HITACHI
Inspire the Next

uCosminexus DocumentBroker Object Loader Version 3

解説書

3020-3-U78

マニュアルの購入方法

このマニュアル，および関連するマニュアルをご購入の際は，巻末の「ソフトウェアマニュアルのサービス ご案内」をご参照ください。

■対象製品

R-1M95E-43 ※ uCosminexus DocumentBroker Object Loader Version 3 03-60（適用 OS：AIX 5L V5.1, AIX 5L V5.2, AIX 5L V5.3）
R-1595E-43 uCosminexus DocumentBroker Object Loader Version 3 03-60（適用 OS：Windows Server 2003, Windows Server 2003 R2, Windows Server 2003 R2（x64), Windows Server 2008 x86）
これらのプログラムプロダクトのほかにもこのマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。
※印の製品については，サポート時期をご確認ください。

■輸出時の注意

本製品を輸出される場合には，外国為替および外国貿易法ならびに米国の輸出管理関連法規などの規制をご確認の上，必要な手続きをお取りください。
なお，ご不明な場合は，弊社担当営業にお問い合わせください。

■商標類

AIX は，米国における米国 International Business Machines Corp. の登録商標です。
CORBA は，Object Management Group が提唱する分散処理環境アーキテクチャの名称です。
GIF は，米国 CompuServe Inc. が開発したフォーマットの名称です。
Microsoft は，米国およびその他の国における米国 Microsoft Corp. の登録商標です。
Microsoft Excel は，米国 Microsoft Corp. の商品名称です。
Microsoft Office Excel は，米国 Microsoft Corp. の商品名称です。
Microsoft Office Word は，米国 Microsoft Corp. の商品名称です。
Microsoft Word は，米国 Microsoft Corp. の商品名称です。
OLE は，米国 Microsoft Corp. が開発したソフトウェア名称です。
TIFF は，米国 Aldus Corp. が開発したフォーマットの名称です。
UNIX は，X/Open Company Limited が独占的にライセンスしている米国ならびに他の国における登録商標です。
Win32 は，米国およびその他の国における米国 Microsoft Corp. の登録商標です。
Windows は，米国およびその他の国における米国 Microsoft Corp. の登録商標です。
Windows Server は，米国 Microsoft Corporation の米国及びその他の国における登録商標です。
ラビニティ，libinity は，株式会社日立システムアンドサービスの登録商標です。

■発行

2009 年 7 月 （第 1 版）3020-3-U78

■著作権

はじめに

このマニュアルは，次に示すプログラムプロダクトの機能と使い方について説明したものです。
- R-1M95E-43　uCosminexus DocumentBroker Object Loader Version 3
- R-1595E-43　uCosminexus DocumentBroker Object Loader Version 3

■対象読者

このマニュアルは，uCosminexus DocumentBroker Object Loader を使用して，uCosminexus DocumentBroker のシステムを構築，運用および管理する方を対象としています。
なお，次の内容を理解されていることを前提としています。
- uCosminexus DocumentBroker のオブジェクトモデルに関する知識
- UNIX または Windows（Windows Server 2003，Windows Server 2003 R2，Windows Server 2003 R2（x64），Windows Server 2008 x86）に関する知識
- 分散オブジェクト技術に関する知識

■マニュアルの構成

このマニュアルは，次に示す七つの章と付録から構成されています。

第 1 章　DocumentBroker Object Loader の概要

uCosminexus DocumentBroker Object Loader の概要について説明しています。

第 2 章　実行環境の設定

uCosminexus DocumentBroker Object Loader の実行環境の設定について説明しています。

第 3 章　ファイル生成

入力ファイル生成ユティリティの機能の概要および生成するファイルの内容について説明しています。

第 4 章　オブジェクトローダで使用するファイル

オブジェクトローダを実行する場合に必要なファイルの記述形式や文法について説明しています。

第 5 章　オブジェクトエクスポートで使用するファイル

オブジェクトエクスポートを実行する場合に必要なファイルの記述形式や文法について説明しています。

第 6 章　コマンドリファレンス

uCosminexus DocumentBroker Object Loader で使用するコマンドの形式と文法について説明しています。

第 7 章　トラブルシュート機能

uCosminexus DocumentBroker Object Loader を保守運用するためのトラブルシュート機能とエラーメッセージについて説明しています。

付録 A　作成コマンドと対象テーブルの関連

uCosminexus DocumentBroker Object Loader のコマンドが使用するテーブルについて説明しています。

付録 B　制御ファイルの状態表

uCosminexus DocumentBroker Object Loader の制御ファイルの状態について説明しています。

付録 C　構成管理コンテナの登録方法

構成管理コンテナの登録方法について説明しています。

付録 D　ConfigurationHistory クラスのエクスポート

異なるクラスのオブジェクト（バージョン付き文書，構成管理コンテナ）が混在する ConfigurationHistory クラスに対してオブジェクトエクスポートを実行する場合の，実行方法および注意事項について説明しています。

付録 E　High-end Option

複数のオブジェクトローダを同時に実行するときに使用する uCosminexus DocumentBroker Object Loader High-end Option について説明しています。

付録 F　同時実行機能の実行環境と運用上の注意事項

uCosminexus DocumentBroker サーバとオブジェクトローダを同時に実行できる同時実行機能を使用する場合の実行環境および運用上の注意事項について説明しています。

付録 G　文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能およびファイル

文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能およびファイルについて説明しています。

■関連マニュアル

このマニュアルの関連マニュアルを次に示します。必要に応じてお読みください。なお，本文に記載のマニュアル名称は，「uCosminexus DocumentBroker」を「DocumentBroker」と表記しています。

uCosminexus DocumentBroker **のマニュアル**

uCosminexus DocumentBroker Version 3　**システム導入・運用ガイド**（3000-3-D01）（3020-3-U71）

uCosminexus DocumentBroker を使用する環境を定義，管理および運用する場合に参照してください。

UNIX の場合は，資料番号が 3000-3-D01 のマニュアルを参照してください。

Windows の場合は，資料番号が 3020-3-U71 のマニュアルを参照してください。

uCosminexus DocumentBroker Version 3　**クラスライブラリ**　C++　**解説**（3000-3-F13）

uCosminexus DocumentBroker Development Kit が提供するクラスライブラリの機能と，クラスライブラリを使用するために必要なオブジェクトモデルについて知りたい場合に参照してください。

uCosminexus DocumentBroker Version 3　**クラスライブラリ**　C++　**リファレンス　基本機能**

編（3000-3-F14）

uCosminexus DocumentBroker Development Kit が提供するクラスライブラリのクラスの詳細とメソッドの文法について知りたい場合に参照してください。

uCosminexus DocumentBroker Version 3　**オブジェクト操作ツール**（3000-3-F15）

uCosminexus DocumentBroker Development Kit または uCosminexus DocumentBroker Runtime で提供するオブジェクト操作ツールの機能について知りたい場合に参照してください。

uCosminexus DocumentBroker Version 3　**メッセージ**（3000-3-F12）

uCosminexus DocumentBroker が出力するメッセージについて知りたい場合に参照してください。

uCosminexus DocumentBroker Version 3　**クラスライブラリ**　Java　**解説**（3000-3-F16）

Java クラスライブラリの機能および環境設定の方法について知りたい場合に参照してください。

uCosminexus DocumentBroker Version 3　**クラスライブラリ**　Java　**リファレンス**（3000-3-F17）

Java クラスライブラリの詳細，インターフェースの詳細，メソッドの文法およびメッセージの詳細について知りたい場合に参照してください。

関連製品のマニュアル（HiRDB）

- **スケーラブルデータベースサーバ**　HiRDB Version 6　**システム定義**（UNIX(R) **用**）（3000-6-233）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 6　**システム定義**（Windows(R) **用**）（3020-6-123）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 7　**システム定義**（UNIX(R) **用**）（3000-6-273）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 7　**システム定義**（Windows(R) **用**）（3020-6-273）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 8　**システム定義**（UNIX(R) **用**）（3000-6-353）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 8　**システム定義**（Windows(R) **用**）（3020-6-353）※1
- **スケーラブルデータベースサーバ**　HiRDB Version 6　**システム運用ガイド**（UNIX(R) **用**）（3000-6-234）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 6　**システム運用ガイド**（Windows(R) **用**）（3020-6-124）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 7　**システム運用ガイド**（UNIX(R) **用**）（3000-6-274）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 7　**システム運用ガイド**（Windows(R) **用**）

（3020-6-274）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 8　**システム運用ガイド**（UNIX(R) **用**）（3000-6-354）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 8　**システム運用ガイド**（Windows(R) **用**）（3020-6-354）※2
- **スケーラブルデータベースサーバ**　HiRDB Version 6　UAP **開発ガイド**（UNIX(R)/Windows(R) **用**）（3000-6-236）※3
- **スケーラブルデータベースサーバ**　HiRDB Version 6　UAP **開発ガイド**（Windows(R) **用**）（3020-6-126）※3
- **スケーラブルデータベースサーバ**　HiRDB Version 7　UAP **開発ガイド**（UNIX(R)/Windows(R) **用**）（3000-6-276）※3
- **スケーラブルデータベースサーバ**　HiRDB Version 7　UAP **開発ガイド**（Windows(R) **用**）（3020-6-276）※3
- **スケーラブルデータベースサーバ**　HiRDB Version 8　UAP **開発ガイド**（3020-6-356）※3
- **スケーラブルデータベースサーバ**　HiRDB Version 6 **メッセージ**（UNIX(R)/Windows(R) **用**）（3000-6-238）※4
- **スケーラブルデータベースサーバ**　HiRDB Version 6 **メッセージ**（Windows(R) **用**）（3020-6-128）※4
- **スケーラブルデータベースサーバ**　HiRDB Version 7 **メッセージ**（UNIX(R)/Windows(R) **用**）（3000-6-278）※4
- **スケーラブルデータベースサーバ**　HiRDB Version 7 **メッセージ**（Windows(R) **用**）（3020-6-278）※4
- **スケーラブルデータベースサーバ**　HiRDB Version 8　**メッセージ**（3020-6-358）※4
- HiRDB **全文検索プラグイン**　HiRDB Text Search Plug-in Version 2（3000-6-245）※5
- HiRDB **全文検索プラグイン**　HiRDB Text Search Plug-in Version 7（3000-6-288）※5
- HiRDB **全文検索プラグイン**　HiRDB Text Search Plug-in Version 8（3020-6-375）※5
- HiRDB Adapter for XML　**ユーザーズガイド**（3000-6-250）
- HiRDB Adapter for XML　**リファレンス**　C++ **編**（3000-6-251）
- HiRDB Adapter for XML　**リファレンス**　Java **編**（3000-6-252）

関連製品のマニュアル（その他）
- Preprocessing Library for Text Search Version 2（3000-7-270）

注※1

このマニュアルでは，これらのマニュアルを「HiRDB　システム定義」と表記しています。

注※2

このマニュアルでは，これらのマニュアルを「HiRDB　システム運用ガイド」と表記しています。

注※ 3

このマニュアルでは，これらのマニュアルを「HiRDB　UAP 開発ガイド」と表記しています。

注※ 4

このマニュアルでは，これらのマニュアルを「HiRDB　メッセージ」と表記しています。

注※ 5

このマニュアルでは，これらのマニュアルを「HiRDB Text Search Plug-in」と表記しています。

■読書手順

このマニュアルは，利用目的に合わせて章を選択してお読みいただけます。利用目的別の流れに従ってお読みいただくことをお勧めします。

■このマニュアルで使用する記号

このマニュアルで使用する記号を次に示します。

記号	意　味
|	横に並べられた複数の項目に対する項目間の区切りを示し，「または」を意味します。 （例）　A | B 　A または B を指定することを示します。

記号	意　味
＿	括弧で囲まれた複数項目のうち 1 項目に対し使用され，括弧内のすべてを省略したときシステムが取る標準値を示します。 （例）〔A ｜ B〕 「何も指定しない」か「A または B を指定する」ことを示します。何も指定しない場合は A が仮定されます。
{ }	この記号で囲まれている複数の項目のうちから一つを選択することを示します。項目が横に並べられ，記号｜で区切られている場合は，そのうちの一つを選択します。 （例）{A ｜ B ｜ C} A，B または C のどれかを指定することを示します。
〔 〕	この記号で囲まれている項目は省略してもよいことを意味します。複数の項目が横に並べて記述されている場合には，すべてを省略するか，記号 { } と同じくどれか一つを選択します。 （例 1）〔A〕 「何も指定しない」か「A を指定する」ことを示します。 （例 2）〔B ｜ C〕 「何も指定しない」か「B または C を指定する」ことを示します。
：：＝	この記号の左にあるものを右にあるもので定義することを示します。 （例） A：：＝ B 「A とは B である」と定義することを示します。
…	記述が省略されていることを示します。 （例） ABC… ABC の後ろに記述があり，その記述が省略されていることを示します。

■このマニュアルで使用する構文要素

このマニュアルで使用する構文要素の種類を次に示します。

種　類	定　義
英　字	A ～ Z　a ～ z
英小文字	a ～ z
英大文字	A ～ Z
数　字	0 ～ 9
英数字	A ～ Z　a ～ z　0 ～ 9
記　号	！　"　#　$　%　&　’ ( )　＋　,　＿　・　.　／　： ；　＜　＝　＞　@　[　]　^　－　{　}　‾　空白　¥

注　すべて半角文字を使用してください。

■このマニュアルの図中で使用する記号

このマニュアルの図中で使用する記号を，次のように定義します。

●データの流れ　●入出力の動作　●その他の流れ

●プログラム　●データベース
またはファイル

■このマニュアルでの表記

このマニュアルでは，製品名称を次に示す略称で表記しています。

製品名称	略称
AIX 5L V5.1	AIX
AIX 5L V5.2	
AIX 5L V5.3	
uCosminexus DocumentBroker Development Kit Version 3	DocumentBroker
uCosminexus DocumentBroker Runtime Version 3	
uCosminexus DocumentBroker Server Version 3	
uCosminexus DocumentBroker Object Loader Version 3	
uCosminexus DocumentBroker Text Search Index Loader Version 3	
uCosminexus DocumentBroker Server Version 3	DocumentBroker サーバ
uCosminexus DocumentBroker Object Loader Version 3	DocumentBroker Object Loader
uCosminexus DocumentBroker Object Loader High-end Option Version 3	High-end Option
HiRDB/Parallel Server Version 6	HiRDB
HiRDB/Parallel Server Version 7	
HiRDB/Parallel Server Version 8	
HiRDB/Single Server Version 6	
HiRDB/Single Server Version 7	
HiRDB/Single Server Version 8	
HiRDB/Workgroup Server Version 6	
HiRDB Adapter for XML - Enterprise Edition	HiRDB Adapter for XML
HiRDB Adapter for XML - Standard Edition	

製品名称	略称
HiRDB Text Search Plug-in Version 2	HiRDB Text Search Plug-in
HiRDB Text Search Plug-in Version 7	
HiRDB Text Search Plug-in Version 8	
Microsoft(R) Excel	Excel
Microsoft(R) Office Excel	
Microsoft(R) Office Word	Word
Microsoft(R) Word	
Microsoft(R) Windows Server(R) 2003, Enterprise Edition	Windows Server 2003
Microsoft(R) Windows Server(R) 2003, Standard Edition	
Microsoft(R) Windows Server(R) 2003 R2, Enterprise Edition	Windows Server 2003 R2
Microsoft(R) Windows Server(R) 2003 R2, Standard Edition	
Microsoft(R) Windows Server(R) 2003 R2, Enterprise x64 Edition 日本語版	Windows Server 2003 R2 または Windows Server 2003 R2（x64）
Microsoft(R) Windows Server(R) 2003 R2, Standard x64 Edition 日本語版	
Microsoft(R) Windows Server(R) 2008 Enterprise 32-bit 日本語版	Windows Server 2008 または Windows Server 2008 x86
Microsoft(R) Windows Server(R) 2008 Standard 32-bit 日本語版	
Preprocessing Library for Text Search Version 2	Preprocessing Library for Text Search

このほか，このマニュアルでは，次に示す表記方法を使用しています。

- AIX を，UNIX と表記することがあります。
- Windows Server 2003，Windows Server 2003 R2，Windows Server 2003 R2（x64），および Windows Server 2008 x86 を総称して，Windows と表記することがあります。

■適用 OS の違いによる機能相違点の表記

このマニュアルは，AIX および Windows に適用します。OS によって記述を書き分ける場合，次に示す表記を使用して，それぞれの説明に OS 名を明記しています。

表記	意味
AIX の場合	AIX に該当
UNIX の場合	AIX に該当
Windows の場合	Windows に該当

■ uCosminexus DocumentBroker のマニュアルで使用する略語

uCosminexus DocumentBroker のマニュアルで使用する英略語を次に示します。

英略語	英字での表記
ACE	Access Control Element
ACFlag	Access Control Flag
ACL	Access Control List
AIIM	Association for Information and Image Management International
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BES	Back End Server
BLOB	Binary Large Object
BMP	Bit Map
BNF	Backus Normal Form
BOA	Basic Object Adapter
BOM	Byte Order Mark
CD-ROM	Compact Disc Read Only Memory
CGI	Common Gateway Interface
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
CR	Carriage Return
CSV	Comma Separated Value
DAP	Directory Access Protocol
DAT	Digital Audio Tape
DB	Database
DBMS	Database Management System
DCD	Document Content Description
DDE	Dynamic Data Exchange
DIT	Directory Information Tree
DLL	Dynamic Linking Library
DMA	Document Management Alliance
DN	Distinguished Name
DTD	Document Type Definition
EOF	End of File
ESIS-B	Element Structure Information Set-Binary Format
EUC	Extended UNIX Code

英略語	英字での表記
GIF	Graphics Interchange Format
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IANA	Internet Assigned Numbers Authority
ID	Identifier
IPF	Itanium(R) Processor Family
ISO	International Organization for Standardization
JIS	Japanese Industrial Standards
JPEG	Joint Photographic Expert Group
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LF	Line Feed
MFC	Microsoft Foundation Class
MIME	Multipurpose Internet Mail Extensions
OCR	Optical Character Reader
OIID	Object Instance Identifier
OLE	Object Linking and Embedding
OMG	Object Management Group
ORB	Object Request Broker
ORDB	Object Relational Database
OS	Operating System
OTS	Object Transaction Service
PC	Personal Computer
PDF	Portable Document Format
PP	Program Product
RDB	Relational Database
RDN	Relative Distinguished Name
RFC	Request for Comment
RTF	Rich Text Format
TCP/IP	Transmission Control Protocol/Internet Protocol
TIFF	Tag Image File Format

英略語	英字での表記
UAP	User Application Program
UCS-2	Universal Character Set coded in 2 octets
UCS-4	Universal Character Set coded in 4 octets
URL	Uniform Resource Locator
UTC	Universal Time Coordinated
UTF-8	8-bit UCS Transformation Format
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	Extensible Markup Language

■常用漢字以外の漢字の使用について

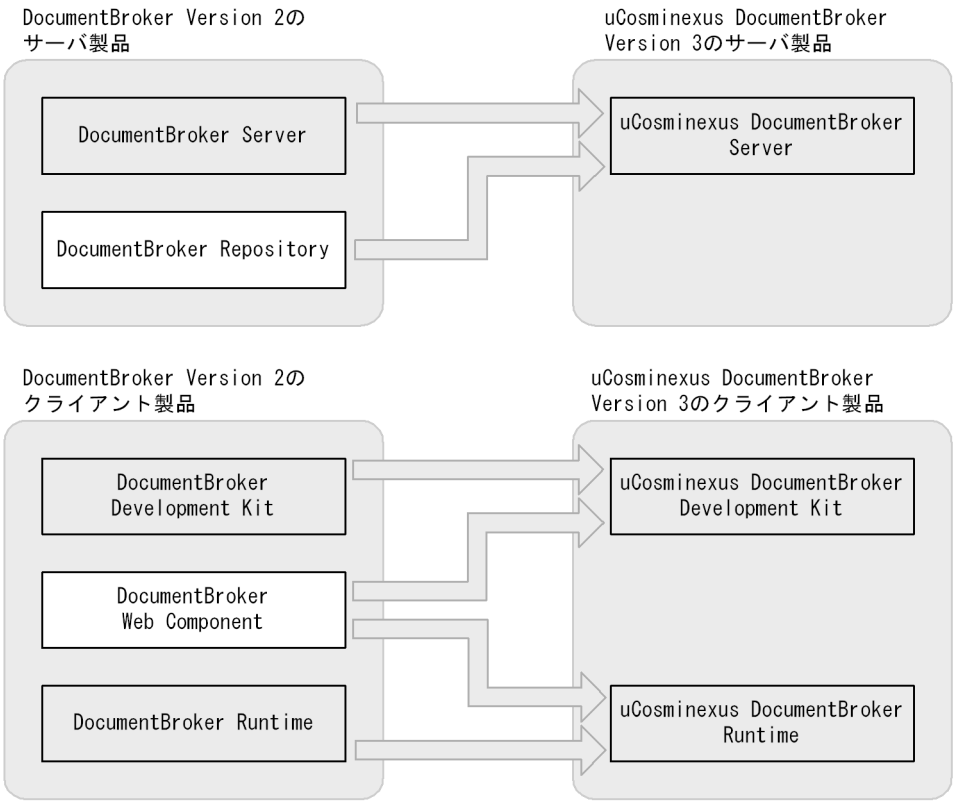
このマニュアルでは常用漢字を使用することを基本としていますが，次に示す用語については，常用漢字以外の漢字を使用しています。
個所（かしょ） 必須（ひっす）

■ KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

■ DocumentBroker Version 2 と uCosminexus DocumentBroker Version 3 の製品体系の違い

uCosminexus DocumentBroker Version 3 では次のように製品の統合を行いました。
DocumentBroker Repository を uCosminexus DocumentBroker Server に統合しました。
DocumentBroker Web Component を uCosminexus DocumentBroker Development Kit および uCosminexus DocumentBroker Runtime に統合しました。
DocumentBroker Version 2 と uCosminexus DocumentBroker Version 3 の製品体系の違いを次に示します。



■ DocumentBroker Version 2 と uCosminexus DocumentBroker Version 3 のマニュアルの対応

バージョンアップおよび製品体系の変更に伴い，uCosminexus DocumentBroker Version 3 では次に示すようにマニュアル名称を変更しました。

Version 2 のマニュアル名称	Version 3 のマニュアル名称
DocumentBroker Version 2　システム導入・運用ガイド	uCosminexus DocumentBroker Version 3　システム導入・運用ガイド
DocumentBroker Version 2　クラスライブラリ　解説	uCosminexus DocumentBroker Version 3　クラスライブラリ　C++　解説
DocumentBroker Version 2　クラスライブラリ　リファレンス　基本機能編	uCosminexus DocumentBroker Version 3　クラスライブラリ　C++　リファレンス　基本機能編
DocumentBroker Version 2　クラスライブラリ　リファレンス　概念 SGML 文書管理機能編	廃版
DocumentBroker Version 2　オブジェクト操作ツール	uCosminexus DocumentBroker Version 3　オブジェクト操作ツール
DocumentBroker Version 2　統計解析ツール	uCosminexus DocumentBroker Version 3　統計解析ツール

Version 2 のマニュアル名称	Version 3 のマニュアル名称
DocumentBroker Version 2　メッセージ	uCosminexus DocumentBroker Version 3　メッセージ
DocumentBroker Web Component Version 2　解説	uCosminexus DocumentBroker Version 3　クラスライブラリ　Java　解説
DocumentBroker Web Component Version 2　リファレンス	uCosminexus DocumentBroker Version 3　クラスライブラリ　Java　リファレンス
DocumentBroker Web Component Version 2　サンプル Web アプリケーション	uCosminexus DocumentBroker Version 3　クラスライブラリ　Java　サンプル Web アプリケーション
DocumentBroker Text Search Index Loader Version 2	uCosminexus DocumentBroker Text Search Index Loader Version 3
DocumentBroker Rendering Option　システム導入運用ガイド	uCosminexus DocumentBroker Rendering Option Version 3
DocumentBroker Object Loader Version 2	uCosminexus DocumentBroker Object Loader Version 3

目次

図目次

表目次

1 DocumentBroker Object Loader の概要

この章では，DocumentBroker Object Loader の概要について説明します。

1.1 DocumentBroker Object Loader とは

ここでは，DocumentBroker Object Loader の概要について説明します。

DocumentBroker Object Loader は，オブジェクトローダとオブジェクトエクスポートで構成される DocumentBroker のためのユティリティ機能製品です。オブジェクトローダは，ユーザが定義した情報に基づいて，オブジェクトを DocumentBroker のデータベースへ登録するユティリティです。オブジェクトエクスポートは，ユーザが定義した情報に基づいて，構築済みである DocumentBroker の文書空間のオブジェクトデータを取得して，オブジェクトローダの入力データを生成するユティリティです。

オブジェクトローダ

オブジェクトローダは，データベースへオブジェクトを登録する場合に，DocumentBroker が提供するクラスライブラリのインタフェースを経由しないで，直接 HiRDB の SQL を発行して登録します。

オブジェクトローダを利用してデータベースに登録できるオブジェクトは次のとおりです。詳細については，「1.4.1　処理概要（オブジェクトローダ）」を参照してください。

- 文書およびコンテナに対する関連オブジェクト
- バージョン付き文書およびコンテナに対する関連オブジェクト
- コンテナおよびコンテナ間の関連を示す関連オブジェクト
- 独立永続化クラス
- 可変長配列
- オブジェクトごとのアクセス制御情報（ACL）データ
  登録できるアクセス制御情報は，アクセス制御フラグ（ACFlag），およびアクセス制御リスト（ローカル ACL，パブリック ACL）です。
- マルチレンディション文書
- 構成管理コンテナ
- XML 文書
- マルチファイル文書
- リファレンスファイル文書

オブジェクトエクスポート

オブジェクトエクスポートは，文書空間のオブジェクトデータを取得する場合に，ユーザ設定条件に該当するオブジェクトを HiRDB の SQL を発行して抽出し，抽出データをオブジェクトローダの入力データファイルのフォーマットで出力します。

オブジェクトエクスポートを利用して抽出できる DocumentBroker のクラスのインスタンスは次のとおりです。詳細については，「1.5.1　処理概要（オブジェクトエクスポート）」を参照してください。

- 文書およびコンテナに対する関連オブジェクト（DCR だけ）
- バージョン付き文書およびコンテナに対する関連オブジェクト（DCR だけ）
- コンテナおよびコンテナ間の関連を示す関連オブジェクト（DCR だけ）

- 独立永続化クラス
- 可変長配列
- オブジェクトごとのアクセス制御情報（ACL）データ
  抽出できるアクセス制御情報は，アクセス制御フラグ（ACFlag）だけです。
- 構成管理コンテナ
- XML 文書

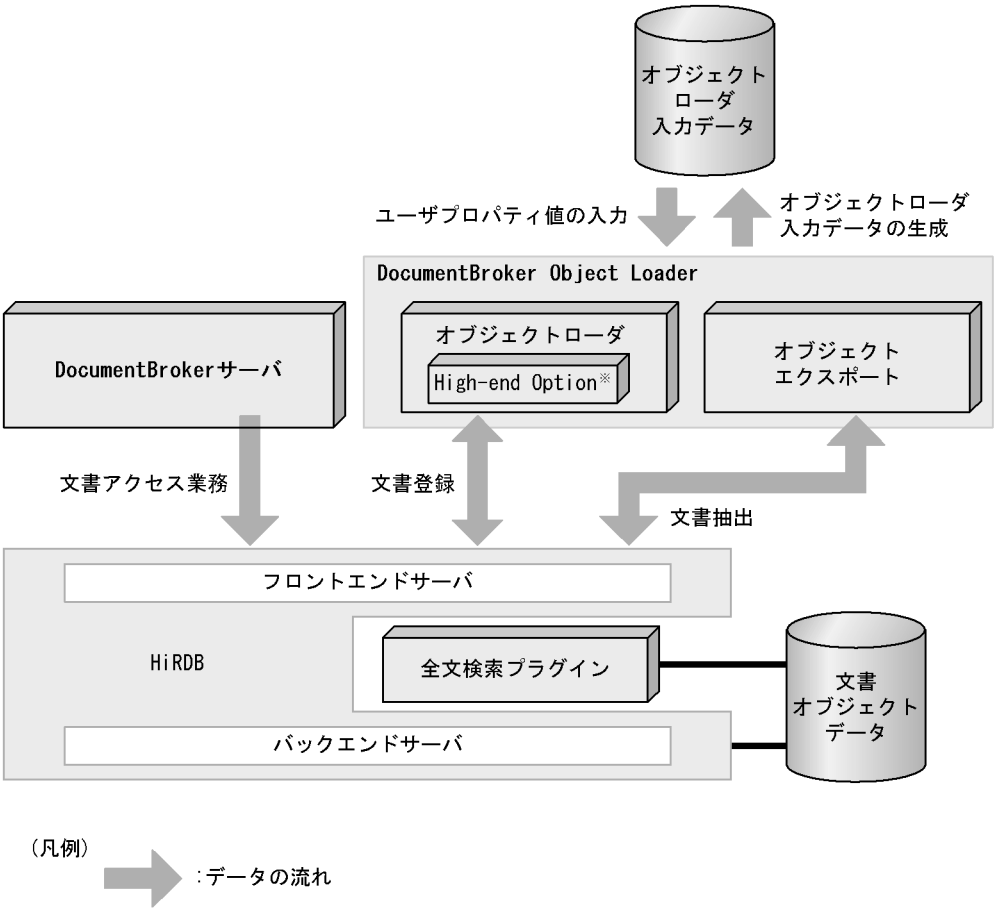
1.2　システム構成

ここでは，DocumentBroker Object Loader のシステム構成について説明します。

DocumentBroker Object Loader が動作するためのシステム構成を図 1-1 に示します。

オブジェクトローダは，DocumentBroker サーバと同一のマシン上で動作することが前提となります。

図 1-1　DocumentBroker Object Loader のシステム構成



注※　同時に複数のオブジェクトローダを実行する場合に必要です。

1.3 DocumentBroker Object Loader の機能

ここでは，DocumentBroker Object Loader の機能について説明します。

1.3.1 提供するユティリティ機能

DocumentBroker Object Loader は，次に示す**ユティリティ機能**を提供しています。

- データベースロードユティリティ（オブジェクトローダ）
- 環境設定ユティリティ
- 入力ファイル生成ユティリティ
- データベース移行ユティリティ（オブジェクトエクスポート）

DocumentBroker Object Loader の処理は，DocumentBroker が定義するすべてのクラスとプロパティに対応する表および列がデータベースに存在することが前提になります。

次の表に各ユティリティに対応するコマンド名と，機能概要を示します。

なお，各ユティリティに対応するコマンド仕様の詳細は，「6. コマンドリファレンス」を参照してください。

表 1-1 DocumentBroker Object Loader のユティリティ機能の一覧

ユティリティ名称	コマンド名	機能概要
オブジェクトローダ	EDMLoad	SQL の発行によってデータベースにオブジェクトをロードする。
環境設定	EDMLodSetup	UNIX の場合，DocumentBroker Object Loader の実行環境を作成・削除する。 Windows の場合，このユティリティは使用しない。
定義ファイル生成	EDMCrtLDF	オブジェクトローダ・オブジェクトエクスポートの実行に必要な定義ファイルを生成する。
制御ファイル生成	EDMCrtLCF	オブジェクトローダ・オブジェクトエクスポートの実行に必要な制御ファイルを生成する。
オブジェクトエクスポート	EDMExport	データベースからオブジェクトを取得してオブジェクトローダの実行時に必要な入力データファイルを生成する。

これ以降の説明では，DocumentBroker Object Loader の全体の総称を「DocumentBroker Object Loader」と表記します。各ユティリティ機能については，それぞれのユティリティ名称を使用して説明します。

1.3.2 ユティリティ機能の関連

ユティリティ機能の相互関連について説明します。

（1） オブジェクトローダ使用時の関連

DocumentBroker Object Loader のオブジェクトローダを使用する場合の各ユティリティ機能の関連を図 1-2 に示します。

図 1-2　ユティリティ機能の関連（オブジェクトローダ使用時）

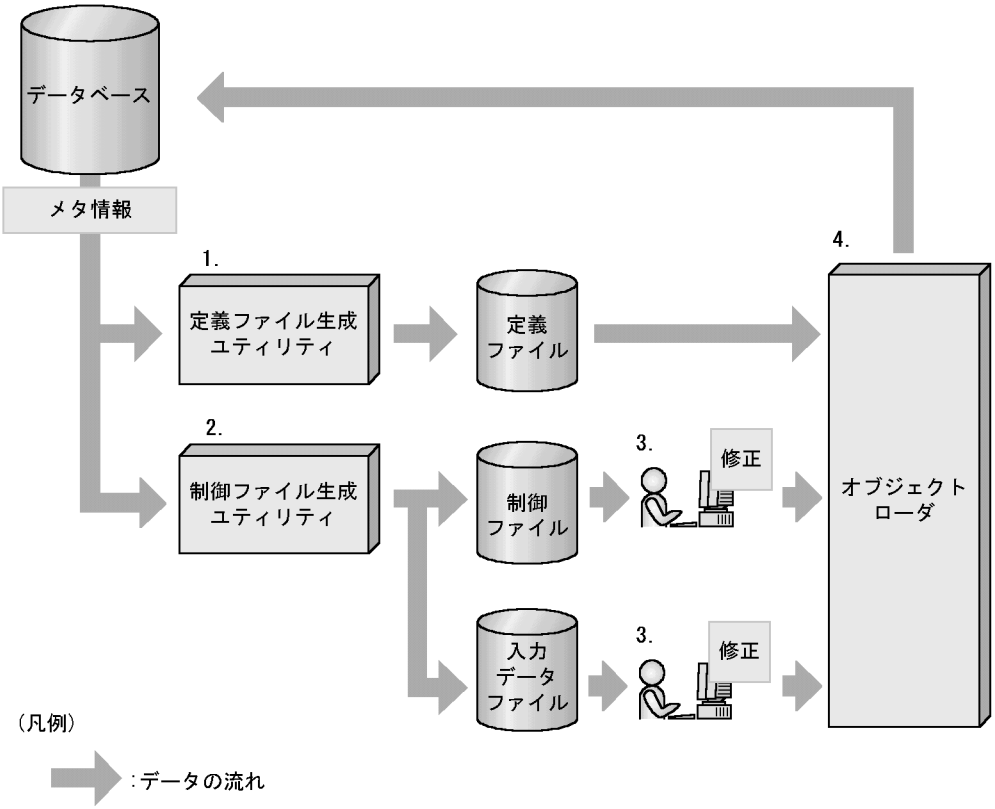


図 1-2 について説明します。

1. 定義ファイル生成ユティリティを実行して，データベースからメタ情報を取得し，定義ファイルを生成します。
   定義ファイル生成ユティリティの詳細については，「6.　コマンドリファレンス」の「EDMCrtLDF（定義ファイル生成）」を参照してください。
2. 制御ファイル生成ユティリティを実行して，データベースからメタ情報を取得し，制御ファイルおよび入力データファイルを生成します。
   制御ファイル生成ユティリティの詳細については，「6.　コマンドリファレンス」の「EDMCrtLCF（制御ファイル生成）」を参照してください。
3. ユーザが必要に応じて，制御ファイルおよび入力データファイルを修正します。
   制御ファイルの詳細については，「4.3　制御ファイル」を参照してください。入力データファイルの詳細については，「4.5　入力データファイル」を参照してください。

