, o DB2 Content Manager devolve um ListSchemaReply XML que devolve uma EntityList de entidades e devolve o esquema para cada entidade na resposta como um anexo.

### Exemplo; resposta XML

```
<ListSchemaReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <EntityList>
 <Entity name="NOINDEX"/><Entity name="ICMBASE"/>
 <Entity name="ICMANNOTATION"/><Entity name="ICMNOTELOG"/>
 <Entity name="ICMSAVEDSEARCH"/><Entity name="ICMFORMS"/>
 <Entity name="ICMDRFOLDERS"/><Entity name="CLAIM_1047"/>
 <Entity name="ICMBASETEXT"/><Entity name="ICMBASESTREAM"/>
 <Entity name="INSURED_1047"/><Entity name="AGENT_1047"/>
 </EntityList>
```

```

<Entity name="CLAIM2_1047"/><Entity name="DOC26"/>
<Entity name="XYZ_ClaimFolder"/><Entity name="XYZ_AdjReport"/>
<Entity name="XYZ_AutoPhoto"/><Entity name="XYZ_ClaimForm"/>
<Entity name="XYZ_InsPolicy"/><Entity name="XYZ_PolReport"/>
</EntityList>
</ListSchemaReply>

```

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Criar artigos do DB2 Content Manager com CreateItemRequest

<CreateItemRequest> ajuda a criar artigos.

Para criar artigos no DB2 Content Manager, crie um <CreateItemRequest > que identifique as seguintes informações de esquema XML (o texto entre parênteses rectos é facultativo):

```

<CreateItemRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item><ItemXML> ... </ItemXML></Item>
</CreateItemRequest>

```

### Elementos <CreateItemRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <Item> (necessário)

Especifica um artigo para criar no servidor do DB2 Content Manager.

O seguinte pedido de exemplo cria uma apólice com uma imagem TIFF anexada e um número de apólice 57904965371.

### Exemplo: pedido XML

```

<CreateItemRequest
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <Item><ItemXML><XYZ_InsPolicy XYZ_Street="532 Camino Viejo"
 XYZ_City="Marina" XYZ_State="CA" XYZ_ZIPCode="90546" XYZ_PolicyNum=
 "57904965371" xmlns=""><XYZ_Insured XYZ_InsrdfName="Edward"
 XYZ_InsrdfLName="Smith" /><XYZ_Insured XYZ_InsrdfName="Jennifer"
 XYZ_InsrdfLName="Smith" /><XYZ_VIN XYZ_VIN="ICLA44P5KL9876543" />
 <ICMBASE><resourceObject MIMEType="image/tiff"
 xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"><label
 name="policyForm" /></resourceObject></ICMBASE></XYZ_InsPolicy>
 </ItemXML></Item>
</CreateItemRequest>

```

Como resultado, o DB2 Content Manager devolve uma <CreateItemReply > em XML que contém o URI da nova apólice. Pode introduzir este URI num navegador da Web para visualizar a estrutura XML do artigo.

**Restrição:** Isto não funciona directamente com os beans XML, dado que o URI é só o PID.

## Exemplo: resposta XML

```
<CreateItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
3 ICM8 icmn1sdb13 XYZ_InsPolicy59 26 A1001001A04I10B33015B9925018
A04I10B33015B992501 14 1029&server=icmn1sdb&dsType=ICM"/>
</CreateItemReply>
```

O seguinte pedido de exemplo cria uma participação (número de participação 8-123456) com uma imagem TIFF e uma imagem JPEG anexadas. Para que este pedido seja efectuado com êxito, terão de existir dois anexos, em que os content-id para os anexos serão claimForm e claimPhoto, que corresponde ao valor dos atributos name dos elementos label no pedido abaixo. Se os anexos do pedido contiverem qualquer outro valor para o content-id, ocorre uma falha no pedido, uma vez que não é possível localizar os anexos correspondentes.

## Exemplo: pedido XML

```
<CreateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
<ServerDef><ServerType>ICM</ServerType>
<ServerName>icmn1sdb</ServerName></ServerDef><LoginData>
<UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item><ItemXML><XYZ_ClaimForm XYZ_ClaimFName="Joannifer"
XYZ_ClaimLName="Smith" XYZ_ClaimNumber="8-123456" XYZ_DriversLic=
"B12004960" XYZ_InsrdFName="Nicholas" XYZ_InsrdLName=
"Smith" XYZ_PolicyNum="57904965371" XYZ_IncDate="2001-01-20">
<ICMBASE><resourceObject MIMEType="image/tiff"
xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"><label
name="claimForm" /></resourceObject></ICMBASE>
<ICMBASE><resourceObject MIMEType="image/jpeg"
xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"><label
name="claimPhoto" /></resourceObject></ICMBASE>
</XYZ_ClaimForm>
</ItemXML></Item>
</CreateItemRequest>
```

Como resultado, o DB2 Content Manager devolve uma <CreateItemReply > em XML que contém o URI da nova participação. Pode introduzir este URI num navegador da Web para visualizar a estrutura XML do artigo.

**Restrição:** Isto não funciona directamente com os beans XML, dado que o URI é só o PID.

## Exemplo: resposta XML

```
<CreateItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
3 ICM8 icmn1sdb13 XYZ_ClaimForm59 26 A1001001A04I10B33016C1746518
A04I10B33016C174651 14 1027&server=icmn1sdb&dsType=ICM"/>
</CreateItemReply>
```

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança



Trabalhar com serviços XML (apenas para Java)

## Pesquisar artigos do DB2 Content Manager com RunQueryRequest

<RunQueryRequest> pesquisa artigos específicos.

Para pesquisar artigos do DB2 Content Manager específicos e obter ligações à Web para estes, crie um <RunQueryRequest > que identifique as seguintes informações de esquema XML (o texto entre parênteses rectos é facultativo):

```
<RunQueryRequest xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema"
[retrieveOption="IDONLY" contentOption="URL" maxResults="integer"
version="LATEST_VERSION"]>
 <AuthenticationData> ... </AuthenticationData>
 <QueryCriteria [queryType="XPath"]>
 <QueryString>string</QueryString>
 </QueryCriteria>
</RunQueryRequest>
```

### Elementos <RunQueryRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <QueryCriteria> (necessário)

Especifica parâmetros de pesquisa para o artigo ou artigos que pretende obter na sintaxe da consulta XPATH (predefinição). Por exemplo, <QueryString>/XYZ\_InsPolicy[ @XYZ\_PolicyNum="47809425673"]</QueryString>.

### Atributos <RunQueryRequest>

#### retrieveOption (facultativo)

Limita as informações nos resultados da pesquisa para melhorar o tempo de resposta. Tenha em atenção que quanto maior for o conteúdo do pedido, mais devagar é efectuada a pesquisa. Pode ter um dos seguintes valores:

##### IDONLY

Devolve apenas os IDs de artigos e favorece um rápido rendimento. É esta a predefinição. Por exemplo, um IDONLY QueryString de /XYZ\_InsPolicy[ @XYZ\_PolicyNum="47809425673"] pesquisará todas as apólices de seguros XYZ com o número de apólice 47809425673.

##### ATTRONLY

Devolve os IDs de artigos e todos os valores de atributos associados.

##### ITEMTREE

Devolve toda a árvore de informações, incluindo IDs dos artigos, valores de atributos, descendentes, subdescendentes e metadados. ITEM TREE não obtém ligações.

##### CONTENT

Devolve todas as informações de ITEM TREE e todo o conteúdo. O conteúdo é devolvido como um URL ou como um anexo, dependendo do que foi especificado no atributo contentOption. Esta situação resulta em rendimento mais lento.

#### contentOption (facultativo)

Especifica se devolve o conteúdo do artigo como uma ligação a um

endereço da Web ou como um anexo binário. Só se aplica quando a `retrieveOption` está definida para `CONTENT`. Pode ter um dos seguintes valores:

#### URL

Solicita que o endereço da Web para o conteúdo do artigo (no gestor de recursos) seja incorporado na descrição XML do artigo. É esta a predefinição.

#### ATTACHMENTS

Solicita que o conteúdo do artigo seja devolvido como um anexo binário na resposta (equivalente ao bean de Java `CMBXMLAttachment`).

#### **maxResults (facultativo)**

Limita o número de resultados da pesquisa a devolver. O valor predefinido é ilimitado (0).

#### **versão (facultativo)**

Especifica a versão dos artigos a pesquisar. Por exemplo, `ALL_VERSIONS` (todas as versões do artigo) ou uma versão específica como 1 ou 2. O valor predefinido é `LATEST_VERSION` (a versão mais recente do artigo).

O exemplo de consulta seguinte pede uma lista de todas as apólices:

### **Exemplo: pedido XML**

```
<RunQueryRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema"
maxResults="0" version="latest-version(.)" contentOption="URL"
retrieveOption="ITEMTREE">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<QueryCriteria>
 <QueryString>XYZ_InsPolicy[@XYZ_State LIKE "/QueryString>
</QueryCriteria>
</RunQueryRequest>
```

Como resultado, o DB2 Content Manager devolve um `<RunQueryReply > XML` que contém uma lista dos endereços da Web para todas as apólices de seguros XYZ. Pode introduzir o URI num navegador da Web para visualizar a estrutura XML de um artigo.

**Restrição:** Isto não funciona directamente com os beans XML, dado que o URI é só o PID.

### **Exemplo: resposta XML**

```
<RunQueryReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<ResultSet count="1">
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26 A1001001A04I10B33609D5857118
A04I10B33609D585711 14 1029&server=icmnlbdb&dsType=ICM">
<ItemXML><XYZ_InsPolicy XYZ_City="Marina" XYZ_PolicyNum="57904965371"
XYZ_State="CA" XYZ_Street="532 Camino Viejo" XYZ_ZIPCode="90546"
cm:PID="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B33609D5857118 A04I10B33609D585711 14 1029"
xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema">
<cm:properties type="document">
```

```

<cm:lastChangeUserId value="ICMADMIN"/>
<cm:lastChangeTime value="2004-09-10T20:36:09.462"/>
<cm:createUserId value="ICMADMIN"/>
<cm:createTime value="2004-09-10T20:36:09.462"/>
<cm:semanticType value="1"/>
<cm:ACL name="XYZInsurancePolicyACL"/>
<cm:lastOperation name="RETRIEVE" value="SUCCESS"/>
<cm:lastRetrieveOption value="32"/></cm:properties>
<XYZ_Insured XYZ_InsrdFName="Edward" XYZ_InsrdLName="Smith"
cm:PID="91 3 ICM8 icmnlsdb11 XYZ_Insured59 26
A1001001A04I10B33609D5857118 A04I10B33609E627501 14 1030"/>
<XYZ_Insured XYZ_InsrdFName="Jennifer" XYZ_InsrdLName="Smith"
cm:PID="91 3 ICM8 icmnlsdb11 XYZ_Insured59 26
A1001001A04I10B33609D5857118 A04I10B33609E637451 14 1030"/>
<XYZ_VIN XYZ_VIN="ICLA44P5KL9876543" cm:PID="86 3 ICM8 icmnlsdb7
XYZ_VIN59 26 A1001001A04I10B33609D5857118 A04I10B33609E646691 14
1031"/></XYZ_InsPolicy>
</ItemXML></Item></ResultSet>
</RunQueryReply>

```

Pode, então, programar a aplicação personalizada para apresentar estas ligações à Web.

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Obter artigos do DB2 Content Manager com RetrieveItemRequest

<RetrieveItemRequest> procura e obtém artigos como ligações à Web ou anexos.

Para pesquisar artigos do DB2 Content Manager específicos e obtê-los como ligações à Web ou anexos binários, crie um elemento <RetrieveItemRequest> que identifica as seguintes informações de esquema XML (o texto entre parênteses rectos é opcional):

```

<RetrieveItemRequest retrieveOption="string"
[contentOption="URL" version="LATEST_VERSION"
checkout="false">]>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI="string" />
 <Item URI="string" />
</RetrieveItemRequest>

```

### Elementos <RetrieveItemRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <Item> (necessário)

Especifica um artigo a obter no servidor do DB2 Content Manager. Cada elemento do artigo especifica um artigo a obter no servidor do DB2 Content Manager. O elemento do artigo terá de conter o seguinte atributo:

##### URI (necessário)

Especifica o identificador único do artigo a obter. Este é o mesmo do que o PID do artigo, conforme armazenado no DB2 Content Manager.

## Atributos <RetrieveItemRequest>

### **retrieveOption (necessário)**

Limita as informações nos resultados da pesquisa para melhorar o tempo de resposta. Pode ter um dos seguintes valores:

#### **IDONLY**

Verifica a existência dos artigos no pedido, mas não obtém os dados.

#### **ATTRONLY**

Devolve os artigos preenchidos apenas com o primeiro nível dos valores do atributo.

#### **ITEMTREE**

Devolve toda a árvore de informações, incluindo IDs dos artigos, valores de atributos, descendentes, subdescendentes e metadados. ITEM TREE não obtém ligações.

#### **CONTENT**

Devolve todas as informações de ITEM TREE e todo o conteúdo. O conteúdo é devolvido como um URL ou como um anexo, dependendo do que foi especificado no atributo contentOption.

#### **CONTENT\_WITH\_LINKS**

Devolve todas as informações de ITEM TREE, todo o conteúdo e todas as informações de ligações que entram e saem do artigo. As informações de ligações aparecem na expressão XML do artigo. Esta situação resulta em rendimento mais lento.

### **contentOption (facultativo)**

Especifica se devolve o conteúdo do artigo como uma ligação a um endereço da Web ou como um anexo binário. Só se aplica quando a retrieveOption está definida para CONTENT ou CONTENT\_WITH\_LINKS. Pode ter um dos seguintes valores:

#### **URL**

Solicita que o endereço da Web para o conteúdo do artigo (no gestor de recursos) seja incorporado na descrição XML do artigo. É esta a predefinição.

#### **ATTACHMENTS**

Solicita que o conteúdo do artigo seja devolvido como um anexo binário na resposta (equivalente ao Javabeans CMBXMLAttachment).

### **checkout (facultativo)**

Especifica se vai ser dada saída do artigo depois de obtido. O valor predefinido é false.

Se o artigo for obtido com êxito, mas ocorrer uma falha ao ser dada saída do artigo, a propriedade errorOnRepeatedCheckoutDuringRetrieve no ficheiro de configuração IBMCMROOT\cmgmt\cmbxmlservices.properties determina se a falha na operação de saída afecta o estado do pedido da mensagem RetrieveItemReply.

Se a propriedade errorOnRepeatedCheckoutDuringRetrieve for definida como true, a mensagem de resposta RetrieveItem é assinalada como falhada. Se a propriedade errorOnRepeatedCheckoutDuringRetrieve estiver definida como false, o erro durante as operações de saída é ignorado. A mensagem de resposta RetrieveItem continua a ser assinalada como bem sucedida.



O seguinte exemplo de consulta obtém a apólice 57904965371 e a respectiva localização do URL da parte de conteúdos, especificando o respectivo URI do artigo (tal como devolvido da pesquisa de consulta):

### Exemplo: pedido XML

```
<?xml version="1.0" encoding="UTF-8"?>
<RetrieveItemRequest contentOption="URL" retrieveOption="CONTENT"
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData configString=""
 connectString="SCHEMA=ICMADMIN">
 <ServerDef>
<ServerType>ICM</ServerType>
<ServerName>icmnlsdb</ServerName>
 </ServerDef>
 <LoginData>
 <UserID>icmadmin</UserID>
 <Password>passw0rd</Password>
 </LoginData>
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=91
 3 ICM6 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A07C06B71945J4200518
 A07C06B71945J420051 14 1030<server=icmnlsdb<dsType=ICM" />
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=92
 3 ICM6 icmnlsdb13 XYZ_InsPolicy60 26 A1001001A07C01B20133C2043418
 A07C01B20133C204342 124 1030<server=icmnlsdb<dsType=ICM" />
</RetrieveItemRequest>
```

Como resultado, o DB2 Content Manager devolve um elemento `<RetrieveItemReply>` XML que contém a apólice 57904965371 e um URL para a respectiva parte de conteúdos:

### Exemplo: resposta XML

```
<?xml version="1.0" encoding="UTF-8"?>
<RetrieveItemReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=91
 3 ICM6 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A07C06B71945J4200518
 A07C06B71945J420051 14 1030&server=icmnlsdb&dsType=ICM">
 <ItemXML>
 <XYZ_InsPolicy XYZ_City="Marina" XYZ_PolicyNum="57904965371"
 XYZ_State="CA"
 XYZ_Street="532 Camino Viejo"
 XYZ_ZIPCode="90546"
 cm:PID="91 3 ICM6 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A07C06B71945J4200518
 A07C06B71945J420051 14 1030"
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema"
 xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <cm:properties type="document">
 <cm:lastChangeUserid value="ICMADMIN" />
 <cm:lastChangeTime value="2007-03-07T01:19:58.593" />
 <cm:createUserid value="ICMADMIN" />
 <cm:createTime value="2007-03-07T01:19:58.593" />
 <cm:semanticType value="1" />
 <cm:lastOperation name="RETRIEVE" value="SUCCESS" />
 </cm:properties>
 <XYZ_Insured XYZ_InsrdfName="Edward"
 XYZ_InsrdfName="Smith"
 cm:PID="89 3 ICM6 icmnlsdb11 XYZ_Insured59
 26 A1001001A07C06B71945J4200518 A07C06B71959C414651 14 1031" />
 <XYZ_Insured XYZ_InsrdfName="Jennifer"
 XYZ_InsrdfName="Smith"
 cm:PID="89 3 ICM6 icmnlsdb11 XYZ_Insured59
```



```

 26 A1001001A07C06B71945J4200518 A07C06B71959C707841 14 1031" />
<XYZ_VIN XYZ_VIN="ICLA44P5KL9876543"
 cm:PID="84 3 ICM6 icmnlsdb7 XYZ_VIN59
 26 A1001001A07C06B71945J4200518 A07C06B71959D546121 14 1032" />
<ICMBASE
 cm:PID="83 3 ICM6 icmnlsdb7 ICMBASE58
 26 A1001001A07C06B71946A8286318 A07C06B71946A828631 13 300"
cm:partNumber="1">
 <cm:properties type="item"
 xsi:type="cm:partPropertyType">
 <cm:lastChangeUserid value="ICMADMIN" />
 <cm:lastChangeTime value="2007-03-07T01:19:58.593" />
 <cm:createUserid value="ICMADMIN" />
 <cm:createTime value="2007-03-07T01:19:58.593" />
 <cm:semanticType value="128" />
 <cm:lastOperation name="RETRIEVE" value="SUCCESS" />
 </cm:properties>
 <cm:resourceObject MIMEType="text/plain"
 RMName="MRMDB" SMSCollName="CBR.CLLCT001"
 externalObjectName=" " resourceFlag="2"
 size="840">
 <cm:URL value=
 "http://hostname/icmr/ICMResourceManager?order=retrieve&
 item-id=A1001001A07C06B71946A82863&version=1&
 collection=CBR.CLLCT001&libname=icmnlsdb&
 update-date=2007-03-07+01%3A19%3A58.284794&
 token=A4E6.FT6Fo_6_k_LVHc2VHR2;&content-length=0" />
 </cm:resourceObject>
 </ICMBASE>
</XYZ_InsPolicy>
</ItemXML>
</Item>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=92
 3 ICM6 icmnlsdb13 XYZ_InsPolicy60 26 A1001001A07C01B20133C2043418
 A07C01B20133C204342 124 1030&server=icmnlsdb&dsType=ICM">
 <ItemXML>
 <XYZ_InsPolicy XYZ_City="Temple City"
 XYZ_PolicyNum="57904965371" XYZ_State="CA"
 XYZ_Street="11132 Industry Blvd"
 XYZ_ZIPCode="90543"
 cm:PID="92 3 ICM6 icmnlsdb13 XYZ_InsPolicy60
 26 A1001001A07C01B20133C2043418 A07C01B20133C204342 124 1030"
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema"
xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <cm:properties type="document">
 <cm:lastChangeUserid value="ICMADMIN" />
 <cm:lastChangeTime value="2007-03-06T18:49:42.752" />
 <cm:createUserid value="ICMADMIN" />
 <cm:createTime value="2007-03-01T20:01:33.425" />
 <cm:semanticType value="1" />
 <cm:lastOperation name="RETRIEVE" value="SUCCESS" />
 </cm:properties>
 <XYZ_Insured XYZ_InsrdFName="Edward"
 XYZ_InsrdLName="Smith"
 cm:PID="90 3 ICM6 icmnlsdb11 XYZ_Insured60
 26 A1001001A07C01B20133C2043418 A07C01B20133E295422 124 1031" />
 <XYZ_Insured XYZ_InsrdFName="Alice"
 XYZ_InsrdLName="Smith"
 cm:PID="90 3 ICM6 icmnlsdb11 XYZ_Insured60
 26 A1001001A07C01B20133C2043418 A07C06B04942H524692 124 1031" />
 <XYZ_VIN XYZ_VIN="1RI035P5RU5435209"
 cm:PID="85 3 ICM6 icmnlsdb7 XYZ_VIN60
 26 A1001001A07C01B20133C2043418 A07C01B20133E218832 124 1032" />
 <ICMBASE
 cm:PID="83 3 ICM6 icmnlsdb7 ICMBASE58
 26 A1001001A07C01B20133C2122218 A07C01B20133C212221 13 300"

```

```

cm:partNumber="1">
 <cm:properties type="item"
 xsi:type="cm:partPropertyType">
 <cm:lastChangeUserid value="ICMADMIN" />
 <cm:lastChangeTime value="2007-03-01T20:01:33.425" />
 <cm:createUserid value="ICMADMIN" />
 <cm:createTime value="2007-03-01T20:01:33.425" />
 <cm:semanticType value="128" />
 <cm:lastOperation name="RETRIEVE" value="SUCCESS" />
 </cm:properties>
 <cm:resourceObject MIMEType="text/plain"
 RMName="MRMDB" SMSCollName="CBR.CLLCT001"
 externalObjectName=" " resourceFlag="2" size="776">
 <cm:URL value=
 "http://hostname/icrm/ICMResourceManager?order=retrieve&
 item-id=A1001001A07C01B20133C21222&version=1&
 collection=CBR.CLLCT001&libname=icmnlbdb&
 update-date=2007-03-01+20%3A01%3A33.650668&
 token=A4E6.FT6Fo_6_WbFRyL48RAw;&content-length=0" />
 </cm:resourceObject>
 </ICMBASE>
 </XYZ_InsPolicy>
 </ItemXML>
</Item>
</RetrieveItemReply>

```

O seguinte exemplo de consulta obtém a participação 8-123456 com a parte de conteúdos binários anexada, especificando o respectivo URI do artigo (tal como devolvido da pesquisa de consulta):

### Exemplo: pedido XML

```

<RetrieveItemRequest contentOption="ATTACHMENTS"
 retrieveOption="CONTENT"
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93 3
 ICM8 icmnlbdb13 XYZ_ClaimForm59 26 A1001001A04I10B33016C1746518
 A04I10B33016C174651 14 1027&server=icmnlbdb&dsType=ICM"/>
</RetrieveItemRequest>

```

Como resultado, o DB2 Content Manager devolve um elemento <RetrieveItemReply> XML que contém a participação 8-123456 e os anexos binários para as respectivas partes de conteúdos:

### Exemplo: resposta XML

```

<RetrieveItemReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93 3
 ICM8 icmnlbdb13 XYZ_ClaimForm59 26 A1001001A04I10B33016C1746518
 A04I10B33016C174651 14 1027&server=icmnlbdb&dsType=ICM">
 <ItemXML><XYZ_ClaimForm XYZ_ClaimFName="Joannifer"
 XYZ_ClaimLName="Smith" XYZ_ClaimNumber="8-123456"
 XYZ_DriversLic="B12004960" XYZ_IncDate="2001-01-20"
 XYZ_InsrdfName="Nicholas" XYZ_InsrdfLName="Smith"
 XYZ_PolicyNum="57904965371" cm:PID="93 3 ICM8 icmnlbdb13
 XYZ_ClaimForm59 26 A1001001A04I10B33016C1746518
 A04I10B33016C174651 14 1027" xmlns:cm=
 "http://www.ibm.com/xmlns/db2/cm/api/1.0/schema" xmlns:xsi=
 "http://www.w3.org/2001/XMLSchema-instance">

```

```

<cm:properties type="document">
...
</cm:properties>
<ICMBASE cm:PID="85 3 ICM8 icmnlsdb7 ICMBASE58 26
A1001001A04I10B33016C1816218 A04I10B33016C181621 13 300"
cm:partNumber="1">
<cm:properties type="item" xsi:type="cm:partPropertyType">
...
</cm:properties>
<cm:resourceObject MIMEType="image/tiff" RMName="rmdb"
SMSCollName="CBR.CLLCT001" externalObjectName=" "
resourceFlag="2" size="61578"> <cm:label name=
"A1001001A04I10B33016C18162A04I10B33016C181621"/>
</cm:resourceObject>
</ICMBASE>
<ICMBASE cm:PID="85 3 ICM8 icmnlsdb7 ICMBASE58 26
A1001001A04I10B33016C1887218 A04I10B33016C188721 13 300"
cm:partNumber="2">
<cm:properties type="item" xsi:type="cm:partPropertyType">
...
</cm:properties>
<cm:resourceObject MIMEType="image/jpeg" RMName="rmdb"
SMSCollName="CBR.CLLCT001" externalObjectName=" "
resourceFlag="6" size="0"/>
</ICMBASE></XYZ_ClaimForm>
</ItemXML></Item>
</RetrieveItemReply>

```

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Visualizar os privilégios do utilizador nos pedidos XML

<GetPrivilegesRequest> ajuda a visualizar privilégios num artigo específico.

Para visualizar os privilégios num determinado artigo do DB2 Content Manager (incluindo privilégios do utilizador), crie um <GetPrivilegesRequest > que identifique as informações de esquema seguintes:

```

<GetPrivilegesRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI=string />
</GetPrivilegesRequest>

```

### Elementos <GetPrivilegesRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <Item URI="string"/> (necessário)

Especifica o identificador persistente para o artigo de que pretende enumerar privilégios. Pode obter o URI através de <RunQueryRequest>. Também aparece em CreateItemReply.

### Exemplo: pedido XML

O seguinte pedido de exemplo enumera todos os privilégios para a apólice 57904965371, especificando o respectivo URI (tal como devolvido da pesquisa de consulta):

```
<GetPrivilegesRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="90 3 ICM8 icmnlsdb10 CLAIM_104759 26
A1001001A04I13B04803F2085118 A04I13B04803F208511 14 1007"/>
</GetPrivilegesRequest>
```

Como resultado, o DB2 Content Manager devolve um elemento `<GetPrivilegesReply>` XML que devolve uma lista `<Privileges>` de todos os privilégios associados ao artigo. Cada objecto `<Privilege>` contém um nome (por exemplo, `VIEW_CONTENT` ou `MODIFY_CONTENT`) e o respectivo estado de de autorização, ou seja, sim, não ou desconhecido.

### Exemplo: resposta XML

```
<GetPrivilegesReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Privileges>
 <Privilege authorized="YES" name="VIEW_CONTENT"/>
 <Privilege authorized="NO" name="MODIFY_CONTENT"/>
 <Privilege authorized="" name="EDIT_ATTRIBUTES"/>
 ...
</Privileges>
</GetPrivilegesReply>
```

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Trabalhar com pastas do DB2 Content Manager através de pedido XML

Uma *pasta* contém zero ou mais artigos do DB2 Content Manager e zero ou mais pastas.

O objecto utiliza o seguinte esquema (o texto entre parênteses rectos é facultativo):

```
<Folder>
<Item URI=string> [<ItemXML> ... </ItemXML>] </Item>]
<FolderItems>
 [<Item URI=string> [<ItemXML> ... </ItemXML>] </Item>
 <Item URI=string> [<ItemXML>...</ItemXML>]</Item>
 ...
]
</FolderItems>]
</Folder>
```

Pode criar uma pasta utilizando `<CreateItemRequest>` e a definição `<cm:properties type="folder" xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"/>` (consulte o exemplo quase no fim desta secção).

Pode pedir as acções seguintes para pastas no DB2 Content Manager.

## Adicionar um artigo a uma pasta

```
<AddItemToFolderRequest [newVersion=false
checkin=true checkout=true]>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI = o PID do artigo a adicionar />
 <Folder URI = o PID da pasta à qual pretende adicionar o artigo/Folder>
</AddItemToFolderRequest>
```

O <AddItemToFolderRequest> pode utilizar opcionalmente os atributos newVersion, checkout e checkin.

## Remover um artigo de uma pasta

```
<RemoveItemFromFolderRequest[newVersion="FALSE"
checkin="TRUE" checkout="TRUE"]>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI = o PID do artigo a remover/Item>
 <Folder URI = o PID da pasta da qual pretende remover o artigo/Folder>
</RemoveItemFromFolderRequest>
```

O <RemoveItemFromFolderRequest> pode utilizar opcionalmente os atributos newVersion, checkout e checkin.

## Obter todos os artigos de uma pasta

```
<RetrieveFolderItemsRequest [retrieveOption="string"
contentOption="URL" version="LATEST_VERSION"] >
 <AuthenticationData> ... </AuthenticationData>
 <Folder URI = o PID da pasta da qual pretende obter os artigos/Folder>
</RetrieveFolderItemsRequest>
```

O <RetrieveFolderItemsRequest> pode utilizar opcionalmente os atributos retrieveOption e contentOption em conjunto com maxItems para fins de obtenção (a predefinição é NO\_MAX).

Se tiver êxito, o DB2 Content Manager devolve a resposta seguinte:

```
<RetrieveFolderItemsReply>
 <FolderItems>
 <Item> ... </Item>
 ...
 </FolderItems>
</RetrieveFolderItemsReply>
```

## Obter todas as pastas que contêm um artigo

```
<RetrieveFoldersForItemRequest
[retrieveOption="string" contentOption="URL"] >
 <AuthenticationData> ... </AuthenticationData>
 <Item URI = o PID do artigo do qual pretende obter as pastas/Item>
</RetrieveFoldersForItemRequest>
```

O <RetrieveFoldersForItemsRequest> pode utilizar opcionalmente os atributos retrieveOption e contentOption.

Se tiver êxito, o DB2 Content Manager devolve a resposta seguinte:

```
<RetrieveFoldersForItemReply>
 <Folder> ... </Folder>
 ...
</RetrieveFoldersForItemReply>
```

O seguinte exemplo cria uma pasta e coloca a apólice 57904965371 dentro dessa pasta, especificando o URI do artigo (tal como devolvido da pesquisa de consulta).

**Importante:** É necessário especificar categorizando o novo artigo como uma pasta, definindo o atributo de tipo no elemento `cm:properties` para a pasta.

### Exemplo: pedido XML

```
<CreateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item><ItemXML><XYZ_InsPolicy XYZ_Street="532 Camino Viejo"
 XYZ_City="Marina" XYZ_State="CA" XYZ_ZIPCode="90546"
 XYZ_PolicyNum="57904965371" xmlns="">
 <cm:properties type="folder" xmlns:cm=
 "http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"/>
 </XYZ_InsPolicy>
</ItemXML></Item>
</CreateItemRequest>
```

Como resultado, o DB2 Content Manager devolve uma `<CreateItemReply >` XML que contém o URI da nova pasta:

### Exemplo: resposta XML

```
<CreateItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B45558I8302118
 A04I10B45558I830211 14 1029&server=icmnlsdb&dsType=ICM"/>
</CreateItemReply>
```

O exemplo seguinte adiciona a participação 8-123456 à mesma pasta, especificando os respectivos URIs:

### Exemplo: pedido XML

```
<AddItemToFolderRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Folder URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B45558I8302118
 A04I10B45558I830211 14 1029&server=icmnlsdb&dsType=ICM"/>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A04I10B45555F4042518
 A04I10B45555F404251 14 1027&server=icmnlsdb&dsType=ICM"/>
</AddItemToFolderRequest>
```

Como resultado, o DB2 Content Manager devolve um `<AddItemToFolderReply >` XML que indica o sucesso:

### Exemplo: resposta XML

```
<AddItemToFolderReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</AddItemToFolderReply>
```

O exemplo seguinte elimina a pasta, especificando o respectivo URI:

## Exemplo: pedido XML

```
<DeleteItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B45558I8302118
A04I10B45558I830211 14 1029&server=icmnlsdb&dsType=ICM" />
</DeleteItemRequest>
```

Como resultado, o DB2 Content Manager devolve um <DeleteItemReply > XML que indica o sucesso:

## Exemplo: resposta XML

```
<DeleteItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</DeleteItemReply>
```

### Informações relacionadas

 Actualizar artigos do DB2 Content Manager com um XML  
UpdateItemRequest

 Obter artigos do DB2 Content Manager com RetrieveItemRequest

## Actualizar artigos do DB2 Content Manager com o UpdateItemRequest de XML

<UpdateItemRequest> actualiza artigos.

Para actualizar artigos do DB2 Content Manager, crie um <UpdateItemRequest > que identifique as informações de esquema XML seguintes:

```
<UpdateItemRequest [newVersion="false"
checkin="true" checkout="true"]>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI="PID or URL"/>
 <Add>
 <ChildItem parentURI="PID or URL">...</ChildItem>
 <Content [index="integer"]>...</Content>
 <Notelog [index="integer"]>...</Notelog>
 <Annotation [index="integer"]>...</Annotation>
 <Part attrName="String">...</Part>
 </Add>
 <Replace>
 <ItemXML>...</ItemXML>
 <Attribute name="String"> </Attribute>
 <ChildItem parentURI="PID or URL">...</ChildItem>
 <Content [index="integer"]>...</Content>
 <Notelog [index="integer"]>...</Notelog>
 <Annotation [index="integer"]>...</Annotation>
 <Part attrName="String">...</Part>
 </Replace>
 <Delete>
 <ChildItem parentURI="PID or URL">...</ChildItem>
 <Content [index="integer"]>...</Content>
 <Notelog [index="integer"]>...</Notelog>
```



```

<Annotation [index="integer"]>...</Annotation>
<Part attrName="String">...</Part>
</Delete>
</UpdateItemRequest>

```

## Elementos <UpdateItemRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (ServerDef), um ID de utilizador (UserID) e uma palavra-passe (Password) - ou uma credencial SSO do WebSphere.

### <Item URI="PID or URL"> (necessário)

Especifica o artigo a actualizar no servidor do DB2 Content Manager.

### <Add> (facultativo)

Especifica todo o conteúdo a adicionar ao artigo neste pedido de actualização. É possível adicionar artigos descendentes, partes de conteúdos, registos de notas, anotações ou valores de partes para atributos.

### <Replace> (facultativo)

Especifica todo o conteúdo a substituir no artigo neste pedido de actualização. É possível substituir a árvore de artigos completa, que descreve os atributos e os componentes descendentes do artigo, utilizando o elemento ItemXML. Também é possível substituir individualmente artigos descendentes, partes de conteúdos, registos de notas, anotações ou valores de partes para atributos.

### <Delete> (facultativo)

Especifica todo o conteúdo a eliminar do artigo neste pedido de actualização. É possível eliminar artigos descendentes, partes de conteúdos, registos de notas, anotações ou valores de partes para atributos.

## Atributos <UpdateItemRequest>

### newVersion (facultativo)

Especifica se deve incrementar o número de versão no artigo em actualização. O valor predefinido é FALSE.

### checkout (facultativo)

Comuta para dar saída do artigo (e, assim, bloqueá-lo) ante de efectuar o respectivo pedido de actualização. O valor predefinido é TRUE.

### checkin (facultativo)

Comuta para dar entrada do artigo depois de efectuar o respectivo pedido de actualização. O valor predefinido é TRUE. Se não for dada saída do artigo esta definição falha.

O seguinte exemplo coloca em lotes as três actualizações para actualizar a apólice 57904965371:

## Exemplos: pedido XML

```

<UpdateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029" />
<Add>

```



```

<ChildItem parentURI="93 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26
 A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029">
 <ItemXML><XYZ_Insured XYZ_InsrdFName="Alice"
 XYZ_InsrdLName="Smith" /></ItemXML></ChildItem>
</Add>
<Replace>
 <Attribute name="XYZ_Street"><Value>123 Cedar Rd</Value></Attribute>
 <Attribute name="XYZ_ZIPCode"><Value>90543</Value></Attribute>
 <ChildItem parentURI="93 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26
 A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029">
 <ItemXML><XYZ_VIN
 xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 XYZ_VIN="1RI035P5RU5435209" cm:PID="86 3 ICM8 icmnlsdb7
 XYZ_VIN59 26 A1001001A04I10B35710G9498818 A04I10B35710J581901
 14 1031" /></ItemXML>
 </ChildItem>
 <Content PID="85 3 ICM8 icmnlsdb7 ICMBASE58 26
 A1001001A04I10B35710G9582118 A04I10B35710G958211 13 300">
 <Attachment content-id="policyForm"/>
 </Content>
</Replace>
<Delete>
 <ChildItem URI="91 3 ICM8 icmnlsdb11 XYZ_Insured59 26
 A1001001A04I10B35710G9498818 A04I10B35710J374681 14 1030" />
</Delete>
</UpdateItemRequest>

```

Como resultado, o DB2 Content Manager devolve uma <UpdateItemReply > XML que contém o URI da apólice actualizada:

### Exemplo: resposta XML

```

<UpdateItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=84
 3 ICM8 icmnlsdb5 DOC2659 26 A1001001A04I02B22524G4418418
 A04I02B22524G441841 14 1019&server=icmnlsdb&dsType=ICM"/>
</UpdateItemReply>

```

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

### Adicionar objectos dentro de artigos do DB2 Content Manager com um UpdateItemRequest XML

Utilizando o elemento <Add > num <UpdateItemRequest >, pode adicionar os tipos seguintes de valores de esquema XML aos artigos do DB2 Content Manager existentes (o texto entre parênteses rectos é facultativo):

```

<Add>
[<ChildItem parentURI="string"><ItemXML> ... </ItemXML></ChildItem>]
[<Content [index="int"] >
 <attachment content-id="string">...</attachment>
 <!-- or --> <URL>...</URL>
</Content>]
[<Annotation [index="string"] >
 <attachment content-id="string">...</attachment>
 <!-- or --> <URL>...</URL>
</Annotation>]
[<Notelog [index="int"] >
 <attachment content-id="string">...</attachment>
 <!-- or --> <URL>...</URL>
</Notelog>]
[<Part attrName="string">

```

```

 <attachment content-id="string">...</attachment>
 <!-- or --> <URL>...</URL>
 </Part>]
</Add>

```

#### **<ChildItem> (facultativo)**

Adicionar um artigo descendente existente a uma recolha de componentes descendentes existente com o que especificar em <ItemXML>. Os atributos incluem:

##### **parentURI (necessário)**

O identificador persistente do artigo ascendente ao qual adiciona o artigo descendente. Pode obter o URI através de <RunQueryRequest>. Também aparece em <CreateItemReply>.

#### **<Content> (facultativo)**

Adiciona um anexo ou URL como uma parte de conteúdos para o artigo. É um procedimento igual ao da parte de base ICM. Os atributos incluem:

##### **index (facultativo)**

Classifica onde adicionar a parte. Por predefinição adiciona a parte no fim.

#### **<Notelog> (facultativo)**

Adiciona um registo de notas a um artigo e pode incluir um anexo ou um URL. Os atributos incluem:

##### **index (facultativo)**

Classifica onde adicionar a anotação. Por predefinição adiciona o registo de notas no fim.

#### **<Annotation> (facultativo)**

Adiciona uma anotação a um artigo e pode incluir um anexo ou um URL. Os atributos incluem:

##### **index (facultativo)**

Classifica onde adicionar a anotação. Por predefinição adiciona a anotação no fim.

#### **<Part> (facultativo)**

Adiciona a parte definida pelo utilizador com o nome de atributo especificado. Pode incluir um anexo ou um URL. Os atributos incluem:

##### **attrName (necessário)**

Especifica o nome do atributo definido pelo utilizador a adicionar.

O seguinte exemplo adiciona um artigo descendente à apólice 57904965371, especificando os respetivos URIs (tal como devolvido da pesquisa de consulta):

### **Exemplo: pedido XML**

```

<UpdateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029" />
<Add>
 <ChildItem parentURI="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029">

```

```

 <ItemXML><XYZ_Insured XYZ_InsrdFName="Alice"
 XYZ_InsrdLName="Smith" /></ItemXML></ChildItem>
 </Add>
</UpdateItemRequest>

```

Como resultado, o DB2 Content Manager devolve uma <UpdateItemReply > XML que contém o URI da apólice actualizada:

### Exemplo: resposta XML

```

<UpdateItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=84
3 ICM8 icmn1sdb5 DOC2659 26 A1001001A04I02B22524G4418418
A04I02B22524G441841 14 1019&server=icmn1sdb&dsType=ICM"/>
</UpdateItemReply>

```

### Informações relacionadas



Trabalhar com serviços XML (apenas para Java)

## Substituir objectos dentro de artigos do DB2 Content Manager com um UpdateItemRequest XML

Utilizando o elemento <Replace > num <UpdateItemRequest >, pode sobrepor os tipos seguintes de valores de esquema XML em artigos do DB2 Content Manager existentes:

```

<Replace>
[<attribute> ... </attribute>]
[<ChildItem parentURI=string><ItemXML> ... </ItemXML></ChildItem>]
[<Annotation PID=string>]
 <attachment content-id=string>...</attachment>
 <!-- or --> <URL>...</URL>
</Annotation>]
[<Content PID=string>
 <attachment content-id=string>...</attachment>
 <!-- or --> <URL>...</URL>
</Content>]
[<Notelog PID=string>
 <attachment content-id=string>...</attachment>
 <!-- or --> <URL>...</URL>
</Notelog>]
[<Part PID=string attrName=string>
 <attachment content-id=string>...</attachment>
 <!-- or --> <URL>...</URL>
</Part>]
</Replace>

```

### <Attribute> (facultativo)

Substitui valores de atributo.

### <ChildItem> (facultativo)

Substitui uma recolha de artigos descendentes. Os atributos incluem:

#### parentURI (necessário)

É o identificador persistente do artigo ascendente onde está o artigo descendente a substituir. Pode obter o URI através de <RunQueryRequest>. Também aparece em <CreateItemReply>.

### <Annotation> (facultativo)

Substitui uma anotação e pode incluir um anexo ou um URL. Os atributos incluem:

#### PID (necessário)

Especifica o identificador persistente da anotação.

**<Content> (facultativo)**

Substitui uma parte do conteúdo do artigo com o anexo ou o URL que tenha especificado. É um procedimento igual ao da parte de base ICM. Os atributos incluem:

**PID (necessário)**

Especifica o identificador persistente do conteúdo.

**<Notelog> (facultativo)**

Substitui um registo de notas e pode incluir um anexo ou um URL. Os atributos incluem:

**PID (necessário)**

Especifica o identificador persistente do registo de notas.

**<Part> (facultativo)**

Substitui a parte definida pelo utilizador com o nome de atributo, o anexo ou o URL de conteúdo que tenha especificado. Os atributos incluem:

**PID (necessário)**

Especifica o identificador persistente da parte.

**attrName (necessário)**

Especifica o nome do atributo definido pelo utilizador a substituir.

O exemplo seguinte substitui dois atributos, um artigo descendente e uma parte de conteúdos, na apólice 57904965371:

**Exemplo: pedido XML**

```
<UpdateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029" />
<Replace>
 <Attribute name="XYZ_Street"><Value>123 Cedar Rd</Value></Attribute>
 <Attribute name="XYZ_ZIPCode"><Value>90543</Value></Attribute>
 <ChildItem parentURI="93 3 ICM8 icmnlbdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029">
 <ItemXML><XYZ_VIN
 xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
 XYZ_VIN="1RI035P5RU5435209" cm:PID="86 3 ICM8 icmnlbdb7
 XYZ_VIN59 26 A1001001A04I10B35710G9498818 A04I10B35710J581901
 14 1031" /></ItemXML>
 </ChildItem>
 <Content PID="85 3 ICM8 icmnlbdb7 ICMBASE58 26
 A1001001A04I10B35710G9582118 A04I10B35710G958211 13 300">
 <Attachment content-id="policyForm"/>
 </Content>
</Replace>
</UpdateItemRequest>
```

Como resultado, o DB2 Content Manager devolve uma <UpdateItemReply > XML que contém o URI da apólice actualizada:

### Exemplo: resposta XML

```
<UpdateItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=84
3 ICM8 icmnlsdb5 DOC2659 26 A1001001A04I02B22524G4418418
A04I02B22524G441841 14 1019&server=icmnlsdb&dsType=ICM"/>
</UpdateItemReply>
```

### Eliminar objectos dentro de artigos do DB2 Content Manager com um UpdateItemRequest XML

Utilizando o elemento <Delete> num <UpdateItemRequest>, pode eliminar os tipos seguintes de valores de esquema XML dos artigos do DB2 Content Manager existentes (remove apenas as referências deste artigo aos valores de atributo):

```
<Delete>
[<ChildItem URI="string"> ... </ChildItem>]
[<Annotation PID="string">]
[<Content PID="string">]
[<Notelog PID="string">]
[<Part PID="string" attrName="string">]
</Delete>
```

#### <ChildItem> (facultativo)

Remove mas não elimina o componente descendente da recolha.

#### <Annotation> (facultativo)

Elimina a anotação do atributo do PID do artigo especificado.

#### <Content> (facultativo)

Elimina o anexo do artigo do PID especificado. É um procedimento igual ao da parte de base ICM.

#### <Notelog> (facultativo)

Elimina o registo de notas do atributo do PID do artigo especificado.

#### <Part> (facultativo)

Elimina a parte definida pelo utilizador associada ao PID do artigo e ao nome de atributo especificados.

O seguinte exemplo elimina um artigo descendente na apólice 57904965371, especificando os respectivos URIs (tal como devolvido da pesquisa de consulta):

### Exemplo: pedido XML

```
<UpdateItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="93 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26
A1001001A04I10B35710G9498818 A04I10B35710G949881 14 1029" />
<Delete>
 <ChildItem URI="91 3 ICM8 icmnlsdb11 XYZ_Insured59 26
A1001001A04I10B35710G9498818 A04I10B35710J374681 14 1030" />
</Delete>
</UpdateItemRequest>
```

Como resultado, o DB2 Content Manager devolve uma <UpdateItemReply> XML que contém o URI da apólice actualizada:

### Exemplo: resposta XML

```
<UpdateItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=84
 3 ICM8 icmnlsdb5 DOC2659 26 A1001001A04I02B22524G4418418
 A04I02B22524G441841 14 1019&server=icmnlsdb&dsType=ICM"/>
</UpdateItemReply>
```

## Eliminar artigos do DB2 Content Manager com DeleteItemRequest

<DeleteItemRequest> elimina artigos.

Para eliminar artigo do DB2 Content Manager, crie um <DeleteItemRequest > que identifique as informações seguintes (os valores de atributos entre parênteses rectos são facultativos):

```
<DeleteItemRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI=string />
</DeleteItemRequest>
```

### Elementos <DeleteItemRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <Item URI=string /> (necessário)

Especifica o artigo a eliminar do servidor do DB2 Content Manager.

### Exemplo: pedido XML

O seguinte exemplo elimina a apólice 57904965371, especificado o respectivo URI do artigo (tal como devolvido da pesquisa de consulta):

```
<DeleteItemRequest xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectionString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B44211H0825818
 A04I10B44211H082581 14 1029&server=icmnlsdb&dsType=ICM" />
</DeleteItemRequest>
```

Como resultado, o DB2 Content Manager devolve um elemento <DeleteItemReply> XML que indica o sucesso:

### Exemplo: resposta XML

```
<DeleteItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
</DeleteItemReply>
```

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Dar saída e entrada de artigos do DB2 Content Manager com CheckoutItemRequest e CheckinItemRequest

Quando dá saída de artigos do DB2 Content Manager, o servidor bloqueia os artigos com o seu ID de utilizador para que mais nenhum utilizador possa editá-los. Isto evita que os utilizadores sobreponham as alterações uns dos outros. É particularmente útil em pedidos de lotes quando tem de editar determinados artigos durante um longo período de tempo.

Para dar saída de artigos do DB2 Content Manager, crie um `<CheckoutItemRequest>` identifique as informações seguintes (os valores de atributos entre parênteses rectos são facultativos):

```
<CheckoutItemRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI="string" />
</CheckoutItemRequest>
```

### Elementos `<CheckoutItemRequest>`

#### `<AuthenticationData>` (necessário)

Identifica o servidor de conteúdos (`<ServerDef>`), um ID de utilizador válido (`<UserID>`) e uma palavra-passe (`<Password>`) - ou uma credencial SSO do WebSphere.

#### `<Item URI=string />` (necessário)

Especifica o artigo a dar saída do servidor do DB2 Content Manager.

O seguinte exemplo de consulta dá saída da apólice 47809425673, especificando o respectivo URI do artigo:

### Exemplo: pedido XML

```
<CheckoutRequest xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef>
 <ServerType>ICM</ServerType><ServerName>concord</ServerName>
 </ServerDef>
 <LoginData>
 <UserID>icmadmin</UserID><Password>ecllent</Password>
 </LoginData>
 </AuthenticationData>
 <Item URI="http://icmsserver/CMBGenericWebService/CMBGetPIDUrl?pid=92 3
 ICM7 concord13 XYZ InsPolicy59 26 A1001001A04B25B14339H1785918
 A04B25B14339H178591 14 1048&server=concord&dsType=ICM" />
</CheckoutItemRequest>
```

Como resultado, o DB2 Content Manager devolve um `<CheckoutItemReply>` XML que indica o sucesso.

### Exemplo: resposta XML

```
<CheckoutItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
</CheckoutItemReply>
```

Para dar entrada de artigos de volta para o DB2 Content Manager, crie um `<CheckinItemRequest>` que identifique as informações seguintes (os valores de atributos entre parênteses rectos são facultativos):

```
<CheckinItemRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI=string />
</CheckinItemRequest>
```

## Elementos <CheckinItemRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <Item URI="string" />

Especifica o artigo a dar entrada no servidor do DB2 Content Manager.

O seguinte exemplo de consulta dá entrada da apólice 47809425673, especificando o respectivo URI do artigo:

### Exemplo: pedido XML

```
<CheckinRequest xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>concord</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>ec1lent</Password></LoginData>
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=92 3
 ICM7 concord13 XYZ_InsPolicy59 26 A1001001A04B25B14339H1785918
 A04B25B14339H178591 14 1048&server=concord&dsType=ICM" />
</CheckinItemRequest>
```

Como resultado, o DB2 Content Manager devolve um <CheckinItemReply > XML que indica o sucesso.

### Exemplo: resposta XML

```
<CheckinItemReply xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>
</CheckinItemReply>
```

#### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Ligar artigos do DB2 Content Manager com CreateLinkRequest ou DeleteLinkRequest

<CreateLinkRequest> ou <DeleteLinkRequest> cria ou elimina artigos.

Para criar ou eliminar ligações entre artigos no DB2 Content Manager, crie um <CreateLinkRequest > ou um <DeleteLinkRequest > que identifiquem as informações de esquema XML seguintes (o texto entre parênteses rectos é facultativo):

```
<CreateLinkRequest> <!-- or --> <DeleteLinkRequest>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI=string>
 <links>
 <inbound fromItem=URI linkType=string
 [linkInfoItem=string] />
 <!-- or -->
 <outbound toItem=URI linkType=string
 [linkInfoItem=string] />
 </links>
 </Item>
</CreateLinkRequest> <!-- or --> </DeleteLinkRequest>
```



## Elementos <CreateLinkRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <Item> (necessário)

Especifica um artigo para criar no servidor do DB2 Content Manager.

### <links> (necessário)

Especifica as ligações de <inbound> (para ligar de um URI de artigo) ou ligações <outbound> (para ligar a um URI de artigo) a criar.

O seguinte exemplo cria uma ligação de saída de uma participação 8-123456 para a apólice 57904965371, especificando os respectivos URIs (tal como devolvido da pesquisa de consulta):

## Exemplo: pedido XML

```
<CreateLinkRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93 3
 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B44211H0825818
 A04I10B44211H082581 14 1029&server=icmnlsdb&dsType=ICM">
<links xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema">
 <outbound toItem=
 "http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93 3 ICM8
 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A04I10B44213F1658218
 A04I10B44213F165821 14 1027&server=icmnlsdb&dsType=ICM"
 linkType="Contains"/></links>
</Item>
</CreateLinkRequest>
```

Como resultado, o DB2 Content Manager devolve um <CreateLinkReply > XML que indica o sucesso.

## Exemplo: resposta XML

```
<CreateLinkReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</CreateLinkReply>
```

O seguinte exemplo elimina a ligação de saída de uma participação 8-123456 para a apólice 57904965371, especificando os respectivos URIs (tal como devolvido da pesquisa de consulta):

## Exemplo: pedido XML

```
<DeleteLinkRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93 3
 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B44211H0825818
 A04I10B44211H082581 14 1029&server=icmnlsdb&dsType=ICM">
```

```

<links xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema">
<outbound toItem=
 "http://hostname/CMBGenericWebService/CMBGetPIDurl?pid=93 3 ICM8
 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A04I10B44213F1658218
 A04I10B44213F165821 14 1027&server=icmnlsdb&dsType=ICM"
 linkType="Contains"/></links>
</Item>
</DeleteLinkRequest>

```

Como resultado, o DB2 Content Manager devolve um DeleteLinkReply XML que indica o sucesso.

### Exemplo: resposta XML

```

<DeleteLinkReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</DeleteLinkReply>

```

#### Conceitos relacionados

“Trabalhar com serviços XML (apenas para Java)” na página 339

#### Informações relacionadas

-  Autenticar pedidos de serviços da Web para segurança
-  Trabalhar com serviços XML (apenas para Java)

## Mover artigos do DB2 Content Manager entre entidades com o MoveItemRequest

<MoveItemRequest> move um artigo.

Para mover um artigo do DB2 Content Manager de uma entidade para outra (para modificar a árvore de artigos do atributo e valores descendentes; não o conteúdo do documento), crie um <MoveItemRequest> que identifique as seguintes informações (os valores de atributos entre parênteses rectos são facultativos):

```

<MoveItemRequest [checkout="TRUE" checkin="TRUE"]>
 <AuthenticationData> ... </AuthenticationData>
 <Item URI="string" />
 <NewValues newEntityName="string">
 <ItemXML><!-- Do not specify document content here --></ItemXML>
 </NewValues>
</MoveItemRequest>

```

### Elementos <MoveItemRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <Item URI=string> (necessário)

Especifica o artigo a re-indexar.

#### <NewValues newEntityName=string> (necessário)

Especifica o novo valor a definir na entidade. O valor tem de ser uma especificação XML da árvore de artigos, de acordo com o modelo de dados do DB2 Content Manager.

## Atributos <MoveItemRequest>

### checkout (facultativo)

Comuta para dar saída do artigo (e, assim, bloqueá-lo) ante de efectuar o respectivo pedido de transferência. O valor predefinido é TRUE.

### checkin (facultativo)

Comuta para dar entrada do artigo depois de efectuar o respectivo pedido de transferência. O valor predefinido é TRUE. Se não for dada saída do artigo esta definição falha.

O exemplo seguinte re-indexa (move) um artigo em CLAIM\_1047 para CLAIM2\_1047:

### Exemplo: pedido XML

```
<MoveItemRequest
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectionString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
</AuthenticationData>
<Item URI="90 3 ICM8 icmnlsdb10 CLAIM_104759 26
A1001001A04I13B04803F2085118 A04I13B04803F208511 14 1007"/>
<NewValues newEntityName="CLAIM2_1047">
 <ItemXML>
 <CLAIM2_1047 xmlns:cm="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema"
DESC_1047="This is a claim regarding the accident.">
 <cm:properties type="document"/>
 <VEHICLE2_1047 VIN_1047="38"></VEHICLE2_1047>
 </CLAIM2_1047>
 </ItemXML>
</NewValues>
</MoveItemRequest>
```

Como resultado, o DB2 Content Manager devolve uma <MoveItemReply > XML que contém o novo URI (para CLAIM2\_1047):

### Exemplo: resposta XML

```
<MoveItemReply
xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
<Item URI="91 3 ICM8 icmnlsdb11 CLAIM2_104759 26
A1001001A04I13B04803F2085118 A04I16B41357H241001 14 1016"/>
</MoveItemReply>
```

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Aceder ao encaminhamento de documentos do DB2 Content Manager utilizando pedidos baseados em XML

O DB2 Content Manager faculta determinados serviços da Web que podem facilitar o encaminhamento de documentos através de um processo empresarial que utiliza as APIs.

**Importante:** Para definir processos de encaminhamento de documentos no DB2 Content Manager Versão 8, tem de utilizar o construtor gráfico de processos.

**Aviso:** Criar um processo de encaminhamento de documentos com APIs Versão 8.4 acarreta o risco de comportamentos inesperados e danos no sistema. O construtor gráfico previne isso, avaliando um processo antes de o guardar no servidor de bibliotecas.

## Enumerar nós de trabalho nos pedidos XML

O *nó de trabalho* remete para um passo no qual pacotes de trabalho aguardam acções a efectuar por utilizadores ou aplicações ou transitam automaticamente.

para enumerar todos os nós de trabalho do DB2 Content Manager, crie um elemento `<ListWorkNodesRequest>` que contenha as seguintes informações:

```
<ListWorkNodesRequest>
 <AuthenticationData> ... </AuthenticationData>
</ListWorkNodesRequest>
```

Como resultado, o DB2 Content Manager devolve `<ListWorkNodesReply>` em XML que contém todos os nós de trabalho no formato seguinte:

```
<WorkNode>
 <WorkNodeName>string</WorkNodeName>
 [<Description>string</Description>]
 <LongDescription>string</LongDescription>
 <ACLName>string</ACLName>
 [<TimeLimit>minutes</TimeLimit>]
 [<OverLoadLimit>nonNegativeInteger</OverLoadLimit>]
 [<ActionListName>string</ActionListName>]
 [<ContainerDefinition> ... </ContainerDefinition>]
 [<WorkNodeExt><NameValuePair>
 <Name>attribute</Name><Value>string</Value>
 </NameValuePair>
 <NameValuePair> ... </NameValuePair>]
 </WorkNodeExt>]
 [<Type>0 | 1 | 6</Type>]
 <!-- Type 0 (work basket) also requires: -->
 [<EnterUserDLL>fileName</EnterUserDLL>
 <!-- or --> <EnterUserFunction>string</EnterUserFunction>]
 [<LeaveUserDLL>fileName</LeaveUserDLL>
 <!-- or --> <LeaveUserFunction>string</LeaveUserFunction>]
 [<OverloadUserDLL>fileName</OverloadUserDLL>]
 <!-- or --> <OverloadUserFunction>string</OverloadUserFunction>]
 <!-- Type 1 (collection pt), also requires: -->
 [<CollectionResumeListEntry>
 <FolderItemTypeName>string</FolderItemTypeName>
 <RequiredItemTypeName>string</RequiredItemTypeName>
 <QuantityNeeded>int</QuantityNeeded>
 </CollectionResumeListEntry>]
</WorkNode>
```

### **<WorkNodeName> (necessário)**

Identifica como invocar o nó de trabalho.