4. 上記の作業で作成した定義ファイル，制御ファイル，入力データファイルを使用して，オブジェクトローダを実行しデータベースにオブジェクトを登録します。
オブジェクトローダの詳細については，「6.　コマンドリファレンス」の「EDMLoad（オブジェクトローダの実行）」を参照してください。

（2）オブジェクトエクスポート使用時の関連

DocumentBroker Object Loader のオブジェクトエクスポートを使用する場合の各ユティリティ機能の関連を図 1-3 に示します。

図 1-3　ユティリティ機能の関連（オブジェクトエクスポート使用時）

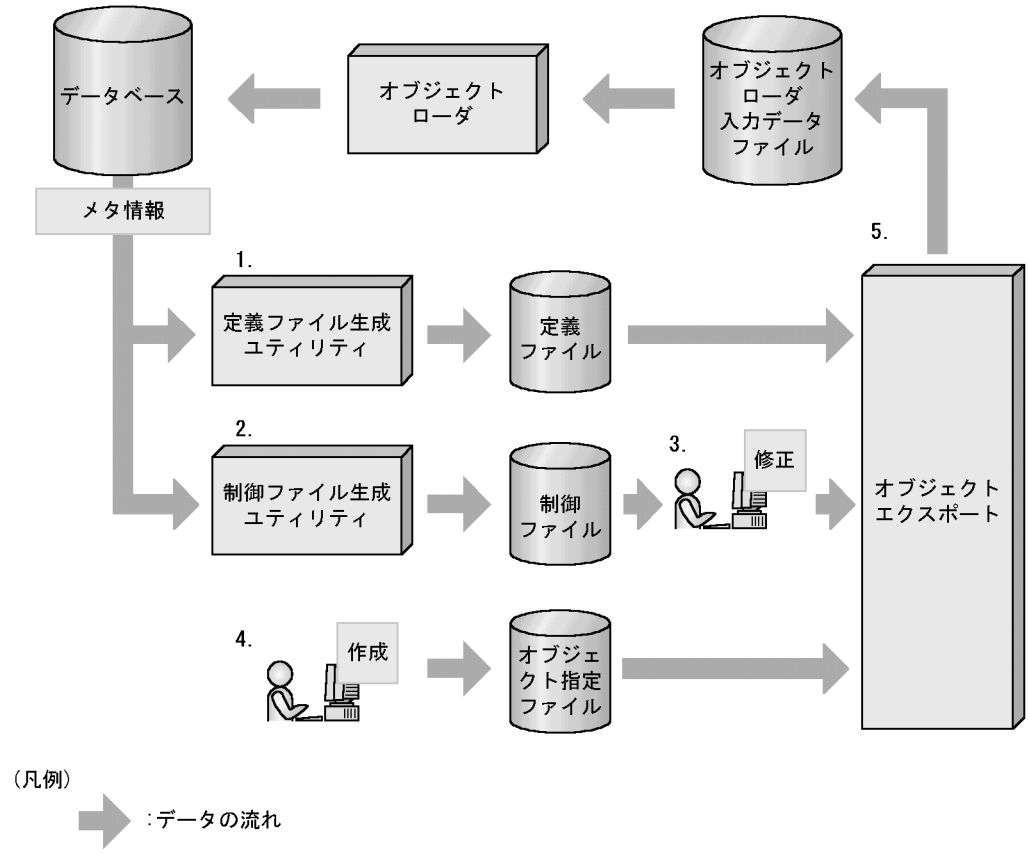


図 1-3 について説明します。

1. 定義ファイル生成ユティリティを実行して，データベースからメタ情報を取得し，定義ファイルを生成します。
定義ファイル生成ユティリティの詳細については，「6.　コマンドリファレンス」の「EDMCrtLDF（定義ファイル生成）」を参照してください。
2. 制御ファイル生成ユティリティを実行して，データベースからメタ情報を取得し，制御ファイルを生成します。

制御ファイル生成ユティリティの詳細については，「6. コマンドリファレンス」の「EDMCrtLCF（制御ファイル生成）」を参照してください。

3. ユーザが必要に応じて，制御ファイルを修正します。
   制御ファイルの詳細については，「5.3 制御ファイル」を参照してください。
4. データベースから移行するオブジェクトを指定するためのオブジェクト指定ファイルを，ユーザが作成します。
   オブジェクト指定ファイルの詳細については，「5.5 オブジェクト指定ファイル」を参照してください。
5. 上記の作業で作成した定義ファイル，制御ファイル，オブジェクト指定ファイルを使用して，オブジェクトエクスポートを実行し，オブジェクトローダ入力データファイルを生成します。
   オブジェクトエクスポートの詳細については，「6. コマンドリファレンス」の「EDMExport（オブジェクトエクスポートの実行）」を参照してください。

1.4　オブジェクトローダの処理概要

ここでは，オブジェクトローダの処理の概要について説明します。

1.4.1　処理概要（オブジェクトローダ）

オブジェクトローダは，制御ファイル，定義ファイルおよび入力データファイルを基に，DocumentBroker で扱うオブジェクトをデータベースに登録します。実行時には，エラー情報やトレース情報も取得します。

オブジェクトローダは，DocumentBroker の次に示すクラスに対するインスタンスのデータベースへの入力を対象とします。

なお，登録対象のクラスおよびコマンド一覧については，「付録 A　作成コマンドと対象テーブルの関連」を参照してください。

(1) 文書およびコンテナに対する関連オブジェクト

文書（DocVersion オブジェクト）と文書をコンテナ（Container オブジェクト）に関連づけるための関連オブジェクト（DirectContainmentRelationship オブジェクトおよび ReferentialContainmentRelationship オブジェクト）を登録できます。

(2) バージョン付き文書およびコンテナに対する関連オブジェクト

バージョン付き文書（バージョンを取得する DocVersion オブジェクトとバージョンを管理するのに必要な ConfigurationHistory オブジェクト，VersionSeries オブジェクトおよび VersionDescription オブジェクト）を登録できます。

また，文書をコンテナ（Container オブジェクト）に関連づけるための関連オブジェクト（DirectContainmentRelationship オブジェクトおよび ReferentialContainmentRelationship オブジェクト）を登録できます。

(3) コンテナおよびコンテナ間の関連を示す関連オブジェクト

コンテナ（Container オブジェクト）を登録できます。

また，コンテナ間の関連を示す関連オブジェクト（DirectContainmentRelationship オブジェクトおよび ReferentialContainmentRelationship オブジェクト）を登録できます。

(4) 独立永続化クラス

独立永続化クラス（edmClass_IndependentPersistence クラス）のインスタンスとして生成するオブジェクトを登録できます。

(5) 可変長配列

可変長配列（edmClass_Struct クラス）のサブクラスのインスタンスとして生成するオブジェクト（基本単位が VariableArray 型のプロパティから参照されるオブジェクト）を登録できます。

登録方法については，VariableArray 型プロパティの要素を HiRDB の別表に格納する場合は，「4.5.3　記述するコマンドの一覧」の「表 4-11　コマンド一覧」，および「4.5.5(2)　作成コマンドとプロパティの関係」の「表 4-14　作成コマンドと指定するプロパティの関係」を参照してください。HiRDB の繰り返しデータ（HiRDB Array 列）に格納する場合は，「4.5.5(7)(d)　繰り返しデータ（HiRDB Array 列）の指定」を参照してください。

(6) オブジェクトごとのアクセス制御情報（ACL）データの登録

DocumentBroker オブジェクトに付与するアクセス制御情報データを登録できます。登録できるアクセス制御情報は，アクセス制御フラグ（ACFlag）およびアクセス制御リスト（ローカル ACL，パブリック ACL）です。アクセス制御リストのセキュリティ ACL は登録できません。

(7) マルチレンディション文書の登録

マルチレンディション文書（文書およびバージョン付き文書）を登録できます。

(8) 構成管理コンテナ

構成管理コンテナ（ContainerVersion オブジェクト），構成管理コンテナと構成要素を関連づけるための関連オブジェクト（VersionTraceableContainmentRelationship オブジェクト）および構成要素のオブジェクト（VersionTracedDocVersion オブジェクトおよび VersionTracedComponentDocVersion オブジェクト）を登録できます。

(9) XML 文書

XML 文書（文書およびバージョン付き文書）を登録できます。

(10) マルチファイル文書の登録

複数のファイルから構成される一つの文書を登録できます。この文書を**マルチファイル文書**といいます。マルチファイル文書を使用すると，複数のファイルをまとめて管理できます。

(11) リファレンスファイル文書の登録

文書の実体であるコンテントをファイルシステムの任意のディレクトリに格納し，文書のプロパティとコンテントの格納先の情報（コンテントロケーション）をデータベースに登録して管理する文書を登録できます。この文書を**リファレンスファイル文書**といいます。リファレンスファイル文書を使用すると，コンテントの格納先を移行する場合な

どは，データベースに影響しないで，コンテント格納先ベースパスを変更するだけで，コンテントの格納領域を変更できます。また，コンテントをデータベースに格納しないため，データベースの容量を削減できます。

（12）同時実行機能

オブジェクトローダは，DocumentBroker サーバと同時に実行できます。これを**同時実行機能**と呼びます。以降，オブジェクトローダの実行時に DocumentBroker サーバが起動している状態を**同時実行モード**，DocumentBroker サーバが停止している状態を**非同時実行モード**と呼びます。なお，同時実行モードと非同時実行モードの詳細については，「付録 F　同時実行機能の実行環境と運用上の注意事項」を参照してください。

1.4.2　ファイル環境（オブジェクトローダ）

オブジェクトローダを実行するために必要なファイルとエラー情報やトレース情報が取得されるファイルについて説明します。

（1）オブジェクトローダを実行するために必要なファイル

オブジェクトローダを実行するために必要なファイルについて説明します。なお，各ファイルの記述形式や文法などについては，「4.　オブジェクトローダで使用するファイル」を参照してください。

（a）制御ファイル

制御ファイルは，オブジェクトローダの制御情報と，ユーザが追加するサブクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを記述するテキストファイルです。

（b）定義ファイル

定義ファイルは，GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルや入力データファイルには，定義ファイルに記述した名称を記述します。

（c）入力データファイル

入力データファイルは，DocumentBroker のデータベースに登録するプロパティの値を，制御ファイルで記述したプロパティの並び順に記述するテキストファイルです。複数のオブジェクトの登録を 1 トランザクションで実行させる，トランザクション単位の記述ができます。

（d）環境変数ファイル（Windows の場合）

Windows の場合，オブジェクトローダが動作するために必要な情報を設定するファイルです。

UNIX の場合，このファイルはありません。

（2）エラー情報およびトレース情報を取得するファイル

オブジェクトローダを実行したときに発生するエラー情報やトレース情報を取得するファイルについて説明します。各ファイルの詳細については，「7. トラブルシュート機能」を参照してください。

(a) エラーデータファイル

エラーデータファイルは，オブジェクトの登録に失敗した場合，失敗した入力データファイルの定義を出力するファイルです。エラーデータファイルを修正したあとは，このファイルを入力データファイルとして，オブジェクトローダを再実行できます。

(b) エラーログファイル

エラーログファイルは，オブジェクトローダ実行時のエラー，ワーニング，インフォメーション情報を出力するファイルです。制御ファイルにエラーログファイルの出力を指定した場合，指定のファイルに出力されます。

(c) トレースログファイル

トレースログファイルは，オブジェクトローダのトレース情報を出力するファイルです。主に，オブジェクトローダの保守に使用します。

1.5 オブジェクトエクスポートの処理概要

ここでは，オブジェクトエクスポートの処理の概要について説明します。

1.5.1 処理概要（オブジェクトエクスポート）

オブジェクトエクスポートは制御ファイルや定義ファイルの情報に基づいて，オブジェクト指定ファイルに記述した条件に該当するオブジェクトを DocumentBroker の文書空間から取得して，オブジェクトローダの入力データファイルの形式に変換したファイルを生成します。

オブジェクトエクスポートは，DocumentBroker の次のクラスのインスタンスのエクスポートをサポートします。

(1) 文書およびコンテナに対する関連オブジェクト（DCR だけ）

文書（DocVersion オブジェクト）と文書をコンテナ（Container オブジェクト）に関連づけるための関連オブジェクト（DirectContainmentRelationship オブジェクト）を抽出できます。

(2) バージョン付き文書およびコンテナに対する関連オブジェクト（DCR だけ）

バージョン付き文書（バージョンを取得する DocVersion オブジェクトとバージョンを管理するのに必要な ConfigurationHistory オブジェクト，VersionSeries オブジェクトおよび VersionDescription オブジェクト）を抽出できます。

また，文書をコンテナ（Container オブジェクト）に関連づけるための関連オブジェクト（DirectContainmentRelationship オブジェクト）を抽出できます。

(3) コンテナおよびコンテナ間の関連を示す関連オブジェクト（DCR だけ）

コンテナ（Container オブジェクト）を抽出できます。

また，コンテナ間の関連を示す関連オブジェクト（DirectContainmentRelationship オブジェクト）を抽出できます。

(4) 独立永続化クラス

独立永続化クラス（edmClass_IndependentPersistence クラス）のインスタンスとして生成するオブジェクトを抽出できます。

(5) 可変長配列

可変長配列（edmClass_Struct クラス）のサブクラスのインスタンスとして生成するオ

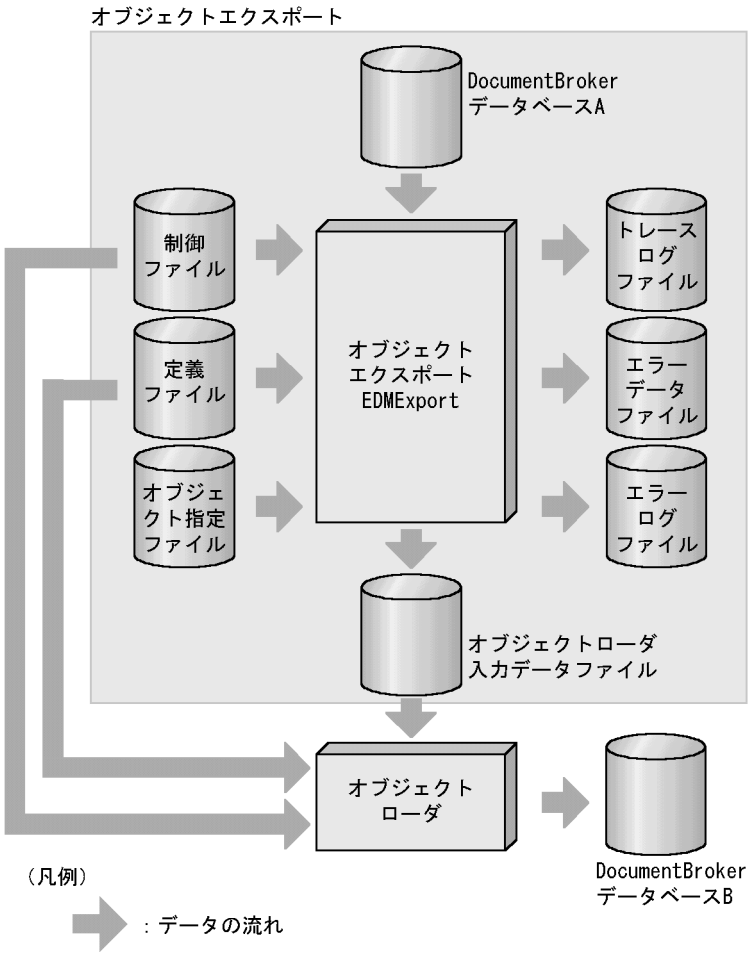
ブジェクト（基本単位が VariableArray 型のプロパティから参照されるオブジェクト）を抽出できます。

（6） オブジェクトごとのアクセス制御情報（ACL）データの抽出

DocumentBroker オブジェクトに付与するアクセス制御情報データを抽出できます。抽出できるアクセス制御情報は，アクセス制御フラグ（ACFlag）だけです。アクセス制御リスト（ローカル ACL，パブリック ACL およびセキュリティ ACL）は抽出できません。

オブジェクトエクスポートの処理の概要を図 1-4 に示します。

図 1-4　オブジェクトエクスポートの処理概要



（7） 構成管理コンテナ

構成管理コンテナ（ContainerVersion オブジェクト），構成管理コンテナと構成要素を関連づけるための関連オブジェクト（VersionTraceableContainmentRelationship オブ

ジェクト）および構成要素のオブジェクト（VersionTracedDocVersion オブジェクトおよび VersionTracedComponentDocVersion オブジェクト）を抽出できます。

（8） XML 文書

XML 文書（文書およびバージョン付き文書）を抽出できます。

1.5.2 ファイル環境（オブジェクトエクスポート）

オブジェクトエクスポートを実行するために必要なファイル，オブジェクトエクスポートが生成するファイルおよびエラー情報やトレース情報が取得されるファイルについて説明します。

（1） オブジェクトエクスポートを実行するために必要なファイル

オブジェクトエクスポートを実行するために必要なファイルについて説明します。なお，各ファイルの記述形式や文法などについては，「5. オブジェクトエクスポートで使用するファイル」を参照してください。

（a） 制御ファイル

制御ファイルは，オブジェクトローダの制御情報と，ユーザが追加したクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加したプロパティの並びを記述するテキストファイルです。このファイルはオブジェクトローダへの入力ファイルとしても使用します。

（b） 定義ファイル

定義ファイルは，GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルの DataMapping セクションにはこのファイルで定義した名称を記述します。このファイルはオブジェクトローダへの入力ファイルとしても使用します。

（c） オブジェクト指定ファイル

DocumentBroker の文書空間から取得するオブジェクトの条件を記述するテキストファイルです。このファイルで指定できる条件式のクラスは次のとおりです。

- コンテナおよびコンテナ間の関連（DCR だけ）
- バージョンなし文書
- バージョン付き文書
- 独立永続化クラス

（d） 環境変数ファイル（Windows の場合）

Windows の場合，オブジェクトエクスポートが動作するために必要な情報を設定するファイルです。

UNIX の場合，このファイルはありません。

(2) オブジェクトエクスポートが生成するファイル

(a) オブジェクトローダ入力データファイル

オブジェクトローダ入力データファイルは，オブジェクトエクスポートが実行時の出力ファイルとして生成されるファイルです。

このファイルは DocumentBroker のデータベースに登録されたプロパティの値が制御ファイルで記述したプロパティの並び順に出力されるテキストファイルです。

(3) エラー情報およびトレース情報を取得するファイル

オブジェクトエクスポートを実行したときに発生するエラー情報やトレース情報を取得するファイルについて説明します。各ファイルの詳細については，「7. トラブルシュート機能」を参照してください。

(a) トレースログファイル

オブジェクトエクスポート実行中の重要な保守情報をトレースログに出力します。主にオブジェクトエクスポートの保守に使用します。

(b) エラーデータファイル

オブジェクトの抽出に失敗した場合，失敗したオブジェクト指定ファイルの条件式をエラーデータファイルに出力します。エラーデータファイルを修正後，次のオブジェクトエクスポート実行時の入力データファイルとして使用できます。

(c) エラーログファイル

エラーログファイルは，オブジェクトエクスポート実行時のエラー，ワーニング，インフォメーション情報を出力するファイルです。制御ファイルにエラーログファイルの出力を指定した場合，指定のファイルに出力されます。

1.6　DocumentBroker Object Loader で使用できる文字コード種別

ここでは，DocumentBroker Object Loader で使用できる文字コード種別について説明します。

DocumentBroker Object Loader で使用する文字コード種別は，DocumentBroker サーバの文書空間で使用する文字コード種別に従います。そのため，DocumentBroker Object Loader では，文字コード種別を設定する必要はありません。

DocumentBroker サーバの文書空間では，次のどちらかの文字コード種別を使用できます。一つの文書空間では複数の文字コード種別を使用できません。

- Shift-JIS
- UTF-8（使用できる文字コードの範囲は UCS-2 または UCS-4 です）

DocumentBroker サーバの文書空間で使用する文字コード種別の設定および運用方法については，マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」を参照してください。

(1) 注意事項

● ユーザが作成するファイルの文字コード種別には，文書空間で使用する文字コード種別および ASCII コードを使用するか，ASCII コードだけを使用してください。
文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。

● 文書空間の文字コード種別として UTF-8 を使用する場合，次に示す項目の最大長は 255 バイトになります。なお，バイト数は，文字列を UTF-8 で表現したときの長さです。
- コンテントのファイル名
- ファイル名から生成されるプロパティ値
- コンテントの絶対パス

また，次に示す項目は印刷可能な ASCII コードで記述してください。
- 文書オブジェクトのコンポーネントタイプ
- コンテント格納先ベースパス
- コンテント格納先パス

2 実行環境の設定

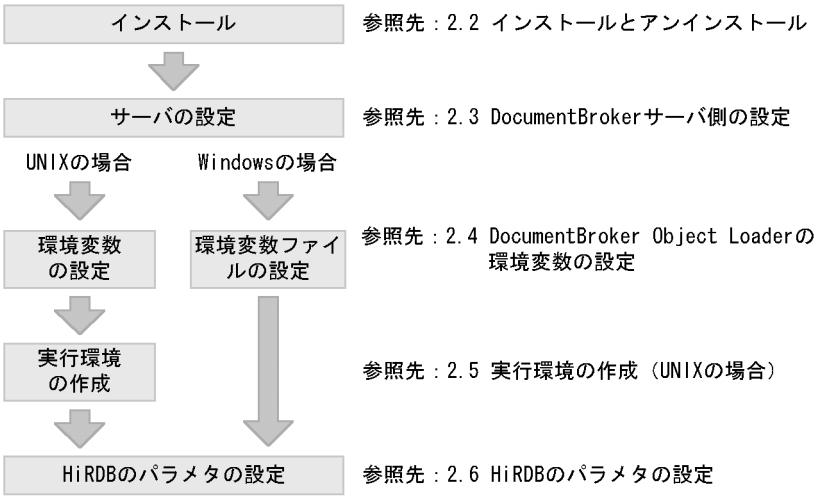
この章では，DocumentBroker Object Loader の実行環境の設定について説明します。

2.1　実行環境設定の流れ

ここでは，DocumentBroker Object Loader の実行環境の設定順序について説明します。

DocumentBroker Object Loader の実行環境を設定する順序を，図 2-1 に示します。

図 2-1　実行環境設定の流れ



説明：

1. **インストール後の環境を確認します。**
   インストール後に作成されるディレクトリと提供されるファイルを確認します。
2. DocumentBroker **サーバ側の環境を設定します。**
   DocumentBroker サーバ側で必要な環境設定をします。
3. UNIX **の場合，**DocumentBroker Object Loader **側の環境変数を設定します。**
   Windows **の場合，環境変数ファイルを設定します。**
4. UNIX **の場合，**DocumentBroker Object Loader **の実行環境を作成します。**
   実行環境のセットアップコマンド（/opt/HiEDMS_Lod/bin/EDMLodSetup）を実行します。Windows の場合，この作業は不要です。
5. HiRDB **のパラメタを設定します。**

2.2　インストールとアンインストール

ここでは，DocumentBroker Object Loader のインストールとアンインストールについて説明します。

2.2.1　インストール

ここでは，DocumentBroker Object Loader のインストール方法について説明します。インストール方法は，UNIX の場合と Windows の場合で異なります。

（1）インストール（UNIX の場合）

UNIX の場合のインストール方法について説明します。

インストールディレクトリは /opt/HiEDMS_Lod です。なお，デバイススペシャルファイル名や CD-ROM のマウントディレクトリは使用する環境によって異なります。詳細は，OS のマニュアルを参照してください。また，インストールの手順は，プログラムの提供媒体によって異なります。

（a）DAT の場合

提供媒体が DAT の場合のインストール方法について次に示します。

＜操作＞

1. root 権限でログインします。
2. DocumentBroker Object Loader のテープを DAT ドライブにセットします。
3. tar xf /dev/rmt/0m を実行します。
   「/dev/rmt/0m」の部分は使用環境によって異なります。使用環境に合わせてファイル名を変更してください。
4. /etc/hitachi_setup -i /dev/rmt/0m を実行します。
5. 初期画面が表示されます。
6. 「I）Install Software」を選択します。
7. インストールするプログラムにカーソルを移動させて，スペースキーで選択します。
   選択したプログラムの左側に＜@＞が付きます。
8. 「I）Install」を選択してから，最下行に表示されるメッセージに対して y または Y を入力します。
   インストールが始まります。
9. インストールが終了したら，「Q）Quit」を選択して終了します。

（b）CD-ROM の場合

提供媒体が CD-ROM の場合のインストール方法について次に示します。

＜操作＞

1. root 権限でログインします。
2. **マウント用の** /cdrom **ディレクトリを作成します。**mkdir /cdrom **を実行します。**
   「/cdrom」の部分には，CD-ROM をファイルシステムとしてマウントするディレクトリ名を指定してください。
3. DocumentBroker Object Loader **の** CD-ROM **を** CD-ROM **ドライブにセットします。**
4. CD-ROM **をマウントします。**
   AIX の場合，mount -r -v cdrfs /dev/cd0 /cdrom を実行します。
   なお，「/dev/cd0 /cdrom」の部分は使用環境によって異なります。使用する CD-ROM デバイススペシャルファイル名を指定してください。
5. **セットアッププログラムを実行します。**
   AIX の場合，/cdrom/aix/setup /cdrom を実行します。
   すでに CD-ROM セットアッププログラムを実行し，日立 PP インストーラがハードディスク上にある場合は，直接日立 PP インストーラ「/etc/hitachi_setup -i /cdrom」を起動してください。
   なお，「/cdrom」の部分は使用環境によって異なります。使用する CD-ROM のマウントディレクトリ名を指定してください。
6. **「I）Install Software」を選択します。**
7. **インストールするプログラムにカーソルを移動させて，スペースキーで選択します。**
   選択したプログラムの左側に＜@＞が付きます。
8. **「I）Install」を選択してから，最下行に表示されるメッセージに対して** y **または** Y **を入力します。**
   インストールが始まります。
9. **インストールが終了したら，「Q）Quit」を選択して終了します。**

（2）インストール（Windows の場合）

Windows の場合のインストール方法について説明します。

＜操作＞

1. Administrators **グループのユーザでログインします。**
2. **インストールを実行する前にすべての** Windows **アプリケーションを終了させてください。**
3. **統合** CD-ROM **を** CD-ROM **ドライブにセットし，統合** CD-ROM **の手順に従ってインストールしてください。**
   インストール説明のダイアログが表示されます。
4. **「次へ」ボタンをクリックします。**
   ユーザ情報として「名前」と「会社名」を入力するダイアログが表示されます。
   上書きインストールの場合には，上書き確認ダイアログが表示されます。ここで「はい」ボタンをクリックすると「9. プログラムフォルダを指定します」から実行されます。

5. **「名前」と「会社名」を入力します。**
6. **「次へ」ボタンをクリックします。**
   インストール先のディレクトリを指定するダイアログが表示されます。
7. **インストール先ディレクトリを指定します。**
   指定したディレクトリの下にインストールされます。デフォルトのインストール先は C:¥Program files¥HITACHI¥DocBroker_Lod です。C:¥ は，OS がインストールされているドライブ名です。
8. **「次へ」ボタンをクリックします。**
   プログラムフォルダを指定するダイアログが表示されます。
9. **プログラムフォルダを指定します。**
   指定したプログラムフォルダにプログラムアイコンが追加されます。
10. **「次へ」ボタンをクリックします。**
    現在の設定（ユーザ情報，インストール先のディレクトリ，プログラムフォルダ）確認のダイアログが表示されます。
11. **設定を確認して，「次へ」ボタンをクリックします。**
    インストールが開始されます。
    インストールが終了すると，インストールが終了したことを通知するダイアログが表示されます。
12. **「完了」ボタンをクリックします。**
    インストールプログラムを終了します。
13. **Windows を再起動します。**

注意事項

- DocumentBroker Object Loader は，次の製品と同じマシン上に同じユーザでインストールできません。
  ・DocumentBroker Object Loader Version 2
  ・ラビニティ ECM Object Loader
  ・Collaboration - File Sharing Object Loader
  すでに該当する製品がインストールされている場合，次のどれかの方法でインストールしてください。
  ・ユーザアカウントを変更してからインストールする
  ・該当する製品をアンインストールしてからインストールする
  ・別のマシンにインストールする
- 文書空間の文字コード種別が UTF-8 の場合，インストール先ディレクトリには，ASCII コードで表せるパスを指定してください。
- 上書きインストールの場合，インストールディレクトリの下の ¥etc，¥tmp および ¥spool は上書きされません。

2.2.2 インストールディレクトリと提供ファイルの内容

（1）インストールディレクトリの内容（UNIX の場合）

DocumentBroker Object Loader のインストールが終了したあとにディレクトリが正し

く作成されているか確認します。

DocumentBroker Object Loader をインストールしたあとのインストールディレクトリ下の構成を図 2-2 に示します。

図 2-2 ディレクトリ構成（UNIX の場合）

```
/opt/HiEDMS_Lod      DocumentBroker Object Loaderインストールディレクトリ
        ├── bin      DocumentBroker Object Loaderコマンド格納ディレクトリ
        ├── lib      ダイナミックリンクライブラリ格納ディレクトリ
        ├── adm      システム情報格納ディレクトリ
        ├── spool    トレースログファイル出力ディレクトリ
        └── manual   マニュアルのデータ格納ディレクトリ
```

(2) インストールディレクトリと提供ファイルの内容（Windows の場合）

(a) インストールディレクトリ

DocumentBroker Object Loader のインストールが終了したあとにディレクトリが正しく作成されているか確認します。

DocumentBroker Object Loader をインストールしたあとのインストールディレクトリ下の構成を図 2-3 に示します。

図 2-3 ディレクトリ構成（Windows の場合）

```
インストールディレクトリ
        ├── adm                システム情報ファイル格納ディレクトリ
        ├── bin                DocumentBroker Object Loaderコマンド格納ディレクトリ
        ├── etc                環境設定ファイル格納ディレクトリ
        │   └── NTconfig.ini
        ├── lib                DLL格納ディレクトリ
        ├── manual             マニュアルのデータ格納ディレクトリ
        ├── spool              トレースログファイル出力ディレクトリ
        ├── sample             サンプルの格納ディレクトリ
        │   └── NTconfig.ini
        └── tmp                一時ディレクトリ
```

(b) 提供ファイル

DocumentBroker Object Loader は，NTconfig.ini ファイルを提供します。

NTconfig.ini **ファイル**は，Windows 環境での DocumentBroker Object Loader の動作に必要な，環境変数を設定するための**環境変数ファイル**です。

NTconfig.ini ファイルは，インストールディレクトリの ¥etc の下に格納されます。

実行環境に合わせて，設定内容を変更してください。

¥sample の下にある提供ファイルは直接変更しないでください。¥sample の下の提供ファイルは，変更した ¥etc の下のファイルを初期状態（インストール直後の状態）に戻す場合に使用してください。

2.2.3 アンインストール

ここでは，DocumentBroker Object Loader のアンインストール方法について説明します。アンインストール方法は，UNIX の場合と Windows の場合で異なります。

（1）アンインストール（UNIX の場合）

UNIX の場合のアンインストール方法について説明します。

＜操作＞

1. root **権限でログインします。**
2. /etc/hitachi_setup **を実行します。**
   初期画面が表示されます。
3. **「**D）Delete**」を選択します。**
4. **アンインストールするプログラム**（DocumentBroker Object Loader）**にカーソルを移動させて，スペースキーで選択します。**
   選択したプログラムの左側に＜@＞が付きます。
5. **「**D）Delete**」を選択してから，最下行に表示されるメッセージに対して** y **または** Y **を入力します。**
   アンインストールが始まります。
6. **アンインストールが終了したら，「**Q）Quit**」を選択して終了します。**

（2）アンインストール（Windows の場合）

Windows の場合のアンインストールの方法について説明します。

Windows の場合，アンインストールは Windows「コントロールパネル」の「アプリケーションの追加と削除」を使用して実行します。

＜操作＞

1. Administrators **グループのユーザでログインします。**
2. **「コントロールパネル」の「アプリケーションの追加と削除」を実行します。**
   アプリケーションの追加と削除のプロパティが表示されます。
3. **インストールと削除タブでアンインストールするプログラム**（DocumentBroker Object Loader Version 3）**を選択して「追加と削除」ボタンをクリックします。**

注意事項

Windows の場合，アンインストール時にインストールディレクトリおよびその下に

ある ¥etc は，自動的には削除されません。必要に応じてユーザが削除してください。

2.3　DocumentBroker サーバ側の設定

ここでは，DocumentBroker サーバ側で必要な環境設定について説明します。

（1） 環境変数の設定

（a） 環境変数の設定場所

UNIX の場合

環境変数は，各サーバマシンのログインシェル環境に合わせて次のファイルに設定されています。なお，「$HOME」は，ログインユーザのホームディレクトリを示します。

- ボーンシェル環境のとき
  $HOME/.profile
- C シェル環境のとき
  $HOME/.cshrc または $HOME/.login

Windows の場合

環境変数は，「コントロールパネル」にある「システム」に設定されています。

（b） 確認する環境変数

DocumentBroker サーバがインストール済みの場合は，次の環境変数が設定されているか確認してください。

UNIX の場合

- LANG
  使用する文字コードセットを設定します。使用できる文字コードセットは，Shift-JIS，UTF-8 または ASCII です。
  使用する文字コード種別が Shift-JIS の場合
  設定する値は「Ja_JP」です。
  使用する文字コード種別が UTF-8 の場合
  使用する言語に合わせて適切な値を設定してください。
  ただし，文書空間で使用する文字コード種別に関係なく，ASCII コードのデータだけを扱う英語環境の場合は，「C」を設定してください。
  なお，この環境変数に設定する値によって，メッセージの言語種別が変わります。「Ja_JP」を設定した場合は，日本語のメッセージが出力されます。「Ja_JP」以外を設定した場合は，英語のメッセージが出力されます。
- TZ
  マシン時間のタイムゾーンを設定します。設定する値は「JST-9」です。
- DOCBROKERDIR
  DocumentBroker サーバの実行環境を作成したディレクトリを設定してください。
- PSALLOC
  環境変数「PSALLOC」には，「early」を指定してください。