### **<Description> (facultativo)**

Resume o objectivo do nó de trabalho e apresenta no cliente de administração do sistema. O número máximo de caracteres é 254.

### **<LongDescription> (necessário)**

Explicita o nó de trabalho em pormenor e apresenta na janela de propriedades. O número máximo de caracteres é 2048.

### **<ACLName> (necessário)**

Especifica o nome de uma lista de controlo de acessos (ACL) para limitar quais os utilizadores que podem ter acesso aos artigos no nó de trabalho.

**<TimeLimit> (facultativo)**

Especifica os minutos que podem decorrer até que os pacotes de trabalho expirem no nó de trabalho (que vai então definir o sinalizador de notificação para 1). O valor predefinido é 0 (sem tempo limite).

**<OverLoadLimit> (facultativo)**

Especifica o número máximo de documentos ou pastas que o nó de trabalho pode conter antes do sistema invocar a saída de sobrecarga tal como especificado pela DLL ou pela função. Pode especificar 0 se não for necessário qualquer limite.

**<ActionListName> (facultativo)**

Especifica a lista de acções que um utilizador pode desempenhar em artigos de trabalho. Pode atribuir uma lista de acções a cada nó num processo para especificar as acções que o utilizador pode efectuar nessa fase do processo.

**<ContainerDefinition> (facultativo)**

Define um contentor que pode transportar variáveis de um pacote de trabalho para o seguinte enquanto o pacote de trabalho continua no processo. Pode conter os seguintes elementos:

**<ReadOnly>*false*</ReadOnly>**

Indica se o nó de trabalho pode ser actualizado. O valor predefinido é *false* (pode ser actualizado). Se for definido como *true*, o nó de trabalho passa a ser apenas de leitura.

**<VariableName>*string*</VariableName>**

Especifica o nome da variável deste nó de trabalho. O limite é 32 caracteres.

**<VariableType>*0*</VariableType>**

Especifica o tipo da variável deste nó de trabalho. Pode ter uma das seguintes constantes:

**0 (CMB\_ICM\_TYPE\_CHARACTER)**

Carácter

**1 (CMB\_ICM\_DR\_WNV\_TYPE\_INTEGER)**

Número inteiro

**3 (CMB\_ICM\_DR\_WNV\_TYPE\_TIMESTAMP)**

Marca de hora

**<VariableValue>*string*</VariableValue>**

Especifica o valor da variável deste nó de trabalho. O limite predefinido é 254 caracteres.

**<VariableLength>*0*</VariableLength>**

Especifica o comprimento máximo permitido para o <VariableValue> da variável deste nó de trabalho. Este atributo só é válido quando o atributo "type" existe com <VariableType> definido como 0. A predefinição é 0 length.</xs:documentation>

**<VariablePrompt>*string*</VariablePrompt>**

Especifica a linha de comandos da variável deste nó de trabalho. É utilizada para pedir ao utilizador para actualizar o atributo de valor da variável deste nó de trabalho. O limite é 32 caracteres.

**<Required>*false*</Required>**

Indica se é necessário o <VariableValue> da variável deste nó de trabalho. A predefinição é *false*. Se for definido como *true*, a

aplicação cliente precisa que o utilizador actualize o <VariableValue> da variável deste nó de trabalho.

**<ShowToUser>false</ShowToUser>**

Indica se a variável deste nó de trabalho é apresentada ao utilizador cliente. A predefinição é false. Se o valor for definido como true, a variável deste nó de trabalho é apresentada ao utilizador cliente para a actualizar.

**<WorkNodeExt> (facultativo)**

Expande um nó de trabalho com atributos definidos pelo utilizador. Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador.

**<Type> (facultativo)**

O tipo de nó de trabalho pode ser 0 (cesto de trabalho) 1 (ponto de recolha) ou 6 (nó de aplicação empresarial):

**0 (cesto de trabalho)**

É uma fase em que o sistema mantém os pacotes de trabalho que estão a ser processados ou à espera de serem processados. Um cesto de trabalho não efectua quaisquer acções no conteúdo. Quando um utilizador ou uma aplicação concluem a tarefa solicitada no pacote de trabalhos, são encaminhados para o nó de trabalho seguinte. Este é o tipo predefinido do nó de trabalho.

**1 (ponto de recolha)**

É um nó de trabalho especial que não corresponde a uma tarefa empresarial e ao qual os utilizadores não têm acesso. Representa uma área em que a pasta aguarda artigos específicos (outras pastas ou documentos) para serem recolhidos antes de continuar. O ponto de recolha encaminha o pacote de trabalhos para outro nó de trabalho em duas circunstâncias: quando a pasta está concluída ou quando o tempo limite expira.

**6 (nó da aplicação empresarial)**

É uma fase de processamento em que o servidor de bibliotecas invoca um DLL de saída de utilizador para executar aplicações de áreas empresariais.

Todos os cestos de trabalho precisam que a saída de utilizador determine as tarefas que o pacote de trabalhos tem de concluir quando entra num cesto de trabalho, quando sai de um cesto de trabalho ou quando um cesto de trabalho fica cheio. A saída de utilizador pode ser uma biblioteca de ligação dinâmica (DLL) ou uma função (dependendo da linguagem de programação utilizada). As selecções são:

**<EnterUserDLL> ou <EnterUserFunction> (facultativo)**

Especifica a DLL ou função a invocar quando o pacote de trabalhos entra no nó de trabalho.

**<LeaveUserDLL> ou <LeaveUserFunction> (facultativo)**

Especifica a DLL ou função a invocar quando o pacote de trabalhos deixa o nó de trabalho.

**<OverloadUserDLL> ou <OverloadUserFunction> (facultativo)**

Especifica a DLL ou função a invocar quando o nó de trabalho atinge o limite.

Todos os pontos de recolha precisam de um elemento

<CollectionResumeListEntry> para atribuir uma lista de tipos de artigo que o tipo

de artigos da pasta tem de aguardar antes de prosseguir o processo. O `<CollectionResumeListEntry>` tem de conter os seguintes elementos:

**`<FolderItemTypeName>` (necessário)**

Identifica o nome da pasta que vai reunir os artigos. O pacote de trabalhos tem de estar associado a este tipo de pasta, caso contrário vai saltar este ponto de recolha.

**`<RequiredItemTypeName>` (necessário)**

Especifica os tipos de artigos reunidos na pasta.

**`<QuantityNeeded>` (necessário)**

Especifica a quantidade de artigos necessários para cumprir o requisito do tipo de artigo.

## Enumerar processos de documentos com pedidos XML

Um *processo* corresponde a uma série de fases que contém, pelo menos, um nó de partida, um nó final e uma selecção ou acção.

Para enumerar todos os processos de DB2 Content Manager, crie um elemento `<ListProcessRequest>` que contenha as seguintes informações:

```
<ListProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
</ListProcessRequest>
```

**Sugestão:** Em alternativa, pode substituir o identificador `<ListProcessRequest>` por `<ListProcessNamesRequest>` se apenas pretender uma lista dos nomes dos processos.

Como resultado, o DB2 Content Manager devolve um elemento `<ListProcessReply>` XML que contém todos os processos no seguinte formato:

```
<Process>
 <ProcessName>string</ProcessName>
 <ACLName>string</ACLName>
 <Route>
 <RouteDefinition>
 <From>string</From><To>string</To>
 <RouteSelected>string</RouteSelected>
 [<Extension><NameValuePair>
 <Name>attribute</Name><Value>string</Value>
 </NameValuePair> ... </Extension>]
 </RouteDefinition>
 [<RouteDefinition> ... </RouteDefinition>]
 </Route>
 [<Description>string</Description>]
 [<LongDescription>string</LongDescription>]
 [<TimeLimit>minutes</TimeLimit>]
 [<Extension><NameValuePair>
 <Name>attribute</Name><Value>string</Value>
 </NameValuePair>
 [<NameValuePair> ... </NameValuePair>]
 </Extension>]
</Process>
```

### Elementos `<Process>`

**`<ProcessName>` (necessário)**

Identifica como invocar o processo.

**`<ACLName>` (necessário)**

Especifica o nome de uma lista de controlo de acessos (ACL) para limitar quais os utilizadores que podem ter acesso ao processo.



**<Route> (necessário)**

Define o nó FROM, o nó TO e a selecção (que é uma cadeia). Se optar por selecção, o pacote de trabalhos passa para o nó de trabalho definido em TO.

**<Description> (facultativo)**

Resume o objectivo do processo e apresenta no cliente de administração do sistema. O número máximo de caracteres é 254.

**<LongDescription> (facultativo)**

Explicita o processo em pormenor e apresenta na janela de propriedades. O número máximo de caracteres é 2048.

**<TimeLimit> (facultativo)**

Especifica os minutos que decorrem até que o processo seja concluído. Quando expira, o sinalizador de notificação no pacote de trabalhos é definido para 1. O valor predefinido é 0 (sem limite de tempo).

**<Extension> (facultativo)**

Expande um processo com atributos definidos pelo utilizador (anexa mais colunas à tabela do servidor de bibliotecas). Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador. Difere das variáveis do contentor na medida em que não é transportado com o pacote de trabalhos.

**Enumerar listas de trabalho com pedidos XML**

Uma *lista de trabalhos* contém uma lista ordenada dos pacotes de trabalhos (documentos ou pastas) que o utilizador tem de cumprir. As listas de trabalho têm características como, por exemplo, ordenar (por prioridade ou data), filtrar (no estado suspenso ou no estado notificar) e quantidade a devolver. Estas características controlam a forma como os utilizadores vêem os seus trabalhos.

Para enumerar todas as listas de trabalho de DB2 Content Manager, crie um <ListWorkListsRequest > que contenha as informações seguintes:

```
<ListWorkListsRequest>
 <AuthenticationData> ... </AuthenticationData>
</ListWorkListsRequest>
```

**Sugestão:** Em alternativa, pode substituir o identificador <ListWorkListRequest> por <ListWorkListNamesRequest> se apenas pretender uma lista dos nomes das listas de trabalho.

Como resultado, o DB2 Content Manager devolve <ListWorkListReply > em XML que contém todos os processos no formato seguinte:

```
<WorkList>
 <WorkListName>string</WorkListName>
 <ACLName>string</ACLName>
 <WorkNodeNames>string</WorkNodeNames>
 [<Description>string</Description>]
 [<SelectionOrder>0 | 1</SelectionOrder>]
 [<SelectionFilterOnNotify>0 | 1 | 2</SelectionFilterOnNotify>]
 [<SelectionFilterOnSuspend>0 | 1 | 2</SelectionFilterOnSuspend>]
 [<SelectionFilterOnOwner>0 | 1</SelectionFilterOnOwner>]
 [<WorkPackagesToReturn>nonNegativeInteger</WorkPackagesToReturn>]
</WorkList>
```

**Elementos <WorkList>****<WorkListName> (necessário)**

Identifica o nome de uma lista de trabalho.



**<ACLName> (necessário)**

Especifica o nome de uma lista de controlo de acessos para limitar quais os utilizadores que podem ter acesso aos nós de trabalho e aos artigos de trabalho contidos na lista de trabalho.

**<WorkNodeNames> (necessário)**

Especifica quais os nós de trabalho a incluir na lista de trabalhos por ordem ou por prioridades.

**<Description> (facultativo)**

Explica a lista de trabalho.

**<SelectionOrder> (facultativo)**

Determina a ordem de apresentação dos pacotes de trabalhos na lista de trabalhos. As escolhas são:

- 0 Ordena e devolve os pacotes de trabalhos pela prioridade definida pelo utilizador.
- 1 Ordena e devolve os pacotes de trabalhos pela respectiva data de criação.

**<SelectionFilterOnNotify> (facultativo)**

Filtra os pacotes de trabalho por estado de notificação. As escolhas são:

- 0 Devolve apenas os pacotes de trabalho que não têm o sinalizador de notificação activado.
- 1 Devolve apenas os pacotes de trabalho que têm o sinalizador de notificação activado.
- 2 Devolve os pacotes de trabalho independentemente da definição do sinalizador de notificação.

**<SelectionFilterOnSuspend> (facultativo)**

Filtra os pacotes de trabalho por estado de suspensão. As escolhas são:

- 0 Devolve apenas os pacotes de trabalho que não têm o sinalizador de suspensão activado.
- 1 Devolve apenas os pacotes de trabalho que têm o sinalizador de suspensão activado.
- 2 Devolve os pacotes de trabalho independentemente da definição do sinalizador de suspensão.

**<SelectionFilterOnOwner> (facultativo)**

Filtra os pacotes de trabalho por utilizador detentor. As escolhas são:

- 0 Devolve os pacotes de trabalho independentemente de quem os detém.
- 1 Devolve apenas os pacotes de trabalho que o utilizador detém.

**<WorkPackagesToReturn> (facultativo)**

Especifica o número máximo de pacotes de trabalho a devolver. O valor predefinido é 0 (todos os pacotes de trabalho são devolvidos).

## **Enumerar pacotes de trabalhos com pedidos XML**

Um *pacote de trabalhos* é um contentor para os artigos que constam de uma lista de trabalho ou de um processo.

Para enumerar todos pacotes de trabalhos do DB2 Content Manager, crie um `<ListWorkPackagesRequest >` que contenha as informações seguintes (os elementos entre parênteses rectos são facultativos):

```

<ListWorkPackagesRequest>
 <AuthenticationData> ... </AuthenticationData>
 <WorkListName>string</WorkListName>
 [<WorkPackageOwner>string</WorkPackageOwner>]
</ListWorkPackagesRequest>

```

## Elementos <ListWorkPackagesRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere. Para obter informações sobre dados de autenticação, consulte “Autenticar pedidos de serviços da Web para segurança” na página 376.

### <WorkListName> (necessário)

Especifica uma lista de trabalhos para enumerar todos os pacotes de trabalhos.

### <WorkPackageOwner> (facultativo)

Especifica um proprietário para enumerar todos os pacotes de trabalhos.

Uma outra alternativa é substituir o identificador <ListWorkPackagesRequest > pelo identificador <ListNextWorkPackagesRequest > para devolver o pacote de trabalhos seguinte no <WorkListName> (necessário) e por <WorkPackageOwner> (facultativo) e dar saída do artigo referido. Se o artigo estiver escondido pelo filtro ou a respectiva saída já tiver sido dada por outro utilizador, o pedido salta o artigo e passa para o seguinte. Quando este pedido for invocado novamente e a lista de pacotes de trabalhos for a mesma, é devolvido o mesmo pacote de trabalhos.

Ainda uma outra alternativa é substituir o identificador <ListWorkPackagesRequest > pelo identificador <ListWorkPackageCheckedOutOptionRequest > para devolver o pacote de trabalhos pelo respectivo URI (e eventualmente dar saída do artigo):

```

<ListWorkPackageCheckedOutOptionRequest>
 <AuthenticationData> ... </AuthenticationData>
 <WorkPackageURI>string</WorkPackageURI>
 <CheckedOutItem>boolean</CheckedOutItem>
</ListWorkPackageCheckedOutOptionRequest>

```

## Elementos <ListWorkPackageCheckedOutOptionRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <WorkPackageURI> (necessário)

Especifica o identificador persistente do pacote de trabalhos a enumerar.

### <CheckedOutItem> (necessário)

Se for definido como true, é dada saída do artigo associado ao pacote de trabalhos.

Como resultado de um dos três tipos de pedidos “list work package”, o DB2 Content Manager devolve o pacote de trabalhos <ListWorkPackagesReply > em XML, <ListNextWorkPackagesReply > em XML ou <ListWorkPackageCheckedOutOptionRequest > em XML, utilizando o seguinte formato:

```

<WorkPackage>
 <WorkPackageURI>aniURI</WorkPackageURI>
 <ItemIDURI>string</ItemIDURI>
 <ProcessName>string</ProcessName>
 <WorkNodeName>string</WorkNodeName>
 [<WorkPackageOwner>string</WorkPackageOwner>]
 [<Priority>nonNegativeInteger</Priority>]
 [<SuspendState>false</SuspendState>]
 [<NotifyState>false</NotifyState>]
 [<NotifyTime>yyy-mo-dd-hh.mi.ss</NotifyTime>]
 [<ResumeList><ResumeListDefinition>
 <RequiredItemTypeName>string</RequiredItemTypeName>
 <QuantityNeeded>0</QuantityNeeded>
 </ResumeListDefinition>]
 </ResumeList>]
 [<ResumeTime>yyy-mo-dd-hh.mi.ss</ResumeTime>]
 [<TimeLastMoved>yyy-mo-dd-hh.mi.ss</TimeLastMoved>]
 [<UserLastMoved>string</UserLastMoved>]
 [<ProcessCompletionTime>yyy-mo-dd-hh.mi.ss</ProcessCompletionTime>]
 [<ContainerData><NameValuePair>
 <Name>attribute</Name><Value>string</Value>
 </NameValuePair>
 [<NameValuePair> ... </NameValuePair>]
 </ContainerData>]
</WorkPackage>

```

## Elementos <WorkPackage>

### <WorkPackageURI> (necessário)

Indica o identificador persistente do pacote de trabalhos.

### <ItemIDURI> (necessário)

Indica o identificador persistente do artigo a que está associado o pacote de trabalhos.

### <ProcessName> (necessário)

Indica o nome do processo em que está o pacote de trabalhos.

### <WorkNodeName> (necessário)

Indica o nome do nó de trabalho onde está actualmente localizado o pacote de trabalhos.

### <WorkPackageOwner> (facultativo)

Indica o proprietário do pacote de trabalhos.

### <Priority> (facultativo)

Indica uma prioridade arbitrária definida pelo utilizador para o pacote de trabalhos. Por exemplo, 1, 2, 3...

### <SuspendState> (facultativo)

Indica se o pacote de trabalhos ficou suspenso num processo.

### <NotifyState> (facultativo)

Indica se o pacote de trabalho notifica os utilizadores quando se move para outro nó de trabalho.

### <NotifyTime> (facultativo)

Indica a hora (no chamado formato europeu: *aaaa-mm-dd-hh.mm.ss*) a que o pacote de trabalhos passou para o estado notificar.

### <ResumeList> (facultativo)

Só se aplica a uma pasta de um pacote de trabalhos e precisa de um identificador <ResumeListDefinition>. Mantém o processo suspenso até que um determinado tipo de artigo (especificado em

<RequiredItemTypeName>) numa determinada quantidade (especificada em <QuantityNeeded>) apareça na pasta. O valor predefinido de <QuantityNeeded> é 0.

**<ResumeTime> (facultativo)**

Indica a hora (no chamado formato europeu: *aaaa-mm-dd-hh.mm.ss*) a que o pacote de trabalhos passou para o estado continuar.

**<TimeLastMoved> (facultativo)**

Indica a hora (no chamado formato europeu: *aaaa-mm-dd-hh.mm.ss*) a que o pacote de trabalhos foi mexido pela última vez.

**<UserLastMoved> (facultativo)**

Indica o utilizador que foi o último a mover o pacote de trabalhos.

**<ProcessCompletionTime> (facultativo)**

Tempo total (no chamado formato europeu: *aaaa-mm-dd-hh.mm.ss*) que o pacote de trabalhos gastou no processo.

**<ContainerData> (facultativo)**

Expande um pacote de trabalhos com atributos definidos pelo utilizador. Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador.

**Informações relacionadas**



Autenticar pedidos de serviços da Web para segurança

## Enumerar acções no DB2 Content Manager com pedidos XML

Uma *acção* específica como o utilizador pode manipular os pacotes de trabalhos num nó de trabalho.

Para enumerar os valores de uma determinada acção do DB2 Content Manager, pode efectuar um dos seguintes passos:

1. Criar um <ListActionNamesRequest> que contenha as informações seguintes (os elementos entre parênteses rectos são facultativos) para devolver uma lista de todos os ActionNames numa determinada ActionList:

```
<ListActionNamesRequest>
 <AuthenticationData> ... </AuthenticationData>
 <ActionListName>string</ActionListName>
</ListActionNamesRequest>
```

2. Criar um <ListActionRequest> que contenha as informações seguintes (os elementos entre parênteses rectos são facultativos) para enumerar os valores de um ActionName:

```
<ListActionRequest>
 <AuthenticationData> ... </AuthenticationData>
 <ActionName>string</ActionName>
</ListActionRequest>
```

Como resultado, o DB2 Content Manager devolve <ListActionReply> em XML que contém todos os processos no formato seguinte:

```
<ListActionReply>
 <ActionName>string</ActionName>
 <Predefined>boolean</Predefined>
 [<AuditComment>string</AuditComment>]
 [<Description>string</Description>]
 [<Icon>string</Icon>]
 [<DisplayName>string</DisplayName>]
 [<Shortcut>string</Shortcut>]
```

```
[<FunctionName>string</FunctionName>]
[<ApplicationName>string</ApplicationName>]
[<DLLName>string</DLLName>]
</ListActionReply>
```

## Elementos <ListActionRequest> numa ActionList

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <ActionListName> (necessário)

Especifica o nome da lista de acções que enumera os nomes de acções.

## Elementos <ListActionRequest> para enumerar os valores de um ActionName

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <ActionName> (necessário)

Especifica o nome da acção a enumerar.

## Elementos <Action>

### <ActionName> (necessário)

Especifica o nome da acção.

### <Predefined> (necessário)

Se estiver definido como true, esta acção será uma acção definida no DB2 Content Manager. Se estiver definido como false, esta acção será uma acção definida pelo utilizador.

### <AuditComment> (facultativo)

Especifica um comentário de seguimento de auditoria.

### <Description> (facultativo)

Explica o conteúdo em detalhe.

### <Icon> (facultativo)

Especifica a localização do ícone da aplicação cliente para a apresentação desta acção. Por exemplo, C:\icon.gif.

### <DisplayName> (facultativo)

Especifica o nome da aplicação cliente para a apresentar como uma escolha de menu ao utilizador. Por exemplo, Insurance calculator.

### <Shortcut> (facultativo)

Especifica as teclas no teclado para acesso rápido à acção. Por exemplo, Ctrl+Shift+F4.

### <FunctionName> (facultativo)

Especifica o nome da função no DLL a invocar quando o utilizador selecciona esta acção. Por exemplo, WXV2UserExitSample. A função é executada na aplicação cliente.

### <ApplicationName> (facultativo)

Especifica a saída de utilizador a invocar quando o utilizador selecciona esta acção. Por exemplo, C:\myapp.jsp. A saída é executada na aplicação cliente.

**<DLLName> (facultativo)**

Especifica o DLL da biblioteca de ligações a invocar quando o utilizador selecciona esta acção. Por exemplo, C:\WXV2UserExitSample.dll. O DLL é executado na aplicação cliente.

**Informações relacionadas**

Autenticar pedidos de serviços da Web para segurança

## Actualizar um pacote de trabalhos com UpdateWorkPackageRequest

<UpdateWorkPackageRequest> actualiza objectos num pacote de trabalho.

Para actualizar determinados objectos num pacote de trabalhos do DB2 Content Manager, crie um <UpdateWorkPackageRequest> XML que identifique as informações seguintes (o texto entre parênteses rectos é facultativo):

```
<UpdateWorkPackageRequest>
 <WorkPackageURI>string</WorkPackageURI>
 [<WorkPackageOwner>string</WorkPackageOwner>]
 [<Priority>nonNegativeInteger</Priority>]
 [<ContainerData><NameValuePair>
 <Name>attribute</Name><Value>string</Value>
 </NameValuePair>
 <NameValuePair> ... </NameValuePair>]
 </ContainerData>]
</UpdateWorkPackageRequest>
```

**Elementos <UpdateWorkPackageRequest>****<WorkPackageURI> (necessário)**

Especifica o identificador persistente do pacote de trabalhos a actualizar.

**<WorkPackageOwner> (facultativo)**

Especifica o proprietário do pacote de trabalhos.

**<Priority> (facultativo)**

Especifica uma prioridade arbitrária definida pelo utilizador para o pacote de trabalhos. Por exemplo, 1, 2, 3...

**<ContainerData> (facultativo)**

Expande um pacote de trabalhos com atributos definidos pelo utilizador. Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador.

O exemplo de consulta seguinte actualiza um ContainerData definido pelo utilizador para o pacote de trabalhos especificado.

**Exemplo: pedido XML**

```
<UpdateWorkPackageRequest xmlns=
"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>cmdb21s</ServerName></ServerDef>
 <LoginData><UserID>icmadmin</UserID><Password>o8rmond</Password>
 </LoginData>
</AuthenticationData>
 <WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
 A1001001A04H17B32803I2340818 A04H17B33216H440631 03
 204</WorkPackageURI>
 <WorkPackageOwner>icmadmin</WorkPackageOwner>
 <Priority>2004</Priority>
 <ContainerData>
```

```

<NameValuePair><NVPName>Loan amount</NVPName>
<NVPValue>1000</NVPValue></NameValuePair>
<NameValuePair><NVPName>last name </NVPName>
<NVPValue>Latariot</NVPValue></NameValuePair>
</ContainerData>
</UpdateWorkPackageRequest>

```

Como resultado, o DB2 Content Manager devolve um <UpdateWorkPackageReply> XML que indica o sucesso.

## Iniciar um processo de encaminhamento de documentos com StartProcessRequest

<StartProcessRequest> inicia um processo de encaminhamento de documentos.

Para iniciar um DB2 Content Manager através do processo de encaminhamento de documentos, crie um <StartProcessRequest> que identifique as informações seguintes (os elementos entre parênteses rectos são facultativos):

```

<StartProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
 <ProcessName>string</ProcessName>
 [<WorkPackageOwner>string<WorkPackageOwner>]
 [<Priority>nonNegativeInteger</Priority>]
 <ItemIDURI> ... </ItemIDURI>
 [<ContainerData> ... </ContainerData>]
</StartProcessRequest>

```

### Elementos <StartProcessRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <ProcessName> (necessário)

Identifica como invocar o processo.

#### <WorkPackageOwner> (facultativo)

Especifica o proprietário do pacote de trabalhos.

#### <Priority> (facultativo)

Especifica uma prioridade arbitrária definida pelo utilizador para o pacote de trabalhos. Por exemplo, 1, 2, 3...

#### <ItemIDURI> (necessário)

Especifica o identificador persistente do artigo do DB2 Content Manager com o qual iniciar o processo.

#### <ContainerData> (facultativo)

Expande um pacote de trabalhos com atributos definidos pelo utilizador. Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador.

O seguinte exemplo de consulta inicia o processo ClaimsProcess e atribui <ContainerData> definido pelo utilizador ao respectivo pacote de trabalhos.

### Exemplo: pedido XML

```

<StartProcessRequest xmlns=
"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN"
 configString=""><ServerDef>
 <ServerType>ICM</ServerType><ServerName>cmdb21s</ServerName>
 </ServerDef><LoginData><UserID>icmadmin</UserID>

```



```

 <Password>o8rmond</Password></LoginData>
</AuthenticationData>
<ProcessName>ClaimsProcess</ProcessName>
<Priority>10</Priority>
<ContainerData><NameValuePair><NVPName>Loan amount</NVPName>
 <NVPValue>1000</NVPValue></NameValuePair>
 <NameValuePair><NVPName>First name</NVPName>
 <NVPValue>Carly </NVPValue></NameValuePair>
 <NameValuePair><NVPName>Last name</NVPName>
 <NVPValue>Morreale</NVPValue></NameValuePair>
</ContainerData>
<ItemIDURI>86 3 ICM7 cmdb21s8 Claims2859 26
 A1001001A04H17B32800G0799718 A04H17B32800G079971
 14 1007</ItemIDURI>
</StartProcessRequest>

```

Como resultado, o DB2 Content Manager devolve um <StartProcessReply > XML que indica o sucesso.

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Terminar um processo com TerminateProcessRequest

Um processo pode ser explicitamente terminado antes deste alcançar um nó final. O que vai remover do sistema o pacote de trabalhos (em encaminhamento). É dada entrada do artigo referido no pacote de trabalhos, caso tenha sido dada saída do mesmo. Para encaminhamentos paralelos com vários pacotes de trabalhos, qualquer um destes pacotes de trabalhos pode ser utilizado para terminar o processo.

Para terminar um processo, crie um <TerminateProcessRequest > que identifique as informações seguintes:

```

<TerminateProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
 <WorkPackageURI> ... </WorkPackageURI>
</TerminateProcessRequest>

```

### Elementos <TerminateProcessRequest>

#### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

#### <WorkPackageURI> (necessário)

Especifica o identificador persistente do pacote de trabalhos que está a ser encaminhado pela instância do processo.

O exemplo de consulta seguinte termina o processo para o pacote de trabalhos especificado.

### Exemplo: pedido XML

```

<TerminateProcessRequest
 xmlns="http://www.ibm.com/software/data/cmb/schemas/">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlbdb</ServerName></ServerDef>
 <LoginData><UserID>icmadmin</UserID>
 <Password>aPassword</Password></LoginData>
 </AuthenticationData>

```



```

<WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
A1001001A04H17B32803I2340818 A04H17B33216H440631 03
204</WorkPackageURI>
</TerminateProcessRequest>

```

Como resultado, o DB2 Content Manager devolve um <TerminateProcessReply > XML que indica o sucesso. Se o processo terminou, não é devolvido qualquer URI de pacote de trabalhos.

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Continuar um processo com ContinueProcessRequest

ContinueProcessRequest continua o pacote de trabalho (identificado pelo URI) para o nó de trabalho seguinte. O que vai remover o pacote de trabalho especificado do sistema e criar um novo pacote de trabalho (no caso de encaminhamentos paralelos). É dada entrada do artigo referido pelo URI de artigo, caso tenha sido dada saída do mesmo. Se o processo terminou, não é devolvido qualquer URI de pacote de trabalho.

Para iniciar um artigo do DB2 Content Manager através do processo de encaminhamento de documentos, crie um <ContinueProcessRequest > que identifique as informações seguintes (os elementos entre parênteses rectos são facultativos):

```

<ContinueProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
 [<WorkPackageURI>string</WorkPackageURI>]
 <WorkPackageOwner>string</WorkPackageOwner>
 <RouteSelected> ... </RouteSelected>
 [<ContainerData> ... </ContainerData>]
</ContinueProcessRequest>

```

## Elementos <ContinueProcessRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <WorkPackageURI> (facultativo)

Identifica o identificador persistente do pacote de trabalho para ir para o nó de trabalho seguinte.

### <WorkPackageOwner> (necessário)

Especifica o proprietário do pacote de trabalho.

### <RouteSelected> (necessário)

Especifica o nome do encaminhamento que o processo deve tomar.

### <ContainerData> (facultativo)

Expande um pacote de trabalho com atributos definidos pelo utilizador. Este elemento pode conter um ou mais elementos <NameValuePair> para atributos definidos pelo utilizador.

## Exemplo: pedido XML

```

<ContinueProcessRequest xmlns=
"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>cmdb21s</ServerName></ServerDef>
 </AuthenticationData>
 <WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
A1001001A04H17B32803I2340818 A04H17B33216H440631 03
204</WorkPackageURI>
 <WorkPackageOwner>string</WorkPackageOwner>
 <RouteSelected> ... </RouteSelected>
 [<ContainerData> ... </ContainerData>]
</ContinueProcessRequest>

```

```

 <LoginData><UserID>icmadmin</UserID>
 <Password>o8rmond</Password></LoginData>
</AuthenticationData>
<WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
 A1001001A04H17B32803I2340818 A04H17B33130C375901
 03 204</WorkPackageURI>
<WorkPackageOwner>icmadmin</WorkPackageOwner>
<RouteSelected>Accept</RouteSelected>
<ContainerData><NameValuePair>
 <NVPName>Loan amount</NVPName>
 <NVPValue>1000</NVPValue></NameValuePair>
 <NameValuePair><NVPName>Last name</NVPName>
 <NVPValue>McKenna</NVPValue></NameValuePair>
</ContainerData>
</ContinueProcessRequest>

```

O seguinte exemplo de consulta atribui <ContainerData> definido pelo utilizador ao pacote de trabalho especificado e continua o processo.

Como resultado, o DB2 Content Manager devolve um <ContinueProcessReply > XML que indica o sucesso.

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Suspending a process with SuspendProcessRequest

pode suspender o processo associado a um pacote de trabalhos específico, enviando um <SuspendProcessRequest >. Isto vai dar saída do artigo no pacote de trabalhos e manter o atributo SuspendState do pacote de trabalhos como true durante um determinado período de tempo. Para pastas de pacotes de trabalhos, pode definir várias <ResumeListDefinitions> que suspendem o processo até que cheguem determinados tipos e quantidades de artigos.

Para suspender um processo de encaminhamento de documentos do DB2 Content Manager, crie um <SuspendProcessRequest > que identifique as informações seguintes (os elementos entre parênteses rectos são facultativos):

```

<SuspendProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
 <WorkPackageURI> ... <WorkPackageURI>
 [<Duration>minutes</Duration>]
 [<ResumeListDefinition>
 <RequiredItemTypeName>string</RequiredItemTypeName>
 <QuantityNeeded>0</QuantityNeeded>
 </ResumeListDefinition>]
</SuspendProcessRequest>

```

## Elementos <SuspendProcessRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <WorkPackageURI> (necessário)

Especifica o identificador persistente do pacote de trabalhos no processo que pretende suspender.

### <Duration> (facultativo)

Especifica por quantos minutos o processo fica suspenso. A predefinição é 0.

### <ResumeListDefinition> (facultativo)

Só se aplica a uma pasta de um pacote de trabalhos. Mantém o processo suspenso até que um determinado tipo de artigo (especificado em <RequiredItemTypeName>) numa determinada quantidade (especificada em <QuantityNeeded>) apareça na pasta. A <QuantityNeeded> predefinida é 0.

O seguinte exemplo de consulta mantém o pacote de trabalhos especificado suspenso num processo com base nas regras seguintes:

- Se o processo for iniciado na pasta Policy28, o pacote de trabalhos fica suspenso durante 30 minutos ou até que 5 artigos apareçam na pasta Policy28 (o que acontecer primeiro).
- Se o processo for iniciado na pasta InsDoc28, o pacote de trabalhos fica suspenso durante 30 minutos ou até que 25 artigos apareçam na pasta InsDoc28 (o que acontecer primeiro).
- Se o processo for iniciado num documento ou numa pasta que não Policy28 nem InsDoc28, o processo fica suspenso durante 30 minutos.

### Exemplo: pedido XML

```
<SuspendProcessRequest xmlns=
"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<AuthenticationData connectionString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>cmdb21s</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>o8rmond</Password></LoginData>
</AuthenticationData>
<WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
A1001001A04H17B32803I2340818 A04H17B33113A150611 03
204</WorkPackageURI>
<Duration>30</Duration>
<ResumeListDefinition>
 <RequiredItemTypeName>Policy28</RequiredItemTypeName>
 <QuantityNeeded>5</QuantityNeeded>
</ResumeListDefinition>
<ResumeListDefinition>
 <RequiredItemTypeName>InsDoc28</RequiredItemTypeName>
 <QuantityNeeded>25</QuantityNeeded>
</ResumeListDefinition>
</SuspendProcessRequest>
```

Como resultado, o DB2 Content Manager devolve um SuspendProcessReply XML que indica o sucesso.

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

### Retomar um processo com ResumeProcessRequest

Pode forçar um processo suspenso a ser retomado, especificando o pacote de trabalho associado num elemento <ResumeProcessRequest>. Este volta a dar entrada do artigo do pacote de trabalhos e repõe o atributo SuspendState do pacote de trabalhos como false. Não são efectuados quaisquer encaminhamentos ou retiradas do artigo de trabalho associado.

Para retomar um processo de encaminhamento de documentos do DB2 Content Manager, crie um <ResumeProcessRequest > que identifique as informações seguintes:

```
<ResumeProcessRequest>
 <AuthenticationData> ... </AuthenticationData>
 <WorkPackageURI> ... <WorkPackageURI>
</ResumeProcessRequest>
```

## Elementos <ResumeProcessRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (elemento <ServerDef>), um ID de utilizador válido (<UserID>) e palavra-passe (<Password>)- ou uma credencial SSO do WebSphere.

### <WorkPackageURI> (necessário)

Especifica o identificador persistente do pacote de trabalhos no processo que pretende retomar.

O exemplo de consulta seguinte retoma um processo para o pacote de trabalhos especificado.

## Exemplo: pedido XML

```
<ResumeProcessRequest xmlns=
"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>cmdb21s</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>o8rmond</Password></LoginData>
 </AuthenticationData>
 <WorkPackageURI>89 3 ICM7 cmdb21s11 WORKPACKAGE58 26
 A1001001A04H17B32803I2340818 A04H17B33118B569521 03
 204</WorkPackageURI>
</ResumeProcessRequest>
```

Como resultado, o DB2 Content Manager devolve um <ResumeProcessReply > XML que indica o sucesso.

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança

## Colocar em lotes vários pedidos de pedidos XML

Pode combinar vários pedidos numa única mensagem com o identificador de translineação <BatchRequest >. Um pedido de lotes executa sequencialmente cada pedido dentro dele na mesma ligação. Pode melhorar o rendimento, diminuindo a quantidade de volume de tráfego de rede necessário para vários pedidos.

Os pedidos de lotes utilizam um único conjunto de dados de autenticação para todos os subpedidos (definido no nível de pedido batch). Os subpedidos imbricados no pedido batch são executados na ligação obtida com os dados de autenticação.

Se imbricar os subpedidos nos identificadores de transacção, todos os subpedidos voltam para trás se um deles falhar.

Para colocar em batch vários pedidos numa mensagem, crie um <BatchRequest > que identifique as seguintes informações (o texto em parênteses rectos é facultativo):

```

<BatchRequest>
 <AuthenticationData> ... </AuthenticationData>
 [<!-- Insert any number of requests here -->]
 [<Transaction><!-- Insert any number of requests here --></Transaction>]
</BatchRequest>

```

## Elementos <BatchRequest>

### <AuthenticationData> (necessário)

Identifica o servidor de conteúdos (<ServerDef>), um ID de utilizador válido (<UserID>) e uma palavra-passe (<Password>) - ou uma credencial SSO do WebSphere.

### <Transaction> (facultativo)

Identifica quais os pedidos a anular durante uma falha. Se um pedido no identificador <Transaction> falha, todos os pedidos com o mesmo identificador retrocedem. Não pode imbricar identificadores <Transaction>.

O exemplo seguinte elimina uma apólice e uma participação no mesmo <BatchRequest>:

## Exemplo: pedido XML

```

<BatchRequest
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 <ServerDef><ServerType>ICM</ServerType>
 <ServerName>icmnlsdb</ServerName></ServerDef><LoginData>
 <UserID>icmadmin</UserID><Password>passw0rd</Password></LoginData>
 </AuthenticationData>
 <Transaction>
 <Requests>

 <DeleteItemRequest
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 ...
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_InsPolicy59 26 A1001001A04I10B44211H0825818
 A04I10B44211H082581 14 1029&server=icmnlsdb&dsType=ICM" />
 </DeleteItemRequest>

 <DeleteItemRequest
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <AuthenticationData connectString="SCHEMA=ICMADMIN" configString="">
 ...
 </AuthenticationData>
 <Item URI="http://hostname/CMBGenericWebService/CMBGetPIDUrl?pid=93
 3 ICM8 icmnlsdb13 XYZ_ClaimForm59 26 A1001001A04I10B44213F1658218
 A04I10B44213F165821 14 1027&server=icmnlsdb&dsType=ICM" />
 </DeleteItemRequest>

 </Requests>
</Transaction>
</BatchRequest>

```

Como resultado, o DB2 Content Manager devolve um <BatchReply> XML que translineia as respostas de todos os pedidos XML que foram colocados em lote.

## Exemplo: resposta XML

```

<BatchReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
 <RequestStatus success="true"></RequestStatus>

```

```

<Transaction>
<Replies>

<DeleteItemReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus>
</DeleteItemReply>
<DeleteItemReply
 xmlns="http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema">
<RequestStatus success="true"></RequestStatus></DeleteItemReply>

</Replies>
</Transaction>
</BatchReply>

```

### Informações relacionadas



Autenticar pedidos de serviços da Web para segurança



---

## Trabalhar com serviços da Web

Pode utilizar um serviço da Web através das suas aplicações, com outras interfaces de serviços da Web ou em processos empresariais complexos para efectuar acções com um sistema DB2 Content Manager.

---

### Descrição geral dos serviços da Web

Os serviços da Web definem um modelo de comunicações entre programas orientadas para serviços baseado no formato de serviço de mensagens XML.

O modelo de serviços da Web é construído segundo normas existentes e emergentes como, por exemplo, Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Hyper Text Transfer Protocol (HTTP) e Services Description Language (WSDL).

XML é uma linguagem de identificador extensível que pode descrever estruturas complicadas de forma a que programas e pessoas as possam compreender facilmente. Os serviços da Web dependem em grande medida do XML. Utiliza dados textuais em vez de dados binários para representar conteúdo, como números inteiros, que são muitas vezes representadas de formas diferentes pelas várias linguagens de programação de hardware e software utilizadas hoje em dia. O XML é independente da linguagem e do sistema operativo, o que economiza tempo e recursos ao integrar aplicações no DB2 Content Manager

O SOAP (Simple Access Object Protocol) é um protocolo de envio de mensagens com base em XML que é utilizado para interacções de serviços da Web entre duas aplicações. Toda a comunicação entre serviços da Web é efectuada com mensagens SOAP. Uma mensagem SOAP contém os seguintes elementos:

- Envelope
  - Cabeçalho (facultativo)
  - Corpo
- Anexos (facultativo)

Normalmente, o envelope SOAP, com zero ou mais anexos, representa uma mensagem SOAP. O envelope de mensagem SOAP contém o cabeçalho e o corpo da mensagem. Os anexos da mensagem SOAP permitem à mensagem conter dados, que podem incluir dados em XML ou não (como ficheiros de texto e binários). Os cabeçalhos SOAP são utilizados para descrever o contexto e propósito da mensagem. Os cabeçalhos SOAP também facultam mecanismos para expandir uma mensagem SOAP para adicionar funções e definir funcionalidades como, por exemplo, segurança, prioridade e auditoria.

Numa arquitectura orientada para serviços, é fundamental a definição da interface. É a definição da interface que serve como contrato entre o que fornece o serviço da Web e o que o cliente pode esperar.

Os serviços da Web utilizam a WSDL, outro conjunto de identificadores XML utilizados para descrever a interface de serviços da Web. A WSDL descreve as localizações do serviço da Web, como estabelecer uma ligação ao mesmo, quais os parâmetros a transmitir no pedido SOAP, quais os valores a devolver, informações de associação, entre outros.



O modelo de serviços da Web faz com que as tecnologias e protocolos XML, HTTP, SOAP e WSDL forneçam um ambiente que torna a integração da aplicação mais fácil, mais rápida e com uma melhor relação entre eficácia e custo. Os serviços da Web permitem que qualquer aplicação activada para a rede e que conheça o XML invoque um serviço da Web, independentemente da linguagem de programação ou do sistema operativo envolvido.

Os serviços da Web apresentam as seguintes vantagens:

**Flexibilidade**

As interfaces universais não têm de ser mudadas com as inevitáveis alterações no software devido a alterações das necessidades empresariais.

**Agilidade de produtividade**

As rápidas ferramentas de montagem de aplicações permitem integrar rapidamente serviços da Web em novos processos empresariais ou experimentá-las com novas ideias empresariais.

**Poupança de custos**

Redução de requisitos de pessoal, substituição do processamento de papel e redução de erros.

**Favorecer os investimentos existentes**

Pode utilizar o antigo software de outras maneiras, construindo um nível de serviços da Web para acesso universal.

Para trabalhar com os serviços da Web do DB2 Content Manager, tem de ter um conhecimento de trabalho sobre a tecnologia destes serviços. Para obter mais informações sobre as normas de serviços da Web, consulte o sítio do World Wide Web Consortium (W3C) na Web: World Wide Web Consortium (W3C).

## **serviços da Web do DB2 Content Manager**

Uma interface de *serviços da Web* é um componente de software reutilizável, livremente acoplado, que pode ser localizado, publicado e invocado através de uma rede, como a Web.

O DB2 Content Manager fornece uma interface autónoma que se auto-descreve, denominada Interface de serviços da Web, que pode utilizar nas aplicações com outras interfaces de serviços da Web ou em processos empresariais complexos para efectuar continuamente acções no sistema do DB2 Content Manager.

Com a interface de serviços da Web, é possível integrar de forma dinâmica as aplicações com o DB2 Content Manager, independentemente da linguagem de programação em que estão escritas e do sistema operativo no qual são executadas. Pode utilizar a interface de serviços da Web para efectuar algo tão simples como ver um documento de texto ou pode incorporar a interface de serviços da Web em processos e aplicações empresariais de maior complexidade.

Por exemplo, numa companhia de seguros, pode incorporar uma interface de serviços da Web numa aplicação da Web existente que permite aos clientes imprimirem a respectiva apólice automóvel actual. Além disso, pode incorporar outras interfaces de serviços da Web na mesma aplicação, permitindo que os seus clientes visualizem o actual valor comercial do seu automóvel.

## As funções dos serviços da Web no DB2 Content Manager Versão 8.4

As funções mais importantes dos serviços da Web do DB2 Content Manager Versão 8.4 são:

- Suporta apenas o WebSphere Application Server Versão 6.1. Já não tem suporte para o WebSphere Application Server versão 5.1.1 e 6.0.x.
- Suporta o WebSphere Network Deployment e o conjunto de unidades do servidor WebSphere.
- Utiliza qualquer servidor de aplicações especificado por um utilizador da instalação. O programa de configuração de serviços não cria um servidor de aplicações designado (cmwebsvc) antes de implementar a aplicação de serviços da Web. Consulte a documentação do WebSphere para criar o servidor, caso este ainda não exista.
- Suporta um conjunto de ligações Websphere actualizado.

Os utilizadores da instalação têm de configurar o servidor da Web para obterem a aplicação dos serviços da Web. O programa de configuração de serviços da Web não configura o servidor da Web na versão 8.4.

---

## Implementação dos serviços da Web do DB2 Content Manager

A arquitectura da interface dos serviços da Web do DB2 Content Manager é baseada num modelo de comunicação de envio de mensagens, no qual documentos inteiros são trocados entre clientes de serviços e servidores. O documento transferido contém todos os detalhes necessários para especificar a operação a executar e os objectos nos quais a operação vai ser executada.

Uma das vantagens do modelo baseado no envio de mensagens de documentos é que a especificação XML foi desenvolvida para permitir que dados normais, que normalmente são bloqueados num formato de proprietário, sejam descritos num formato aberto que se descreva e valide automaticamente e que seja legível pelos utilizadores. Quando um serviço da Web utiliza o envio de mensagens, pode recorrer a todas as capacidades de XML para descrever e validar documentos empresariais de alto nível. Outra vantagem é que, mesmo que sejam efectuadas melhorias e alterações ao esquema XML, não ocorrem falhas na aplicação de chamada.

O modelo de envio de mensagens de documento torna a troca de objectos mais flexível, uma vez que a concepção de um modelo empresarial normalmente adapta-se bem a arquitecturas orientadas para objectos. Como resultado, podem ser concebidas duas aplicações para trocar o estado de um objecto, utilizando XML. Ao contrário da numeração em série de objectos, numa troca de objectos, cada extremidade da troca fica livre para conceber o objecto conforme necessário, desde que essa troca esteja de acordo com o formato do documento XML. Uma das razões para não utilizar a numeração em série de objectos está relacionada com o suporte de implementações do cliente e do servidor de um objecto. Muitos dos actuais esquemas XML especializados para uma determinada indústria são concebidos como arquitecturas cliente-servidor nas quais o processamento efectuado no cliente está separado do processamento pretendido no servidor. Como sucede muitas vezes, o cliente só está a pedir ou a guardar informações num determinado formato de documento que continua no servidor.

## Arquitectura dos serviços da Web do DB2 Content Manager

Os principais componentes da arquitectura dos serviços da Web do DB2 Content Manager incluem o solicitador, o servidor de serviços da Web, o nível de beans XML e o repositório do DB2 Content Manager.

Segue-se o fluxo geral de interações dos principais componentes dos serviços da Web:

1. O solicitador efectua uma chamada ao servidor de serviços da Web.
2. O servidor de serviços da Web do DB2 Content Manager analisa e extrai a mensagem XML do envelope SOAP e extrai os dados binários que são carregados dos anexos SOAP.
3. A mensagem XML é enviada para o nível de beans XML do DB2 Content Manager.
4. Os beans XML transformam o XML em várias chamadas para as APIs do DB2 Content Manager subjacentes.
5. As APIs acedem aos dados no repositório e devolvem valores para os beans XML.
6. Os valores devolvidos das APIs são transformados numa mensagem de resposta XML pelo bean XML. Esta mensagem contém o estado do pedido, os dados da resposta e informações de excepção (se aplicável). Se as APIs devolverem qualquer binário, não são incluídas na mensagem, mas são indicadas em separado.
7. A mensagem é devolvida para o servidor de serviços da Web do DB2 Content Manager.
8. O servidor de serviços da Web cria uma mensagem SOAP.
9. A mensagem SOAP é devolvida ao solicitador.

A Figura 31 na página 429 descreve os passos envolvidos no processamento do documento utilizando os serviços da Web. É enviado um pedido SOAP para o servidor de serviços da Web para armazenar uma participação de seguros. O intermediário dos serviços da Web, que inclui o servidor de serviços da Web, a servlet de serviços da Web, beans XML e APIs processa o pedido SOAP e envia os dados para o servidor de bibliotecas. A participação é armazenada no servidor de bibliotecas e as imagens associadas à participação são armazenadas no gestor de recursos.

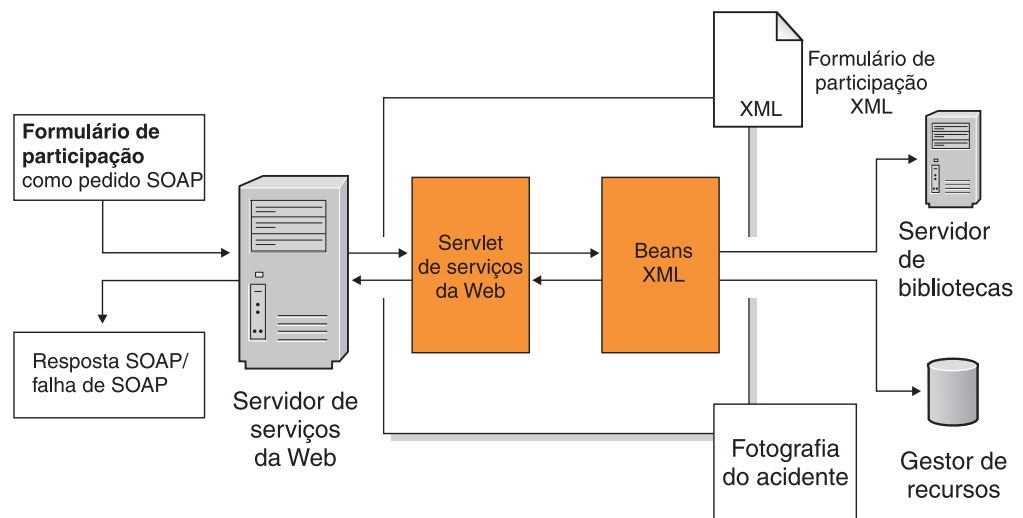


Figura 31. Processar um documento com serviços da Web

## Comparação das interfaces de serviços da Web do DB2 Content Manager

O DB2 Content Manager disponibiliza a mesma funcionalidade de serviços da Web através de duas interfaces diferentes: CMBGenericWebService e CMWebService, de acordo com os diferentes requisitos para a integração dos serviços da Web do DB2 Content Manager em aplicações de clientes.

As interfaces do CMWebService definem detalhadamente todas as operações na WSDL. A WSDL também faz referência a esquemas XML que definem as mensagens de pedido e de resposta para cada operação e os elementos de construção para estas mensagens. Quando a WSDL e os esquemas XML são convertidos em classes proxy de um ambiente de programação específico, estas podem fornecer APIs para ajudar a criar mensagens de pedidos e a decompor as mensagens de respostas para aplicações cliente. Por conseguinte, os programadores do cliente não necessitam de agregar ou desagregar as mensagens XML manualmente. As APIs de classe proxy também têm tipificação forte, o que evita erros que podem ocorrer durante a agregação ou desagregação da mensagem XML manualmente. Isto é denominado de estilo de gerador de uma interface de serviços da Web.

Opcionalmente, pode utilizar o esquema XML para que tipos de artigos especificados do DB2 Content Manager gerem classes proxy de cliente. Neste caso, as APIs de classe proxy são detalhadas de tal modo que os programadores das aplicações cliente não necessitam de ter informações sobre a sintaxe das mensagens XML. Os programadores podem seguir as directrizes da programação orientada para objectos de modo a utilizarem estas classes proxy para agrupar mensagens XML de pedidos e desagrupar mensagens XML de respostas que conduz a um rápido desenvolvimento nas aplicações cliente. Para além disso, também significa que se os tipos de artigos forem alterados, adicionados ou removidos da definição, os programadores têm de obter o novo esquema XML para os tipos de artigos, regenerar as classes proxy e compilar novamente a aplicação com as novas classes.

Contudo, se o esquema XML para os tipos de artigos específicos não for incluído ao gerar as classes proxy com WSDL de CMWebServices, os programadores da

aplicação cliente terão de escrever o respectivo código para ordenar os elementos XML para os tipos de artigos nas mensagens de pedido e converter os elementos XML para os tipos de artigos nas mensagens de resposta. A vantagem de não incluir os tipos de artigos específicos nas classes proxy é o facto das aplicações cliente terem a flexibilidade para trabalhar com qualquer tipo de artigo.

A WSDL de CMBGenericWebServices define o estilo genérico de interfaces de serviços da Web. Não define as operações dos serviços da Web nem a mensagem de pedido e de resposta. Em vez disso, faculta uma API genérica de modo a ser possível enviar qualquer forma de mensagens XML de pedido para o servidor de serviços da Web do DB2 Content Manager e receber qualquer forma de mensagens XML de resposta a partir do servidor. Este estilo genérico ou de associação tardia de interfaces de serviços da Web concede a liberdade ao servidor de serviços da Web do DB2 Content Manager para adicionar, alterar ou remover as operações dos serviços da Web e respectivos parâmetros no futuro, mas os programadores da aplicação cliente não são obrigados a obter a nova WSDL e a compilar novamente a aplicação.

Contudo, devido a esta flexibilidade da interface CMBGenericWebServices, os programadores da aplicação cliente terão de conhecimento completo das operações de serviços da Web e das mensagens XML de pedido e de resposta. Terão de agrupar ou desagrupar a mensagem manualmente. Podem utilizar interfaces como, por exemplo, a WSDL para CMBGenericWebServices, o esquema XML para mensagens definido no ficheiro cmbmessages.xsd, o esquema XML para elementos definido no ficheiro cmdatamodel.xsd para a construção de XMLs de tipos de artigo (ambos os ficheiros de definição de esquema XML encontram-se em *IBMCMROOT/config*, em que *IBMCMROOT* corresponde ao directório onde o DB2 Content Manager Information Integrator for Content está instalado) e os exemplo de serviços da Web que são fornecidos com o DB2 Content Manager Information Integrator for Content.

No que diz respeito ao estilo de mensagens SOAP, tanto o CMWebService como o CMBGenericWebService utilizam documento/literal, o que significa que o pedido e a resposta são mensagens de estilo documento e não são codificadas.

Para uma aplicação específica, tenha em consideração a capacidade do anexo SOAP ao escolher o terminal de serviços da Web.

Os tipos de anexo suportados pelo CMWebServices são:

- DIME (Direct Internet Message Encapsulation)
- MIME (Multipurpose Internet Mail Extensions)
- MTOM (Message Transmission Optimization Mechanism)

Em tempo de execução, apenas um dos três tipos de formato de anexos SOAP é activado pela configuração. O tipo de anexo suportado por CMBGenericWebServices é MIME.

Muitas das ferramentas destinadas ao desenvolvimento de serviços da Web geram código de cliente, por vezes denominadas classes proxy, para representar a interface e as operações de serviços da Web suportadas pelo serviço da Web. Este facto permite ocultar os detalhes da construção de mensagens SOAP, anexos, programação do socket, entre outros, de modo a que o cliente tenha apenas de construir e enviar pedidos para uma versão de proxy local do serviço da Web.

O servidor de serviços da Web do DB2 Content Manager responde a qualquer mensagem de pedido que esteja em conformidade com as interfaces e as mensagens de resposta que são enviadas do mesmo são igualmente baseadas nas interfaces. Os programadores da aplicação cliente decidem como gerar e compreender as mensagens, incluindo a selecção dos terminais de serviços da Web a visitar e o ambiente de desenvolvimento e a linguagem de programação a utilizar. Contudo, cada uma das diferentes ferramentas de desenvolvimento de cliente tem pontos fortes e limitações. A escolha é baseada no tipo de anexo:

- Microsoft Visual Studio .NET 2003, que utiliza a .NET Framework SDK versão 1.1 e Web Services Enhancements (WSE) SDK versão 1.0 sp1 suporta apenas os anexos DIME
- Microsoft Visual Studio 2005 que utiliza Microsoft .NET Framework SDK versão 2.0 ou 3.0 e Web Services Enhancements (WSE) SDK versão 3.0 suporta apenas MTOM
- O Rational Application Developer versão 6.0 e as ferramentas do cliente WebSphere, suportam apenas MIME

Entre o software de serviços da Web não IBM, o Apache Axis suporta DIME e MIME. O Apache Axis2 suporta MIME e MTOM. Por conseguinte, dependendo do ambiente de desenvolvimento do cliente, terá de escolher o terminal de serviços da Web correspondente (CMWebService ou CMBGenericWebService) para estabelecer ligação ao mesmo com base na capacidade da mensagem SOAP.

Para ver as diferenças de códigos entre as duas interfaces, consulte os exemplos de programas incluídos no IBM DB2 Information Integrator for Content.

#### **Conceitos relacionados**

“Exemplos de serviços da Web” na página 464

É facultado um conjunto de exemplos de serviços da Web com comentários precisos para ajudar o utilizador a trabalhar com os serviços da Web.