- NODISCLAIM
  環境変数「NODISCLAIM」には，「true」を指定してください。

Windows の場合

- DOCBROKERDIR
  DocumentBroker サーバのインストールディレクトリが設定されていることを確認してください。

（2） XML 文書を登録する場合の認可識別子

XML 文書を登録する場合，docspace.ini に指定された認可識別子を使用します。そのため，HiRDB Text Search Plug-in の所有者を変更した場合は，変更した所有者を docspace.ini の PdTSPluginOwner に指定してください。

docspace.ini は，次のディレクトリに格納されています。

UNIX の場合

（環境変数 DOCBROKERDIR に設定したディレクトリ）/etc/docspace.ini

Windows の場合

（環境変数 DOCBROKERDIR に設定したディレクトリ）¥etc¥docspace.ini

2.4 DocumentBroker Object Loader の環境変数の設定

ここでは，DocumentBroker Object Loader の環境変数の設定について説明します。

(1) 環境変数の設定（UNIX の場合）

(a) 環境変数の設定場所

環境変数は，各サーバマシンのログインシェル環境に合わせて次のファイルに設定してください。

なお，「$HOME」は，ログインユーザのホームディレクトリを示します。

● ボーンシェル環境のとき
$HOME/.profile

● C シェル環境のとき
$HOME/.cshrc または $HOME/.login

(b) 設定する環境変数の種類

OBJLOADERDIR

環境変数「OBJLOADERDIR」には，DocumentBroker Object Loader の実行環境を作成したディレクトリを設定してください。

PATH

環境変数「PATH」には，「:（OBJLOADERDIR に設定したディレクトリ）/bin」を追加してください。

LIBPATH

環境変数「LIBPATH」には，関連プログラムのディレクトリを設定します。次の値を追加してください。

- （OBJLOADERDIR に設定したディレクトリ）/lib
- （HCCLib のインストールディレクトリ）/lib
- （HiRDB のインストールディレクトリ）/client/lib
- （HiRDB Adapter for XML のインストールディレクトリ）/lib ※1
- （DABroker のインストールディレクトリ）/lib
- （DocumentBroker サーバのインストールディレクトリ）/lib
- /opt/hitachi/common/lib
- /opt/hitachi/xpk/lib ※1
- /usr/lib ※2
- /lib ※2

注※1

プロパティマッピングをして XML 文書を登録する場合またはインデクスファイルを作成して XML 文書を登録する場合に必要です。

注※ 2

AIX の共用ライブラリのディレクトリです。

NLSPATH

環境変数「NLSPATH」には，「:（DocumentBroker Object Loader のインストールディレクトリ）/adm/%L/%N」を追加してください。

TMPDIR

XML 文書登録用の一時ファイル作成用のディレクトリを指定してください。ディレクトリは，XML 文書の文書サイズ以上の空きがある格納先を指定してください。一時ファイルについては，「2.8.1(2)(f)　一時ファイルについて」を参照してください。
オブジェクトローダ実行時，一時保管ディレクトリに一時ファイルを作成します。
一時ファイルの作成に失敗した場合は，ワーニングメッセージ（KMBV11078-W）が出力されます。文書登録後に一時ファイルは削除されます。

XMLBRKDIR

プロパティマッピングをして XML 文書を登録する場合またはインデクスファイルを作成して XML 文書を登録する場合，HiRDB Adapter for XML のインストールディレクトリを指定してください。

（2） 環境変数ファイルの設定（Windows の場合）

Windows の場合，必要に応じて，入力ファイル生成ユティリティ，オブジェクトローダおよびオブジェクトエクスポートの実行に関する情報を環境変数ファイル（NTconfig.ini）の次のパラメタに設定します。

_HIEDMS_TRACE_DIR

トレースログファイルの出力先ディレクトリを設定します。

_HIEDMS_TRACE_NUM

トレースの出力情報がトレースファイルのサイズの上限を超えた場合，トレースの出力先を別のファイルに切り替えられます。この場合の切り替えるファイル数を設定します。

_HIEDMS_TRACE_SIZE

トレースを出力するファイルのサイズを設定します。

_HIEDMS_TRACE_LEVEL

トレースの出力レベルを設定します。

TMPDIR

XML 文書登録用の一時保管ディレクトリを指定します。
これは，オブジェクトローダの実行で，必要に応じて設定するパラメタです。

入力ファイル生成ユティリティの実行での，環境変数ファイルの各パラメタの設定内容については，「3.5　環境変数ファイル（Windows の場合）」を参照してください。

オブジェクトローダの実行での，環境変数ファイルの各パラメタの設定内容については，「4.6　環境変数ファイル（Windows の場合）」を参照してください。

オブジェクトエクスポートの実行での，環境変数ファイルの各パラメタの設定内容については，「5.7　環境変数ファイル（Windows の場合）」を参照してください。

2.5　実行環境の作成（UNIX の場合）

ここでは，UNIX の場合の DocumentBroker Object Loader の実行環境の作成について説明します。Windows の場合，この作業は不要です。

DocumentBroker Object Loader の実行環境作成コマンド（/opt/HiEDMS_Lod/bin/EDMLodSetup）を作成モードで実行してください。形式は次のとおりです。

```
EDMLodSetup -m create -d 実行環境を作成するディレクトリ名
```

このコマンドを実行すると，-d で指定した実行環境を作成したディレクトリ（以降，「実行環境ディレクトリ」と呼びます）の下に，「bin」「lib」「adm」「spool」というディレクトリを作成します。そして，インストールディレクトリ下の「bin」「lib」のファイルへのリンクを実行環境ディレクトリ下の同じ名前の各ディレクトリに作成します。

また，インストールディレクトリ下の「adm」の下にあるファイルを，実行環境ディレクトリ下の同じ名前の各ディレクトリにコピーします。

実行環境作成コマンドの実行後，環境設定などに使用するファイルは，実行環境ディレクトリの下にあるファイルを使用してください。なお，実行環境作成コマンドの文法，注意事項などの詳細については，「6.　コマンドリファレンス」を参照してください。

2.6　HiRDB のパラメタの設定

ここでは，HiRDB のパラメタに設定する値について説明します。

HiRDB では，DocumentBroker に定義されたクラス（標準クラスとユーザが追加するサブクラス）分の排他資源を確保するために，pd_lck_pool_size の値を設定します。

pd_lck_pool_size に設定する値は，オブジェクトローダの実行環境および実行方法によって異なります。次の表にオブジェクトローダの実行方法による排他資源の算出方法の参照先を示します。

表 2-1　オブジェクトローダの実行方法による排他資源の算出方法の参照先

実行方法	参照先
非同時実行※（単一実行の場合）	(1)
非同時実行※（並列実行の場合）	(2)
同時実行	(3)

注※
非同時実行の場合，High-end Option をインストールしていない環境では，単一実行の場合と同じ算出方法を使用してください。なお，単一実行と並列実行の区別については，「付録 E High-end Option」を参照してください。

(1) 単一実行の場合

次の計算式で算出した値（小数点切り上げ）以上の数値を設定してください。ただし，算出した値が 45 未満の場合，45 以上の値を指定してください。デフォルト値は 1024 です。

値の算出方法

（ユーザ追加クラス数＋20）/6

(2) 並列実行の場合

トランザクションで使用する最大の排他資源を設定してください。TRANBEGIN と TRANCOMMIT を指定して，複数のコマンドを 1 トランザクションとする場合は，TRANBEGIN と TRANCOMMIT の間に記述したコマンドで使用する排他資源の総和を設定します。ただし，算出した値が 40 未満の場合，40 以上の値を指定してください。

値の算出方法

DocumentBroker サーバで設定する値に，作成コマンド（CREATE_xxx）を使用する場合の値および選択コマンド（SELECT_OBJECT）を使用する場合の値を加算して設定してください。

作成コマンド（CREATE_xxx）を使用する場合および選択コマンド

（SELECT_OBJECT）を使用する場合の算出方法について説明します。

(a) 作成コマンド（CREATE_xxx）を使用する場合

「INSERT（INSERT ～ VALUES 句指定，LOCK TABLE なし，および WITHOUT LOCK 指定なしの場合）」の条件の値に，作成するクラスの個数を加算して設定してください。排他資源数の見積もりに関する詳細については，マニュアル「HiRDB　システム定義」を参照してください。

次の表に作成するクラスの個数の算出方法を示します。

表 2-2　作成するクラスの個数の算出方法

作成コマンド名（CREATE_ は省略）	作成するクラス	個数
RFCT	ユーザ指定クラス（dmaClass_Container）	d + r + 1
	dmaClass_DirectContainmentRelationship	
	dmaClass_ReferentialContainmentRelationship	
DOC	ユーザ指定クラス（dmaClass_DocVersion）	d + r + s + 1
	dmaClass_ContentTransfer	
	dmaClass_ContentTransfers	
	dmaClass_ContentReference	
	dmaClass_DirectContainmentRelationship	
	dmaClass_ReferentialContainmentRelationship	
	dmaClass_Rendition	
	edmClass_ContentReference	
VRDOC	ユーザ指定クラス（dmaClass_ConfigurationHistory）	d + r + s + 8
	dmaClass_VersionSeries	
	dmaClass_VersionDescription	
	ユーザ指定クラス（dmaClass_DocVersion）	
	dmaClass_DocVersion	
	dmaClass_ContentTransfer	
	dmaClass_ContentTransfers	
	dmaClass_ContentReference	
	dmaClass_DirectContainmentRelationship	
	dmaClass_ReferentialContainmentRelationship	
	dmaClass_Rendition	
	edmClass_ContentReference	

作成コマンド名（CREATE_ は省略）	作成するクラス	個数
DATA	ユーザ指定クラス（edmClass_IndependentPersistence）	1
VARRAY	ユーザ指定クラス（edmClass_Struct）	1
VRCV	ユーザ指定クラス（dmaClass_ConfigurationHistory）	d＋r＋v＋5
	dmaClass_VersionSeries	
	dmaClass_VersionDescription	
	ユーザ指定クラス（edmClass_ContainerVersion）	
	edmClass_ContainerVersion	
	edmClass_VersionTraceableContainmentRelationship	
	dmaClass_DirectContainmentRelationship	
	dmaClass_ReferentialContainmentRelationship	
CV	ユーザ指定クラス（edmClass_ContainerVersion）	d＋r＋v＋1
	edmClass_VersionTraceableContainmentRelationship	
	dmaClass_DirectContainmentRelationship	
	dmaClass_ReferentialContainmentRelationship	
PACL	edmClass_PublicACL	1

（凡例）

d：dmaClass_DirectContainmentRelationship クラスの個数

r：dmaClass_ReferentialContainmentRelationship クラスの個数

s：（サブレンディションの数）× 2

v：edmClass_VersionTraceableContainmentRelationship クラスの個数

(b) 選択コマンド（SELECT_OBJECT）を使用する場合

「SELECT（LOCK TABLE なしおよび WITHOUT LOCK NOWAIT 指定ありの場合）」の条件の値に，選択するクラスの個数を加算してください。排他資源数の見積もりに関する詳細については，マニュアル「HiRDB　システム定義」を参照してください。

(3) 同時実行の場合

オブジェクトローダごとに次の計算式で算出し，その総和値以上の数値を設定してください。ただし，総和値が 40 未満の場合，40 以上の値を設定してください。

値の算出方法

単一実行をする場合および並列実行をする場合の値を算出するには，作成コマンドごとに使用する OIID 数が必要になります。

次の表に作成するコマンドごとに使用する OIID 数を示します。なお，すべてのコ

マンドに共通して，登録するユーザ定義プロパティが VariableArray 型の場合は，VariableArray **型プロパティごとの登録する最大の要素数×** 2 となります。

表 2-3　作成コマンドごとに使用する OIID 数

作成コマンド名	使用する OIID 数
CREATE_RFCT	1 + DCR + RCR
CREATE_DOC	3 + DCR + RCR + SUB_RT
CREATE_VRDOC	6 + DCR + RCR + SUB_RT
CREATE_DATA	1
CREATE_VARRAY	1
CREATE_VRCV	4 + DCR + RCR + VCR
CREATE_CV	1 + DCR + RCR + VCR
CREATE_PACL	1

（凡例）

DCR
PROP_DCR を指定する場合は ＋ 1

RCR
PROP_RCR を指定する場合は ＋（**PROP_RCR** の指定数 × 1）

VCR
PROP_VCR を指定する場合は ＋ 1

SUB_RT
PROP_SUB_RT を指定する場合は ＋（**PROP_SUB_RT** に指定したサブレンディション数 × 2）

単一実行をする場合および並列実行をする場合で算出する方法を次に説明します。

単一実行をする場合

次の値を求めて設定してください。

{「(1)　単一実行の場合」で求めた算出値} ＋ {DocumentBroker サーバの OIID 通番管理機能が使用する排他資源（使用する OIID 数／ 128 ／ 6）※}

注※　小数点切り上げ。

並列実行をする場合

次の値を求めて設定してください。

並列実行をしているオブジェクトローダごとの「単一実行をする場合」の算出値の総和。

2.7　実行環境の削除（UNIX の場合）

ここでは，DocumentBroker Object Loader の実行環境の削除について説明します。

DocumentBroker Object Loader の実行環境が不要になった場合は，環境設定コマンド（/opt/HiEDMS_Lod/bin/EDMLodSetup）を削除モード（-m delete）で実行して，指定したディレクトリを削除します。

コマンドの形式は次のとおりです。

```
EDMLodSetup -m delete -d 実行環境を削除するディレクトリ名
```

コマンドの詳細は，「6.　コマンドリファレンス」を参照してください。

2.8 DocumentBroker Object Loader の実行環境および運用に関する注意事項

ここでは，DocumentBroker Object Loader の実行環境および運用に関する注意事項について説明します。

2.8.1 オブジェクトローダに関する注意事項

(1) 実行環境に関する注意事項

(a) 実行時の権限について

オブジェクトローダの実行者には，次の権限が必要です。

- UNIX の場合，DocumentBroker サーバの管理者の権限
  Windows の場合，Administrators グループの権限
- DocumentBroker サーバが提供する docspace.ini ファイルへの READ 権限

(b) アクセス制御情報（ACL）データを登録する場合

オブジェクトローダの実行者は，あらかじめ DocumentBroker サーバが提供する docspace.ini と docaccess.ini ファイルに READ 権限を与えておく必要があります。READ 権限がなく，ファイルの参照に失敗した場合はエラーメッセージ（KMBV11004-E）を出力して終了します。

(c) HiRDB のインデクスキー値無排他の適用

HiRDB のインデクスキー値無排他を適用しない場合は，EDMLoad コマンドでデッドロックが発生することがあります。そのため，インデクスキー値無排他を適用することを推奨します。インデクスキー値無排他については，マニュアル「HiRDB　UAP 開発ガイド」を参照してください。

(2) 運用上の注意事項

(a) SELECT_OBJECT コマンドを使用する場合

SELECT_OBJECT はすでにデータベースに登録されているデータに対して有効なコマンドです。このため，SELECT_OBJECT を使用する場合，先に検索対象となる定義クラスをデータベースに登録しておく必要があります。

(b) CREATE_VRDOC，および CREATE_VRCV コマンドを使用する場合

CREATE_VRDOC，および CREATE_VRCV コマンドでバージョン付きオブジェクトを作成する場合，パブリック ACL やローカル ACL を指定すると，ConfigurationHistory オブジェクトと，DocVersion オブジェクトまたは ContainerVersion オブジェクトにパブリック ACL やローカル ACL を登録したバージョン付きオブジェクトを作成します。

(c) ACL データ登録，マルチレンディション登録機能および VTMODE 値を登録する機能を使用する場合

それぞれの機能の実行に必要なシステム定義プロパティは，制御ファイル生成コマンド（EDMCrtLCF）を使用しても制御ファイルの DataMapping セクションに自動生成されません。特に ACL データ登録機能を使用する場合は，あらかじめテキストエディタなどで，ACL を設定したいユーザクラスにシステム定義プロパティを追加しておく必要があります。

(d) XML 文書を登録する場合

XML 文書の全文検索インデクスを生成する場合，一時ファイルを作成します。ファイルの作成に失敗した場合は，ワーニングメッセージ（KMBV11078-W）を出力します。なお，XML 文書の登録処理終了後，作成された一時ファイルは削除されます。一時ファイルについては，「(f)　一時ファイルについて」を参照してください。

注意事項

PROP_XML_MAP に対応するプロパティ値として，処理コードに "OP" を指定してプロパティマッピングをする場合，または **PROP_XML_INDEX** に対応するプロパティ値として処理コードに "OP" を指定してインデクスファイルを作成する場合は，環境変数 XMLBRKDIR に HiRDB Adapter for XML のインストールディレクトリを必ず指定してください。

また，環境変数 LIBPATH（AIX の場合）には，次の値を指定してください。

- （HiRDB Adapter for XML のインストールディレクトリ）/lib
- /opt/hitachi/xpk/lib

指定していない場合，ワーニングメッセージ（KMBV11075-W）を出力して該当するコマンドの処理をスキップします。

(e) 指定できる XML 文書

XML 文書登録機能を使用する場合，指定できる XML 文書は，次に示す，DocumentBroker サーバが管理できる XML 文書だけです。

- 外部参照を含まない，単一ファイルで構成された XML ファイル。
- イメージファイルのように，構文解析の対象外の外部参照を含む XML ファイル。ただし，この場合も，XML ファイルそのものは単一ファイルで構成されている必要があります。

DocumentBroker サーバが管理できる XML 文書の詳細については，マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」を参照してください。

(f)　一時ファイルについて

オブジェクトローダ実行処理中にプロセスを強制終了させたり，サポート外の文書ファイルを読み込ませたりすると，一時ディレクトリ内に一時ファイルが残る場合があります。一時ファイルの名前は特に決まっていません。一時ファイルができる場所は，次の

順番で決定されます。

UNIX の場合

1. 環境変数 TMPDIR に設定されているディレクトリ
2. /var/tmp または /tmp

注　OS やディレクトリ容量の不足によって，一時ディレクトリが変わることがあります。

Windows の場合

1. 環境変数ファイルのパラメタ TMPDIR に設定されているディレクトリ
2. インストールディレクトリ下の ¥tmp

この一時ファイルは，データ登録処理が終われば不要になります。そのままにしておくとディスク容量が圧迫されますので，適宜削除してください。

(g) オブジェクトローダおよび DocumentBroker の実行環境

オブジェクトローダは，DocumentBroker サーバと同一のマシン上で動作することが前提となります。

オブジェクトローダの実行中は，DocumentBroker のデータベース運用コマンドの排他モードが SH のコマンド以外は使用できません。データベース運用コマンドについては，マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」を参照してください。

(h) High-end Option を使用する場合

High-end Option をインストールすると，同時に複数のオブジェクトローダを実行できます。

High-end Option を使用して，同時に複数のオブジェクトローダを実行する場合は，次の点に注意してください。

- High-end Option を使用してオブジェクトローダを実行する前に，定義ファイル，制御ファイル，入力データファイルおよび実行環境が正しく設定されていることを確認してください。
- 同時に複数のオブジェクトローダを実行する場合，それぞれの EDMLoad コマンドに対して制御ファイルを作成してください。

High-end Option の詳細については，「付録 E　High-end Option」を参照してください。

2.8.2 オブジェクトエクスポートに関する注意事項

(1) 実行環境に関する注意事項

(a) 実行時の権限について

オブジェクトエクスポートの実行者には，次の権限が必要です。

- UNIX の場合，DocumentBroker サーバの管理者の権限
  Windows の場合，Administrators グループの権限
- DocumentBroker サーバが提供する docspace.ini ファイルへの READ 権限

(b) 他機能との関係

オブジェクトエクスポートは DocumentBroker サーバを起動中，およびほかの DocumentBroker コマンド実行中でも実行できます。

(2) 運用上の注意事項

(a) 文書空間のオブジェクトをほかの文書空間に移行する場合

オブジェクトエクスポートが出力するオブジェクトローダ入力データファイルは，移行先の DocumentBroker 文書空間に合わせてカスタマイズする必要があります。移行先の DocumentBroker 文書空間が移行元の DocumentBroker 文書空間の構成（クラス定義，クラス中のプロパティの定義）と同じであれば，オブジェクトエクスポートが出力するオブジェクトローダ入力データファイルに手を加える必要はありません。

(b) 並列実行をする場合

オブジェクトエクスポートを複数のプロセスで並行して実行する場合は次の点について注意してください。

- オブジェクトローダ入力データファイル，エラーログファイル，エラーデータファイルの出力先をプロセスごとに重複しないように指定してください。
- 並列実行をして生成したオブジェクトローダ入力データファイルのデータは重複したデータが存在する可能性があるため，オブジェクトローダに入力する前に内容を確認する必要があります。重複したデータは，オブジェクトローダに入力するすべてのオブジェクトローダ入力データファイルのデータのうち，同じクラスでプロパティ値も同一のものが存在しているかどうかで確認できます。重複したデータは，必要に応じて削除してください。

(c) モデルについて

同一のクラス間を DCR で関連づけているモデルはエクスポートできません。例えば，次のようなモデルの場合です。

モデルの例

クラス B―DCR―クラス B

(d) 文書付きオブジェクトに対してオブジェクトエクスポートを実行する場合

文書付きオブジェクトに対してオブジェクトエクスポートを実行する場合，制御ファイルの DocDir エントリに指定したディレクトリ下に文書ファイルを作成します。オブジェクトエクスポートが生成した入力データファイルを使用し，オブジェクトローダを実行して，文書ファイルが不要になった場合は，ユーザが文書ファイルを削除する必要があります。

(e) VariableArray 型のデータに対してオブジェクトエクスポートを実行する場合

VariableArray 型のデータに対してオブジェクトエクスポートを実行できます。また，生成結果の入力データファイルをそのまま使用してオブジェクトローダを実行し，VariableArray 型のデータを保有するオブジェクトを登録できます。

(f) 文書クラスに対してオブジェクトエクスポートを実行する場合

オブジェクト指定ファイルに指定する文書クラス（DocVersion クラスまたは DocVersion クラスのサブクラス）が，バージョン付き文書または構成管理の対象となる文書の文書クラスを含んでいる場合，オブジェクトエクスポートの結果にバージョン付き文書または構成管理の対象となる文書のオブジェクトが出力されます。

(g) XML 文書に対してオブジェクトエクスポートを実行する場合

オブジェクトエクスポートはサブレンディションの出力はできないため，**PROP_SUB_XML** プロパティを制御ファイルの DataMapping セクションに指定しないでください。指定した場合，エラーメッセージ（KMBV11049-E）を出力してプログラムを終了します。

XML 文書登録時は，構文解析レベルやフィルタリング定義の指定によってデータベースへの登録値が異なります。しかし，登録時に指定された内容は保持されていないので，オブジェクトエクスポートでは値を仮定して抽出します。このため，エクスポートした結果を用いて，オブジェクトローダを実行する場合は，次の点に注意が必要です。

入力データファイルのプロパティの設定

オブジェクトエクスポートの結果として各プロパティに対して出力される値，およびその結果を使用して再度オブジェクトローダを実行するときの注意事項について説明します。なお，プロパティ値はオブジェクトローダ入力データファイルに出力されます。

PROP_XML_MAP **プロパティ**

- 出力内容
  ["NP"]
- 注意事項
  ["OP"] を指定してオブジェクトローダを実行した XML 文書やプロパティマッピング機能を用いて登録された XML 文書は，マッピングデータがすでにデータベースに登録されています。このような XML 文書は，プロパティマッピング定義に従ってプロパティに値が設定された状態でエクスポートさ

れます。この場合，**PROP_XML_MAP** プロパティの内容を変更する必要はありません。

PROP_XML_INDEX **プロパティ**

- 出力内容
  全文検索インデクスが登録されている場合：["OP"]
  全文検索インデクスが登録されていない場合：["NP"]
- 注意事項
  フィルタリング定義ファイルは，オブジェクトエクスポート時は出力されません。
  再度オブジェクトローダを実行する場合，フィルタリングをするときは，フィルタリング定義ファイルのパスを追加してください。形式は次のとおりです。
  ["OP"/" フィルタリング定義ファイルのパス "]
  なお，オブジェクトエクスポートは，全文検索インデクスデータを出力しません。エクスポートした結果を用いてオブジェクトローダを実行するときに再生成します。

PROP_PARSE_LEVEL **プロパティ**

- 出力内容
  ,0,
- 注意事項
  再度オブジェクトローダを実行する場合，構文解析レベルを指定するときは，必要に応じてプロパティ値を変更してください。

XML 文書以外の文書に対して再度オブジェクトローダを実行する場合

XML 文書を登録したクラスに XML 文書以外の文書が登録されている場合，すべての文書が XML 文書としてエクスポートされます。また，XML 文書以外の文書の全文検索インデクスが登録済みの場合，**PROP_XML_INDEX** プロパティの値が ["OP"] で出力されるため，再度オブジェクトローダを実行すると XML 文書として全文検索インデクスが生成されます。
このため，XML 文書以外の文書に対してオブジェクトローダを実行する場合は，DataMapping セクションの XML 文書登録のシステムプロパティ，およびそれに対する入力データファイルのプロパティ値を削除してから，再度オブジェクトロードしてください。

(h) 同じ ConfigurationHistory クラスに混在するオブジェクトに対してオブジェクトエクスポートを実行する場合

バージョン付き文書，構成管理コンテナでは，同一の ConfigurationHistory クラスを使用できます。ただし，文書クラスが混在している場合は，オブジェクトエクスポートを実行できません。例えば，バージョン付き文書と構成管理コンテナで同じ dmaClass_ConfigurationHistory クラスを使用している場合は，オブジェクトエクスポートを実行できません。異なるクラスのオブジェクト（バージョン付き文書，構成管

理コンテナ）が混在する ConfigurationHistory クラスに対してオブジェクトエクスポートを実行する場合の実行方法および注意事項については，「付録 D ConfigurationHistory クラスのエクスポート」を参照してください。

(i) オブジェクトエクスポートおよび DocumentBroker の実行環境

オブジェクトエクスポートは，DocumentBroker サーバと同一のマシン上で動作することが前提となります。

(j) マルチファイル文書のエクスポート

オブジェクトエクスポートは，マルチファイル文書に対応していません。

オブジェクトエクスポートの実行時，オブジェクト指定ファイルに指定した条件で出力されるオブジェクトにマルチファイル文書が含まれる場合，マルチファイル文書の処理はスキップされ，スキップされた文書オブジェクトの数がメッセージ（KMBV11427-I）に出力されます。マルチファイル文書以外の文書オブジェクトはエクスポートされます。

オブジェクト指定ファイルの詳細については，「5.5　オブジェクト指定ファイル」を参照してください。

(k) リファレンスファイル文書のエクスポート

オブジェクトエクスポートは，リファレンスファイル文書に対応していません。そのため，制御ファイルの DataMapping セクションのエントリに，**PROP_REF_TYPE**，**PROP_REF_BASEPATH** および **PROP_REF_PATH** は指定しないでください。指定した場合，オブジェクトエクスポートの実行時，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

オブジェクトエクスポートの実行時，オブジェクト指定ファイルに指定した条件で出力されるオブジェクトにリファレンスファイル文書が含まれる場合，リファレンスファイル文書の処理をスキップして，スキップした文書オブジェクトの数をメッセージ（KMBV11427-I）に出力します。なお，リファレンスファイル文書以外の文書オブジェクトはエクスポートされます。

オブジェクト指定ファイルの詳細については，「5.5　オブジェクト指定ファイル」を参照してください。

(l) パブリック ACL，およびローカル ACL のエクスポート

オブジェクトエクスポートは，パブリック ACL，およびローカル ACL に対応していません。そのため，制御ファイルの DataMapping セクションのエントリに **PROP_PACL**，**PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** は指定しないでください。指定した場合，オブジェクトエクスポートの実行時，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

オブジェクト指定ファイルに edmClass_PublicACL クラスのオブジェクトは指定できません。指定した場合は，エラーメッセージ（KMBV11420-E）を出力して，処理を終了

します。

パブリック ACL，およびローカル ACL が登録されたオブジェクトに対して，オブジェクトエクスポートを実行した場合，パブリック ACL，およびローカル ACL の情報のないオブジェクトをエクスポートします。

2.8.3 DocumentBroker Object Loader 共通の注意事項

- DocumentBroker Object Loader の処理は，DocumentBroker が定義するすべてのクラス，プロパティに対応する表および列がデータベースに存在することが前提です。マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」に記述してあるデータベースの初期設定の方法でデータベース環境を構築してください。
- ファイル生成機能を使用して定義ファイルを生成すると，半角空白を含むクラス名またはプロパティ名を出力することがあります。オブジェクトローダおよびオブジェクトエクスポートでは半角空白を含むクラス名またはプロパティ名を使用できませんので，半角空白を含むクラス名またはプロパティ名は半角空白を含まない名前に変更してから使用してください。
- DocumentBroker サーバの一つの文書空間に対して複数の実行環境を構築している場合，DocumentBroker Object Loader を実行できるのは，実行環境識別子が 0 の実行環境だけです。
- 日本語メッセージの挿入句やトレース情報の一部に，UTF-8 の文字列が混在して出力される場合があります。UTF-8 の文字列の内容を確認するには，UTF-8 に対応したエディタなどで参照してください。

3 ファイル生成

この章では，入力ファイル生成ユティリティの機能の概要および生成するファイルの内容について説明します。また，Windows の場合に必要に応じて設定する，環境変数ファイルの記述形式や文法について説明します。

3.1 生成できるファイルの種類

ここでは，DocumentBroker Object Loader が生成できるファイルについて説明します。

DocumentBroker Object Loader は実行時に使用するファイルの作成を支援するために，ファイル生成の機能を提供しています。生成できるファイルは次の三つのファイルです。

- **定義ファイル**
- **制御ファイル**
- **入力データファイル**

表 3-1 に生成できるファイルの種類と作成要領を示します。

表 3-1 生成できるファイルの種類と作成要領

ファイル名称	機能概要		作成要領
	セクション名	内容	
定義ファイル	ClassNameDefinition	ユーザクラスの別名と GUID 値（データベースのテーブル名の元情報）の対応を記述する。	EDMCrtLDF コマンドで作成する。
	PropNameDefinition	ユーザプロパティの別名と GUID 値（データベースの列名の元情報）の対応を記述する。	
制御ファイル	Control	DocumentBroker Object Loader の制御情報を記述する。	クラス関連ファイルを作成後，EDMCrtLCF コマンドで作成する。
	DataMapping	ユーザクラスとユーザプロパティの対応を記述する。 ユーザクラスのクラス種別によってシステム定義プロパティを追記する。	
	Export	オブジェクトエクスポート実行時に必要な制御情報を記述する。	
入力データファイル	コマンド，ラベルを記述後，制御ファイルの DataMapping に記述した定義順に入力データを記述する。		クラス関連ファイルを作成後，EDMCrtLCF コマンドで作成する。

3.2　ファイルの生成手順

DocumentBroker Object Loader でのファイル生成機能は，DocumentBroker サーバの EDMInitMeta コマンドの実行で HiRDB に生成される表（EDMS_xxx 表）とディクショナリ表を参照して，各入力ファイルを生成します。ここでは，ファイル生成の流れと，生成されるファイルの内容の概要を説明します。

（1）ファイル生成の流れ

DocumentBroker Object Loader で**定義ファイル**，**制御ファイル**，および**入力データファイル**は，DocumentBroker Object Loader が提供する EDMCrtLCF コマンドと EDMCrtLDF コマンドを実行して生成します。ファイル生成の前に DocumentBroker サーバと HiRDB 側での準備作業が必要です。次にファイル生成の流れを示します。

1. HiRDB **の** CONNECT **を行うユーザ** ID **およびパスワードを** DocumentBroker **サーバの** docspace.ini **に定義します。**
2. DocumentBroker **サーバの** EDMInitMeta **コマンドを実行します。**
3. DocumentBroker **サーバの** EDMCrtSql **コマンドで生成されたデータベース定義文で，** HiRDB **のテーブルを作成します。**
4. EDMCrtLDF **コマンドを実行して，定義ファイルを作成します。**
5. **クラス関連ファイルを作成します。**
6. EDMCrtLCF **コマンドを実行して，制御ファイルおよび入力データファイルを作成します。**
7. **必要に応じて定義ファイルの設定値を変更します。**
8. **必要に応じて制御ファイルの** Control **セクションの設定を変更・追記します。**
9. **制御ファイルの** DataMapping **セクションに必須以外のシステム定義プロパティを追記します。**
10. **作成された入力データファイルの注釈文の書式に従い，制御ファイルの** DataMapping **セクションに記述されたプロパティの並び順に登録データ値を設定します。**
11. **必要に応じて制御ファイルの** Export **セクションの設定値を変更します。**

なお，各コマンドの形式については，「6.　コマンドリファレンス」を参照してください。

また，DocumentBroker サーバのコマンドについては，マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」を参照してください。

（2）定義ファイルの生成

定義ファイルは EDMCrtLDF コマンドを実行して生成します。生成された定義ファイルはそのまま利用できますが，必要に応じて変更してください。ここでは，生成される各

セクションの概要を示します。

(a) ClassNameDefinition セクション

DocumentBroker Object Loader で規定された最上位のシステムクラスのクラス名（エントリ名）と GUID を出力後，ユーザクラス名（エントリ名）と GUID をデータベース定義（ディクショナリ）の登録順に出力します。

生成されたエントリ名を変更する場合は，制御ファイルの DataMapping セクションに生成されるユーザクラスのエントリ名も変更してください。

(b) PropNameDefinition セクション

データベース定義（ディクショナリ）に登録されたユーザクラス順で，ユーザプロパティの名称（エントリ名）と GUID をディクショナリの登録順に出力します。

GUID 値は，PROP_TYPE=guid の PROP_VALUE を用いて英文字は小文字で出力します。

生成されたエントリ名を変更する場合は，制御ファイルの DataMapping セクションに生成される列の並び定義のエントリ名も変更してください。

(3) 制御ファイルの生成

制御ファイルは EDMCrtLCF コマンドを実行して作成します。ここでは，生成される各セクションの概要を示します。

(a) Control セクション

すべてのエントリを生成します。エントリの指定値はデフォルト値になりますが，デフォルト値がないエントリはエントリ名を出力します。必要に応じて指定値を変更してください。

(b) DataMapping セクション

ユーザクラス名をデータベース定義（ディクショナリ）順に出力します。ユーザプロパティの並びはディクショナリ順に，定義ファイルに出力したエントリ名を用いてコンマ区切りで生成されます。必要に応じて指定値を変更してください。

(c) Export セクション

DocDir エントリを生成します。エントリの指定値のデフォルトは，UNIX の場合「/tmp/ObjectExport」，Windows の場合「< システムドライブ >:¥tmp¥ObjectExport」です。必要に応じて変更してください。

(4) 入力データファイルの生成

DataMapping セクションに生成したクラス数分，入力する書式を注釈文で生成します。注釈文は，シャープ（#）に続けてコマンド，ラベル，クラス名を出力後に，DataMapping セクションのプロパティの並び順にコンマ区切り形式で出力します。

3.3　生成ファイルの内容

ここでは，**生成されるファイルの内容**を説明します。

3.3.1　定義ファイル

(1) 出力形式

生成される定義ファイルは ClassNameDefinition セクションと PropNameDefinition セクションの二つのセクションで構成されています。定義ファイルの出力形式を次に示します。出力される内容については，セクションごとに説明します。

```
[ClassNameDefinition]
エントリ=GUID値<改行コード>
エントリ=GUID値<改行コード>
          :
[PropNameDefinition]
エントリ=GUID値<改行コード>
エントリ=GUID値<改行コード>
          :
```

(2) ClassNameDefinition セクション

(a) 出力内容

セクション名「[ClassNameDefinition]」を出力したあとに，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。そのあと，ClassNameDefinition セクションには DocumentBroker Object Loader で規定された最上位のシステムクラスのクラス名（エントリ名）と GUID を出力後，ユーザクラス名（エントリ名）と GUID をデータベース定義（ディクショナリ）の登録順に出力します。定義ファイルの出力形式を示します。

```
エントリ名=GUID値<改行コード>
```