#### **Tarefas relacionadas**

“Gerar o ficheiro WSDL para o serviço da Web CMWebService a partir do cliente de administração do sistema” na página 446

A exportação de tipos de artigos para WSDL ainda é suportada no cliente de administração do sistema DB2 Content Manager Versão 8.4. Gera o ficheiro WSDL e os ficheiros de esquema de suporte para um grupo de tipos de artigos.

## **Funções dos serviços da Web**

Os serviços da Web não suportam o conjunto completo das operações do DB2 Content Manager.

As funções suportadas pelos serviços da Web incluem as funções de tempo de execução utilizadas mais frequentemente, que não incluem a criação de modelos de dados e de administração. A seguinte lista contém as operações suportadas pelos serviços da Web:

- Criar, obter, actualizar e eliminar artigos
- Consultar
- Operações relacionadas com pastas
- Operações relacionadas com ligações
- Dar entrada e dar saída de artigos
- Alterar palavra-passe
- Apresentar lista dos privilégios do utilizador para um artigo
- Apresentar lista dos servidores disponíveis (servidores da biblioteca)

- Apresentar lista de tipos e esquemas de artigos: Obter uma definição de esquema XML para um ou para todos os tipos de artigos.
- Mover um artigo: Alterar os metadados de um artigo.
- Operações de encaminhamento de documentos
- Lote: Agrupar várias operações num único pedido (um tempo de ida e volta para o servidor)
- Transacções: Agrupar vários pedidos incluídos num pedido de lote numa única transacção com alterações removidas em caso de erro

#### **Referências relacionadas**

“Lista de pedidos de serviços da Web”

Nos serviços da Web, as operações que permitem ao utilizador ligar, criar, consultar, obter e modificar dados no sistema têm a forma de pedidos.

## **Lista de pedidos de serviços da Web**

Nos serviços da Web, as operações que permitem ao utilizador ligar, criar, consultar, obter e modificar dados no sistema têm a forma de pedidos.

Pode criar pedidos de serviços da Web de acordo com o esquema definido nos ficheiros `cmbmessages.xsd` e `cmdatamodel.xsd`. Pode encontrar o pedido no esquema para pesquisar o nome do pedido. O esquema para este pedido descreve os parâmetros que a operação requer. O ficheiro está localizado no directório `IBMCMROOT\config`.

Pode executar os seguintes pedidos através dos serviços da Web:

#### **AddItemToFolderRequest**

Adiciona um artigo existente a uma pasta existente.

#### **BatchRequest**

Permite agrupar pedidos independentes num único pedido, efectuado numa deslocação ao servidor. Os pedidos são efectuados sequencialmente no servidor e as respostas individuais para cada pedido são agrupadas numa resposta de lote.

O pedido de lote pode incluir vários tipos de pedidos diferentes, várias instâncias do mesmo tipo de pedido ou qualquer combinação. Pode criar um pedido de lote 10 `RetrieveItemRequests` para 10 artigos diferentes (uma obtenção num só volume) ou pode criar um pedido de lote com um `RetrieveItemRequest`, seguido de um `UpdateItemRequest`, seguido de uma consulta, em seguida, um `ListWorkNodesRequest`, em seguida, mais cinco `RetrieveItemRequests`, e assim sucessivamente.

#### **ChangePasswordRequest**

Permite alterar a palavra-passe de um utilizador.

#### **CheckInItemRequest**

Permite dar entrada de um artigo que se encontra actualmente com saída dada.

#### **CheckOutItemRequest**

Permite dar saída de um artigo do servidor.

#### **CreateItemRequest**

Permite criar um novo artigo no servidor. Terá de transmitir um documento XML de acordo com o esquema para o tipo de artigo especificado e os dados que passarão a ser o conteúdo do novo artigo terão de ser transmitidos ao pedido como anexos.



**CreateLinkRequest**

Cria uma ligação entre dois artigos existentes.

**DeleteLinkRequest**

Elimina uma ligação entre dois artigos.

**DeleteItemRequest**

Elimina um artigo do servidor.

**GetPrivilegesRequest**

Devolve uma lista de privilégios de acesso que o utilizador actual tem para um artigo específico.

**ListSchemaRequest**

Devolve uma lista dos tipos de artigos (entidades) que estão disponíveis a um utilizador e inclui o esquema XML para esse tipo de artigo como um anexo na resposta. Pode determinar um ou mais tipos de artigos específicos cujos esquemas deverão ser devolvidos ou pode obter os esquemas para todos os tipos de artigos especificados pelo utilizador no pedido ao qual tem acesso.

**ListServersRequest**

Devolve uma lista de todos os servidores de um tipo específico como, por exemplo, ICM, que estão disponíveis para os utilizadores estabelecerem ligação. Este é o único pedido que não requer qualquer autenticação do utilizador, uma vez que não é necessário estabelecer ligação para obter esta informação.

**MoveItemRequest**

Move ou volta a colocar no índice um artigo de um tipo de artigo para outro. É necessário fornecer os metadados para o novo artigo como um fragmento de um documento XML, de acordo com o esquema do novo tipo de artigo.

**RemoveItemFromFolderRequest**

Remove um artigo existente de uma pasta no servidor.

**RetrieveFolderItemsRequest**

Devolve uma lista de artigos incluídos numa pasta. Pode optar por devolver apenas os IDs de documento dos artigos, os respectivos metadados ou todos os dados, incluindo o conteúdo.

**RetrieveFoldersForItemRequest**

Devolve uma lista de IDs de documento de todas as pastas que contêm o artigo indicado.

**RetrieveItemRequest**

Devolve o conteúdo de um artigo. Pode optar por obter apenas os metadados para o artigo, o conteúdo para o artigo e respectivos metadados ou por obter as ligações, o conteúdo e os metadados.

**RunQueryRequest**

Executa uma cadeia de consulta indicada no pedido e devolve os resultados da consulta. Pode optar por obter apenas uma lista dos IDs de documento na resposta ou pode optar por obter qualquer nível de conteúdo a partir dos metadados, dos metadados e do conteúdo, do conteúdo ou das ligações.

**UpdateItemRequest**

Actualiza um artigo. Pode optar por adicionar e remover componentes descendentes, registos de notas, anotações ou partes de um documento. Também pode substituir os valores dos metadados do artigo, especificando



novos valores para o artigo actualizado num fragmento de um documento XML, de acordo com o esquema para o tipo de artigo e substituir componentes descendentes, registos de notas, anotações e partes existentes. O ID de documento para o artigo é devolvido na resposta, uma vez que, em alguns casos, a actualização do artigo altera o ID de documento.

## **pedidos de encaminhamento de documentos**

A seguinte lista apresenta uma descrição dos pedidos de encaminhamento de documentos:

### **ContinueProcessRequest**

Move o pacote de trabalho especificado para o nó de trabalho seguinte. O pacote de trabalho é removido do servidor e é criado um novo pacote de trabalho ou vários (em caso de encaminhamento paralelo) para o proprietário do pacote de trabalho especificado.

### **ListActionRequest**

Apresenta a acção para um nome de acção especificado.

### **ListActionNamesRequest**

Apresenta os nomes de acção para uma lista de acções especificada.

### **ListNextWorkPackageRequest**

Devolve o pacote de trabalho seguinte na lista de trabalho especificada para o proprietário especificado dá saída do artigo indicado. A resposta disponibiliza o pacote de trabalho seguinte, no qual não foi dada saída do artigo indicado no pacote de trabalho por outro utilizador.

### **ListProcessRequest**

Apresenta o processo para o nome do processo indicado.

### **ListProcessesRequest**

Apresenta uma lista de todos os processos.

### **ListProcessNamesRequest**

Apresenta uma lista dos nomes de processos.

### **ListWorkListNamesRequest**

Apresenta uma lista dos nomes da lista de trabalho.

### **ListWorkListsRequest**

Apresenta as listas de trabalho.

### **ListWorkNodeRequest**

Apresenta o nó de trabalho especificado pelo nome do nó de trabalho.

### **ListWorkNodesRequest**

Apresenta uma lista dos nós de trabalho.

### **ListWorkPackageCheckOutOptionRequest**

Apresenta uma lista dos atributos do pacote de trabalho para o URI do pacote de trabalho indicado. É possível dar saída do artigo associado ao pacote de trabalho, dependendo do valor especificado para o elemento checkOutItem.

### **ListWorkPackagesRequest**

Apresenta uma lista de todos os pacotes de trabalho para o nome da lista de trabalho e para o proprietário do pacote de trabalho especificados.

### **ResumeProcessRequest**

Retoma um processo. Quando um processo é retomado, o atributo do

pacote de trabalho suspend flag é definido como false. Não existe encaminhamento do pacote de trabalho e não é dada saída do artigo associado ao pacote de trabalho.

#### **SuspendProcessRequest**

Suspende um processo com o pacote de trabalho especificado durante o número de minutos especificado no pedido para a lista de retoma indicada.

#### **StartProcessRequest**

Inicia um artigo num processo de encaminhamento de documentos.

#### **TerminateProcessRequest**

Termina um processo. O pacote de trabalho especificado é removido do servidor. É dada entrada do artigo indicado no pacote de trabalho, caso tenha sido dada saída do mesmo.

#### **UpdateWorkPackageRequest**

Atualiza os dados de contentor do pacote de trabalho, o proprietário do pacote de trabalho e a prioridade do pacote de trabalho.

#### **Conceitos relacionados**

“Exemplos de serviços da Web” na página 464

É facultado um conjunto de exemplos de serviços da Web com comentários precisos para ajudar o utilizador a trabalhar com os serviços da Web.

---

## **Instalar e configurar os serviços da Web do DB2 Content Manager**

Os seguintes tópicos indicam os procedimentos necessários para a instalação e configuração dos serviços da Web DB2 Content Manager Versão 8.4.

### **Plataformas suportadas pelos serviços da Web**

Pode utilizar os serviços da Web em várias plataformas.

#### **Pré-requisitos de software do servidor de serviços da Web do DB2 Content Manager**

Base and Network Deployment do WebSphere Application Server Versão 6.1

**Importante:** O DB2 Content Manager V8.4 não suporta o WebSphere Application Server Versão 5.1.1 e o WebSphere Application Server Versão 6.0.2.

#### **Pré-requisitos de software de um cliente de serviços da Web do DB2 Content Manager 8.4**

##### **Clientes Java**

- As aplicações clientes de Java para o CMBGenericWebService têm de ser desenvolvidas com o WebSphere Application Server 6.1.
- As aplicações cliente de Java para o CMWebService com um tipo de anexo DIME ou MTOM têm de ser desenvolvidas com o Apache Axis versão 1.1 ou superior
- As aplicações cliente de Java para o CMWebService com o tipo de anexo MTOM têm de ser desenvolvidas com o Apache Axis2 versão 1.1.1

##### **Cliente .NET 1.1**

- Microsoft .NET Framework SDK 1.1
- Microsoft Web Services Enhancement (WSE) SDK 1.0 Service Pack 1
- Microsoft Hotfix 892202

Terá de contactar a Microsoft para obter a correcção dinâmica. Caso não tenha a correcção dinâmica instalada, irá receber erros ao gerar o WSDL.

#### **Cliente .NET 2.0**

- Microsoft® .NET Framework SDK 2.0 ou 3.0
- Microsoft Web Services Enhancement (WSE) SDK 3.0

**Importante:** Os serviços da Web para DB2 Content Manager Versão 8.4 não suportam WebSphere Application Server Network Deployment em Linux para system Z.

#### **Conceitos relacionados**

“Topologias do WebSphere Application Server suportadas” na página 437  
Esta secção disponibiliza uma breve descrição das topologias do WebSphere Application Server suportadas pelos serviços da Web do DB2 Content Manager.

## **Instalação e configuração de serviços da Web**

Ao efectuar a instalação, o programa de instalação copia os ficheiros de programa do suporte de instalação para a unidade de disco rígido local. Ao efectuar a configuração, o utilizador cria ficheiros de configuração e bases de dados necessários e implementa aplicações da Web em WebSphere Application Servers.

Existem várias diferenças significativas na instalação dos serviços da Web do DB2 Content Manager Versão 8.3 para o DB2 Content Manager Versão 8.4.

No DB2 Content Manager Versão 8.3, a instalação e configuração foi efectuada através da utilização de uma única interface de utilizador (GUI). Por exemplo:

1. O programa de instalação do DB2 Information Integrator for Content solicita ao utilizador a selecção dos componentes que pretende instalar (conectores, serviços da Web, infocenter, entre outros.).
2. O utilizador terá de introduzir os parâmetros de configuração para os componentes seleccionados.
3. O programa de instalação copia os ficheiros de programa dos componentes seleccionados para a unidade de disco rígido local.
4. Depois de os ficheiros serem copiados, o programa de instalação abre os programas de configuração para configurar os componentes seleccionados. Para os serviços da Web, a configuração implementa a aplicação de serviços da Web no WebSphere Application Server.

O DB2 Content Manager Versão 8.4 divide a instalação e a configuração em dois programas em separado. O programa de instalação copia apenas os ficheiros de programa do suporte de instalação para a unidade de disco rígido local. O programa de configuração executa tarefas de configuração para os componentes instalados. Pode abrir cada um dos programas em separado a quantidade de vezes necessária. Por exemplo, pode configurar um componente instalado várias vezes sem ser necessário instalar e desinstalar o mesmo. Também é possível desinstalar ou voltar a instalar um componente sem ser necessário desinstalar ou voltar a instalar o produto completo.

A instalação e configuração de serviços da Web utilizando duas GUIs em separado, proporciona flexibilidade, uma vez que é um dos componentes do produto DB2 Information Integrator for Content na versão 8.4.

Os serviços da Web dependem do conector ICM, que é um componente do DB2 Information Integrator for Content. Desta forma, o conector ICM tem de ser

instalado se optar por instalar os serviços da Web quando executar a instalação do DB2 Information Integrator for Content. O conector ICM também tem de ser configurado antes de configurar os serviços da Web.

## **Actualizar os serviços da Web da Versão 8.3 para a Versão 8.4**

O DB2 Content Manager Versão 8.4 suporta a actualização dos serviços da Web Versão 8.3 para a Versão 8.4. Para obter mais informações, consulte o tópico [Upgrading Web services to Version 8.4](#)

## **Topologias do WebSphere Application Server suportadas**

Esta secção disponibiliza uma breve descrição das topologias do WebSphere Application Server suportadas pelos serviços da Web do DB2 Content Manager.

### **Destino de implementação**

Os serviços da Web são um dos componentes do DB2 Information Integrator for Content. No DB2 Content Manager Versão 8.3, ao configurar os serviços da Web (implementar a aplicação de serviços da Web no WebSphere Application Server), o nó da aplicação tem de residir na mesma máquina na qual o DB2 Information Integrator for Content está instalado, o que significa que o destino de implementação (nó do servidor de aplicações) tem de ser local para o DB2 Information Integrator for Content. Por conseguinte, se o DB2 Information Integrator for Content estiver instalado numa máquina onde existe apenas um nó de gestor de implementação, não é possível implementar a aplicação de serviços da Web em nenhum nó do servidor de aplicações no ambiente WebSphere Network Deployment gerido por este gestor de implementação.

Contudo, os serviços da Web do DB2 Content Manager Versão 8.4 suportam o WebSphere Network Deployment. Por conseguinte, é possível implementar a aplicação de serviços da Web num nó do servidor de aplicações remoto, que não necessita de ser na máquina onde o DB2 Information Integrator for Content está instalado, caso exista um nó (um nó de gestor de implementação ou um nó do servidor de aplicações federado para o gestor de implementação) na máquina. A gestão do WebSphere Network Deployment é responsável pela transferência da aplicação para o nó de destino que pode estar localizado numa máquina remota. A aplicação de serviços da Web do DB2 Content Manager Versão 8.4 Web pode ser executada, dependendo apenas dos recursos locais. Não faz referência a qualquer informação incluída na máquina onde o DB2 Information Integrator for Content está instalado.

### **Topologias suportadas**

As seguintes topologias são suportadas pelo DB2 Content Manager Versão 8.4.

#### **Topologia A - Nó não gerido**

Um nó não gerido é uma ilha. É um nó do servidor de aplicações do WebSphere com funcionalidades de administração próprias. Nesta topologia, o DB2 Information Integrator for Content tem de ser instalado nesta máquina. O nó não gerido pode ser criado através da instalação de um pacote WebSphere Application Server ou de um pacote Network Deployment. Em relação ao último, o nó não é federado para o gestor de implementação.

#### **Topologia B - Nó gerido**

Um nó gerido é um nó do servidor de aplicações que já está federado para um gestor de implementação. Este nó pode ser local ou remoto em relação à máquina onde o DB2 Information Integrator for Content está instalado. Se for local, tem a mesma topologia de um nó não gerido. Se for remoto, a máquina onde o DB2 Information Integrator for Content está instalado também terá de ter um nó (um nó de gestor de implementação ou um nó do servidor de aplicações federado para o gestor de implementação). A máquina onde o DB2 Information Integrator for Content está instalado e a máquina onde se encontra o nó gerido têm de estar na mesma hierarquia do Network Deployment.

Em ambas as topologias, A e B, é necessário fornecer o nome de um servidor de aplicações para especificar um destino de implementação para além do nome do nó. Uma vez que o programa de configuração dos serviços da Web não cria um servidor designado, o servidor já terá de existir no momento da implementação da aplicação.

### **Topologia C - Conjunto de unidades**

Um conjunto de unidades do servidor é um grupo de servidores de aplicações em diferentes nós do servidor de aplicações, que partilham o carregamento de aplicações específicas. Esta topologia é semelhante ao cenário do nó gerido. A diferença é o facto do destino de implementação consistir em vários nós do servidor de aplicações. Estes nós estão agrupados num conjunto de unidades. O nome do conjunto de unidades é utilizado como o destino de implementação. O gestor de implementação implementa automaticamente as aplicações dos serviços da Web em cada um dos servidores membro no conjunto de unidades quando o nome do mesmo é utilizado.

O utilizador terá de ter capacidade para aceder ao conjunto de unidades a partir da máquina onde o DB2 Information Integrator for Content está instalado. A máquina na qual o DB2 Information Integrator for Content está instalado também terá de conter um nó (um nó de gestor de implementação ou um nó do servidor de aplicações federado para o gestor de implementação).

A actualização dos serviços da Web a partir do DB2 Content Manager Versão 8.3 para a Versão 8.4 não suporta conjuntos de unidades. Para obter mais informações, consulte o tópico *Upgrading Web services to Version 8.4*.

**Importante:** Os serviços da Web para DB2 Content Manager Versão 8.4 não suportam WebSphere Application Server Network Deployment em Linux para system Z.

## **Verificar, iniciar e parar serviços da Web**

Após a instalação dos serviços da Web, terá de verificar se foram instalados correctamente, iniciar e pará-los antes de ser possível trabalhar com os mesmos.

Para iniciar a aplicação de serviços da Web, terá de iniciar o WebSphere Application Server ou o conjunto de unidades no qual a aplicação está implementada.

A aplicação de serviços da Web pára se parar o WebSphere Application Server ou o conjunto de unidades no qual a aplicação está implementada.

Para verificar se a aplicação de serviços da Web foi instalada correctamente, terá de iniciar a aplicação em primeiro lugar. Após o início da aplicação, estabeleça ligação aos seguintes URLs num navegador da Web.

- <http://hostname/CMBSpecificWebService/services/CMWebService>
- <http://hostname/CMBGenericWebService/services/CMBGenericWebService>

Deverá visualizar texto genérico que não é uma mensagem de erro.

**Restrição:** Caso não tenha o HTTP Server instalado, terá de adicionar a porta utilizada pelo WebSphere Application Server ao nome do sistema central. Por exemplo: <http://hostname:9082/CMBGenericWebService/services/CMWebService>.

-  Iniciar um servidor de aplicações
-  Parar um servidor de aplicações
-  Iniciar conjuntos de unidades
-  Parar conjuntos de unidades

## Configuração pós-instalação

Após a instalação e implementação dos serviços da Web no WebSphere Application Server, é necessário executar as seguintes tarefas antes de ficar funcional.

### Configurar a biblioteca partilhada para o controlador JDBC

Os serviços da Web do DB2 Content Manager Versão 8.4 disponibilizam diferentes soluções para diferentes tipos de configurações do controlador JDBC.

Os serviços da Web do DB2 Content Manager invoca APIs do conector ICM. A API do conector ICM depende do controlador JDBC para estabelecer ligação às bases de dados do servidor de bibliotecas. Relativamente aos serviços da Web do DB2 Content Manager Versão 8.4 para várias plataformas com o DB2, o controlador JDBC é enviado com o Information Integrator for Content. Por conseguinte, os ficheiros JAR do controlador JDBC são incluídos no ficheiro EAR dos serviços da Web antes de ser implementado no WebSphere Application Server. Após a implementação, os ficheiros JAR do controlador JDBC estão localizados no directório:

`{WAS_HOME}/profiles/{profileName}/installedApps/{cellName}/{appName}/lib`, em que *WAS\_HOME* corresponde ao directório inicial do IBM WebSphere Application Server, *profileName* corresponde ao nome do directório do perfil onde a aplicação de serviços da Web está instalada, *cellName* corresponde ao nome da célula WebSphere onde a aplicação de serviços da Web está instalada e *appName* corresponde ao nome da aplicação WebSphere especificado para a aplicação de serviços da Web.

Este directório está localizado no caminho da classe da aplicação de serviços da Web em tempo de execução. As APIs do conector ICM podem localizar o JDBC no caminho da classe.

Relativamente aos serviços da Web do DB2 Content Manager V8.4 for zOS com o DB2 Universal Database, os ficheiros JAR do controlador JDBC não são incluídos no ficheiro EAR dos serviços da Web. O programa de configuração de serviços da Web cria as seguintes bibliotecas partilhadas após a implementação da aplicação:



**sharedLibrary\_1\_name=DB2 JDBC**

```
sharedLibrary_1_classpath=${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar;
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar;
${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cisuz.jar
sharedLibrary_1_nativepath=${DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH}
```

**sharedLibrary\_2\_name=DB2 JDBC2**

```
sharedLibrary_2_classpath=${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar;
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar;
${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cisuz.jar
sharedLibrary_2_nativepath=${DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH}
```

As bibliotecas partilhadas utilizam as duas seguintes variáveis WebSphere existentes nos respectivos caminhos:

- *DB2UNIVERSAL\_JDBC\_DRIVER\_PATH*
- *DB2UNIVERSAL\_JDBC\_DRIVER\_NATIVEPATH*

O administrador da aplicação de serviços da Web do DB2 Content Manager V8.4 terá de garantir que o valor destas duas variáveis está definido nos directórios onde o controlador JDBC se encontra actualmente instalado.

Os serviços da Web do

**Actualizar ficheiros de configuração**

Ao configurar os serviços da Web, tem de copiar os ficheiros para as localizações em que a aplicação dos serviços da Web foi implementada.

Os seguintes ficheiros são copiados para as localizações nas quais a aplicação de serviços da Web é implementada:

- *IBMCMROOT/cmgmt/cmbcmenv.properties*
- *IBMCMWorkingDirectory/cmgmt/cmbxmlservices.properties*
- *IBMCMWorkingDirectory/cmgmt/cmbwebservices.properties*
- *IBMCMWorkingDirectory/cmgmt/cmbconnectionpool.properties*
- *IBMCMWorkingDirectory/cmgmt/cmbcmenv.properties*
- ficheiros de configuração INI para o conector DB2 Content Manager que estão no directório especificado pelo ficheiro de propriedade CMCFGDIR no ficheiro *IBMCMROOT/cmgmt/cmbcmenv.properties*

em que *IBMCMROOT* corresponde ao directório de instalação de IBM DB2 Information Integrator for Content e *IBMCMWorkingDirectory* corresponde à propriedade definida no ficheiro *IBMCMROOT/cmgmt/ibmcmconfig.properties*.

Cada servidor de aplicações WebSphere no qual a aplicação de serviços da Web é implementada inclui uma cópia destes ficheiros, com a modificação para o ambiente de execução específico no servidor de aplicações.

Para sincronizar estas cópias com o conteúdo dos originais na lista acima, tem de utilizar o assistente de configuração do IBM DB2 Information Integrator for Content para configurar os serviços da Web novamente.

**Não se esqueça:** Tem de introduzir exactamente os mesmos parâmetros utilizados na última vez que configurou os serviços da Web.

## Configurar o conjunto de ligações do WebSphere Application Server para serviços da Web

A colecção de ligações do WebSphere Application Server permite que um datastore seja escalado para um número indefinido de utilizadores. A colecção de ligações também melhora o desempenho dos serviços da Web através da redução do tempo de início de sessão e da utilização da memória.

### Importante:

Para serviços da Web, o ficheiro cmbpool.ini está localizado em `${WAS_HOME}/profiles/${profileName}/installedApps/${cellName}/${appName}/cmgmt/connectors` e não no directório `/home/ibmcmadm/cmgmt/connectors` para AIX, Linux ou Solaris, ou no directório `IBMCMROOT\cmgmt\connectors` para Windows.

## Registar e rastrear

No DB2 Content Manager Versão 8.4, a aplicação de serviços da Web contém um ficheiro de registo próprio em cada um dos nós do servidor da aplicação.

A aplicação de serviços da Web do DB2 Content Manager Versão 8.3 escreve as respectivas mensagens de registo num registo beans, que corresponde a `IBMCMROOT/log/beans/ ${user.name}.beans.log`.

Na Versão 8.4, a aplicação de serviços da Web contém um registo de ficheiro próprio em cada um dos nós do servidor da aplicação. Os ficheiros de registo estão em `${SERVER_LOG_ROOT}/${appName}/websvc`, em que `SERVER_LOG_ROOT` corresponde a uma variável WebSphere que é definida no âmbito de servidor de aplicações e `${appName}` corresponde ao nome da aplicação escolhido ao implementar a aplicação de serviços da Web. O valor predefinido de `SERVER_LOG_ROOT` é `${WAS_HOME}/profiles/${profileName}/logs/${serverName}`, em que `serverName` corresponde ao nome do servidor de aplicações no qual a aplicação de serviços da Web é implementada.

Os ficheiros de registo de beans estão em `${SERVER_LOG_ROOT}/${appName}/beans..`

Os ficheiros de registo OOAPI estão em `${SERVER_LOG_ROOT}/${appName}/connectors..`

Estes directórios são configurados em `${WAS_HOME}/profiles/${profileName}/installedApps/${cellName}/${appName}/cmgmt/connectors/cmblogconfig.properties`.

Além de `cmblogconfig.properties`, `${WAS_HOME}/profiles/${profileName}/installedApps/${cellName}/${appName}/cmgmt/cmbwebservices.properties` inclui duas propriedades que controlam o registo e o rastreio apenas na camada de serviços da Web. As propriedades são:

### **messageTracingEnabled**

Para depurar o cliente de serviços da Web. O valor predefinido de `messageTracingEnabled` é `false`.

**Importante:** Se esta opção for `true`, afecta o desempenho dos serviços da Web. Se esta opção for definida para `true` e se o nível de rastreio do componentes de serviços da Web do ficheiro de registo comum `cmblogconfig.properties` estiver definido para



DEBUG, as mensagens de camada de serviços da Web são registadas no ficheiro de registo de serviços da Web.

#### **prependLoggerNameToMessage**

Determina se o nome de classe é adicionado antes da mensagem de registo no ficheiro de registo de serviços da Web. O valor predefinido de `prependLoggerNameToMessage` corresponde a `true`.

## **Desinstalar e remover a configuração**

Pode desinstalar os serviços da Web através da utilização da aplicação de instalação do DB2 Information Integrator for Content. Pode remover a configuração dos serviços da Web através da aplicação de configuração do DB2 Information Integrator for Content.

A remoção da configuração dos serviços da Web faz com que a aplicação empresarial de serviços da Web seja desinstalada do WebSphere Application Server Versão 6.1. Os ficheiros que não foram modificados no directório no qual a aplicação empresarial de serviços da Web foi instalada serão removidos. Se um ficheiro for modificado desde a respectiva implementação no WebSphere Application Server, o ficheiro não será removido, por exemplo, os ficheiros de configuração e os ficheiros de registo.

Os ficheiros de configuração presentes no directório no qual a aplicação de serviços da Web foi instalada são cópias dos ficheiros presentes em *IBMCMROOT/cmgmt* e *IBMCMWORKINGDIRECTORY/cmgmt*, em que *IBMCMROOT* corresponde ao directório do DB2 Information Integrator for Content e *IBMCMWORKINGDIRECTORY* definidos no ficheiro *IBMCMROOT/cmgmt/ibmcmconfig.properties*. Se alterar a configuração de forma a que exerça impacto no comportamento da aplicação de serviços da Web, terá de alterar os ficheiros em *IBMCMROOT/cmgmt* e *IBMCMWORKINGDIRECTORY/cmgmt* e utilizar uma consola de administração do WebSphere para actualizar os ficheiros de configuração no directório no qual a aplicação de serviços da Web está instalada.

**Importante:** Não deverá alterar directamente os ficheiros de configuração no directório no qual a aplicação de serviços da Web está instalada, dado que todas as alterações que efectuar serão perdidas quando implementar novamente a aplicação de serviços da Web.

**Importante:** Tem de guardar uma cópia dos ficheiros de registo na aplicação de serviços da Web instalada no directório antes de desinstalar ou instalar novamente a aplicação de serviços da Web para que o WebSphere Application Server Versão 6.1 armazene as informações de depuração e rastreio da instalação anterior.

---

## **Integrar serviços da Web básicos nas aplicações ou processos**

Pode integrar a aplicação cliente ou o processo empresarial utilizando os serviços da Web.

Pode desenvolver aplicações cliente para interagir com as seguintes interfaces dos serviços da Web do DB2 Content Manager Versão 8 Edição 4:

- CMWebService
- CMBGenericWebService

## Integrar o CMWebService

Para integrar o CMWebService, terá de criar o ficheiro WSDL e os respectivos ficheiros de esquema XML de suporte, gerar classes proxy para WSDL e escrever a aplicação para utilizar essas classes proxy.

Quando utiliza o CMWebService, os principais passos que deverá executar são os seguintes:

1. Criar o ficheiro WSDL e os respectivos ficheiros de esquema XML a partir do cliente de administração do sistema, utilizando uma ferramenta de linha de comandos ou a partir do URL: `http://hostname/CMBSpecificWebService/services/CMWebService/CMWebService.wsdl`.  
O WSDL e os ficheiros de esquema XML contêm todas as operações disponíveis suportadas pelo serviço da Web e qualquer definição de artigos que pretenda incluir.
2. Gerar classes proxy para o ficheiro WSDL e os respectivos ficheiros de esquema XML.
3. Escrever o programa de aplicação de modo a utilizar as classes proxy para comunicarem com o CMWebService. O URL do CMWebService é `http://hostname/CMBSpecificWebService/services/CMWebService`.

Existem três exemplos de cliente de serviços da Web no DB2 Content Manager Versão 8.4 que demonstram a forma de desenvolver clientes de serviços da Web para serviços da Web específicos (CMWebService) em C# e Java.

O exemplo C# com .NET1.1 está localizado em *IBMCMROOT/samples/webservices/CMWebService/CS/WSE1/base*. O exemplo requer o suporte do serviço da Web e o utilitário WSDL facultado pelo Microsoft .NET 1.1 SDK, WSE 1.0 SP1 SDK e Visual Studio .NET 2003. Para obter mais informações, consulte o ficheiro *readme* de exemplo em *IBMCMROOT/samples/webservices/CMWebServiceClient.readme*.

O exemplo C# com .NET 2.0 está em *IBMCMROOT/samples/webservices/CMWebService/CS/WSE3/base*. Requer o suporte do serviço da Web e o utilitário WSDL facultado pelo Microsoft .NET 2.0 SDK, WSE 3.0 SDK e Visual Studio 2005. Para obter mais informações, consulte o ficheiro *readme* em *IBMCMROOT/samples/webservices/CMWebServiceClientWSE30.readme*.

O exemplo Java em *IBMCMROOT/samples/webservices/CMWebService/java/Axis/base* demonstra a forma de desenvolver um cliente Java para CMWebService. Requer JDK 1.5 e Apache Axis Versão 1.4. Para obter mais informações, consulte o ficheiro *readme* de exemplo em *IBMCMROOT/samples/webservices/CMWebServiceClientJava.readme*.

### Informações relacionadas



Como iniciar o CMWebService num ambiente .NET

## Gerar o ficheiro WSDL para o serviço da Web CMWebService

É possível gerar um ficheiro WSDL utilizando o cliente de administração do sistema ou a partir de uma linha de comandos. Contudo, é mais prático efectuar a transferência a partir do URL do terminal, com os parâmetros adequados na cadeia de consulta do URL do terminal CMWebService.

No DB2 Content Manager Versão 8.4, as interfaces dos serviços da Web CMWebService estão definidas nos seguintes ficheiros:

**CMWebService.wsdl**

Ficheiro de definição principal para descrição do terminal de serviços da Web específico (CMWebService). Define os tipos de portas (como, por exemplo, as operações), os nomes de mensagens de entrada e saída para cada tipo de porta e a ligação ao terminal de serviços da Web. O WSDL importa cmbmessages.xsd para as definições detalhadas das mensagens.

**cmbmessages.xsd**

Esquema da representação XML das mensagens de entrada e saída para as operações do terminal de serviços da Web específico (CMWebService). Importa o ficheiro cmdatamodel.xsd para definir os detalhes da representação XML do modelo de dados do DB2 Content Manager Versão 8.4 em mensagens. Opcionalmente, inclui o ficheiro itemtype.xsd se os tipos de artigo forem especificados ao gerar o ficheiro cmbmessages.xsd. Neste caso, o ficheiro cmbmessages.xsd assume que itemtype.xsd contém as declarações do esquema XML para estes tipos de artigo e o ficheiro cmbmessages.xsd faz referência a estas declarações.

**cmdatamodel.xsd**

Esquema para a representação XML para o modelo de dados do DB2 Content Manager Versão 8.4.

**itemtype.xsd**

Esquema para a representação XML dos tipos de artigos seleccionados. Importa o ficheiro cmdatamodel.xsd para definir a representação XML do modelo de dados do DB2 Content Manager Versão 8.4.

Pode descarregar os ficheiros a partir dos seguintes URLs URLs correctamente construídos do terminal de serviços da Web.

**CMWebService.wsdl**

`http://hostname/CMBSpecificWebService/services/CMWebService/CMWebService.wsdl`

**cmbmessages.xsd**

`http://hostname/CMBSpecificWebService/services/CMWebService/cmbmessages.xsd?itemtype=itemtype1,itemtype2,itemtype3&b=true`, em que

**itemtype(optional)**

Cadeia com nomes de tipos de artigo concatenados, separados por uma vírgula. Se o parâmetro for apresentado na cadeia de consulta URL, o ficheiro cmbmessages.xsd descarregado contém um elemento `xs:include` cujo atributo `schemaLocation` contém um URL relativo que aponta para o ficheiro itemtype.xsd. Assume que o ficheiro itemtype.xsd contém as declarações de esquema XML dos tipos de artigos cujos nomes estão especificados no parâmetro e faz referência a estes nomes de tipos de artigos. Os nomes dos tipos de artigos têm de ser abreviaturas destes tipos de artigos, em vez dos respectivos nomes de apresentação.

Se o parâmetro não estiver na cadeia de consulta, o ficheiro cmbmessages.xsd não inclui o ficheiro itemtype.xsd e não faz referência a qualquer nome de tipo de artigo.

Não é possível validar o valor do parâmetro. Defina o mesmo com os nomes dos tipos de artigos correctos, de modo a ser possível fazer referência correctamente às declarações do esquema XML no ficheiro itemtype.xsd.

**b(optional)**

Parâmetro para a retrocompatibilidade de utilizadores que tenham as aplicações cliente de serviços da Web ligadas ao terminal CMWebService, que foram desenvolvidas com o DB2 Content Manager Versão 8.3 Fix Pack 4 ou anterior. Sem ser necessário efectuar qualquer alteração, estas aplicações cliente podem funcionar correctamente com o servidor de serviços da Web do DB2 Content Manager 8.4. W. Se pretender gerar novamente as classes proxy utilizando novos ficheiros WSDL e de esquema descarregados a partir do servidor de serviços da Web do DB2 Content Manager 8.4 e não pretender alterar as outras partes nas respectivas aplicações que utilizam as classes proxy, pode descarregar o ficheiro cmbmessages.xsd, utilizando o parâmetro com o valor true. Se o parâmetro não estiver definido ou o respectivo valor não é true, o ficheiro cmbmessages.xsd descarregado gera classes proxy incompatíveis para as aplicações cliente acima mencionadas.

**cmdatamodel.xsd**

<http://hostname/CMBSpecificWebService/services/CMWebServices/cmdatamodel.xsd>

**itemtype.xsd**

<http://hostname/CMBSpecificWebService/services/CMWebService/itemtype.xsd?itemtype=itemtype1,itemtype2,itemtype3&server=ICMNLSDB&credential=true&b=true>, em que

**itemtype(required)**

Consulte a explicação para o URL para efectuar a transferência do ficheiro cmbmessages.xsd. Os serviços da Web vão extrair os tipos de artigos especificados e guardar as respectivas declarações de esquema XML no ficheiro itemtype.xsd.

**server(required)**

Servidor ICM a partir do qual os serviços da Web vão extrair os tipos de artigos.

**b(optional)**

Consulte a explicação para o URL para efectuar a transferência do ficheiro cmbmessages.xsd. Se pretender gerar novamente as classes proxy utilizando novos ficheiros WSDL e de esquema descarregados a partir do servidor de serviços da Web do DB2 Content Manager 8.4 e não pretender alterar as outras partes nas respectivas aplicações que utilizam as classes proxy, é possível descarregar o ficheiro itemtype.xsd, utilizando o parâmetro com o valor true. Se o parâmetro não estiver definido ou o respectivo valor não é true, o ficheiro itemtype.xsd descarregado gera classes proxy incompatíveis para as aplicações cliente acima mencionadas.

**credential(optional)**

True indica que o pedido HTTP para o URL tem a credencial SSO (Single Sign-On - Início de Sessão Único).

**Autenticar para o arquivo de dados ICM**

Se o parâmetro **credential** estiver na cadeia da consulta e o valor do parâmetro for true, o servidor de serviços da Web procuram sempre um cookie chamado LtpaToken no pedido HTTP e tenta extrair uma cadeia de credencial. Se a cadeia

de credenciais existir, o servidor de serviços da Web utiliza a mesma para estabelecer ligação ao servidor ICM e, em seguida, extrai os tipos de artigos especificados.

Se o parâmetro **credential** não existir ou o valor do parâmetro não for true, o servidor de serviços da Web envia uma resposta HTTP de volta ao cliente HTTP com um código de estado 401 (falha na autenticação) e um cabeçalho WWW-Authenticate. O cliente HTTP terá de fornecer o ID de utilizador e a palavra-passe e codificar os mesmos no cabeçalho de autorização do pedido HTTP seguinte para o servidor de serviços da Web. Após a recepção do pedido HTTP, o servidor de serviços da Web descodifica o ID de utilizador e a palavra-passe a partir do cabeçalho de autorização e utiliza os mesmos para estabelecer ligação ao servidor ICM e, em seguida, extrai os tipos de artigos especificados.

**Importante:** O parâmetro **itemtype** nos URLs para efectuar a transferência dos ficheiros cmbmessages.xsd e itemtype.xsd tem de ser exactamente o mesmo. Caso contrário, não é possível ao ficheiro cmbmessages.xsd fazer correctamente referência às declarações do esquema XML para os tipos de artigos no ficheiro itemtype.xsd. Consequentemente, não é possível gerar as classes proxy.

Se o parâmetro **itemtype** não estiver incluído no URL para descarregar o ficheiro cmbmessages.xsd, o ficheiro cmbmessages.xsd não inclui o ficheiro itemtype.xsd. Neste caso, os ficheiros CMWebService.wsdl, cmbmessages.xsd e cmdatamodel.xsd são suficientes para criar as classes proxy para uma aplicação cliente que não esteja associada a tipos de artigos específicos. A aplicação pode funcionar com qualquer tipo de artigo. Contudo, os programadores da aplicação têm de escrever o seu próprio programa para a organização ou anulação da organização de qualquer representação XML para os tipos de artigos nas mensagens de entrada e de saída.

#### **Gerar o ficheiro WSDL para o serviço da Web CMWebService a partir do cliente de administração do sistema:**

A exportação de tipos de artigos para WSDL ainda é suportada no cliente de administração do sistema DB2 Content Manager Versão 8.4. Gera o ficheiro WSDL e os ficheiros de esquema de suporte para um grupo de tipos de artigos.

Para gerar o ficheiro WSDL e os respectivos ficheiros de esquema de suporte para um grupo de tipos de artigos utilizando um cliente de administração do sistema, execute estes passos:

1. No cliente de administração do sistema, seleccione os tipos de artigos.
2. Faça clique com o botão direito do rato, e seleccione **Export to WSDL file (Exportar para ficheiro WSDL)**.
3. Introduza o nome predefinido, CMWebServiceWSDL.zip. O directório predefinido é *IBMCMROOT/config*, em que *IBMCMROOT* corresponde ao directório inicial do DB2 Content Manager V8.4.
4. Altere o directório predefinido para guardar o ficheiro ZIP.
5. Faça clique em **Save (Guardar)**. O ficheiro CMWebServiceWSDL.zip é guardado na unidade de disco rígido local.

O ficheiro CMWebServiceWSDL.zip contém estes ficheiros:

- CMWebService.wsdl
- cmbmessages.xsd

- cmbdatamodel.xsd
- itemtype.xsd

As limitações de utilização do cliente de administração do sistema para exportar tipos de artigos para WSDL são:

- O ficheiro cmbmessages.xsd está sempre associado a tipos de artigos. Não é possível gerar um ficheiro cmbmessages.xsd sem as referências a qualquer tipo de artigo.
- A retrocompatibilidade suportada pelo parâmetro de cadeia de consulta **b** nos URLs para descarregar os ficheiros cmbmessages.xsd e itemtype.xsd não está disponível. Os ficheiros de esquema WSDL e XML em CMWebServiceWSDL.zip exportados a partir do cliente de administração do sistema gera as classes proxy incompatíveis com as aplicações cliente dos serviços da Web ligadas a CMWebService, que foram desenvolvidas com o Fix Pack 4 do DB2 Content Manager Versão 8.3 ou anterior.

#### **Referências relacionadas**

“Utilizar a propriedade forceItemNamespace ao desenvolver aplicações cliente que utilizem serviços da Web” na página 465

A propriedade forceItemNamespace permite a definição de um espaço de nomes para informações de esquema de tipos de artigo.

#### **Gerar o ficheiro WSDL para o serviço da Web CMWebService a partir de uma linha de comandos:**

Pode utilizar a ferramenta Java de linha de comandos, CMBGetSpecificWSDL para exportar o ficheiro CMWebServiceWSDL.zip para CMWebService. Para obter mais informações, consulte WEB SERVICES UTILITY - WSDL GENERATION TOOL . A utilização da ferramenta Java de linha de comandos constitui uma alternativa para gerar o ficheiro WSDL a partir do cliente de administração do sistema, mas contém um argumento de linha de comandos para suportar a retrocompatibilidade para as aplicações cliente de serviços da Web, utilizando o CMWebService, que foi desenvolvido com o DB2 Content Manager Versão 8.3 Fix Pack 4 ou anterior. Contudo, ainda não é possível à ferramenta Java gerar o ficheiro cmbmessages.xsd que não está associado a tipos de artigos especiais.

#### **Como iniciar o CMWebService num ambiente .NET**

Ao trabalhar com os serviços da Web num ambiente .NET, pode gerar o XML utilizando o Microsoft SOAP Toolkit para criar rotinas de nível baixo ou pode utilizar a ferramenta WSDL para gerar classes proxy.

Os toolkits, como o Microsoft SOAP Toolkit, facultam APIs de nível baixo para gerar e trocar mensagens SOAP. Estas APIs permitem especificar um documento XML que representa o corpo de uma mensagem SOAP, enviar o documento para o URL de serviço da Web e devolver o documento de resposta como parte da mensagem SOAP. É uma interacção de nível baixo com os serviços da Web. Este tipo de interacção permite maior flexibilidade, dado que a interface dos serviços da Web não muda, mesmo que o esquema XML seja alterado.

A desvantagem da utilização do serviço da Web desta forma é o facto de ser necessário gerar mensagens XML e utilizar APIs de nível baixo para enviar e receber mensagens SOAP. Neste caso, a WSDL é utilizada apenas para especificar o URL do ponto terminal do serviço da Web.

Para criar pedidos XML e SOAP, pode escrever um cliente de serviços da Web que utilize um utilitário WSDL. Um utilitário WSDL pode processar automaticamente



os pedidos e respostas XML e SOAP, representando-os como classes proxy. A Microsoft faculta um utilitário WSDL que pode representar documentos XML como classes proxy de C#.

Para escrever uma aplicação que utilize as classes geradas pelo utilitário WSDL .NET da Microsoft, execute os seguintes passos:

1. Certifique-se de que o seguinte software está instalado:

#### **Cliente .NET 1.1**

- .NET Framework SDK Versão 1.1 da Microsoft
- .NET Web Service Enhancements SP1 da Microsoft
- .NET Framework SDK, Versão 1, Edição 1, Hotfix 892202 da Microsoft

#### **Cliente .NET 2.0**

- .NET Framework SDK versão 2.0 ou 3.0
  - Web Services Enhancements SDK 3.0
2. Crie classes C# a partir dos ficheiros WSDL e de esquema.
  3. Utilizando os exemplo C# para orientação, programe o cliente da aplicação de serviços da Web através da utilização de classes proxy C#.

#### **Importante:**

Quando utiliza exemplos C#, assume-se que o servidor é local. No entanto, o servidor e os exemplos de serviços da Web podem estar instalados em diversas máquinas e, nesse caso, é necessário alterar o URL do servidor em conformidade.

Além disso, as classes proxy que são enviadas com os exemplos podem não corresponder aos dados exemplo armazenados no primeiro passo. Esta situação pode ocorrer se os dados exemplo foram importados com um nível de fix pack diferente do nível no qual os serviços da Web estão a ser executados. Nestes casos, o exemplo não vai funcionar como enviado e o utilizador terá de gerar uma nova WSDL e novas classes proxy.

#### **Conceitos relacionados**

“Programar pedidos de serviços da Web num ambiente .NET” na página 450

#### **Tarefas relacionadas**

“Gerar o ficheiro WSDL para o serviço da Web CMWebService a partir do cliente de administração do sistema” na página 446

A exportação de tipos de artigos para WSDL ainda é suportada no cliente de administração do sistema DB2 Content Manager Versão 8.4. Gera o ficheiro WSDL e os ficheiros de esquema de suporte para um grupo de tipos de artigos.

“Gerar classes proxy para o cliente .NET 1.1”

Depois de ter uma WSDL, é necessário transformar essa WSDL em classes proxy utilizáveis para WSDL.

“Gerar classes proxy para o cliente .NET 2.0” na página 449

### **Gerar classes proxy para o cliente .NET 1.1**

Depois de ter uma WSDL, é necessário transformar essa WSDL em classes proxy utilizáveis para WSDL.

Para gerar as classes proxy, é necessário instalar o seguinte software:

- .NET framework SDK, Versão 1, Edição 1
- .NET Web Services Enhancements, Versão 1, Service Pack 1

- .NET framework SDK, Versão 1, Edição 1, o Hotfix 892202 está instalado. Contacte a Microsoft para obter esta correcção dinâmica.

Para gerar classes proxy, execute estes passos:

1. Extraia o ficheiro CMWebService.zip para um directório temporário. É necessário extrair os seguintes ficheiros:
  - CMWebService.wsdl
  - cmbmessages.xsd
  - cmdatamodel.xsd
  - itemtype.xsd
2. Abra uma janela de linha de comandos e introduza o seguinte comando numa linha:

```
C:\VisualStudio2003\SDK\v1.1\Bin\wsdl.exe /language:CS
/out:CMWebService.cs file:///C:\CMWebServiceWSDL\CMWebService.wsdl
```

Pode executar este passo no Visual Studio .NET 2003, adicionando uma referência Web ao ficheiro:///C:\CMWebServiceWSDL\CMWebService.wsdl.

O comando assume que o .NET SDK v1.1 está instalado em C:\VisualStudio2003\SDK\v1.1 e o ficheiro CMWebService.zip é extraído em C:\CMWebServiceWSDL. O comando gera um ficheiro de classe C#, CMWebService.cs

Poderá receber mensagens de aviso durante a execução deste comando, mas pode ignorá-las. O ficheiro de classe proxy é gerado correctamente.

3. Modifique a classe proxy C#. Por predefinição, a classe proxy CMWebService.cs deriva de System.Web.Services.Protocols.SoapHttpClientProtocol. Para utilizar a funcionalidade de WSE 1.0 SDK, altere a categoria base para WebServicesClientProtocol:
  - a. Abra o ficheiro CMWebService.cs num editor. Adicione uma cláusula de espaço de nomes using. using Microsoft.Web.Services;
  - b. Substitua o código de declaração de classe no exemplo original pelo seguinte exemplo modificado:

#### Original

```
public class CMWebService :
 System.Web.Services.Protocols.SoapHttpClientProtocol {
```

#### Modificado

```
public class CMWebService :
 Microsoft.Web.Services.WebServicesClientProtocol {
```

- c. Guarde o ficheiro CMWebService.cs.

## Gerar classes proxy para o cliente .NET 2.0

Para gerar as classes proxy, é necessário instalar o seguinte software:

- .NET framework SDK, Versão 2.0
- .NET Web Services Enhancements, Versão 3.0

Para gerar classes proxy, execute estes passos:

1. Extraia o ficheiro CMWebService.zip para um directório temporário. É necessário extrair os quatro ficheiros seguintes:
  - CMWebService.wsdl
  - cmbmessages.xsd
  - cmdatamodel.xsd



- itemtype.xsd
2. Introduza o seguinte comando: C:\Program Files\Microsoft WSE\v3.0\Tools\WseWsd13.exe /language:CS /out:CMWebService.cs /type:webClient file:///C:\CMWebServiceWSDL\CMWebService.wsdl  
Pode executar este passo no Visual Studio 2005, adicionando uma referência Web ao ficheiro:///C:\CMWebServiceWSDL\CMWebService.wsdl.

#### **Referências relacionadas**

“Utilizar a propriedade `forceItemNamespace` ao desenvolver aplicações cliente que utilizem serviços da Web” na página 465

A propriedade `forceItemNamespace` permite a definição de um espaço de nomes para informações de esquema de tipos de artigo.

### **Programar pedidos de serviços da Web num ambiente .NET**

Pode utilizar o exemplo `CMWebServiceClient.cs` para o ajudar a compreender como criar um pedido de serviços da Web. O exemplo `CMWebServiceClient.cs` depende dos dados incluídos no First Steps e, por este motivo, certifique-se de que instalou o componente First Steps.

Pode executar as seguintes tarefas para desenvolver uma aplicação num ambiente .NET:

#### **Criar um objecto de serviços da Web**

Para criar uma instância de objecto de serviços da Web, pode utilizar o exemplo seguinte:

```
CMWebService webservice = null;
webservice = new CMWebService();
webservice.Timeout = 60000;
```

#### **Autenticar o pedido de serviços da Web**

Por questões de segurança, tem de criar um objecto de autenticação em cada pedido. Para obter instruções, consulte a secção “Autenticar pedidos de serviços da Web para fins de segurança” na página 455.

#### **Criar uma nova instância de um artigo**

O exemplo `CMWebServiceClient.cs` cria uma nova instância de um objecto `XYZ_InsPolicy`. Para obter informações mais detalhadas, consulte a secção “Criar uma nova instância de um artigo através de serviços da Web” na página 456

#### **Translinear pedidos XML**

O exemplo `CMWebServiceClient.cs` especifica pedidos XML directos através do utilitário de conversão WSDL, por exemplo:

```
elementRequest request = new RetrieveItemRequest();
request.AuthenticationData = authData;
request.attribute = elementattribute.att_value;
elementReply reply=webservice.element(request);
```

Para obter informações mais detalhadas sobre a sintaxe de pedidos XML, consulte a secção Programar operações de tempo de execução através de XML JavaBeans.

#### **Anexar conteúdos binários (se aplicável)**

Pode utilizar um destes três formatos padrão de mensagens binárias para anexar partes de conteúdos a um pedido. Para obter instruções, consulte a secção “Anexar partes de conteúdos binários a artigos em serviços da Web” na página 458.

O seguinte exemplo de translineação de um pedido XML para obter uma apólice de seguros XYZ (e um URL para a respectiva parte de recurso) através do seu identificador persistente.

#### Exemplo C#

```
public XYZ_ClaimForm retrieveClaimWithResourceURL(
 AuthenticationData authData,
 string pidURI)
{
 RetrieveItemRequest request = new RetrieveItemRequest();
 request.AuthenticationData = authData;
 request.retrieveOption =
 RetrieveItemRequestRetrieveOption.CONTENT_WITH_LINKS;
 request.contentOption = RetrieveItemRequestContentOption.URL;
 request.Item = new RetrieveItemRequestItem();
 request.Item.URI = pidURI;
 RetrieveItemReply reply = webservice.RetrieveItem(request);
 if (reply.RequestStatus.success == true) {
 return reply.Item.ItemXML.XYZ_ClaimForm;
 } else {
 Console.WriteLine("Retrieve Policy failed.");
 displayErrorInfo(reply.RequestStatus.ErrorData);
 return null;
 }
}
```

## Integrar o CMBGenericWebService

Utilizando o CMBGenericWebService, pode escrever uma aplicação para enviar os seus próprios pedidos XML através de um envelope SOAP para o servidor de serviços da Web.

Este servidor traduz o pedido em chamadas no bean do Operador XML e, de seguida, envia as respostas XML/SOAP para a primeira aplicação. Os principais passos a executar são os seguintes:

1. Obter o ficheiro WSDL a partir do seguinte URL: `http://hostname/CMBGenericWebService/services/CMBGenericWebService/wsd/`  
`CMBGenericWebService.wsd`
2. Gerar classes proxy para o WSDL.
3. Escrever o programa de aplicação de modo a utilizar as classes proxy para comunicarem com o CMBGenericWebService. O URL do CMBGenericWebService é `http://hostname/CMBGenericWebService/services/CMBGenericWebService`.

Os exemplos Java com o WebSphere Application Server 6 estão localizados em `IBMCMROOT/samples/webservices/GenericWebService/java/WAS6/base`.

Para obter informações mais detalhadas, consulte a secção “Personalizar aplicações Java para trabalharem com o CMBGenericWebService” na página 452.

### Obter o ficheiro WSDL para o serviço da Web CMBGenericWebService

O ficheiro WSDL é um artigo fundamental ao trabalhar com os serviços da Web.

O ficheiro WSDL para o CMBGenericWebService é estático e está instalado no servidor da Web quando o serviço da Web genérico é instalado. Para visualizar o ficheiro WSDL, é necessário ter os serviços da Web instalados no sistema.

Para obter o ficheiro WSDL para o CMBGenericWebService, execute os seguintes passos:

1. Introduza o seguinte URL num navegador: `http://localhost/CMBGenericWebService/services/CMBGenericWebService/wsdl/CMBGenericWebService.wsdl`.
2. Introduza o nome do sistema central e o número de porta correcto para o sistema.
3. Para guardar o WSDL, no menu do navegador, faça clique em **File (Ficheiro)** → **Save As (Guardar como)** e seleccione o directório no qual pretende guardar o ficheiro WSDL. Faça clique em **Save (Guardar)**.

Pode agora visualizar o WSDL num navegador.

## **Personalizar aplicações Java para trabalharem com o CMBGenericWebService**

Pode personalizar a aplicação Java para criar os pedidos XML para enviar para os serviços da Web.

Pode utilizar qualquer toolkit de cliente com base em JAX-RPC para gerar as classes que invocam os serviços da Web e transmitem os parâmetros de um lado para o outro para os serviços da Web. O WebSphere Application Server Versão 6 faculta uma ferramenta cliente denominada WSDL2Java que pode ser utilizada para gerar classes para serviços da Web. Como o ficheiro WSDL não define a sintaxe dos documentos XML, a interface dos serviços da Web não muda se o esquema XML para o pedido mudar.

Para escrever uma aplicação no ambiente Java, execute os seguintes passos:

1. Instale o WebSphere Application Server Versão 6.1.
2. Utilize a ferramenta WSDL2Java do WebSphere Application Server para gerar classes proxy de Java a partir da WSDL para o serviço da Web CMBGenericWebService.
3. Utilizando os exemplos Java para orientação, programe a aplicação de serviços da Web utilizando as classes proxy de Java.

### **Conceitos relacionados**

“Programar pedidos de serviços da Web num ambiente Java”

### **Tarefas relacionadas**

“Obter o ficheiro WSDL para o serviço da Web CMBGenericWebService” na página 451

O ficheiro WSDL é um artigo fundamental ao trabalhar com os serviços da Web.

**Programar pedidos de serviços da Web num ambiente Java:** Depois de configurar o ambiente de serviços da Web para Java, pode utilizar GenericWebServiceSample.java como uma base para criar um pedido de serviços da Web. Em resumo, a aplicação Java contém o código seguinte:

### **Criar um objecto de serviços da Web**

Para criar uma instância de objecto de serviços da Web, utilize o seguinte exemplo:

```
CMBGenericWebServiceService cs =
 new CMBGenericWebServiceServiceLocator();
cmbservice = cs.getCMBGenericWebService();
```

### Autenticar o pedido de serviços da Web

Por questões de segurança, tem de criar um objecto de autenticação em cada pedido.

### Criar uma nova instância de um artigo (se aplicável)

O exemplo `GenericWebServiceSample.java` cria uma instância de um artigo `XYZ_InsPolicy`.

### Translinear o pedido XML

O exemplo `GenericWebServiceSample.java` transmite os parâmetros para modelos de mensagens predefinidos para criar pedidos XML. Por exemplo:

```
String requestXML = MessageFormat.format(
 SampleMessageTemplate.TEMPLATE,
 new Object[] { authenticationDataXML, pid });
CMBXMLResponse response = null;
response = cmbservice.processXMLRequest(requestXML, null);
String replyXML = response.getXmlResponseText();
return replyXML;
```

Todos os modelos de mensagens de serviços da Web estão definidos no ficheiro, `SampleMessageTemplate.java`. Os modelos de operações básicas incluem:

- AUTHENTICATION\_DATA\_TEMPLATE
- CREATE\_ITEM\_TEMPLATE
- QUERY\_TEMPLATE
- RETRIEVE\_ITEM\_WITH\_RESOURCE\_URL\_TEMPLATE
- RETRIEVE\_ITEM\_WITH\_ATTACHMENTS\_TEMPLATE
- UPDATE\_CLAIM\_TEMPLATE
- CREATE\_LINKS\_TEMPLATE
- DELETE\_LINKS\_TEMPLATE
- UPDATE\_POLICY\_TEMPLATE
- UPDATE\_POLICY\_EXT\_TEMPLATE
- DELETE\_ITEM\_TEMPLATE
- BATCH\_DELETE\_TEMPLATE
- ADD\_TO\_FOLDER

Os modelos de artigos XML incluem:

- XYZ\_InsPolicy\_TEMPLATE
- XYZ\_InsPolicy\_FOLDER\_TEMPLATE
- XYZ\_InsPolicy\_\_XYZ\_Insured\_TEMPLATE
- XYZ\_InsPolicy\_\_XYZ\_VIN\_TEMPLATE
- ICM\_BASE\_TEMPLATE
- XYZ\_ClaimForm\_TEMPLATE

Os modelos de encaminhamento de documentos incluem:

- LIST\_PROCESS\_TEMPLATE
- START\_PROCESS\_TEMPLATE
- LIST\_WORKPACKAGES\_TEMPLATE
- CONTINUE\_PROCESS\_TEMPLATE
- TERMINATE\_PROCESS\_TEMPLATE

Para obter mais informações sobre a sintaxe de pedidos XML, consulte a secção “Programar operações de tempo de execução através dos JavaBeans XML” na página 373.

#### **Criar um DOM a partir de uma cadeia XML**

Construir um documento DOM (Document Object Model) a partir de uma cadeia de mensagens XML ajuda a ler os elementos incluídos nas respostas XML.

```
if (factory == null) {
 factory = DocumentBuilderFactory.newInstance();
}
builder = factory.newDocumentBuilder();
Document document = null;
document = builder.parse(
 new InputSource(new StringReader(replyXML)));
```

#### **Anexar conteúdos binários (se aplicável)**

O cliente de serviços da Web que utiliza o terminal CMBGenericWebService pode carregar ou descarregar dados binários para o terminal CMBGenericWebService utilizando MIME como o formato de anexo.

#### **Analisar a resposta dos serviços da Web**

O exemplo GenericWebServiceSample.java contém vários métodos para validar respostas XML e reportar erros.

O seguinte exemplo translineia uma resposta XML para obter uma apólice de seguros XYZ (e um URL para a respectiva parte de recurso) utilizando o respectivo identificador persistente.

#### **Exemplo de Java**

```
public CMDocument retrievePolicyWithResourceURL(
 String authenticationDataXML,
 String pid) {

 String requestXML = MessageFormat.format(
 SampleMessageTemplate
 .RETRIEVE_ITEM_WITH_RESOURCE_URL_TEMPLATE,
 new Object[] { authenticationDataXML, pid });

 CMBXMLResponse response = null;

 // call the web service with the xml request message

 try {
 response =
 cmbservice.processXMLRequest(requestXML, null);
 } catch (RemoteException e) {
 e.printStackTrace();
 return null;
 }

 // Get the DOM object representing the xml message
 Document document = getDocument(
 response.getXmlResponseText());

 if (document == null) {
 return null;
 }

 // parse the status of the response from web service
 if (parseRequestStatus(document) != true) {
 return null;
 }
}
```

```
// return the array of PIDs for the resources associated
// with the retrieved document
Element policyElement =
 getElement(document, "XYZ_InsPolicy");

if (policyElement == null) {
 return null;
}

CMDocument policy = parsePolicy(policyElement);
return policy;
}
```

### Conceitos relacionados

“Exemplos de serviços da Web” na página 464

É facultado um conjunto de exemplos de serviços da Web com comentários precisos para ajudar o utilizador a trabalhar com os serviços da Web.

## Autenticar pedidos de serviços da Web para fins de segurança

Cada vez que efectua um pedido aos serviços da Web, tem de transmitir um ID de utilizador do DB2 Content Manager e respectiva palavra-passe, ou um token credencial do WebSphere associado ao utilizador do DB2 Content Manager.

Se o utilizador não tiver o privilégio para efectuar esse pedido, o pedido não é processado e é devolvido um erro da resposta SOAP. Por exemplo, se um utilizador pretender efectuar uma alteração a uma apólice de seguro, mas tiver apenas privilégios de visualização, não poderá efectuar a alteração à apólice.

**Importante:** Por predefinição, o ID de utilizador e respectiva palavra-passe transmitidos no pedido de serviços da Web não são encriptados. Se todos os pedidos de serviços da Web forem processados dentro de uma firewall, os IDs e palavras-passe encriptados poderão não constituir um problema. No entanto, se o cliente estiver fora de uma firewall, deve utilizar SSL para os pedidos SOAP.

Para autenticar os pedidos de serviço da Web, crie um objecto `AuthenticationData`. Tem de incluir este objecto em cada pedido.

O seguinte exemplo de C# específico devolve um objecto `AuthenticationData` inicializado em que *server* representa o nome da base de dados da biblioteca do servidor do DB2 Content Manager (por exemplo, *cmdb2ls*), *userid* representa o ID de utilizador do DB2 Content Manager (por exemplo, *icmadmin*), e *password* representa a palavra-passe de início de sessão.

### Exemplo C#

```
public AuthenticationData setupAuthenticationData(
 string server, string userid, string password)
{
 AuthenticationData authData = new AuthenticationData();
 ServerDef serverDef = new ServerDef();
 serverDef.ServerName = server;
 authData.ServerDef = serverDef;
 AuthenticationDataLoginData loginData =
 new AuthenticationDataLoginData();
 loginData.UserID = userid;
```

```

 loginData.Password = password;
 authData.LoginData = loginData;
 return authData;
 }

```

O seguinte exemplo de Java cria um objecto de autenticação, em que *dstype* representa o tipo de origem de dados (por exemplo, ICM), *server* representa o nome da base de dados da biblioteca do servidor do DB2 Content Manager (por exemplo, cmdb21s), *username* representa o ID de utilizador do DB2 Content Manager (por exemplo, icmadmin), e *password* representa a palavra-passe de início de sessão.

#### Exemplo de Java

```

public String createAuthenticationDataXML(
 String dstype,
 String server,
 String username,
 String password) {

 return MessageFormat.format(
 SampleMessageTemplate.AUTHENTICATION_DATA_TEMPLATE,
 new Object[] { dstype, server, username, password });

}

```

## Criar uma nova instância de um artigo através de serviços da Web

Este exemplo cria uma instância de um artigo XYZ\_InsPolicy.

#### Exemplo C#

As seguintes classes proxy de segmentos de código C# utilizadas para construir a representação XML de um artigo de tipo XYZ\_Policy, que serão uma parte de CreateItemRequest.

```

private XYZ_InsPolicy setupPolicy(
 string[,] insured,
 string street,
 string city,
 string state,
 string zip,
 string policyNumber,
 string[] vins,
 string[,] resources)
{
 XYZ_InsPolicy policy = new XYZ_InsPolicy();
 policy.XYZ_City = city;
 policy.XYZ_Street = street;
 policy.XYZ_State = state;
 policy.XYZ_ZIPCode = zip;
 policy.XYZ_PolicyNum = policyNumber;
 policy.XYZ_VIN = new XYZ_InsPolicyXYZ_VIN[vins.Length];

 for (int i=0; i<vins.Length; i++) {
 policy.XYZ_VIN[i] = new XYZ_InsPolicyXYZ_VIN();
 policy.XYZ_VIN[i].XYZ_VIN = vins[i];
 }

 policy.XYZ_Insured =
 new XYZ_InsPolicyXYZ_Insured[insured.GetLength(0)];

 for (int i=0; i<insured.GetLength(0); i++) {
 policy.XYZ_Insured[i] = new XYZ_InsPolicyXYZ_Insured();
 policy.XYZ_Insured[i].XYZ_InsrdFName = insured[i, 0];
 policy.XYZ_Insured[i].XYZ_InsrdLName = insured[i, 1];
 }
}

```

```

 }

 // Document parts of the policy (ICM base parts only)
 policy.ICMBASE = new ICMBASE[resources.GetLength(0)];
 for (int i=0; i<resources.GetLength(0); i++) {
 policy.ICMBASE[i] = new ICMBASE();
 policy.ICMBASE[i].resourceObject = new LobObjectType();
 policy.ICMBASE[i].resourceObject.label=
 new LobObjectTypeLabel();
 policy.ICMBASE[i].resourceObject.label.name =
 resources[i, 0];
 policy.ICMBASE[i].resourceObject.MIMEType =
 resources[i, 1];
 }
 return policy;
}

```

### Exemplo de Java

Utilizando os modelos fornecidos no exemplo de Java dos serviços da Web `SampleMessageTemplate.java`, o seguinte fragmento de código Java cria um XML que representa um artigo de tipo `XYZ_Policy`.

```

public String generatePolicyDataXML(
 String[] policyInfo,
 String[] [] insured,
 String[] vins,
 String[] [] resources)
{
 StringBuffer insuredXML = new StringBuffer();
 StringBuffer vinsXML = new StringBuffer();
 StringBuffer resourcesXML = new StringBuffer();
 for (int i = 0; i < insured.length; i++)
 insuredXML.append(
 MessageFormat.format(
 SampleMessageTemplate
 .XYZ_InsPolicy__XYZ_Insured_TEMPLATE,
 new Object[] {
 insured[i][0], insured[i][1] }));

 for (int i = 0; i < vins.length; i++)
 vinsXML.append(
 MessageFormat.format(
 SampleMessageTemplate
 .XYZ_InsPolicy__XYZ_VIN_TEMPLATE,
 new Object[] { vins[i] }));

 if (resources != null) {
 for (int i = 0; i < resources.length; i++)
 resourcesXML.append(
 MessageFormat.format(
 SampleMessageTemplate.ICM_BASE_TEMPLATE,
 new Object[] {
 resources[i][0],
 resources[i][1] }));
 }
 else {
 resourcesXML.append("");
 }
 return MessageFormat.format(
 SampleMessageTemplate.XYZ_InsPolicy_TEMPLATE,
 new Object[] {
 policyInfo[0],
 policyInfo[1],
 policyInfo[2],
 policyInfo[3],
 policyInfo[4],

```



```

 insuredXML.toString(),
 vinsXML.toString(),
 resourcesXML.toString())});
 }

```

### Conceitos relacionados

“Itens” na página 66

Um *artigo* é a entidade básica gerida pelo servidor de bibliotecas.

## Anexar partes de conteúdos binários a artigos em serviços da Web

Os exemplos de serviços da Web do DB2 Content Manager dependem de dois formatos de mensagens binárias para enviar anexos (como, por exemplo, partes base, anotações ou registos de notas) com os pedidos XML.

Os seguintes formatos de mensagens binárias padrão são utilizados pelos serviços da Web do DB2 Content Manager:

### MTOM (Message Translation Optimization Mechanism)

MTOM é um método padrão do W3C para enviar dados binários para serviços da Web e a partir dos mesmos. Utiliza XOP (XML-binary Optimized Packaging) para transmitir dados binários e destina-se a substituir anexos MIME e DIME. Apenas o CMWebService suporta tipos MTOM.

O seguinte exemplo cria um objecto de anexo MTOM para manter as partes de conteúdo (na matriz resources) dentro dos artigos do DB2 Content Manager. O código encontra-se no projecto exemplo CMWebServiceClientWSE30.

#### Exemplo C#

Para utilizar um anexo MTOM, os dados binários têm de ser carregados para um elemento XML denominado MTOMAttachment.

```

//<summary> Sets up an array of MTOMAttachments.
//The MTOMAttachments are used to send the content
// of the documents parts to the server in the soap message
//</summary>
//<param name="resources"> array containing information
//about the document part to be associated
//with the policy </param>
//<returns> new array of MTOMAttachments</returns>

```

```

MTOMAttachment[] setupAttachments(string[,] resources)
{
 //Create MTOMAttachment objects for all the resources
 //passed in the resources array provides
 // the information for the document parts to be
 //sent to the server
 MTOMAttachment[] attachments = new
 MTOMAttachment[resources.GetLength(0)];
 for (int i=0;i<resources.GetLength(0); i++)
 {
 // resources[i,0] is the ID of the attachment.
 //This should match the label of the document part
 // on the server side, the ID of the attachment and the
 //label of the document part are used to
 // match the attachment with its associated document part
 // resources[i,1] is the mime type of the document part
 // resources[i,2] is the file path to the document part

 attachments[i] = new MTOMAttachment();
 attachments[i].ID = resources[i, 0];
 attachments[i].MimeType = resources[i, 1];
 }
}

```

```

 attachments[i].Value = File.ReadAllBytes(resources[i, 2]);
 }
 return attachments;
}

```

Em seguida, o objecto MTOMAttachment tem de ser designado por um elemento descendente da mensagem XML de pedido mtomRef.

```

//<summary>
//Creates a new policy on the Content Manager server.
//</summary>
//<param name="authData">authentication data to be used for
//connecting to the datastore
//</param>
//<param name="policy">insurance policy to be created</param>
//<param name="attachments">array of MTOMAttachment representing
//the content of the document parts to be created for
//the policy on the server
//</param>
//<returns> the PID URI of the created policy
If an error occurs, null is returned. </returns>
public string createPolicy(AuthenticationData authData,
 XYZ_InsPolicy policy, MTOMAttachment[] attachments)
{ //Create a CreateItemRequest object to set up the request for
 //creating a XYZ_InsPolicy
 CreateItemRequest request = new CreateItemRequest();
 request.AuthenticationData = authData;
 //set the authentication information for the create request
 request.Item = new CreateItemRequestItem();
 //Because we are creating one item, we have an array
 //of one CreateItemRequestItem object
 //CreateItemRequestItem has an ItemXML object which
 //points to the in-memory instance of the item to be created.
 //In this case the item is an instance of XYZ_InsPolicyrequest.
 Item.ItemXML = new ItemXML();
 request.Item.ItemXML.XYZ_InsPolicy = policy;
 // Add MTOM attachments
 if (attachments != null)
 {
 request.mtomRef = new MTOMAttachment[attachments.Length];
 for (int i = 0; i < attachments.Length; i++)
 {
 request.mtomRef[i] = attachments[i];
 }
 }
 CreateItemReply reply = webservice.CreateItem(request);
 //invoke the Content Manager web service
 //with the CreateItem operation
 // check to see if the operation was successful or not
 if (reply.RequestStatus.success == true)
 { // return the PID URI of the created item
 return reply.Item.URI;
 }
 else
 {
 Console.WriteLine("Create failed.");
 displayErrorInfo(reply.RequestStatus.ErrorData);
 return null;
 }
}
}

```

### MIME (Multipurpose Internet Mail Extensions)

O protocolo padrão de envio de mensagens MIME pode identificar ficheiros e tipos binários através dos cabeçalhos Content-ID ou Content-Location num envelope SOAP. O CMBGenericWebService e o CMWebService suportam tipos MIME. Os dois artigos XML no DB2

Content Manager utilizam o atributo `MIMEType` no elemento `<resourceObject>` para definir a codificação. Por exemplo:

```
<ICMBASE>
 <resourceObject MIMEType="image/jpeg"
 xmlns="http://www.ibm.com/xmlns/db2/cm/api/1.0/schema">
 <label name="image 1" />
 </resourceObject>
</ICMBASE>
```

Os exemplos de serviços da Web de Java utilizam tipos MIME para anexar partes de conteúdos de recurso.

O seguinte exemplo cria um objecto `MimeMultipart` para manter as partes de conteúdo (na matriz `resources`) dentro dos artigos do DB2 Content Manager.

#### Exemplo de Java

```
public MimeMultipart setupAttachments(String[] [] resourceData) {
 MimeMultipart mp = new MimeMultipart();
 for (int i = 0; i < resourceData.length; i++) {
 MimeBodyPart mbp = new MimeBodyPart();
 DataHandler handle =
 new DataHandler(new FileDataSource(resourceData[i][2]));

 try {
 mbp.setDataHandler(handle);
 // we need to create a header called ID which matches
 // the label value
 // of the resourceObject specified in the policy
 // xml on the server
 // this ID value is used to match the
 // appropriate resourceObject

 mbp.addHeader("ID", resourceData[i][1]);
 mbp.addHeader("MimeTypes", resourceData[i][0]);
 } catch (MessagingException e) {
 System.out.println(
 "Failed in creating a MimeBodyPart for attachments");
 e.printStackTrace(); return null;
 }

 try {
 mp.addBodyPart(mbp);
 } catch (MessagingException e1) {
 System.out.println(
 "Failed in creating a MimeMultiPart for attachments");
 e1.printStackTrace();
 return null;
 }
 }
 return mp;
}
```

#### DIME (Direct Internet Message Encapsulation)

DIME é um protocolo definido pela Microsoft que pode especificar o comprimento, a codificação e o tamanho de um ficheiro previamente nos campos de cabeçalho para poupar tempo no respectivo cálculo. Este protocolo é o único formato suportado para processar ficheiros com assinaturas digitais ou ficheiros binários de grandes dimensões num ambiente .NET. Apenas o CMWebService suporta tipos DIME. Os exemplos de serviços da Web de C# utilizam DIME para anexar partes de conteúdos de recurso.

O seguinte exemplo cria um objecto de anexo DIME para manter as partes de conteúdo (na matriz resources) dentro dos artigos do DB2 Content Manager:

#### Exemplo C#

```
DimeAttachment[] setupAttachments(string[,] resources)
{
 DimeAttachment[] attachments =
 new DimeAttachment[resources.GetLength(0)];
 for (int i=0;i<resources.GetLength(0);i++) {
 // resources[i,0] is the ID of the attachment.
 // This should match
 // the label of the document part on the server side,
 //the ID of the
 // attachment and the label of the document part
 //are used to match
 // the attachment with its associated document part
 // resources[i,1] is the mime type of the document part
 // resources[i,2] is the file path to the document part

 attachments[i] = new DimeAttachment(
 resources[i,0],
 resources[i,1],
 Microsoft.Web.Services.Dime.TypeFormatEnum.MediaType,
 resources[i,2]);
 }
 return attachments;
}
```

## Configurar início de sessão único

O início de sessão único é utilizado pelas aplicações da Web para garantir que o utilizador da aplicação tenha de fornecer o ID de utilizador e a palavra-passe apenas uma vez.

Para configurar o início de sessão único (SSO) para os serviços da Web, tem de configurar o SSO para o DB2 Content Manager e para o WebSphere Application Server. Em seguida, pode estabelecer ligação ao servidor do DB2 Content Manager Versão 8 através dos serviços da Web utilizando as credenciais geradas pelo WebSphere Application Server em cada pedido, em vez de fornecer o ID de utilizador e palavra-passe em todos os pedidos. Para obter instruções sobre a configuração para SSO, consulte o tópico *Configuring the eClient for single sign-on (SSO)*

Para verificar se o SSO está correctamente configurado, reinicie a aplicação de serviços da Web e execute os seguintes passos.

1. Aceda ao URL <http://hostname/CMBSpecificWebService/index.html> num navegador da Web.
2. Introduza o ID de utilizador e a palavra-passe definida durante a configuração da segurança do WebSphere Application Server.

Se o ID de utilizador e palavra-passe estiverem correctos, será apresentada a mensagem de texto DB2 Content Manager Specific Web Service no navegador da Web.

## Tokens de LTPA e início de sessão único

O início de sessão único é utilizado pelas aplicações da Web para garantir que o utilizador da aplicação tenha de fornecer o ID de utilizador e a palavra-passe apenas uma vez.

Quando um utilizador inicia sessão num servidor activado para início de sessão único, cria um token de LTPA (Lightweight Third Party Access). Esse token de LTPA é utilizado para autenticação desse utilizador em pedidos futuros.

O token de LTPA contém informações credenciais utilizadas para aceder ao servidor. Estas informações são usadas nos pedidos subsequentes efectuados às aplicações de serviços da Web pela máquina cliente. Existem várias formas de obter as informações credenciais, que estão armazenadas como um cookie na máquina cliente e estão também disponíveis no pedido HTTP em futuras viagens ao servidor. Uma das formas é implementar uma servlet no servidor que pode extrair as informações de credenciais dos pedidos HTTP que entram.

### Exemplo

```
/**
 * Retrieve the LTPA token from the HttpServletRequest
 * if the SSO authentication was successful,
 * any subsequent request from the browser will
 * have an SSO cookie in the header
 *
 * @return String
 */
public String getCookie(HttpServletRequest request) {
 Cookie cookies[] = request.getCookies();
 String cookievalue = null;
 for (int i=0; i<cookies.length; i++){
 if (cookies[i]
 .getName().equalsIgnoreCase("LtpaToken")) {
 cookievalue = cookies[i].getValue(); break;
 }
 }
 return cookievalue;
}
```

Ao incluir esta credencial nos pedidos ao serviço da Web, é necessário convertê-la primeiro para a codificação de base 64, como mostra o exemplo seguinte:

```
String cookie = getCookie(request)
// client application does not necessarily need to run
//inside the servlet, just implemented here as an example.;
Object credential = org.apache.soap.encoding.soapenc.Base64.encode
(cookie.getBytes());
```

Pode então utilizar a credencial na secção de Autenticação de Dados de um pedido aos serviços da Web. Quando um pedido que contém uma credencial é recebido, é efectuada uma busca no WebSphere Application Server para identificar o utilizador.

### Exemplo

```
protected static final String AUTHENTICATION_DATA_TEMPLATE =
 "<AuthenticationData connectString=\"SCHEMA=ICMADMIN\"
 configString=\"\">"
 + "<ServerDef>"
 + "<ServerType>{0}</ServerType>"
 + "<ServerName>{1}</ServerName>"
 + "</ServerDef>"
 + "<Credential>{2}</Credential>"
 + "</AuthenticationData>"
protected static final String QUERY_TEMPLATE =
 "<RunQueryRequest xmlns=\"http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema\"
 maxResults=\"0\"
 + \"version=\"latest-version(.)\" contentOption=\"{0}\" retrieveOption=\"{1}\">"
```

```

 + "{2}"
 + "<QueryCriteria>"
 + "<QueryString>{3}</QueryString>"
 + "</QueryCriteria>"
 + "</RunQueryRequest>";
 .
 .
 .
 String queryString = "/XYZ_InsPolicy[@XYZ_State LIKE \"
 String authenticationXML =
 MessageFormat.format(
 AUTHENTICATION_DATA_TEMPLATE,
 new Object[] { "ICM", "icmnlsdb", credential });
 String requestXML =
 MessageFormat.format(
 QUERY_TEMPLATE,
 new Object[] { "URL", "ITEMTREE", authenticationXML, query });

 CMBXMLResponse response = null;
 try {
 response = cmbservice.processXMLRequest(requestXML, null);
 } catch (RemoteException e) {
 e.printStackTrace();
 return null;
 }
}

```

---

## Desempenho dos serviços da Web

Para melhorar o desempenho de um serviço da Web, tenha em consideração o tamanho da pilha, a quantidade de dados obtida e o conjunto de ligações da base de dados.

Se o tamanho da pilha não estiver definido correctamente no WebSphere Application Server, poderá receber excepções. Se receber uma excepção `OutOfMemory`, significa que o tamanho máximo da pilha poderá ser demasiado reduzido. Tente aumentar o valor, mas certifique-se de que o valor do tamanho da pilha de Java Virtual Machine está directamente relacionado com a quantidade de memória física do sistema. Nunca defina o tamanho da pilha de Java Virtual Machine superior à memória física do sistema, de modo a evitar a E/S de disco provocada pela comutação. As informações incluem a definição dos parâmetros de Java Virtual Machine relativamente a opções de recolha de dados não utilizados como, por exemplo, **-Xgcpolicy:optavgpause** e argumentos de Java Virtual Machine genéricos como, por exemplo, **-Xnoclassgc**.

O modo como a aplicação cliente está programada também pode contribuir para a excepção `OutOfMemory`. Pode definir um limite de quantidade de dados a obter a partir do servidor de serviços da Web. Utilize os parâmetros nos pedidos de serviços da Web para definir o limite.

Em `RunQueryRequest`, o atributo `maxResults` pode ser utilizado para limitar o número de artigos a devolver pelo pedido. `RunQueryRequest` não só pesquisa os PIDs de artigos que correspondem à cadeia de consulta especificada, como também devolve os mesmos, de acordo com o atributo `retrieveOption`. Para obter um melhor desempenho, uma aplicação cliente normal pode implementar a respectiva função de paginação. Com a paginação, o cliente pode enviar um `RunQueryRequest` com `retrieveOption` um `RetrieveItemRequest` para obter artigos em cada página que esteja actualmente a ser lida. Esta abordagem irá reduzir o tempo de resposta para os utilizadores finais da aplicação cliente.

**Importante:** Utilize o atributo `maxResults` no `RunQueryRequest` para evitar a obtenção de uma quantidade demasiado grande de PIDs.

`RetrieveFolderItemsRequest` tem o atributo `maxItems` que permite definir o limite de artigos a devolver ao abrir uma pasta. A paginação do cliente também é recomendada na implementação de funções como, por exemplo, conteúdos de pastas.

`ListWorkPackagesRequest` tem uma forma diferente de definir o limite de pacotes de trabalho a devolver. Ao definir uma lista de trabalho no cliente de administração do sistema, defina um valor máximo para **Quantity to Return (Quantidade a devolver)** na janela *Worklist definition* (Definição da lista de trabalho).

Além disso, para pedidos de serviços da Web concorrentes de grande volume, active o conjunto de ligações da base de dados no WebSphere Application Server. As recomendações estão realçadas no manual de optimização para activar o conjunto de ligações da base de dados para a aplicação eClient. Pode utilizar essas mesmas informações para os serviços da Web.

Para obter um melhor desempenho, siga as sugestões que estão realçadas na secção *Optimização de desempenho para o Content Manager relativas à WebSphere Application Server optimização para o eClient*. As instruções incluídas nestes IBM Redbooks também são úteis para a configuração dos serviços da Web.

---

## Exemplos de serviços da Web

É facultado um conjunto de exemplos de serviços da Web com comentários precisos para ajudar o utilizador a trabalhar com os serviços da Web.

Para começar a trabalhar com os exemplos de serviços da Web, consulte o ficheiro adequado ao seu ambiente em `IBMCMROOT/samples/webservices`:

### **GenericWebServiceSampleWAS6.readme**

Comece por este ficheiro se estiver a trabalhar com os serviços da Web no WebSphere Application Server Versão 6.1

### **CMWebServiceClient.readme**

Comece por este ficheiro se estiver a trabalhar com os serviços da Web num ambiente .NET 1.1/WSE 1.0.

### **CMWebServiceClientWSE30.readme**

Comece por este ficheiro se estiver a trabalhar com os serviços da Web num ambiente .NET 2.0 ou 3.0/WSE 3.0

### **CMWebServiceClientJava.readme**

Comece por este ficheiro se estiver a trabalhar com os serviços da Web num ambiente Apache Axis 1.4

---

## Detecção e correcção de problemas dos serviços da Web

As informações de detecção e correcção de problemas podem ajudar o utilizador a diagnosticar e resolver problemas relacionados com os serviços da Web.

As informações incluídas nesta secção facultam informações de detecção e correcção de problemas ajudar o utilizador a resolver alguns problemas comuns dos serviços da Web.



## Definir a instrução em memória cache no conjunto de ligações do WebSphere Application Server para serviços da Web

Se seleccionar um número elevado para as instruções em memória cache no conjunto de ligações do WebSphere Application Server, a aplicação de serviços da Web pode ficar sem espaço de memória e deixar de funcionar.

### Acção

Ao utilizar o conjunto de ligações do WebSphere Application Server, seleccione um valor inferior a 10 para as instruções em memória cache.

## Utilizar a propriedade `forceItemNamespace` ao desenvolver aplicações cliente que utilizem serviços da Web

A propriedade `forceItemNamespace` permite a definição de um espaço de nomes para informações de esquema de tipos de artigo.

No DB2 Content Manager Versão 8.4, a propriedade `forceItemNamespace` no ficheiro `cmbxmlservices.properties` está sempre activada e o respectivo valor é definido por predefinição como `http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema`. Não deverá alterá-la a não ser que tenha problemas de retrocompatibilidade nas aplicações cliente de serviços da Web que utilizem `CMWebService` e que foram desenvolvidas com o DB2 Content Manager Versão 8.3.

Por predefinição, no DB2 Content Manager Versão 8.3, o ficheiro `cmbxmlservices.properties` não tem a propriedade `forceItemNamespace`. No entanto, é possível adicionar a linha `forceItemNamespace=http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema` manualmente ao ficheiro `cmbxmlservices.properties`.

Se desenvolveu uma aplicação cliente de serviços da Web ligada aos serviços da Web específicos do DB2 Content Manager Versão 8.3 (`CMWebService`), a aplicação não obtém artigos do servidor de serviços da Web do DB2 Content Manager Versão 8.4 Web services se a propriedade `forceItemNamespace` não estiver no ficheiro `cmbxmlservices.properties` ou se a propriedade tiver sido excluída com comentários

Para resolver este problema, pode optar por uma das seguintes duas hipóteses:

- Defina como comentário a linha `forceItemNamespace=http://www.ibm.com/xmlns/db2/cm/beans/1.0/schema` no ficheiro `cmbxmlservices.properties` e reinicie o servidor de serviços da Web do DB2 Content Manager Versão 8.4. Não é necessário efectuar alterações à aplicação.
- Descarregue ou exporte os ficheiros de esquema WSLD e XML para o `CMWebService` do DB2 Content Manager Versão 8.4 e gere novamente as classes proxy dos ficheiros de esquema WSLD e XML. Não é necessário alterar o código de outras partes da aplicação. Contudo, é necessário compilar e construir novamente a aplicação completa.

**Importante:** O cliente .NET 2.0/WSE 3.0 só pode trabalhar com o servidor de serviços da Web do Content Manager activando a propriedade `forceItemNamespace`. A propriedade `forceItemNamespace` tem de ser



activada ao gerar o WSDL Specific Web Service e os ficheiros de esquema para o cliente .NET 2.0/WSE 3.0.

## Actualizar a configuração do servidor de serviços da Web para consultar as bibliotecas de 32 bits ao utilizar uma instância de 64 bits do DB2

As APIs do DB2 Content Manager requerem as bibliotecas de 32 bits. Os serviços da Web também requerem as mesmas bibliotecas.

### Sintoma

É apresentado o erro: DGL0394A: Error in DriverManager.getConnection; No suitable driver quando envia pedidos através dos serviços da Web do DB2 Content Manager ao utilizar uma instância de 64 bits do DB2.

### Causa possível

A instalar os serviços da Web do DB2 Content Manager num sistema com uma instância de 64 bits do DB2, as bibliotecas predefinidas na configuração do servidor de serviços da Web no WebSphere Application Server são versões de 64 bits.

### Acção

Actualize a configuração do servidor de serviços da Web para consultar as bibliotecas de 32 bits, executando estes passos:

1. Certifique-se de que o Fix Pack 1 está instalado.
2. Actualize a propriedade LIBPATH utilizada pela configuração do servidor para incluir a localização das bibliotecas de 32 bits:
  - a. Abra a consola administrativa do WebSphere Application Server.
  - b. Faça clique em **Application Servers (Servidores de aplicações)** → **cmwebsvc** → **Process Definition Environment Entries (Entradas de ambiente de definição de processos)**.
  - c. Em AIX, faça clique na variável LIBPATH de modo a poder editar a mesma. Em Solaris, faça clique em LD\_LIBRARY\_PATH. Esta variável contém um ou mais caminhos para várias localizações de bibliotecas no sistema de ficheiros.
  - d. As bibliotecas UDB do DB2 encontram-se em DB2instancehome/lib, em que DBinstancehome é /home/db2inst1/sqllib por predefinição.
  - e. Altere o directório lib para lib32.
  - f. Faça clique em **OK** para guardar o novo valor.
  - g. Guarde as alterações efectuadas à configuração principal.
3. Reinicie o servidor cmwebsvc.

---

## Trabalhar com o toolkit do visualizador de documentos de Java

O toolkit do visualizador de documentos é um conjunto de APIs de Java™ que facilitam a apresentação, a manipulação e a anotação de documentos.

Este toolkit inclui uma interface gráfica de utilizador com base em Swing, bem como classes não visuais que facultam capacidades de conversão e apresentação de documentos. Existem muitas capacidades de personalização e extensão que permitem definir o visualizador para se adequar a aplicações do Content Manager especializadas.

---

### Descrição geral do toolkit do visualizador de Java

O toolkit do visualizador de Java™ suporta a visualização, a anotação e a manipulação de vários formatos de documentos, principalmente: TIFF, MO:DCA (com IOCA, PTOCA), GIF, JPEG, PCX, Bitmap, documentos de texto e documentos Microsoft® Office que utilizem uma ponte para o toolkit Stellent Outside-In

É possível expandir o visualizador para suportar outros formatos de documento, escrevendo motores de documentos que são classes que facultam funções específicas de formatos de documento.

O toolkit do visualizador de Java faculta as capacidades seguintes:

- Interface multi-documento com vários esquemas diferentes, incluindo esquemas com separadores e lado a lado. A visualização lado a lado permite duas vistas diferentes de um único documento ou de dois documentos diferentes. As vistas podem ser configuradas na horizontal ou na vertical e editadas de forma independente.
- Vista de miniaturas das páginas de um documento: ao fazer clique numa miniatura de uma página, passa para essa página. A área actualmente visível na página é indicada por um rectângulo vermelho. Pode arrastar ou redimensionar o rectângulo para ir para outra secção na página ou alterar a escala da página.
- Barras de ferramentas configuráveis e acopláveis
- Menus emergentes e objectos de acção personalizáveis para todos os artigos de menu (que podem ser utilizados para construir uma barra de menus)
- Rodar, ampliar e reduzir, inverter e aperfeiçoar páginas
- Ajuste de contraste da imagem e equilíbrio de páginas no visualizador de Java. A janela Balance (Equilíbrio) inclui controlos de deslocamento e campos de texto para ajustar a luminosidade e o contraste.
- Navegação de páginas e documentos: primeiro, seguinte, anterior, última página e saltar para uma página.
- Adaptar largura, altura e à janela
- Impressão, incluindo selecção de páginas a imprimir e a capacidade de adicionar opções personalizadas para controlar a impressão de informações adicionais. As capacidades de personalização também permitem que uma aplicação faculte uma marca de água nas páginas impressas.
- Criar e editar anotações gráficas. Os tipos de anotações fornecidos incluem caixa, círculo, linha, seta, texto, destaque, nota persistente, caneta e marca. As anotações de marca podem ser texto ou imagens.

- Manipulação de páginas, incluindo copiar, reordenar, rodar e eliminar páginas em formatos de documentos de páginas-imagens e guardar o documento alterado.
- Multi-selecção de páginas nas miniaturas, útil para a rotação e outras manipulações das páginas.
- Copiar e colar anotações e páginas dentro de um documento e entre documentos.
- Copiar uma área seleccionada de uma página para a área de transferência (para colar noutra aplicação).
- Anular e repetir operações que modificam o documento como, por exemplo, editar anotações e manipular páginas
- Filtrar páginas: esta função permite a uma aplicação apresentar um subconjunto das páginas de um documento. Esta capacidade é facultada só através da interface de programação.
- Capacidade de exportar um documento para o sistema de ficheiros. O documento pode incluir anotações.
- Preferências controláveis pelo utilizador, que são guardadas localmente e são utilizadas em sessões seguintes do visualizador. As preferência incluem definições para documentos relacionadas com zoom e rotação, definições para o visualizador relacionadas com o tamanho e visibilidade das miniaturas, localização e visibilidade das barras de ferramentas e definições para anotações relacionadas com cada tipo de anotação (por exemplo, cor de contorno e de preenchimento). São facultadas classes para adicionar preferências personalizadas.

A Figura 32 é um exemplo de um visualizador de documentos genérico que utiliza todas as predefinições.

The screenshot displays the IBM DB2 Content Manager - Java Document Viewer interface. The window title is "IBM DB2 Content Manager - Java Document Viewer". The menu bar includes "File", "Edit", "View", "Image", and "Annotation". The toolbar contains various icons for document manipulation. The main area shows two document pages side-by-side.

**Left Page: Policy-10285265410 - Automobile Policy Declarations**

XYZ Insurance Company  
Freedom Ave.  
My Town, CA  
(888) 888-8888  
Page 1 of 1

Declaration Type: Renewal  
Insured Last Name: Heruda  
Insured First Name: Gabriela  
Policy Type: Member  
Insured Last Name: Heruda  
Insured First Name: Francisco  
Policy Number: 10285265410  
Street: 203 Karamia St.  
City: Marina  
State: CA  
Zip: 90546  
Policy Period From: 11/13/2000  
Policy Period To: 11/13/2001  
Process Date: 10/01/2000  
Insured Since: 1990  
Home Phone Number: (888) 288-2723  
Office Phone Number: (888) 147-3200

| Drivers |           |   |   |   |
|---------|-----------|---|---|---|
| 1       | 2         | 3 | 4 | 5 |
| Garza   | Francisco |   |   |   |

| Vehicles |        |            |           |               |
|----------|--------|------------|-----------|---------------|
| Item     | Make   | Model Year | Body Type | VIN           |
| 01       | Accord | 1998       | 4D SED    | 80AL0NLS85214 |

| Coverage / Premiums |                  |                 |             |      |         |
|---------------------|------------------|-----------------|-------------|------|---------|
| Coverage            | Liability Limits |                 | Item #1     | Item | Premium |
|                     | Each Person      | Each Occurrence |             |      |         |
| Bodily Injury       | 100,000          | 300,000         | \$145       |      |         |
| Medical Payments    | 10,000           | 10,000          | \$77        |      |         |
| Uninsured Motorist  | 100,000          | 300,000         | \$55        |      |         |
| Property Damage     |                  | 50,000          | \$94        |      |         |
| Comprehensive       |                  |                 | \$118       |      |         |
| Collision           |                  |                 | \$260       |      |         |
| Auto Club           |                  |                 | No Coverage |      |         |
| Total Premium       |                  |                 | \$641       |      |         |

Usage Deductible \$100.00  
Renewal Premiums \$741.00

**Right Page: Claim\_4-852369 - Car Insurance Claim Form**

XYZ Insurance Company  
Freedom Ave.  
My Town, CA  
(888) 888-8888

Claim Number: 4-852369

**Claimant**

First Name: George  
Last Name: Tolstoy  
Street: 372 Willow St.  
City: Marina  
State: CA  
Zip: 90548  
Home Phone Number: (888) 310-1395  
Office Phone Number: (888) 789-14  
Driver's License: B05678914

**Insured**

First Name: Gabriela  
Last Name: Heruda  
Driver's License: B02481308  
Policy Number: 10285265410

**Incident Information**

Date of Incident: 07/29/2001  
Driver Last Name: George  
Driver First Name: Tolstoy  
Driver's License: B05678914

**Vehicle**

Make: Acura  
Model: 4D SEDAN  
Year: 1998  
VIN: 80ALM1004P112385  
License Plate: 8JSA676

**Location of Incident**

Intersection: Highway 405 South bound close to Sepulveda but  
Team: Marina  
State: CA

**Description of Incident**

I was driving in lane 3 of Highway 405 South at approximately 10 MPH during rush hour on 07/29/2001. As the cars in front of me stopped, I proceeded to apply the break in my vehicle to come to a full stop. Soon after my car came to a fully stopped, I felt a hard impact on the rear end of my car. This impact was strong enough that made my car impact slightly the car in front of me.

Page 1 of 4  
Scale: 156%

Figura 32. Visualizador de documentos genérico

### Conceitos relacionados

“Trabalhar com as funções de manipulação de páginas” na página 490  
O toolkit de visualizador de Java faculta capacidades para editar documentos de páginas-imagens ao nível da página.

### Informações relacionadas

 Trabalhar com as funções de manipulação de páginas

---

## Arquitectura do visualizador

O toolkit do visualizador de Java contém um bean de Visualizador de Documentos e um bean de Serviços de Documentos.

Os beans facultam um mecanismo para integrar o visualizador em aplicações com base em JavaBeans do DB2 Information Integrator for Content.

O toolkit também contém as classes Visualizador de Documentos Genérico, Serviços de Documentos de Sequenciação e Serviços de Anotação. Com estas classes pode utilizar o visualizador nas aplicações em que não pode utilizar os beans como, por exemplo, numa visualização autónoma ou em situações de processo distribuído, em que a ligação a arquivos de conteúdos não é local.

As classe de Serviços de Documentos de Sequenciação gerem um conjunto de motores de processamento de documentos que interpretam documentos, apresentam páginas e facultam ao utilizador a capacidade de manipular as páginas de um documento.

Os motores facultam imagens de documento identificado em vários formatos como, por exemplo, TIFF e IOCA, bem como documentos de texto e texto complexo (rich text) e formatos de Microsoft Office. O utilizador pode também gravar motores de documentos adicionais e ligá-los à arquitectura de toolkit do visualizador para suportar formatos adicionais ou a apresentação alternativa de formatos de documentos.

Com as classes de Serviços de Anotações, é possível manipular as anotações. O motor de anotações identifica apenas formatos de anotação específicos de DB2 Content Manager. No entanto, pode gravar motores de anotações adicionais para lidar com outros formatos de anotações. O diagrama seguinte apresenta os componentes que caracterizam o toolkit do visualizador de Java:

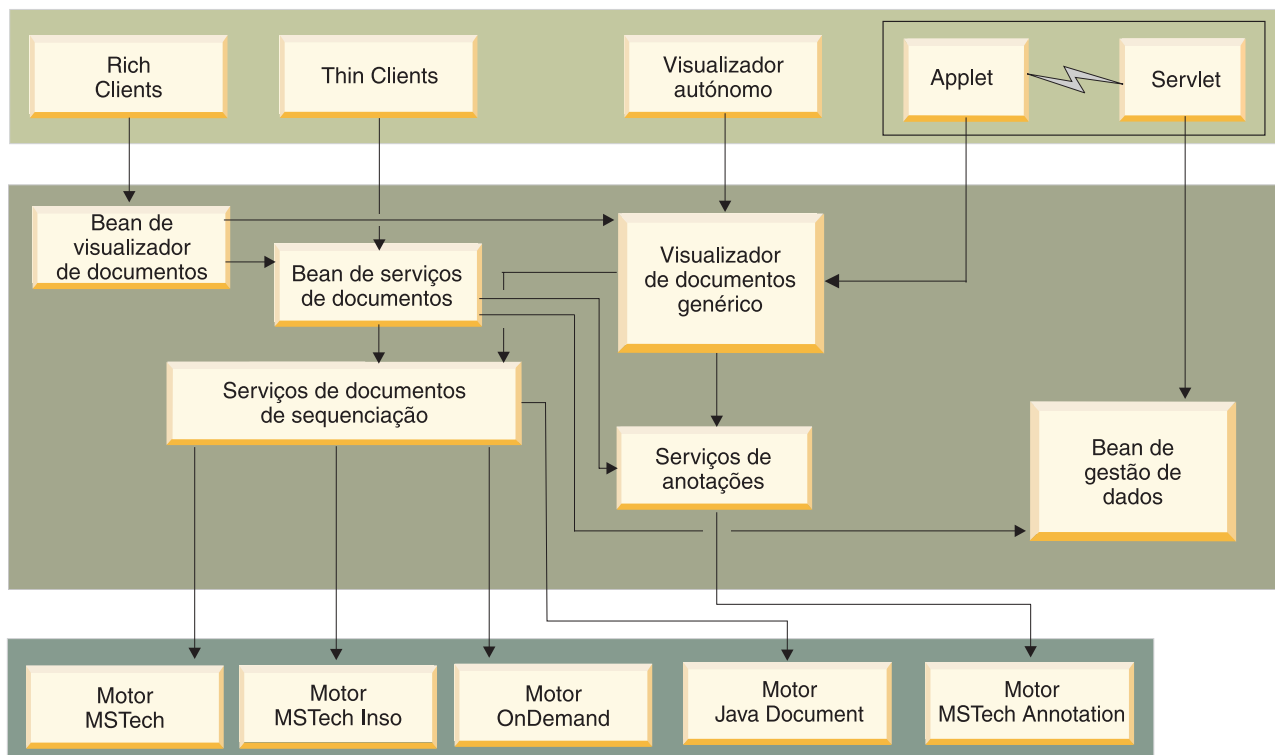


Figura 33. Os componentes no toolkit do visualizador de Java

## Os motores de documentos

Os motores de documentos são utilizados para processar documentos. Pode utilizar os motores de documentos para interpretar documentos, apresentar páginas e manipular as páginas de um documento. Um motor de documento pode ser utilizado com um conjunto de tipos de documento como, por exemplo, TIFF e JPEG.

O toolkit do visualizador de Java faculta os seguintes motores de documentos:

### **Motor de Documento MS-Tech (CMBMSTechDocumentEngine)**

Este motor devolve páginas como imagens e processa tipos de conteúdos normalmente localizados no DB2 Content Manager. Os tipos de documento suportados por este motor incluem TIFF e MO:DCA. Este motor também suporta GIF, JPEG e texto simples.

### **Motor MS-Tech INSO (CMBMSTechInsoEngine)**

Este motor suporta Microsoft Office, Lotus SmartSuite e outros formatos de documentos do Office.

### **Motor DB2 Content Manager OnDemand (CMBODDocumentEngine)**

Este motor converte dados de linha em texto simples e combina grandes documentos de objectos AFP num único documento AFP. Também contém uma ponte para o toolkit AFP2Web (disponível em separado) para converter páginas de documentos AFP em HTML.

### **Motor de Documentos Java**

Este motor converte documentos que são representados por URLs em HTML com uma ligação ao URL.

Os motores referidos acima são interfaces públicas, mas o utilizador não deve efectuar programações directamente nas interfaces. Deve utilizar as interfaces facultadas pelo bean Document Services ou pela classe Streaming Doc Services para aceder à funcionalidade facultada pelos motores de documentos.

Alguns destes motores não são só de Java e possuem limitações de portabilidade, o que pode limitar a utilização do toolkit em alguns sistemas operativos. Os motores de Documento MS-Tech e de Java são Só de Java e podem ser utilizados na maioria dos sistemas operativos. Os outros motores contêm uma lógica específica de sistema operativo que restringe a respectiva utilização ao Windows.

#### **Conceitos relacionados**

“Tipos e conversões de documentos suportados no toolkit do visualizador de Java” na página 487

A tabela seguinte apresenta um resumo dos tipos e conversões de documentos suportados por predefinição (utilizando os motores de conversão fornecidos).

#### **Informações relacionadas**

 Tipos e conversões de documentos suportados no toolkit do visualizador de Java

## **O motor de anotações**

O toolkit do visualizador de Java faculta o motor de anotação MS-Tech para obter e actualizar anotações de DB2 Content Manager.

O motor de anotações MS-Tech suporta anotações do DB2 Content Manager Versão 8.2, Versão 8.3 e Versão 8.4, do IBM Content Manager Versão 8.1, do Content Manager Versão 7 e do VisualInfo for AS/400.

## **Arquitecturas de visualizador exemplo**

Para as várias formas de utilização do toolkit do visualizador de Java, consulte, nesta secção, as cinco concepções arquitectónicas de exemplo que utilizam o toolkit do visualizador e os beans de Java.

Tenha em atenção que os exemplos nesta secção não constituem as únicas formas de utilização do toolkit.

### **Visualizador autónomo**

Pode utilizar o visualizador de documentos genérico para implementar um visualizador autónomo. Pode utilizar um visualizador autónomo para visualizar ficheiros ou documentos que obtiver de URLs. Pode utilizar um visualizador autónomo como uma paleta numa página da Web ou para visualizar documentos que obteve por correio electrónico.

A Figura 34 na página 472 ilustra a arquitectura de um visualizador autónomo.

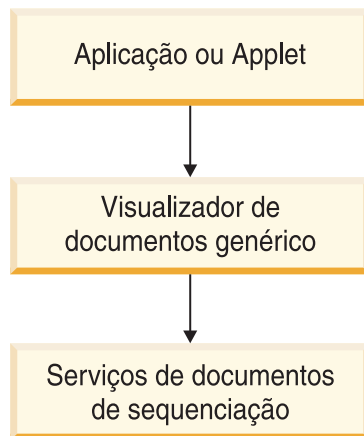


Figura 34. Visualizador autónomo

### Aplicação Java

Para criar uma aplicação Java de produção, utilize os beans visuais do DB2 Information Integrator for Content, que utilizam o bean visual CMBDocumentViewer .

Este bean utiliza o visualizador de documentos genérico internamente para visualizar documentos. O bean CMBDocumentViewer também pode iniciar outros visualizadores para ver documentos. Tenha em atenção que estes visualizadores poderão depender do sistema operativo utilizado. A Figura 35 ilustra a arquitectura que pode utilizar para construir uma aplicação Java.

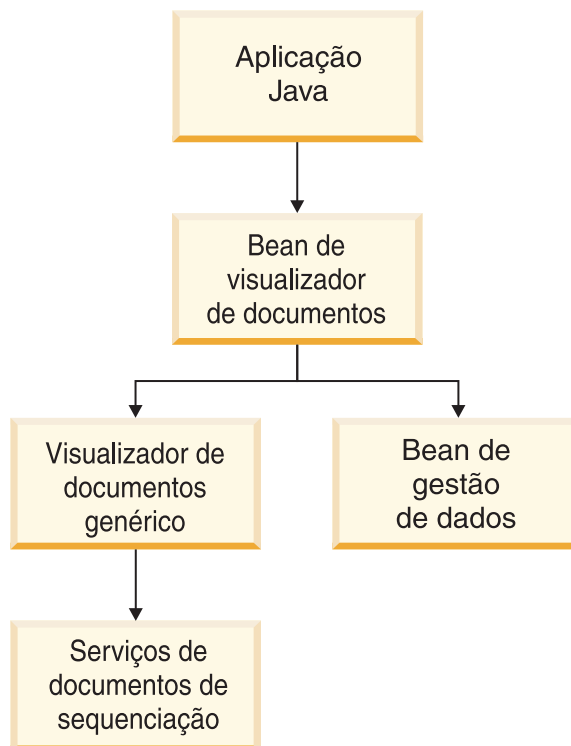


Figura 35. Aplicação Java

## Thin client

Pode utilizar o bean `CMBDocumentServices` para executar conversões de documentos do lado do servidor numa aplicação baseada na Web. Pode converter documentos de tipos de conteúdo que não são processados pelo navegador (documentos que necessitam de um plug-in ou de uma aplicação nativa para serem iniciados) para tipos de conteúdo processados nativamente pelo navegador como, por exemplo, HTML, GIF, JPEG ou para os quais os plug-ins estão prontamente disponíveis como, por exemplo, PDF.

A Figura 36 ilustra uma arquitectura de um thin client.

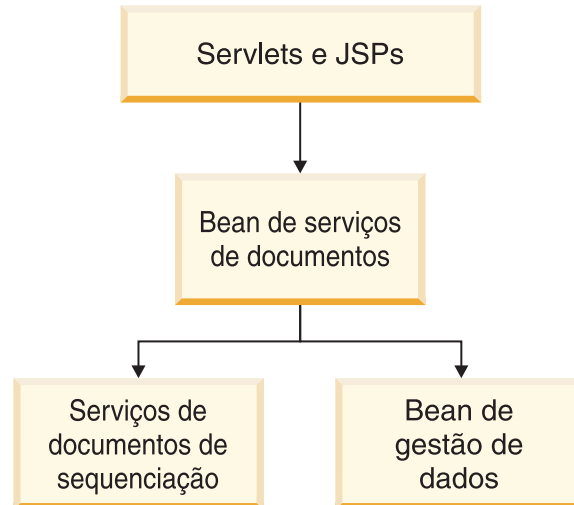


Figura 36. Thin client

## Applet e servlet

Uma aplicação baseada na Web também pode facultar capacidades de visualização de documentos e de edição de anotações utilizando uma abordagem de applet e servlet. A applet do visualizador `eClient` utiliza esta arquitectura.

A applet pode utilizar o visualizador de documentos genérico para visualizar o documento. Alguns tipos de documentos são armazenados em várias partes do servidor de conteúdos para tornar mais eficaz a partilha da maioria das informações, tais como formulários de segundo plano. As partes adicionais são solicitadas pelo visualizador de documentos genérico quando for necessário. A applet que contém o visualizador atende os pedidos enviando pedidos de HTTP para a servlet.

No que diz respeito à servlet, o servidor de conteúdos que utiliza o bean `CMBDataManagement` obtém as informações solicitadas. A Figura 37 na página 474 ilustra uma arquitectura de applet ou servlet.



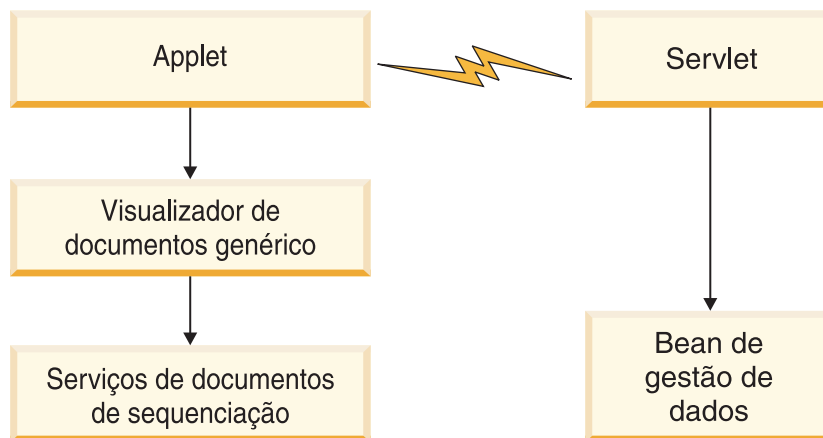


Figura 37. Applet e servlet

### Modo dual com applet ou servlet

Pode utilizar uma variação dos exemplos facultados para aplicações de visualização com base na Web. Utilize CMBDocumentServices no servidor e na applet.

Esta abordagem é útil quando os documentos não são apresentados na applet, mas são convertidos no servidor, o que poderá suceder quando os motores de documentos subjacentes da applet e do servidor possuem capacidades diferentes.

Por exemplo, se o servidor for Windows NT ou Windows 2000 e a applet estiver em execução em Linux, a applet poderá não apresentar todos os tipos de documentos. A applet apresenta formatos de documentos que suporta, tais como TIFF. Para tipos que a applet não consegue apresentar, tais como formatos Office, esta solicita conversão da servlet. Segundo a perspectiva do utilizador final, a mesma interface é apresentada com a mesma funcionalidade. No entanto, o rendimento do servidor para documentos convertidos poderá ser mais lento do que para documentos apresentados localmente.

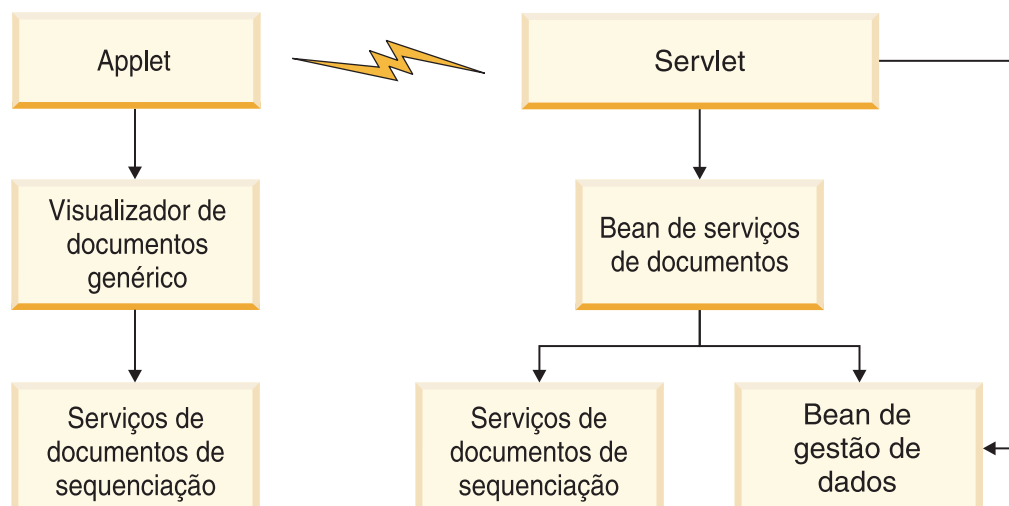


Figura 38. Visualizador de modo dual

---

## Criar um visualizador de documentos

Pode utilizar programaticamente as classes e APIs do toolkit do visualizador de Java para construir a sua própria aplicação para visualizar os documentos.

Para criar um visualizador de documentos, irá trabalhar principalmente com a classe `CMBGenericDocViewer`, que é a interface gráfica de utilizador (GUI) para o visualizador de documentos. Esta GUI é baseada na Java Foundation Class (JFC). O `CMBGenericDocViewer` é uma GUI que depende de outras classes para processar a apresentação de documentos e as funções de anotação. Estas funções são facultadas por duas classes não visuais: a classe de serviços de documentos `CMBStreamingDocServices` e a classe de serviços de anotação `CMBAnnotationServices`.

## Criar uma aplicação ou applet de visualizador autónomo

Este é um exemplo de como construir um visualizador de documentos que seja uma aplicação autónoma.

À medida que vai concluindo os passos seguintes, consulte o manual *Application Programming Reference* no information center. Consulte também exemplos de aplicação do visualizador autónomo, `TGenericDocViewer.java` e de applet do visualizador, `TViewerApplet.java`, localizados no directório `samples`.

Para começar, inicie o objecto `CMBGenericDocViewer`. Para iniciar o objecto `CMBGenericDocViewer`, tem de criar uma instância de cada um dos objectos de classe `CMBStreamingDocServices` e `CMBAnnotationServices`, implementando as seguintes classes de repetição de marcação:

- Classe abstracta `CMBStreamingDocServicesCallbacks`
- Classe abstracta `CMBAnnotationServicesCallbacks`

Para criar uma aplicação ou applet de visualizador autónomo, execute os seguintes procedimentos:

1. Crie uma classe que expanda `CMBStreamingDocServicesCallbacks`. Esta classe é utilizada pelo `CMBGenericDocViewer` para obter e actualizar partes e recursos de documentos. Inclui métodos para obter uma forma de segundo plano de um documento e o privilégio imprimir de um documentos. No exemplo de código, esta classe é denominada `StreamingDocServicesCallbacks`.
2. Crie uma instância de `CMBStreamingDocServices` utilizando as marcações de repetição implementadas no primeiro passo. No exemplo de código, este é o objecto `docServices`.
3. Crie uma classe que expanda `CMBAnnotationServicesCallbacks`. Esta classe é utilizada pelo `CMBGenericDocViewer` para obter e actualizar anotações. Tem de facultar a implementação dos métodos necessários, como obter e actualizar. Pode ignorar os métodos que não forem relevantes para o tipo de documentos visualizados. No exemplo de código, esta é a classe `AnnotationServicesCallbacks`.
4. Crie uma instância de `CMBAnnotationServices` utilizando as marcações de repetição implementadas no terceiro passo. No exemplo de código, este é o objecto `annoServices`.
5. Crie uma instância de `CMBGenericDocViewer`.
  - a. Inicialize a mesma com `CMBStreamingDocServices`, `CMBAnnotationServices` e com o objecto de propriedades de configuração.