生成されたエントリ名を変更する場合は，制御ファイルの DataMapping セクションに生成されるユーザクラスのエントリ名も変更してください。

(b) システムクラスのエントリ名

最上位クラスとユーザ登録先クラスのシステムクラスを出力します。表 3-2 に出力する**エントリ名とシステムクラスの対応**を示します。

表 3-2　出力するエントリ名とシステムクラスの対応

エントリ名	対応するシステムクラス
dmaClass_C	dmaClass_Container

エントリ名	対応するシステムクラス
dmaClass_CH_vrdoc	dmaClass_ConfigurationHistory
dmaClass_CD	edmClass_ComponentDocVersion
dmaClass_DV	dmaClass_DocVersion
edmClass_IP	edmClass_IndependentPersistence
edmClass_CV	edmClass_ContainerVersion
edmClass_VTDV	edmClass_VersionTracedDocVersion
edmClass_VTCDV	edmClass_VersionTracedComponentDocVersion
dmaClass_VRCH	dmaClass_ConfigurationHistory
edmClass_PublicACL	edmClass_PublicACL

(c) ユーザクラスのエントリ名

エントリ名は，EDMS_META_xxx の PROP_NAME=dmaProp_DisplayName の SECTION_NAME とします。

(d) GUID 値

「xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx（8 けた -4 けた -4 けた -4 けた -12 けた）」の形式とし，x は 0 ～ 9，a ～ f の文字とします。

(3) PropNameDefinition セクション

(a) 出力内容

セクション名「[PropNameDefinition]」を出力したあとに，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。そのあと，データベース定義（ディクショナリ）に登録されたユーザクラス順で，ユーザプロパティの名称（エントリ名）と GUID をディクショナリの登録順に出力します。定義ファイルの出力形式を示します。

エントリ名=GUID値<改行コード>

生成されたエントリ名を変更する場合は，制御ファイルの DataMapping セクションに生成される列の並び定義のエントリ名も変更してください。

(b) ユーザプロパティのエントリ名

EDMS_META_xxx の PROP_NAME=dmaProp_DisplayName の SECTION_NAME とします。

(c) GUID 値

GUID 値は，「xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx」の形式と PROP_TYPE=guid

の PROP_VALUE を用いて英文字は小文字で出力します。

x は 0 ～ 9，a ～ f の文字とします。

3.3.2 制御ファイル

(1) 出力形式

生成される制御ファイルは Control セクション，DataMapping セクションおよび Export セクションの三つのセクションで構成されています。制御ファイルの出力形式を示します。

出力される内容については，セクションごとに説明します。

```
[Control]<改行コード>
ErrorLog=ErrorLog<改行コード>
ErrorDataFile=ErrorDataFile<改行コード>
StopCount=1<改行コード>
ColumnSeparator=","<改行コード>
RecordSeparator="¥n" <改行コード>
StartPoint=1<改行コード>
MsgInterval=10000<改行コード>
OIID_Count=OIID_Count<改行コード>
XmlBroker=HAX<改行コード>
[DataMapping]<改行コード>
クラスのエントリ名=プロパティのエントリ名の並び<改行コード>
                    ⋮
                    ⋮
#UNIXの場合
[Export]<改行コード>
DocDir=/tmp/ObjectExport<改行コード>

#Windowsの場合
[Export]<改行コード>
DocDir=C:¥tmp¥ObjectExport<改行コード>
```

(2) Control セクション

(a) 出力内容

セクション名「[Control]」を出力したあとに，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。そのあと，DocumentBroker Object Loader の動作を制御する情報のすべてのエントリを出力します。エントリの設定値はデフォルト値になりますので，必要に応じて設定値を変更してください。

出力されるエントリの種類を表 3-3 に示します。各エントリの指定の詳細は「4.3.3(1) Control セクション」を参照してください。

表 3-3 Control セクションに出力されるエントリの種類

エントリ名	エントリの機能
ErrorLog	エラーログファイルの出力先を指定します。
ErrorDataFile	エラーデータファイルの出力先を指定します。
StopCount	警告レベルのエラーの許容件数を指定します。
ColumnSeparator	入力データファイル中のカラムの区切り文字（1 バイト）を引用符（"）で囲んで指定します。
RecordSeparator	入力データファイル中の行間区切り文字（1 バイト）を引用符（"）で囲んで指定します。
StartPoint	データの入力を開始する位置を，入力データファイルの行数で指定します。
MsgInterval	処理の経過を示すメッセージ（KMBV11007-I）を出力する間隔（何件ごとに出力するか）を指定します。
OIID_Count	データベースの OIID テーブルから一度に取得する OIID の個数を指定します。
XmlBroker	XML 文書を登録する場合に指定します。

制御情報の出力形式を示します。

```
[Control]<改行コード>
ErrorLog＝ErrorLog<改行コード>
ErrorDataFile=ErrorDataFile<改行コード>
StopCount=1<改行コード>
ColumnSeparator=","<改行コード>
RecordSeparator="¥n" <改行コード>
StartPoint=1<改行コード>
MsgInterval=10000<改行コード>
OIID_Count=OIID_Count<改行コード>
XmlBroker=HAX<改行コード>
```

(3) DataMapping セクション

(a) 出力内容

- セクション名「[DataMapping]」を出力したあとに，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。そのあと，定義ファイルの ClassNameDefinition セクションに定義したクラス名とプロパティの対応関係の定義をユーザクラスの個数分だけ出力します。次の形式で表と表の列の並びの定義を出力します。

  クラスのエントリ名=プロパティのエントリ名の並び<改行コード>

- ユーザクラスの種別に応じて，必須のシステム定義プロパティをユーザプロパティの並びの前に出力します。
- 最上位クラスと関連を持つクラスの中で，ユーザプロパティが指定可能なクラスがあ

る場合（例えば，バージョン付き文書の場合の最上位クラスは dmaClass_ConfigurationHistory クラス，ユーザプロパティを指定できるクラスは dmaClass_DocVersion クラス），最上位クラスとユーザクラスとを関連づける情報（クラス関連ファイル）を作成してください。その情報を取り込むことによって，「登録先クラス＠ユーザプロパティ」をユーザプロパティの並びのあとに出力します。

- エントリ名を変更する場合は，定義ファイルの ClassNameDefinition セクションに定義したエントリ名も変更してください。また，入力データファイルに生成される注釈文の登録クラス名も変更しておいてください。
- ユーザプロパティの並びのエントリ名を変更する場合は，定義ファイルの PropNameDefinition セクションに定義したエントリ名も変更してください。エントリ名はそれぞれのセクション内でユニークな名称を設定してください。
- 入力するデータの並びが，データベース定義（ディクショナリ）の列順と異なる場合には，DataMapping セクションのユーザプロパティの並び順を入力するデータの並び順に変更するか，入力するデータの並び順を変更してください。

(b) クラスのエントリ名

定義ファイルの ClassNameDefinition セクションのエントリ名です。

(c) プロパティのエントリ名の並び

プロパティのエントリ名をコンマ（,）で区切って出力します。プロパティのエントリ名には「システム定義プロパティ」,「ユーザプロパティ」,「ユーザ登録先クラス＠ユーザプロパティ」および「VariableArray 型プロパティ . 繰り返しデータ（HiRDB Array 列）プロパティ」があります。

● システム定義プロパティ

PROP_xxx 形式のシステムで定義しているプロパティ名です。クラスの種別を判別して表 3-4 に示すプロパティを出力します。

表 3-4　出力するシステム定義プロパティ

クラス	DocumentBroker サーバが ACL 非対応の場合	DocumentBroker サーバが ACL 対応の場合（ACL 対応の場合に追加出力されるプロパティ）
dmaClass_Container	出力プロパティなし	**PROP_ACL_OID**
dmaClass_DocVersion	**PROP_RTYPE**, **PROP_CT**	**PROP_ACL_OID**
dmaClass_ConfigurationHistory（CREATE_VRDOC 用）	**PROP_RTYPE**, **PROP_DOC_CLASS**, **PROP_CT**	**PROP_ACL_OID**
edmClass_IndependentPersistence	出力プロパティなし	**PROP_ACL_OID**
edmClass_Struct	**PROP_OBJECT**	出力プロパティなし
dmaClass_ConfigurationHistory（CREATE_VRCV 用）	**PROP_CV_CLASS**	**PROP_ACL_OID**

クラス	DocumentBroker サーバが ACL 非対応の場合	DocumentBroker サーバが ACL 対応の場合（ACL 対応の場合に追加出力されるプロパティ）
edmClass_ContainerVersion	出力プロパティなし	**PROP_ACL_OID**
edmClass_PublicACL	対象外※	**PROP_ACL_SUBJECT**, **PROP_ACL_STYPE**, **PROP_ACL_PERM**, **PROP_ACL_OID**

注※

DocumentBroker サーバの設定でアクセス制御機能に対応していない場合，クラスのエントリ自体が出力されません。

● **ユーザプロパティ**

ユーザプロパティは定義ファイルの PropNameDefinition セクションのエントリ名になります。

● **ユーザ登録先クラス @ ユーザプロパティ**

ユーザ登録先クラスは定義ファイルの ClassNameDefinition セクションのエントリ名になり，ユーザプロパティは定義ファイルの PropNameDefinition セクションのエントリ名になります。

● VariableArray **型プロパティ** . 繰り返しデータ（HiRDB Array 列）

VariableArray 型プロパティは定義ファイルの ClassNameDefinition セクションのエントリ名になり，繰り返しデータ（HiRDB Array 列）は定義ファイルの PropNameDefinition セクションのエントリ名になります。

（4） Export セクション

(a) 出力内容

セクション名「[Export]」を出力したあとに，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。そのあと，EDMExport コマンド実行時に使用するエントリを出力します。エントリの設定値はデフォルト値になりますので，必要に応じて設定値を変更してください。出力されるエントリの種類を表 3-5 に示します。

表 3-5　Export セクションに出力されるエントリの種類

エントリ名	エントリの機能
DocDir	文書をエクスポートする時に，データベースから抽出した文書を保管するためのディレクトリパスを指定します。 デフォルトでは，UNIX の場合「DocDir=/tmp/ObjectExport」，Windows の場合「DocDir=< システムドライブ >:¥tmp¥ObjectExport」が出力されます。

出力形式の例を次に示します。

```
#UNIXの場合
[Export]<改行コード>
DocDir=/tmp/ObjectExport<改行コード>

#Windowsの場合
[Export]<改行コード>
DocDir=C:¥tmp¥ObjectExport<改行コード>
```

3.3.3 入力データファイル

(a) 出力内容

制御ファイルの DataMapping セクションに出力したエントリの数だけ，コメントの形式で入力する書式を出力します。一つのエントリの書式を出力したあとには，改行コード（UNIX の場合は（0x0a），Windows の場合は（0x0d0a））を出力します。入力データファイルの出力形式を示します。

```
#コマンド,ラベル,クラス名,プロパティ値の並び<改行コード>
```

(b) コマンド

「command」を出力します。入力ファイルを作成するときには，このカラムには「command」の代わりに，実際に使用するコマンド名を記述します。

(c) ラベル

「label」を出力します。入力ファイルを作成するときには，ラベル指定が必要であれば，このカラムに「label」の代わりに，実際に使用するラベル名を記述します。

(d) クラス名

定義ファイルの ClassNameDefinition セクションのエントリ名を出力します。

(e) プロパティ値の並び

DataMapping セクションの列エントリ名の並びと同じ内容を出力します。

3.4 生成例・定義例

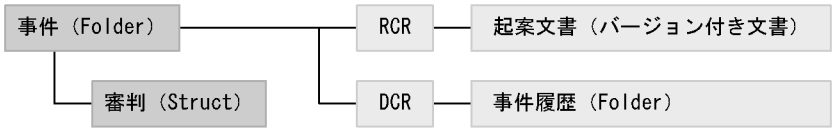
ここでは，定義ファイルの**生成例**と，クラス関連ファイルの**定義例**によって生成される制御ファイル，入力データファイルの生成例を示します。

3.4.1 オブジェクト構成例

生成例・定義例の元になる，オブジェクト構成例および使用するオブジェクトとクラスを図 3-1 に示します。

図 3-1 オブジェクト構成例および使用するオブジェクトとクラス

オブジェクト構成例



オブジェクトとクラス

オブジェクト		クラス
Jiken		dmaClass_Container
shinpan		edmClass_Struct
kian		dmaClass_ConfigurationHistory
	DocVersion	dmaClass_DocVersion
Jiken_rireki		dmaClass_Container

オブジェクト構成の説明

起案文書（バージョン付き文書）の最上位クラスは dmaClass_ConfigurationHistory クラスのサブクラスの kian で，登録先クラスはシステム標準クラスの dmaClass_DocVersion クラスになります。

3.4.2 定義ファイルの生成例

定義ファイルの生成例を示します。

```
[ClassNameDefinition]
dmaClass_C=01a3a8ca-7aec-11d1-a31b-0020af9fbb1c
dmaClass_DV=01a3a8c2-7aec-11d1-a31b-0020af9fbb1c
dmaClass_CH_vrdoc=01a3a8c6-7aec-11d1-a31b-0020af9fbb1c
Class_Jiken=76fa6abe-0406-11d2-b29a-0060b0ea4840
Class_Kian=76fa595c-0406-11d2-b29a-0060b0ea4840
Class_Jiken_rireki=f5d8e3a0-284c-11d2-9177-0000e2130367
shinpan =bb6830c1-0b53-11d2-9a68-0000e20838eb
USER_CV =bb6830c1-0b53-11d2-9a68-0000e20838ed
USER_VRCH =bb6830c1-0b53-11d2-9a68-0000e20838ee
```

```
[PropNameDefinition]
Author=76fa162c-0406-11d2-b29a-0060b0ea4840
JudgeNumber=76fa3440-0406-11d2-b29a-0060b0ea4840
etc=76fa5222-0406-11d2-b29a-0060b0ea4840
shinpan_kubun=ded7b340-88b2-11d2-963f-00c04fbc3de9
shinpan_no=df0050c0-9fe6-11d2-9640-00c04fbc3de9
                        :
```

3.4.3 クラス関連ファイルの定義例

クラス関連ファイルの定義例を示します。

```
[VRDOC]
#最上位クラスはユーザクラス，登録先クラス（DocVersion）はシステム標準クラス
Class_Kian/dmaClass_DV

[VRCV]
USER_VRCH/USER_CV
```

3.4.4 制御ファイルの生成例

制御ファイルの生成例を図 3-2 に示します。ここでは，Windows の場合の例を示します。UNIX の場合，ディレクトリの区切り文字が「/」になります。

図 3-2 制御ファイルの生成例

```
[Control]
ErrorLog=ErrorLog
ErrorDataFile=ErrorDataFile
StopCount=1
ColumnSeparator=","
RecordSeparator="¥n"
StartPoint=1
MsgInterval=10000
EmptyValue=0
XmlBroker=HAX

[DataMapping]
Class_Jiken=Author,etc
Kian=**PROP_CT**,**PROP_RTYPE**,**PROP_DOC_CLASS**,**PROP_RCR**,JudgeNumber
shinpan =**PROP_OBJECT**,shinpan_kubun,shinpan_no
Class_Jiken_rireki=**PROP_DCR**,shinpan_kubun,shinpan_no
Class_Kian=usr_Array.Author,usr_Array.etc
USER_VRCV=**PROP_CV_CLASS**,USER_CV@Author

[Export]
DocDir=C:¥tmp¥ObjectExport
```

（凡例）

二重線：オブジェクト構成によって，ユーザが変更する部分

太　線：任意の値であるため，ユーザの実行環境に合わせて変更する部分

破　線：デフォルト値であるため，ユーザの実行環境に合わせて変更する部分

3.4.5 入力データファイルの生成例

入力データファイルの生成例を示します。

```
#command,label,Class_Jiken,Author,etc

#command,label,
Kian,**PROP_CT**,**PROP_RTYPE**,**PROP_DOC_CLASS**,
JudgeNumber
#command,label,shinpan,**PROP_OBJECT**,shinpan_kubun,shinpan_no
#command,label,Class_Jiken_rireki,shinpan_kubun,shinpan_no

#command,label,USER_VRCV,**PROP_CV_CLASS**,USER_CV@Author
```

3.5 環境変数ファイル（Windows の場合）

Windows の場合，ファイル生成で必要な情報を環境変数ファイル（NTconfig.ini）に設定します。

UNIX の場合，このファイルはありません。

3.5.1 記述形式

環境変数ファイルの記述形式を次に示します。

```
[ConfigObjLoader]
_HIEDMS_TRACE_DIR = トレースログファイルの
                    出力先ディレクトリのパス名 <改行コード>
_HIEDMS_TRACE_NUM = トレースログファイルの
                     ファイル数<改行コード>
_HIEDMS_TRACE_SIZE = トレースログファイルの
                      ファイルサイズ<改行コード>
_HIEDMS_TRACE_LEVEL = トレースレベル<改行コード>
```

3.5.2 記述規則

環境変数ファイルの記述規則を次に示します。

規則

- 各定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字列の後ろに半角空白またはタブを含めることができます。
- 使用できる改行コードは（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 文字列の途中に半角空白を含めることはできません。
- 同じエントリが複数存在する場合は最上位の指定が有効になります（ワーニングメッセージ（KMBV20101-W）が出力されます）。
- [ConfigObjLoader] セクションのあとに記述されたエントリが有効になります。それ以外の場所に記述された文字列はコメントとして扱われます。
- 指定できるエントリ以外の文字列はコメントとして扱われます。
- NTconfig.ini にアクセス権限がない場合，NTconfig.ini ファイルがない場合，またはパス名が 256 バイト以上の場合は，エラーメッセージ（KMBV20102-E）を出力して処理を終了します。

注意事項

インストール直後の NTconfig.ini の初期状態では，パラメタの設定内容はコメント行になっています。

3.5.3 環境変数ファイルに設定するパラメタ

環境変数ファイルに設定するパラメタを次に示します。必要なパラメタを NTconfig.ini ファイルに記述してください。

（1） _HIEDMS_TRACE_DIR

形式

```
_HIEDMS_TRACE_DIR = トレースログファイルの出力先ディレクトリのパス名
```

設定内容

- トレースログファイルの出力先ディレクトリのパス名を指定します。指定したディレクトリの下の ¥loader がトレースログファイルの出力先となります。
- このパラメタを省略した場合，インストールディレクトリ下の ¥spool¥loader が出力先ディレクトリになります。
- _HIEDMS_TRACE_DIR に指定したディレクトリが存在しない場合，または 256 文字以上の場合には，ワーニングメッセージ（KMBV20100-W）を出力し，インストールディレクトリ下の ¥spool¥loader を使用します。

（2） _HIEDMS_TRACE_NUM

形式

```
_HIEDMS_TRACE_NUM = トレースログファイルのファイル数
```

設定内容

- トレースの出力情報がトレースファイルのサイズの上限を超えた場合，トレースの出力先を別のファイルに切り替えられます。この場合の切り替えるファイル数を設定します。
- 設定できるファイルの数は 2 ～ 16 の値です。デフォルトは 2 です。
- 設定したファイル数までファイルが切り替わると，そのあとは最初に使用したファイルから順に上書きされます。

（3） _HIEDMS_TRACE_SIZE

形式

```
_HIEDMS_TRACE_SIZE = トレースログファイルのファイルサイズ
```

設定内容

トレースを出力するファイルのサイズを 4096 ～ 2147483647（バイト）で設定します。デフォルトは 1048576（1 メガバイト）です。

（4） _HIEDMS_TRACE_LEVEL

形式

```
_HIEDMS_TRACE_LEVEL = トレースレベル
```

設定内容

トレースの出力レベルを設定します。出力レベルには，0 または 10 が設定できます。デフォルトは 10 です。
それぞれを設定した場合に出力される情報を次に示します。

0 を設定した場合

- エラー情報
- サーバの開始と終了

10 を設定した場合

- トレースレベルが 0 の場合に出力される情報
- ユーザインターフェースの情報
- 他プログラムとのインターフェースの情報
- データベースへの接続と切断

4 オブジェクトローダで使用するファイル

この章では，オブジェクトローダを実行する場合に必要な，制御ファイル，定義ファイルおよび入力データファイルの記述形式や文法について説明します。また，Windows の場合に必要に応じて設定する，環境変数ファイルの記述形式や文法について説明します。

4.1 使用するファイルの一覧

ここでは，オブジェクトローダの実行時に使用するファイルについて説明します。オブジェクトローダの実行時に使用するファイルの一覧を表 4-1 に示します。

表 4-1 オブジェクトローダで使用するファイル

ファイル名称	機能	作成場所
制御ファイル （ファイル名は任意）	オブジェクトローダの制御情報と，ユーザが追加するサブクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを記述するテキストファイルです。	任意
定義ファイル （ファイル名は任意）	GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルや入力データファイルには，定義ファイルに記述した名称を記述します。	任意
入力データファイル （ファイル名は任意）	DocumentBroker のデータベースに登録するプロパティの値を，制御ファイルで記述したプロパティの並び順に記述するテキストファイルです。複数のオブジェクトの登録を 1 トランザクションで実行させる，トランザクション単位の記述ができます。	任意
Windows の場合 環境変数ファイル （NTconfig.ini）	Windows 環境でのオブジェクトローダの動作に必要な環境変数を設定するためのファイルです。このファイルは，DocumentBroker Object Loader をインストールする際に作成されます。ファイルの設定内容はカスタマイズできます。 UNIX の場合，このファイルはありません。	インストールディレクトリ¥etc

4.2 ファイル形式と文法

ここでは，オブジェクトローダを実行する場合に必要な，制御ファイル，定義ファイルおよび入力データファイルの記述形式や文法について説明します。

なお，これらのファイルを事前に生成するための機能を提供しています。ファイル生成についての詳細は，「3. ファイル生成」を参照してください。

4.2.1 ファイル形式

オブジェクトローダを実行する場合に必要なファイルの形式は次のとおりです。

- テキストファイル
- 使用できる文字コード種別は，文書空間で使用する文字コード種別および ASCII コードか，ASCII コードだけです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。

4.2.2 文法

制御ファイルおよび定義ファイルは，幾つかのセクションとセクションを構成するエントリで構成されます。セクション名およびエントリ名は英字の大文字，小文字を区別します。したがって，セクション名およびエントリ名は各ファイルの説明に記述してあるとおりの文字列を記述してください。

(1) セクション

セクションを記述する場合は，次の文法に従います。

- セクション名は角括弧（[ ]）で囲みます。
- 角括弧とセクション名の間には，半角空白（0x20）およびタブ（0x09）以外の文字を入れないでください。

(2) エントリ

セクションを構成するエントリを記述する場合は，次の文法に従います。

- エントリは「**エントリ = 値**」の形式で，1 行で記述します。
- 「=」の前後には，半角空白（0x20）およびタブ（0x09）を挿入できます。
- 行間区切り文字は，<EOF>，<CR>，<LF>，<CR>+<LF> です。
- 1 行は，半角空白および行間区切り文字を含めて 2,048 バイト以内で記述します。

(3) パスの指定方法

パスは，次の規則に従って指定してください。パスとは，ディレクトリ名の文字列にファイル名の文字列を追加した文字列を表します。

(a) UNIX の場合

指定規則

- 指定できる文字列は，UNIX で作成できるディレクトリ名やファイル名の文字列です。
- ディレクトリの区切り文字には，「/」を使用します。
- 指定できるパスの長さは 255 バイト以内です。パス名は大文字と小文字を区別して記述してください。
- パスは，ディレクトリ名の先頭文字がディレクトリ指定文字のスラント（/）で始まる場合は絶対パス，それ以外の文字で始まる場合は相対パスとみなされます。
- パスに特殊文字を含む場合は，必ず引用符（"）でパスを囲んでください。
- 特殊文字の定義は UNIX に従います。例えば，空白を含むディレクトリ名を作成する場合は，ディレクトリ作成コマンドに続いてディレクトリ名を引用符（"）で囲みます。このように UNIX で扱える文字のうちで，引用符（"）で囲んで扱う必要のある文字を特殊文字にします。

指定例

パスの指定例を次に示します。

```
/TMP/InputDataFileName.txt                ←絶対パスの指定
./InputDataFileName.txt                   ←相対パスの指定
InputDataFileName.txt                     ←相対パスの指定
"Doc Broker/DATA/InputDataFileName.txt"   ←パスに特殊文字を含む場合
```

(b) Windows の場合

指定規則

- 指定できる文字列は，Windows で作成できるディレクトリ名やファイル名の文字列です。
- ディレクトリの区切り文字には，「¥」を使用します。
- 指定できるパスの長さは 255 バイトです。大文字と小文字は区別されません。
- パスは，ディレクトリ名が「＜ドライブ指定文字＞:」で始まる場合は絶対パス，それ以外の文字で始まる場合は相対パスとみなされます。
- パスに特殊文字を含める場合は，必ず引用符（"）でパスを囲んでください。
- 特殊文字の定義は Windows での定義に従います。例えば，空白を含むディレクトリ名を作成する場合は，ディレクトリ作成コマンドに続いてディレクトリ名を引用符（"）で囲みます。このように Windows で扱える文字のうちで，引用符（"）で囲んで扱う必要のある文字を特殊文字にします。

指定例

パスの指定例を次に示します。

```
C:¥TMP¥InputDataFileName.txt                ←絶対パスの指定
.¥InputDataFileName.txt                     ←相対パスの指定
InputDataFileName.txt                       ←相対パスの指定
"Doc Broker¥DATA¥InputDataFileName.txt"     ←パスに特殊文字を含む場合
```

(4) その他

- 先頭1カラムがシャープ（#）の行はコメント行になります。
- 引用符（"）で囲んだ文字列中にカラム間区切り文字が含まれている場合，カラム間区切り文字として設定する文字と引用符（"）の間に空白およびタブを記述しないでください。コンマ（,）をカラム間区切り文字に設定している場合の指定例を次に示します。

```
,"ABC,DEF" ,    ←正しい指定例
,△"ABC,DEF",   ←誤った指定例（データ中のコンマ（,）がカラム間区切り文字と認識されます）

（凡例）
    △：空白またはタブ
```

4.3 制御ファイル

ここでは，制御ファイルについて説明します。

制御ファイルはオブジェクトローダの制御情報と，ユーザが追加するクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを記述するテキストファイルです。

4.3.1 記述形式

制御ファイルの記述形式を次に示します。

```
[Control]
エントリ名 = 設定値<改行コード>
エントリ名 = 設定値<改行コード>
エントリ名 = 設定値<改行コード>
        ：
[DataMapping]
クラス名 = プロパティ名[,プロパティ名…]<改行コード>
クラス名 = プロパティ名[,プロパティ名…]<改行コード>
        ：
```

4.3.2 記述規則

制御ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- セクション名や各エントリの定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字間に半角空白またはタブを含むことができます。
- ＜改行コード＞は UNIX の場合は（0x0a），Windows の場合は（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- Control セクションは，1 行につき 2,048 バイト以内で記述してください。
  DataMapping セクションは，1 行につき 8,192 バイト以内で記述してください。
  なお，1 行の長さには半角空白，行間区切り文字を含みます。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。

4.3.3 制御ファイルの構成

制御ファイルは，二つのセクションとセクションを構成する幾つかのエントリで構成されます。

表 4-2 に制御ファイルに記述するセクションと機能の概要を示します。

表 4-2 制御ファイルのセクションと機能概要

セクション名	機能概要
Control	入力データファイルの記述形式やオブジェクトローダの動作を制御する情報を定義する。
DataMapping	ユーザが追加するクラスに対するプロパティの並びや，DocumentBroker 標準クラスにユーザが追加するプロパティの並びを定義する。

(1) Control セクション

DocumentBroker Object Loader の動作を制御する情報を指定するセクションです。
Control セクションに指定するエントリは次のとおりです。

(a) ErrorLog エントリ

機能

エラーログファイルの出力先をファイルパスで指定します。

指定形式

```
ErrorLog = 出力先のファイルパス
```

指定規則

- ファイルの出力先をファイルパスで指定します。パスの指定方法については，「4.2.2(3) パスの指定方法」を参照してください。
- このエントリの記述を省略した場合，またはエントリに指定する値を省略した場合は，エラーログファイルは出力されません。
- パスの指定方法が間違っている場合，またはファイルに権限がない場合は，エラーメッセージを出力して，オブジェクトローダを終了します。また，指定したエラーログファイルがすでに存在する場合は，既存のファイルを上書きします。
表 4-3 に指定値に対するオブジェクトローダの動作を示します。

表 4-3 ErrorLog 指定値に対するオブジェクトローダの動作

ErrorLog	状態	動作
省略時	―	エラーログファイルは出力しない
指定時	パス不正	エラー（オブジェクトローダ実行不可能）
	ファイル権限なし	エラー（オブジェクトローダ実行不可能）
	ファイルがすでに存在	上書きでエラーログファイルを出力する

(b) ErrorDataFile エントリ

機能

エラーデータファイルの出力先をファイルパスで指定します。

指定形式

```
ErrorDataFile = 出力先のファイルパス
```

指定規則

- エラーデータファイルの出力先をファイルパスで指定します。パスの指定方法については，「4.2.2(3) パスの指定方法」を参照してください。
- このエントリの記述を省略した場合，またはエントリに指定する値を省略した場合は，エラーメッセージを出力して，オブジェクトローダを終了します。
- パスの指定方法が間違っている場合，またはファイルに権限がない場合は，エラーメッセージを出力して，オブジェクトローダを終了します。また，指定したエラーデータファイルがすでに存在する場合は，既存のファイルを上書きします。表 4-4 に指定値に対するオブジェクトローダの動作を示します。

表 4-4 ErrorDataFile 指定値に対するオブジェクトローダの動作

ErrorDataFile	状態	動作
省略時	－	エラー（オブジェクトローダ実行不可能）
指定時	パス不正	エラー（オブジェクトローダ実行不可能）
	ファイル権限なし	エラー（オブジェクトローダ実行不可能）
	ファイルがすでに存在	上書きでエラーデータファイルを出力する

(c) StopCount エントリ

機能

警告レベルのエラーの許容件数を指定します。エラー発生件数が，このエントリで指定した値になった場合は，オブジェクトローダの実行を停止します。停止する場合，実行されていない入力データはエラーデータファイルに出力しません。

指定形式

```
StopCount = 許容件数
```

指定規則

- 警告レベルのエラーの許容件数は，0 ～ 2147483647 の間の 10 進表記で指定してください。範囲以外の値を指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 許容件数の指定を省略したとき「1」が仮定されます。
- 許容件数に「0」を指定した場合はオブジェクトローダは停止しません。

(d) ColumnSeparator エントリ

機能

入力データファイル中の**カラム間区切り文字**（1 バイト）を引用符（"）で囲んで指定します。

指定形式

```
ColumnSeparator = "区切り文字（1バイト）"
```

指定規則

- 区切り文字に，「[」，「]」，「/」，「=」，「'」，「"」，「.」，数値（0 ～ 9），「#」，「{」，「}」，「|」，「+」，「-」は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 区切り文字には，RecordSeparator エントリおよび LineContinue エントリと同じ文字は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 指定を省略したときは，コンマ（,）が仮定されます。
- 使用できるエスケープシーケンス文字は，「¥」またはバックスラッシュに続く 1 文字で表します。指定可能なエスケープシーケンス文字は，次のとおりです。
  "¥n"（復帰改行）
  "¥t"（水平タブ）
  "¥f"（改ページ）
  "¥¥"（文字の ¥）

指定例

ColumnSeparator エントリの指定例を表 4-5 に示します。

表 4-5　ColumnSeparator エントリの指定例

指定例	動作
ColumnSeparator="¥n"	復帰改行コードを区切り文字にする
ColumnSeparator="¥a"	指定できない文字なのでエラーになる
ColumnSeparator="*"	アスタリスク（*）を区切り文字にする
ColumnSeparator=*	引用符（"）で囲んでいないため，エラーになる

(e) RecordSeparator エントリ

機能

入力データファイル中の**行間区切り文字**（1 バイト）を引用符（"）で囲んで指定します。

指定形式

```
RecordSeparator = "区切り文字（1バイト）"
```

指定規則

- 区切り文字に，「[」，「]」，「/」，「=」，「'」，「"」，「.」，数値（0 ～ 9），「#」，「{」，「}」，「|」，「+」，「-」は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 指定を省略したときは，「"¥n"」（復帰改行：UNIX の場合は 0x0a，Windows の場合は 0x0d0a）が仮定されます。
- 使用できるエスケープシーケンス文字は，「¥」またはバックスラッシュに続く 1 文字で表します。指定可能なエスケープシーケンス文字は，次のとおりです。

"¥n"（復帰改行）
"¥t"（水平タブ）
"¥f"（改ページ）
"¥¥"（文字の ¥）

- 区切り文字には，ColumnSeparator エントリおよび LineContinue エントリと同じ文字は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- テキストエディタによってはデータの終端（[EOF]）の前に復帰改行コード（¥n）が入る場合があります。「RecordSeparator="¥n"」以外を設定している場合は，上記の（¥n）は行間区切り文字とみなされず通常の文字と同じ扱いになりますので注意してください。

指定例

RecordSeparator エントリの指定例を表 4-6 に示します。

表 4-6　RecordSeparator エントリの指定例

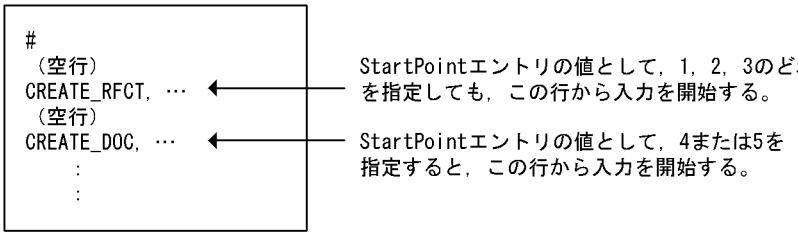
指定例	動作
RecordSeparator="¥n"	復帰改行コードを区切り文字にする
RecordSeparator="¥a"	指定できない文字なのでエラーになる
RecordSeparator="*"	アスタリスク（*）を区切り文字にする
RecordSeparator=*	引用符（"）で囲んでいないため，エラーになる

(f)　StartPoint エントリ

機能

データの入力を開始する位置を，入力データファイルの行数で指定します。
StartPoint エントリの値と入力データファイルの入力開始位置について図 4-1 に示します。

図 4-1　StartPoint エントリの値と入力データファイルの入力開始位置



指定形式

```
StartPoint = 入力データファイル内の行数
```

指定規則

- データの入力を開始する位置を，入力データファイルの行数で指定します。行数

には，コメント行や空行も含めます。
- データの入力開始位置は，1 ～ 2147483647 の間の 10 進表記で指定してください。
- 省略時は，「1」が仮定されます。
- 数値以外を指定した場合や，範囲以外の数値を指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 指定した行数が入力データファイルに存在しない場合も，エラーメッセージを出力してオブジェクトローダを終了します。

（g）OIID_Count エントリ

機能

データベースの OIID テーブルから一度に取得する OIID の個数を指定します。
次の二つの条件をすべて満たす場合にだけ有効なエントリです。
- High-end Option をインストールしている環境である
- オブジェクトローダの並列実行をする（"-S" オプション，"-M" オプションを指定しない）