- b. Transmita nulo para o objecto de propriedades, caso pretenda utilizar o objecto de configuração predefinido. A configuração predefinida é construída a partir do ficheiro de configuração predefinida `cmbviewerconfiguration.properties`, incluído em `cmbview81.jar`.

A cadeia de mensagens da configuração predefinida, como etiquetas de acção e sugestões de ferramentas, é lida na linguagem específica do ficheiro `cmbViewerMessages.properties` em `cmbview81.jar`.

6. Adicione o visualizador à estrutura principal da aplicação ou da applet.
7. Prepare uma barra do menu obtendo as acções do visualizador de documentos genérico e adicione o menu à aplicação, como mostra a secção relativa a personalização.
8. Implemente as interfaces de ouvinte do visualizador como, por exemplo, `CMBGenericDocSaveListener` e `CMBGenericDocClosedListener`. Não se esqueça de utilizar o método `genericDocSave` do `CMBGenericDocSaveListener` para guardar um conjunto de documentos e de anotações.
9. Adicione os ouvintes de visualizador utilizando os métodos adicionar ouvintes do `CMBGenericDocViewer` como, por exemplo, `addDocSaveListener`.
10. Especifique os motores e respectiva ordem utilizando o ficheiro propriedades de motor, denominado `cmbviewerengine.properties`. O objecto `CMBStreamingDocServices` depende das classes de motor de documentos para processar a manipulação de imagens. O ficheiro `cmbviewerengine.properties` está incluído em `cmbview81.jar`. Pode também fazer isto de forma programática, como demonstrado no seguinte fragmento de código:

```
// Create engine properties
engineProperties = new Properties();
engineProperties.put("ENGINES", "2");
engineProperties.put("ENGINE1_CLASSNAME",
 "com.ibm.mm.viewer.mstech.CBMSTechDocumentEngine");
engineProperties.put("ENGINE2_CLASSNAME",
 "com.ibm.mm.viewer.CMBJavaDocumentEngine");
// Create a streaming doc services object and associate
// the document engine classes
docServices = new CMBStreamingDocServices(
 new StreamingDocServicesCallbacks(), engineProperties);
```

## Exemplo

O código seguinte mostra como criar o objecto `CMBGenericDocViewer`:

```
// Create streaming doc services object
docServices = new CMBStreamingDocServices(
 new StreamingDocServicesCallbacks(), null);

// Create annotation services object
annoServices = new CMBAnnotationServices(
 new AnnotationServicesCallbacks());

// Create generic doc viewer object
genericDocViewer = new CMBGenericDocViewer(
 docServices, annoServices, configProps);

// Add generic doc viewer GUI to content pane
getContentPane().add(genericDocViewer, BorderLayout.CENTER);
```

### Conceitos relacionados

“Personalizar o visualizador” na página 477

É incluído um visualizador no produto, fornecido pelo `CMBGenericDocViewer`. Pode personalizar a função facultada pelo visualizador através do ficheiro de propriedades configuração.

## Informações relacionadas



Personalizar o visualizador

## Trabalhar com documentos e anotações

Esta secção apresenta as instruções de utilização dos componentes do toolkit de visualizador para trabalhar como documentos e anotações.

Efectue os seguintes passos para carregar o documento ou qualquer anotação e visualizar um documento no visualizador:

1. Invoque `loadDocument` em `CMBStreamingDocServices` para carregar um documento para os serviços de documento. Os serviços de documento são o nível não visual do visualizador e facultam um modelo para trabalhar com o documento, de forma visual ou não visual. Esta acção devolve uma instância de `CMBDocument`.
2. Invoque `loadAnnotationSet` em `CMBAnnotationServices` para carregar quaisquer anotações para os serviços de anotação. Os serviços de anotação facultam um modelo para o conjunto de anotações de um documento e permitem que as anotações sejam manipuladas de forma visual ou não visual. Esta acção devolve uma instância de `CMBAnnotationSet`.
3. Invoque `showDocument` em `CMBGenericDocViewer` para apresentar o documento no visualizador de documentos genérico. O visualizador de documentos genérico é o componente visual do visualizador.
4. Invoque `terminate` no `CMBGenericDocViewer` para fechar a aplicação do visualizador. isto fechar todos os documentos e limpa a interface de utilizador.

## Exemplo de documentos e anotações

```
// Load the document into document services
CMBDocument document =
 docServices.loadDocument(
 inStream,
 nParts,
 mimetype,
 mimetype,
 null,
 null);

int position = document.getAnnotationPosition();

// Load the annotations into annotation services.
CMBAnnotationSet annotationSet =
 annoServices.loadAnnotationSet(
 annoStream,
 "application/vnd.ibm.modcap",
 position,
 1,
 0);

// Show the document in the generic doc viewer.
genericDocViewer.showDocument(document, annotationSet,
 filename);
```

## Personalizar o visualizador

É incluído um visualizador no produto, fornecido pelo `CMBGenericDocViewer`. Pode personalizar a função facultada pelo visualizador através do ficheiro de propriedades configuração.

O ficheiro de propriedades de configuração contém uma série de parâmetros que controla o que a barra de ferramentas apresenta, onde apresenta e os botões de ferramentas disponíveis. Pode ainda personalizar estas funções como menus emergentes e como resolução de apresentação predefinida.

Pode personalizar o visualizador, facultando um objecto `configProperties` personalizado quando constrói o visualizador. Para ver um exemplo de um visualizador personalizado, consulte o exemplo de aplicação de visualizador `TGenericDocViewer`. O código fonte do visualizador exemplo, `TGenericDocviewer.java`, bem como o ficheiro de propriedades personalizadas que utiliza, `TViewerDefaultConfiguration.properties` estão localizados no directório `samples`. Durante a leitura desta secção, consulte o ficheiro `TViewerDefaultConfiguration.properties` com frequência.

A configuração do visualizador contém barras de ferramentas predefinidas como, por exemplo, **Operation (Operação)**, **Annotation (Anotação)** e **Page (Página)**, uma série de ferramentas predefinidas e algumas propriedades que controlam determinados comportamentos do visualizador como, por exemplo, manipulação de páginas, selecção múltipla, etc. Com o ficheiro de configuração do visualizador personalizado, pode controlar quais as barras de ferramentas visíveis, personalizar as barras de ferramentas existentes, criar novas barras de ferramentas e acções e activar ou desactivar qualquer comportamento do visualizador configurável.

Pode personalizar o objecto de configuração predefinido, `CMBViewerConfiguration.properties`, localizado no ficheiro `cmbview81.jar` ou pode criar um novo objecto de configuração.

## Facultar o objecto `configProperties` ao visualizador

O seguinte fragmento de código demonstra como facultar um objecto `configProperties` personalizado ao visualizador.

```
//Load the custom configuration file.
Properties customProperties = null;
try {
 URL url =
 this.getClass().getClassLoader().getResource(
 "TViewerDefaultConfiguration.properties");
 InputStream configFile = null;
 Properties defaultProperties = null;
 if (url != null) {
 configFile = url.openStream();
 defaultProperties = new Properties();
 defaultProperties.load(configFile);
 customProperties = defaultProperties;
 }
} catch (Exception e) {
 System.out.println(e);
 e.printStackTrace();
}
// Create the generic doc viewer and
// add it to the window
genericDocViewer =
 new CMBGenericDocViewer
 (docServices, annoServices, customProperties);
```

## Personalizar a barra de ferramentas

O seguinte fragmento de código corresponde à secção do ficheiro `TViewerDefaultConfiguration.properties` que personaliza a barra de ferramentas.

Com a utilização de várias propriedades, pode personalizar quais as barras de ferramentas que aparecem, as respectivas posições, as ferramentas que aparecem, a respectiva ordem de apresentação e respectivos agrupamentos. A propriedade `toolbarname.position` especifica a posição da barra de ferramentas em relação ao documento. A posição da barra de miniatura também é especificada na mesma forma. A propriedade `toolbarname.tools` especifica a lista de ferramentas, separadas por uma vírgula, que aparecem na barra de ferramentas. As ferramentas aparecem na barra de ferramentas pela mesma ordem que forma especificadas. Para agrupar as ferramentas utilize a palavra-chave `separator`.

```
//The Toolbar defines all the toolbars listed
Toolbars=OperationToolbar,PageToolbar,AnnoToolbar

//You can customize the position and the
// tools of each toolbar
OperationToolbar.position=NORTH
OperationToolbar.tools=new_doc,open_doc,save_doc,
save_as,separator,
export_doc,print,separator,cut,copy,paste,separator,
undo,redo,separator,
rotate_90,rotate_180,rotate_270,rotate_pages,separator
,zoom_in,zoom_out,
zoom_custom,separator, \

fit_height,fit_width,fit_window,fit_actualsize,separator,
enhance,invert,
separator,hide_show,showhidethumb,separator,close_doc,
close_all_doc,separator,help

AnnoToolbar.position=WEST
AnnoToolbar.tools=selectArea,pointer,separator,Arrow,Circle,
Highlight,
Line,Note,Pen,Rect,Stamp,Text,eraser,separator, \
move_front,send_back,properties

PageToolbar.position=SOUTH
PageToolbar.tools=page_first,page_prev,goto_page,
page_next,page_last,
separator, \
doc_first,doc_prev,doc_next,doc_last

Thumbnailbar.position=EAST
```

Todas as ferramentas enumeradas no fragmento de código Personalizar a barra de ferramentas, à excepção de `open_doc`, são acções predefinidas e é o visualizador que as implementa. A ferramenta `open_doc` é uma ferramenta personalizada que pode ser adicionada. Pode ainda especificar onde a ferramenta aparece, adicionando a ferramenta `open_doc` à `OperationToolbar` depois da ferramenta `new_doc`.

## Adicionar uma ferramenta personalizada

O seguinte fragmento de código demonstra como adicionar uma ferramenta personalizada:

```
open_doc.label=Open
open_doc.tooltip=Open
Document open_doc.icon=OpenDocument_normal.gif
open_doc.key=control Open_doc.className=TGenericDocViewer$OpenAction
```

O visualizador cria a ferramenta como `CMBViewerAction`. Terá de fornecer a classe que faculta a acção da ferramenta personalizada.

## Definir a classe de acção para uma ferramenta personalizada

O seguinte fragmento de código demonstra como definir a classe que faculta a acção da ferramenta personalizada:

```
// This class handles the Open action. This is a custom action
// added to the operations toolbar. It must be static since it is
// instantiated from within CMBGenericDocViewer.
public static class OpenAction extends CMBViewerAction {
 public OpenAction(CMBGenericDocViewer viewer) {
 super(viewer);
 }
 public OpenAction(CMBGenericDocViewer viewer, String name) {
 super(viewer, name);
 }
 public OpenAction(CMBGenericDocViewer viewer, String name, Icon icon) {
 super(viewer, name, icon);
 }
 // @see java.awt.event.ActionListener#actionPerformed(ActionEvent)
 public void actionPerformed(ActionEvent e) {
 // Find the TGenericDocViewer instance(it is a parent of CMBGenericDocViewer)
 TGenericDocViewer gdvFrame = (TGenericDocViewer) getView()
 .getParent()
 .getParent()
 .getParent()
 .getParent();
 JFileChooser chooser = new JFileChooser(System.getProperty("user.dir"));
 chooser.setMultiSelectionEnabled(true);
 int ret = chooser.showOpenDialog(gdvFrame);
 if (ret == JFileChooser.APPROVE_OPTION) {
 File [] files = chooser.getSelectedFiles();
 for (int i = 0; i < files.length; i++) {
 gdvFrame.openDocument(files[i].getAbsolutePath(), i == 0);
 }
 }
 }
}
```

## Utilizar acções de visualizador na aplicação

O seguinte fragmento de código demonstra como adicionar uma acção de visualizador à barra de menus de uma aplicação.

```
//create a menubar and menu
JMenuBar mainMenuBar = new JMenuBar();
JMenu viewerMenu = new JMenu("Viewer");
mainMenuBar.add(viewerMenu);
// get an action object that has been loaded into the viewer
Action zoomInAction = genericDocViewer.getAction("zoom_in");
// add the viewer action to the menu
viewerMenu.add(zoomInAction);
// add the menu bar
getRootPane().setJMenuBar(mainMenuBar);
```

## Personalizar menus emergentes

O visualizador também fornece menus emergentes que pode personalizar. Os menus emergentes incluem **Page (Página)**, **Thumbnail (Miniatura)**, **Annotation (Anotação)**, **Document Tab (Separador de documentos)** e **Select Area (Área de selecção)**. Os nomes dos menus emergentes são fixos e indicam onde o menu aparece. As ferramentas disponíveis para cada menu emergente estão enumeradas no valor da propriedade `popupmenuName.items`, que pode ser personalizado.

O seguinte fragmento de código corresponde à secção do menu emergente do ficheiro `TViewerDefaultConfiguration.properties`:

```
//Popup Menu
//annotation popup
annotationpopup.items=cut,copy,paste,delete,separator,
move_front,send_back,separator,properties
//thumbnail popup
thumbpopup.items=cut,copy,paste,delete,separator,deselectAll,
selectAll,separator,rotate
//page popup
pagepopup.items=cut,copy,paste,delete,separator,rotate,zoom,
fit,navigate,separator,paste
//doc tab popup (appears when right-click on document tabs)

doctabpopup.items=close_doc,close_all_doc,separator,
save_doc,print
//select area popup

selectareapopup.items=copy
```

## Adicionar submenus

Pode adicionar submenus aos menus emergentes criando, para tal, entradas de `label` e `item` para os valores de `popupmenu.name.item`. O seguinte fragmento de código demonstra como adicionar submenus para rodar, ampliar e reduzir, adaptar e navegar:

```
//submenus (appear on some of the popup menus)
rotate.label=Rotate Pages
zoom.label=Zoom
fit.label=Fit
navigate.label=Navigate
rotate.items=rotate_90,rotate_180,rotate_270
zoom.items=zoom_in,zoom_out,zoom_custom
fit.items=fit_height,fit_width,fit_window,fit_actualseize
navigate.items=page_first,page_prev,goto_page,page_next,
page_last,showhidethumb
```

As secções seguintes apresentam uma descrição de outras opções do visualizador que podem ser personalizadas. Cada uma destas opções tem uma explicação nos comentários que acompanham cada propriedade.

## Inverter documentos

Defina a propriedade `Document.invert` como `true` para inverter automaticamente um documento assim que é aberto pela primeira vez. Defina como `false`, para que os documentos não sejam invertidos automaticamente quando são abertos pela primeira vez. Tenha em atenção que, por predefinição, os documentos não estão invertidos. Exemplo:

```
Document.invert=false
```

## Ampliar documentos

Para ampliar automaticamente documentos quando são abertos, defina a propriedade `Document.enhance` como `true`. A predefinição é ampliar documentos. Note que quando amplia determinados documentos de imagens pode aumentar a utilização da memória. Exemplo:

```
Document.enhance=true
```



## Rodar documentos

O comportamento predefinido é não rodar o documento quando este é aberto. Para rodar automaticamente um documento quando é aberto, defina a propriedade `Document.rotate` de acordo com os seguintes graus:

- Não rodar: `rotate_0`
- Rodar 90 graus: `rotate_90`
- Rodar 180 graus: `rotate_180`
- Rodar 270 graus: `rotate_270`

Exemplo:

```
Document.rotate=rotate_0
```

## Apresentar anotações

Para apresentar automaticamente todas as anotações associadas ao documento quando este é aberto, defina a propriedade `Annotations.show` como `true`. A predefinição é `true`. Defina a propriedade como `false` se pretender que as anotações fiquem ocultas quando o documento é aberto. Exemplo:

```
Annotations.show=true
```

## Ampliar e Reduzir

Para ampliar e reduzir um documento, defina a propriedade `Zoom.factor` como um valor numérico maior do que zero e menor do que cem. Este valor representa uma percentagem. O valor predefinido de ampliação e redução é dez por cento.

Exemplo:

```
Zoom.factor=10
```

## Adaptar à janela

Para adaptar a página do documento a uma janela, defina a propriedade `Page.fit` de acordo com as opções seguintes:

- `fit_width` : Adaptar páginas à largura da vista
- `fit_height`: Adaptar páginas à altura da vista
- `fit_in_window`: Adaptar toda a página à vista
- `fit_none`: Apresentar a página com a ampliação inicial (a predefinição)

Exemplo:

```
Page.fit=fit_width
```

## Redimensionar miniaturas

As propriedades `ThumbnailSize` definem a largura e a altura de cada vista de miniaturas. Os valores no exemplo seguinte são indicados para uma página com 8,5x11 polegadas. Exemplo:

```
ThumbnailSize.width=60
ThumbnailSize.height=77
```

O tamanho da miniatura poderá ser também um factor desta largura e altura utilizando `ThumbnailSize`. Exemplo:

```

small - actual thumbnail size would be ThumbnailSize.width/2,
ThumbnailSize.height/2
medium - actual thumbnail size would be ThumbnailSize.width,
ThumbnailSize.height
large - actual thumbnail size would be ThumbnailSize.width * 2,
ThumbnailSize.height * 2
By default ThumbnailSize is medium
ThumbnailSize=medium

```

## Manipular páginas

Pode copiar, cortar, colar, eliminar e rodar a página de um documento e fazer com que todas estas alterações sejam guardadas automaticamente. Para guardar todas as alterações efectuadas a uma página, defina a propriedade `PageManipulation` como `true`. Exemplo:

```
PageManipulation=true
```

## Personalizar o comportamento do visualizador

O fragmento de código demonstra como personalizar outras opções de comportamento do visualizador.

```

MultiplePageSelection enables multi-select of pages
within the thumbnails.
MultiplePageSelection=true

Set NewDocument.mimetype property to the MIME type
to be used when creating new empty document.
Examples include, image/gif, image/jpeg,
image/tiff, application/pdf, text/plain,
text/richtext, application/vnd.lotus_wordpro
NewDocument.mimetype=image/tiff

Set NewDocument.annotationtype property to a string
constant for the
type of annotations when creating a new empty document.
For example, 'application/vnd.ibm.modcap'
for IBM DB2 Content Manager annotations
or a two letter representation of the server type such
as 'DL', 'OD', 'V4'
NewDocument.annotationtype=application/vnd.ibm.modcap
#PrintMargins
#Specify the print margins for a mimetype.
#Default is 0,0,0,0 (top, left, bottom, right)
#printmargins.image/tiff=1,1,1,1

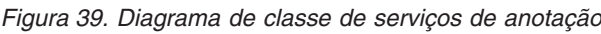
```

---

## Trabalhar com serviços de anotação

O toolkit do visualizador de Java do DB2 Information Integrator for Content 8.4 faculta capacidades para produzir e converter anotações de documentos.

Semelhantes aos serviços de documentos, os motores de anotação ligáveis facultam funções adicionais que pode utilizar nas suas aplicações para interpretar diferentes tipos de anotações. A Figura 39 na página 484 mostra o diagrama de classe de serviços de anotação.



CMBAnnotationServices fornece as principais interfaces utilizadas no toolkit do visualizador de Java.

Os serviços de anotação permitem ao utilizador carregar, manipular e guardar objectos de anotação utilizando os serviços de anotação, independentemente dos programas emissores. Para trabalhar com anotações, o utilizador deve transmitir os dados de anotação como uma sequência e ligar um motor de anotação adequado, que converte os objectos de anotação em instâncias CMBPageAnnotation. Em seguida, poderá manipular e editar as anotações e guardá-las no programa emissor original, no seu formato original.

A Figura 40 na página 485 ilustra como o visualizador genérico de documentos e os serviços de documentos e anotação se encaixam. Os serviços de documento e os serviços de anotação são utilizados pelo visualizador de documentos genérico. No entanto, as marcações de repetição para ambos são fornecidas pela aplicação que incorpora o visualizador (que é o bean `visual.CMBDocumentViewer` na Figura 40 na página 485). Os serviços de documento utilizam as instâncias `CMBDocument` e `CMBPage` para representar documentos e páginas de documentos. Os serviços de

anotação utilizam CMBAnnotationSet e CMBPageAnnotation (e as respectivas subclasses) para representarem as anotações num documento e cada anotação individual.

**Não se esqueça:** Não existe uma relação directa entre CMBDocument e CMBAnnotationSet ou entre CMBPage e CMBPageAnnotation. A associação é mantida dentro do visualizador de documentos genérico. Isto permite aos dois serviços serem utilizados independentemente um do outro.

Para implementar o motor de anotação, o utilizador deve expandir a classe abstracta CMBAnnotationEngine. O motor de anotação utiliza as interfaces CMBAnnotationServicesCallbacks e CMBAnnotationEngineCallbacks para comunicar com uma aplicação e com serviços de anotação. O motor de anotação no DB2 Information Integrator for Content 8.4 só reconhece o formato de anotação DB2 Content Manager. Este formato de anotação é utilizado pelos servidores de fundo do Content Manager Versão 7, DB2 Content Manager Versão 8.1, DB2 Content Manager Versão 8.2, DB2 Content Manager Versão 8.3, DB2 Content Manager Versão 8.4 e VisualInfo for AS/400.

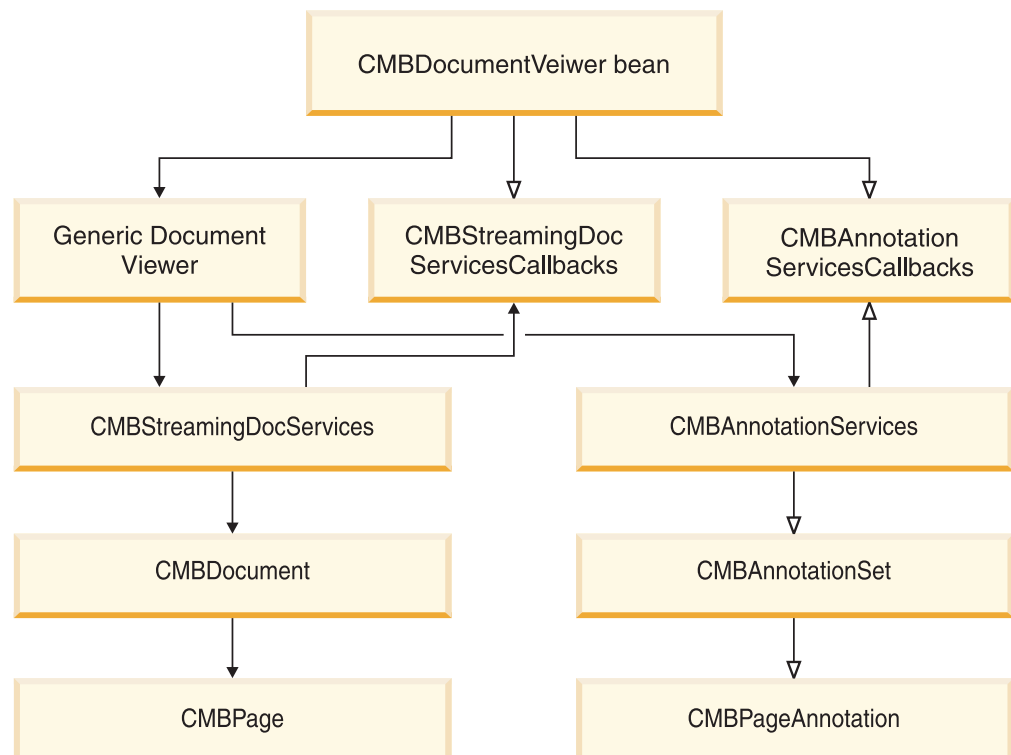


Figura 40. Associação entre os serviços de anotação e o visualizador genérico de documentos

## Compreender o suporte de edição de anotações

O padrão de concepção do Model View Controller (MVC) é utilizado para implementar a funcionalidade de edição das anotações.

CMBAnnotationSet funciona como o modelo que representa os dados de anotação. CMBAnnotationSet contém métodos que funcionam sobre os dados, mas não tem interfaces do utilizador. A classe CMBAnnotationSet mantém a lista de objectos CMBPageAnnotation. Cada documento é associado a um objecto

CMBAnnotationSet que representa as suas anotações. CMBAnnotationView funciona como a vista que apresenta os dados do modelo ao utilizador. CMBAnnotation processa toda a elaboração de anotações no componente de vista (um JComponent). CMBAnnotationComponent é uma classe auxiliar que pode ser utilizada como componente de vista no qual as anotações são elaboradas. O controlador faz parte do toolkit de visualizador e processa eventos do rato e do teclado para criação e edição de anotações.

CMBPageAnnotation é a classe de base que descreve uma única anotação numa página de um documento. Se necessitar de definir tipos adicionais de anotações gráficas, terá de expandir a classe CMBPageAnnotation. Existem nove tipos de anotações do DB2 Content Manager que pode criar: CMBArrowAnnotation, CMBCircleAnnotation, CMBHighlightAnnotation, CMBLineAnnotation, CMBNoteAnnotation, CMBPenAnnotation, CMBRectAnnotation, CMBStampAnnotation e CMBTextAnnotation.

**Nota:** Pode utilizar serviços de anotação em aplicações que não são GUI.

## Construir uma aplicação utilizando os serviços de anotação

Esta secção inclui os passos a seguir e as APIs a utilizar para construir uma aplicação de serviços de anotação.

Consulte Application Programming Reference para obter mais informações relativas à utilização da API.

1. Crie uma subclasse de CMBAnnotationServicesCallbacks para implementar os métodos abstractos de processamento de chamadas de anotações.

2. Crie uma instância de CMBAnnotationServices.

```
CMBAnnotationServices annoServices = new
CMBAnnotationServices(annoServicesCallbacks);
```

3. Obtenha uma instância de CMBAnnotationSet carregando uma sequência de anotação.

```
CMBAnnotationSet annotationSet = annoServices.loadAnnotationSet(annoStream,
format, documentResolution, annotationPartNumber);
```

4. Prepare a vista da anotação.

```
CMBAnnotationComponent annoComponent = new CMBAnnotationComponent();
CMBAnnotationViewer annoView = annoServices.prepareAnnotationView(
 annoComponent, annotationSet);
annoView.refreshEntireDrawingArea();
```

5. Adicione o componente de anotação a um contentor Swing como, por exemplo, JFrame ou JPanel, da aplicação.

6. Agora poderá utilizar a API de serviços de anotação para adicionar novas anotações, editar ou eliminar anotações já existentes.

```
annoServices.prepareToAddAnnotation();
annoServices.addAnnotation();
annoServices.removeAnnotation();
annoServices.reorderAnnotation();
...
```

7. Guarde as anotações modificadas.

```
annoServices.saveAnnotationset(annotationSet);
```

Um exemplo de serviços de anotação, TAnnotationServices encontra-se disponível no directório samples.

## Adicionar um tipo de anotação personalizado à aplicação do utilizador

Pode adicionar um tipo de anotação personalizado à aplicação.

Para adicionar um tipo de anotação personalizada aos serviços de anotação, conclua os seguintes passos. Consulte Application Programming Reference para obter mais informações relativas à utilização da API.

**Não se esqueça:** Não é possível guardar esta anotação personalizada, uma vez que o motor de anotação incorporado não suporta nem compreende a mesma.

1. Crie uma subclasse de CMBPageAnnotation.  

```
public class TImageAnnotation extends CMBPageAnnotation
```
2. Defina uma constante para a anotação personalizada com um valor superior a cem. O limite entre 1 e 99 é reservado.  

```
public static final int ANN_IMAGE = 101;
```
3. Sobreponha os seguintes métodos de CMBPageAnnotation.  

```
public void draw(Graphics2D g2);
public void drawOutline(Graphics2D g2);
public CMBPropertiesPanel getAnnotationPropertiesPanel();
```
4. Implemente a interface CMBAnnotationPropertiesInterface para criar o painel de propriedades. O painel de propriedades surge quando um utilizador opta por editar as propriedades da anotação personalizada.
5. Faculte os métodos de definição e obtenção específicos às propriedades da anotação personalizada.
6. Adicione o tipo de anotação personalizada.  

```
annoServices.prepareToAddAnnotation(TImageAnnotation.ANN_IMAGE,
"TImageAnnotation",1);
```

O exemplo de tipo de anotação TImageAnnotation está incluído no directório samples. O exemplo demonstra como adicionar um tipo de anotação personalizada aos serviços de anotação.

---

## Tipos e conversões de documentos suportados no toolkit do visualizador de Java

A tabela seguinte apresenta um resumo dos tipos e conversões de documentos suportados por predefinição (utilizando os motores de conversão fornecidos).

Podem ser incluídas capacidades de conversão adicionais através da criação de classes que expandam o CMBDocumentEngine e adicionando as mesmas à lista de motores nas propriedades de motores transmitidas no construtor de CMBStreamingDocServices.

### Tipos de documento e conversões no Windows

Tabela 45. Tipos de documento no Windows

| Tipo MIME       | <nome>DocumentEngine | Paginado         | Com anotações | Suporte de manipulação de páginas | Convertido para              |
|-----------------|----------------------|------------------|---------------|-----------------------------------|------------------------------|
| application/afp | CMBOD                | Sim <sup>1</sup> | Não           | Não                               | text/html                    |
| application/afp | CMBOD                | Não              | Não           | Não                               | application/afp <sup>2</sup> |

Tabela 45. Tipos de documento no Windows (continuação)

| Tipo MIME                                                                                                                                                                                                                       | <nome>DocumentEngine | Paginado | Com anotações | Suporte de manipulação de páginas | Convertido para                                                                                    |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|----------|---------------|-----------------------------------|----------------------------------------------------------------------------------------------------|
| application/lin<br>application/<br>ondemand line                                                                                                                                                                                | CMBOD                | Não      | Não           | Não                               | text/plain <sup>3</sup>                                                                            |
| image/tiff<br>image/tif<br>image/gif<br>image/jpeg<br>image/bmp<br>text/plain<br>application/<br>vnd.ibm.modcap                                                                                                                 | CMBMSTech            | Sim      | Sim           | Sim                               | image/gif<br>image/jpeg<br>image/tiff<br>application/<br>vnd.ibm.modcap<br>text/plain <sup>5</sup> |
| image/pcx<br>image/dcx                                                                                                                                                                                                          | CMBMSTech            | Sim      | Sim           | Não                               | image/gif<br>image/jpeg                                                                            |
| text/enriched<br>text/richtext<br>text/rtf<br>text/html<br>application/<br>vnd.lotus-1-2-3<br>application/<br>vnd.ms-excel<br>application/<br>wordperfect5.1<br>application/<br>msword<br>application/<br>vnd.lotus-<br>wordpro | CMBMSTechInso        | Sim      | Sim           | Não                               | image/gif<br>image/jpeg                                                                            |
| text/url                                                                                                                                                                                                                        | CMBJava              | Não      | Não           | Não                               | text/html <sup>4</sup>                                                                             |

## Tipos de documento e conversões em AIX e Solaris

Tabela 46. Tipos de documento em AIX e Solaris

| Tipo MIME                                                                                                       | <nome>DocumentEngine | Paginado | Com anotações | Suporte de manipulação de páginas | Convertido para                                                                                    |
|-----------------------------------------------------------------------------------------------------------------|----------------------|----------|---------------|-----------------------------------|----------------------------------------------------------------------------------------------------|
| application/afp                                                                                                 | CMBOD                | Não      | Não           | Não                               | application/afp <sup>2</sup>                                                                       |
| image/tiff<br>image/tif<br>image/gif<br>image/jpeg<br>image/bmp<br>text/plain<br>application/<br>vnd.ibm.modcap | CMBMSTech            | Sim      | Sim           | Sim                               | image/gif<br>image/jpeg<br>image/tiff<br>application/<br>vnd.ibm.modcap<br>text/plain <sup>5</sup> |
| image/pcx<br>image/dcx                                                                                          | CMBMSTech            | Sim      | Sim           | Não                               | image/gif<br>image/jpeg                                                                            |

Tabela 46. Tipos de documento em AIX e Solaris (continuação)

| Tipo MIME                                                                                                                                                                                                                    | <nome>DocumentEngine | Paginado | Com anotações | Suporte de manipulação de páginas | Convertido para         |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|----------|---------------|-----------------------------------|-------------------------|
| text/enriched<br>text/richtext<br>text/rtf text/html<br>application/<br>vnd.lotus-1-2-3<br>application/<br>vnd.ms-excel<br>application/<br>wordperfect5.1<br>application/<br>msword<br>application/<br>vnd.lotus-<br>wordpro | CMBMSTechInso        | Sim      | Sim           | Não                               | image/gif<br>image/jpeg |
| text/url                                                                                                                                                                                                                     | CMBJava              | Não      | Não           | Não                               | text/html <sup>4</sup>  |

## Tipos de documento e conversões em Linux

Tabela 47. Tipos de documento em Linux

| Tipo MIME                                                                                              | <nome>DocumentEngine | Paginado | Com anotações | Suporte de manipulação de páginas | Convertido para                                                                                    |
|--------------------------------------------------------------------------------------------------------|----------------------|----------|---------------|-----------------------------------|----------------------------------------------------------------------------------------------------|
| application/afp                                                                                        | CMBOD                | Não      | Não           | Não                               | application/afp <sup>2</sup>                                                                       |
| image/tiff image/tif<br>image/gif image/jpeg<br>image/bmp text/plain<br>application/<br>vnd.ibm.modcap | CMBMSTech            | Sim      | Sim           | Sim                               | image/gif<br>image/jpeg<br>image/tiff<br>application/<br>vnd.ibm.modcap<br>text/plain <sup>5</sup> |
| image/pcx image/dcx                                                                                    | CMBMSTech            | Sim      | Sim           | Não                               | image/gif<br>image/jpeg                                                                            |
| text/url                                                                                               | CMBJava              | Não      | Não           | Não                               | text/html <sup>4</sup>                                                                             |

### Nota:

- 1 O toolkit AFP2Web executa a conversão paginada para text/hml. Este toolkit não está equipado com o DB2 Information Integrator for Content. Encontra-se disponível em separado. Além disso, a ponte para o AFP2Web incluída no CMBODDocumentEngine é apenas Windows e, por este motivo, a conversão de AFP para HTML está disponível apenas em Windows utilizando este motor.
- 2 A conversão de AFP para AFP também concatena segmentos de grandes documentos de objectos.
- 3 A conversão de dados de linha para texto simples também converte EBCDIC para ASCII.
- 4 text/url é um tipo MIME especial utilizado para indicar que um documento de texto simples contém um URL. O CMBStreamingDocServices converte o URL para um documento HTML com uma ligação de reencaminhamento automático.



- 5      Só é possível gravar um documento de qualquer tipo de origem para qualquer um destes tipos de destino se o tipo de destino permitir. Não é possível gravar um documento com várias páginas para os tipos image/jpeg e image/bmp. Não é possível gravar um documento de imagem para o tipo plain/text.

---

## Trabalhar com as funções de manipulação de páginas

O toolkit de visualizador de Java faculta capacidades para editar documentos de páginas-imagens ao nível da página.

Esta funcionalidade simula as capacidades de trabalhar como documentos em papel.

A funcionalidade completa inclui:

### **Criar, guardar e exportar**

- Criar documento
- Guardar como novo documento (não exposto na barra de ferramentas predefinida)
- Guardar documento modificado com anotações
- Exportar documento, com ou sem anotações

### **Seleccção múltipla**

- Seleccção múltipla em miniaturas com o rato ou o teclado
- Seleccionar tudo ou Anular Seleccionar tudo

### **Manipulação**

- Cortar, copiar e colar páginas seleccionadas num documento ou entre documentos.
- Eliminar páginas seleccionadas
- Arrastar e largar páginas de miniaturas seleccionadas para mover ou copiar
- Rodar páginas permanentemente

## Activar uma aplicação personalizada para a manipulação de páginas

Pode alterar as aplicações personalizadas existentes que contêm o CMBGenericDocViewer para suportarem a manipulação de páginas.

Por predefinição, o CMBGenericDocViewer não activa a manipulação de páginas.

Para activar o suporte numa aplicação personalizada; efectue os seguintes passos:

1. Defina a propriedade de configuração do visualizador para a manipulação de páginas e para a selecção de várias páginas para true. No objecto CMBViewerConfiguration.properties, adicione as seguintes linhas no ficheiro:  
PageManipulation=trueMultiplePageSelection=true
2. Actualize a configuração do visualizador para adicionar novas barras de ferramentas e artigos de menus relacionados com a manipulação de páginas. O DB2 Content Manager faculta as seguintes acções:

| Acção             | Descrição                                                                                                                                                                                                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>new_doc</b>    | Cria um documento. O documento criado no visualizador inicialmente não contém páginas. Ao guardar o documento, cria-o no servidor.                                                                                                                                                                     |
| <b>save_as</b>    | Guarda o documento no visualizador como um novo documento no servidor.                                                                                                                                                                                                                                 |
| <b>export_doc</b> | Exporta o documento apresentado no visualizador para o sistema de ficheiros. Esta acção apresenta uma caixa de diálogo que permite ao utilizador especificar o nome do ficheiro, tipo e localização no directório. O quadrado de confirmação activa a inclusão das anotações nas páginas do documento. |

3. Crie um operador para implementar o ouvinte `CMBGenericDocSaveEvent`. Este operador é registado com a instância de `CMBGenericDocViewer`. Toma o controlo quando uma acção de chamada de visualizador tem de ser guardada. O operador também toma controlo quando grava o documento com o documento é fechado.

```
class SaveHandler implements CMBGenericDocSaveListener
{
 public void genericDocSave(CMBGenericDocSaveEvent evt)
 {
 CMBDocument document = evt.getDocument();
 CMBAnnotationSet annoSet = evt.getAnnotationSet();
 if (evt.getSaveAsNew()) {
 .. logic to save new documents
 }
 if (document.isModified()) {
 OutputStream docstream = ..
 docServices.setPreferredFormats(
 new String[] { document.getMimeType()});
 document.write(docstream);
 }
 if (annoSet.isDirty()) {
 OutputStream annostream = ..
 annoSet.write(annostream);
 }
 ..other save logic
 document.setModified(false);
 annoSet.setDirty(false);
 }
}
```

Normalmente, quando efectua operações de manipulação de páginas em documentos com anotações, estas anotações são alteradas. Analise o impacto da actualização de um documento e respectivas anotações para não ter de actualizar o documento e as anotações separadamente, caso a operação falhe.

O operador `CMBGenericDocSaveEvent` deve guardar o conteúdo das anotações e do documento. Antes da V8.3, os utilizadores utilizavam a marcação de repetição para guardar anotações. As marcações de repetição são utilizadas quando o sinalizador `isDirty()` para um `CMBAnnotationSet` é true e ou não existem operadores `CMBGenericDocSaveEvent` registados ou os operadores registados não redefinem o sinalizador `isDirty()` no conjunto de anotações.

Se o operador `CMBGenericDocSaveEvent` guardar as anotações em todos os casos, a lógica de guardar nas marcações de repetição é facultativa.

4. Adicione logic para activar ou desactivar capacidades de manipulação de páginas de acordo com os privilégios do utilizador. Os seguintes métodos activam ou desactivam capacidades de manipulação de páginas no CMBGenericDocViewer:

| Método                                                                   | Descrição                                                                                                                                          |
|--------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>setPageManipulationEnabled(boolean enable)</b>                        | Activa ou desactiva todas as capacidades de manipulação de páginas no visualizador.                                                                |
| <b>setPrivilege(CMBDocument document, int privilege, boolean enable)</b> | Este método pode activar e desactivar a manipulação de páginas em documentos individuais, e activar e desactivar a capacidade de criar documentos. |

5. Adicione logic para responder às operações de manipulação de páginas. Isto é normalmente necessário se houver informações adicionais mantidas pela aplicação relacionada com as páginas do documento. Quando os utilizadores adicionam ou removem páginas, a aplicação tem de sincronizar as informações específicas da página com as informações do visualizador.

A eliminação de páginas num documento não destrói essas páginas. A eliminação remove as páginas e coloca-as num objecto de páginas removidas. Com uma operação de corte, o objecto de páginas removidas é colocado na área de transferência. A eliminação coloca este objecto de páginas removidas na fila de tarefas efectuadas, para a utilização numa operação de Anulação. Para ser possível guardar informações específicas da aplicação, o CMBGenericDocStateChangedEvent faculta os métodos getRelatedInfo e setRelatedInfo, que permitem a uma aplicação associar um objecto seriável a páginas eliminadas ou copiadas (utilizando setRelatedInfo) e obtidas quando as páginas são adicionadas (utilizando getRelatedInfo). Exemplos de consequências de adicionar, modificar ou eliminar páginas: CMBGenericDocStateChangedEvent.

```

gdv.addDocStateChangedListener(new ChangeHandler());

class ChangeHandler implements CMBGenericDocStateChangedListener {
public void genericDocStateChanged
(CMBGenericDocStateChangedEvent evt) {
if (evt.getChangeType() == CHANGETYPE_PAGES_DELETED) {
.. add logic for pages deleted
} else if (evt.getChangeType() == CHANGETYPE_PAGES_COPIED) {
.. add logic for pages copied (to clipboard)
} else if (evt.getChangeType() == CHANGETYPE_PAGES_ADDED) {
.. add logic for pages added
}
}
}
}

```

---

## Definições de propriedades de motor para compressão

O DB2® Content Manager fornece opções de configuração para ficheiros TIFF (Formato de Ficheiro de Imagem de Comando) ao utilizar o motor de documentos CMBMSTechDocumentEngine.

Esta opções de configuração permitem especificar a compressão de páginas alteradas com base na quantidade de bits do documento. A compressão especificada é utilizada para comprimir novamente apenas as páginas alteradas quando o documento é guardado. As propriedades do motor base utilizadas pelos Streaming Doc Services, localizadas no ficheiro cmbviewerengine.properties, incluem as seguintes propriedades de motor:

#### Ficheiros TIFF de 1 bit

As seguintes compressões estão disponíveis com G3\_COMPRESSION como predefinição:

- G3\_COMPRESSION
- G4\_COMPRESSION
- HUFFMAN\_COMPRESSION
- PACKEDBITS
- UNCOMPRESSED

#### Ficheiros TIFF de 4 bits

As seguintes compressões estão disponíveis com PACKEDBITS como predefinição:

- PACKEDBITS
- UNCOMPRESSED

#### Ficheiros TIFF de 8 bits

As seguintes compressões estão disponíveis com PACKEDBITS como predefinição:

- PACKEDBITS
- UNCOMPRESSED

#### Ficheiros TIFF de 24 bits

JPEG\_COMPRESSION é a única opção disponível.

**Importante:** Quando utiliza uma opção de configuração para compressão, o desempenho é afectado. Terá de avaliar e seleccionar uma definição que seja ideal para o seu sistema.

---

## Especificar informações de codificação de caracteres para documentos de texto simples

Para decodificar correctamente documentos de texto simples, o toolkit do visualizador de Java requer que a codificação de caracteres seja fornecida pela aplicação.

As informações CCSID de uma parte podem ser obtidas a partir do método `getCCSID` na API de Java `DKTextICM` ou do bean de Java `CMBObject`. Esta informações podem ser convertidas para uma codificação de caracteres Java utilizando o método `CMBDocumentServices.getEncodingforCCSID`. A codificação de caracteres resultante pode ser fornecida no método `CMBStreamingDocServices.loadDocument` durante o carregamento de um documento e na repetição de marcação `getPartEncoding` para documentos com várias partes. Tenha em atenção que se a codificação não for fornecida em `loadDocument` e a repetição de marcação `getPartEncoding` não for implementada, a codificação de caracteres que vai ser utilizada é a predefinição, conforme definido pela Java para o locale actual.

Para obter mais informações sobre a implementação, consulte os seguintes artigos no manual *Application Programming Reference*:

#### Métodos `DKTextICM`

- `getCCSID`
- `setCCSID`

#### Beans de Java

- CMBOBJECT.setCCSID
- CMBOBJECT.getCCSID
- CMBDOCUMENTSERVICES.getEncodingforCCSID
- CMBDOCUMENTSERVICES.getCCSIDforEncoding
- CMBSTREAMINGDOC SERVICES.loadDocument
- CMBSTREAMINGDOC SERVICESCALLBACKS.getPartEncoding
- CMBDOCUMENTENGINE.loadDocument
- CMBDOCUMENTENGINECALLBACKS.getPartEncoding

## Modificar preferências do visualizador

CMBGenericDocViewer contém uma ação e uma caixa de diálogo que permitem modificar as preferências do visualizador.

As preferências incluem o tamanho das miniaturas, o indicador visual para documentos visualizados, as definições da janela de impressão e as propriedades predefinidas das anotações. As preferências são armazenadas no registo da estação de trabalho local. Pode exportar as preferências como um ficheiro e importá-las para outra estação de trabalho.

A janela Preferences (Preferências) tem uma árvore de páginas de preferências no lado esquerdo e as preferências para a página seleccionada no lado direito. Quando abrir a janela Preferences (Preferências) pela primeira vez, são apresentadas as preferências gerais do visualizador, consulte Figura 41:

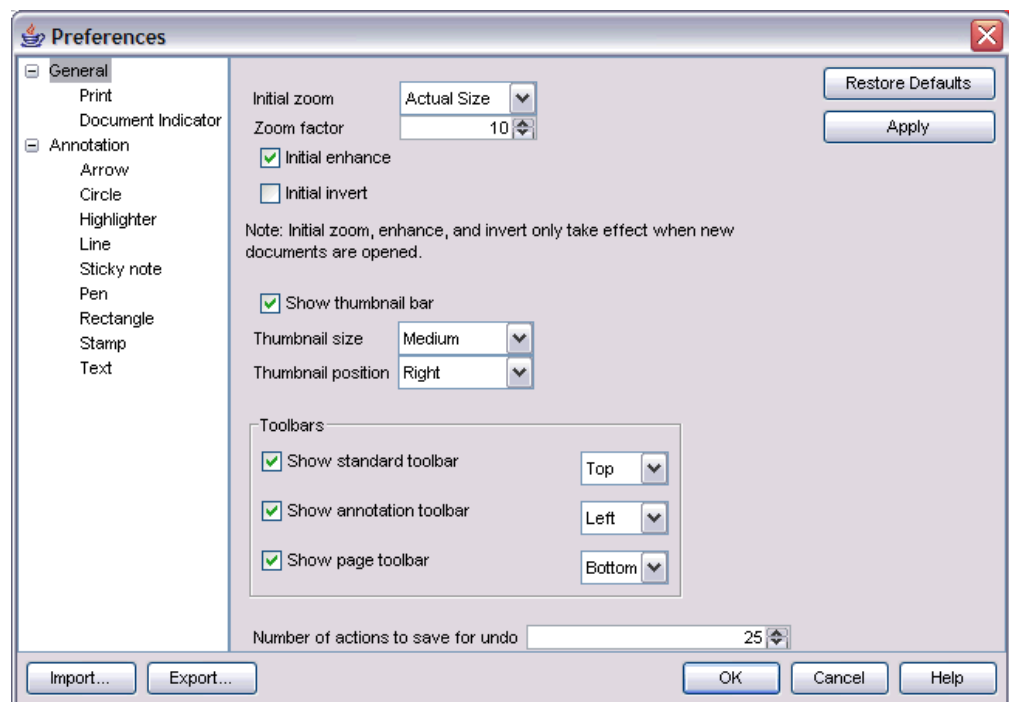


Figura 41. A janela das preferências

Em seguida, é apresentada uma descrição da lista de botões existentes na janela Preferences (Preferências):

### Restore Defaults (Restaurar Predefinições)

Define as preferências na página para os valores predefinidos.

**Apply (Aplicar)**

Aplica as alterações às preferências na página.

**Import (Importar)**

Importa as preferências de um ficheiro.

**Export (Exportar)**

Exporta as preferências para um ficheiro.

**OK**

Aplica as alterações a todas as preferências e fecha a caixa de diálogo.

**Cancel (Cancelar)**

Fecha a caixa de diálogo e restaura todas as preferências que não tiverem sido aplicadas para os valores com que estavam definidas quando a caixa de diálogo foi aberta.

Pode alterar a configuração de páginas e preferências em cada página, usando as propriedades de configuração do visualizador. As propriedades predefinidas incluem:

- `preference.elements=general,annotation`
- `preference.general=ps.document,ps.thumbnails,ps.toolbars,ps.general.common`
- `preference.general.elements=ps.printing,ps.appearance.doc.indicator`
- `preference.annotation=ps.annotation.common`
- `preference.annotation.elements=ps.arrow,ps.circle,ps.highlight,ps.line,ps.note,ps.pen,ps.rectangle,ps.stamp,ps.text`

Embora o utilizador só veja um conjunto de preferências, na realidade existem dois conjuntos armazenados: Preferências do Sistema e Preferências do Utilizador. As Preferências do Sistema derivam inicialmente dos valores nas propriedades de configuração do visualizador. É possível modificá-las com o método `importSystemPreferences`, descrito na secção seguinte. As Preferências do Sistema são os valores usados quando o utilizador faz clique em **Restore Defaults (Restaurar Predefinições)**. As Preferências do Utilizador são criadas quando o utilizador altera os valores predefinidos do sistema.

No Windows, as preferências do sistema são guardadas em `HKEY_LOCAL_MACHINE/JavaSoft/Prefs/com/ibm/mm/viewer` e as preferências do utilizador são guardadas no registo em `HKEY_CURRENT_USER/JavaSoft/Prefs/com/ibm/mm/viewer`.

Para facilitar a importação e exportação, as preferências são escritas como um ficheiro XML. O esquema utilizado para este XML encontra-se descrito na documentação relativa às preferências de Java 1.5. A chave usada é `com/ibm/mm/viewer`.

O suporte para as preferências é fornecido pela classe `CMBGenericDocViewer` utilizando os seguintes métodos:

- `importSystemPreferences(InputStream is)`
- `importUserPreferences(InputStream is)`
- `haveUserPreferencesChanged()`
- `exportSystemPreferences()`
- `exportUserPreferences()`

Para obter mais informações sobre a implementação e desenvolvimento de uma aplicação, consulte as informações sobre a classe CMBGenericDocViewer no manual *Application Programming Reference* no Information Center.

Por predefinição, a ferramenta **Preferences (Preferências)** encontra-se na barra de ferramentas **Operation (Operação)**, também designada por barra de ferramentas padrão. Se esta barra de ferramentas for removida ao não seleccionar o quadrado de confirmação **Show standard toolbar (Mostrar barra de ferramentas padrão)** na janela Preferences (Preferências), utilize a tecla de atalho da ferramenta **Preferences (Preferências)**, ctrl+shift+P para apresentar a janela Preferences (Preferências).

## Preferências de impressão

Ao fazer clique em **Print (Imprimir)** na vista em árvore, são apresentadas as preferências gerais da impressão.

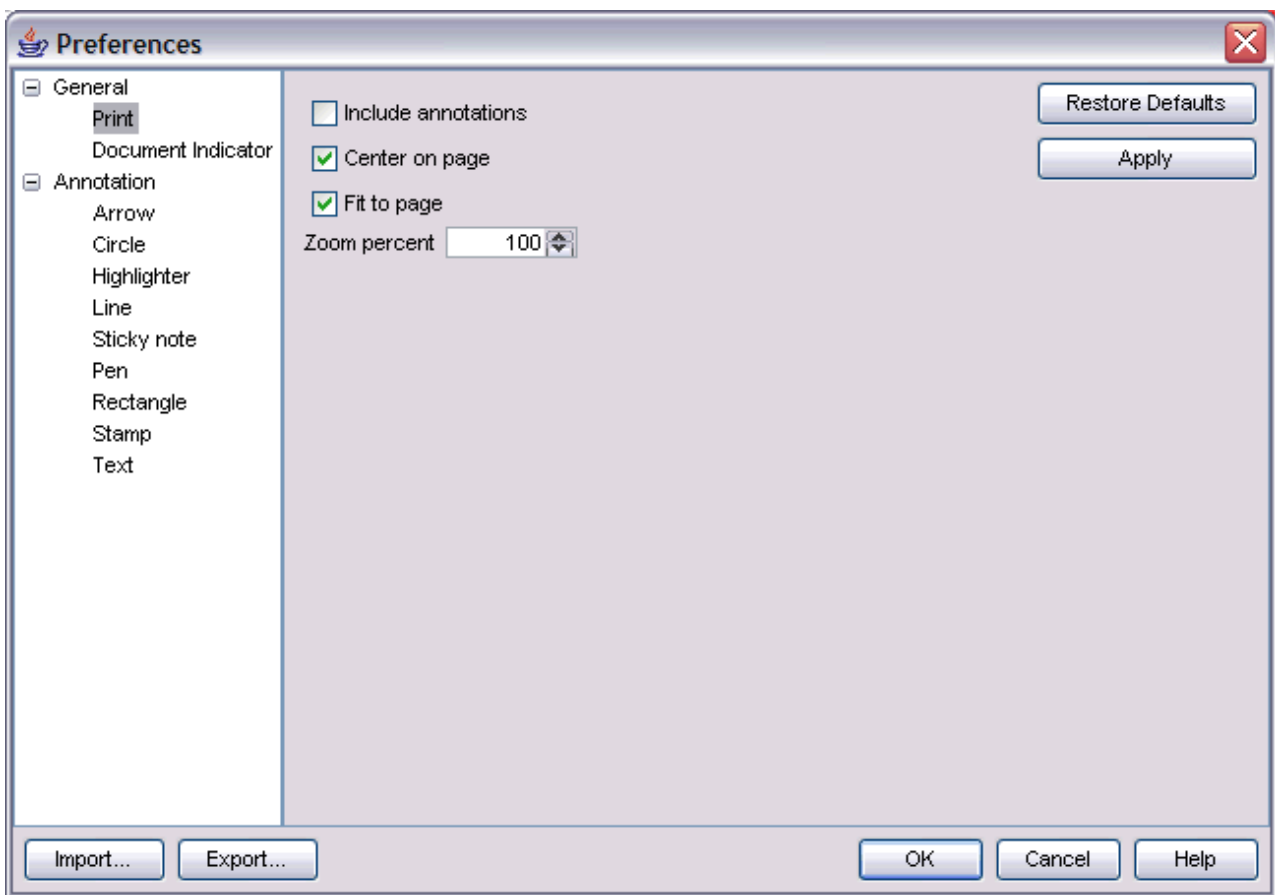


Figura 42. Preferências de impressão no visualizador de Java

É possível especificar as definições na caixa de diálogo Imprimir utilizando as novas preferências de impressão. Estas preferências incluem imprimir um documento com ou sem anotações e dimensionar e centrar a página do documento. Utilize a janela Preferences (Preferências) para efectuar alterações às preferências de impressão.

O conjunto de preferências, ps.printing, está incluído no ficheiro de propriedades de configuração do visualizador predefinido, cmbviewerconfiguration.properties.

Este conjunto de preferências contém as preferências de impressão utilizadas como definições iniciais para as caixas de diálogo de impressão. O conjunto inclui as seguintes preferências:

**Printing.annotations.include**

Controla o quadrado de confirmação para incluir anotações na caixa de diálogo de impressão.

**Printing.centerOnPage**

Controla o quadrado de confirmação Center on page (Centrar na página) na caixa de diálogo Print (Imprimir).

**Printing.fitToPage**

Controla o quadrado de confirmação Fit to page (Ajustar à página) na caixa de diálogo Print (Imprimir).

**Printing.zoom**

Controla a percentagem de zoom na caixa de diálogo Imprimir. Esta percentagem corresponde a um número inteiro de 25 a 500 que especifica a percentagem de zoom do documento quando impresso, sendo 100 a predefinição.

As classes, `CMBGenericDocPrintStatusEvent`, `CMBGenericDocPrintStatusEvent` e `CMBGenericDocPrintStatusListener` reúnem comentários sobre o progresso da impressão. Consulte estas classes no manual *Application Programming Reference* no Information Center.

## Imprimir

Também poderá imprimir vários documentos num único passo. O botão de acção do visualizador Print All Documents (Imprimir todos os documentos) imprime todos os documentos que se encontrem abertos.

É possível especificar as definições nas janelas Print (Imprimir) utilizando as novas preferências de impressão. Estas preferências incluem imprimir um documento com ou sem anotações e dimensionar e centrar a página do documento. Utilize a janela Preferences (Preferências) para efectuar alterações às preferências de impressão. Poderá ajustar cada página do documento numa página impressa ou especificar a percentagem de zoom. A opção Fit to page (Ajustar à página) tem prioridade. Se a opção Fit to page (Ajustar à página) estiver seleccionada, as páginas do documento são ajustadas à página quando é impressa. Caso contrário, é aplicado o valor da percentagem de zoom.

Além disso, uma vez que serão impressos muitos documentos, a impressão de vários documentos ordena a obtenção de documentos de modo a gerir a utilização da memória.

Foram adicionadas novas classes para recolher comentários sobre o progresso da impressão. Consulte as novas classes `CMBGenericDocPrintStatusEvent` e `CMBGenericDocPrintStatusListener` no manual *Application Programming Reference* no Information Center.

## Preferências do indicador de documentos

Ao fazer clique em Document Indicator (Indicador de documentos) na vista em árvore, proporciona o seguinte:



- Capacidade para alterar o estilo e a cor do tipo de letra do nome do documento nos separadores de selecção de documentos no visualizador para indicar os documentos que já foram visualizados
- Um mecanismo temporizador para atrasar a alteração do tipo de letra ou cor, uma vez que um documento poderá ser visualizado apenas durante alguns instantes durante a navegação de um documento para outro
- Controlo sobre o estilo e a cor do tipo de letra e sobre o tempo de atraso nas preferências do visualizador

Figura 43. Indicador de documentos Janela do indicador de documentos

O conjunto de preferências `ps.appearance.doc.indicator` inclui as seguintes preferências:

**Appearance.tab.indicate**

Activa ou desactiva a definição do estilo e cor do tipo de letra para documentos que tenham sido visualizados. A predefinição é `true`.

**Appearance.tab.fontstyle**

Especifica o estilo do tipo de letra para os separadores do documento para documentos que tenham sido visualizados. Esta preferência é utilizada se `Appearance.tab.indicate=true`. O valor predefinido é `bold`. Os valores válidos são `bold`, `italic`, `bold_italic` e `plain`.