これら二つの条件を満たさない場合は，指定しても無視されます。

指定形式

```
OIID_Count = OIIDの個数
```

指定規則
- OIID の個数は，0 ～ 2147483647 の間の 10 進表記で指定してください。
- 数値以外を指定した場合や，範囲以外の数値を指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 「0」を指定した場合，オブジェクトは登録されないで，OIID テーブルも更新されません。この場合，オブジェクトローダの終了時に，使用した OIID の個数がメッセージ（KMBV11005-I）に出力されます。
- オブジェクトを登録する前に，OIID_Count エントリの指定を「0」にしてオブジェクトローダを実行して，OIID の個数を取得したあと，取得した OIID の個数を OIID_Count エントリに指定してオブジェクトローダを実行することをお勧めします。必要な OIID の個数より指定値が大きい場合，オブジェクトに対応しない不要な OIID ができるためです。
- OIID_Count エントリが有効である場合，OIID_Count エントリを省略すると，作成コマンド（CREATE_xxx）ごとに OIID が取得されます。

（h）MsgInterval エントリ

機能

処理の経過を示すメッセージ（KMBV11014-I）を出力する間隔を指定します。

指定形式

```
MsgInterval = 行数
```

指定形式

- 指定する行数はオブジェクトローダ入力データファイルのコマンド行数に相当します。0 ～ 2147483647 の間の 10 進表記で指定してください。
- 省略時は，「10000」が仮定されます。
- 0 を指定した場合はメッセージを出力しません。
- 数値以外や範囲外の数値が指定された場合は，省略時の値を仮定します。
- 指定値を小さくする程，処理の経過を示すメッセージが頻繁に出力されますので，登録するデータ行に応じて適切な値を設定してください。

(i) LineContinue エントリ

機能

入力値を複数の行（行は RecordSeparator エントリで指定した文字）で記述する場合の，**ライン継続文字**（1 バイト）を引用符（"）で囲んで指定します。

指定形式

```
LineContinue = "区切り文字（1バイト）"
```

指定規則

- 区切り文字には，ColumnSeparator エントリおよび RecordSeparator エントリと同じ文字は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 区切り文字に，「[」，「]」，「/」，「=」，「'」，「"」，「.」，数値（0 ～ 9），「#」，「{」，「}」，「|」は指定できません。指定した場合は，エラーメッセージを出力してオブジェクトローダを終了します。
- 入力値を複数の行で記述する場合には，ライン継続文字の直後に RecordSeparator エントリの指定値（行間区切り文字）が存在する必要があります。直後に行間区切り文字が存在しない場合は，ライン継続文字で指定した文字は入力値の文字とみなされます。
- エントリは指定していても値を記述していない場合には「+」が仮定されます。エントリ省略時は入力値を複数の行で記述することはできません。
- エスケープシーケンス文字は，「¥」またはバックスラッシュに続く 1 文字で表します。指定可能なエスケープシーケンス文字は，次のとおりです。
  "¥n"（復帰改行）
  "¥t"（水平タブ）
  "¥f"（改ページ）
  "¥¥"（文字の ¥）

指定例

- LineContinue エントリの指定例を表 4-7 に示します。

表 4-7 LineContinue エントリの指定例

指定例	動作
LineContinue="+"	「+」をライン継続文字にする。
LineContinue="¥a"	指定できない文字なのでエラーになる。
LineContinue="*"	アスタリスク（*）をライン継続文字にする。
LineContinue=*	引用符（"）で囲んでいないため，エラーになる。

- 入力値を複数の行で記述する場合の入力データファイルの記述例を ColumnSeparator：','，RecordSeparator：'¥n'，LineContinue：'+' の場合で示します。

```
CREATE_RFCT,LABEL-C,JIKEN-CONTAINER,AAA,+(¥n)
BBB,CCC,…
```

(j) EmptyValue エントリ

機能

入力データファイル中のプロパティ値（MVARCHAR 型）中に，「"0x00"」または「0x00」が存在する場合，データベースへ格納する「¥0」値の種別を指定します。

指定形式

```
EmptyValue = （0または1）
```

指定規則

- データベースに格納する文字を 0x00 にする場合は「0」を，空文字（長さ 0 バイトの文字列）を格納する場合は「1」を指定します。
- 省略時は，「EmptyValue=0」が仮定されます。
- 「0」または「1」以外の値を設定した場合にはエラーリターンします。
- 「0x00」はバイナリコードのため，入力データファイルへの設定はバイナリコードのエディタ（xi など）およびアプリケーションプログラムで行ってください。テキストエディタ（vi など）での入力はできません。
- バイナリコードのエディタ・アプリケーションプログラムなどで設定された入力データファイルの「0x00」はテキストエディタでは見えません。

指定例

EmptyValue エントリの指定例を表 4-8 に示します。

表 4-8 EmptyValue エントリの指定例

指定例	動作
EmptyValue=0	データベースのカラム属性が MVARCHAR 型のプロパティ値として「0x00」または「"0x00"」が存在する場合，データベースのカラム値としてデータ長 1 バイトの値「0x00」を設定する。

指定例	動作
EmptyValue=1	データベースのカラム属性が MVARCHAR 型のプロパティ値として「0x00」または「"0x00"」が存在する場合，データベースのカラム値として空文字（長さ 0 バイトの文字列）を設定する。
EmptyValue= 上記以外	エラーメッセージ（KMBV11020-E）を出力して終了する。

注　データベースのカラム設定値について
0x00　：データ長 1 バイトの文字列。データ内容は「0x00」。
空文字：データ長 0 バイトの文字列。データ内容は不定。

(k) XmlBroker エントリ

機能

XML 文書を登録する場合に指定するエントリです。

指定形式

```
XmlBroker = HAX
```

指定規則

- 「HAX」を指定します。
- このエントリの記述は省略できません。
- XmlBroker エントリに指定がない場合，または「HAX」以外の値を指定した場合は，ワーニングメッセージ（KMBV11075-W）を出力し，そのオブジェクトを無視して処理を続行します。

指定例

XmlBroker エントリの指定例を表 4-9 に示します。

表 4-9　XmlBroker エントリの指定例

指定例	動作
XmlBroker=HAX	XML 文書を HiRDB Adapter for XML で登録します。
XmlBroker= 上記以外	ワーニングメッセージ（KMBV11075-W）を出力し，そのオブジェクトを無視して処理を続行します。

(2) DataMapping セクション

機能

定義ファイルの ClassNameDefinition セクションで定義したクラスとプロパティの対応関係を定義するセクションです。

指定形式

エントリは定義ファイルの ClassNameDefinition セクションで定義したクラスの最上位クラスごとに次の形式で記述します。

クラス名=プロパティ名[,プロパティ名…]

プロパティ名::={プロパティ名 ｜ VariableArray型プロパティ名 ｜ システム定義プロパティ名}

指定規則

- プロパティ名には，定義ファイルの PropNameDefinition セクションで定義したプロパティ名を指定してください。クラス関連ファイルで関連を定義している場合は，「登録先クラス名 @」をプロパティ名に修飾して指定してください。
- VariableArray 型プロパティ名の場合，edmClass_Struct クラスのサブクラスを要素とするプロパティを指定するときは，「edmClass_Struct サブクラス名 .」をプロパティ名に修飾して指定してください。
- システム定義プロパティ名は，システム定義のプロパティを「**PROP_xxx**」という形式で指定してください。指定できるシステム定義プロパティについては，「4.5.5(6)　プロパティ値の指定」を参照してください。

指定例

DataMapping セクションの指定例を図 4-2 に示します。

図 4-2　DataMapping セクションの指定例

```
[DataMapping]
VERDOC=V-AUTHOR,DOC@TITLE,USR_ARRAY.NAME,**PROP_RTYPE**,**PROP_CT**,**PROP_DCR**
         1        2          3
```

1：登録先クラスが省略されたユーザ定義プロパティなので，エントリ名に指定した最上位クラス「VERDOC」に格納されます。
2：VERDOC下のdmaClass_DocVersionクラスに格納されます。定義ファイルのClassNameDefinitionセクションにDOCを定義しておく必要があります。
3：VERDOCクラスの，USR_ARRAYプロパティを示すVariableArray型のNAMEプロパティに格納されます。定義ファイルのClassNameDefinitionセクションにUSR_ARRAYを定義しておく必要があります。

4.3.4　制御ファイルの記述例

制御ファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

```
[Control]
ErrorLog    = "DocBroker¥errorlog"
ErrorDataFile = "DocBroker¥errordata"
StopCount =50
ColumnSeparator = ","
RecordSeparator = "¥n"
StartPoint =1
OIID_Count = 50
MsgInterval = 10000
LineContinue = "+"
XmlBroker = HAX
```

```
[DataMapping]
JIKEN=C-AUTHOR,C-TITLE,C-DATE
KIAN=C-AUTHOR,C-TITLE,C-DATE,C-COMMENT,**PROP_DCR**
PAT-CONFIG=D-AUTHOR,D-TITLE,D-COMMENT,**PROP_RTYPE**,
           **PROP_DOC_CLASS**,**PROP_CT**,**PROP_DCR**
PAT-DOC=D-AUTHOR,D-TITLE,D-DATE,D-COMMENT,D-STATE,
        **PROP_RTYPE**,**PROP_CTYPE**,**PROP_DCR**
MEMO=D-TITLE,D-DATE,**PROP_RTYPE**,**PROP_CTYPE**,
     **PROP_RTYPE**,**PROP_CTYPE**,**PROP_CT**,**PROP_RCR**
I-DATA= I-AUTHOR, I-ARRAY
V-ARRAY=**PROP_OBJECT**,SET-INDEX,SET-DATA
USER_VRCH=**PROP_CV_CLASS**,**PROP_VCR**,**PROP_VTMODE**
USER_CV=**PROP_VCR**,**PROP_VTMODE**
USER_CH=**PROP_DOC_CLASS**,**PROP_RTYPE**,**PROP_CT**
XML-CONFIG1=E-PREPARER,XML-PREPARE_DATE,XCSDV@XP1,XCSDV@P1,
            **PROP_DOC_CLASS,**PROP_XML_MAP**,
            **PROP_XML_INDEX**,**PROP_PARSE_LEVEL**,
            **PROP_CT**,**PROP_RTYPE**,**PROP_DCR**
XML-CSDV=F-TITLE,XML-AUTHOR,**PROP_XML_MAP**,
         **PROP_XML_INDEX**,**PROP_PARSE_LEVEL**,
         **PROP_CT**,**PROP_RTYPE**,**PROP_DCR**
XML-DOC=XML-RELEASE_DATE,**PROP_XML_MAP**,
        **PROP_XML_INDEX**,**PROP_PARSE_LEVEL**,
        **PROP_CT**,**PROP_RTYPE**,****PROP_DCR**
XML-CSDV2=F-TITLE,XML-AUTHOR,**PROP_XML_MAP**,
          **PROP_XML_INDEX**,**PROP_PARSE_LEVEL**,
          **PROP_CT**,**PROP_RTYPE**,**PROP_DCR**,
          **PROP_SUB_XML**,**PROP_SUB_RT**
```

4.4 定義ファイル

ここでは，定義ファイルについて説明します。

定義ファイルは，GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルや入力データファイルには，定義ファイルに記述した名称を記述します。

4.4.1 記述形式

定義ファイルの記述形式を次に示します。

```
[ClassNameDefinition]
クラス名 = GUID値<改行コード>
クラス名 = GUID値<改行コード>
          :
[PropNameDefinition]
プロパティ名 = GUID値<改行コード>
プロパティ名 = GUID値<改行コード>
          :
```

4.4.2 記述規則

定義ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- セクション名や各エントリの定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字間に半角空白またはタブを含むことができます。
- ＜改行コード＞は UNIX の場合は（0x0a），Windows の場合は（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。

4.4.3 定義ファイルの構成

定義ファイルは，二つのセクションとセクションを構成する幾つかのエントリで構成されます。表 4-10 に，定義ファイルに記述するセクションとセクションの機能について説明します。

表 4-10 定義ファイルのセクションと機能概要

セクション名	機能概要
ClassNameDefinition	制御ファイルの DataMapping セクションおよび入力データファイルに記述するクラス ID に対応するクラス名を定義する。
PropNameDefinition	制御ファイルの DataMapping セクションおよび入力データファイルで記述するプロパティ ID に対するプロパティ名を指定する。

(1) ClassNameDefinition セクション

機能

制御ファイルの DataMapping セクションおよび入力データファイルに記述するクラス ID に対応するクラス名を定義するセクションです。

指定形式

エントリは定義するクラスごとに次の形式で記述します。

```
クラス名 = GUID値
```

指定規則

- クラス ID に記述される英大文字の A ～ F と英小文字の a ～ f は同一として扱います。
- クラス名は 128 バイト以内で記述します。先頭が「**」で始まる名称は指定できません。
- クラス名には「@」,「=」,「,」,「.」および半角空白は使用できません。
- GUID 値には，クラス名に対応するクラス ID の GUID 値を指定します。
- 同一のクラスに対して複数のクラス名を定義できます。例えば，一つの文書クラスにマルチファイル文書とその他の文書を登録する場合，マルチファイル文書用のクラス名とその他の文書用の文書名を定義することで，1 回で登録できます。同一のクラスに対して複数のクラス名を定義した場合，制御ファイルおよび入力データファイルに，それぞれのクラス名に対する記述が必要です。

(2) PropNameDefinition セクション

機能

制御ファイルの DataMapping セクションおよび入力データファイルで記述するプロパティ ID に対するプロパティ名を指定するセクションです。

指定形式

エントリは定義するプロパティごとに次の形式で記述します。

```
プロパティ名 = GUID値
```

指定規則

- プロパティ ID に記述される英大文字の A ～ F と英小文字の a ～ f は同一として

扱います。

- プロパティ名は 128 バイト以内で記述します。先頭が「**」で始まる名称は指定できません。
- プロパティ名には「@」,「=」,「,」,「.」および半角空白は使用できません。
- GUID 値には，プロパティ名に対応するプロパティ ID の GUID 値を指定します。

4.4.4 定義ファイルの記述例

定義ファイルの記述例を次に示します。

```
[ClassNameDefinition]
# Containerクラス
JIKEN=01234567-8901-2345-6789-012345678901
KIAN=01234567-8901-2345-6789-012345678902
# ConfigurationHistoryクラス
PAT-CONFIG=01234567-8901-2345-6789-012345678903
# DocVersionクラス
PAT-DOC=01234567-8901-2345-6789-012345678904
# DocVersionクラス
MEMO=01234567-8901-2345-6789-012345678905
# Independentクラス
I-DATA=01234567-8901-2345-6789-012345678906
# Structクラス
V-ARRAY=01234567-8901-2345-6789-012345678908
#ComponentDocVersionクラス
COMPONENT = bb6830ca-0bf0-11d2-9a68-0000e20838e7
#構成管理用ConfigurationHistoryクラス
USER_VRCH= bb6830ab-0bf0-11d2-9a68-0000e20838e8
USER_CH= bb6830ab-0bf0-11d2-9a68-0000e20838e9
#構成管理用ContainerVersionクラス
USER_CV= bb6830ab-0bf0-11d2-9a68-0000e20838eA
#全文検索機能付きDocVersionクラス
XML-CSDV=648bb885-0000-259a-391f-8a28000c26a
XML-CSDV2=648bb885-0000-259a-391f-8a28000c26b
XCSDV = 648bb885-0000-259a-391f-8a28000c26d

[PropNameDefinition]
# フォルダ作成者，フォルダ名，作成日時，コメント
C-AUTHOR=01234567-8901-2345-6789-012345678911
C-TITLE=01234567-8901-2345-6789-012345678912
C-DATE=01234567-8901-2345-6789-012345678913
C-COMMENT=01234567-8901-2345-6789-012345678914

# 文書著者，文書名，作成日時，コメント，文書状態
D-AUTHOR=01234567-8901-2345-6789-012345678915
D-TITLE=01234567-8901-2345-6789-012345678916
D-DATE=01234567-8901-2345-6789-012345678917
D-COMMENT=01234567-8901-2345-6789-012345678918
D-STATE=01234567-8901-2345-6789-012345678919

# 作成者，データ
I-AUTHOR=01234567-8901-2345-6789-012345678920
I-ARRAY=01234567-8901-2345-6789-012345678907
```

```
# Struct型プロパティ
SET-INDEX=01234567-8901-2345-6789-012345678922
SET-DATA=01234567-8901-2345-6789-012345678923

# CSDVクラスのプロパティ
CSDV-P1=01234567-8901-2345-6789-012345678924
#プロパティマッピング用プロパティ
XML-PREPARE_DATE=01234567-8901-2345-6789-012345678924
XML-REVIEW_DATE=01234567-8901-2345-6789-012345678925
XML-RELEASE_DATE=01234567-8901-2345-6789-012345678926
XML-AUTHOR=01234567-8901-2345-6789-012345678927

#作成者,レビュー者,承認者
E-PREPARER= 01234567-8901-2345-6789-012345678928
E-REVIEWER= 01234567-8901-2345-6789-012345678929
E-ISSUER=01234567-8901-2345-6789-012345678930

#文書名, 作成日時, コメント
F-TITLE=01234567-8901-2345-6789-012345678931
F-DATE=01234567-8901-2345-6789-012345678932
F-COMMENT=01234567-8901-2345-6789-012345678933

#XCSDVクラスのプロパティ
XP1=01234567-8901-2345-6789-012345678934
P1=01234567-8901-2345-6789-012345678935
```

4.5 入力データファイル

ここでは，入力データファイルについて説明します。

入力データファイルは，DocumentBroker のデータベースに登録するプロパティの値を，制御ファイルで記述したプロパティの並び順に記述するテキストファイルです。複数のオブジェクトの登録を 1 トランザクションで実行させる，トランザクション単位の記述ができます。

4.5.1 記述形式

入力データファイルの記述形式を次に示します。

```
#コメント
TRANBEGIN<改行コード>
作成コマンド名,ラベル名,クラス名,プロパティ値1,プロパティ値2,…プロパティ値n<
改行コード>
        :
選択コマンド名,ラベル名,パラメタ<改行コード>
        :

TRANCOMMIT<改行コード>
#コメント
TRANBEGIN<改行コード>
作成コマンド名,ラベル名,クラス名,プロパティ値1,プロパティ値2,…プロパティ値n<
改行コード>
        :
選択コマンド名,ラベル名,パラメタ<改行コード>
        :

TRANCOMMIT<改行コード>
        :
```

4.5.2 記述規則

入力データファイルの記述規則を次に示します。

規則

- 使用できる文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。
- セクション名や各エントリの定義は 1 行で記述します。
- 1 文字目から記述します。
- ＜改行コード＞は UNIX の場合は（0x0a），Windows の場合は（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。

4.5.3 記述するコマンドの一覧

入力データファイルに記述できるコマンドの一覧を表 4-11 に示します。コマンドは，作成コマンド，選択コマンド，指示コマンドに分類します。コマンド名は，英字の大文字で記述します。

表 4-11 コマンド一覧

分類	コマンド名	説明
作成	CREATE_RFCT	コンテナの作成
	CREATE_DOC	文書の登録
	CREATE_VRDOC	バージョン付文書の登録
	CREATE_DATA	独立オブジェクトの登録
	CREATE_VARRAY	別表の可変長配列へのデータの登録
	CREATE_VRCV	バージョン付き構成管理コンテナの作成
	CREATE_CV	バージョンなし構成管理コンテナの作成
	CREATE_PACL	パブリック ACL の登録
選択	SELECT_OBJECT	オブジェクトの選択
指示	TRANBEGIN	トランザクションの開始の指示
	TRANCOMMIT	トランザクションの終了の指示
	#	コメント行

4.5.4 カラムの位置と機能

入力ファイル内の各行には，1 オブジェクトの登録に対するプロパティ値をカラム間区切り文字を付けて順に記述し，終端に行間区切り文字を記述します。カラム間区切り文字，行間区切り文字は，制御ファイルで指定します。カラム間区切り文字で区切られた各カラムの機能を表 4-12 に示します。

コマンドの分類によって使用するカラム数やカラムの意味が異なります。選択コマンドと指示コマンドでは，使用しないカラムに記述された文字は無視されます。

表 4-12 カラム間区切り文字で区切られた各カラムの機能

コマンド分類	カラム位置	カラムの意味	機能
作成コマンド	1	コマンドカラム	作成コマンド名の指定
	2	ラベルカラム	ラベル名の指定
	3	クラス名カラム	登録するオブジェクトのクラス名の指定
	4 以降	プロパティ値カラム	登録するクラスのプロパティ値の指定
選択コマンド	1	コマンドカラム	選択コマンド名の指定
	2	ラベルカラム	ラベル名の指定

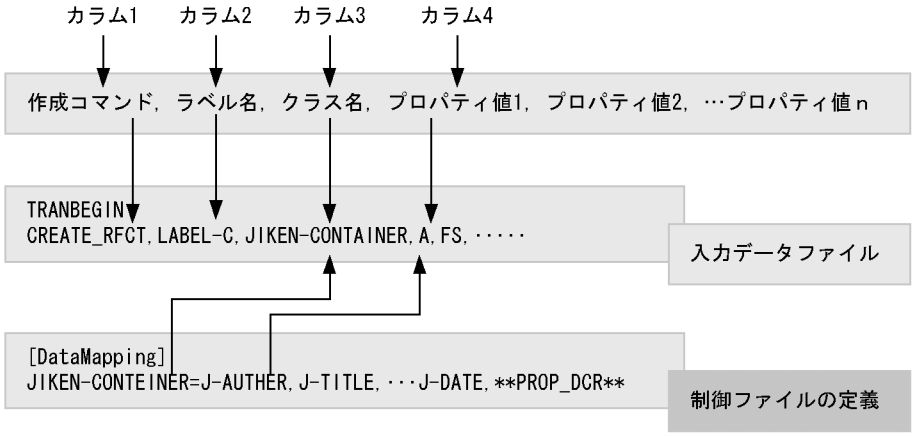
コマンド分類	カラム位置	カラムの意味	機能
	3	パラメタカラム	コマンドのパラメタの指定
指示コマンド	1	コマンドカラム	指示コマンドの指定

4.5.5　作成コマンドの記述方法

(1) 作成コマンドの記述形式

作成コマンドを入力データファイルに記述する場合の記述形式を，制御ファイルの DataMapping セクションの定義内容と対比して図 4-3 に示します。

図 4-3　制御ファイルの DataMapping セクションの定義内容と対比



- 入力データファイルのクラス名カラムには，制御ファイルの DataMapping セクションのエントリ名を記述します。
- プロパティ値カラムには，エントリの値として定義したプロパティの並び順に値を記述します。
- プロパティ値カラムに使用できる文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。
- 文書空間で使用する文字コード種別が UTF-8 の場合，次に示す項目は印刷可能な ASCII コードで記述してください。
  - 文書オブジェクトのコンポーネントタイプ
  - コンテント格納先ベースパス
  - コンテント格納先パス

(2) 作成コマンドとプロパティの関係

作成コマンドの種別によって，必ず指定するプロパティや指定できないプロパティがあ

ります。したがって，使用する作成コマンドの種別に応じて，制御ファイルの DataMapping セクションの定義に必要なシステム定義プロパティ名を記述しておかなければなりません。表 4-13 から表 4-19 に作成コマンドと制御ファイルの DataMapping セクションに指定するプロパティの関係を示します。なお，システム定義プロパティ名は，英大文字で記述してください。

表 4-13　作成コマンドと指定するプロパティの関係（1/7）

プロパティ種別	ユーザ定義	システム定義（**PROP_x** の x の部分）				
作成コマンド（CREATE は省略）		CT_PATH	RNAME	CT	RTYPE	CTYPE
RFCT	○	×	×	×	×	×
DOC（マルチファイル文書）	○	●	○	●	●	○
DOC（リファレンスファイル文書）	○	×	×	●	●	○
DOC（シングルファイル文書）	○	×	×	●	●	○
VRDOC（マルチファイル文書）	○	●	○	●	●	○
VRDOC（リファレンスファイル文書）	○	×	×	●	●	○
VRDOC（シングルファイル文書）	○	×	×	●	●	○
DATA	○	×	×	×	×	×
VARRAY	○	×	×	×	×	×
VRCV	○	×	×	×	×	×
CV	○	×	×	×	×	×
PACL	○	×	×	×	×	×

（凡例）
○：指定できる　●：必ず指定する　×：エラー

表 4-14　作成コマンドと指定するプロパティの関係（2/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	DCR	RCR	DOC_CLASS	OBJECT
RFCT	○	○複数	×	×
DOC（マルチファイル文書）	○	○複数	×	×

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	DCR	RCR	DOC_CLASS	OBJECT
DOC（リファレンスファイル文書）	○	○複数	×	×
DOC（シングルファイル文書）	○	○複数	×	×
VRDOC（マルチファイル文書）	○	○複数	●	×
VRDOC（リファレンスファイル文書）	○	○複数	●	×
VRDOC（シングルファイル文書）	○	○複数	●	×
DATA	×	×	×	×
VARRAY	×	×	×	●
VRCV	○	○複数	×	×
CV	○	○複数	×	×
PACL	×	×	×	×

（凡例）
○：指定できる　●：必ず指定する　×：エラー

表の説明
DCR は一つ指定できます。RCR は複数指定できます。

表 4-15　作成コマンドと指定するプロパティの関係（3/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	VTMODE	SUB_RT	CV_CLASS	VCR
RFCT	×	×	×	×
DOC（マルチファイル文書）	×	×	×	×
DOC（リファレンスファイル文書）	×	×	×	×
DOC（シングルファイル文書）	×	○	×	×
VRDOC（マルチファイル文書）	×	×	×	×
VRDOC（リファレンスファイル文書）	×	×	×	×
VRDOC（シングルファイル文書）	×	○	×	×

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	VTMODE	SUB_RT	CV_CLASS	VCR
DATA	×	×	×	×
VARRAY	×	×	×	×
VRCV	○	×	●	○
CV	○	×	×	○
PACL	×	×	×	×

（凡例）
○：指定できる　●：必ず指定する　×：エラー

表 4-16　作成コマンドと指定するプロパティの関係（4/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）				
作成コマンド（CREATE は省略）	ACL_OID	ACL_GID	ACL_OFLAG	ACL_GFLAG	ACL_EFLAG
RFCT	●	○	○	○	○
DOC（マルチファイル文書）	●	○	○	○	○
DOC（リファレンスファイル文書）	●	○	○	○	○
DOC（シングルファイル文書）	●	○	○	○	○
VRDOC（マルチファイル文書）	●	○	○	○	○
VRDOC（リファレンスファイル文書）	●	○	○	○	○
VRDOC（シングルファイル文書）	●	○	○	○	○
DATA	●	○	○	○	○
VARRAY	×	×	×	×	×
VRCV	●	○	○	○	○
CV	●	○	○	○	○
PACL	●	×	×	×	×

（凡例）
○：指定できる　●：必ず指定する　×：エラー

表 4-17 作成コマンドと指定するプロパティの関係（5/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	PACL	ACL_SUBJECT	ACL_STYPE	ACL_PERM
RFCT	○	○	○	○
DOC（マルチファイル文書）	○	○	○	○
DOC（リファレンスファイル文書）	○	○	○	○
DOC（シングルファイル文書）	○	○	○	○
VRDOC（マルチファイル文書）	○	○	○	○
VRDOC（リファレンスファイル文書）	○	○	○	○
VRDOC（シングルファイル文書）	○	○	○	○
DATA	○	○	○	○
VARRAY	×	×	×	×
VRCV	○	○	○	○
CV	○	○	○	○
PACL	×	●	●	●

（凡例）
○：指定できる ●：必ず指定する ×：エラー

表 4-18 作成コマンドと指定するプロパティの関係（6/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	XML_MAP	XML_INDEX	PARSE_LEVEL	SUB_XML
RFCT	×	×	×	×
DOC（マルチファイル文書）	×	×	×	×
DOC（リファレンスファイル文書）	×	×	×	×
DOC（シングルファイル文書）	○	○	○	○
VRDOC（マルチファイル文書）	×	×	×	×

プロパティ種別	システム定義（**PROP_x** の x の部分）			
作成コマンド（CREATE は省略）	XML_MAP	XML_INDEX	PARSE_LEVEL	SUB_XML
VRDOC（リファレンスファイル文書）	×	×	×	×
VRDOC（シングルファイル文書）	○	○	○	○
DATA	×	×	×	×
VARRAY	×	×	×	×
VRCV	×	×	×	×
CV	×	×	×	×
PACL	×	×	×	×

（凡例）
○：指定できる　×：エラー

表 4-19　作成コマンドと指定するプロパティの関係（7/7）

プロパティ種別	システム定義（**PROP_x** の x の部分）		
作成コマンド（CREATE は省略）	REF_TYPE	REF_BASEPATH	REF_PATH
RFCT	×	×	×
DOC（マルチファイル文書）	×	×	×
DOC（リファレンスファイル文書）	●	●	●
DOC（シングルファイル文書）	×	×	×
VRDOC（マルチファイル文書）	×	×	×
VRDOC（リファレンスファイル文書）	●	●	●
VRDOC（シングルファイル文書）	×	×	×
DATA	×	×	×
VARRAY	×	×	×
VRCV	×	×	×
CV	×	×	×
PACL	×	×	×

（凡例）
●：必ず指定する　×：エラー

(3) 作成コマンドの指定

作成コマンドの指定方法を説明します。作成コマンドと指定する最上位クラスの対応を次の表に示します。

表 4-20　作成コマンドと最上位クラスの対応

作成コマンド	最上位クラス
CREATE_RFCT	dmaClass_Container
CREATE_DOC	dmaClass_DocVersion
CREATE_VRDOC	dmaClass_ConfigurationHistory
CREATE_DATA	edmClass_IndependentPersistence
CREATE_VARRAY	ユーザが定義した edmClass_Struct クラスのサブクラス
CREATE_VRCV	dmaClass_ConfigurationHistory
CREATE_CV	edmClass_ContainerVersion
CREATE_PACL	edmClass_PublicACL

(4) ラベル名の指定

- ラベル名は 16 バイト以内の文字列で指定します。
- ラベル名にコンマ（,）を使用する場合は引用符（"）で囲んでください。
- ラベル名を記述した場合，ラベル指定したオブジェクトへリンクする場合のリンク情報を保持します。ラベル名を記述しなければリンク情報は保持しません。例えば，コンテナを作成して，そのコンテナに文書を登録する場合は，コンテナの作成時にラベル指定しておけば，文書作成時には保持したリンク情報を基に文書のオブジェクトを生成するので実行性能が向上します。
- 同じラベル名が指定された場合は，リンク情報を上書き（あとから指定されたオブジェクトへのリンク情報に書き換え）します。
- 保持するリンク情報はオブジェクトローダの実行ごとに有効です。

(5) クラス名の指定

- 制御ファイルの DataMapping セクションのエントリ名として定義したクラス名を指定します。
- 英字の大文字と小文字を区別するので，定義した名称をそのまま指定する必要があります。
- 作成コマンドと指定するクラス名のスーパークラスの整合がとれていない場合は，エラーメッセージ（KMBV11067-E）を出力して終了します。
- 作成コマンドと最上位クラスの対応を次の表に示します。

表 4-21 作成コマンドと最上位クラスの対応

作成コマンド	最上位クラス
CREATE_RFCT	dmaClass_Container
CREATE_DOC	dmaClass_DocVersion
CREATE_VRDOC	dmaClass_ConfigurationHistory
CREATE_DATA	edmClass_IndependentPersistence
CREATE_VARRAY	edmClass_Struct
CREATE_VRCV	dmaClass_ConfigurationHistory
CREATE_CV	edmClass_ContainerVersion
CREATE_PACL	edmClass_PublicACL※

注※

CREATE_PACL コマンドは，edmClass_PublicACL クラスだけを対象とします。

(6) プロパティ値の指定

- 制御ファイルの DataMapping セクションのエントリの値として定義したプロパティに対する値を，プロパティの並び順に指定します。
- 定義したプロパティがユーザ定義プロパティ（String 型，Integer32 型，Boolean 型，VariableArray 型）の場合は，データベースに定義した型（それぞれ CHAR/MVARCHAR 型，INTEGER 型，INTEGER 型，ARRAY 型）に従った値を指定します。
- ユーザ定義プロパティの値を省略した場合，制御ファイルの DataMapping セクションでユーザ定義プロパティを記述しているときは，メタデフォルト値が仮定されます。制御ファイルの DataMapping セクションからユーザ定義プロパティを削除したときは，NULL 値が登録されます。
- プロパティ値の指定方法については，「4.5.5(7) データの指定」を参照してください。

****PROP_CT_PATH** の指定方法**

マルチファイル文書を登録する場合に，**PROP_CT** に指定するファイルを格納したディレクトリを絶対パス名で指定します。記述方法を次に示します。

コンテントを持つ文書オブジェクトを登録する場合

```
"ディレクトリパス" または {"ディレクトリパス","ディレクトリパス",…}
```

コンテントを持たない文書オブジェクトを登録する場合

```
NO_DIR
```

- ディレクトリパスを引用符（"）で囲んで記述します。
- 複数のディレクトリを指定する場合，全体を波括弧（{ }）で囲みます。一つのディレクトリパスを波括弧（{ }）で囲んだ場合，ワーニングメッセージ（KMBV11052-W）を出力し，そのオブジェクトを無視して処理を続行します。
- ディレクトリパスと **PROP_CT** に指定するファイル名を合わせた長さが 255

バイト以内になるように指定してください。255 バイトを超えた場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

- 指定されたディレクトリパスにディレクトリが存在しない場合は，ワーニングメッセージ（KMBV11034-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 指定されたディレクトリに存在するサブディレクトリは無視されます。
- **PROP_CT_PATH** にディレクトリパスを指定し，**PROP_CT** および **PROP_CTYPE** にマルチファイル文書以外の場合の指定方法で指定すると，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- NO_DIR を指定した場合，コンテントを持たないマルチファイル文書のオブジェクトが登録されます。この場合，**PROP_CT** および **PROP_CTYPE** の指定は無視されます。
- 複数のディレクトリを指定した場合，**PROP_CT** には ALL_FILE を指定したと仮定されます。
- 複数のディレクトリを指定し，ディレクトリパス下のファイル名が，ほかのディレクトリ下のファイル名と重複した場合，ワーニングメッセージ（KMBV11100-W）を出力し，該当オブジェクトを無視して処理を続行します。

****PROP_RNAME** の指定方法**

マルチファイル文書の場合に，登録する文書のリトリーバルネームを指定します。リトリーバルネームとは，マルチファイル文書の文書名です。記述方法を次に示します。

"リトリーバルネーム"

- 全体を引用符（"）で囲んで記述します。
- リトリーバルネームは文字列 255 バイト以内で指定してください。255 バイトを超えた場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- **PROP_CT_PATH** の指定がない場合，**PROP_RNAME** を設定するとエラーメッセージ（KMBV11049-E）を出力し，処理を終了します。
- 制御ファイルに **PROP_RNAME** を指定しない場合，および入力データファイルでプロパティ値の指定を省略した場合，データベースの値は NULL 値になります。

****PROP_CT** の指定方法**

文書オブジェクトに格納するコンテントを指定します。

マルチファイル文書の場合

ContentTransfers オブジェクトにコピーするコンテントのファイルを指定します。**PROP_CT_PATH** に NO_DIR または複数のディレクトリを指定した場合，**PROP_CT** の指定は無視されます。記述方法を次に示します。