**Appearance.tab.color**

Especifica a cor do texto para os separadores do documento para documentos que tenham sido visualizados. Esta preferência é utilizada se `Appearance.tab.indicate=true`. O valor predefinido é `green` (`#00ff00`). Os valores válidos são cadeias hexadecimais que representam o valor `rgb` como `#rrggbb`.

**Appearance.tab.delay**

Especifica o número de segundos durante o qual o documento tem de ser visualizado antes de o separador ser indicado. Esta preferência é utilizada se `Appearance.tab.indicate=true`. O valor predefinido é `10`. O valor válido será um número inteiro de `0` a `1024`.

Foram adicionados dois novos métodos, `hasDocumentBeenViewed` e `setDocumentViewed` a `CMBGenericDocViewer`. Para obter mais informações, consulte os novos métodos em `CMBGenericDocViewer` no manual *Application Programming Reference* no Information Center.

## Preferências para anotações

Ao fazer clique em **Annotation (Anotação)** na vista em árvore, são apresentadas as preferências para anotações:

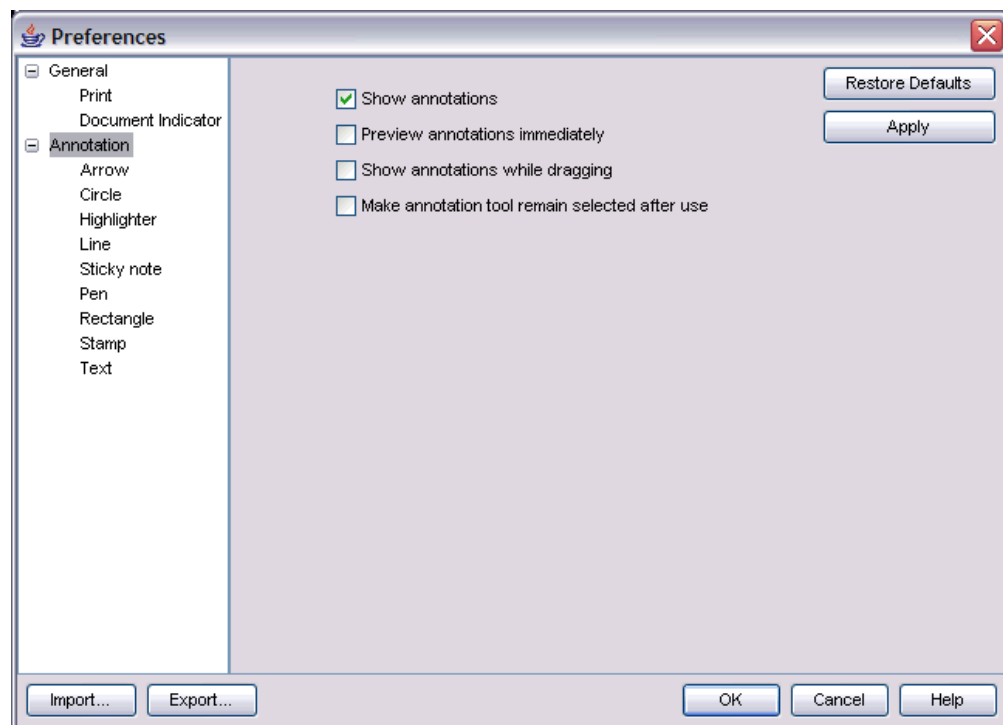


Figura 44. Lista das preferências gerais para anotações

Ao fazer clique num determinado tipo de anotação, aparecem as preferências para esse tipo de anotação. Por exemplo, consulte a janela Sticky Note (Nota Persistente) na Figura 45:

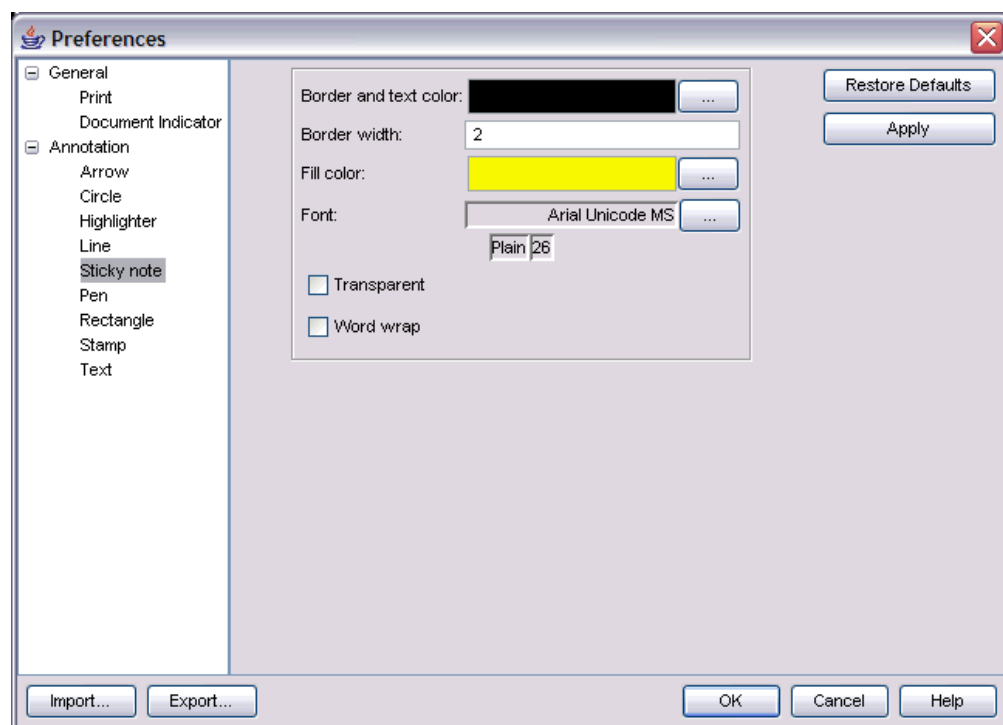


Figura 45. Preferências para a anotação da nota persistente

## Ferramentas de esquema do visualizador de Java

Para permitir a apresentação de várias vistas de um ou mais documentos em simultâneo na applet do visualizador de Java, o toolkit do visualizador de Java faculta as seguintes ferramentas de esquema:

### **Tabbed (Com separadores)**

Apresenta um documento no visualizador e disponibiliza separadores para seleccionar outros documentos. Esta é a vista predefinida. Tecla de atalho: Ctrl – 6.

### **Split Horizontal (Dividir na horizontal)**

Divide o visualizador numa área esquerda e numa área direita. É possível visualizar documentos diferentes dentro de cada área utilizando as listas pendentes. Os esquemas lado a lado apresentam duas vistas para o mesmo conjunto de documentos no visualizador. As anotações ou as manipulações de página efectuadas numa área da janela também serão apresentadas no mesmo documento na outra área da janela. Tecla de atalho: Ctrl – 7.

### **Split Vertical (Dividir na vertical)**

Divide o visualizador numa área de visualização superior e inferior. Tecla de atalho: Ctrl – 8.

### **Thumbnails Only (Apenas miniaturas)**

Apresenta as miniaturas para um documento. Pode seleccionar documentos numa área da janela com separadores e aceder às páginas fazendo duplo clique nos mesmos. Tecla de atalho: Ctrl – 9.

O toolkit do visualizador de Java também permite a visualização lado a lado. Para activar a visualização lado a lado, consulte os métodos que foram adicionados a `CMBGenericDocViewer`, `CMBAnnotationServices`, `CMBAnnotationView`, `CMBPageAnnotation`, `CMBAnnotationSet`, e `CMBAnnotationSelectedEvent` no manual *Application Programming Reference* no Information Center.

Para definir as barras de ferramentas e os menus, as ferramentas de esquema incluídas nas propriedades de configuração do visualizador são as seguintes:

#### **layout\_tabbed**

Acção para mudar para um esquema com separadores.

#### **layout\_split\_horizontal**

Acção para mudar para um esquema horizontal lado a lado.

#### **layout\_split\_vertical**

Acção para mudar para um esquema vertical lado a lado.

#### **layout\_thumbnails\_only**

Acção para mudar para um esquema apenas de miniaturas.

---

## Limitações do visualizador de documentos Java

Existem algumas limitações no visualizador de documentos Java.

De seguida, encontra-se a lista das limitações conhecidas do visualizador de documentos de Java:

### **A tecla de atalho do Intel Graphic Controller entram em conflito com as teclas de atalho do visualizador**

Quando um cliente usa um toolkit do visualizador de Java, como o visualizador do applet `eClient`, e a aplicação do Intel Graphic Controller

está activa, existe um conflito com o atalho Ctrl+Alt+I. Este atalho é usado no visualizador para inverter um documento.

O Graphic Controller possui teclas de comutação activadas por predefinição. Se a função das teclas de comutação estiver desactivada no Intel Graphic Controller, as teclas do visualizador funcionam correctamente.

Para modificar o visualizador de applet eClient para utilizar outras teclas, proceda do seguinte modo:

1. Adicione a entrada `Circle.key=control alt I` ao ficheiro `eClient cmbViewerConfigurations.properties`.
2. Defina a chave para outra opção, como `Circle.key=shift 0`.

#### **O leitor de ecrã JAWS Versão 8.0 interfere com as teclas de atalho do visualizador Java que começam pela tecla Alt**

O leitor de ecrã JAWS interfere com as teclas de atalho do visualizador Java que começam pela tecla Alt. No visualizador Java, a combinação Alt+Enter abre as propriedades de anotação, Alt+F4 fecha qualquer janela, Alt+Seta para a Esquerda ou Seta para a Direita move as anotações. Ao tentar mover as anotação com Alt+Seta para Cima ou Seta para Baixo pode fazer com que as anotações saltem ou não se movam.

Para mover as anotações, feche o leitor de ecrã JAWS e mova as anotações com as teclas do teclado. Se tiver o JAWS aberto, mova as anotações com o rato.

#### **A janela Imprimir do visualizador Java e do visualizador do applet eClient não tem foco por predefinição**

Por predefinição, quando a janela Imprimir é aberta no visualizador não tem foco. Por conseguinte, não é possível navegar na janela com o teclado (Tab para navegar e Alt + carácter mnemónico para seleccionar um determinado campo). Para resolver este problema, clique na janela para ter foco. Prima Alt+Tab para passar para outra janela.



---

## Trabalhar com a biblioteca de identificadores JSP e com a servlet de controlo

O DB2 Information Integrator for Content inclui uma biblioteca de identificadores de Java Server Pages (JSP) e uma servlet que pode utilizar ao escrever JSPs ou servlets para aplicações da Web. Utilizar a biblioteca de identificadores reduz a necessidade de scriplets Java em JSP escritas em JavaBeans do DB2 Information Integrator for Content.

Esta biblioteca de identificadores funciona como servlet que pode funcionar como o controlador de uma aplicação da Web de design controladora de vista de modelo e executar a inicialização de beans e outras acções.

---

### Configurar a biblioteca de identificadores e a servlet

Deve instalar a biblioteca de identificadores e a servlet num servidor da Web com o IBM WebSphere Application Server e configurar o servidor da Web para que este as utilize.

Para obter informações relativas à instalação da biblioteca de identificadores e da servlet, da sua configuração e para obter informações relativas à construção de ficheiros WAR/EAR, consulte Planning and Installing DB2 Information Integrator for Content.

---

### Utilizar a biblioteca de identificadores

Este exemplo mostra um exemplo de JSP que utiliza o identificador de modelos de pesquisa para obter uma lista de modelos de pesquisa.

#### Exemplo

```
<%@ taglib uri="cmb" prefix="cmb" %>
<%@ page import="com.ibm.mm.beans.*" %>
<jsp:useBean id="connection" scope="session"
 class="com.ibm.mm.beans.CMBConnection" />
<%
 CMBSchemaManagement schema = connection.getSchemaManagement();
 CMBSearchTemplate[] searchtemplates = schema.getSearchTemplate();
 request.setAttribute("searchtemplates", searchtemplates);
%>

<html>
 <head>
 <title>Search Templates Tag Test</title>
 </head>

 <body bgcolor="white">
 <table border=2 cellpadding=3 cellspacing=3>
 <tr>
 <td>Available Search Templates</td>
 </tr>
 <tr>
 <td><%= searchtemplate.getName() %></td>
 </tr>
 </table>
 </body>
</html>
```

```

 </cmb:searchtemplates>
 </table>
</body>
</html>

```

A directiva `taglib` declara que a página utiliza a biblioteca de identificadores de DB2 Information Integrator for Content e lhes associa o prefixo `cmb`. Em seguida, é invocado o identificador `<searchtemplates>` e o método `getName()` devolve o nome de cada modelo de pesquisa.

## Convenções utilizadas na biblioteca do identificador

Os identificadores JSP têm atributos para especificar os beans que utilizam ou geram.

Quando esses atributos não são especificados, são utilizados os valores predefinidos. Estes parâmetros são opcionais. Os seus valores são retirados a partir de variáveis locais e os atributos no intervalo do pedido e da sessão. Quando utilizados junto com o toolkit do servlet, as variáveis locais, os atributos de pedido ou os atributos de sessão contêm as predefinições apropriadas. A Tabela 48 mostra os bens predefinidos e o âmbito no qual são assumidos ou colocados. Estas convenções também são seguidas pelo servlet; siga estas convenções noutros servlets que compuser para trabalhar com a biblioteca de identificadores.

*Tabela 48. Convenções da biblioteca de identificadores*

| Âmbito    | Nome           | Tipo                | Descrição                                                                               |
|-----------|----------------|---------------------|-----------------------------------------------------------------------------------------|
| aplicação | connectionPool | CMBConnectionPool   | O bean do conjunto de ligações que é partilhado em várias sessões                       |
| sessão    | ligações       | CMBConnection       | A instância da CMBConnection para a sessão                                              |
| sessão    | esquema        | CMBSchemaManagement | Bean da gestão de esquemas                                                              |
| sessão    | dados          | CMBDataManagement   | Bean da gestão de esquemas                                                              |
| sessão    | utilizador     | CMBUserManagement   | Bean da gestão de utilizadores                                                          |
| sessão    | consulta       | CMBQuesryService    | Bean do serviço de consultas                                                            |
| sessão    | traceLog       | CMBTraceLog         | Bean de registo de rastreio; todos os outros beans enviam o seu rastreio para este bean |
| sessão    | docservices    | CMBDocumentServices | Bean de serviços de documentos                                                          |
| pedido    | artigo         | CMBItem             | O último artigo no qual o utilizador trabalhou                                          |
| pedido    | artigos        | CMBItem[ ]          | A recolha dos artigos nos quais o utilizador trabalhou pela última vez                  |
| pedido    | searchTemplate | CMBSearchTemplate   | O modelo de pesquisa seleccionado                                                       |
| pedido    | searchResults  | CMBSearchResults    | Os resultados da última pesquisa                                                        |

---

## Resumo dos identificadores

Esta secção apresenta um resumo da biblioteca de identificadores.

### Identificadores relacionados com ligações

Esta secção apresenta uma lista de todos os identificadores relacionados com ligações.

**<cmb:datasources connection="connection">datasource ... </cmb:datasources>**

Este identificador itera através das origens de dados disponíveis.

#### **ligações**

Especificar o nome de uma variável do tipo CMBCConnection que contém a ligação.

#### **origem de dados**

Uma variável de cadeia que contém o nome da origem de dados como uma cadeia.

### Identificadores relacionados com esquemas

Esta secção apresenta uma lista de todos os identificadores relacionados com esquemas.

**<cmb:searchtemplates searchTemplates="searchTemplate"> ...**

**</cmb:searchtemplates>**

Este identificador itera através dos modelos de pesquisa disponíveis.

#### **searchTemplates**

Especificar o nome de uma matriz de tipo CMBSearchTemplate[] para conter os modelos de pesquisa.

#### **searchTemplate**

Uma variável de tipo CMBSearchTemplate para conter o modelo de pesquisa.

**<cmb:searchcriteria searchTemplate="searchTemplate">criteria ...**

**</cmb:searchcriteria>**

Este identificador itera através do critério de pesquisa de um modelo de pesquisa.

#### **searchTemplate**

Especificar o nome do modelo de pesquisa.

#### **criterion**

Uma variável de tipo CMBSTCriterion para conter o critério de pesquisa.

**<cmb:displaycriteria searchTemplate="searchTemplate">criteria ...**

**</cmb:displaycriteria>**

Este identificador itera através do critério de pesquisa que pode ser apresentado para um modelo de pesquisa.

#### **searchTemplate**

Especificar o nome do modelo de pesquisa.

#### **criterion**

Uma variável de tipo CMBSTCriterion para conter o critério de pesquisa.



**<cmb:allowedoperators criterion="criterion">operator ...  
</cmb:allowedoperators>**

Este identificador itera através dos operadores permitidos para um critério de pesquisa.

**criterion**

Especificar o nome de uma variável de tipo CMBSTCriterion para conter o critério de pesquisa.

**operator**

Uma cadeia para conter o valor do operador.

**<cmb:predefinedvalues criterion="criterion"> value... </cmb:predefinedvalues>**

Este identificador itera através dos valores predeterminados de um critério de pesquisa.

**criterion**

Especificar o nome de uma variável de tipo CMBSTCriterion para conter o critério de pesquisa.

**value** Uma cadeia para conter o valor predeterminado do critério de pesquisa.

**<cmb:entities schema="schema">entity ... </cmb:entities>**

Este identificador itera através das entidades federadas disponíveis.

**schema**

Especificar o nome de uma variável de tipo CMBSchemaManagement que contém o esquema.

**entity** Uma cadeia para conter o nome da entidade.

**<cmb:attributes entity="entity" schema="schema">attribute ... </cmb:attributes>**

Este identificador itera através dos atributos federados de uma entidade federada.

**schema**

Especificar o nome de uma variável de tipo CMBSchemaManagement para conter o esquema.

**entity** Especificar o nome da entidade.

**attribute**

Uma cadeia para incluir o nome de uma variável de tipo CMBAttribute que contém o atributo.

## Identificadores relacionados com pesquisa

Esta secção apresenta uma lista dos identificadores relacionados com pesquisa.

**<cmb:searchresults searchresults="searchResults">item ... </cmb:searchresults>**

Este identificador itera através dos resultados da pesquisa.

**searchResults**

Especificar o nome de uma variável de tipo CMBSearchResults que contém os resultados da pesquisa.

**item** Uma cadeia para incluir o nome de uma variável de tipo CMBItem para conter o artigo resultante da pesquisa.

## Identificadores relacionados com artigos

Esta secção apresenta uma lista dos identificadores relacionados com artigos.

**<cmb:itemattributes item="item"> attrname ... attrtype... attrvalue...  
</cmb:itemattributes>**

Este identificador itera através dos atributos de um artigo.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**attrname**  
Uma variável de cadeia para conter o nome do atributo.

**attrtype**  
Uma variável de cadeia para conter o tipo do atributo.

**attrvalue**  
Uma variável de cadeia para conter o valor do atributo.

**<cmb:itemcontents " data="data" item="item"> content... </cmb:itemcontents>**  
Este identificador itera através do conteúdo de um artigo.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**content**  
Uma variável de tipo CMBObject para os conteúdos do artigo.

**<cmb:itemnotelogs " data="data" item="item">notelog ... </cmb:itemnotelogs>**  
Este identificador itera através dos registos de nota de um artigo.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**notelog**  
Uma variável de tipo CMBObject para conter o registo de nota do artigo.

**<cmb:itemprivileges data="data" item="item">privilege ... </cmb:itemprivileges>**  
Este identificador itera através dos privilégios de um artigo.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**privilege**  
Uma variável de tipo CMBPrivilege para conter o privilégio do artigo.

**<cmb:itemresources data="datamanagement" item="item"> resource...  
</cmb:itemresources>**  
Este identificador itera através dos recursos de um artigo.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**resource**  
Uma variável de tipo CMBResources para conter o recurso do artigo.

**<cmb:unmappeditem data="data" item="item">unmappeditem...</cmb:unmappeditem>**

Este identificador devolve um artigo sem correlação de um dado artigo com correlação.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo correlacionado.

**unmappeditem**

Uma variável de tipo CMBItem para conter o artigo não correlacionado.

**<cmb:viewdata data="data " item="item">viewdata... </cmb:viewdata>**

Este identificador devolve uma vista de um artigo.

**data** Especificar o nome de uma variável de tipo CMBDataManagement.

**item** Especificar o nome de uma variável de tipo CMBItem que contém o artigo.

**viewdata**

Uma variável de tipo CMBViewData para conter os dados visualizáveis.

## Identificadores relacionados com pastas

Esta secção apresenta uma lista dos identificadores relacionados com pastas.

**<cmb:folderitems folder="folder"> item... </cmb:folderitems>**

Este identificador itera através do conteúdo de uma pasta.

**folder** Especificar o nome de uma variável de tipo CMBItem para incluir os conteúdos da pasta.

**item** Uma variável de tipo CMBItem que representa a pasta.

## Identificadores relacionados com documentos

Esta secção apresenta uma lista dos identificadores relacionados com documentos.

**<cmb:viewerdocuments docservices="docservices">document ... </cmb:viewerdocuments>**

Este identificador itera através de documentos que estão actualmente a ser carregados.

*docservices*

Especificar o nome de uma variável de tipo CMBDocumentServices.

**document**

Uma variável de tipo CMBDocument para conter o documento.

**<cmb:documentpages document="document"> docpage... </cmb:documentpages>**

Este identificador itera através das páginas de um documento.

*document*

Especificar o nome de uma variável de tipo CMBDocument para conter o documento.

**docPage**

Uma variável de tipo CMBPage para conter a página.

---

## Servlet de controlo do DB2 Information Integrator for Content

O DB2 Information Integrator for Content faculta uma servlet com acções passíveis de ligação que podem ser utilizadas ao construir aplicações da Web.

Esta servlet funciona como um controlador de uma aplicação da Web concebida através de um controlador de visualização de modelos, executando acções e iniciando os beans (o modelo) que são depois acedidos em JSP (as visualizações), directa ou indirectamente, através da utilização de identificadores de JSP.

As acções são fornecidas para tarefas típicas de aplicações:

- Iniciar e terminar uma sessão.
- Pesquisar.
- Criar, obter, modificar e eliminar documentos.
- Criar pastas e adicionar ou remover documentos de pastas.
- Iniciar documentos e páginas de documentos para visualização.

Além disso, a servlet executa tarefas comuns antes e depois da acção, como por exemplo a gestão da ligação ao servidor de conteúdos. Após cada acção, é invocado um JSP para formatar os resultados e enviá-los de volta para o navegador.

Pode personalizar a servlet para adicionar novas acções e associar JSPs às acções.

---

## O que a servlet pode fazer

Esta secção destaca alguns dos aspectos da servlet que pode utilizar.

Estes são alguns aspectos da servlet que pode utilizar:

### Agrupamento de ligações

A servlet de controlo utiliza o agrupamento de ligações de DB2 Information Integrator for Content para facultar uma gestão de ligações de elevado rendimento. O período de atribuição de uma ligação a uma sessão pode ser relativo ao pedido ou ao tempo durante o qual a sessão está iniciada. Actualmente, o agrupamento de ligações encontra-se no âmbito da aplicação.

### Início de sessões de tempo de espera excedido

Se uma sessão exceder o tempo de espera e um pedido chegar à servlet, é apresentada a JSP de início de sessão, permitindo ao utilizador iniciar novamente uma sessão. O pedido original é executado após um início de sessão bem sucedido.

### Limpar no fim da sessão

A servlet limpa totalmente a sessão quando esta é terminada, seja através da conclusão da sessão ou através de um tempo de espera esgotado. Isto significa que a sessão é destruída ou devolvida ao conjunto. Todos os outros beans de DB2 Information Integrator for Content criados pela servlet são terminados e os seus recursos são libertados sem aguardar a execução de um ciclo de recolha de dados.

**Locale** A servlet garante a definição correcta do locale nos beans subjacentes, o que significa que as cadeias de mensagens e caracteres dependem do locale.

### Utilizar diferentes conjuntos de JSP

Um ficheiro de propriedades, denominado `cmbServlet.jsp.properties`, por predefinição, descreve as JSPs a utilizar para respostas a acções da servlet. A localização do ficheiro de propriedades é um parâmetro da aplicação. Consequentemente, é possível compor várias aplicações Web diferentes com conjuntos diferentes de JSPs.

### Expandir a servlet

Todas as acções conhecidas da servlet estão definidas num ficheiro de propriedades chamado `cmbServlet.properties` (predefinição). Pode adicionar, modificar ou eliminar acções da servlet alterando este ficheiro. Para adicionar uma nova acção, siga estes passos:

1. Implemente uma classe para executar a acção. A classe deve expandir `com.ibm.mm.servlets.CMBServletAction`.

2. Adicione o nome da classe e o nome da acção ao ficheiro `cmbServlet.properties`. Este possui a seguinte sintaxe:

```
actions = list of actions
action.<action_name>.class = class_name
```

`actions` apresenta uma lista das acções compreendidas pela servlet. Para cada acção, uma linha dentro do ficheiro de propriedades define a classe da acção. Por exemplo, para adicionar uma acção denominada reprodução, numa classe denominada `ReplayAction`:

```
actions =... replay
action.replay.class = ReplayAction
```

**Importante:** A função que permite adicionar acções personalizadas para preceder ou suceder qualquer acção predeterminada não é suportada.

A convenção de nomenclatura utilizada para todas as acções predeterminadas é `com.ibm.mm.servlets.CMBactionAction`, em que *action* é o nome da acção, sendo a primeira letra maiúscula.

## Referência de servlet

Um conjunto de parâmetros de aplicação, parâmetros de pedido e um ficheiro de propriedades servem para utilizar a servlet de controlo nas aplicações do utilizador.

### Convenções

A servlet define a sessão e valores de pedido seguintes, que podem ser utilizados noutras JSPs ou servlets. Estas convenções são seguidas pela biblioteca e identificadores de JSP. Estas convenções são iguais às da biblioteca de identificadores de DB2 Information Integrator for Content.

### Parâmetros de aplicação

O servlet compreende os seguintes parâmetros de aplicação (uma alternativa será colocar estes parâmetros no ficheiro `cmbServlet.properties`).

| Parâmetro da aplicação            | Valores | Descrição                                                    |
|-----------------------------------|---------|--------------------------------------------------------------|
| <code>servletPropertiesURL</code> | URL     | A localização do ficheiro <code>cmbServlet.properties</code> |

| Parâmetro da aplicação   | Valores                     | Descrição                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>defaultServerType</b> | Assoc, ICM, OD, V4, IP, ... | Informações de início de sessão predefinido. Estas informações, juntamente com <b>defaultServer</b> , <b>defaultUserid</b> e <b>defaultPassword</b> podem ser utilizadas em situações de ID de utilizador partilhado. Em vez de pedir informações com uma página de início de sessão, as informações de início de sessão predefinido serão utilizadas para executar o início de sessão.     |
| <b>defaultServer</b>     |                             | Informações de início de sessão predefinido.                                                                                                                                                                                                                                                                                                                                                |
| <b>defaultUserid</b>     |                             | Informações de início de sessão predefinido.                                                                                                                                                                                                                                                                                                                                                |
| <b>defaultPassword</b>   |                             | Informações de início de sessão predefinido.                                                                                                                                                                                                                                                                                                                                                |
| <b>connectionpool</b>    | Booleano: TRUE   FALSE      | Para activar o agrupamento de ligações                                                                                                                                                                                                                                                                                                                                                      |
| <b>maxfreeconnection</b> | inteiro                     | Número máximo de ligações disponíveis num conjunto de ligações.                                                                                                                                                                                                                                                                                                                             |
| <b>minfreeconnection</b> | inteiro                     | Número mínimo de ligações disponíveis num conjunto de ligações                                                                                                                                                                                                                                                                                                                              |
| <b>timeout</b>           | inteiro                     | A duração de tempo (em milisegundos) depois do qual uma ligação livre irá ser desligada e destruída.                                                                                                                                                                                                                                                                                        |
| <b>noSessionPage</b>     | URL                         | Esta é uma página que serve para ser apresentada no início de sessão, caso o servlet seja invocado sem uma sessão ou ligação estabelecida. Pode ser utilizada para pedir o início de sessão e efectuar o encadeamento de volta à acção original, permitindo ligações marcadas ao DB2 Information Integrator for Content para trabalhar, mesmo que o utilizador tenha de iniciar uma sessão. |
| <b>timedOutPage</b>      | URL                         | Esta página serve para apresentação, caso a sessão tenha esgotado o tempo de espera devido a inactividade.                                                                                                                                                                                                                                                                                  |
| <b>serverErrorPage</b>   | URL                         | Esta página serve para apresentação caso tenha ocorrido um erro ao aceder a um servidor.                                                                                                                                                                                                                                                                                                    |
| <b>connectFailedPage</b> | URL                         | Esta página serve para apresentação caso tenha ocorrido um erro ao estabelecer ligação a um servidor. Pode ser apresentado um pedido de para introduzir o id/palavra-passe de utilizador correctos para o servidor e pode ser efectuada uma nova tentativa.                                                                                                                                 |

| Parâmetro da aplicação       | Valores                 | Descrição                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------------------------|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>tracelevel</b>            | 0, 1 ou 2               | Para indicar o nível de rastreio: <ul style="list-style-type: none"> <li>• 0 - sem registo</li> <li>• 1 - excepções de registo (predefinição)</li> <li>• 2 - excepções de registo, mensagens de alerta, cabeçalhos e atributos do WebSphere Application Server, ficheiros ini do DB2 Information Integrator for Content, propriedades de sistema JVM, informações de rastreio interno de DB2 Information Integrator for Content</li> </ul> |
| <b>connectiontype</b>        | 0, 1 ou 2               | A localização da base de dados de DB2 Information Integrator for Content e dos tempos de execução do servidor de conteúdos: <ul style="list-style-type: none"> <li>• 0 - local (predefinição)</li> <li>• 1 - remoto</li> <li>• 2 - dinâmico</li> </ul>                                                                                                                                                                                     |
| <b>cmbclient</b>             | URL                     | Localização do cmbclient.ini                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>cmbcs</b>                 | URL                     | Localização do cmbcs.ini                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>serviceconnectiontype</b> | 0, 1 ou 2               | Localização dos tempos de execução de serviço <ul style="list-style-type: none"> <li>• 0 - local (predefinição)</li> <li>• 1 - remoto</li> <li>• 2 - dinâmico</li> </ul>                                                                                                                                                                                                                                                                   |
| <b>cmbsvclient</b>           | URL                     | Localização do cmbsvclient.ini                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>cmbsvcs</b>               | URL                     | Localização do cmbsvcs.ini                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>cmbcc2mime</b>            | URL                     | Localização do cmbcc2mime.ini                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>cachedir</b>              | nome de um directório   | Directório para colocação de documentos em memória cache durante a conversão de documentos                                                                                                                                                                                                                                                                                                                                                 |
| <b>jnitrace</b>              | nome de um ficheiro     | Ficheiro no qual gravar as informações de rastreio JNI para a lógica JNI utilizada na conversão de documentos (para finalidades de diagnóstico IBM)                                                                                                                                                                                                                                                                                        |
| <b>conversion</b>            | Booleano: TRUE ou FALSE | Se for TRUE, os documentos são convertidos para formatos que podem ser apresentados num navegador de escalão intermédio, se possível. Se for FALSE, o documento original, não convertido, é enviado para o navegador.                                                                                                                                                                                                                      |
| <b>maxresults</b>            | inteiro                 | Máximo de acertos devolvido; -1 (predefinição) significa todos os acertos.                                                                                                                                                                                                                                                                                                                                                                 |
| <b>valuedelimiter</b>        | carácter                | Define o carácter que irá delimitar valores em critérios de pesquisa. O carácter predefinido é dependente de locale e consiste numa vírgula (,) no caso do Inglês dos Estados Unidos.                                                                                                                                                                                                                                                      |

| Parâmetro da aplicação             | Valores                   | Descrição                                                                                                                                                                                                                                                                                                                                                                         |
|------------------------------------|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>conversion.&lt;mimetype&gt;</b> | <none   document   page > | Opções de conversão para visualizar documentos de um tipo de mime específico. Isto afecta o comportamento do servlet viewDocument. Pagar significa a tentativa de pagar o documento. Documento significa a conversão do documento para uma forma passível de ser lida num navegador. Nenhum significa a execução de nenhuma conversão -- devolver o documento à sua forma nativa. |
| <b>nameseparator</b>               | carácter                  | Define o carácter que irá separar o atributo de componente descendente a partir do atributo de componente ascendente em nomes qualificados. A predefinição é dependente do locale, e é uma barra (/) para o Inglês dos EUA.                                                                                                                                                       |

## Ficheiro de propriedades

Este servlet procura um ficheiro de propriedades, `cmbServlet.properties`. Este ficheiro define as acções que o servlet pode utilizar, incluindo as acções aqui definidas. Também define os nomes dos ficheiros de JSP que são utilizados.

O utilizador pode também definir as propriedades do servlet no servidor de aplicação da Web (motor de servlet). A sintaxe é igual à utilizada no ficheiro.

O conteúdo do ficheiro `cmbServlet.properties` é armazenado num objecto Propriedades pelo servlet de controlo. Pode ser acedido através do atributo da aplicação `cmbServletProperties`, como é demonstrado no exemplo seguinte.

### Exemplo

```
// check to see if connection pooling is enabled
String name = "connectionpool";
Properties props = (Properties) application.getAttribute(
 "cmbServletProperties");
String value = props.getProperty(name);
// "true" if enabled, "false" otherwise
```

## Parâmetros de pedido

O servlet compreende os seguintes parâmetros de pedido. Podem ser especificados parâmetros adicionais para serem utilizados na JSP de resposta.

### Geral

#### **action = action**

A acção a ser executada. Os parâmetros adicionais permitidos têm por base a acção e são descritos de seguida.

Este parâmetro é opcional. Se não for especificado, a página de resposta será executada pelo servlet após realizar a configuração padrão como, por exemplo, verificar o tempo de espera excedido ou o início de sessão na ligação.

#### **reply=<URL>**

Opcional. Envia para a JSP especificada neste parâmetro em vez da JSP definida como a resposta para a acção no ficheiro `cmbServlet.properties`. Se o parâmetro de acção não for



especificado, e a resposta for especificada, a página de resposta é executado pelo servlet do controlados depois de executar a configuração padrão, tal como a verificação da ligação para tempo excedido e início de sessão.

#### Relacionados com ligações

**action=logon serverType=<> server=<> userid=<> password=<> [connstring=<>] [configstring=<>]**

Iniciar uma sessão num servidor. Deve especificar o tipo de servidor, o nome do servidor, ID e palavra-passe de utilizador. A cadeia de ligação e a cadeia init são facultativas e diferentes, dependendo do tipo de servidor.

**action=logoff [endSession=<true|false>]**

Terminar a ligação ao servidor. A sessão é também terminada por predefinição.

#### Relacionados com a pesquisa

**action=searchTemplate template=<> {<criterionname>.op=<> <criterionname>=<>}**

Executar uma pesquisa utilizando o modelo de pesquisa e os valores de critérios especificados.

**action=searchEntityentity=<> {attribute.<attrname>.op=<> attribute.<attrname>=<>}[conjunction=<and|or>]**

Executar uma pesquisa utilizando uma entidade. Os valores e operadores de atributos também podem ser especificados. Vários valores no valor de atributo são separados por um delimitador de valor, tal como está especificado nos parâmetros de aplicação. Os atributos são combinados para formar uma consulta utilizando e(predefinição) ou ou , tal como está especificado no parâmetro de intersecção.

**action=searchQuery queryString=<> {queryParameter.<parametername>=<>}**

Executar uma pesquisa utilizando a cadeia de pesquisa especificada. A sintaxe da consulta depende do servidor que está a ser pesquisado.

Não existe um máximo de parâmetros de consulta adicionais que podem ser especificados. Estes também são dependentes do servidor.

#### Relacionados com artigos

**action=lock itemId=<>**

Bloquear um artigo, normalmente para acesso exclusivo ao actualizar.

**action=unlock itemId=<>**

Desbloquear um artigo bloqueado.

**action=createItem type=<document|folder> entity=<> {attribute.<attrname>=<attrvalue>}**

Criar um artigo. Se este for registado, o conteúdo pode ser facultado.

**action=retrieveItem itemId=<>**

Obter os atributos e o conteúdo de um artigo. É útil para garantir que o conteúdo mais recente armazenado no servidor é utilizado.

**action=updateItem itemId=<> [entity=<>]  
{attribute.<attrname>=<attrvalue>}**

Actualizar os atributos de um artigo. Se a entidade for especificada, o artigo é novamente indexado. O conteúdo é também actualizado se o servlet for invocado através de um suporte.

**action=deleteItem itemId=<>**

Eliminar um artigo.

**action=addContent itemId=<>**

Adicionar uma parte do conteúdo a um artigo. Os dados do conteúdo são registados.

**action=getContent itemId=<> contentIndex=<>**

Obtém a parte do conteúdo e devolve-a ao navegador.

**action=updateContent itemId=<> contentIndex=<>**

Actualizar uma parte do conteúdo num artigo; os dados do conteúdo são registados. Se não existir um conteúdo, é adicionada uma parte de conteúdo.

**action=deleteContent itemId=<> contentIndex=<>**

Eliminar uma parte do conteúdo para o artigo especificado.

**action=addNoteLog itemId=<>**

Modifica o registo de nota de um artigo. O texto do registo de nota é registado.

**action=updateNoteLog itemId=<> notelogIndex=<>**

Modifica o registo de nota de um artigo; o texto do registo de nota é registado. Se não existir um registo de nota, este é adicionado.

**action=deleteNoteLog itemId=<> notelogIndex=<>**

Elimina o texto do registo de nota de um artigo. O texto do registo de nota é registado.

#### Relacionados com pastas

**action=addItemToFolder itemId=<> folderId=<>**

Adiciona o artigo especificado à pasta especificada.

**action=removeItemFromFolder itemId=<> folderId=<>**

Remove o artigo especificado da pasta especificada.

#### Relacionado com documentos

**action=viewDocument itemId=<>**

Obtém o documento e revê-o. Se o documento se encontrar paginado, esta acção remete para uma JSP que gera a estrutura do visualizador. Se o documento não se encontrar paginado, esta acção devolve o conteúdo actual do documento.

**action=viewPage itemId=<> page=<> scale=<> rotation=<>  
annotations=<yes|no>**

Obtém uma página do documento.

---

## Matriz de funções de toolkit da servlet

Esta secção apresenta uma matriz de funções de toolkit da servlet.

Tabela 49. Matriz de funções da servlet

| Acção                | CMv8 | VI/400 | IP/390 | OD/Wkstn | OD/390 |
|----------------------|------|--------|--------|----------|--------|
| logon                | Y    | Y      | Y      | Y        | Y      |
| logoff               | Y    | Y      | Y      | Y        | Y      |
| searchTemplate       | N/A  | N/A    | N/A    | Y        | Y      |
| searchEntity         | Y    | Y      | Y      | Y        | Y      |
| searchQuery          | Y    | Y      | Y      | Y        | Y      |
| lock                 | Y    | Y      | Y      | N/A      | N/A    |
| unlock               | Y    | Y      | Y      | N/A      | N/A    |
| createItem           | Y    | N/A    | N/A    | N/A      | N/A    |
| retrieveItem         | Y    | Y      | Y      | Y        | Y      |
| updateItem           | Y    | Y      | Y      | N/A      | N/A    |
| deleteItem           | Y    | Y      | Y      | N/A      | N/A    |
| addContent           | Y    | N/A    | N/A    | N/A      | N/A    |
| getContent           | Y    | Y      | Y      | Y        | Y      |
| updateContent        | Y    | Y      | Y      | N/A      | N/A    |
| deleteContent        | Y    | Y      | Y      | N/A      | N/A    |
| addNoteLog           | Y    | Y      | N/A    | N/A      | N/A    |
| updateNoteLog        | Y    | Y      | N/A    | N/A      | N/A    |
| deleteNoteLog        | Y    | Y      | N/A    | N/A      | N/A    |
| addItemToFolder      | Y    | Y      | N/A    | N/A      | N/A    |
| removeItemFromFolder | Y    | Y      | N/A    | N/A      | N/A    |
| viewDocument         | Y    | Y      | Y      | Y        | Y      |
| viewPage             | Y    | Y      | Y      | Y        | Y      |

**Notas:**

**Y** A função é suportada

**N/A** A função não é suportada

As acções logon, logoff, getContent, retrieveItem, viewDocument e viewPage são suportadas em todos os servidores de conteúdos.

Tabela 50. Matriz de funções da servlet Continuação

| Acção          | Fed |
|----------------|-----|
| logon          | Y   |
| logoff         | Y   |
| searchTemplate | Y   |
| searchEntity   | Y   |
| searchQuery    | Y   |
| lock           | Y   |
| unlock         | Y   |
| createItem     | N/A |
| retrieveItem   | Y   |
| updateItem     | Y   |

Tabela 50. Matriz de funções da servlet Continuação (continuação)

| Acção                | Fed |
|----------------------|-----|
| deleteItem           | Y   |
| addContent           | Y   |
| getContent           | Y   |
| updateContent        | Y   |
| deleteContent        | Y   |
| addNoteLog           | Y   |
| updateNoteLog        | Y   |
| deleteNoteLog        | Y   |
| addItemToFolder      | N/A |
| removeItemFromFolder | N/A |
| viewDocument         | Y   |
| viewPage             | Y   |

**Notas:**

**Y** A função é suportada

**N/A** A função não é suportada

As acções login, logout, getContent, retrieveItem, viewDocument e viewPage são suportadas em todos os servidores de conteúdos.



---

## Detecção e correcção de problemas das aplicações de gestão de conteúdos

Esta secção contém os seguintes tópicos sobre detecção e correcção de problemas.

---

### Registo de conectores Java, C++ e DB2 Information Integrator for Content

Pode utilizar o utilitário de controlo de registos no cliente de administração do sistema para configurar as definições de registo para as APIs de Java e de C++.

Os utilitários de registo de conectores armazenam todas as excepções de programas, incluindo excepções que não são erros. Ocasionalmente, as mensagens de erro são apresentadas no ficheiro de registo que não é enviado ao utilizador. Em alguns casos, a API ou a aplicação do utilizador pode recuperar ou continuar sem ser necessário a interacção do utilizador. O ficheiro de registo `${user.name}.dklog.log` contém os resultados do registo das acções dos conectores DB2 Information Integrator for Content.

`${user.name}` corresponde ao ID de utilizador de início de sessão do sistema da máquina que é utilizado para qualificar o resultado do registo. O ID de utilizador de início de sessão do sistema será utilizado para substituir `${user.name}`, que ajudará a evitar colisões de resultados entre múltiplos utilizadores.

**Importante:** Ao ler os ficheiros de registo, tenha em atenção o contexto no qual as excepções e as mensagens são registados. Além disso, ao executar a depuração, concentre os resultados do registo nos erros actuais e nos eventos que os antecederam, definindo restrições no ficheiro de registo da API. Pesquisas mais limitadas podem melhorar o desempenho do sistema.

Se o cliente de administração do sistema não estiver disponível, pode configurar as APIs alterando os valores no ficheiro `cmblogconfig.properties`. No utilitário de controlo do registo, é possível alterar os nomes dos ficheiros de registo predefinidos de Java e de C++, as definições do caminho do ficheiro de registo, o tamanho máximo do ficheiro e o nível de informações de erros a enviar para o ficheiro de registo. Para obter assistência na utilização do utilitário de controlo de registos para configurar o registo da API, utilize a ajuda online da administração do sistema.

### Registo de conectores Java

Os utilizadores de Java podem utilizar dois gestores de registos: o predefinido e o `log4j-1.2.8.jar`. A instalação do produto define o gestor de registos como `log4j-1.2.8`. Pode configurar e utilizar apenas um gestor de registo de cada vez.

É utilizado o mesmo ficheiro de configuração `cmblogconfig.properties` para controlar o tipo de gestor de registos utilizado e a configuração específica de cada tipo de gestor de registos. Para obter mais informações sobre o gestor de registos, consulte a secção relativa ao gestor de registos que pretende utilizar no ficheiro `cmblogconfig.properties`.

Quando o utilitário de registo de conectores é iniciado pela primeira vez, pesquisa a CLASSPATH da instância da estação de trabalho virtual Java para localizar o ficheiro `ibmcmconfig.properties`. O caminho para o directório de trabalho é apresentado em *IBMCMWorkingDirectory*. O ficheiro de configuração de registo `cmblogconfig.properties` encontra-se no subdirectório `cmgmt` do directório de trabalho.

## Registo de conectores C++

C++ tem um gestor de registos. C++ faz referência ao mesmo ficheiro de configuração de registo do que Java. No entanto, os conectores C++ do DB2 Information Integrator for Content fazem referência apenas às definições de registo do gestor de registos predefinido.

O conector C++ depende da variável de ambiente *IBMCMROOT*.

O ficheiro `ibmcmconfig.properties` tem de estar localizado no directório *IBMCMROOT/cmgmt*.

## Modificar a configuração de registo para os conectores do DB2 Information Integrator for Content

O ficheiro de configuração está instalado no directório `cmgmt` do directório de trabalho do DB2 Content Manager. O ficheiro é denominado `cmblogconfig.properties`.

O ficheiro `cmblogconfig.properties` contém as seguintes predefinições. Não altere estas predefinições na eventualidade de não ser possível localizar o ficheiro de configuração ou de ocorrerem outros erros relacionados com as definições do utilizador:

- `Log4j` é definido como o gestor de registos predefinido. O nome do ficheiro de registo predefinido é `userid.dklog.log`. A localização predefinida do ficheiro de registo é *IBMCMWorkingDirectory/log/connectors*.
- O ficheiro `dklog.log` está localizado no directório de trabalho actual onde é executada uma aplicação activada para DB2 Information Integrator for Content.
- A prioridade de registo está definida como `Error`.
- O número máximo de excepções do mesmo identificador de mensagem de erro permitido é cinco (5).

Para actualizar as definições no ficheiro `cmblogconfig.properties`:

1. Abra o ficheiro `cmblogconfig.properties` num editor de texto.
2. Altere as definições conforme necessário de modo a cumprir os requisitos de registo do sistema.

| Opção                                   | Descrição                                                                    |
|-----------------------------------------|------------------------------------------------------------------------------|
| Section 0 - Global Settings             | Determina o número máximo de excepções.                                      |
| Section 1 - Log Manager Factory Setting | Determina se utiliza o gestor de registo predefinido ou <code>log4j</code> . |

| Opção                                 | Descrição                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Section 2 - Default Log Manager Setup | <p>A secção 2 tem três subsecções:</p> <ul style="list-style-type: none"> <li>Section 2.1 - Specify Log Priority. Estão disponíveis oito definições de prioridade. A predefinição de prioridade é Error.</li> <li>Section 2.2 - Log Output Destination Setting. Estão disponíveis três definições: <ol style="list-style-type: none"> <li>Log to a file</li> <li>Log to Standard Error</li> <li>Log to Standard Console</li> </ol> <p>A predefinição é Log to file.</p> </li> <li>Section 2.3 - Log File Name Setting. Utilize apenas quando a opção na secção 2.2 está definida como Log to file. <p>O nome do ficheiro predefinido é <i>userid.dklog.log</i>.</p> <p>A localização predefinida do ficheiro de registo é <i>IBMCMWorkingDirectory/log/connectors</i>. O caminho do ficheiro de registo tem de incluir caracteres de barra invertida dupla (\\) ou de barra única (/) em vez de barras invertidas duplas, caso sejam utilizadas. Se não especificar um caminho, o registo está sempre localizado no directório de trabalho actual, onde é executado um problema.</p> <p>A localização do ficheiro de registo é controlada pelas seguintes definições:</p> <ul style="list-style-type: none"> <li>DKLogOutputFileName para registador predefinido (C++ e Java)</li> <li>Log4j.appender.apiAppender.File para log4j</li> </ul> <p>O número máximo de ficheiros de cópia de segurança a manter definindo DKLogOutputFileSave para registador predefinido, C++ e Java ou log4j.appender.apiAppender.MaxBackupIndex para log4j.</p> <p>O tamanho máximo do ficheiro definindo DKLogOutputFileSize para registador predefinido, C++ e Java ou log4j.appender.apiAppender.MaxFileSize para log4j.</p> </li> </ul> |

3. Opcional: Modifique os seguintes níveis de prioridade no ficheiro cmblogconfig.properties.

| Opção   | Descrição                                                                                                                                                                                              |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DISABLE | Desactiva o registo.                                                                                                                                                                                   |
| FATAL   | Notificação a informar que o programa detectou erros irrecuperáveis e tem de terminar o funcionamento imediatamente. Ao parar o programa, não é possível iniciar o mesmo através da função de registo. |
| ERROR   | Notificação a informar que o programa detectou erros recuperáveis ou irrecuperáveis, mas está em condições de continuar a funcionar.                                                                   |
| PERF    | Recolhe informações de saída para medir o desempenho.                                                                                                                                                  |



| Opção            | Descrição                                                                                                                                                                                |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| INFO             | Faculta mensagens de eventos significativos como, por exemplo, início de sessão bem sucedido.                                                                                            |
| TRACE_NATIVE_API | Regista parâmetros e devolve informações de dados antes e depois de uma chamada nativa.                                                                                                  |
| TRACE_ENTRY_EXIT | Entradas e saídas de sinais de módulos de programas (ou blocos de código).                                                                                                               |
| TRACE            | Regista informações de diagnóstico adicionais como, por exemplo, alterações ao estado do programa, informações sobre parâmetros de funções e informações de valor de retorno de funções. |
| DEBUG            | Apresenta informações para depuração de erros.                                                                                                                                           |

4. Guarda o ficheiro.

## Receber um erro ao compilar aplicações de C++ que estejam activadas para Unicode

### Causa possível

As APIs C++ do Content Manager V8 não suportam Unicode.

Para armazenar objectos numa base de dados activada para Content Manager Unicode, utilizando as APIs de C++, tem de reunir as seguintes condições:

- A estação de trabalho onde o cliente está a ser executado tem de ter instalado pelo menos o fix pack 8 do DB2 Versão 7 Run-Time Client.
- O utilizador deverá processar a conversão em cadeia na sua aplicação. Por exemplo, conversão da direita para a esquerda e manuseamento de duplo byte. Todas as APIs de CM C++ podem processar cadeias de bytes ASCII com uma cadeia de caracteres terminada em zero.
- A estação de trabalho tem de ser executada no idioma local. O DB2 Run-Time Client retira a cadeia de caracteres terminada em zero no idioma local da API de C++ do CM e armazena-a no servidor de bibliotecas e no gestor de recursos activados para Unicode.
- Ao trabalhar com XDOs, algumas APIs têm um parâmetro da página de códigos que tem de ser definido como CCSID (número da página de códigos).

### Acção

Ao compilar aplicações de C++ que utilizem as APIs do Content Manager (CM) Versão 8 C++, tem de compilar com o sinalizador de Unicode para OFF.

O produto CM é instalado utilizando o inglês como idioma predefinido. No entanto, após a conclusão da instalação, para apresentar descrições de atributos de um artigo no idioma local de uma aplicação cliente, pode utilizar o cliente de administração do sistema para definir outros idiomas que já existam no sistema. Ao criar um novo atributo ou tipo de artigo, o utilizador tem a opção de apresentar o atributo ou artigo em qualquer um dos idiomas definidos no sistema.

---

## Receber um erro ao utilizar atributos de referência

### Sintomas

Está a utilizar um atributo de referência e a receber o seguinte erro quando actualiza ou adiciona um artigo: "Unexpected SQL Error (RC 7015)" with (Ext) SQL RC of '-910'.

### Causa

Em geral, não pode utilizar atributos de referência para ligar a um artigo do mesmo tipo de artigo. As referências têm como objectivo referenciar artigos de tipos de artigos diferentes. Pode criar uma referência para um artigo do mesmo tipo de artigo ao definir e modificar o atributo de referência num artigo que seja persistente. Tem de tornar as modificações persistentes apenas para o valor do atributo de referência.

Ocorre um erro inesperado de SQL com o código de retorno de SQL -910 se as alterações relativas ao atributo de referência não forem efectuadas de forma independente das outras alterações.

### Acção

Para criar um atributo de referência, complete os passos seguintes:

1. Crie os DDOs A e B.
2. Defina os outros valores em A e B, incluindo a adição de componentes descendentes.
3. Adicione A e B ao arquivo de dados.
4. Dê saída de ou bloqueie A.
5. Defina o atributo de referência de A para B.
6. Actualize A.

---

## Receber um erro quando o conjunto de ligações do WebSphere Application Server está activado

### Sintoma

Ao executar uma aplicação que utiliza o tipo de transacção de contentor para o bean da sessão e o conjunto de ligações do WebSphere Application Server está activado, o utilizador recebe o seguinte erro:

```
[11/23/05 18:50:52:934 CST] 6be0947b SystemErr???? R?????????????????
java.sql.SQLException: DSRA9350E: Operation Connection.commit
is not allowed during a global transaction.
```

### Causa possível

A API do DB2 Content Manager não suporta beans de Java quando o conjunto de ligações do WebSphere Application Server está activado.

### Acção

Para evitar este erro, pode utilizar uma das seguintes soluções:

- Utilize a propriedade TX\_NOT\_SUPPORTED com um bean de sessão gerido pelo contentor.

- Não utilize beans de Java com o conjunto de ligações do WebSphere Application Server ao utilizar as APIs do DB2 Content Manager.

---

## Receber um erro de pesquisa no idioma tailandês quando a configuração da base de dados está incorrecta

### Sintomas

Ao efectuar uma pesquisa de texto de conteúdo no idioma tailandês, pode executar pesquisas de uma palavra ou pesquisas de expressões. Quando efectua uma pesquisa de expressões, obtém um erro.

### Solução

Para suportar pesquisas de expressões em conteúdo tailandês no DB2 Content Manager, terá de criar a base de dados e o índice de texto numa das seguintes formas:

- Crie a base de dados de DB2 na página de códigos de tailandês ou em Unicode.
- Crie o índice de texto na página de códigos de tailandês com o idioma especificado como TH\_TH ou na página de códigos de tailandês CCSID 1208.

Para pesquisar uma palavra em tailandês utilizando a sintaxe da pesquisa de texto avançada: `/MyItemTextView[contains-text(ICMPARTS/@TIEREF, " 'Thai word' ")=1]`

Para pesquisar expressões em tailandês utilizando a sintaxe da pesquisa de texto avançada específica do tailandês do DB2: `/MyItemTextView[contains-text-db2(ICMPARTS/@TIEREF, \" IS ABOUT TH_TH 'Thai phrase' \")=1]`

---

## Receber um erro ao utilizar a pesquisa de texto do ICM para expressões em tailandês no DB2

### Sintomas

Quando utiliza o cliente Windows do IBM® DB2® Content Manager ou as APIs do DB2® Content Manager APIs para efectuar uma pesquisa de texto para expressões em tailandês, nem sempre obtém os resultados esperados quando o programa emissor da base de dados é o DB2.

### Causa

A cadeia de consultas predefinida gerada pelo cliente Windows do DB2® Content Manager para a pesquisa de texto está no seguinte formato:  
`/MyItemTextView[contains-text(ICMPARTS/@TIEREF, " 'Thai phrase' ")=1]`

Esta é a sintaxe de pesquisa básica e suporta apenas a pesquisa de uma palavra em tailandês de cada vez.

### Acção

Modifique a cadeia de consultas para tirar partido da sintaxe de pesquisa de texto do DB2 para pesquisar expressões em tailandês (mais do que uma palavra sem espaços entre as mesmas). `"/MyItemTextView[contains-text-db2(ICMPARTS/@TIEREF, \" IS ABOUT TH_TH 'Thai phrase' \")=1]"`

Se estiver a utilizar o cliente Windows do DB2 Content Manager, utilize a opção de pesquisa de texto avançada para especificar 'IS ABOUT TH\_TH' na caixa de parâmetros opcional antes de efectuar a pesquisa. Para obter mais informações sobre o parâmetro IS ABOUT, consulte o manual *IBM DB2 Net Search Extender Administration and User's Guide*.

---

## Não é possível adicionar, armazenar, obter, ou actualizar um artigo

### Sintomas

O utilizador pode encontrar erros quando tentar adicionar, armazenar, obter, ou actualizar artigos de recurso (artigos que estejam armazenados no gestor de recursos) ou DB2 Content Manager artigos de modelo de documento em partes.

### Causas

Os problemas mais comuns com o facto de não poder armazenar artigos de recurso é que o gestor de recursos que está a tentar ter acesso não está em execução ou não está disponível.

### Acção

Verifique que o gestor de recursos que está a tentar ter acesso se encontra em execução. Para o fazer, modifique o exemplificativo URL abaixo para indicar o seu gestor de recursos ao substituir smith.stl.ibm.com pelo nome do sistema central do seu gestor de recursos. Se o seu gestor de recursos tiver um número de porta diferente (verifique com o seu administrador de sistema), substitua o número 80 do URL por esse número para indicar o seu gestor de recursos.

Cole o URL exemplo num navegador da Web, `http://smith.stl.ibm.com:80/icrm/ICMResourceManager`. Se aparecer uma página da Web, o gestor de recursos está em execução. Deverá ser apresentada uma mensagem semelhante à que está indicada abaixo: `IBM Content Manager V8 Your request: null Return Code: 9716`

---

## Não é possível importar um objecto DKDDO do XML

### Sintoma

Está a receber uma `IllegalArgumentException` com a mensagem `DGL0303A: Parâmetro inválido`

### Causa possível

Está a utilizar o formato de data, hora ou marca de hora errado.

### Acção

Consulte a Referência de Programação Online e certifique-se de que os valores de data, hora e marca de hora nos seus ficheiros XML seguem os formatos `DKDate`, `DKTime` e `DKTimestamp` documentados no respectivo método `valueOf()`.

Se receber o erro do exemplo abaixo, o seu valor não será válido. Certifique-se de que o valor da hora esteja especificado no formato: `hh.mm.ss`. Certifique-se de que utilizou pontos finais (.) em vez de dois pontos (:).

```
java.lang.IllegalArgumentException: DGL0303A: Invalid parameter at
com.ibm.mm.sdk.common.DKTime.valueOf(DKTime.java:98) at
com.ibm.mm.sdk.common.DKXMLUtil.processElemDataValue(DKXMLUtil.java:2045) ...at
com.ibm.mm.sdk.common.DKXMLUtil.xmlImport(DKXMLUtil.java:237)at
com.ibm.mm.sdk.common.DKDDO.fromXML(DKDDO.java:285) at
XMLImporter.main(XMLImporter.java:32)
```

---

## Receber um erro ao actualizar, reorganizar ou utilizar índices de texto para componentes pesquisáveis por texto

### Sintoma

Recebe o seguinte erro ao actualizar, reorganizar ou utilizar índices de texto para componentes pesquisáveis por texto:

```
DKUsageError: DGL5203A: A palavra-passe não é válida para o ID de utilizador usado
para administrar índices de texto.; ICM7172:
A palavra-passe indicada não é válida para este ID de utilizador ou é NULL.
(STATE) : [LS RC = 7172, SQL RC = -1]
```

.