ファイル名を指定する場合

{"ファイル名","ファイル名","ファイル名",…}

指定したディレクトリのすべてのファイルをコピーする場合

ALL_FILE

- ファイル名を指定する場合，ファイル名を引用符（"）で囲み，制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。
- ファイル名を指定する場合，全体を波括弧（{　}）で囲みます。
- ALL_FILE を指定した場合，**PROP_CT_PATH** に指定したディレクトリに格納されたすべてのファイルがコピーされます。
- 各ファイル名と **PROP_CT_PATH** に指定するディレクトリの絶対パスを合わせた長さが 255 バイト以内になるように指定してください。255 バイトを超えた場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 指定できるファイル数の上限を超えた場合は，ワーニングメッセージ（KMBV11066-W）を出力し，該当オブジェクトを無視して処理を続行します。なお，指定できるファイル数の上限は，データベースの定義（DocumentBroker Server の EDMInitMeta の引数に指定した数）に依存します。
- 指定したファイルの合計サイズがファイルを格納する BLOB の定義長を超えた場合は，ワーニングメッセージ（KMBV11035-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 指定したファイルが存在しない場合，および指定したファイルにアクセス権限がない場合は，ワーニングメッセージ（KMBV11055-W）を出力し，該当オブジェクトを無視して処理を続行します。
- **PROP_CT_PATH** にディレクトリパスを指定した場合，**PROP_CT** に，マルチファイル文書以外の場合の指定方法で指定すると，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

リファレンスファイル文書の場合

コンテント格納先ディレクトリにコピーするコンテントの所在を絶対パス名で指定します。

- 絶対パスを 255 バイト以内で指定します。
- 入力していない場合，およびファイルに参照権限がない場合は，エラーメッセージが出力されます。
- 指定したファイルが存在しなかった場合は，NULL 値が指定されたものと仮定します。

シングルファイル文書の場合

ContentTransfer オブジェクトにコピーするコンテントの所在を絶対パス名で指定します。

- 絶対パスを 255 バイト以内で指定します。
- 入力していない場合，およびファイルに参照権限がない場合は，エラーメッセージが出力されます。
- 指定したファイルが存在しなかった場合は，NULL 値が指定されたものと仮定します。

****PROP_RTYPE** の指定方法**

Rendition オブジェクトに保持する RenditionType（コンテントの表現形式を表す文字列）を指定します。記述方法を次に示します。

```
MIME::<typename>
```

****PROP_CTYPE** の指定方法**

文書オブジェクトのコンポーネントタイプ（コンテントを識別する文字列）を指定します。文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで指定してください。

マルチファイル文書の場合

ContentTransfers オブジェクトにコピーするコンテントのコンポーネントタイプを指定します。コンポーネントタイプを設定しない場合，**PROP_CTYPE** の指定を省略してください。なお，**PROP_CT_PATH** に NO_DIR または複数のディレクトリを指定した場合，**PROP_CTYPE** の指定は無視されます。記述方法を次に示します。

{"コンポーネントタイプ","コンポーネントタイプ",…}

- コンポーネントタイプを引用符（"）で囲み，制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。
- 全体を波括弧（{　}）で囲みます。
- 指定した値は，**PROP_CT** の指定順に対応します。
- 特定のファイルのコンポーネントタイプを省略する場合，省略するコンポーネントタイプを記述しないで，次のように指定してください。次の例では，2 番目のファイルのコンポーネントタイプを省略しています。

  {" コンポーネントタイプ ",," コンポーネントタイプ ",…}

  指定を省略したファイルのコンポーネントタイプの値は，「0x00（長さ 0）」になります。
- **PROP_CYPE** の指定を省略した場合，すべてのファイルのコンポーネントタイプの値は，「0x00（長さ 0）」になります。

- 各コンポーンネントタイプの文字列長が 255 バイト以内になるように指定してください。255 バイトを超えた場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- **PROP_CTYPE** に指定したコンポーネントタイプの数が **PROP_CT** に指定したファイル数と異なる場合，ワーニングメッセージ（KMBV11036-W）を出力し，該当オブジェクトを無視して処理を続行します。
- **PROP_CT_PATH** に複数のディレクトリを指定しないで，**PROP_CT** に ALL_FILE を指定した場合，**PROP_CTYPE** を指定すると，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

リファレンスファイル文書またはシングルファイル文書の場合

ContentTransfer または ContentReference オブジェクトに保持するコンポーネントタイプを，255 バイト以内の文字列で指定します。

****PROP_DCR** の指定方法**

DirectContainmentRelationship オブジェクトを利用してリンクする親コンテナを指定します。

指定方法の詳細については，「(a)　指定方法 1」を参照してください。

****PROP_RCR** の指定方法**

ReferentialContainmentRelationship オブジェクトを利用してリンクする親コンテナを指定します。

親オブジェクトの指定方法について，一つのオブジェクトにリンクする場合は，「(a)　指定方法 1」と同じです。複数のオブジェクトにリンクする場合は，「(a)　指定方法 1」を繰り返し指定します。

****PROP_DOC_CLASS** の指定方法**

文書登録のためのオブジェクトを定義ファイルに定義した置き換え文字で指定します。指定方法を次に示します。なお，クラス名とは，定義ファイルの ClassNameDefinition セクションで定義したエントリ名です。

```
[DocVersionのクラス名]
[VersionTracedDocVersionまたはVersionTracedComponentDocVersionのクラス名]
[全文検索機能付きDocVersionのクラス名]
```

- 全体を角括弧（[　]）で囲んで記述します。
- ユーザ定義プロパティを追加しないクラスは，この指定を省略できます。すなわち，CREATE_VRDOC の場合，DocVersion のクラスにユーザプロパティがなく，標準の dmaClass_DocVersion クラスを使用する場合は，クラス名の記述を省略して，[] と指定します。
- 角括弧の前後に半角空白は指定できません。

- CREATE_VRDOC での XML 文書登録で，全文検索インデクスを作成するように指定した場合は，全文検索機能付き DocVersion のクラス名を必ず指定してください。

****PROP_OBJECT** の指定方法**

別表の可変長配列に VariableArray 型プロパティの要素を格納する場合，VariableArray 型のオブジェクトを指定します。指定方法の詳細については，「(a) 指定方法 1」を参照してください。

****PROP_VTMODE** の指定方法**

構成管理コンテナの VTMODE 値を指定します。記述方法を次に示します。

```
FIX　またはFLOAT
```

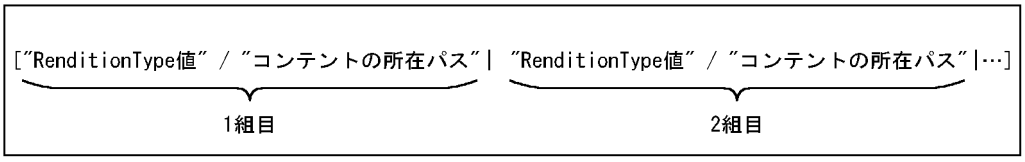
- 半角の英大文字で記述します。
- 指定内容が不正な場合はワーニングメッセージ（KMBV11052-W）を出力し登録されません。
- **PROP_VTMODE** を設定して CREATE_VRCV コマンドまたは CREATE_CV コマンドを実行する場合，**PROP_VCR** の指定がないときは，エラーメッセージ（KMBV11049-E）を出力し，処理を終了します。
- 指定値と格納結果を次の表に示します。

表 4-22　**PROP_VTMODE の指定値と格納結果**

指定値	格納結果
システムプロパティなし	1
指定値なし（,,）	1
FIX	1
FLOAT	2
「FIX」または「FLOAT」以外	KMBV11052-W のエラー

****PROP_SUB_RT** の指定方法**

マルチレンディション機能のサブレンディションとして登録する RenditionType 値とコンテントの所在パスを指定します。記述形式を次に示します。



- 全体を角括弧（[　]）で囲み，一組の RenditionType 値とコンテントの所在パスをスラント（/）区切りで指定し，最大 9 個「|」で区切って記述します。
- RenditionType 値およびコンテントの所在パスは引用符（"）で囲む必要がありま

す。

- RenditionType 値は必ず指定します。
- コンテントの所在パスは省略可能です。
- 指定内容が不正な場合はワーニングメッセージ（KMBV11052-W）を出力し登録されません。
- システムプロパティを定義するユーザクラスに指定されている「**PROP_RTYPE**」および「**PROP_CT**」の指定値は，マスタレンディションとして登録されます。
- RenditionType 値は，「**PROP_RTYPE**」に対する指定値も含めて一意になるように指定してください。
- 文書が存在しなかった場合は，NULL 値が指定されたものと仮定します。

****PROP_CV_CLASS** の指定方法**

バージョン付き構成管理コンテナを作成するためのコンテナバージョンオブジェクトを定義ファイルに定義した置き換え文字で指定します。記述方法を次に示します。なお，クラス名とは，定義ファイルの ClassNameDefinition セクションで定義したエントリ名です。

```
[ContainerVersionのクラス名]
```

- 全体を角括弧（[ ]）で囲んで記述します。
- ContainerVersion のクラスにユーザプロパティがなく，標準の edmClass_ContainerVerison クラスを使用する場合は，クラス名の記述を省略して，[] と指定します。

****PROP_VCR** の指定方法**

VersionTraceableContainmentRelationship クラスを使用してコンテナバージョンと文書を関連づけるプロパティです。バージョン付き文書を指定します。指定方法の詳細については，「(a)　指定方法 1」を参照してください。

****PROP_ACL_OID** の指定方法**

オブジェクトの所有者を指定します。1 バイト以上 n バイト以内の半角英数字で指定してください。前後に半角空白は指定できません。

DocumentBroker サーバで，ユーザ識別子の最大長を拡張していない場合，n は 254 バイトです。

ユーザ識別子の最大長を拡張している場合，n は次のどちらかに指定する値です。

- DocumentBroker サーバのメタ情報の初期設定コマンド（EDMInitMeta）の -n オプション
- DocumentBroker サーバの文書空間情報ファイルの UserIDMaxLength エントリ（文書空間の定義コマンド（EDMCDefDocSpace）を使用するとき）

ユーザ識別子の最大長の拡張の詳細は，マニュアル「DocumentBroker Version 3 システム導入・運用ガイド」を参照してください。

アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ

（KMBV11049-E）を出力して，処理を終了します。

****PROP_ACL_GID** の指定方法**

オブジェクトのグループを指定します。1 バイト以上 n バイト以内の半角英数字で指定してください。前後に半角空白は指定できません。指定しなかった場合，および「,,」を指定した場合，データベースの値は NULL 値になります。

DocumentBroker サーバで，グループ識別子の最大長を拡張していない場合，n は 254 バイトです。

グループ識別子の最大長を拡張している場合，n は次のどちらかに指定する値です。

- DocumentBroker サーバのメタ情報の初期設定コマンド（EDMInitMeta）の -g オプション
- DocumentBroker サーバの文書空間情報ファイルの GroupIDMaxLength エントリ（文書空間の定義コマンド（EDMCDefDocSpace）を使用するとき）

グループ識別子の最大長の拡張の詳細は，マニュアル「DocumentBroker Version 3 システム導入・運用ガイド」を参照してください。

アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

****PROP_ACL_OFLAG** の指定方法**

アクセス制御フラグ（ACFlag）の所有者のパーミッションを指定します。指定方法の詳細については「(b)　指定方法 2」を参照してください。

アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

****PROP_ACL_GFLAG** の指定方法**

アクセス制御フラグ（ACFlag）のグループのパーミッションを指定します。指定方法の詳細については「(b)　指定方法 2」を参照してください。

アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

****PROP_ACL_EFLAG** の指定方法**

アクセス制御フラグ（ACFlag）の全ユーザのパーミッションを指定します。指定方法の詳細については「(b)　指定方法 2」を参照してください。

アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

****PROP_PACL** の指定方法**

文書やフォルダから参照するパブリック ACL のラベルを指定します。アクセス制御機能が利用できる環境で，パブリック ACL を使用するクラスのエントリに指定します。記述形式を次に示します。

{[Label/ラベル名],[Label/ラベル名],…}

- 全体を波括弧（{　}）で囲んで記述します。
- 「Label/」と「ラベル名」を，角括弧（[　]）で囲んで記述します。
- 制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。
- **PROP_PACL** の記述例を次に示します。
  {[Label/PACL001],[Label/PACL002],[Label/PACL003]}
- 一致するラベル名がない場合は，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。また，指定したラベル名がパブリック ACL 以外のオブジェクトの場合は，ワーニングメッセージ（KMBV11057-W）を出力して，そのオブジェクトを無視して処理を続行します。
- ラベルは 10 個まで指定できます。10 個を超えて指定した場合は，ワーニングメッセージ（KMBV11066-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 同じパブリック ACL を示すラベルを複数指定した場合，一つだけ登録します。
- 波括弧（{　}）内のラベル指定を省略した場合（例えば，{[Label/PACL001],} と指定した場合），またはラベルの指定が不正な場合，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。波括弧だけを指定した場合，および「,,」を指定した場合は，エラーにはなりません。
- アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

PROP_ACL_SUBJECT

パブリック ACL，またはローカル ACL に設定するサブジェクトを指定します。サブジェクトとは，ユーザ識別子，グループ識別子，およびシステムサブジェクトのことです。
システムサブジェクトには，"self" と "everyone" があります。"self" はオブジェクトの所有者を表し，"everyone" はすべてのユーザを表します。
アクセス制御機能が利用できる環境で，ローカル ACL に対応したクラスのエントリ，または edmClass_PublicACL クラスのエントリに指定します。
edmClass_PublicACL クラスのエントリに指定した場合は，パブリック ACL になります。それ以外のローカル ACL に対応したクラスのエントリに指定した場合は，ローカル ACL になります。
記述形式を次に示します。
{"サブジェクト","サブジェクト",…}

- 全体を波括弧（{　}）で囲んで記述します。
- サブジェクトは，サブジェクトの値を引用符（"）で囲み，制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。

- **PROP_ACL_SUBJECT** の記述例を次に示します。
  {"user01","user02","group01","self"}
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** は，必ず一緒に指定してください。どれか一つでも指定していない場合は，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，同じ数だけ指定してください。値の数が一致していない場合は，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 値は 64 個まで指定できます。64 個を超えて指定した場合は，ワーニングメッセージ（KMBV11066-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 波括弧（{ }）内の値を省略した場合（例えば，{"user01",} と指定した場合），ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。波括弧だけを指定した場合，および「,,」を指定した場合は，エラーにはなりません。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，それぞれの値が指定順に対応します。
- アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- サブジェクトの値は，1 バイト以上 254 バイト以内の半角英数字で指定します。DocumentBroker サーバで，ユーザ識別子の最大長を拡張している場合は，**PROP_ACL_OID** の指定方法で説明する最大長に従ってください。グループ識別子の最大長を拡張している場合は，**PROP_ACL_GID** の指定方法で説明する最大長に従ってください。前後に半角空白は指定できません。

PROP_ACL_STYPE

パブリック ACL，またはローカル ACL のサブジェクト種別を指定します。アクセス制御機能が利用できる環境で，ローカル ACL に対応したクラスのエントリ，または edmClass_PublicACL クラスのエントリに指定します。edmClass_PublicACL クラスのエントリに指定した場合は，パブリック ACL となります。それ以外のローカル ACL に対応したクラスのエントリに指定した場合は，ローカル ACL になります。記述形式を次に示します。
{サブジェクト種別,サブジェクト種別,…}

- 全体を波括弧（{ }）で囲んで記述します。
- 制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。
- サブジェクト種別には，**PROP_ACL_SUBJECT** で指定した値に対応するサ

ブジェクト種別を設定します。

- サブジェクト種別は，対応するサブジェクトがユーザ識別子の場合は USR，グループ識別子の場合は GRP，システムサブジェクトの場合は SYS を指定します。
- **PROP_ACL_STYPE** の記述例を次に示します。
  {USR,USR,GRP,SYS}
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** は，必ず一緒に指定してください。どれか一つでも指定していない場合は，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，同じ数だけ指定してください。値の数が一致していない場合は，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 値は 64 個まで指定できます。64 個を超えて指定した場合は，ワーニングメッセージ（KMBV11066-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 波括弧（{ }）内の値を省略した場合（例えば，{USR,} と指定した場合），ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。波括弧だけを指定した場合，および「,,」を指定した場合は，エラーにはなりません。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，それぞれの値が指定順に対応します。
- アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- サブジェクト種別に SYS を指定して，対応するサブジェクトがシステムサブジェクトではない場合，ワーニングメッセージ（KMBV11102-W）を出力して，そのオブジェクトを無視して処理を続行します。
- サブジェクト種別が USR，GRP，SYS 以外の場合，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。

PROP_ACL_PERM

パブリック ACL，またはローカル ACL のパーミッションを指定します。
アクセス制御機能が利用できる環境で，ローカル ACL に対応したクラスのエントリ，または edmClass_PublicACL クラスのエントリに指定します。
edmClass_PublicACL クラスのエントリに指定した場合はパブリック ACL になります。それ以外のローカル ACL に対応したクラスのエントリに指定した場合には，ローカル ACL になります。
記述形式を次に示します。
{パーミッション,パーミッション,…}

- 全体を波括弧（{ }）で囲んで記述します。

- 制御ファイルの ColumnSeparator エントリに指定したカラム間区切り文字で区切って記述します（上記の例では，デフォルトのコンマ（,）をカラム間区切り文字にしています）。
- ACL に設定するサブジェクトのパーミッションを記述します。指定するパーミッションの値は，ACFlag のパーミッションの指定に従います。詳細については，「(b)　指定方法 2」を参照してください。
- パーミッションには，**PROP_ACL_SUBJECT** で指定した値に対応するパーミッションを設定します。
- **PROP_ACL_PERM** の記述例を，次に示します。
  {FULL_CONTROL,READ_WRITE,READ_WRITE,READ}
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** は，必ず一緒に指定してください。どれか一つでも指定していない場合は，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，同じ数だけ指定してください。値の数が一致していない場合は，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 値は 64 個まで指定できます。64 個を超えて指定した場合は，ワーニングメッセージ（KMBV11066-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 波括弧（{　}）内の値を省略した場合（例えば，{FULL_CONTROL,} と指定した場合），ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。波括弧だけを指定した場合，および「,,」を指定した場合は，エラーにはなりません。
- **PROP_ACL_SUBJECT**，**PROP_ACL_STYPE**，および **PROP_ACL_PERM** の値は，それぞれの値が指定順に対応します。
- アクセス制御機能が利用できない環境で指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。
- パーミッションの組み合わせが不正な場合は，ワーニングメッセージ（KMBV11069-W）を出力して，そのオブジェクトを無視して処理を続行します。

****PROP_XML_MAP** の指定方法**

プロパティマッピングを実行するかどうかを示すコード，マッピング定義名，および XMS ファイルの所在パスを指定します。指定方法の詳細については，「(c)　指定方法 3」および「(g)　指定方法 7」を参照してください。

****PROP_XML_INDEX** の指定方法**

インデクスファイルを作成するかどうかを指定するコード，およびフィルタリング定義ファイルの所在パスを指定します。指定方法の詳細については，「(d)　指定方法 4」および「(g)　指定方法 7」を参照してください。

****PROP_PARSE_LEVEL** の指定方法**

XML 文書の構文解析レベルを指定します。指定方法の詳細については，「(e)　指定方法 5」および「(g)　指定方法 7」を参照してください。

****PROP_SUB_XML** の指定方法**

サブレンディション（**PROP_SUB_RT**）に XML 文書を指定する場合，入力データファイルに記述された **PROP_SUB_RT** に該当するオブジェクトの何番目が XML 文書であるかを指定します。1 けたの正の整数を指定してください。指定方法の詳細については，「(f)　指定方法 6」および「(g)　指定方法 7」を参照してください。

****PROP_REF_TYPE** の指定方法**

リファレンスファイル文書を登録する場合に，パスを操作するモードを指定します。記述方法を次に示します。

NONEまたはRELATIVE

NONE

コンテントを持たないで，コンテントロケーションを管理しないリファレンスファイル文書を登録する場合に指定します。

RELATIVE

ユーザが指定した任意のディレクトリにコンテントを格納して，コンテントロケーションを相対パスで管理する場合に指定します。この場合，「コンテント格納先ベースパス（**PROP_REF_BASEPATH**）」にコンテントの格納先のベースとなるパスを指定して，「コンテント格納先パス（**PROP_REF_PATH**）」にコンテント格納先ベースパスからの相対パスを指定します。

RELATIVE を指定した場合，コンテント格納先のファイル名のフルパスは次のようになります（条件によってフルパスが異なります）。

● DocumentSpace **構成定義ファイル**（docspace.ini）**の** ReferenceStorageMode **エントリに** Origin **を指定した場合，または** DocumentBroker **サーバのバージョンが** 03-11 **以前の場合**

aaaa/bbbb/cccc/dddd

aaaa：コンテント格納先ベースパス

bbbb：コンテント格納先パス

cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（33 バイト）

dddd：**PROP_CT** に指定したコンテントのファイル名（拡張子を含む）

● DocumentSpace **構成定義ファイル**（docspace.ini）**の** ReferenceStorageMode **エントリに** Divide **を指定した場合**

aaaa/bbbb/cccc/ddddeeee

aaaa：コンテント格納先ベースパス

bbbb：コンテント格納先パス

cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（4 バイト）
dddd：DocumentBroker サーバがコンテントを管理するためのファイル名（42 バイト）
eeee：**PROP_CT** に指定したコンテントのファイル名の拡張子（ピリオドを含めて 0 ～ 17 バイト）

- **PROP_REF_TYPE** に NONE を指定した場合，**PROP_CT**，**PROP_CTYPE**，**PROP_REF_BASEPATH**，および **PROP_REF_PATH** に指定した値は無視されます。
- **PROP_REF_TYPE** に RELATIVE を指定した場合，**PROP_CT** に指定したファイルがないときは，インフォメーションメッセージ（KMBV11107-I）を出力して，コンテントを持たないで，コンテントロケーションを管理しないリファレンスファイル文書として登録します。

****PROP_REF_BASEPATH** の指定方法**

リファレンスファイル文書を登録する場合に，コンテント格納先ベースパスを指定します。文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで指定してください。記述形式を次に示します。
"コンテント格納先ベースパス名"

- コンテント格納先ベースパス名を引用符（"）で囲んで記述します。Windows の場合，コンテント格納先ベースパスには，ローカルドライブまたはネットワークドライブを指定してください。UNIX の場合，コンテント格納先ベースパスには，ローカルドライブまたは NFS（ネットワークファイルシステム）を指定してください。
- コンテント格納先のファイル名のフルパスは次のようになります（条件によってフルパスが異なります）。

  ● DocumentSpace 構成定義ファイル（docspace.ini）の ReferenceStorageMode エントリに Origin を指定した場合，または DocumentBroker サーバのバージョンが 03-11 以前の場合

  aaaa/bbbb/cccc/dddd
  aaaa：コンテント格納先ベースパス
  bbbb：コンテント格納先パス
  cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（33 バイト）
  dddd：**PROP_CT** に指定したコンテントのファイル名（拡張子を含む）

  ● DocumentSpace 構成定義ファイル（docspace.ini）の ReferenceStorageMode エントリに Divide を指定した場合

  aaaa/bbbb/cccc/ddddeeee
  aaaa：コンテント格納先ベースパス
  bbbb：コンテント格納先パス

cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（4 バイト）
dddd：DocumentBroker サーバがコンテントを管理するためのファイル名（42 バイト）
eeee：**PROP_CT** に指定したコンテントのファイル名の拡張子（ピリオドを含めて 0 ～ 17 バイト）

文書空間で使用する文字コード種別が Shift-JIS の場合，UTF-8 の場合共に，コンテント格納先のファイル名の長さが次の範囲内になるように，「コンテント格納先ベースパス」，「コンテント格納先パス」，および「コンテントのファイル名」を指定してください。

UNIX の場合

1 ～ 1,022 バイト

Windows の場合

1 ～ 259 バイト

指定できる範囲を超えた場合は，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視して処理を続行します。

- **PROP_REF_TYPE** に NONE を指定して，**PROP_REF_BASEPATH** に値を指定した場合，**PROP_REF_BASEPATH** に指定した値は無視されます。
- **PROP_REF_TYPE** に RELATIVE を指定した場合，**PROP_REF_BASEPATH** に「,,」，または「,"",」を指定すると，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- 「..」（カレントディレクトリの一つ上位のディレクトリ）は指定できません。「..」を指定した場合，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- コンテント格納先ベースパスがない場合，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視して処理を続行します。
- コンテント格納先ベースパス下にディレクトリ，およびファイルが作成できない場合，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視して処理を続行します。
- コンテント格納先ベースパスにネットワークドライブを指定する場合，事前に，ネットワークドライブの割り当ておよび接続をしてください。
  Windows Server 2008 の場合，ネットワークドライブの割り当ておよび接続を管理者権限で行い，かつ割り当ておよび接続を行った管理者権限と同一権限でオブジェクトローダを実行する必要があります。
  Windows Server 2008 での，リファレンスファイル文書登録の実行例を以下に示します。

＜実行例 1＞

1	OS ビルトインのシステム管理者（Administrator）でログオンします。

2	エクスプローラの [ツール] ― [ネットワークドライブの割り当て] で，ネットワークドライブの割り当ておよび接続を行います。
3	コマンドプロンプトで EDMLoad コマンドを実行します。

＜実行例 2 ＞

1	システム管理者グループ（Administrators グループ）に所属するユーザでログオンします。
2	コマンドプロンプトを管理者として実行します。
3	上記のコマンドプロンプトで NET USE コマンドを実行し，ネットワークドライブの割り当ておよび接続を行います。
4	ネットワークドライブの割り当ておよび接続を行ったコマンドプロンプトで EDMLoad コマンドを実行します。

- コンテント格納先ベースパスに NFS を指定する場合，コンテントの格納先となるサーバマシンと，DocumentBroker サーバが存在するマシンは同一のオペレーティングシステムを使用してください。また，パスワードファイルに設定するシステム管理者ユーザのユーザ ID，およびシステム管理者ユーザが所属するシステム管理者グループのグループ ID は，マシン間で同一の内容にしておく必要があります（同じパスワードファイルを使用することを推奨します）。
- コンテント格納先ベースパスに指定したディレクトリに対して，次のアクセス権限が必要です。アクセス権限がない場合，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視して処理を続行します。
  ・Windows の場合：変更（読み取り，書き込み，実行，および削除）権限
  ・UNIX の場合：読み取り，書き込み，および実行権

****PROP_REF_PATH** の指定方法**

リファレンスファイル文書のコンテント格納先パスを指定します。文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで指定してください。記述形式を次に示します。

"コンテント格納先パス名"

- コンテントの格納先パス名に，コンテント格納先ベースパスからの相対パスを，引用符（"）で囲んで指定します。
- コンテント格納先のファイル名のフルパスは次のようになります（条件によってフルパスが異なります）。

  ● DocumentSpace **構成定義ファイル（**docspace.ini**）の** ReferenceStorageMode **エントリに** Origin **を指定した場合，または** DocumentBroker **サーバのバージョンが** 03-11 **以前の場合**

  aaaa/bbbb/cccc/dddd

  aaaa：コンテント格納先ベースパス

  bbbb：コンテント格納先パス

  cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（33

バイト）
dddd：**PROP_CT** に指定したコンテントのファイル名（拡張子を含む）

● DocumentSpace 構成定義ファイル（docspace.ini）の ReferenceStorageMode エントリに Divide を指定した場合

```
aaaa/bbbb/cccc/ddddeeee
```

aaaa：コンテント格納先ベースパス
bbbb：コンテント格納先パス
cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（4 バイト）
dddd：DocumentBroker サーバがコンテントを管理するためのファイル名（42 バイト）
eeee：**PROP_CT** に指定したコンテントのファイル名の拡張子（ピリオドを含めて 0 ～ 17 バイト）

コンテント格納先のファイル名の長さが次の範囲内になるように，「コンテント格納先ベースパス」，「コンテント格納先パス」，および「コンテントのファイル名」を指定してください。

UNIX の場合

1 ～ 1,022 バイト

Windows の場合

1 ～ 259 バイト※

注※　指定できるパスの長さは，文書空間で使用する文字コード種別が Shift-JIS の場合，UTF-8 の場合共に，1 ～ 259 バイトです。

指定できる範囲を超えた場合は，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視して処理を続行します。

- **PROP_REF_TYPE** に NONE を指定して，**PROP_REF_PATH** に値を指定した場合，**PROP_REF_PATH** に指定した値は無視されます。
- **PROP_REF_TYPE** に RELATIVE を指定した場合，**PROP_REF_PATH** に「,,」または「,"",」を指定すると，コンテント格納先ベースパスのディレクトリ下にコンテントを登録します。
- コンテント格納先ベースパス，およびコンテント格納先パス（相対パス）を結合して，絶対パスとして使用します。パスの区切り文字は，DocumentBroker Object Loader が挿入するため，指定する必要はありません。ただし，コンテント格納先ベースパスの末尾，またはコンテント格納先パスの先頭にパスの区切り文字を指定している場合，パスの区切り文字は挿入しないで，結合して絶対パスとして使用します。
- 「..」（カレントディレクトリの一つ上位のディレクトリ）は指定できません。「..」を指定した場合，ワーニングメッセージ（KMBV11052-W）を出力して，そのオブジェクトを無視して処理を続行します。
- コンテント格納先パス下にディレクトリおよびファイルが作成できない場合，ワーニングメッセージ（KMBV11103-W）を出力して，そのオブジェクトを無視

して処理を続行します。

(a) 指定方法 1

親オブジェクト（コンテナや文書）の記述方法は次のとおりです。

[Label/**ラベル名**]

- 全体を角括弧（[　]）で囲んで記述します。
- スラント（/）の前後に半角空白（0x20）を挿入できます。
- 親オブジェクト生成時または SELECT_OBJECT 時にラベル指定をしたラベル名をそのまま記述します。ただし，一致するラベル名が存在しない場合はエラーになります。
- 指定できないラベルを指定した場合はメッセージを出力します。
- プロパティによってリンクできるオブジェクトが決まっています。プロパティ値に指定できるオブジェクト生成時のラベルを次の表に示します。

表 4-23　プロパティ値に指定できるオブジェクト生成時のラベル

プロパティ種別	システム定義プロパティ（**PROP_x** の x の部分）				
ラベル指定したコマンド	DCR	RCR	OBJECT	VCR	PACL
CREATE_RFCT	○	○	○	×	×
CREATE_DOC	×	×	○	×	×
CREATE_VRDOC（マルチファイル文書）	×	×	○	×	×
CREATE_VRDOC（リファレンス文書）	×	×	○	○※	×
CREATE_VRDOC（シングルファイル文書）	×	×	○	○※	×
CREATE_DATA	×	×	×	×	×
CREATE_VARRAY	×	×	×	×	×
CREATE_VRCV	○	○	○	×	×
CREATE_CV	○	○	○	○※	×
CREATE_PACL	×	×	×	×	○
SELECT_OBJECT	○	○	○	○	○

（凡例）

○：指定できる　×：指定できない

注※

DocVersion クラスのスーパークラスが edmClass_VersionTracedDocVersion，または edmClass_VersionTracedComponentDocVersion のバージョン付き文書のラベルだけ指定できます。指定できないラベルを指定した場合，ワーニングエラー（KMBV11057-W）を出力して，そのオブジェクトを無視して処理を続行します。

(b) 指定方法 2

アクセス制御フラグ（ACFlag）の記述方法は次のとおりです。

パーミッション1｜パーミッション2｜…｜パーミッションn
（「｜」：区切り文字）

- システム定義プロパティの指定がない場合や入力データが「,,」の場合は，docaccess.ini ファイルの Entry0001 セクションに設定されているデフォルト値を設定します。
- デフォルト値が設定されていない場合は，FULL_CONTROL を設定します。
- ACFlag に設定できるパーミッションを表 4-24 と表 4-25 に示します。

表 4-24 基本パーミッションと対応する文字列

パーミッション文字列	パーミッション	意味
PRIM_READ_PROPS	基本プロパティ参照権	プロパティを参照する権限
PRIM_WRITE_PROPS	基本プロパティ更新権	プロパティを更新する権限
PRIM_READ_CONTENTS	基本コンテント参照権	文書の内容を参照する権限
PRIM_WRITE_CONTENTS	基本コンテント更新権	文書の内容を更新する権限
PRIM_LINK	基本リンク権	オブジェクト同士を関連づける権限
PRIM_VERSION	基本バージョン管理権	オブジェクトのバージョンを操作する権限
PRIM_DELETE	基本削除権	オブジェクトを削除する権限

表 4-25 組み合わせパーミッションと対応する文字列

パーミッション文字列	パーミッション	意味
READ_PROPS	プロパティ参照権	基本プロパティ参照権と同じ権限
READ	参照権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本コンテント参照権
WRITE_PROPS	プロパティ更新権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本プロパティ更新権
READ_WRITE	参照更新権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本プロパティ更新権 基本コンテント参照権 基本コンテント更新権
DELETE	削除権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本削除権
LINK	リンク権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本リンク権

パーミッション文字列	パーミッション	意味
VERSION	バージョン権	次の基本パーミッションを組み合わせ 基本プロパティ参照権 基本プロパティ更新権 基本コンテント参照権 基本コンテント更新権 基本バージョン管理権
FULL_CONTROL	フルコントロール	すべての基本パーミッションを組み合わせた権限
NONE	－	指定した場合，オブジェクト作成時に権限が設定されない（ほかのパーミッション文字列と一緒には指定できない）

- パーミッションの組み合わせが不正な場合はワーニングメッセージ（KMBV11069-W）を出力します。
- 組み合わせについての詳細は，マニュアル「DocumentBroker Version 3　システム導入・運用ガイド」を参照してください。

(c) 指定方法 3

PROP_XML_MAP の記述方法は次のとおりです。

["処理コード"/"マッピング定義名"/"XMSファイルの所在パス"]

- 処理コードの値には次の値が指定できます。

  "NP"
  　プロパティマッピングを実行しません。

  "OP"
  　プロパティマッピングを実行します。

  ""（省略）
  　プロパティマッピングを実行しません。
- 処理コードに上記以外の値を指定した場合はワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 制御ファイルに **PROP_XML_MAP** が指定されていない場合は，プロパティマッピングは実行しません。
- **PROP_XML_MAP** に指定する値は必ず引用符（"）で囲み，スラント（/）の前後に半角空白文字を入れることはできません。この規則に従わない場合，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 処理コードに "OP" を指定した場合，**PROP_CT** に値（XML ファイルのパス）を指定するか **PROP_SUB_XML** と **PROP_SUB_RT** に値（レンディションタイプと XML ファイルのパス）を指定します。値を指定しない場合，ワーニングメッセージ（KMBV11071-W）を出力し，該当オブジェクトを無視して処理を続行します。ただし，**PROP_SUB_XML** と **PROP_SUB_RT** に値を指定した場合で

も，**PROP_CT** には値を指定します。値を指定しない場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

- マッピング定義名には EDMXmlMap コマンドでマッピング定義を実行したときに使用したマッピング定義名を指定します。
- 処理コードに "OP" を指定した場合，マッピング定義名は必ず指定してください。指定しない場合はワーニングメッセージ（KMBV11073-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 指定したファイルが存在しない場合，パスが 255 文字を超えている場合またはファイルに参照権限がない場合は，ワーニングメッセージ（KMBV11074-W）を出力し，該当オブジェクトを無視して処理を続行します。
- プロパティリスト作成時にエラーが発生した場合，ワーニングメッセージ（KMBV11075-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 処理コードに "OP" を指定した場合で，XMS ファイルの所在パスが省略されている場合（["OP"/" マッピング定義名 "]，["OP"/" マッピング定義名 "/] または ["OP"/" マッピング定義名 "/""]）は，「$DOCBROKERDIR/etc/xml_files」（UNIX の場合）または「（環境変数 DOCBROKERDIR に設定したディレクトリ）¥etc¥xml_files」（Windows の場合）下の文書空間 ID.xms を参照します。DocumentBroker サーバの EDMXmlMap コマンドで作成した XMS ファイルが上記のディレクトリ下に存在するかを確認してください。
- 処理コードに "NP" を指定した場合または省略した場合，マッピング定義名および XMS ファイルの所在パスに指定された値は無視して処理を行います。プロパティマッピングを実行しない場合，省略形として ["NP"]，[""] または [] という書式が指定できます。
- 処理コードが "NP" のとき，XMS ファイルの所在パスが省略されている場合は，「（環境変数 DOCBROKERDIR に設定したディレクトリ）/etc/xml_files」（UNIX の場合）または「（環境変数 DOCBROKERDIR に設定したディレクトリ）¥etc¥xml_files」（Windows の場合）下の文書空間 ID.xms を参照します。
- **PROP_XML_MAP** の記述例（Windows の場合）を次に示します。UNIX の場合，ディレクトリの区切り文字が「/」になります。

  プロパティマッピング実行
  …,["OP"/"sample"/"C:¥tmp¥sample.xms"],…
  プロパティマッピング非実行
  …,[""],…

- 次の場合は，ワーニングメッセージ（KMBV11075-W）を出力して，該当オブジェクトを無視して処理を続行します。
  - HiRDB Adapter for XML がインストールされていない場合
  - 環境変数 XMLBRKDIR，LIBPATH（AIX の場合）にパスが設定されていない場合

(d) 指定方法 4

PROP_XML_INDEX の記述方法は次のとおりです。

["処理コード"/"フィルタリング定義ファイルの所在パス"]

- 処理コードの値には次の値が指定できます。

  "NP"
  　インデクスファイルを作成しません。

  "OP"
  　構造指定全文検索インデクスファイルを作成します。

  ""（省略）
  　インデクスファイルを作成しません。
- 処理コードに上記以外の値を指定した場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 制御ファイルに **PROP_XML_INDEX** が指定されていない場合は，インデクスファイルは作成しません。
- **PROP_XML_INDEX** に指定する値は必ず引用符（"）で囲み，スラント（/）の前後に半角空白文字を入れることはできません。この規則に従わない場合，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 処理コードに "OP" を指定した場合，**PROP_CT** に値を指定するか **PROP_SUB_XML** と **PROP_SUB_RT** の両方に値を指定します。指定しない場合，ワーニングメッセージ（KMBV11071-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 処理コードに "OP" を指定した場合で，XML 文書登録先のクラスが全文検索機能付き DocVersion クラスでない場合，ワーニングメッセージ（KMBV11076-W）を出力して処理を続行します。
- フィルタリング定義ファイルに指定したファイルが存在しない場合，パスが 255 文字を超えている場合またはファイルに参照権限がない場合は，ワーニングメッセージ（KMBV11077-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 構造指定全文検索ファイル作成時にエラーが発生した場合，ワーニングメッセージ（KMBV11078-W）を出力し，該当オブジェクトを無視して処理を続行します。
- 処理コードに "OP" を指定した場合で，フィルタリング定義ファイルの所在パスを指定しない場合は，フィルタリング処理を行わずにインデクスファイルを作成します。この場合，**PROP_XML_INDEX** には ["OP"]，["OP"/] または ["OP"/""] と指定します。
- 処理コードに "OP" を指定した場合で，**PROP_XML_MAP** に "NP" が指定されている場合は，「$DOCBROKERDIR/etc/xml_files」（UNIX の場合）または「(環境変数 DOCBROKERDIR に設定したディレクトリ）¥etc¥xml_files」（Windows の場合）下の文書空間 ID.xms を参照します。DocumentBroker サーバの EDMXmlMap コマ

ンドで作成した XMS ファイルが上記のディレクトリ下に存在するかを確認してください。

- 処理コードに "NP" を指定した場合または省略した場合はフィルタリング定義ファイルの所在パスに指定された値は無視して処理を行います。インデクスファイルを作成しない場合，省略形として ["NP"]，[""] または [] という書式が指定できます。
- **PROP_XML_INDEX** の記述例を次に示します。

  構造指定全文検索ファイル作成
  …,["OP"/"sample.tfd"],…
  構造指定全文検索ファイル未作成
  …,[""],…

- 次の場合は，ワーニングメッセージ（KMBV11075-W）を出力して，該当オブジェクトを無視して処理を続行します。
  - HiRDB Adapter for XML がインストールされていない場合
  - 環境変数 XMLBRKDIR，LIBPATH（AIX の場合）にパスが設定されていない場合

(e) 指定方法 5

PROP_PARSE_LEVEL には次の値を指定できます。省略した場合は 0 を仮定します。

0

DTD の解析を行いません。外部エンティティは無視します。構文解析時に実体解決を行わないため，図などの外部エンティティがなくても構文解析エラーになりません。

1

DTD を含めた構文解析をして外部エンティティを読み込みますが，検証はしません。構文解析時に実態解決を行いますが，妥当性の検証は行いません。

2

DTD を使った検証を行います。構文解析時に実態解決を行います。また，妥当性の検証も行います。

- **PROP_PARSE_LEVEL** に上記以外の値を指定した場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

(f) 指定方法 6

- **PROP_SUB_XML** の値が指定されている場合は，**PROP_CT** ではなく **PROP_SUB_XML** で指定されるオブジェクトを解析する XML 文書とみなします。
- **PROP_SUB_XML** の値が，制御ファイルに記述されている **PROP_SUB_RT** の数よりも大きい場合，および **PROP_SUB_XML** の値が 1 ～ 9 の値以外である場合は，ワーニングメッセージ（KMBV11052-W）を出力し，該当オブジェクトを無視して処理を続行します。

- **PROP_SUB_XML** の指定が「,,」の場合は，**PROP_CT** に指定されている文書を解析する XML 文書とみなします。

(g) 指定方法 7

XML 文書登録時（プロパティマッピング，構造指定全文検索ファイル登録あり）のシステム定義プロパティの指定について次の表に示します。

表 4-26 XML 文書登録時に指定できるシステム定義プロパティの組み合わせ

組み合わせ	システム定義プロパティ （**PROP_x** の x の部分）					
	XML_MAP	XML_INDEX	PARSE_LEVEL	SUB_XML	CT	SUB_RT
1	●	●	●	×	●	○
2	●	●	●	●	●	●

（凡例）
○：指定できる ●：必ず指定する ×：エラー

XML 文書登録時に制御ファイルに上記六つのプロパティを指定する場合は，上記二つの組み合わせで指定します。これ以外の組み合わせで記述した場合は，エラーメッセージを出力します。

(7) データの指定

文字列データの指定，数値データの指定，Boolean データの指定，繰り返しデータ（HiRDB Array 列）の指定，および予約語の指定について説明します。

(a) 文字列データの指定

- 文字列は，必ず引用符（"）で囲む必要があります。
- カラム区切り文字または行間区切り文字と引用符（"）の間は空けないでください。間に半角空白などのほかの文字列が入っている場合，エラーになります。
- 0 ～（データベースの定義サイズ）のバイトの長さで指定してください。
- データの前の半角空白（0x20）（行先頭またはカラム間区切り文字からデータの先頭文字まで半角空白）とあとの半角空白（データの最後の文字からカラム間区切り文字または行間区切り文字までの半角空白）がある場合は，エラーになります。
  なお，データを指定しなかった場合（「,,」または「,"",」）は，データベースに登録されている DocumentBroker のメタ情報に定義された初期値を設定します。メタ情報ファイルに定義されていない場合は NULL 値を設定します。
- 文字列データ中の引用符（"）は二つ並べて（""）記述します。
- 「¥0」の指定について，0x00 およびデータ長 0 バイトの文字列（空文字）をデータベースに格納するための記述仕様を次の表に示します。

表 4-27 データベース（MVARCHAR）に格納する「¥0」値に対応する記述例と格納結果

記述例	EmptyValue の指定値	格納結果	説明
"0x00"	制御ファイルの Control セクションの EmptyValue=0 指定時	0x00	データ長 1 バイトの 0x00
	制御ファイルの Control セクションの EmptyValue=1 指定時	空文字	データ長 0 バイトの文字列

- NULL 文字（0x00）を含む文字列データを格納する場合は，NULL 文字（0x00）を含む文字列データをそのまま指定します。
- 文字列データの記述例と格納結果を次の表に示します。

表 4-28 文字列データの記述例と格納結果

記述例	格納結果	説明
ABC	エラー	データの前後に " がないためエラーになる
"ABC"	ABC	―
"ABC	エラー	終わりの " がないためエラーになる
"ABC,DE"	ABC,DE	データにカラム間区切り文字がある場合
"ABC¥nDE"	ABC¥nDE	データに行間区切り文字がある場合
"ABC""DE"	ABC"DE	データに " がある場合
△ ABC △	エラー	データの前後に " がないためエラーになる
△ " △ ABC △ " △	エラー	" の前後に空白があるためエラーになる
0x00 ※	エラー	データの前後に " がないためエラーになる
"0x00"	0x00	EmptyValue=0 の場合
	空文字（長さ 0）	EmptyValue=1 の場合
"""0x00"""	"0x00"	―
"0x09"	0x09	―
0x09	エラー	データの前後に " がないためエラーになる
指定なし	初期値または NULL 値	メタ情報ファイルに定義された初期値 初期値が定義されていない場合は NULL 値
""	初期値または NULL 値	メタ情報ファイルに定義された初期値 初期値が定義されていない場合は NULL 値
△	初期値または NULL 値	メタ情報ファイルに定義された初期値 初期値が定義されていない場合は NULL 値

（凡例）
△：半角空白
，：カラム間区切り文字
¥n：行間区切り文字

注※
- 「0x00」はバイナリコードのため，入力データファイルへの設定はバイナリコードのエディタ（xi など）およびアプリケーションプログラムで行う必要があります。
- テキストエディタ（vi など）での入力はできません。
- バイナリコードのエディタ・アプリケーションプログラムなどで設定された入力データファイルの「0x00」はテキストエディタでは見えません。

(b) 数値データの指定

- 記述形式は次のとおりです。「△」は半角空白の並びを示します。

10 進表記の場合

△ [{'+' または ' − '}] 数字（0 ～ 9）の並び△

16 進表記の場合

△ [{'0X' または '0x'}] 英数字（0 ～ 9，A ～ F，a ～ f）の並び△

- 10 進数値または 16 進数値を文字列で指定します。16 進表記の場合は，先頭を "0X" または "0x" で指定してください。
- 指定できるのは，-2147483648 ～ 2147483647 の整数値です。
- データの前の半角空白（0x20）（行先頭またはカラム間区切り文字からデータの先頭文字まで半角空白）とあとの半角空白（データの最後の文字からカラム間区切り文字または行間区切り文字までの半角空白）がある場合は，半角空白を削除したあとの文字列をデータにします。
  なお，データを指定しなかった場合（「,,」または「,"",」）は，データベースに登録されている DocumentBroker のメタ情報に定義された初期値を設定します。メタ情報ファイルに定義されていない場合は NULL 値を設定します。
- 数値データを引用符（"）で囲んだ場合は，文字列データとみなされてエラーになります。数値データの記述例と格納結果を次の表に示します。

表 4-29 数値データの記述例と格納結果

記述例	格納結果	説明
+10	10	—
-009	-9	—
△ 10 △	10	データの前後の空白は削除する
" △ 10 △ "	エラー	文字列データとみなされてエラーになる
0xFF	255	—
0xF	15	—
指定なし	初期値	メタ情報ファイルに定義された初期値。初期値が定義されていない場合は NULL 値。

（凡例）
△：半角空白

(c) Boolean データの指定

- Boolean データの論理値を次の文字列で指定します。
  - 真値："TRUE"，TRUE，または 1
  - 偽値："FALSE"，FALSE，または 0
  - 不定値："UNKNOWN"，UNKNOWN，または 2
- データの前の半角空白（0x20）（行先頭またはカラム間区切り文字からデータの先頭文字まで半角空白）とあとの半角空白（データの最後の文字からカラム間区切り文字または行間区切り文字までの半角空白）がある場合は，半角空白を削除したあとの文字列をデータにします。
  なお，データを指定しなかった場合（「,,」または「,"",」）は，データベースに登録されている DocumentBroker のメタ情報に定義された初期値を設定します。メタ情報ファイルに定義されていない場合は NULL 値を設定します。
- Boolean データの記述内容と格納結果を次の表に示します。

表 4-30 Boolean データの記述内容と格納結果

記述内容	格納結果
TRUE	1
"TRUE"	1
1	1
true	エラー
FALSE	0
"FALSE"	0
0	0
false	エラー
UNKNOWN	2
"UNKNOWN"	2
2	2
unknown	エラー

(d) 繰り返しデータ（HiRDB Array 列）の指定

繰り返しデータ（HiRDB Array 列）で指定した型の値を指定します。記述方法を次に示します。

{要素値1,要素値2,…,要素値n}　　（「,」：カラム間区切り文字）

- 全体を波括弧（{　}）で囲んで記述します。
- 各要素値はカラム間区切り文字で区切って記述します。
- 波括弧の前後，カラム間区切り文字の前後に半角空白（0x20）を挿入できません。
- 繰り返しデータ（HiRDB Array 列）に値を指定しない場合は指定を省略します。

- 指定できる要素数は，1から繰り返しデータ（HiRDB Array 列）に定義した最大要素数までです。
- 要素数が繰り返しデータ（HiRDB Array 列）に定義した最大要素数を超える場合はエラーになります。
- 繰り返しデータ（HiRDB Array 列）の記述例と格納結果を次の表に示します。

表 4-31　繰り返しデータ（HiRDB Array 列）の記述例と格納結果

記述例	格納結果
{"aa","bb","cc"}	aa，bb，cc
{"aa","bb"}	aa，bb（繰返し列の 3 番目には何も格納されない）
{"aa",,"cc"}	aa，初期値（メタ情報ファイルに定義された初期値。初期値が定義されていない場合は NULL 値），cc
指定なし	初期値（メタ情報ファイルに定義された初期値。初期値が定義されていない場合は NULL 値）
△	初期値（メタ情報ファイルに定義された初期値。初期値が定義されていない場合は NULL 値）
{}	初期値（メタ情報ファイルに定義された初期値。初期値が定義されていない場合は NULL 値）
{"aa","bb","cc","dd"}	要素数が範囲外のためエラー
△{"aa","bb","cc"}△	波括弧（{}）の前後に半角空白が指定されているためエラー
{△"aa"△,"bb","cc"}	文字列データの 1 要素目の前後に半角空白が指定されているためエラー

（凡例）
,：カラム間区切り文字
△：半角空白

(e) 予約語の指定

- 予約語はオブジェクトローダで使用する特定の語で，記述された予約語に応じてオブジェクトローダが処理を行います。
- 予約語は，String 型ユーザ定義プロパティのプロパティ値として指定できます。
- 使用できる**予約語**を次の表に示します。

表 4-32　予約語一覧

予約語	内容
P_XML	XML 文書登録時にプロパティマッピングを実行する場合，入力データファイルにプロパティマッピング用のプロパティの識別子として指定する。
NULLCHARACTER	String 型で定義するプロパティに長さ 0 バイトの文字列を登録する場合，入力データファイルにプロパティ値として指定する。
P_CHID ※	プロパティ値として，登録先クラスのオブジェクトと同時に作成する最上位クラスのオブジェクトの dmaProp_OIID プロパティ値（133 バイト）を設定する場合に，バージョン付きクラスの登録先クラスの文字列データに対して P_CHID を指定する。

予約語	内容
P_CHID16※	プロパティ値として，登録先クラスのオブジェクトと同時に作成する最上位クラスのオブジェクトの dmaProp_OIID プロパティ値（末尾 16 バイト）を設定する場合に，バージョン付きクラスの登録先クラスの文字列データに対して P_CHID16 を指定する。
P_CHID52※	プロパティ値として，登録先クラスのオブジェクトと同時に作成する最上位クラスのオブジェクトの dmaProp_This プロパティ値（52 バイト）を設定する場合に，バージョン付きクラスの登録先クラスの文字列データに対して P_CHID52 を指定する。

注※

次のクラスの文字列データに対して指定できます。

- DocVersion クラス
- VersionTracedDocVersion クラス
- VersionTracedComponentDocVersion クラス
- 全文検索機能付き DocVersion クラス

予約語 P_XML の指定方法

予約語 P_XML を指定する場合，前後に半角空白または引用符（"）を指定したときはエラーになり，ワーニングメッセージ（KMBV11052-W）を出力します。

XML 文書登録時にプロパティマッピングを実行する場合の制御ファイルと入力データファイルの記述例を次に示します。

制御ファイルの記述例

```
[DataMapping]
XML-DOC=P_A,P_B,XML_P_A,XML_P_B,**PROP_XML_MAP**,
**PROP_XML_INDEX**,**PROP_PARSE_LEVEL**,
**PROP_CT**,**PROP_RTYPE**
```

この中でプロパティマッピング用のユーザプロパティが XML_P_A,XML_P_B であるとします。プロパティマッピングを実行する場合（**PROP_XML_MAP** の処理コードを "OP" にする），プロパティマッピング用のプロパティに P_XML を指定します。

入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

入力データファイルの記述例

```
CREATE_DOC,,XML_DOC,"A","B",P_XML,P_XML,
["OP"/"sample"/"C:¥tmp¥sample.xms"] ,
["OP"/"C:¥tmp¥sample.tfd"], 2 , C:¥tmp¥sample.xml,
"MIME::text/xml"
```

プロパティマッピングを実行しない場合（**PROP_XML_MAP** の処理コードを "NP" にする），プロパティマッピング用のプロパティには通常のプロパティ値を指定します。記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

入力データファイルの記述例

```
CREATE_DOC,,XML_DOC,"A","B",,"C",["NP"/"sample"/"C:¥tmp¥
sample.xms"] , ["OP"/"C:¥tmp¥sample.tfd"] ,2 ,
C:¥tmp¥sample.xml, "MIME::text/xml"
```

予約語 NULLCHARACTER の指定方法

String 型で定義するプロパティに長さ 0 バイトの文字列を登録する場合の制御ファイルの記述例と，入力データファイルの記述例を次に示します。

制御ファイルの記述例

```
[DataMapping]
JIKEN-CONTAINER=J-TITLE,…
```

入力データファイルの記述例

```
TRANBEGINCREATE_RFCT,LABEL-C,JIKEN-CONTAINER,NULLCHARACTER,
…
```

予約語 P_CHID，P_CHID16，または P_CHID52 の指定方法

予約語 P_CHID16 を指定する場合の制御ファイルの記述例と，入力データファイルの記述例を次に示します。予約語 P_CHID または P_CHID52 を指定する場合も，同様の方法で記述してください。

制御ファイルの記述例

```
[DataMapping]

dwcClass_VerDoc_ch=**PROP_DOC_CLASS**,**PROP_RTYPE**,**PROP
_CT**,dw
cProp_Name,dwcClass_VerDoc_dv@dwcProp_VersioningOIID,…
```

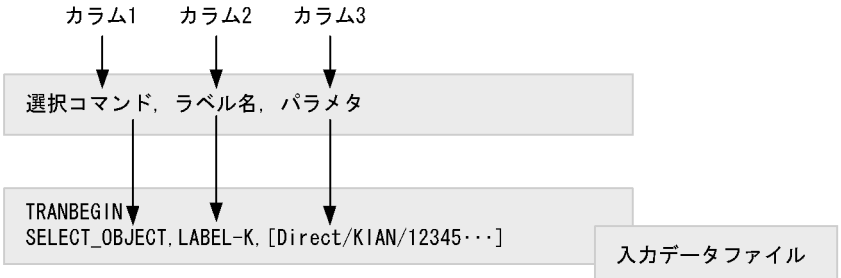
入力データファイルの記述例

```
CREATE_VRDOC,,dwcClass_VerDoc_ch,dwcClass_VerDoc_dv,"text/
plain","C:¥content.txt","文書名",P_CHID16,…
```

4.5.6 選択コマンドの記述方法

選択コマンドの記述方法を図 4-4 に示します。

図 4-4 選択コマンドの記述方法



カラム 3 には，選択するオブジェクトの検索条件を指定します。指定方法は次の 2 通りです。ただし，一致するオブジェクトが存在しない場合，および一致するオブジェクトが複数ある場合はエラーになります。

指定形式 1

```
[Direct/クラスID/OIID]
```

指定形式 2

```
[Query/クラスID/プロパティID=プロパティ値の並び]
```

指定規則

- パラメタカラムに使用できる文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。
- 指定形式 1 および指定形式 2 どちらの指定方法の場合でも，全体を角括弧（[ ]）で囲みます。
- 「=」の前後に半角空白（0x20）を挿入できます。
- クラス ID およびプロパティ ID は，定義ファイルに記述した文字列をそのまま記述します。
- 指定方法 1 の場合，指定したクラス ID と OIID 値が一致したオブジェクトを取得します。OIID は，「/」と「]」で囲まれた文字列の前後の空白を除いた部分を OIID として扱います。OIID 値は選択するオブジェクトの OIID の末尾 16 バイトを指定します。OIID 値は英大文字の A ～ F と数字 0 ～ 9 で記述します。なお，選択するオブジェクトの OIID がわかっている場合にこの指定方法を使用します。
- 指定方法 2 の場合，指定したクラス ID とプロパティ ID= プロパティ値とで一致したオブジェクトを取得します。
- プロパティ ID= プロパティ値は，AND 条件で最大 5 個指定できます。複数指定する場合は「/」を区切り文字として挿入します。
- プロパティ値は，「=」と「/」または「]」で囲まれた文字列の前後の半角空白を除いた文字列をプロパティ値として扱います。区切り文字を記述して，プロパティ ID= プロパティ値を記述しない場合はエラーメッセージが出力されます。なお，この指定方法を使用する場合は，指定するクラスでユニークなプロパティを追加しておく必要があります。
- 引用符（"）を含む文字列をプロパティ値として指定する場合は，文字列の前後を引用符（"）で囲み，文字列中の引用符（"）は引用符（"）を二つ並べて記述します。
- プロパティ値の記述例と検索対象になる文字列の対応を，次の表に示します。

表 4-33　プロパティ値の記述例と検索対象になる文字列の対応

プロパティ値	検索対象文字列	プロパティの型
指定なし	エラー	String 型，Integer32 型，および

プロパティ値	検索対象文字列	プロパティの型
△	エラー	Boolean 型
" △ "	△	String 型
""" △ """	" △ "	
"a"	a	
△ "a" △	エラー	
" あ "	あ	
△ a △	エラー	
" △ a △ "	△ a △	
"a""b"	a"b	
"0x00"（文字列の 0x00）	0x00（文字列の 0x00）	
1	1	Integer32 型
"1"	エラー	
0	0	Boolean 型
"0"	エラー	
TRUE	エラー	
FALSE	エラー	
UNKNOWN	エラー	

（凡例）

△：半角空白

- プロパティ ID に VariableArray 型のプロパティの要素を指定する場合は，プロパティ ID の前に「VariableArray 型プロパティ名 .」を付加します。VariableArray 型プロパティ名とは，edmClass_Struct クラスのサブクラスを要素にするプロパティのことです。

指定例

```
[Query/クラスID/プロパティID1=プロパティ値1/プロパティID2=プロパティ値2]
```

4.5.7 指示コマンドの記述方法

指示コマンドの記述方法について説明します。

（1） TRANBEGIN と TRANCOMMIT（トランザクション）の指定

- TRANBEGIN，TRANCOMMIT 共にカラム 1 に記述します。TRANBEGIN と TRANCOMMIT で囲んだオブジェクトが 1 トランザクションになるので，必ず対で記述します。
- TRANBEGIN と TRANCOMMIT の間に，TRANBEGIN および TRANCOMMIT は記述できません。TRANBEGIN および TRANCOMMIT を記述しない場合は，1 オブ

ジェクトを 1 トランザクションとして扱います。

- トランザクション単位に，データベースへの登録が成功した場合はコミット，データベースへの登録が失敗した場合はロールバックを実行します。ただし，ジャーナルの出力を指定しない場合は，ロールバックは実行されません。
- 1 トランザクション内で発生したエラーについては，エラーレベルによって次のとおりの処理が実行されます。

  警告レベルのエラー（メッセージ ID の種別が -W の場合）

  実行中のトランザクションをロールバックして，次のトランザクションの処理を実行します。

  障害レベルのエラー（メッセージ ID の種別が -E の場合）

  実行中のトランザクションをロールバックして，プログラムを終了します。

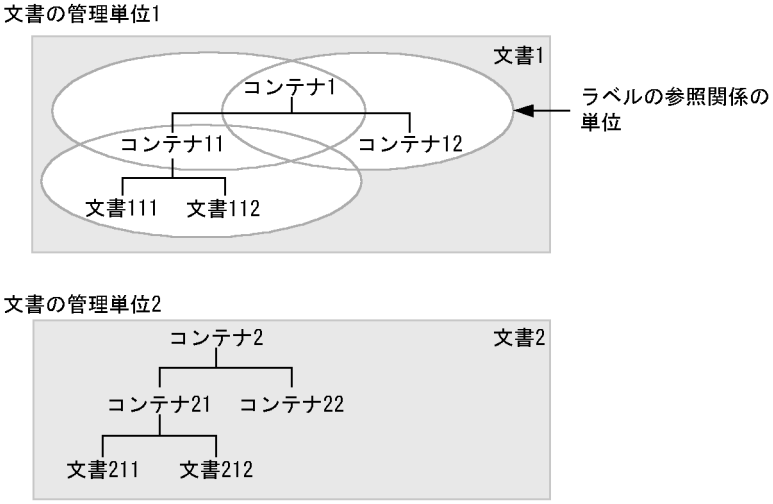
(2) トランザクションの管理

トランザクション単位でコミットするため，複数のコマンドを 1 トランザクションで実行すると処理能力は向上します。処理能力を向上させるためのトランザクションの決め方を次に示します。

1. **ラベルの包含関係が解決する単位を決めます。**
2. 1. **をすべて包含する文書の管理単位を 1 トランザクションにします。**

図 4-5 に文書の管理単位とトランザクション単位の例を示します。

図 4-5 文書の管理単位とトランザクション単位



(3) #（コメント）の指定

カラム 1（行の先頭文字）に「#」を記述した場合は，その行のデータはコメントとみなされます。

4.5.8 コマンドの記述順序

ここでは，コマンドの記述に順序性がある場合の記述順所を説明します。

(1) データベースに未登録のオブジェクトとリンクさせて登録する場合の記述順序

次の順序で記述してください。

1. **リンク先のオブジェクトのコマンドをラベル指定で記述する。**
2. **登録するコマンドを記述する。**
3. **リンク先はラベル指定したラベルを記述する。**

登録するコマンドを記述する前に記述しておく必要があるコマンド（リンク先のオブジェクトのコマンド）を次の表に示します。

表 4-34 登録するコマンドとリンク先のオブジェクトのコマンドの関係

登録するコマンド	システムプロパティ	リンク先のオブジェクトのコマンド
CREATE_RFCT	**PROP_DCR**，**PROP_RCR**	CREATE_RFCT，CREATE_VRCV，CREATE_CV
	PROP_PACL	CREATE_PACL
CREATE_DOC	**PROP_DCR**，**PROP_RCR**	CREATE_RFCT，CREATE_VRCV，CREATE_CV
	PROP_PACL	CREATE_PACL
CREATE_VRDOC	**PROP_DCR**，**PROP_RCR**	CREATE_RFCT，CREATE_VRCV，CREATE_CV
	PROP_PACL	CREATE_PACL
CREATE_DATA	**PROP_PACL**	CREATE_PACL
CREATE_VARRAY	**PROP_OBJECT**	CREATE_RFCT，CREATE_DOC，CREATE_VRDOC，CREATE_VRCV，CREATE_CV
CREATE_VRCV	**PROP_DCR**，**PROP_RCR**	CREATE_RFCT，CREATE_VRCV，CREATE_CV
	PROP_VCR	CREATE_VRDOC
	PROP_PACL	CREATE_PACL
CREATE_CV	**PROP_DCR**，**PROP_RCR**	CREATE_RFCT，CREATE_VRCV，CREATE_CV
	PROP_VCR	CREATE_VRDOC
	PROP_PACL	CREATE_PACL

記述例

入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

```
#リンク先のオブジェクトをラベル(LABEL_CT)を使用して記述する。
CREATE_RFCT,LABEL_CT,USER_PCon,"Parent"
#登録するコマンドと，リンク先ラベルを記述する。(**PROP_DCR**を
[DataMapping]に定義)
CREATE_RFCT,,USER_Ccon,"Child",[Label/LAEBL_CT]
```

説明：CREATE_RFCT でコンテナオブジェクトのリンクを行います。

```
#リンク先のオブジェクトをラベル(LABEL_CH)を使用して記述する。
CREATE_VRDOC,LABEL_CH,USER_CH,"MIME::/ms-word","C:¥tmp¥FS.doc"
#登録するコマンドと，リンク先ラベルを記述する。(**PROP_VCR**を
[DataMapping]に定義)
CREATE_VRCV,,USER_VRCH,[Label/LAEBL_CH],[]
```

説明：CREATE_VRCV で構成管理オブジェクトのリンクを行います。

(2) データベースに登録済みのオブジェクトとリンクさせて登録する場合の記述順序

次の順序で記述してください。

1. SELECT_OBJECT **コマンドをラベル指定で記述する。**
2. **登録するコマンドを記述する。**
3. **リンク先は** 1. **で指定したラベルを記述する。**

(3) 複数のコマンド（TRANBEGIN および TRANCOMMIT 以外）の実行を 1 トランザクションで行う場合の記述順序

次の順序で記述してください。

1. TRANBEGIN **コマンドを記述する。**
2. **実行するコマンドを記述する。**
3. TRANCOMMIT **コマンドを記述する。**

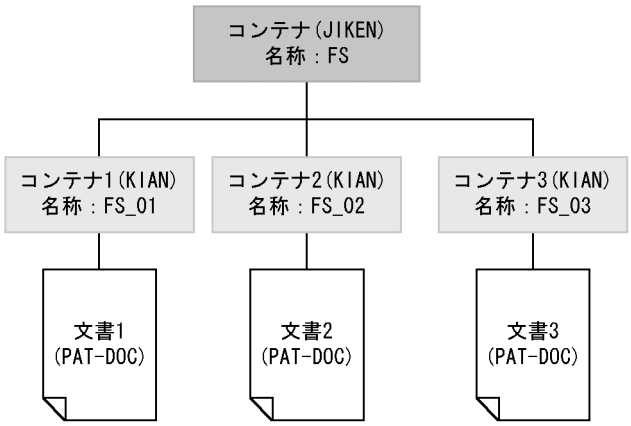
4.5.9 入力データファイルの記述例

入力データファイルの記述例を示します。

(1) TRANBEGIN ～ TRANCOMMIT でコンテナと文書を登録する例

登録するコンテナと文書の管理構成を図 4-6 に示します。

図 4-6　登録するコンテナと文書の管理構成



(a) コンテナの深度優先（文書の管理構成の縦方向）に作成する場合

作成順序

1. **コンテナ**（JIKEN）
2. **コンテナ** 1（KIAN）
3. **文書** 1（PAT-DOC）
4. **コンテナ** 2（KIAN）
5. **文書** 2（PAT-DOC）
6. **コンテナ** 3（KIAN）
7. **文書** 3（PAT-DOC）

記述例

入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。
この場合，コンテナにラベルを指定しながら作成します。

```
TRANBEGIN
#
#コンテナ（JIKEN）を作成する。
CREATE_RFCT, LABEL-J, JIKEN, "A", "FS", "19980819"
#カレントコンテナ（JIKEN）にコンテナ1（KIAN）を直接型で生成する。
CREATE_RFCT, LABEL-K1, KIAN, "A", "FS_01", "19980819", "第1版",
[Label/LABEL-J]
#カレントコンテナ1（KIAN）にバージョン付文書（fs01.doc）を作成する。
CREATE_VRDOC, , PAT-CONFIG,"A","DocBro-FS","第1版",
"MS-WORD",[],"C:¥tmp¥fs01.doc",[Label/LABEL-K1]
#
#コンテナ（JIKEN）にコンテナ2（KIAN）を直接型で生成する。
CREATE_RFCT, LABEL-K2, KIAN, "A", "FS_02", "19980821", "第2版",
[Label/LABEL-J]
#コンテナ2（KIAN）にバージョン付文書（fs02.doc） を作成する。
CREATE_VRDOC, , PAT-CONFIG,"A","DocBro-FS","第2版",
"MS-WORD",[],"C:¥tmp¥fs02.doc",[Label/LABEL-K2 ]
```

```
#
#コンテナ（JIKEN）にコンテナ3（KIAN）を直接型で生成する。
CREATE_RFCT,LABEL-K3, KIAN,"A", "FS_03", "19980821", "第3版",
[Label/KABEL-J]
#コンテナ3（KIAN）にバージョン付文書（fs03.doc） を作成する。
CREATE_VRDOC, , PAT-CONFIG,"A","DocBro-FS","第3版",
"MS-WORD",[],"C:¥tmp¥fs03.doc",[Label/LABEL-K3 ]
#
TRANCOMMIT
```

説明

- コンテナ（JIKEN）をラベル（LABEL-J）指定で作成します。
- コンテナ 1（KIAN）は [Label/LABEL-J] でリンク，作成できます。
- コンテナ 1（KIAN）をラベル（LABEL-K1）指定で作成することで，文書 1（PAT-DOC）は [Label/LABEL-K1] でリンクできます。
- コンテナ 2（KIAN）は [Label/LABEL-J] でリンク，作成できます。
- コンテナ 2（KIAN）をラベル（LABEL-K2）指定で作成することで，文書 2（PAT-DOC）は [Label/LABEL-K2] でリンクできます。
- コンテナ 3（KIAN），文書 3（PAT-DOC）の作成は，コンテナ 2（KIAN），文書 2（PAT-DOC）の場合と同様です。

(b) コンテナの階層優先（文書の管理構成の横方向）に作成する場合

作成順序

1. **コンテナ**（JIKEN）
2. **コンテナ** 1（KIAN）
3. **コンテナ** 2（KIAN）
4. **コンテナ** 3（KIAN）
5. **文書** 1（PAT-DOC）
6. **文書** 2（PAT-DOC）
7. **文書** 3（PAT-DOC）

記述例

入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。
この場合，コンテナをラベル指定ですべて生成後に，文書を生成します。

```
 TRANBEGIN
 #コンテナ（JIKEN）を作成する。
 CREATE_RFCT,LABEL-J, JIKEN,"A","FS","19980819"
 #コンテナ（JIKEN）にコンテナ1（KIAN）を直接型で生成する。
 CREATE_RFCT,LABEL-K1, KIAN,"A","FS_01","19980819",
"第1版",[Label/LABEL-J]
 #コンテナ（JIKEN）にコンテナ2（KIAN）を直接型で生成する。
 CREATE_RFCT,LABEL-K2, KIAN,"A","FS_02","19980821",
"第2版", [Label/LABEL-J]
 #コンテナ（JIKEN）にコンテナ3（KIAN）を直接型で生成する。
 CREATE_RFCT,LABEL-K3, KIAN,"A","FS_03","19980821",
```

```
"第3版", [Label/LABEL-J]
 #
 #コンテナ1（KIAN）にバージョン付文書（fs01.doc） を作成する。
 CREATE_VRDOC, ,PAT-CONFIG,"A","DocBro-FS","第1版",
 "MS-WORD",[],"C:¥tmp¥fs01.doc",[Label/LABEL-K1 ]
 #コンテナ2（KIAN）にバージョン付文書（fs02.doc） を作成する。
 CREATE_VRDOC, ,PAT-CONFIG,"A","DocBro-FS","第2版",
 "MS-WORD",[],"C:¥tmp¥fs02.doc",[Label/LABEL-K2 ]
 #コンテナ3（KIAN）にバージョン付文書（fs03.doc） を作成する。
 CREATE_VRDOC, ,PAT-CONFIG,"A","DocBro-FS","第3版",
 "MS-WORD",[],"C:¥tmp¥fs03.doc",[Label/LABEL-K3 ]
 TRANCOMMIT
```

説明

- コンテナ（JIKEN）をラベル（LABEL-J）指定で作成します。
- コンテナ 1（KIAN）をラベル（LABEL-K1）指定で作成し，[Label/LABEL-J] でリンクさせます。
- 同じように，コンテナ 2（KIAN）をラベル（LABEL-K2）指定で作成し，[Label/LABEL-J] でリンクさせます。
- 文書 1（PAT-DOC）の作成時，[Label/LABEL-K1] でリンクさせます。
- 同じように，文書 2（PAT-DOC）の作成時は [Label/LABEL-K2] でリンクし，文書 3（PAT-DOC）の作成時は [Label/LABEL-K3] でリンクさせます。

（2） 独立永続クラスに登録する例

独立永続クラスにオブジェクトを登録する入力データファイルの記述例を次に示します。

```
#独立永続化クラスに登録
CREATE_DATA, ,I-DATA,AUTHOR
#
```

（3） 繰り返しデータ（HiRDB Array 列）にオブジェクトを登録する例

コンテナ（KIAN）に属する，繰り返しデータ（HiRDB Array 列）に格納する VariableArray 型のプロパティにデータを登録する場合の例を示します。なお，繰り返しデータ（HiRDB Array 列）を使用する場合は，制御ファイルの DataMapping セクションに VariableArray 型プロパティ名を指定する必要があります。入力データファイルの定義例を次に示します。

```
#繰り返しデータ（HiRDB Array列）にオブジェクトを登録する例
CREATE_RFCT, , KIAN, "A", "FS_01", "19980819", "第1版", [], {1,2},
{"1章","2章"}
```

（4） 別表にオブジェクトを登録する例

コンテナ（KIAN）に属する，別表に格納する VariableArray 型のプロパティにデータを登録する場合の例を示します。なお，別表を使用する場合は，繰り返し用のクラス

（V-ARRAY）を定義する必要があります。入力データファイルの定義例を次に示します。

```
#別表にオブジェクトを登録する例
CREATE_RFCT, LABEL-K1, KIAN, "A", "FS_01", "19980819", "第1版", []
CREATE_VARRAY, ,V-ARRAY, [Label/LABEL-K1], 1, "1章"
CREATE_VARRAY, ,V-ARRAY, [Label/LABEL-K1], 2, "2章"
```

（5）繰り返しデータ（HiRDB Array 列）に選択コマンド（SELECT_OBJECT）を実行する例

繰り返しデータ（HiRDB Array 列）に選択コマンド（SELECT_OBJECT）を実行する入力データファイルの記述例を次に示します。

```
#繰り返しデータ（HiRDB Array列）のプロパティIDが'TYPE',VariableArray型プロパティ名が'USR_ARRAY'の場合に選択コマンド（SELECT_OBJECT）を実行する例

SELECT_OBJECT,LABEL-K,[Query/KIAN/USR_ARRAY.TYPE=1]
```

（6）構成管理コンテナを登録する例

構成管理コンテナを登録する入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

```
#構成管理用の文書の作成
CREATE_VRDOC,LABEL-VT,USER_CH,[],"MIME::/application/msword",
"C:¥tmp¥fs1.doc"
#バージョン付き構成管理コンテナの作成
CREATE_VRCV,,USER_VRCH,[USER_CV],[Label/LABEL_VT],FIX
#構成管理コンテナのバージョンの作成
CREATE_CV,,USER_CV,[Label/LABEL_VT],FIX
```

（7）XML 文書を登録する例

XML 文書を登録する入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

XML 文書を登録する場合，プロパティマッピングに使用するプロパティデータには「P_XML」を指定します。「P_XML」はオブジェクトローダの予約語でプロパティマッピング用のプロパティであることを示します。

```
#XML文書を登録する例
CREATE_RFCT,LABEL-X,KANPOU,D,work,20000518
#バージョンなしXML文書を登録(対象クラス:全文検索機能付きDocVersion，プロパティマッピングあり，検索ファイル登録あり)
CREATE_DOC,,XML-CSDV,"採決",P_XML,
["OP"/"kanpoum1"/"C:¥tmp¥kanp.xms"] ,
```

```
["OP"/"C:¥tmp¥kanp.tfd"] , 2 ,"C:¥tmp¥kanpou1.xml",
"MIME::text/xml", [Label/LABEL-X]
#バージョンなしXML文書を登録(対象クラス: 全文検索機能付きDocVersion, プロパティマッピングあり, 検索ファイル登録なし)
CREATE_DOC,,XML-CSDV, ,"20000427", P_XML,
["OP"/"kanpoum2"/"C:¥tmp¥kanp.xms"] , ["NP"] , 1 ,
"C:¥tmp¥kanpou2.xml","MIME::text/xml", [Label/LABEL-X]
#バージョンなしXML文書を登録(対象クラス: 全文検索機能付きDocVersion, プロパティマッピングなし, 検索ファイル登録あり,サブレンディションにXML文書を登録)
CREATE_DOC,,XML-CSDV, "取り扱い注意", P_XML,["NP"],
["OP"/"C:¥tmp¥kanp.flt"],0,1, , ,
[ "MIME::text/xml"/"C:¥tmp¥kanpou3.xml"] , [Label/LABEL-X]
#バージョンなしXML文書を登録(対象クラス:DocVersion, プロパティマッピングあり, 検索ファイル登録なし)
CREATE_DOC,, XML-DOC, P_XML,
["OP"/"kanpoum3","C:¥tmp¥kanp.xms"] , [""] ,
0 ,"C:¥tmp¥kanpou4.xml", "MIME::text/xml", [Label/LABEL-X]
#バージョンありXML文書を登録(対象クラス: 全文検索機能付きDocVersion, プロパティマッピングあり, 検索ファイル登録あり)
CREATE_VRDOC,,XML-CONFIG,
"木村",P_XML,P_XML,"A",[XCSDV],["OP"/"kanpoum4"/"C:¥tmp¥kanp.xms"],
["OP"/"C:¥tmp¥kanp.flt"],2,"C:¥tmp¥kanpou5.xml","MIME::text/xml",
[Label/LABEL-X]
#バージョンなしXML文書を登録(対象クラス: 全文検索機能付きDocVersion, プロパティマッピングなし, 検索ファイル登録あり)
CREATE_VRDOC,,XML-CONFIG,"館田",[XCSDV],["NP"] , ["OP"] , 0 ,
"C:¥tmp¥kanpou1.xml","MIME::text/xml", [Label/LABEL-X]
#バージョンなしXML文書を登録(対象クラス: 全文検索機能付きDocVersion, プロパティマッピングあり, 検索ファイル登録あり, サブレンディションの指定あり)
CREATE_DOC,,XML-CSDV,"採決",P_XML,
["OP"/"kanpoum1"/"C:¥tmp¥kanp.xms"] ,
["OP"/"C:¥tmp¥kanp.tfd"] , 2 ,"C:¥tmp¥kanpou1.doc",
"MIME::ms-word", [Label/LABEL-X], 2 ,[ "MIME::ms-excel"/
"C:¥tmp¥kanpou1.csv"| "MIME::text/xml"/"C:¥tmp¥kanpou1.xms"]
```

(8) マルチファイル文書を登録する例

マルチファイル文書を登録する場合の制御ファイルおよび入力データファイルの記述例を次に示します。なお, 次の記述例は, Windows の場合です。UNIX の場合, ディレクトリの区切り文字が「/」になります。

制御ファイルの記述例

```
[DataMappping]
usrClass_DocVersion=**PROP_RTYPE**,**PROP_CT_PATH**,**PROP_CT**
```

入力データファイルの記述例

```
CREATE_DOC,,usrClass_DocVersion,"text/plain","C:¥tmp",
ALL_FILE …1.
CREATE_DOC,,usrClass_DocVersion,"text/plain","C:¥tmp",
{"text1.txt","text2.txt"} …2.
```

1. 特定のディレクトリにあるファイルをすべて登録します。
2. 特定のディレクトリにある特定のファイルを複数登録します。

(9) リファレンスファイル文書を登録する例

リファレンスファイル文書を登録する場合の制御ファイル，および入力データファイルの記述例を次に示します。なお，次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

● コンテントを持たないでコンテントロケーションを管理しないリファレンスファイル文書を登録する場合（**PROP_REF_TYPE** に NONE を指定した場合）

制御ファイルの記述例

```
[DataMapping]
Class_A=**PROP_CT**,**PROP_RTYPE**,**PROP_REF_TYPE**,
**PROP_REF_BASEPATH**,**PROP_REF_PATH**
```

入力データファイルの記述例

```
CREATE_DOC,,Class_A,,"MIME::application/
x-edm-undefined",NONE,,
```

● ユーザが指定した任意のディレクトリにコンテントを格納して，コンテントロケーションを相対パスで管理する場合（**PROP_REF_TYPE** に RELATIVE を指定した場合）

制御ファイルの記述例

```
[DataMapping]
Class_B=**PROP_CT**,**PROP_RTYPE**,**PROP_REF_TYPE**,
**PROP_REF_BASEPATH**,**PROP_REF_PATH**
```

入力データファイルの記述例

```
CREATE_DOC,,Class_B,"C:¥tmp¥fs01.doc","MIME::application/
msword",RELATIVE,"C:¥dir01","USER01"
```

(10) パブリック ACL またはローカル ACL を登録する例

パブリック ACL またはローカル ACL を登録する場合の制御ファイルおよび入力データファイルの記述例を次に示します。

● ローカル ACL を設定したオブジェクトの作成

コンテナ作成時にローカル ACL を指定する記述例を次に示します。

制御ファイルの記述例

```
[DataMapping]
usrClass_Container=**PROP_ACL_OID**,**PROP_ACL_SUBJECT**,
**PROP_ACL_STYPE**,**PROP_ACL_PERM**
```

入力データファイルの記述例

```
CREATE_RFCT,001,usrClass_Container,"owner",{"user1",
"user2","group1"},{USR,USR,GRP},{FULL_CONTROL,
READ_WRITE,READ}
```

● パブリック ACL を設定したオブジェクトの作成（パブリック ACL を登録する場合）

パブリック ACL を作成して，コンテナにバインドする記述例を次に示します。

制御ファイルの記述例

```
[DataMapping]
edmClass_PublicACL=**PROP_ACL_OID**,**PROP_ACL_SUBJECT**,
```

```
**PROP_ACL_STYPE**,**PROP_ACL_PERM**
usrClass_Container=**PROP_ACL_OID**,**PROP_PACL**
```

入力データファイルの記述例

```
CREATE_PACL,002,edmClass_PublicACL,"owner",{"user1",
"user2","group1"},{USR,USR,GRP},{FULL_CONTROL,
READ_WRITE,READ}
CREATE_PACL,003,edmClass_PublicACL,"owner",{"user4",
"user3","group2"},{USR,USR,GRP},{FULL_CONTROL,
READ_WRITE,READ}
CREATE_RFCT,004,usrClass_Container,"owner",{[Label/002],
[Label/003]}
```

● **パブリック ACL を設定したオブジェクトの作成（オブジェクトを選択する場合）**

登録済みのパブリック ACL をコンテナにバインドする記述例，およびパブリック ACL をバインドしないコンテナを作成する記述例を次に示します。

制御ファイルの記述例

```
[DataMapping]
usrClass_Container=**PROP_ACL_OID**,**PROP_PACL**
```

入力データファイルの記述例

```
SELECT_OBJECT,005,[Direct/edmClass_PublicACL/
0000000000000001]
CREATE_RFCT,006,usrClass_Container,"owner",
{[Label/005]}  …1.
CREATE_RFCT,007,usrClass_Container,"owner",  …2.
```

1．登録済みのパブリックACLをコンテナにバインドします。
2．パブリックACLをバインドしないコンテナを作成します。

（11）String 型のプロパティに対して長さが 0 バイトの文字列を登録する例

コンテナ作成時に，String 型のプロパティに対して長さが 0 バイトの文字列を登録する場合の制御ファイルおよび入力データファイルの記述例について説明します。

String 型のプロパティに対して長さが 0 バイトの文字列を登録する場合，0 バイトの文字列を登録するプロパティデータには「NULLCHARACTER」を指定します。「NULLCHARACTER」はオブジェクトローダの予約語で 0 バイトの文字列を登録するプロパティであることを示します。

制御ファイルの記述例

```
[DataMapping]
JIKEN-CONTAINER=J-TITLE,…
```

入力データファイルの記述例

```
TRANBEGIN
CREATE_RFCT,LABEL-C,JIKEN-CONTAINER,NULLCHARACTER,…
```

4.6 環境変数ファイル（Windows の場合）

Windows の場合，オブジェクトローダが動作するために必要な情報を環境変数ファイル（NTconfig.ini）に設定します。

UNIX の場合，このファイルはありません。

4.6.1 記述形式

環境変数ファイルの記述形式を次に示します。

```
[ConfigObjLoader]
_HIEDMS_TRACE_DIR = トレースログファイルの
                    出力先ディレクトリのパス名 <改行コード>
_HIEDMS_TRACE_NUM = トレースログファイルの
                    ファイル数<改行コード>
_HIEDMS_TRACE_SIZE = トレースログファイルの
                     ファイルサイズ<改行コード>
_HIEDMS_TRACE_LEVEL = トレースレベル<改行コード>
TMPDIR = XML文書登録用の一時ディレクトリのパス名 <改行コード>
```

4.6.2 記述規則

環境変数ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- 各定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字列の後ろに半角空白またはタブを含めることができます。
- 使用できる改行コードは（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 文字列の途中に半角空白を含めることはできません。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。また，ワーニングメッセージ（KMBV20101-W）が出力されます。
- [ConfigObjLoader] セクションのあとに記述されたエントリが有効になります。それ以外の場所に記述された文字列はコメントとして扱われます。
- 指定できるエントリ以外の文字列はコメントとして扱われます。
- NTconfig.ini にアクセス権限がない場合，NTconfig.ini ファイルがない場合，またはパス名が 256 バイト以上の場合は，エラーメッセージ（KMBV20102-E）を出力して処理を終了します。

注意事項

インストール直後の NTconfig.ini の初期状態では，パラメタの設定内容はコメント行になっています。

4.6.3 環境変数ファイルに設定するパラメタ

環境変数ファイルに設定するパラメタを次に示します。必要なパラメタを NTconfig.ini ファイルに記述してください。

（1）_HIEDMS_TRACE_DIR

形式

```
_HIEDMS_TRACE_DIR = トレースログファイルの出力先ディレクトリのパス名
```

設定内容

- トレースログファイルの出力先ディレクトリのパス名を指定します。指定したディレクトリの下の ¥loader がトレースログファイルの出力先となります。
- このパラメタを省略した場合，インストールディレクトリ下の ¥spool¥loader が出力先ディレクトリになります。
- _HIEDMS_TRACE_DIR に指定したディレクトリが存在しない場合，または 256 文字以上の場合には，ワーニングメッセージ（KMBV20100-W）を出力し，インストールディレクトリ下の ¥spool¥loader を使用します。

（2）_HIEDMS_TRACE_NUM

形式

```
_HIEDMS_TRACE_NUM = トレースログファイルのファイル数
```

設定内容

- トレースの出力情報がトレースファイルのサイズの上限を超えた場合，トレースの出力先を別のファイルに切り替えられます。この場合の切り替えるファイル数を設定します。
- 設定できるファイルの数は 2 ～ 16 の値です。デフォルトは 2 です。
- 設定したファイル数までファイルが切り替わると，そのあとは最初に使用したファイルから順に上書きされます。

（3）_HIEDMS_TRACE_SIZE

形式

```
_HIEDMS_TRACE_SIZE = トレースログファイルのファイルサイズ
```

設定内容

トレースを出力するファイルのサイズを 4096 ～ 2147483647（バイト）で設定します。デフォルトは 1048576（1 メガバイト）です。

（4） _HIEDMS_TRACE_LEVEL

形式

```
_HIEDMS_TRACE_LEVEL = トレースレベル
```

設定内容

トレースの出力レベルを設定します。出力レベルには，0 または 10 が設定できます。デフォルトは 10 です。
それぞれを設定した場合に出力される情報を次に示します。

0 を設定した場合

- エラー情報
- サーバの開始と終了

10 を設定した場合

- トレースレベルが 0 の場合に出力される情報
- ユーザインターフェースの情報
- 他プログラムとのインターフェースの情報
- データベースへの接続と切断

（5） TMPDIR

形式

```
TMPDIR = XML文書登録用の一時ディレクトリのパス名
```

設定内容

- XML 文書登録用の一時ディレクトリのパス名を指定します。
- このパラメタを省略した場合，インストールディレクトリ下の ¥tmp が一時ディレクトリになります。
- ディレクトリは，XML 文書の文書サイズ以上の空きがある格納先を指定してください。
- オブジェクトローダ実行時，一時保管ディレクトリに一時ファイルを作成します。一時ファイルの作成に失敗した場合は，ワーニングメッセージ（KMBV11078-W）が出力されます。文書登録後に一時ファイルは削除されます。
- TMPDIR に指定したディレクトリが存在しない場合，または 256 文字以上の場合には，ワーニングメッセージ（KMBV20100-W）を出力し，インストールディレクトリ下の ¥tmp を使用します。

5 オブジェクトエクスポートで使用するファイル

この章では，オブジェクトエクスポートを実行する場合に必要な，制御ファイル，定義ファイルおよびオブジェクト指定ファイルの記述形式や文法について説明します。また，Windows の場合に必要に応じて設定する，環境変数ファイルの記述形式や文法について説明します。

5.1 使用するファイルの一覧

ここでは，オブジェクトエクスポートの実行時に使用するファイルについて説明します。

オブジェクトエクスポートの実行時に使用するファイルの一覧を次に示します。

表 5-1 オブジェクトエクスポートで使用するファイル

ファイル名称	機能	作成場所
制御ファイル （ファイル名は任意）	オブジェクトエクスポートの制御情報と，ユーザが追加するサブクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを記述するテキストファイルです。	任意
定義ファイル （ファイル名は任意）	GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルや入力データファイルには，定義ファイルに記述した名称を記述します。	任意
オブジェクト指定ファイル （ファイル名は任意）	DocumentBroker 文書空間から取得するオブジェクトの検索条件を指定するファイルです。	任意
オブジェクトローダ入力データファイル （ファイル名は任意）	オブジェクトローダの実行時に使用する入力データファイルです。DocumentBroker のデータベースに登録するプロパティの値を，制御ファイルで記述したプロパティの並び順に記述するテキストファイルです。複数のオブジェクトの登録を 1 トランザクションで実行させる，トランザクション単位の記述ができます。	任意
Windows の場合 環境変数ファイル （NTconfig.ini）	Windows 環境でのオブジェクトエクスポートの動作に必要な環境変数を設定するためのファイルです。このファイルは，DocumentBroker Object Loader をインストールする際に作成されます。ファイルの設定内容はカスタマイズできます。 UNIX の場合，このファイルはありません。	インストールディレクトリ ¥etc

5.2　ファイル形式と文法

ここでは，オブジェクトエクスポートを実行する場合に必要なファイルの形式と文法について説明します。

オブジェクトエクスポートを実行する場合に必要な，制御ファイル，定義ファイルおよびオブジェクト指定ファイルの記述形式や文法については，オブジェクトローダを実行する場合と同じです。「4.2　ファイル形式と文法」を参照してください。

なお，制御ファイル，定義ファイルを生成するために支援ユティリティ（入力ファイル生成ユティリティ）があります。詳細は，「3.　ファイル生成」を参照してください。

5.3 制御ファイル

ここでは，制御ファイルについて説明します。

制御ファイルはオブジェクトエクスポートの制御情報と，ユーザが追加するクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを記述するテキストファイルです。

5.3.1 記述形式

制御ファイルの記述形式を次に示します。

```
[Control]
エントリ名 = 設定値<改行コード>
エントリ名 = 設定値<改行コード>
          :
[DataMapping]
クラス名 = プロパティ名[,プロパティ名…]<改行コード>
クラス名 = プロパティ名[,プロパティ名…]<改行コード>
       :
[Export]
エントリ名 = 設定値
```

5.3.2 記述規則

制御ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- セクション名や各エントリの定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字間に半角空白またはタブを含むことができます。
- ＜改行コード＞は UNIX の場合は（0x0a），Windows の場合は（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 1 行の長さは半角空白，行間区切り文字を含めて 2,048 バイト以内です。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。
- DataMapping セクションは，8,192 バイト以内で記述してください。
- Export セクションの DocDir エントリに使用できる文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。

5.3.3　制御ファイルの構成

制御ファイルは，三つのセクションとセクションを構成する幾つかのエントリで構成されます。

次に制御ファイルに記述するセクションと機能の概要を示します。

表 5-2　制御ファイルのセクションと機能概要

セクション名	機能概要
Control	入力データファイルの記述形式やオブジェクトエクスポートの動作を制御する情報を定義する。
DataMapping	ユーザが追加するクラスに対するプロパティの並びや DocumentBroker 標準クラスにユーザが追加するプロパティの並びを定義する。
Export	文書の出力先を定義する。

（1）Control セクション

DocumentBroker Object Loader の動作を制御する情報を指定するセクションです。

（a）ErrorLog エントリ

エラーログファイルの出力先をファイルパスで指定します。詳細は「4.3.3(1)(a) ErrorLog エントリ」を参照してください。また，エラーログファイルの機能については，「1.5.2　ファイル環境（オブジェクトエクスポート）」を参照してください。

表 5-3 に指定値に対するオブジェクトエクスポートの動作を示します。省略時にはエラーログを出力しません。指定時でエラーが発生した場合は，エラーメッセージを出力してプログラムを終了します。

表 5-3　ErrorLog 指定値に対するオブジェクトエクスポートの動作

ErrorLog	状態	動作
省略時	－	エラーログファイルは出力しない
指定時	パス不正	エラー（DocumentBroker Object Loader 実行不可能）
	ファイル権限なし	エラー（DocumentBroker Object Loader 実行不可能）
	ファイルがすでに存在	上書きでエラーログファイルを出力する

（b）ErrorDataFile エントリ

エラーデータファイルの出力先をファイルパスで指定します。詳細は「4.3.3(1)(b) ErrorDataFile エントリ」を参照してください。また，エラーデータファイルの機能については，「1.5.2　ファイル環境（オブジェクトエクスポート）」を参照してください。

表 5-4 に指定値に対するオブジェクトエクスポートの動作を示します。省略時または指定時でエラーが発生した場合は，エラーメッセージを出力してプログラムを終了します。

表 5-4 ErrorDataFile 指定値に対するオブジェクトエクスポートの動作

ErrorDataFile	状態	動作
省略時	－	エラー（オブジェクトエクスポート実行不可能）
指定時	パス不正	エラー（オブジェクトエクスポート実行不可能）
	ファイル権限なし	エラー（オブジェクトエクスポート実行不可能）
	ファイルがすでに存在	上書きでエラーデータファイルを出力する

（c） ColumnSeparator エントリ

入力データファイル中のカラム間区切り文字（1 バイト）を引用符（"）で囲んで指定します。詳細は「4.3.3(1)(d)　ColumnSeparator エントリ」を参照してください。

（d） RecordSeparator エントリ

入力データファイル中の行間区切り文字（1 バイト）を引用符（"）で囲んで指定します。詳細は「4.3.3(1)(e)　RecordSeparator エントリ」を参照してください。

（e） MsgInterval エントリ

機能

処理の経過を示すメッセージ（KMBV11014-I）を出力する間隔を指定します。

指定形式

```
MsgInterval = 行数
```

指定規則

- 指定する行数はオブジェクトローダ入力データファイルのコマンド行数に相当します。0 ～ 2147483647 の間の 10 進表記で指定してください。
- 省略時は，「10000」が仮定されます。
- 0 を指定した場合はメッセージを出力しません。
- 数値以外や範囲外の数値が指定された場合は，省略時の値を仮定します。
- 指定値を小さくする程，処理の経過を示すメッセージが頻繁に出力されますので，登録するデータ行に応じて適切な値を設定してください。

（2） DataMapping セクション

機能

定義ファイルの ClassNameDefinition セクションで定義したクラスとプロパティの対応関係を定義するセクションです。オブジェクトエクスポートは，DataMapping セクションのエントリの右辺に記述したプロパティに対して，オブジェクトローダの入力データファイルにプロパティ値を出力します。

指定形式

エントリは定義ファイルの ClassNameDefinition セクションで定義したクラスの最上位クラスごとに次の形式で記述します。

クラス名=プロパティ名[,プロパティ名…]
プロパティ名::={プロパティ名 | VariableArray型プロパティ名 | システム定義プロパティ名}

指定規則

- プロパティ名には，定義ファイルの PropNameDefinition セクションで定義したプロパティ名を指定してください。クラス関連ファイルで関連を定義している場合は，「登録先クラス名 @」をプロパティ名に修飾して指定してください。
- 登録先クラス名は省略可能です。省略した場合に適用するクラスを次に示します。

表 5-5 登録先クラス名を省略した場合に適用するクラス

プロパティ種別	省略時のクラス
ユーザ定義	左辺に定義したクラス
システム定義	DocumentBroker 標準のクラス

登録先クラス名が定義ファイルに定義されていない場合や登録先クラスが存在しない場合は，エラーメッセージ（KMBV11049-E）を出力してプログラムを終了します。

- 定義ファイルの ClassNameDefinition セクションで一つの ConfigurationHistory クラスに対して複数のクラス名を定義している場合，クラス名の記述順序は，定義ファイルの ClassNameDefinition セクションの記述順序に合わせてください。
- VariableArray 型プロパティ名の場合，edmClass_Struct クラスのサブクラスを要素とするプロパティを指定するときは，「edmClass_Struct サブクラス名 .」をプロパティ名に修飾して指定してください。
- システム定義プロパティ名は，システム定義のプロパティを「**PROP_xxx**」という形式で指定してください。指定できるシステム定義プロパティについては，「5.6.2(4)(d) プロパティ値カラム」を参照してください。
- **PROP_CT_PATH** プロパティは指定できません。指定した場合，エラーメッセージ（KMBV11049-E）が出力されます。

指定例

DataMapping セクションの指定例を次に示します。

図 5-1 DataMapping セクションの指定例

```
[DataMapping]
VERDOC=V-AUTHOR,DOC@TITLE,USR_ARRAY.NAME,**PROP_RTYPE**,**PROP_CT**,**PROP_DCR**
          1         2          3
```

1：登録先クラスが省略されたユーザ定義プロパティなので，エントリ名に指定した最上位クラス「VERDOC」に格納されているプロパティ値を取得します。
2：VERDOC下のdmaClass_DocVersionクラスに格納されているプロパティ値を取得します。定義ファイルのClassNameDefinitionセクションにDOCを定義しておく必要があります。
3：VERDOCクラスの，USR_ARRAYプロパティを示すVariableArray型のNAMEプロパティに格納されているプロパティ値を取得します。定義ファイルのClassNameDefinitionセクションにUSR_ARRAYを定義しておく必要があります。

定義ファイルの ClassNameDefinition セクションで，一つの ConfigurationHistory クラスに対して複数のクラス名を定義しているときの DataMapping セクションの記述順序を図 5-2 に示します。

定義ファイルの ClassNameDefinition セクションの詳細については，「5.4.3(1) ClassNameDefinition セクション」を参照してください。

図 5-2 ClassNameDefinition セクションと DataMapping セクションの記述順序

定義ファイルのClassNameDefinition
セクションの記述内容

```
[ClassNameDefinition]
                    :
                    :
# 同じConfigurationHistoryクラスに2つの
# 名前を定義する。
CH1=01234567-8901-2345-6789-012345678903
CH2=01234567-8901-2345-6789-012345678903
                    :
                    :
```

制御ファイルのDataMapping
セクションの記述内容

```
[DataMapping]
                    :
                    :
# CH1のプロパティ
 CH1=D-AUTHOR,D-TITLE,D-COMMENT,…
# CH2のプロパティ
 CH2=D-AUTHOR,D-TITLE,D-COMMENT,…
                    :
                    :
```

(3) Export セクション

オブジェクトエクスポートの実行時に必要な制御情報を記述するセクションです。

(a) DocDir エントリ

文書のコンテントを格納するディレクトリパスを指定します。ここで指定したディレクトリパスが **PROP_CT** プロパティのパス名となります。ディレクトリパスは，絶対パス（255 バイト以内）で指定します。ディレクトリパスには，文書空間で使用する文字コード種別または ASCII コードを使用できます。文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。パスの指定方法については「4.2.2(3) パスの指定方法」を参照してください。

表 5-6 に，指定値に対するオブジェクトエクスポートの動作を示します。省略時または

指定時にエラーが発生した場合はエラーメッセージ（KMBV11421-E）を出力してプログラムを終了します。

表 5-6　DocDir 指定値に対するオブジェクトエクスポートの動作

DocDir	状態	動作
省略時	－	エラー（オブジェクトエクスポート実行不可能）
指定時	パスがすでに存在する	エラー（オブジェクトエクスポート実行不可能）
	パスが不正	エラー（オブジェクトエクスポート実行不可能）
	パスが存在しない	パスを作成する

実際のコンテントのパスの構成は次のとおりです。

DocDir エントリの値＋ユニークな名称＋ **PROP_CT** の値（ファイル名）

（凡例）Windows の場合

ユニークな名称：　"¥"+ 最上位オブジェクトの OIID
PROP_CT の値（ファイル名）：　"¥"+dmaProp_RetrievalName プロパティに設定されている値

- dmaProp_RetrievalName プロパティが未設定の場合は，「DocDir エントリの値＋ユニークな名称」が仮定されます。
- dmaProp_RetrievalName プロパティが設定されているが，コンテントが取得できない（コンテント格納先プロパティが未設定）場合，ファイルは作成されないで，「"/"+dmaProp_RetrievalName プロパティに設定されている値」が仮定されます。

UNIX の場合，ディレクトリの区切り文字が「/」になります。

5.3.4　制御ファイルの記述例

制御ファイルの記述例を次に示します。次の記述例は，Windows の場合です。UNIX の場合，ディレクトリの区切り文字が「/」になります。

なお，制御ファイルの Control セクションに，OIID_Count エントリ，EmptyValue エントリ，XmlBroker エントリがあっても，エクスポート実行時には影響しません。

```
[Control]
ErrorLog    = "DocBroker¥errorlog"
ErrorDataFile = "DocBroker¥errordata"
ColumnSeparator = ","
RecordSeparator = "¥n"
MsgInterval = 10000

[DataMapping]
JIKEN=C-AUTHOR,C-TITLE,C-DATE
```

```
KIAN=C-AUTHOR,C-TITLE,C-DATE,C-COMMENT,**PROP_DCR**
FOLD=C-AUTHOR,C-TITLE,C-DATE,C-COMMENT,**PROP_DCR**
PAT-CONFIG=D-AUTHOR,D-TITLE,D-COMMENT,
**PROP_DOC_CLASS**,**PROP_DCR**,**PROP_CT**
PAT-DOC=D-AUTHOR,D-TITLE,D-DATE,D-COMMENT,
D-STATE,**PROP_CTYPE**,**PROP_DCR**
MEMO=D-TITLE,D-DATE, **PROP_CTYPE**
I-DATA= I-AUTHOR, I-ARRAY
V-ARRAY=**PROP_OBJECT**,SET-INDEX,SET-DATA

[Export]
DocDir=C:¥tmp¥doc_data
```

5.4 定義ファイル

ここでは，定義ファイルについて説明します。

定義ファイルは，GUID 値と GUID 値に対する名称を記述するテキストファイルです。制御ファイルや入力データファイルには，定義ファイルに記述した名称を記述します。

5.4.1 記述形式

定義ファイルの記述形式を次に示します。

```
[ClassNameDefinition]
JIKEN_CONTAINER=01234567-8901-2345-6789-012345678901
VERDOC=01234567-8901-2345-6789-012345678902
DOC= 01A3A8C2-7AEC-11D1-A31B-0020AF9FBB1C
        :
[PropNameDefinition]
J-AUTHER=01234567-8901-2345-6789-012345678910
J-TITLE=01234567-8901-2345-6789-012345678911
```

5.4.2 記述規則

定義ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- セクション名や各エントリの定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字間に半角空白またはタブを含むことができます。
- ＜改行コード＞は UNIX の場合は（0x0a），Windows の場合は（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。

5.4.3 定義ファイルの構成

定義ファイルは，二つのセクションとセクションを構成する幾つかのエントリで構成されます。

定義ファイルのセクションの機能概要を次に示します。

表 5-7 定義ファイルのセクションと機能概要

セクション名	機能概要
ClassNameDefinition	ユーザが追加するクラスや入力データで使用するクラスを定義する。

セクション名	機能概要
PropNameDefinition	ユーザが追加するプロパティや入力データで使用するプロパティを定義する。

（1） ClassNameDefinition セクション

機能

制御ファイルの DataMapping セクションおよび入力データファイルに記述するクラス ID に対応するクラス名を定義するセクションです。

指定形式

エントリは定義するクラスごとに次の形式で記述します。

```
クラス名 = GUID値
```

指定規則

- クラス ID に記述される英大文字の A ～ F と英小文字の a ～ f は同一と扱います。
- 左辺にクラス名（xxxx），右辺にクラス名に対するクラス ID の GUID 値を記述し，左辺と右辺の関連を「=」で記述してください。
- クラス名（xxxx）は 128 バイト以内で記述します。先頭が「**」で始まる名称は指定できません。
- クラス名（xxxx）には，「@」，「=」，「,」，「.」および半角空白文字は使用できません。
- 同じ dmaClass_ConfigurationHistory クラスに混在する複数の種類のオブジェクトに対してオブジェクトエクスポートを実行する場合，オブジェクトの種類ごとにクラス名を分けて記述できます。
- ClassNameDefinition セクションの記述形式を示します。

```
xxxx=GUID値
```

（2） PropNameDefinition セクション

機能

制御ファイルの DataMapping セクションおよび入力データファイルで記述するプロパティ ID に対するプロパティ名を指定するセクションです。

指定形式

PropNameDefinition セクションの記述形式を示します。

```
yyyy = GUID値
```

指定規則

- プロパティ ID に記述される英大文字の A ～ F と英小文字の a ～ f は同一と