### Causas possíveis

A palavra-passe do DB2 Net Search Extender (DB2 UDB Versão 8) não está definida ou está definida incorrectamente.

### Acção

Redefina a palavra-passe no cliente de administração do sistema do DB2 Content Manager.

Conclua os seguintes passos para definir a palavra-passe:

1. Inicie sessão no cliente de administração do sistema.
2. Expanda a categoria **Library Server Parameters (Parâmetros do Servidor de Bibliotecas)** no painel da esquerda.
3. Selecione **Configurations (Configurações)** no painel da esquerda. As propriedades da configuração do servidor de bibliotecas aparecem no painel da direita.
4. Faça clique no separador **Features (Funções)**.
5. Introduza o ID de utilizador e respectiva palavra-passe do DB2 Net Search Extender.

---

## Impossível descarregar conteúdos do tipo de parte ICMBASESTREAM

### Sintoma

Quando cria um tipo de artigo de documento e adiciona um tipo de parte ICMBASESTREAM a esse documento, tem de ter definido previamente uma recolha de VideoCharger do DB2 Content Manager no servidor ou não será possível descarregar o conteúdo. Quando tenta descarregar o conteúdo, acabará por receber o seguinte erro: `R com.ibm.mm.beans.CMBException: DGL3980A: The resource manager returned an invalid response (O gestor de recursos devolveu uma resposta inválida)`

## Acção

O tipo de parte ICMBASESTREAM tem de ser sempre associado a uma recolha de VideoCharger. Quando se associam partes que não são recolhidas VideoCharger, é necessário utilizar o tipo de parte ICMBASE.

---

## A pesquisa de texto não devolve resultados para documentos criados em determinados idiomas

### Sintoma

A pesquisa de texto não devolve resultados para documentos criados em determinados idiomas.

### Acção

É necessário activar a pesquisa de texto em idiomas não Europeus ou delimitados por espaços em branco, especificando, para tal, a preferência LEXER ou usando CTX\_SYS.BASIC\_LEXER.

Crie uma preferência LEXER que seja adequada ao idioma dos documentos no índice.

Para indexar documentos em Japonês, é necessário criar uma preferência LEXER da seguinte forma:

1. Estabeleça ligação à base de dados como utilizador administrativo do DB2 Content Manager. Exemplo usando sqlplus: sqlplus id\_utilizador/p\_passe@nome\_bd
2. Crie a preferência: 

```
exec
ctx_ddl.create_preference('MY_JAPANESE_LEXER','japanese_lexer');
```
3. A preferência criada será apresentada na lista no cliente de administração do sistema. É necessário sair do cliente de administração do sistema e voltar a iniciar o cliente de administração do sistema. Em LEXER, seleccione a preferência criada quando criar o índice.

---

## DKString DKLobICM::getContent () const não recomendada para o conector ICM

### Causa possível

O método DKString DKLobICM::getContent() const não é recomendado para o conector ICM, uma vez que se a instância DKString contiver dados binários e o utilizador invocar funções de cadeia como substring() na instância, poderá receber resultados incorrectos. Não deverá utilizar dados binários com uma classe DKString.

### Acção

Use o método DKByteArray\* DKLobICM::getContent().

---

## Não é possível carregar saída de utilizador

### Sintoma

Não é possível carregar a minha saída de utilizador.

### Causa possível

A variável `$ICMDLL` pode não ter sido definida.

### Acção

A saída de utilizador tem de estar localizada no directório `$ICMDLL/ICMNLSDB/DLL`, em que `ICMNLSDB` é o nome do servidor de bibliotecas.

É também necessário executar os passos da pré-instalação para definir a variável do ambiente `$ICMDLL` na instância do DB2 Content Manager. Para saber quais estes passos, consulte as informações de *Planning and Installing Your Content Management System*.

Para apontar `$ICMDLL` para a localização correcta:

1. Estabeleça ligação à base de dados do servidor de bibliotecas.
2. Seleccione `pathicmdl` em `icmstsyscontrol`.
3. Actualize `db2InstHome/sql/lib/userprofile`:  

```
ICMDLL= <value from #2 above>
export ICMDLL
```
4. Execute `recycle db2 instance`.
5. Coloque as saídas de utilizador do servidor de bibliotecas na localização apontada nos passos anteriores:
  - Saída de Utilizador com Privilégios Gerais sobre ACL: `icmgenxt`
  - Saída de Utilizador de Pesquisa de Texto Completo: `icmdecxt`

---

## Não é possível invocar algumas APIs de administração numa transacção explícita

### Sintoma

Se invocar as APIs específicas, receberá a seguinte excepção: `DKUsageError DGL0594A: This operation cannot be performed within an explicit transaction`.

### Causas possíveis

Não é possível invocar algumas APIs de administração numa transacção explícita devido às restrições de uma base de dados subjacente. Em alguns casos, invocar as APIs pode criar ou eliminar tabelas, o que provoca bloqueios que produzem impasses ou tempos de espera.

Os seguintes exemplos de codificação podem devolver a excepção ao invocar o método `del()`:

```
//..Get an iterator to the list of item types in the system
dkCollection itemTypeDefs = dsDefICM.listEntities();
dkIterator iter = itemTypeDefs.createIterator();
//..Pick the first item type
```

```

DKItemTypeDefICM itemType = (DKItemTypeDefICM) iter.next();

//..Start an explicit transaction
_ds.startTransaction();

//..Attempt to delete an item type in an explicit transaction
// This will throw a DKUserError
itemType.del();

//..Commit the explicit transaction
_ds.commit();

```

Em seguida, apresentamos uma lista de métodos que não são permitidos em transacções explícitas:

```

DKDatastoreDefICM.add(dkEntityDef itemTypeObj); //create item type or view
DKDatastoreDefICM.del(dkEntityDef itemTypeObj); //delete item type or view
DKDatastoreDefICM.update(dkEntityDef itemTypeObj); //update item type or view
DKDatastoreDefICM.add(DKComponentTypeIndexDefICM componentTypeIndexObj);
//create component type index
DKDatastoreDefICM.del(DKComponentTypeIndexDefICM componentTypeIndexObj);
//delete component type index
DKDatastoreDefICM.update(DKComponentTypeIndexDefICM componentTypeIndexObj);
//update component type index
DKDatastoreDefICM.rebuildComponentType(String compTypeName);
//rebuild component type

DKComponentTypeIndexDefICM.add();
DKComponentTypeIndexDefICM.del();

//..The restrictions apply to the following methods in
//DKComponentTypeDefICM and all of its subclasses,
//including DKItemTypeICM, DKComponentTypeViewDefICM,
//and DKItemTypeViewDefICM.
DKComponentTypeDefICM.add();
DKComponentTypeDefICM.update();
DKComponentTypeDefICM.del();

DKUserMgmtICM.add(dkUserDef userDef); //create an user
DKUserMgmtICM.del(dkUserDef userDef); //delete an user
DKUserMgmtICM.update(dkUserDef userDef); //update an user

```

**Importante:** A restrição do tipo de artigo é aplicável aos sistemas DB2 Universal Database Content Manager.

---

## A API Connect() não verifica palavras-passe nulas ou vazias no modo de início de sessão fidedigno

### Sintoma

As aplicações clientes do DB2 Content Manager que utilizam a API connect() podem autenticar IDs de utilizador sem uma palavra-passe sempre que o modo de início de sessão fidedigno está activado.

### Causa

Quando o servidor do DB2 Content Manager estiver configurado para o modo de início de sessão fidedigno, qualquer aplicação cliente que utilize o método connect() pode efectuar o registo na biblioteca do servidor sem fornecer uma palavra-passe, uma vez que assume que o WebSphere Application Server efectua a autenticação. A API connect() não verifica uma palavra-passe nula ou vazia. Depois de a API transmitir o nível de autenticação da base de dados (ou seja, utilizando o



ID de ligação da base de dados partilhada no ficheiro INI), a API invoca o início de sessão sem uma palavra-passe. Por este motivo, quando o modo de início de sessão fidedigno está activado, o DB2 Content Manager permite que o início de sessão seja efectuado com êxito.

## Acção

Utilize sempre o método `connectWithCredential()` juntamente com o início de sessão fidedigno para forçar a autenticação através do WebSphere Application Server. A API `connectWithCredential()` valida os tokens LTPA antes de enviar o ID de utilizador para o procedimento início de sessão armazenado no servidor de biblioteca.

---

## Por vezes, a pesquisa de caracteres globais do servidor da biblioteca no z/OS pode devolver dados incorrectos

### Sintoma

Por vezes, uma pesquisa de caracteres globais do servidor da biblioteca pode devolver dados incorrectos. O resultado de erro de uma pesquisa é semelhante à seguinte excepção:

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
X !!! Exception !!! X
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Name: com.ibm.mm.sdk.common.DKUsageError
Message: DGL0303A: Invalid parameter: Component Type View Identifier
= -2147483648
com.ibm.mm.sdk.common.DKUsageError: DGL0303A: Invalid parameter: Component
Type View Identifier = -2147483648
 at com.ibm.mm.sdk.common.DKDatastoreDefICM.retrieveComponentTypeView
(DKDatastoreDefICM.java:776)
 at com.ibm.mm.sdk.server.DKResultSetCursorICM.getDDOFromCursor
(DKResultSetCursorICM.java:1761)
 at com.ibm.mm.sdk.server.DKResultSetCursorICM.loadDDOBlock
(DKResultSetCursorICM.java:1942)
 at com.ibm.mm.sdk.server.DKResultSetCursorICM.setToNext
(DKResultSetCursorICM.java:466)
 at com.ibm.mm.sdk.server.DKResultSetCursorICM.fetchNext
(DKResultSetCursorICM.java:546)
 at com.ibm.mm.sdk.server.DKDatastoreICM.evaluate
(DKDatastoreICM.java:4320)
```

### Acção

Para corrigir este problema, aplique PTF do DB2 Universal Database z/OS que corresponde ao nível do DB2 Universal Database que está a utilizar:

- PTF UK08338 (DB2 UDB zOS Edição 710)
- PTF UK08339 (DB2 UDB zOS Edição 810)

---

## Recepção de uma mensagem `DKException` quando a consulta esgota o tempo de espera

### Sintoma

A aplicação DB2 Content Manager está a enviar um valor de tempo de espera esgotado da consulta como uma das opções de consulta com todas as consultas da aplicação DB2 Content Manager. Quando a consulta esgotar o tempo de espera,

será apresentada uma mensagem `DKException`

## Solução

Quando uma consulta DB2 Content Manager esgota o tempo de espera, o método de consulta `DKDatastoreICM` lança uma `DKQueryException` com o ID de erro `DKMessageICM.DK_ICM_MSG_QUERY_TIMEOUT` (7254).

## Exemplo

Consulte o seguinte exemplo Java para conhecer a excepção de consulta numa chamada de tempo de espera esgotado.

```
dkResultSetCursor cursor = dsICM.execute(
 "/NOINDEX", DKConstantICM.DK_CM_XQPE_QL_TYPE, options);
catch (DKException exc) {
 // Check for query timeout exception
 if (exc.getErrorId() == DKMessageICM.DK_ICM_MSG_QUERY_TIMEOUT){
 // Insert exception handling for timeout here...
 } else {
 // Insert exception handling here...
 }
}
```

Para consultas de chamada de retorno (`DKDatastoreICM.executeWithCallback()`), a excepção pode ser detectada no método `reportException` do objecto de chamada de retorno. Consulte o seguinte fragmento de código Java no objecto de chamada de retorno.

```
public void reportException(DKException exc) {
 // Check for query timeout exception
 if (exc.getErrorId() == DKMessageIdICM.DK_ICM_MSG_QUERY_TIMEOUT) {
 // Insert exception handling for timeout here...
 } else {
 // Insert exception handling here...
 }
}
```

---

## O visualizador de Java não suporta documentos Excel com a protecção de documentos activada

### Sintoma

Se tiver a protecção de documentos activada num documento Excel, o documento não será apresentado correctamente.

### Causas possíveis

O visualizador de Java não suporta protecção e encriptação de dados, uma funcionalidade disponível em algumas aplicações do Microsoft Office.

### Acção

Para apresentar o documento, terá de desactivar a funcionalidade de protecção de dados no Excel.

---

## **Visualizador de Java: os documentos de texto simples UTF-8 e UTF-16 aparecem sempre alinhados à esquerda**

### **Sintoma**

O visualizador de Java suporta a visualização de documentos de texto simples UTF-8 e UTF-16. Contudo, estes documentos serão sempre alinhados à esquerda, mesmo que a totalidade do texto do documento esteja num idioma que é alinhado à direita, como o árabe ou o hebraico.

### **Acção**

Pode contornar o problema utilizando uma das seguintes opções:

- Utilize codificação hebraica ou árabe em vez de UTF-8 para documentos que sabe que estão em árabe ou hebraico.
- Utilize marcas bidireccionais no documento UTF-8 para indicar as linhas que devem ser lidas da direita para a esquerda. Tenha em atenção que com este método, a linha continuará a ficar alinhada à esquerda, mas será lida da direita para a esquerda.

---

## **Visualizador de Java: A translineação provoca o posicionamento incorrecto de anotações gráficas em relação ao texto**

### **Sintoma**

As quebras de linha entre os limites das palavras alteram o posicionamento do texto nas páginas onde é feita a transposição da palavra. As anotações gráficas estão incorrectamente posicionadas em relação ao texto. Este facto é especialmente evidente nas anotações realçadas.

### **Acção**

Existem duas soluções alternativas possíveis. Se muito poucos documentos de texto simples tiverem anotações ou se raramente tiverem translineação, ocorrerão poucos erros de alinhamento de anotações. Qualquer caso detectado pode ser corrigido manualmente. Se este método não for eficaz, a melhor solução alternativa é converter os documentos de texto simples com anotações existentes para TIFF, utilizando a versão DB2 Content Manager V8.3 Fix Pack 1 do toolkit do visualizador de Java.

---

## Avisos

Estas informações foram desenvolvidas para produtos e serviços disponibilizados nos E.U.A..

Os produtos, serviços ou funções descritos neste documento poderão não ser disponibilizados pela IBM noutros países. Consulte o seu representante local da IBM para obter informações sobre os produtos e serviços actualmente disponíveis na sua área. Quaisquer referências, nesta publicação, a produtos, programas ou serviços IBM não significam que apenas esses produtos, programas ou serviços IBM possam ser utilizados. Qualquer outro produto, programa ou serviço, funcionalmente equivalente, poderá ser utilizado em substituição daqueles, desde que não infrinja nenhum direito de propriedade intelectual da IBM. No entanto, é da inteira responsabilidade do utilizador avaliar e verificar o funcionamento de qualquer produto, programa ou serviço não IBM.

Neste documento, podem ser feitas referências a patentes ou a pedidos de patente pendentes da IBM. O facto de este documento lhe ser fornecido não lhe confere nenhum direito sobre essas patentes. Caso solicite pedidos de informação sobre licenças, tais pedidos deverão ser endereçados, por escrito, para:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

Para questões sobre licenças relacionadas com informações sobre byte duplo (DBCS), contacte o Departamento de Propriedade Intelectual da IBM no seu país ou envie as questões, por escrito, para:

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106-0032, Japan

**O parágrafo seguinte não se aplica ao Reino Unido nem a nenhum outro país onde estas cláusulas sejam inconsistentes com a lei local:** A INTERNATIONAL BUSINESS MACHINES CORPORATION FORNECE ESTA PUBLICAÇÃO “TAL COMO ESTÁ” (AS IS), SEM GARANTIA DE QUALQUER ESPÉCIE, EXPLÍCITA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS GARANTIAS IMPLÍCITAS DE NÃO INFRACÇÃO, COMERCIALIZAÇÃO OU ADEQUAÇÃO A UM DETERMINADO FIM. Alguns Estados não permitem a exclusão de garantias, explícitas ou implícitas, em determinadas transacções; esta declaração pode, portanto, não se aplicar ao seu caso.

Esta publicação pode conter imprecisões técnicas ou erros de tipografia. A IBM permite-se fazer alterações periódicas às informações aqui contidas; essas alterações serão incluídas nas posteriores edições desta publicação. A IBM pode introduzir melhorias e/ou alterações ao(s) produto(s) e/ou programa(s) descrito(s) nesta publicação em qualquer momento, sem aviso prévio.

Quaisquer referências, nesta publicação, a sítios da Web não IBM são fornecidas apenas para conveniência e não constituem, em caso algum, aprovação desses

sítios da Web. Os materiais existentes nesses sítios da Web não fazem parte dos materiais destinados a este produto IBM e a utilização desses sítios da Web será da exclusiva responsabilidade do utilizador.

A IBM pode usar ou distribuir quaisquer informações que lhe forneça, da forma que julgue apropriada, sem incorrer em nenhuma obrigação para consigo.

Os Licenciados deste programa que pretendam obter informações sobre o mesmo com o objectivo de permitir: (i) a troca de informações entre programas criados independentemente e outros programas (incluindo este) e (ii) a utilização recíproca das informações que tenham sido trocadas, deverão contactar a IBM através do seguinte endereço:

IBM Corporation  
J46A/G4  
555 Bailey Avenue  
San Jose, CA 95141-1003  
U.S.A.

Tais informações poderão estar disponíveis, sujeitas aos termos e às condições adequados, incluindo, em alguns casos, o pagamento de um encargo.

O programa licenciado descrito nestas informações e todo o material licenciado disponível para o programa são fornecidos pela IBM nos termos das Condições Gerais IBM (IBM Customer Agreement), do Acordo de Licença Internacional para Programas IBM (IPLA, IBM International Program License Agreement) ou de qualquer acordo equivalente entre ambas as partes.

Quaisquer dados de desempenho aqui contidos foram determinados num ambiente controlado. Assim sendo, os resultados obtidos noutros ambientes operativos podem variar significativamente. Algumas medições podem ter sido efectuadas em sistemas ao nível do desenvolvimento, pelo que não existem garantias de que estas medições sejam iguais nos sistemas disponíveis habitualmente. Para além disso, algumas medições podem ter sido calculadas por extrapolação. Os resultados reais podem variar. Os utilizadores deste documento devem verificar os dados aplicáveis ao seu ambiente específico.

As informações relativas a produtos não IBM foram obtidas junto dos fornecedores desses produtos, dos seus anúncios publicados ou de outras fontes de divulgação ao público. A IBM não testou esses produtos e não pode confirmar a exactidão do desempenho, da compatibilidade ou de quaisquer outras afirmações relacionadas com produtos não IBM. Todas as questões sobre as capacidades dos produtos não IBM deverão ser endereçadas aos fornecedores desses produtos.

Todas as afirmações relativas às directivas ou tendências futuras da IBM estão sujeitas a alterações ou descontinuação sem aviso prévio, representando apenas metas e objectivos.

Estas informações contêm exemplos de dados e relatórios utilizados em operações comerciais diárias. Para ilustrá-los o melhor possível, os exemplos incluem nomes de indivíduos, firmas, marcas e produtos. Todos estes nomes são fictícios e qualquer semelhança com nomes e moradas reais é mera coincidência.

#### LICENÇA DE COPYRIGHT:

Esta publicação contém programas de aplicação exemplo na linguagem origem, que ilustra técnicas de programação em várias plataformas operativas. Pode copiar,

modificar e distribuir estes programas exemplo de qualquer forma, sem encargos para com a IBM, com a finalidade de desenvolver, utilizar, comercializar ou distribuir programas de aplicação conformes à interface de programação de aplicações e destinados à plataforma operativa para a qual os programas exemplo são escritos. Estes exemplos não foram testados exaustivamente sob todas as condições. Deste modo, a IBM não garante nem se responsabiliza pela fiabilidade, assistência ou funcionamento implícito destes programas.

Algumas partes deste código são:

- Copyright © 1994-2007 EMC Corporation. Todos os Direitos Reservados
- Oracle® Outside In Viewer Technology, Copyright © 1992, 2007, Oracle. Todos os Direitos Reservados.
- Document Viewer © 1991-2007 MS Technology, Inc. Charlotte, NC. Todos os Direitos Reservados.
- Copyright © 2000-2007 The Apache Software Foundation. Todos os Direitos Reservados.
- Copyright 1994-2007 W3C (Massachusetts Institute of Technology, European Research Consortium for Informatics and Mathematics, Keio University). Todos os Direitos Reservados.
- Copyright © 1996-1999 by Scott Hudson, Frank Flannery, C. Scott Ananian. Todos os Direitos Reservados.
- Copyright ©1998-2003 The OpenSSL Project. Todos os Direitos Reservados.

Este produto é Suportado pelo Eclipse™ (<http://www.eclipse.org>).

---

## Marcas Comerciais

Este tópico descreve as marcas comerciais da IBM e algumas marcas comerciais que não pertencem à IBM.

Consulte o sítio <http://www.ibm.com/legal/copytrade.shtml> para informações sobre marcas comerciais da IBM.

Os termos que se seguem são marcas comerciais ou marcas comerciais registadas de outras empresas:

Adobe, o logotipo Adobe, PostScript e o logotipo PostScript são marcas comerciais ou marcas comerciais registadas da Adobe Systems Incorporated nos Estados Unidos e/ou noutros países.

Intel e Pentium são marcas comerciais da Intel Corporation nos Estados Unidos e/ou noutros países.

Linux é uma marca comercial da Linus Torvalds nos Estados Unidos e/ou noutros países.

Microsoft, Windows, Windows NT e o logotipo Windows são marcas comerciais da Microsoft Corporation nos Estados Unidos e/ou noutros países.

Java e todas as marcas comerciais baseadas em Java são marcas comerciais da Sun Microsystems, Inc. nos Estados Unidos e/ou noutros países.

UNIX é uma marca registada da Open Group nos Estados Unidos e noutros países.

Outros nomes de empresas, produtos e serviços podem ser marcas comerciais ou marcas de serviços de terceiros.

---

## Glossário

Este glossário define termos e abreviaturas específicas a este sistema. Os termos apresentados em *itálico* encontram-se definidos noutra local deste glossário.

### A

**classe abstracta.** Uma *classe* de programação orientada para objectos que representa um conceito; as classes derivadas da mesma representam implementações do conceito. Não é possível construir um objecto a partir de uma classe abstracta; ou seja, não pode ser instanciado.

**controlo de acessos.** O processo de garantir que determinadas funções e *objectos* armazenados só possam ser acedidos por utilizadores autorizados e por formas autorizadas.

**lista de controlo de acessos (ACL, access control list).** Uma lista que consiste num ou mais IDs de utilizadores ou grupos de utilizadores e respectivos *privilégios* associados. As listas de controlo de acessos são usadas para controlar o acesso de utilizadores a *artigos* e *objectos* no sistema DB2 Content Manager. As listas de controlo de acessos usam-se para controlar o acesso de utilizadores a *modelos de pesquisa* no sistema DB2 Information Integrator for Content.

**acção.** (1) No encaminhamento de documentos do DB2 Content Manager, especifica a forma como um utilizador pode manipular os *pacotes de trabalho* num *nó de trabalho*. O DB2 Content Manager faculta algumas acções e o utilizador pode criar as suas. As acções têm de ser incluídas numa *lista de acções* antes de poderem ser aplicadas num *nó de trabalho*. (2) No DB2 Information Integrator for Content, especifica os *artigos de trabalho* num *nó no fluxo de trabalho*. O DB2 Information Integrator for Content faculta algumas acções e o utilizador pode criar as suas. As acções têm de ser incluídas numa *lista de acções* antes de poderem ser aplicadas num *nó*.

**lista de acções.** (1) No encaminhamento de documentos do DB2 Content Manager, é um conjunto de *acções* que podem ser executadas pelo utilizador em pacotes de trabalho, num *nó de rede*. As acções que forem especificadas na lista de acções são apresentadas como opções de menu para os *pacotes de trabalho* nas *listas de trabalho* dos utilizadores cliente. (2) No DB2 Information Integrator for Content, é um conjunto de *acções* que podem ser executadas pelo utilizador em artigos de trabalho num *fluxo de trabalho*. As acções que forem especificadas na lista de acções são apresentadas como opções de menu para os *artigos de trabalho* nas *listas de trabalho* dos utilizadores cliente.

**processo ad hoc.** No encaminhamento de documentos do DB2 Content Manager, um *processo* constituído por um único passo definido pelo utilizador, normalmente para “ligar” dois outros processos.

**ACL administrativa.** Qualquer *lista de controlo de acessos (ACL)* criada por um administrador com *privilégios* SystemSetACL e SystemDefineACL e para ser usada com objectos administrativos, tais como *tipos de artigos* e vistas de tipos de artigos ou *artigos*. As ACLs criadas anteriormente à Versão 8 Edição 3 são todas ACLs administrativas.

**ADSM.** Consulte *TivoliStorage Manager*.

**API.** Consulte *interface de programação de aplicações*.

**interface de programação de aplicações (API).** Um interface de software que permite que as aplicações comuniquem entre si. Uma API é o conjunto de construções ou instruções de linguagem de programação que podem ser codificadas num programa de aplicações para obtenção das funções e serviços específicos facultados pelo programa licenciado subjacente.

**arquivo.** Armazenamento persistente usado para retenção de informações a longo prazo, normalmente de custos baixos para cada unidade armazenada e de acesso lento. Encontra-se habitualmente numa localização geográfica diferente para protecção relativamente a falhas no equipamento e desastres naturais.

**atributo.** Uma unidade de dados que descreve uma determinada característica ou propriedade (por exemplo, nome, morada, idade e assim por diante) de um artigo e que pode ser utilizada para localizar esse artigo. Um atributo possui um tipo, que indica o intervalo da informação armazenada por esse atributo, e um valor, que se encontra dentro desse intervalo. Por exemplo, informações sobre um ficheiro de um sistema de ficheiros multimédia, tais como título, duração e tipo de codificação (MPEG1, H.263 e assim por diante). Para o DB2 Information Integrator for Content, consulte também *atributo federado* e *atributo nativo*.

**grupo de atributos.** Agrupamento conveniente de um ou mais *atributos*. Por exemplo, Morada pode incluir os atributos Rua, Cidade e Código Postal.

**Audio/Video Interleaved (AVI).** Uma especificação de ficheiro RIFF (*Resource Interchange File Format*) que permite que dados áudio e vídeo sejam intercalados num ficheiro. As pistas independentes podem ser



accedidas em porções alternadas para reprodução ou gravação mantendo o acesso sequencial no dispositivo de ficheiro.

**AVI.** Consulte *Audio/Video Interleaved*.

## B

**atributos base.** Um conjunto de índices que é atribuído a cada *objecto*. Todos os *objectos* do DB2 Content Manager têm *atributos base*.

**objecto binário grande (BLOB, binary large object).** Uma sequência de bytes com um tamanho que vai de 0 bytes a 2 gigabytes. Esta cadeia não tem página de códigos nem conjunto de caracteres associados. Os *objectos* de imagem, áudio e vídeo são armazenados em BLOBs. Consulte também *objecto de caracteres grande (CLOB, character large object)*.

**BLOB.** Consulte *objecto binário grande*.

**aplicação empresarial.** No encaminhamento de documentos do DB2 Content Manager, é um *nó de trabalho* que direcciona o trabalho para uma aplicação empresarial externa desenvolvida pelo utilizador. O *nó de trabalho* da aplicação empresarial tem uma DLL identificada ou uma biblioteca partilhada que é executada no servidor e pode iniciar uma aplicação empresarial externa, tal como um programa CICS ou IMS.

## C

**memória cache.** Uma memória tampão com uma finalidade particular, mais pequena e mais rápida do que a memória principal, utilizada para guardar uma cópia de dados que possam ser acedidos com frequência. A utilização da memória cache reduz o tempo de acesso mas pode fazer aumentar as necessidades em termos de memória. Consulte também *memória cache do gestor de recursos* e *memória cache da LAN*.

**cardinalidade.** O número de linhas numa tabela de base de dados.

**categoria.** Consulte *tipo de artigo*.

**CGI.** Consulte *Common Gateway Interface*.

**script CGI.** Um programa de computador executado num servidor Web e que utiliza a *Common Gateway Interface (CGI)* para executar tarefas que não são normalmente efectuadas por um servidor Web (por exemplo, acesso à base de dados e processamento de formulários). Um script CGI é um programa CGI escrito numa linguagem de scripts como, por exemplo, Perl.

**objecto de caracteres grande (CLOB, character large object).** Uma sequência de caracteres (byte único, vários bytes ou ambas) com um tamanho que vai de 0 bytes a 2 gigabytes menos 1 byte. Consulte também *objecto binário grande (BLOB, binary large object)*.

**componente descendente.** Segundo nível (ou inferior) opcional de um *tipo de artigo* hierárquico. Cada componente descendente está directamente associado ao nível acima.

**CIF.** Consulte *ficheiro de intercâmbio comum*.

**CIFS.** Consulte *Common Internet File System*.

**CIU.** Consulte *unidade de intercâmbio comum*.

**classe.** Em concepção e programação orientada para *objectos*, consiste num modelo que pode ser iniciado para criar *objectos* com uma definição comum e por conseguinte, propriedades, operações e comportamentos comuns. Um *objecto* é uma instância de uma classe.

**aplicação cliente.** Uma aplicação escrita com as APIs do DB2 Content Manager para personalizar uma interface de utilizador. Uma aplicação escrita com APIs orientadas para *objectos* ou de Internet para aceder a *servidores de conteúdos* a partir do DB2 Information Integrator for Content.

**Aplicação Cliente paraWindows.** Um sistema completo de gestão de *objectos* fornecido com o DB2 Content Manager e escrito com APIs do DB2 Content Manager. Suporta a criação, o armazenamento, a apresentação, o processamento e o controlo de acessos a documentos e pastas. Poderá personalizá-lo com rotinas de saídas de utilizador e chamá-lo parcialmente com APIs.

**cliente/servidor.** Em comunicações, o modelo de interacção em processamento de dados distribuído, no qual um programa num dado local envia um pedido a um programa noutra local e fica a aguardar a sua resposta. O programa que faz o pedido é chamado cliente; o programa que responde é chamado servidor.

**CLOB.** Consulte *objecto de caracteres grande*.

**recolha.** Grupo de *objectos* com um conjunto semelhante de regras de gestão.

**lista de eventos de recolha.** No fluxo de trabalho avançado do DB2 Information Integrator for Content, são os critérios usados por um ponto de recolha para determinar qual o caminho a seguir pela pasta federada. Corresponde a uma quantidade específica de tempo ou a uma lista de eventos, que são *nós de eventos*. Cada ponto de recolha tem de ter, pelo menos, duas listas de eventos de recolha: uma com um período de tempo de espera (o caminho de tempo de espera) e

a outra com uma lista de eventos que têm de ocorrer para que a pasta federada possa prosseguir (caminho direccionado por eventos).

**ponto de recolha.** (1) No encaminhamento de documentos do DB2 Content Manager, um *nó de trabalho* especial no qual a pasta aguarda pela chegada de outros documentos ou pastas, mas que não corresponde a uma tarefa empresarial. (2) No fluxo de trabalho avançado do DB2 Information Integrator for Content, um *nó* especial no qual uma *pasta federada* aguarda pela chegada de outros objectos, tais como documentos ou pastas ou que sejam atingidas determinadas condições. Um ponto de recolha consiste num *nó* de recolha, num a 20 *nós de eventos* e em dois ou mais conectores denominados *listas de eventos de recolha*.

**pesquisa combinada.** Uma consulta que combina um ou mais dos seguintes tipos de pesquisas: *paramétrica*, texto ou imagens.

**Common Gateway Interface (CGI).** Uma norma para o intercâmbio de informações entre um servidor da Web e programas que lhe são externos. Os programas externos podem ser escritos em qualquer linguagem de programação suportada pelo sistema operativo no qual o servidor da Web está a ser executado. Consulte também *script de CGI*.

**ficheiro de intercâmbio comum (CIF, common interchange file).** m ficheiro que contém uma cadeia de dados do ImagePlus Interchange Architecture (IPIA).

**unidade de intercâmbio comum (CIU, common interchange unit).** A unidade independente de transferência para um ficheiro comum de intercâmbio (CIF, common interchange file). É a parte do CIF que identifica as relações nas bases de dados de recepção. Um CIF pode incluir várias CIUs.

**Common Internet File System (CIFS).** Um protocolo que permite a colaboração na Internet definindo um protocolo de acesso a ficheiros remotos que é compatível com a forma como as aplicações já partilham dados em discos locais e em servidores de ficheiros em rede.

**componente.** Termo genérico para um *componente raiz* ou um *componente descendente*.

**gestor de ligações.** Um componente do DB2 Content Manager que ajuda a manter ligações ao servidor de bibliotecas, em vez de iniciar uma nova ligação para cada consulta. O gestor de ligações tem uma interface de programação de aplicações.

**classe de conectores.** Uma *classe* de programação orientada para objectos que fornece o acesso padrão a APIs que são nativas de *servidores de conteúdos* específicos.

**construtor.** Em linguagens de programação, consiste num método com o mesmo nome que a classe e que é utilizado para criar e inicializar objectos dessa classe.

**contentor.** Um elemento da interface de utilizador que retém objectos. No *gestor de pastas*, é um *objecto* que pode conter outras pastas ou documentos.

**classe de conteúdos.** Consulte *tipo MIME*.

**servidor de conteúdos.** Um repositório para dados multimédia, formulários empresariais, documentos e dados relacionados e os metadados relacionados necessários para os utilizadores trabalharem com os conteúdos. O DB2 Content Manager e o ImagePlus for OS/390 são exemplos de servidores de conteúdos.

**cursor.** Uma estrutura de controlo denominada utilizada por um programa de aplicações para apontar para uma linha específica num conjunto ordenado de linhas. O cursor é utilizado para obter linhas a partir do conjunto.

## D

**formato de dados.** Consulte *tipo MIME*.

**arquivo de dados.** (1) Termo genérico para designar um local (tal como um sistema de bases de dados, ficheiro ou directório) onde os dados são armazenados. (2) Num programa de aplicação, uma representação virtual de um *servidor de conteúdos*.

**DCA.** Consulte *arquitectura de conteúdos de documentos*.

**DDO.** Consulte *objecto de dados dinâmicos*.

**transferidor.** Uma função do *gestor de recursos* do DB2 Content Manager que moves objectos da *área de transferência ascendente* para o primeiro passo da *política de migração* do objecto.

**gestor de dispositivos.** No sistema DB2 Content Manager, é a interface entre o *gestor de recursos* e um ou mais dispositivos físicos.

**documento.** Um *artigo* que pode ser armazenado, obtido e trocado entre sistemas do DB2 Content Manager e entre e utilizadores como uma unidade separada. Um artigo com um *tipo semântico* e documento deve conter informação que forma um documento, mas isso não implica que seja necessariamente uma implementação do modelo de documento do DB2 Content Manager.

Um artigo criado a partir de um documento classificado como tipo de artigo (uma implementação específica de um modelo de documento do DB2 Content Manager), deve conter partes de documento. Pode utilizar documentos classificados como tipos de artigo para criar artigos com um tipo semântico de documento ou de pasta.

As partes de documento podem incluir diferentes tipos de conteúdo, incluindo, por exemplo, texto, imagens e folha de cálculo.

#### **arquitectura do conteúdo de documentos (DCA).**

Uma arquitectura que garante a integridade das informações de um documento que é trocado no sistema de rede de um escritório. A DCA fornece a regra de especificação da forma e significado de um documento. Define texto de forma sujeita a revisão (alterável) e texto de forma final (não alterável).

**processo de encaminhamento de documentos.** No DB2 Content Manager, é uma sequência de *passos de trabalho* e regras que governam esses passos, através da qual passa um *documento* ou uma *pasta* que está a ser processado/a.

**definição de tipo de documento (DTD, document type definition).** As regras que especificam a estrutura de uma classe particular de documentos XML. A DTD define a estrutura com elementos, atributos e notações e estabelece restrições para o modo como cada elemento, atributo e notação pode ser utilizado dentro de uma classe particular de documentos. Uma DTD é análoga a um esquema de base de dados, na medida em que a DTD descreve completamente a estrutura para dada linguagem de markup.

**DTD.** Consulte *definição de tipo de documento*.

**objecto de dados dinâmico (DDO, dynamic data object).** Num programa de aplicações, uma representação genérica de um objecto armazenado utilizada para inserir e retirar esse objecto do armazenamento.

## **E**

**elemento.** Um *objecto* que é atribuído pelo *gestor de listas* a uma aplicação.

**nó de eventos.** No fluxo de trabalho avançado do DB2 Information Integrator for Content, é o conjunto de critérios que especifica os objectos ou condições necessárias a um nó de recolha. Cada ponto de recolha pode incluir até 20 nós de eventos.

**objecto de dados expandido (XDO, extended data object).** Num programa de aplicações, uma representação genérica de um *objecto* multimédia complexo armazenado utilizada para inserir e retirar esse objecto do armazenamento. Os XDOs estão normalmente contidos em DDOs.

**Extensible Markup Language (XML).** Uma metalinguagem padrão para definir linguagens de markup; derivado e subconjunto de SGML. A XML omite as partes mais complexas e menos utilizados da SGML e facilita a concepção de aplicações para processar tipos de documentos, criar e gerir informações estruturadas e transmitir e partilhar

informações estruturadas através de sistemas informáticos diversos. A utilização de XML não requer as aplicações e o processamento robustos necessários à SGML. A XML está a ser desenvolvida com o apoio do World Wide Web Consortium (W3C).

## **F**

**função.** As informações de conteúdo visual armazenadas no servidor de pesquisa de imagens. Além disso, as características visuais utilizadas pelas aplicações de pesquisa de imagens para determinar correspondências. As quatro funções do QBIC são a cor média, a cor de histograma, a cor posicional e a textura.

**atributo federado.** Uma categoria de metadados do DB2 Information Integrator for Content que é correlacionada com *atributos nativos* num ou mais *servidores de conteúdos*. Por exemplo, o atributo federado número de política, pode ser correlacionado com um *atributo*, nº de apólice, no DB2 Content Manager e com um atributo ID de política, no ImagePlus for OS/390.

**recolha federada.** Um grupo de objectos que resultam de uma *pesquisa federada*.

**arquivo de dados federado.** Representação virtual de qualquer número de *servidores de conteúdos* específicos, como por exemplo, o DB2 Content Manager.

**entidade federada.** Um objecto de metadados do DB2 Information Integrator for Content composto por *atributos federados* e opcionalmente associado a um ou mais *índices de texto federados*.

**pasta federada.** No DB2 Information Integrator for Content, uma pasta com objectivos especiais que armazena entidades nativas de um ou mais servidores de conteúdos.

**pesquisa federada.** Uma consulta emitida a partir do DB2 Information Integrator for Content que pesquisa simultaneamente dados em um ou mais *servidores de conteúdos*, os quais podem ser heterogéneos.

**índice de texto federado.** Um objecto de metadados do DB2 Information Integrator for Content que é correlacionado com um um ou mais *índices de texto nativo* num ou mais *servidores de conteúdos*.

**sistema de ficheiros.** Em sistemas UNIX, o método de dividir em partições uma unidade de disco rígido para armazenamento. .

**pasta.** Um *artigo* de qualquer *tipo de artigo*, independentemente da classificação, com a pasta *tipo semântico*. Qualquer artigo com a pasta tipo semântico contém uma função específica de pasta que é facultada pelo DB2 Content Manager, além de capacidades de artigos que não sejam de recurso e qualquer outra

função disponível numa classificação de tipo de artigo, como *documento* ou artigo de recurso. As pastas podem conter inúmeros artigos de qualquer tipo, incluindo documentos e subpastas. Cada pasta é indexada por *atributos*.

**gestor de pastas.** O modelo do DB2 Content Manager para gerir dados como pastas e documentos online. Pode utilizar as APIs do gestor de pastas como interface principal entre as suas aplicações e os servidores de conteúdos do DB2 Content Manager.

## H

**parâmetro identificador.** Uma cadeia de caracteres que representa um objecto e é utilizada para obter o documento.

**histórico.** Um ficheiro que mantém um registo de actividades relativo a um *fluxo de trabalho*.

**HTML.** Consulte *Hypertext Markup Language*.

**Hypertext Markup Language (HTML).** Uma linguagem de markup em conformidade com a norma SGML, principalmente concebida para suportar a apresentação online de informações gráficas e textuais que incluam ligações de hipertexto.

## I

**Image Object Content Architecture (IOCA).** Uma recolha de instruções utilizadas para intercâmbio e apresentação de imagens.

**índice.** Usa-se para adicionar ou editar os valores do atributo que identificam um *artigo* ou *objecto* específico de modo a ser obtido mais tarde.

**classe de índice.** Consulte *tipo de artigo*.

**subconjunto de classe de índice.** No DB2 Content Manager anterior, uma vista de uma *classe de índice* que uma aplicação utiliza para armazenar, recuperar e apresentar pastas e objectos.

**vista de classe de índice.** No DB2 Content Manager anterior, o termo utilizado em APIS para o *subconjunto de classe de índice*.

**information mining (extração de informações).** O processo automatizado de extrair informações essenciais de texto (resumo), localizando temas predominantes numa recolha de documentos (categorização) e pesquisando documentos relevantes utilizando consultas flexíveis e potentes.

**incluído (inline).** No DB2 Content Manager, um objecto que está online e numa unidade, mas sem *instalações activas*. Compare com *instalado*.

**intercâmbio.** A capacidade de importar ou exportar uma imagem com o respectivo índice, de um sistema de ImagePlus for OS/390 para outro sistema ImagePlus utilizando um *ficheiro de intercâmbio comum* ou uma *unidade de intercâmbio comum*.

**IOCA.** Consulte *Image Object Content Architecture*.

**artigo.** No DB2 Content Manager, termo genérico para uma instância de um *tipo de artigo*. Por exemplo, um artigo pode ser uma *pasta*, um *documento*, um vídeo ou uma imagem. Termo genérico para designar a unidade mais pequena de informação administrada pelo DB2 Information Integrator for Content. Cada artigo tem um identificador. Por exemplo, um artigo pode ser uma *pasta* ou um *documento*.

**tipo de artigo.** Um modelo para definir e mais tarde localizar *artigos* semelhantes que consiste num *componente raiz*, zero ou mais *componentes descendentes* e uma *classificação*.

**classificação do tipo de artigo.** Uma categorização dentro de um *tipo de artigo* que identifica os *artigos* desse tipo de artigo. Todos os artigos do mesmo tipo de artigo partilham a mesma classificação de tipo de artigo.

O DB2 Content Manager fornece as seguintes classificações de tipo de artigo: *pasta*, *documento*, *objecto*, *vídeo*, *imagem* e *texto*; os utilizadores também podem definir as suas próprias classificações de tipo de artigo.

**gente iterativo.** Uma classe ou construção que pode ser utilizada para se deslocar por uma recolha de *objectos*, um de cada vez.

## J

**JavaBeans.** É uma tecnologia de componente de software, independente de plataforma, para construção de componentes Java reutilizáveis denominados "beans." Depois da respectiva construção, estes beans podem ser disponibilizados para utilização por outros engenheiros de software ou pode ser usado em aplicações Java. Com os JavaBeans, os programadores poderão manipular e montar beans num ambiente de desenvolvimento gráfico de arrastar e largar.

**Joint Photographic Experts Group (JPEG).** (1) Um grupo que estabeleceu a norma para a compressão de imagens de tom contínuo digitalizadas. (2) A norma para fotografias desenvolvidas por este grupo.

**JPEG.** Consulte *Joint Photographic Experts Group*.

## K

**campo-chave.** Consulte *atributo*.



## L

**LAN.** Consulte *rede local*.

**Memória cache de rede local.** Uma área de armazenamento temporário num *gestor de recursos* local que contém uma cópia de objectos armazenados num gestor de recursos remoto.

**cliente da biblioteca.** O componente de um sistema DB2 Content Manager que fornece uma interface de programação de baixo nível ao sistema de bibliotecas. O cliente da biblioteca inclui APIs que fazem parte do conjunto de ferramentas do programador de software.

**objecto da biblioteca.** Consulte *artigo*

**servidor de bibliotecas.** O componente de um sistema DB2 Content Manager que armazena, gere e processa consultas em *artigos*.

**ligação.** Relação direccional entre dois *artigos*: a origem e o destino. Pode utilizar um conjunto de ligações para construir associações de um para vários. Compare com *referência*.

**rede local (LAN).** Uma rede na qual um conjunto de diapositivos estão ligados uns aos outros para comunicação e que podem ser ligados a uma rede mais abrangente.

## M

**estrutura de dados gerados pela máquina (MGDS, machine-generated data structure).** (1) Um protocolo de formato de dados estruturados da IBM para transmitir dados de caracteres entre os vários programas de ImagePlus for OS/390. (2) Dados extraídos de uma imagem e colocados no formato de sequência de dados normal (GDS).

**classe de gestão.** O termo utilizado nas APIs para a *política de migração*.

**arquivador multimédia.** Um dispositivo físico utilizado para armazenar sequências de dados de áudio e vídeo. O VideoCharger é um tipo de arquivador multimédia.

**classe de objectos multimédia.** Classificação que descreve os dados contidos num objecto e a forma de actuar sobre esses dados. O DB2 Content Manager faculta quatro classes de objectos multimédia predefinidas: DKLobICM, DKStreamICM, DKTextICM e DKVideoStreamICM. Semelhante ao *tipo MIME*.

**servidor multimédia.** Um componente baseado em AIX do sistema DB2 Content Manager utilizado para armazenar e aceder a ficheiros de vídeo.

**função de membro.** Funções e operadores de C++ declarados como membros de uma classe. Uma função

de membro tem acesso aos membros de dados privados e protegidos e às funções de membros de um objecto da respectiva classe.

**método.** (1) Na concepção ou programação Java, o software que implementa o comportamento especificado por uma operação. (2) Sinónimo de *função de membro* em C++.

**MGDS.** Consulte: *estrutura de dados gerada por máquina*.

**migração.** (1) O processo de mover dados e origem de um sistema de computadores para outro sem converter os dados, como acontece quando são movidos para um novo ambiente operativo. (2) Instalação de uma nova versão ou edição de um programa para substituir uma versão ou edição anterior.

**política de migração.** Uma marcação definida pelo utilizador para mover *objectos* de uma *classe de armazenamento* para a seguinte. Descreve as características de transição de classes e retenção de um grupo de objectos numa hierarquia de armazenamento.

**migrador.** Uma função do *gestor de recursos* que verifica as *políticas de migração* e move objectos para a *classe de armazenamento* seguinte quando existir uma marcação para mover objectos.

**tipo MIME.** Uma norma da Internet para identificar o tipo de objecto que está a ser transferido através da Internet. Os tipos MIME incluem múltiplas variantes de áudio, imagem e vídeo. Cada objecto possui um tipo MIME.

**Mixed Object Document Content Architecture (MO:DCA).** É arquitectura da IBM desenvolvida para permitir o intercâmbio de dados de objectos entre aplicações no ambiente de intercâmbio e entre ambientes.

**Mixed Object Document Content Architecture–Presentation (MO:DCA–P).** É uma arquitectura de sub-conjuntos do MO:DCA usada como invólucro para conter documentos enviados para a estação de trabalho de ImagePlus for OS/390, para apresentação ou impressão.

**MO:DCA.** *Mixed Object Document Content Architecture*

**MO:DCA–P.** *Mixed Object Document Content Architecture—Presentation*

**montar.** Colocar um suporte de dados numa posição que lhe permita funcionar.

**montado.** No DB2 Content Manager, um objecto que está online e numa unidade, com *montagens* activas. Compare com *incluído (inline)*.

**multimédia.** Combinação de diferentes meios de suporte de dados (texto, gráfico, áudio, imagem parada, vídeo, animação) para apresentação e controlo a partir de um computador.

**sistema de ficheiros multimédia.** Um *sistema de ficheiros* que está optimizado para o armazenamento e processamento de vídeo e áudio.

#### **Multipurpose Internet Mail Extensions (MIME) .**

Uma norma da Internet para identificar o tipo de objecto que está a ser transferido através da Internet. Consulte também *tipo MIME*.

## **N**

**NAS.** Consulte *armazenamento ligado em rede*.

**atributo nativo.** Uma característica de um objecto gerido em determinado *servidor de conteúdos* e que é específico a esse servidor de conteúdos. Por exemplo, *campo-chave* nº política pode ser um atributo nativo num servidor de conteúdos do DB2 Content Manager, ao passo que o campo ID de política poderá ser um atributo nativo num servidor de conteúdos do DB2 Content Manager OnDemand.

**entidade nativa.** Um *objecto* gerido num *servidor de conteúdos* específico e que é composto por *atributos nativos*. Por exemplo, as *classes de índice* do DB2 Content Manager são entidades nativas compostas por *campos-chave* do DB2 Content Manager.

**índice de texto nativo.** Um índice dos *artigos* do texto que são geridos num *servidor de conteúdos* específico. Por exemplo, um índice de pesquisa de texto único num servidor de conteúdos do DB2 Content Manager.

**armazenamento ligado em rede (NAS, network-attached storage).** Uma tecnologia na qual um sistema de armazenamento integrado é anexado a uma rede de serviço de mensagens que usa protocolos de comunicações comuns, tal como o TCP/IP.

**ficheiro de tabelas de rede.** Ficheiro de texto que contém as informações de configuração específicas do sistema para cada nó de um sistema DB2 Content Manager. Cada nó do sistema tem de ter um ficheiro de tabelas de rede que identifique o nó e enuncie os nós com os quais tem de estabelecer ligação.

O nome de uma tabela de rede é FRNOLINT.TBL.

**nó.** No fluxo de trabalho avançado do DB2 Information Integrator for Content, é um termo genérico para qualquer ponto discreto num processo de fluxo de trabalho.

## **O**

**objecto.** Qualquer conteúdo digital que um utilizador possa armazenar, obter e manipular como uma unidade

simples, por exemplo, imagens JPEG, áudio MP3, vídeo AVI e um bloco de texto de um livro.

**Ligação e Incorporação de Objectos (OLE, Object Linking and Embedding).** Especificação da Microsoft para aplicações de ligação e incorporação para que possam ser activadas a partir de outras aplicações.

**servidor de objectos.** Consulte *gestor de recursos*.

**memória cache do servidor de objectos.** Consulte *memória cache do gestor de recursos*.

**OLE.** Consulte *Ligação e Incorporação de Objectos (OLE, Object Linking and Embedding)*.

**sobreposição.** Recolha de dados predeterminados, como por exemplo, linhas, sombra, texto, caixas ou logótipos, que podem ser intercalados com dados variáveis numa página, durante a impressão.

## **P**

**pacote.** Recolha de interfaces e *classes* relacionadas que permitem protecção de acesso e gestão de espaço de nome.

**pesquisa paramétrica.** Consulta de *objectos* baseada nas *propriedades* dos *objectos*.

**parte.** Consulte *objecto*.

**patrono.** O termo utilizado nas APIs do DB2 Content Manager para *utilizador*.

**identificador persistente (PID, persistent identifier).** Identificador que identifica de forma exclusiva um *objecto*, independentemente da respectiva localização. O PID consiste num ID de artigo e uma localização.

**PID.** Consulte *identificador persistente (PID, persistent identifier)*.

**privilégio.** Direito de aceder a um determinado *objecto* de uma determinada forma. Os privilégios incluem direitos como por exemplo a criação, eliminação e selecção de *objectos* armazenados no sistema. Os privilégios são atribuídos pelo administrador.

**conjunto de privilégios.** Uma recolha de *privilégios* para trabalhar com funções e componentes do sistema. O administrador atribui conjuntos de privilégios a utilizadores (IDs de utilizador) e *grupos de utilizadores*.

**processo.** No encaminhamento de documentos do DB2 Content Manager, uma série de passos através dos quais o trabalho é encaminhado. Um processo contém, pelo menos, um nó de início, um *nó de trabalho* e um nó de paragem.

**propriedade.** Uma característica de um *objecto* que descreve o objecto. Uma propriedade pode ser alterada ou modificada. O estilo de escrita é um exemplo de uma propriedade.

**supressor.** Uma função do *gestor de recursos* que remove *objectos* do sistema.

## Q

**QBIC.** Consulte *Query by Image Content*.

**Query by Image Content (QBIC).** Uma tecnologia de consulta que permite pesquisas baseadas no conteúdo visual, denominadas funções, em vez de somente no texto simples. Com o QBIC, pode pesquisar *objectos* segundo as suas características visuais tais como a cor e a textura.

**cadeia de consulta.** Uma cadeia de caracteres que especifica as propriedades e valores de propriedades para uma consulta. Pode criar a cadeia de consulta numa aplicação e transmiti-la à consulta.

## R

**ordem.** Um valor de número inteiro que significa a importância de uma determinada parte para os resultados de uma consulta. Uma ordem mais elevada significa uma correspondência mais exacta.

**ficheiro README.** Um ficheiro que deve ser consultado antes da instalação ou execução do programa ao qual está associado. Um ficheiro README contém, normalmente, informações de última hora referentes ao produto, informações sobre a instalação ou sugestões relativas à utilização do produto.

**referência.** Associação de direcção única, um-a-um entre um componente raiz ou *componente descendente* e outro *componente raiz*. Comparar com *ligação*.

**libertar.** Remover a suspensão de critérios de um *artigo*. Um artigo suspenso é libertado quando os critérios forem cumpridos ou quando um utilizador com a autoridade adequada sobrepuser os critérios e o libertar manualmente.

**Remote Method Invocation (RMI).** Um conjunto de APIs que permite a programação distribuída. Um *objecto* numa Java Virtual Machine (JVM) pode invocar métodos em *objectos* noutras JVMs.

**converter.** Tornar dados que não consistam normalmente em imagens e apresentá-los como uma imagem. No DB2 Content Manager, os documentos de processamento de texto podem ser convertidos como imagens para fins de apresentação.

**Resource Interchange File Format (RIFF) .** Utilizado para armazenar som e gráficos para reprodução em diferentes tipos de equipamento informático.

**gestor de recursos.** O componente de um sistema DB2 Content Manager que gere *objectos*. Estes *objectos* são referidos pelos *artigos* armazenados no *servidor de bibliotecas*.

**memória cache do gestor de recursos.** A área de armazenamento de trabalho para o *gestor de recursos*. Também chamada *área de transferência ascendente*.

**RIFF.** Consulte *Resource Interchange File Format*.

**servidor RMI.** Um servidor que implementa o modelo de *objectos* distribuídos do *Remote Method Invocation (RMI)* de Java.

**componente raiz.** O primeiro e único nível de um *tipo de artigo* hierárquico que consiste em *atributos* relacionados com o sistema e definidos pelo utilizador.

## S

**critérios de pesquisa.** No DB2 Content Manager, valores de *atributo* usados para obtenção de um *artigo* armazenado. No DB2 Information Integrator for Content, campos específicos que um administrador define para um *modelo de pesquisa* que limita ou define ainda mais as opções disponíveis para os *utilizadores*.

**modelo de pesquisa.** Um formulário, composto por *critérios de pesquisa*, concebidos por um administrador, para um tipo específico de pesquisa federada. O administrador identifica também os *utilizadores* e *grupos de utilizadores* que podem aceder a cada modelo de pesquisa.

**tipo semântico.** A utilização ou regras para um *artigo*. Base, anotação ou nota são tipos semânticos facultados pelo DB2 Content Manager; os utilizadores podem também definir os seus próprios tipos semânticos.

**definição de servidor.** As características de um *servidor de conteúdos* específico que o identifica de modo único perante o DB2 Information Integrator for Content.

**inventário do servidor.** A lista abrangente de *entidades nativas* e *atributos nativos* de *servidores de conteúdos* específicos.

**definição do tipo de servidor.** A lista de características, tal como identificadas pelo administrador, necessárias para identificar de forma exclusiva um servidor personalizado de um determinado tipo perante o DB2 Information Integrator for Content.

**SMS.** Consulte *armazenamento gerido pelo sistema*.

**transferência ascendente.** O processo de mover um *objecto* armazenado a partir de um dispositivo off-line e de baixa prioridade para um dispositivo on-line e de elevada prioridade, normalmente a pedido do sistema ou a pedido de um utilizador. Quando um utilizador pede um *objecto* armazenado em memória permanente, uma cópia de trabalho é gravada na *área de transferência ascendente*.

**área de transferência ascendente.** A área de armazenamento de trabalho para o *gestor de recursos*. Também referida como *memória cache do gestor de recursos*.

**sistema autónomo.** Um sistema DB2 Content Manager pré-configurado que instala todos os componentes de um sistema DB2 Content Manager num único computador pessoal.

**classe de armazenamento.** Identifica o tipo de suporte de dados em que determinado *objecto* está armazenado. Não está directamente associada a uma localização física, mas está directamente associada ao *gestor de dispositivos*. Os tipos de classe de armazenamento incluem:

- Disco fixo
- VideoCharger
- Arquivo multimédia
- Tivoli Storage Manager (incluindo óptico, por sequência e banda)

**grupo de armazenamento.** Associa um *sistema de armazenamento* a uma *classe de armazenamento*.

**sistema de armazenamento.** Um termo genérico para armazenamento no sistema do DB2 Content Manager. Consulte também *volume TSM*, *arquivador multimédia* e *volume*.

**dados sequenciados.** Quaisquer dados enviados através de uma ligação em rede a uma velocidade especificada. Uma sequência pode ser um tipo de dados ou uma combinação de tipos. As velocidades dos dados, expressas em bits por segundo, variam conforme os diferentes tipos de sequências e redes.

**subclasse.** Uma *classe* derivada de outra classe. Uma ou mais classes podem estar entre a classe e a subclasse.

**subprocesso.** No encaminhamento de documentos do DB2 Content Manager, um processo existente que definiu para execução noutro processo.

**sub-fluxo de trabalho.** No fluxo de trabalho avançado do DB2 Information Integrator for Content, um processo de fluxo de trabalho existente que é verificado no servidor de fluxos de trabalho que definiu para execução noutro fluxo de trabalho.

**superclasse.** Uma *classe* da qual deriva uma classe. Uma ou mais classes podem estar entre a classe e a superclasse.

**suspender.** Remover um *objecto* do respectivo *fluxo de trabalho* e definir os critérios de suspensão necessários para o activar. A activação posterior do *objecto* permite-lhe continuar o processamento.

**armazenamento gerido pelo sistema (SMS, system-managed storage).** A abordagem do DB2 Content Manager à gestão de armazenamento. O sistema determina a colocação de *objectos* e gere automaticamente cópia de segurança, movimento, espaço e segurança de *objectos*.

## T

**índice (TOC, table of contents).** Em versões anteriores do Content Manager, é a lista de *documentos* e *pastas* contidos numa pasta ou *cesto de trabalho*. Os resultados são apresentados como um índice numa pasta.

**TCP.** Consulte *Transmission Control Protocol*.

**TCP/IP.** Consulte *Transmission Control Protocol/Internet Protocol*.

**cliente dependente.** Um cliente que tem pouco ou nenhum software instalado mas tem acesso a software gerido e fornecido por servidores de rede a ele ligados. Um cliente dependente constitui uma alternativa a um cliente com funções completas, como por exemplo, uma estação de trabalho.

**Tivoli Storage Manager (TSM).** Um produto *cliente/servidor* que fornece gestão de armazenamento e serviços de acesso a dados num ambiente heterogéneo. Suporta vários métodos de comunicação, fornece funções administrativas para gerir a cópia de segurança e armazenamento de ficheiros e funções de marcação de operações de cópia de segurança.

**TOC.** Consulte *índice*.

**Transmission Control Protocol (TCP).** Um *protocolo* de comunicações utilizado na *Internet* e em quaisquer redes que seguem as normas da Internet Engineering Task Force (IETF) para protocolo inter-redes. O TCP fornece um protocolo fidedigno de sistema central para sistema central entre sistemas centrais em redes de comunicações comutadas de pacote e nos sistemas interligados de tais redes. Usa o *Protocolo Internet (IP, Internet Protocol)* como protocolo subjacente.

**Transmission Control Protocol/Internet Protocol (TCP/IP).** A série de *protocolos* e transporte e aplicação que são executados em Protocolo Internet.

**TSM.** Consulte *TivoliStorage Manager*.

**Volume de TSM.** Uma área lógica de armazenamento que é gerida pelo *Tivoli Storage Manager*.



## U

**uniform resource locator (URL).** Uma sequência de caracteres que representam os recursos de informação num computador ou numa rede, como a Internet. Esta sequência de caracteres inclui o nome abreviado do protocolo utilizado para aceder aos recursos de informação e a informações utilizadas pelo protocolo para localizar o recurso de informação. Por exemplo, no contexto da Internet, estes são nomes abreviados de alguns protocolos utilizados para aceder a vários recursos de informação: http, ftp, gopher, telnet e news.

**utilizador.** Um indivíduo que necessita dos serviços do DB2 Content Manager. Este termo refere-se geralmente a utilizadores de aplicações cliente, e não tanto aos programadores das aplicações que usam as APIs do DB2 Content Manager. No DB2 Information Integrator for Content, quem quer que esteja identificado no programa de administração do DB2 Information Integrator for Content.

**ACL de utilizador.** Qualquer *lista de controlo de acessos* (ACL, *access control list*) criada por um *utilizador* final com o *privilégio* UserACLOwner e que só possa ser atribuída a *artigos*. Os utilizadores podem pesquisar em ACLs de utilizador. As ACLs de utilizador não são apresentadas no cliente de administração do sistema. Um utilizador que se encontre listado na ACL de utilizador e que tenha privilégio UserACLOwner, ou um administrador, pode modificar uma ACL de utilizador mediante as APIs.

**saída de utilizador.** Um ponto num programa fornecido pela IBM no qual pode ser dado controlo a uma rotina de saída de utilizador.

**rotina de saída de utilizador.** Uma rotina escrita pelo utilizador que recebe controlo em *saídas de utilizador* predefinidas.

**grupo de utilizadores.** Um grupo constituído por um ou mais *utilizadores* individuais, identificados por um nome de grupo único.

**correlação de utilizador.** A associação de IDs de utilizador e palavras-passe do DB2 Information Integrator for Content a IDs de utilizador e palavras-passe correspondentes em um ou mais servidores de conteúdos. A correlação de utilizadores permite um início de sessão único no DB2 Information Integrator for Content e em vários *servidores de conteúdos*.

**servidor de utilitários.** Um componente do DB2 Content Manager utilizado pelos utilitários da base de dados para fins de marcação. Pode configurar um servidor de utilitários ao configurar um *gestor de recursos* ou *servidor de bibliotecas*. Existe apenas um servidor de utilitários para cada gestor de recursos e servidor de bibliotecas.

## V

**nó virtual.** No encaminhamento de documentos do DB2 Content Manager, um ponto distinto no diagrama do *processo* no qual não é executado qualquer trabalho nem tomada nenhuma decisão, mas que é necessário para transmitir efectivamente o fluxo de processo. Iniciar, parar, dividir e juntar são nós virtuais.

**volume.** Uma representação de uma unidade ou dispositivo de armazenamento físico real no qual estão armazenados os objectos existentes no sistema.

## W

**carácter global.** Um carácter especial, tal como asterisco (\*) ou ponto de interrogação (?), que pode ser utilizado para representar um ou mais caracteres. Qualquer carácter ou conjunto de caracteres pode substituir um carácter global.

**cesto de trabalho.** (1) No encaminhamento de documentos do DB2 Content Manager, representa uma localização na qual o trabalho aguarda a acção a realizar por um utilizador ou por uma aplicação. A acção pode ser realizada no trabalho que aguarda no cesto de trabalho ou a acção poderá encaminhar o trabalho para outro *nó de trabalho*. (2) No fluxo de trabalho do Content Manager Versão 7, é sinónimo de *lista de trabalhos*.

**fluxo de trabalho.** (1) No DB2 Content Manager anterior, uma sequência de *cestos de trabalho* através dos quais um *documento* ou *pasta* passam enquanto estão a ser processados. (2) No DB2 Information Integrator for Content, uma sequência de *passos de trabalho*, e as regras que regem esses passos, através da qual o trabalho é encaminhado. Um processo de fluxo de trabalho contém, pelo menos, um nó de início, um *nó de trabalho* e um nó de paragem.

**coordenador de fluxo de trabalho.** No fluxo de trabalho do DB2 Content Manager anterior, um utilizador que recebe notificação de que um *artigo de trabalho* no *fluxo de trabalho* não foi processado num tempo especificado. O utilizador é seleccionado para um *grupo de utilizadores* específico ou após a criação do fluxo de trabalho.

**estado do fluxo de trabalho.** O estado de um *fluxo de trabalho* completo.

**artigo de trabalho.** No fluxo de trabalho do DB2 Content Manager anterior e no fluxo de trabalho avançado do DB2 Information Integrator for Content anterior, um documento ou objecto necessário ao utilizador para concluir uma actividade de *fluxo de trabalho*.

**nó de trabalho.** (1) No encaminhamento de documentos do DB2 Content Manager, um passo no

âmbito de um processo no qual os artigos aguardam que sejam tomadas acções pelos utilizadores finais ou pelas aplicações, ou através do qual os artigos se movem automaticamente. Termo genérico para um dos seguintes três tipos de nós de trabalho: cesto de trabalho, ponto de recolha e aplicação empresarial. (2) No *fluxo de trabalho* avançado do DB2 Information Integrator for Content, um passo no âmbito de um fluxo de trabalho onde o trabalho é realizado por utilizadores ou grupos especificados.

**lista de trabalhos.** (1) No encaminhamento de documentos do DB2 Content Manager, um filtro de *pacotes de trabalho* disponíveis que são atribuídos a um utilizador ou grupo. (2) No fluxo de trabalho avançado do DB2 Information Integrator for Content, um filtro de *artigos de trabalho* disponíveis atribuídos a um utilizador ou grupo.

**pacote de trabalhos.** No encaminhamento de documentos do DB2 Content Manager, um objecto definido pelo sistema que referencia o artigo no qual o utilizador trabalha durante um processo. Para além do ID do artigo, o pacote de trabalhos contém informações adicionais que identificam o processo ao qual pertence bem como a respectiva prioridade, o estado e o tempo para retomar (caso esteja suspenso). O utilizador ignora a existência do pacote de trabalhos pois o utilizador trabalha no artigo referenciado e não no próprio pacote.

**conjunto de trabalhos.** Non Enterprise Information Portal Versão 7.1, uma recolha de *documentos* que é encaminhada de um local para outro. Os utilizadores acedem e trabalham com conjuntos de trabalhos através de *listas de trabalhos*.

**passo de trabalho.** (1) Um ponto discreto num processo de encaminhamento de documentos do DB2 Content Manager através do qual deve passar um *pacote de trabalhos* individual. (2) Um ponto discreto num *fluxo de trabalho* do DB2 Information Integrator for Content, através do qual deve passar um *artigo de trabalho* individual.

## X

**XDO.** Consulte *objecto de dados expandido*.

**XML.** Consulte *Extensible Markup Language*.



---

# Índice Remissivo

## A

acções  
  aceder a listas 295  
  criar 295  
  DKWorkflowActionICM 227  
  enumerar em XML 413  
  fluxo de trabalho 224  
  predeterminado 413  
ACL 29  
ActionListName 405  
actualizar o conteúdo  
  XDO 49  
AddItemToFolderRequest 389, 432  
administração  
  objecto  
    obter 169  
    privilégios 73  
agrupamento de ligações 441  
  servlet 509  
  valores de servlet 510  
AIX  
  iniciar o servidor de RMI 21  
  objectos partilhados para C++ 21  
AllPrivSet 73, 76  
ambiente, definir  
  C++ 21  
  Java 19  
âmbito  
  biblioteca de identificadores 504  
  transacção 176  
âmbito da anotação  
  GLOBAL 345  
  LOCAL 345  
  VIEW 345  
anexos  
  MIME 458  
anotações  
  adicionar um objecto a um XDO 48  
  beans 312  
  classes de serviços 469  
  constante 97  
  documentos 467  
  motor 471  
  OnDemand 271  
  personalizar 487  
  serviços 483  
  suporte de edição 485  
  tipo semântico 97  
API de C++ 28, 29, 163  
API de Java 29, 163  
ApplicationName 413  
armazenamento de conteúdo 163  
arquivo de dados, 5  
artigo com saída dada  
  consultar 183  
  consultar exemplo 200  
  consultar por marca de hora 200  
  consultar por pessoa 200  
artigos  
  adicionar a uma pasta 121  
  beans 312, 504, 507  
  artigos (*continuação*)  
    constante em árvore 124  
    Content Manager Versão 8 66  
    criação de versões 72, 118  
    criar um tipo de artigo  
      documento 169  
    dar entrada e saída 115  
    definir e obter atributos no CM 8 102  
    efectuar referências 87  
    estabelecer ligações através de  
      pastas 71  
    federados 9  
    gerir com o servlet controlador 513  
    importar e exportar instâncias como  
      XML 365  
    ligar 127  
    modificar atributos 106  
    obter artigos ligados 129  
    obter no CM 8 110  
    pesquisar no CM 8 109  
    remover de pastas 122  
  artigos de dados, DDO 39  
  artigos de recurso  
    constante 89  
    Content Manager Versão 8 67  
    definir um tipo de artigo para 107  
    representar em serviços da Web 458  
    tipo de artigo 89  
  artigos de trabalho  
    aceder 293  
  artigos que não são de recurso  
    Content Manager Versão 8 67  
  ASCENDING 219  
  atributo de versão 381  
  atributos  
    aceder num DDO 39  
    actualizar grupos no CM 8 94  
    agrupamento 67  
    apresentar em beans 324, 333  
    beans 312, 505, 507  
    comparar com DDOs e XDOs 8  
    consultar 200  
    Content Manager Versão 8 67  
    criação de versões 110  
    criar no CM 8 93  
    de vários valores 67  
    definição 67  
    definidos pelo utilizador 183  
    tornar pesquisáveis por texto 187  
  definir 259  
  definir e obter no CM 8 102  
  eliminar para DKAny 61  
  enumerar em IP OS/390 273  
  enumerar no CM for AS/400 279  
  enumerar para OnDemand 262  
  enumerar para um tipo de artigo 96  
  gerir grupos no CM 8 94  
  gerir no servlet controlador 513  
  modificar para um artigo 106  
  obter para um tipo de artigo  
    documento 169

atributos (*continuação*)  
  obter propriedades de 41  
  referência 183  
  representar através de DDOs 87  
atributos de referência 183  
atributos definidos pelo utilizador 183  
  tornar pesquisáveis por texto 187  
ATTRONLY, ImagePlus for OS/390 277  
AuditComment 413  
autenticação  
  serviços da Web 455  
autenticação, XML 376  
AuthenticationData 376  
avaliar a pesquisa 30  
avaliar consultas 63

## B

base de dados de administração  
  correlação de esquemas 5  
bases de dados relacionais  
  correlacionar a partir de federado 9  
BasicExpression 219  
BatchRequest 421, 432  
bean CMBConnectionPool 318  
bean CMBXMLMessage 373  
bean de ligação 505  
bean searchTemplate 505  
BeanInfo 312  
beans 312  
  anotação 312  
  anotações 312  
  artigos 504  
  auxiliar 312  
  auxiliares 312  
  características  
    análise introspectiva 307  
    eventos 307  
    métodos 307  
    persistência 307  
    personalização 307  
    propriedades 307  
  classes de BeanInfo 312  
  classes de eventos e de receptor 312  
  classes de excepção 312  
  CMBSchemaManagement 310  
  compreender conceitos básicos 307  
  conjunto de ligações  
    CMBConnectionPool 504  
  construir WebSphere Studio  
    Application Developer 309  
  criação de versões 324, 334  
  critérios de pesquisa 505  
  documentos 508  
  esquema 310  
  esquema de mensagem 373  
  ficheiros JAR 309  
  fluxo de trabalho 310, 312  
  gestão de dados 310, 504  
  gestão de esquemas 504  
  gestão de utilizadores 504

- beans (*continuação*)
  - introdução 307
  - invocar 310
  - ligações 310, 504, 505
  - lista de trabalhos 310
  - locale, alterar 322
  - Model View Controller (MVC) 311
  - modelo de pesquisa 504, 505
  - não visuais
    - categorias 312
    - configurações 311
    - considerações 317
    - construir aplicações 323
    - definição 307
    - eventos 323
    - funções 311
    - introdução 310
    - propriedades 323
  - origem de dados
    - bean de origem de dados 505
  - outros construtores 308
  - pastas 508
  - pedido de pesquisa 310
  - pesquisar OnDemand 270
  - rastrear 322
  - receptores de sessão 312
  - registo de rastreio 504
  - relacionado com Java Viewer
    - definição 307
  - requisitos 308
  - resultados da pesquisa 504
  - serviço de consulta 504
  - serviços de documento 312, 504
  - serviços XML 373
  - singletons 317
  - suporte de lotes 310
  - utilizar em construtores 308
  - visuais
    - área de janela em árvore 328
    - áreas de pesquisa de texto 326
    - comportamentos
      - especializados 336
    - comportamentos gerais 334
    - construir aplicações 336
    - definição 307
    - editor de atributos 333
    - especificações do
      - visualizador 332
    - eventos chave 336
    - eventos de ajuda 335
    - força de ordenação 334
    - guardar/repôr configuração 335
    - introdução 324
    - ligação 336
    - ligações 334
    - lista de modelos de pesquisa 326
    - menus emergentes 328
    - nomes 324
    - ocultar e apresentar botões 334
    - painel de início de sessão 324
    - painel de pesquisa 327
    - substituir 336
    - substituir menus emergentes 329
    - utilizar em janelas 338
    - visualizador de documentos 331
    - visualizador de modelos de
      - pesquisa 326

- beans (*continuação*)
  - visuais (*continuação*)
    - visualizador de pastas 330
    - visualizador de resultados da
      - pesquisa 328
    - visualizador de versões 334
    - visualizadores externos 333
    - visualizadores predefinidos 333
  - web.xml 309
  - XML 312
- beans auxiliares 312
- Beans não visuais 307
- beans XML 428
- BETWEEN 207, 219
- biblioteca
  - Java 19, 21
  - listagem de C++ 21
- biblioteca de identificadores
  - relacionados com artigos 507
  - relacionados com documentos 508
  - relacionados com esquemas 505
  - relacionados com ligações 505
  - relacionados com pastas 508
  - relacionados com pesquisa 506
- BLOB, 44
- busca prévia de objectos 253
- Business Process Choreographer 442

## C

- C++
  - bibliotecas 21
  - configurar o ambiente 21
  - constantes 32
  - DKAny 56
  - DKConstant.h 17
  - ficheiros DLL 21
  - objectos partilhados para AIX 21
  - sequências de escape 211
  - suporte de XML 17
- campo de parâmetros adicionais,
  - OnDemand 260
- caracteres globais
  - consulta 208
  - pesquisa de texto 187
- Catálogo de Informações
  - correlacionar a partir de federado 9
- cenário, exemplo de seguro para o
  - Content Manager Versão 8 86
- changePassword 89
- ChangePasswordRequest 377, 432
- CheckinItemRequest 400
- CheckInItemRequest 432
- CheckoutItemRequest 400
- CheckOutItemRequest 432
- classes de artigo, 67
- classes de definição de dados 259
- classes proxy 448, 449
- Cliente .NET 1.1 448
- Cliente .NET 2.0 449
- cliente de administração de sistemas
  - criar listas de trabalho 292
  - personalizar 15
  - saídas do utilizador 15
- cmadmin.xsd 339
- cmb81.jar 308
- CMBAnnotationPropertiesInterface 487
- CMBAttribute
  - alterar locale em nomes de
    - apresentação 322
- cmbcc2mime.ini 510
- cmbclient.ini 510
- cmbcm81.lib 21
- cmbcm81d.lib 21
- CMBConnection 270, 310, 504
- cmbcs.ini 311, 510
- CMBDataManagement 310, 504
- CMBDocumentServices 473, 504
- CMBDocumentViewer 324, 331
  - especificações 332
  - terminar 331
- CMBEntity
  - alterar locale em nomes de
    - apresentação 322
- CMBFolderViewer 324, 330
- CMBGenericWebService 451
  - integrar 451
- CMBItem 504
- CMBItemAttributesEditor 333
- CMBLoginPanel 324
- cmbmessages.xsd 373
- CMBObject 507
- CMBPage 508
- CMBPageAnnotation 485
- CMBQueryService 310, 504
- cmbregist81.bat 21
- cmbregist81.sh 21
- CMBSchemaManagement 504
- CMBSearchPanel 327
- CMBSearchResults 504
- CMBSearchResultsView 270
- CMBSearchResultsViewer 324, 328
- CMBSearchTemplate 504
- CMBSearchTemplateList 270, 324, 326
- CMBSearchTemplateViewer 270, 324, 326
- cmbServlet.properties 509
- CMBServletAction 509
- cmbServletjsp.properties 509
- CMBSpecificWebService.cs 447
- CMBSTCriterion 505
- cmbsvclient.ini 510
- cmbsvcs.ini 510
- CMBTraceLog 504
- CMBUserManagement 504
- cmbview81.jar 475
- CMBViewerConfiguration.properties 475
- CMBXMLAttachment 312
- CMBXMLMessage 312
- cmbxmbservice.jar 339
- CMBXMLServices 312
- cmdatamodel.xsd 339
- CMWebService
  - integrar 443
- CMWebServiceClient.cs 450
- CMWebServiceClient.readme 464
- CMWebServiceClientWSE30.readme 464
- COBRA
  - Serviço de Dados Persistentes
    - (PDS) 6
  - Serviço de Objecto Persistente 6
- CollectionResumeListEntry 405
- colocação em memória cache, servlet
  - controlador 510

- colocar em lotes os pedidos XML 421
- Coluna DOCROUTINGFREQ
  - alterar 227
- com.ibm.mm.sdk.client 18
- com.ibm.mm.sdk.server 18
- CompareOperator 219
- Comparison 219
- compilador Microsoft Visual C++, 21
- compilador visual de C++ 21
- componentes descendentes
  - Content Manager Versão 8 67
  - criação de versões 72
  - criar no CM 8 97
  - diagrama 67
  - efectuar referências 69
  - representação no modelo de dados 183
  - representar como atributos de DDO 87
- componentes raiz
  - Content Manager Versão 8 67
  - criação de versões 72
  - criar no CM 8 97
  - efectuar referências 69
  - ligar 69
  - representação no modelo de dados 183
- conector de DB2 4
- conector de DD (Domino Doc 4
- conector de DL (Content Manager versão 7) 4
- conector de Fed 4
- conector de JCC 83
- conector de JDBC 4
- conector de ODBC 4
- conector de QBIC (Image Search) 4
- conector de TS (Text Search) 4
- conector ICM, 65
- conectores
  - valores de servlet controlador 510
- configString 376
- configuração de ACLs 80
- configuração de ACLs em tempo de execução 80
- configuração dinâmica, beans 311
- configurar o JDBC 439
- conjunto de ligações 318
- connectString 376
- connectToWorkflow 376
- constantes 32
  - encaminhamento 251
- construir aplicação
  - beans não visuais 323
  - beans visual 336
  - criar no Content Manager Versão 8 87
  - planear 81
  - utilizar serviços de anotação 486
- Consulta
  - Java 63
    - dkResultSetCursor vs DKResults 64
    - tipo paramétrico 64
    - tipos de objectos de consulta 63
- consulta aritmética
  - operações 200
  - sintaxe 219
- consulta combinada
  - definição 30
- Consulta combinada
  - Java 194
- consultas
  - artigos XML 381
  - avaliar 63
  - CM para AS/400 281
  - compreender a linguagem 181
  - Content Manager Versão 8 182
  - enumerar artigos com saída dada 200
  - enumerar os resultados 200
  - executar 63
  - exemplo de caracteres globais 200
  - exemplos 196, 200
  - exemplos de encaminhamento 247
  - federadas
    - processar 11
  - gramática da linguagem 219
  - linguagem 207
  - modelo de dados 196
  - operações aritméticas 200
  - pesquisa de texto 187
  - pesquisa OnDemand 265
  - recolha federada 62
  - sequências de escape 207
  - sintaxe 200
  - sintaxe de ImagePlus 277
  - versão 200
- ContainerData 410
- ContainerDefinition 405
- contains-text-db2
  - exemplo de consulta 200
- contém pesquisa de texto básica 187
- contém texto, pesquisa de texto 188
- CONTENT
  - ImagePlus for OS/390 277
- Content Manager, versões anteriores
  - correlacionar a partir de federado 9
- Content Manager para AS/400
  - classes de índices 279
  - correlacionar a partir de federado 9
  - enumerar entidades e atributos 279
  - execução da consulta 281
  - introdução 279
- Content Manager Versão 8
  - actualizar documentos 173
  - artigos 66
  - artigos de recurso 67
  - artigos que não são de recurso 67
  - atributos 67
  - coerência de servidor de bibliotecas e do gestor de recursos 176
  - componentes descendentes 67
  - componentes raiz 67
  - compreender a consulta de pesquisa 181
  - compreender a linguagem de consulta 155
  - conceitos básicos 66
  - conjunto de privilégios 73
  - consultar 182
  - controlar o acesso
    - grupos de utilizadores 76
    - listas de acesso 76
  - controlar privilégios de acesso 73
- Content Manager Versão 8 (*continuação*)
  - controlo de acessos 73
  - correlacionar a partir de federado 9
  - criação de versões 72
  - criar atributos 93
  - criar tipos de artigo 89
  - criar um documento 171
  - criar um servidor de conteúdos 88
  - criar uma aplicação 87
  - criar uma palavra-passe 89
  - dar entrada e saída de artigos 115
  - definir e obter atributos de artigo 102
  - definir um tipo de artigo de recurso 107
  - documentos 67, 71
  - eliminar um documento 174
  - encaminhar documentos 223
  - enumerar atributos para um tipo de artigo 96
  - enumerar tipos de artigo 91
  - exemplo de cenário de seguro 86
  - exemplos 86
  - exemplos de consulta 196
  - gerir documentos 168
  - gerir grupos de atributos 94
  - gestor de recursos 65
  - ilustração de componentes 65
  - introdução 65
  - ligação a 88
  - ligações 69
  - ligar artigos 127
  - modificar atributos de artigo 106
  - objectos 68
  - obter artigos 110
  - obter um documento 174
  - partes de documento 67
  - pastas 71
  - pesquisar artigos 109
  - planear uma aplicação 81
  - referências 69
  - servidor de bibliotecas 65
  - Servidor de System Managed Storage (SMS) 65
  - tipos de artigo 67
  - trabalhar com controlo de acessos 130
  - trabalhar com o gestor de recursos 156
  - trabalhar com pastas 120
  - transacções 176
- contentOption 381
- contentores
  - constante 97
  - Content Manager Versão 8 69
  - tipo semântico 97
- conteúdo de multimédia 7
- conteúdo do recurso 7
- ContinueProcessRequest 418, 432
- Controlador IBM® DB2® para JDBC e SQLJ 83
- controlo de acessos 152
  - atribuir uma lista a um artigo 136
  - Content Manager Versão 8 73
  - definir listas (ACLs) 133
  - diagrama 73
  - listas (ACL) 73, 76



- controle de acessos (*continuação*)
  - NoAccessACL 76
  - obter e apresentar listas (ACLs) 134
  - PublicReadACL 76
  - regras para listas (ACLs) 76
  - SuperUserACL 76
  - trabalhar com 130
  - trabalhar com listas para encaminhamento de documentos 249
- conversão de página de códigos
  - definir o subsistema da consola em Windows 23
- conversões 487
- converter, esquemas 370
- correlação de esquemas
  - associar em federada 9
  - introdução 5
- createChildDDO 97
- createDDO 38
- CreateItemRequest 379, 432
- CreateLinkRequest 401, 432
- criação de versões
  - artigos 110
  - atributos 110
  - beans 324, 334
  - Content Manager Versão 8 72
  - controlada pela aplicação 72
  - definir para um artigo 118
  - identificador persistente (PID) 72
  - partes no modelo de dados de gestão de documentos 175
  - política 72
  - políticas de controlo de versões 118
  - tipos de artigo 118
- criar 97
- critério, pesquisa, bean 505

## D

- dados binários, 7
- dar entrada 392
- dar saída 392
- data\_id 41
- databaseNameStr 88
- DB2
  - assistente de configuração cliente 21
  - suporte cliente 21
- DB2 Content Manager V8 4
- DB2 Extenders 7
- ddoDocument 48
- definição de tabela 255
- definições
  - ambiente C++ 21
  - ambiente Java 19
- Definições
  - C++
    - Construir em NT 23
  - Java
    - Em NT 19, 20, 22, 23
    - Ligação de desligação de cliente 34
    - Sugestões de programação 18
  - Utilizar exemplos de applets e servlet de Java 503
    - Acesso local 504
    - Acesso remoto 505

- Definições (*continuação*)
  - Utilizar exemplos de applets e servlet de Java (*continuação*)
    - aplicação Java em cliente 503
    - servlet de Obtenção 503
  - deleteDKAttributeDef 61
  - DeleteItemRequest 399, 432
  - DeleteLinkRequest 401, 432
  - deleteSchemaMapping 370
  - DemoSimpleAppl.java 323
  - DESCENDING 219
  - desinstalar 442
  - DfltACLCode 76
  - DIGIT 219
  - DisplayName 413
  - displayNamesEnabled 322
  - DIV 219
  - DK\_CM\_OPT\_ACCESS\_MODE 259
  - DK\_CM\_READWRITE 259
  - DK\_CM\_VERSION\_LATEST 110
  - dkAbstractWorkFlowUserExit 295
  - DKAny
    - atribuição a 57
    - atribuição de 57
    - código de tipo, obter 57
    - destruir 58
    - ecrã de 58
    - eliminar recolhas 61
    - gestão de memória 57
    - sugestões de programação 59
    - Typecode 56
    - utilizar construtores do tipo 57
    - utilizar dentro de uma recolha 59
    - verificar o tipo 57
  - dkAttrDef 259
  - DKAttrGroupDefICM 94
  - dkBlob 259
  - DKCollectionResumeListEntryICM 227
  - DKConstant 32
  - DKConstant.h 17
  - DKDatastore 259
  - dkDatastoreDef 259
  - DKDatastoreDL
    - Java 32
      - enumerar esquemas e atributos de esquemas 35
      - Enumerar servidores 34
      - ligação 33
      - opções de DKDatastoreDL 34
  - DKDatastoreICM 87
  - DKDatastoreIP 272
  - DKDatastoreOD 260
    - listEntities 262
  - DKDatastoreV4 279
  - DKDatastorexx 259
  - DKDatstoreIP
    - listEntityAttrs 273
  - DKDatstoreOD
    - listSearchTemplates 264
  - DKDDO, 37
  - DKDocRoutingServiceICM 227
  - DKDocRoutingServiceMgmtICM 227
  - DKDocumentConverter 370
  - dkEntityDef 259
  - DkEntityDefIP
    - getAttr 273

- DkEntityDefIP
  - listAttrNames 273
- DKException 530
- DKException 31, 81
- DKFederatedCollection 62
- dkFederatedIterator 62
- dkFederatedQuery 62
- DKFederatedQuery 11
- DKFixedView 260
- DKFixedViewDataOD 260
- dkIterator 62
- DKLink 127
- DKLinkCollection 128
- DKPrivilegeSetICM 131
- DKProcessICM 227
- DKResource 260
- DKResourceGrpOD 260
- DKResults
  - posicionar o iterador 62
- dkResultSetCursor 259
- DKResumeListEntryICM 227
- DKRMConfiguration 88
- DKRMConstantSMS 164
- DKRMControlServiceSMS 164
- DKRMServerDefinitionDefSMS 164
- DKRMVolumeDefSMS 164
- DKRouteListEntryICM 227
- DKSchemaConverter 370
- dkServerDef 259
- DKTimestamp
  - suspender um fluxo de trabalho 290
- DkViewOD 260
- DKViews 260
- DKWorkflowActionICM 227
- DKWorkflowFed
  - retomar 291
  - suspender 290
- DKWorkflowServiceFed
  - svWf 290
- DKWorkflowServicesFed 287
- dkWorkFlowUserExit 295
- DKWorkItemFed
  - dar entrada 294
  - dar saída 294
- DKWorkListFed 292
- DKWorkListICM 227, 238
- DKWorkNodeContainerDefICM 227
- DKWorkNodeICM 227, 233
- DKWorkPackageICM 227, 245
- DKXMLExportList 342
- DKXMLSysAdminService
  - constantes 342
  - métodos 342
- DLLName 413
- do Action 295
- Documento e codificado 429
- Documento e literal 429
- documentos 97
  - atualizar no CM 8 173
  - anotar 467
  - apresentar em beans 324, 331
  - beans 312, 504, 508
  - colocação em memória cache no servlet controlador 510
  - Content Manager Versão 8 67, 71
  - criar no CM 8 171

- documentos (*continuação*)
  - criar o modelo de dados de gestão no CM 8 169
  - criar um tipo de artigo
    - documento 169
  - criar versões de partes no modelo de dados de gestão 175
  - eliminar no CM 8
    - SDocModelItemICM 174
  - encaminhar através de um processo 223
  - gerir no CM 8 168
  - iniciar um processo de encaminhamento 240
  - obter no CM 8 174
  - pesquisa de texto de 187
  - representar no modelo de dados 183
  - tipo semântico 97
  - visualizar, 467
- dsType 270

## E

- encaminhamento 312
  - acção do fluxo de trabalho 224
  - ad hoc 246
  - atribuir privilégios 248
  - beans 312
  - compatibilidade com a Versão 8.2 227
  - configurar 227
  - constantes 251
  - construtor gráfico de processos 224
  - continuar um processo 242
  - criar objectos de serviço 231
  - definir um novo nó de trabalho regular 232
  - definir um novo ponto de recolha 234
  - definir um novo processo e encaminhamento associado 239
  - definir uma lista de trabalho 236
  - divisão 224
  - enumerar listas de trabalho 238
  - enumerar nós de trabalho 233
  - enumerar PIDs de pacotes numa lista de trabalhos 244
  - enumerar processos de encaminhamento de documentos 245
  - exemplos 247
  - iniciar 240
  - iniciar em XML 416
  - introdução 223
  - junção 224
  - lista de continuação 227
  - lista de trabalhos 227
  - listas de controlo de acessos (ACLs) 249
  - nós de trabalho 223
  - obter informações relativas a pacotes de trabalho 245
  - pacote de trabalho 227
  - paralelo 224
  - ponto de decisão 224
  - pontos de recolha 223
  - retomar um processo 243
- encaminhamento (*continuação*)
  - subprocessos 224
  - suspender um processo 242
  - terminar um processo 241
  - variáveis de contentor 224
  - XML 404
- encaminhamento ad hoc 246
- encaminhamento de documentos 140
- encaminhamento de documentos, 223
- encaminhamento paralelo 224
- encaminhamentos de divisão 224
- encaminhamentos de junção 224
- Enterprise Information Portal
  - bases de dados 5
  - correlação de esquemas 5
  - identificador persistente (PID) 8
  - motores de documentos 470
  - objectos de dados dinâmicos (DDO) 6
  - objectos de dados expandidos (XDOs) 7
  - servidores de conteúdos suportados 5
- Enterprise Java Beans (EJBs) 310
- EnterUserDLL 405
- EnterUserFunction 405
- entidades
  - armazenar pastas federadas em 15
  - beans 312, 505
  - controladas, Content Manager Versão 8 73
  - correlacionar em federado 9
  - enumerar em IP OS/390 273
  - enumerar em XML 378
  - enumerar no CM 8 91
  - enumerar no CM for AS/400 279
  - hierarquia de classes 259
  - identificar 8
  - ImagePlus for OS/390 277
  - mover artigos através de XML 403
  - nativa 270
  - pastas OnDemand como nativas 270
  - privilégios 73
- entidades controladas 73
- ENTITY\_TYPE 260, 269
- EntityList 378
- ErrorCode 81
- ErrorState 81
- ESCAPE,KEYWORD, consulta 219
- ESCAPE\_LITERAL 219
- esquemas
  - beans 504
  - cmadmin.xsd 339
  - cmdatamodel.xsd 339
  - ferramenta de conversão 370
  - importar e exportar esquemas de armazenamento 345
  - tipos de XML não suportados em esquemas de armazenamento 364
- esquemas de armazenamento
  - importar e exportar 345
  - tipos de XML não suportados 364
- eventos
  - beans 312
- excepções, tratar 31
- EXCEPT 196, 219
- executar a pesquisa 30

- executar a pesquisa com repetição de marcação 30, 182
- executar consultas 63
- exemplo de políticas 87
- exemplos C# 442
- expiração de tempo 28
- exponente, consulta 219
- exportar XML 342
- expressão, consulta 219
- ExpressionList 219
- ExpressionWithOptionalSortBy 219
- expressões, consulta 200
- Extended Search
  - correlacionar a partir de federado 9
- Extenders 7
- extrair XML
  - exportar dependências de objectos 369
  - formatos de entrada e de saída 342
  - instâncias de dados 365, 366
  - metadados de administração 343
  - metadados de modelo de dados 345

## F

- federada
  - beans 505
  - consulta
    - criar 11
    - ilustração de processamento 11
  - pesquisar
    - ilustração relacional 11
    - introdução 9
    - processo 11
  - recolha 11
- federadas 15
  - consulta
    - exemplos 13
    - sintaxe 13
  - funções de administração de sistemas 15
  - iteradores 62
  - pastas 15
  - pesquisa de imagens 13
  - recolhas 62
- federado
  - atributos 9
  - correlação de esquemas 10
  - correlacionar 9
  - correlacionar de IDs de utilizador 11
  - entidades 9
  - nível para beans 311
  - pesquisar
    - ilustração 9
  - registar servidores de conteúdos 11
- federados
  - artigos 9
  - consulta
    - resultados 11
  - modelo de documentos 9
- ferramentas
  - Content Manager Versão 8 86
- ficheiro, adicionar um XDO de 48
- ficheiro de configuração de registo APIs 520
  - definir valores 520
  - modificar 520





- ligações (*continuação*)
  - de chegada e partida 128
  - definir entre artigos 127
  - destino 127
  - nomes de tipos 128
  - obter artigos ligados 129
  - origem 127
  - percorrer em consulta 200
  - representação no modelo de dados 183
  - tipo de atributo 87
- ligações de chegada 128
- ligações de partida 124, 128
- LIKE 208, 219
- linguagem de consulta 207
- linguagem de marcação expansível 17
- lista de continuação 227
- lista de predicados 219
- ListActionNamesRequest 413, 432
- ListActionRequest 432
- listas de trabalho
  - aceder 292
  - aceder a artigos de trabalho 293
  - beans 310
  - definir 236
  - DKWorkListICM 227
  - enumerar 238, 291
  - enumerar em XML 409
  - enumerar PIDs de pacotes 244
  - mover artigos 294
- ListConstructor 219
- ListContent 219
- listDataSources 88
- listEntityNames 91
- ListNextWorkPackageRequest 432
- ListNextWorkPackagesRequest 410
- ListProcessNamesRequest 408, 432
- ListProcessRequest 408, 432
- listResourceMgrs 88
- ListSchemaRequest 378, 432
- ListServerRequest 375
- ListServersRequest 432
- listWorkFlowTemplates 294
- ListWorkListNamesRequest 409, 432
- listWorkLists 291
- ListWorkListsRequest 409, 432
- ListWorkNodeRequest 432
- ListWorkNodesRequest 405, 432
- ListWorkPackagesRequest 410
- literais
  - ImagePlus for OS/390 277
- literais, consulta 200
- literal, consulta 219
- LOCAL 345
- locale
  - beans, alterações em 322
  - servlet 509
- localização especificada pelo utilizador 163
- LocalPart 219
- LogicalOrSetExpression 219
- LogicalOrSetPrimitive 219
- LoginData 376
- LorgicalOrSetTerm 219

## M

- manipulação de páginas
  - interfaces 490
- maxResults 381
- melhoramento de API 164
- memória tampão, adicionar um XDO 47
- mensagens
  - Informações de DKException 81
- mensagens XML
  - actualizar artigos 392
  - alterar uma palavra-passe 377
  - autenticar 376
  - colocar em lotes 421
  - continuar um processo 418
  - criar um artigo 379
  - dar saída e entrada de artigos 400
  - eliminar artigos 399
  - enumerar acções 413
  - enumerar entidades 378
  - enumerar listas de trabalho 409
  - enumerar nós de trabalho 405
  - enumerar pacotes de trabalhos 410
  - enumerar processos 408
  - enumerar servidores 375
  - iniciar um processo 416
  - ligar artigos 401
  - mover artigos entre entidades 403
  - obter artigos 383
  - pesquisar artigos 381
  - relação com outros serviços da Web 427
  - ResumeListDefinition 419
  - retomar um processo 420
  - SOAP 425
  - suspender um processo 419
  - terminar um processo 417
  - trabalhar com pastas 389
  - visualizar privilégios de utilizador 388
- metadados de administração
  - importar e exportar 343
- Microsoft .NET Studio 2003 429
- Microsoft .NET Studio 2005 429
- Microsoft SOAP Toolkit 447
- Microsoft Visual Studio .NET
  - manual de iniciação de serviços da Web 447
- MOD 219
- Model View Controller (MVC) 311, 485
- modelo de dados
  - aplicar a linguagem de consulta 183
  - artigos
    - representação no modelo de dados 183
    - representar consulta em XML 196
    - representar em XML 342
  - modelo de documento, 67
  - modelo de documento federado 9
  - Modelo de Objecto de Documento (DOM)
    - construir a partir de uma mensagem XML 452
  - modelo de pesquisa, OnDemand
    - utilizar pastas como 270
    - visualizador 269
- modelos
  - fluxo de trabalho 294
- MoveItemRequest 403, 432

- MTOM (Message Translation Optimization Mechanism) 458
- Multipurpose Internet Mail Extensions (MIME) 458
- myExit (saída de utilizador de encaminhamento de documentos) 147

## N

- nameOfAttrStr 106
- NameText 219
- NewValues 403
- newVersion 392
- nó de aplicação empresarial 224
- nó de trabalho
  - definir 232
  - encaminhamento 223
- NoAccessACL 76
- NodeGenerator 219
- Nome do Tipo de Ligação 127
- nomes de apresentação
  - alterar a linguagem de 322
- nomes que não são de apresentação 322
- NONZERO 219
- NoPrivSet 73
- nós de trabalho
  - aplicação empresarial 224
  - DKWorkNodeContainerDefICM 227
  - enumerar 233
  - enumerar em XML 405
  - expandir 405
  - tipos 405
- NOT 219
- nota
  - constante 97
  - tipo semântico 97
- NotifyState 410
- NotifyTime 410
- NULL, consulta 219

## O

- Object Management Group (OMG) 6
- objecto binário grande (BLOB)
  - classes 259
  - criar XDOs para 44
- objecto de dados dinâmico (DDO)
  - aceder a atributos 39
  - aceder a conteúdos de pasta 54
  - adicionar propriedades 39
  - comparar com atributos 8
  - efectuar referências 69
  - especificação de COBRA 6
  - federados 9
  - grupos de atributos no CM 8 94
  - identificador persistente 8
  - introdução 6
  - obter propriedades de atributos 41
  - obter todas as pastas que contém 125
  - preencher com setData 97
  - relação com servidores de conteúdos 8
  - representar conteúdo de multimédia 7
  - retrieveObject 110

## Objecto de Dados Dinâmicos (DDO)

- C++
  - eliminar 43
- Java 37
  - adicionar 39
  - apresentar 43
  - atributo, DKPARTS 51, 54
  - criar 38
  - Informações, Biblioteca Digital 87
  - PID 39
  - propriedades 41
  - valores de artigos de dados 39
- objecto grande (LOB) 87
- objecto MimeMultipart
  - tipos MIME 458
- objectos
  - actualizar no CM 8 106
  - armazenar valores de atributo 102
  - beans 507
  - Content Manager Versão 8 68
  - pesquisa de texto de conteúdo 186
- objectos de dados expandidos, 7
- objectos de modelo de dados
  - importar e exportar 345
- obter
  - considerações sobre filtros baseados em linhas 212
- obter elementos 29
- OnDemand
  - activar o modo de pastas 269
  - anotações 271
  - apresentar atributos 267
  - apresentar documentos 267
  - consultar um grupo de aplicações 265
  - correlacionar a partir de federado 9
  - desligar de 261
  - enumerar grupos de aplicações 262
  - enumerar informações relativas ao servidor 262
  - enumerar pastas 264
  - introdução 260
  - ligação a 261
  - nomes de propriedades 260
  - obter documentos 264
  - pesquisar assincronamente 269
  - pesquisar um documento 265
  - rastrear 271
  - representar servidores e documentos 260
  - utilizar pastas como modelo de pesquisa 270
- opção de tempo limite da consulta 109
- opções de consulta 182
- operadores
  - consulte de ImagePlus for OS/390 277
  - pesquisa paramétrica 185
- OptionalExpressionList 219
- OptionalPredicateList 219
- OR 219
- output\_option 34
- OverLoadLimit 405
- OverloadUserDLL 405
- OverloadUserFunction 405

## P

- pacote cs 18
- pacote de trabalho
  - introdução 287
- pacotes
  - cliente de Java 18
  - cliente e servidor (cs) 18
  - hierarquia em Java 18
  - servidor de Java 18
- pacotes de trabalhos
  - actualizar em XML 415
  - DKWorkPackageICM 227
  - enumerar cadeias de PID numa lista de trabalhos 244
  - enumerar em XML 410
  - enumerar o seguinte em XML 410
  - obter informações 245
- página
  - manipulação
    - funções 490
- páginas
  - beans 508
- palavra-passe
  - alterar em beans 335
  - alterar em XML 377
  - correlacionar em federado 11
  - criar no Content Manager Versão 8 89
- paragem do INSO 189
- partes
  - criação de versões 175
- partes de artigo
  - comparar com DDOs e XDOs 8
- partes de conteúdo
  - anexar em serviços da Web 458
- partes de documento
  - constante 89
  - Content Manager Versão 8 67
  - tipo de artigo 89
- passo, consulta 219
- pastas
  - aceder 54
  - activar o modo em OnDemand 269
  - adicionar conteúdos a 121
  - apresentar em beans 324, 330
  - beans 508
  - comportamento do Content Manager 54
  - Content Manager Versão 8 71
  - criar no CM 8 120
  - enumerar em OnDemand 264
  - gerir no servlet controlador 513
  - obter conteúdos 124
  - obter todas as pastas com um DDO específico 125
  - pastas OnDemand como entidades nativas 270
  - remover conteúdo de 122
  - tipo semântico 97
  - XML 389
- pedidos de serviços da Web 432
- PENDING, ImagePlus for OS/390 277
- pEnt, C++ 35
- permissões, 130
- pesquisa de imagens
  - definição 30
  - federadas 13
- pesquisa de texto
  - beans 326
  - caracteres globais 187
  - compreender 186
  - consulta básica 187
  - contém texto 188
  - DB2 Net Search Engine (NSE) 30
  - definição 30
  - exemplo de consulta 200
  - OnDemand 260
  - operadores 187
  - pesquisa avançada 188
  - pesquisar conteúdo de objectos 186
  - pesquisar documentos 187
  - resultado básico 188
  - sintaxe da consulta 187
  - tornar os atributos definidos pelo utilizador pesquisáveis 187
- pesquisa paramétrica
  - compreender 185
  - consultar no CM para AS/400 285
  - definição 30
  - federadas 13
  - operadores 185
- pesquisar
  - beans 312, 504, 506
  - compreender a linguagem de consulta 181
  - módulos de API 87
- ponto de decisão 224
- ponto de recolha
  - definir 234
  - descrição 223
- pré-requisitos de software 435
- preferências do visualizador 494
- prioridade, encaminhamento de documentos 410
- privilégios
  - acesso a dados 73
  - ACLs pré-configuradas 76
  - apresentar propriedades de um conjunto 132
  - atribuir para encaminhar 248
  - beans 312, 507
  - códigos 73
  - conjunto 73
  - conjuntos predefinidos do Content Manager Versão 73
  - Content Manager Versão 8 73
  - criar no CM 8 130
  - criar um conjunto 131
  - definidos pelo utilizador 73
  - listas de controlo de acessos (ACLs) 76
  - tipos de conjuntos 73
  - utilizadores e grupos de utilizadores no CM 8 76
  - visualizar em XML 388
- privilégios de utilizador
  - Content Manager Versão 8 73
- PrivSetCode 73
- ProcessCompletionTime 410
- processos
  - continuar 242, 418
  - definir para um encaminhamento associado 239
  - divisão 224

processos (*continuação*)  
 DKProcessICM 227  
 encaminhamento 227  
 enumerar 245  
 enumerar em XML 408  
 iniciar em XML 417  
 junção 224  
 retomar 243  
 retomar em XML 420  
 subprocessos 224  
 suspender 242, 419  
 terminar 241  
 programar o  
 DB2 Content Manager 3  
 propriedade Mime2App 332  
 PublicReadACL 76

## Q

QName 219  
 QueryCriteria 381

## R

rastrear 441  
 beans 322, 504  
 contendor de código de retorno 81  
 estado de erro 81  
 ferramentas exemplo 81  
 JNI, servlet 510  
 OnDemand 271  
 pilha de memória 81  
 servlet controlador 510  
 tratar erros com DKException 81  
 rastreio de pilha de memória 81  
 rastreio JNI, servlet 510  
 receptores de sessão, beans 312  
 recolhas  
 eliminar 61  
 federadas 62  
 recolhas e iteradores  
 C++  
 gestão de memória 61  
 Java 59  
 iterador sequencial 59  
 recolha sequencial 59  
 REFERENCEDBY 183  
 REFERENCER 183  
 referência de programação de aplicações  
 (APR) 86  
 referências  
 Content Manager Versão 8 69  
 percorrer em consulta 200  
 tipo de atributo 87  
 registo de conectores 519  
 C++ 520  
 Java 519  
 registo de notas 507  
 remote method invocation (RMI)  
 iniciar em Java 21  
 ligar com beans 311  
 utilizar com APIs de Java 21  
 Remote Procedure Call (RPC) 425  
 removeAllElement 61  
 RemoveItemFromFolderRequest 389,  
 432

repetição de marcação, executar 30, 182  
 replaceElementAt 61  
 ReplayAction, servlet 509  
 resultado básico, pesquisa de texto 187,  
 188  
 ResumeList 410  
 ResumeProcessRequest 420, 432  
 ResumeTime 410  
 retomar um fluxo de trabalho 291  
 RetrieveFolderItemsRequest 389, 432  
 RetrieveFoldersForItemRequest 432  
 RetrieveItemRequest 383, 432  
 retrieveOption 381  
 rotinas de saída de utilizador 152  
 RouteSelected 418  
 RunQueryRequest 381, 432

## S

saída de utilizador 140  
 saídas do utilizador  
 administração do sistema 15  
 fluxo de trabalho 295  
 SampleMessageTemplate.java 452  
 SAttributeDefinitionCreationICM 67  
 SConnectDisconnectICM 81  
 SDocModellItemICM 71  
 SDocRoutingDefinitionCreationICM 231,  
 232  
 SelectionFilterOnNotify 409  
 SelectionFilterOnOwner 409  
 SelectionFilterOnSuspend 409  
 SelectionOrder 409  
 SequencedValue 219  
 sequenciação múltipla 17  
 sequências de escape, exemplos de  
 consulta 207  
 ServerType 375  
 serviço da Web CMWebService 446, 447  
 Serviço de Objecto Persistente,  
 COBRA 6  
 Serviço de Objecto Persistente (PDS),  
 COBRA 6  
 serviços da Web  
 arquitectura 427  
 autenticar 455  
 C# walkthrough sample 447  
 comparação de interfaces 429  
 configuração 439  
 criar uma nova instância 456  
 descrição geral 425  
 exemplos 464  
 exemplos de Java 442  
 implementação 427  
 iniciar, parar, verificar 438  
 instalação 436  
 integrar em aplicações 442  
 introdução 425, 426  
 Java walkthrough sample 452  
 lista de modelos Java 452  
 localização de exemplos 442  
 manual de iniciação de ambiente  
 Java 452  
 manual de iniciação do .NET 447  
 obter o ficheiro WSDL 451  
 padrões 425  
 plataformas suportadas 435

serviços da Web (*continuação*)  
 pós-instalação 439  
 Processamento de documentos 428  
 programar em .NET 450  
 relação com outros serviços XML 339  
 rendimento 463  
 segurança 455  
 situação 426  
 topologias WAS suportadas 437  
 vantagens 425  
 walkthrough sample 450  
 Serviços da Web  
 agrupamento de ligações 441  
 configurar 440  
 serviços de documentos de  
 sequenciação 469  
 serviços de fluxo de trabalho de EIP 287  
 serviços XML  
 descrição geral 339  
 diagrama 339  
 exportar dependências de  
 objects 369  
 importar e exportar metadados 342  
 programar com JavaBeans 373  
 programar níveis 339  
 situações 342  
 tipos de XML não suportados 364  
 servidor controlador  
 predefinições 510  
 servidor de bibliotecas 140  
 servidor de bibliotecas, Content Manager  
 Versão 8 65  
 criação de versões 72  
 enumerar 88  
 servidor de RMI, 21  
 servidores de conteúdos  
 aceder a informações em 259  
 beans 312  
 Content Manager para AS/400 279  
 Content Manager Versão 8.4 65  
 criar 259  
 cursor do conjunto de resultados 259  
 definir 259  
 federados 9  
 gerir dados 5  
 hierarquia de definição de dados 259  
 identificadores persistentes (PIDs) 8  
 ImagePlus for OS/390 272  
 nomes de classe 259  
 objecto de dados dinâmico (DDO) 6  
 objecto de dados expandido (XDO) 7  
 obter o objecto de administração 169  
 OnDemand 260  
 registar em federado 11  
 relação com DDOs 8  
 servidores suportados pelo EIP 5  
 utilizar com RMI 21  
 valores de servlet controlador 510  
 servlet, 509  
 servlet controlador  
 ficheiro de propriedades 513  
 páginas que apresentam mensagens  
 de erro 510  
 parâmetro de separador de  
 nomes 510  
 parâmetros 510, 513  
 parâmetros de conversão 510



- servlet controlador (*continuação*)
  - parâmetros de pedido
    - acção 513
    - artigos 513
    - geral 513
    - gerir conteúdos 513
    - pastas 513
    - pesquisar 513
    - relacionados com
      - documentos 513
    - relacionados com ligações 513
    - reply 513
  - rastrear 510
  - tempos de execução de serviço 510
  - valores de agrupamento de
    - ligações 510
- servlet de controlo
  - acções 509
  - agrupamento de ligações 509
  - convenções 510
  - definições de JSP 509
  - expandir 509
  - ficheiro de propriedades 509
  - gestão de sessões 509
  - limpar 509
  - locale 509
  - matriz de funções de toolkit 516
  - referência 510
  - replay 509
  - utilizar 509
- sessões
  - servlet 509
- sessões expiradas, servlet 509
- setData 97
- setLocale 322
- setToFirstCollection 62
- setToLastCollection 62
- setToNextCollection 62
- setToPreviousCollection 62
- SFolderICM 71, 120, 125
- Shortcut 413
- Simple Object Access Protocol (SOAP) 425
- sinalizador de apresentação, chave remota 103
- SItemCreationICM 71
- SItemDeletionICM 115
- SItemRetrievalICM 110
- SItemTypeCreationICM 67
- SItemTypeRetrievalICM 118
- SItemUpdateICM 115
- SLinksICM 69, 127, 128
- SLinkTypeDefinitionCreationICM 128
- SOAP 429
- SORTBY 219
- SortSpec 219
- SortSpecList 219
- SOURCEITEMREF 183
- SReferenceAttrDefCreationICM 69, 183
- SResourceItemMimeTypeICM 108
- SSearchICM 109
- StartProcessRequest 416
- STRING\_LITERAL 219
- subprocessos 224
- subsistema da consola, definir em
  - Windows 23
- SuperUserACL 76

- suporte de IPv6 82
- suspender um fluxo de trabalho 290
- suspender um processo 242
- SuspendProcessRequest 419, 432
- SuspendState 410
- svWF 290
- SYSREFERENCEATTRS 183
- System Managed Storage (SMS), Content Manager Versão 8 65
- SystemAdmin 73
- SystemAdminPrivSet 73
- SystemDefineItemTypes 73

## T

- tabela ICMRMPREFETCH 253
- Tabela ICMSTYSYSCONTROL
  - configurar 227
- taglib.tld 309
- TARGETITEMREF 200
- TCallbackOD 269
- tempo de espera esgotado da
  - consulta 530
- TerminateProcessRequest 417, 432
- terminologia do nome da classe 259
- thin client 473
- TImageAnnotation 487
- TimeLastMoved 410
- TimeLimit 405
- tipo semântico
  - definição 97
  - definido pelo utilizador 97
  - tipos predeterminados 97
- tipos de artigo
  - classificações 89
  - componentes, raiz e descendentes 67
  - consultar vários 200
  - Content Manager Versão 8 67
  - criação de versões 72, 118
  - criar no Content Manger Versão 8 89
  - enumerar 91
  - enumerar atributos para 96
  - importar e exportar como esquemas XML 345
- tipos de documento 487
- tokens de LTPA 462
- toolkit de cliente com base em
  - JAX-RPC 452
- toolkit do visualizador de documentos
  - Aplicação Java 472
  - aplicações exemplo 471
  - applet ou servlet 473
  - arquitectura 469
  - criar um visualizador genérico 475
  - introdução 467
  - manipulação de páginas 490
  - modo dual e applet ou servlet 474
  - motores
    - MS-Tech Document 470
  - personalizar o visualizador genérico de documentos 475
  - serviços de anotação 483
  - thin client 473
  - visualizador autónomo 471
- Topologias do Websphere Application Server 437
- toXML() 365

- transacções, Content Manager Versão 8 176
  - considerações 177
  - dar entrada e saída 178
  - lista 178
  - precaução em relação a transacções explícitas 177
  - processar 178
- transformXMLDocument 370
- TRetrieveFolderWithCallbackOD 269
- TRetrieveWithCallbackOD 269

## U

- UNICODE\_CHARACTER 219
- UNION 196, 219
- UpdateItemRequest 392, 432
- updateObject 106
- UpdateWorkPackageRequest 432
  - mensagens XML
    - actualizar pacotes de trabalhos 415
- UserLastMoved 410
- utilizadores
  - controlar acesso no CM 8 76

## V

- validação de palavras-passe 140, 149
- valores de atributo de chave remota 103
- valores predeterminados 103
- valueObj 106
- variáveis de contentor, fluxo de trabalho 224
- versão, consultar 200
- VideoCharger 65
- VIEW 345
- visualizador de documentos
  - genéricos 467
- visualizador de Java
  - preferências 494
  - tipos de documento 487
- Visualizador de Java
  - definições de propriedades de motor 492
  - selecção de compressão 492
- visualizadores, documento de Java 467

## W

- W3C XML Query 181
- Web Services Description Language (WSDL) 425
- web.xml 309
- WebSphere Application Server
  - autenticar 376
- WebSphere Studio Application Developer (WSAD)
  - construir beans 308
  - criar serviços da Web com 447
  - diagrama 339
- Windows
  - definir o subsistema da consola 23
  - ficheiros DLL de C++ 21
  - iniciar o servidor de RMI 21
- WorkListName 410

- WorkPackageOwner 410
- WorkPackagesToReturn 409
- WSDL 446, 447
  - a partir de uma linha de comandos 447
  - do cliente de administração do sistema 446
- wsdl.exe
  - manual de iniciação de 447
- WSDL2Java 452

## X

- XDO
  - actualizar 49
  - adicionar a partir da memória tampão 47
  - adicionar de um ficheiro 48
  - adicionar um objecto de anotação a 48
  - comparar com atributos 8
  - criar um artigo de recurso 108
  - eliminar 49
  - exemplos 49
  - hierarquia de tipo de classes 108
  - introdução 7
  - Java 44
    - autónomo 47
    - DDO, parte de 45
    - PID 44
    - Sugestões de programação 44
  - objecto grande 87
  - objectos 68
  - obter 49
  - pesquisar texto 200
  - representar conteúdo de multimédia 7
- XML
  - artigos 365
  - beans 312
  - biblioteca de identificadores de beans 309
  - consulta de W3C 181
  - encaminhamento de documentos 404
  - extrair
    - endereço da Web 369
    - ficheiro 369
    - memória tampão 369
  - ferramenta de conversão de esquema 370
  - metadados de administração 343
  - representação do modelo de dados de exemplos de consulta 196
  - representar metadados 365
  - serviços 339
  - suporte de Java 17
  - tipos de artigo 345
- XMLItem 365
- XQuery Path Expressions (XQPE) 181









Número do Programa: 5724-B19  
5697-H60

SC27-1347-07

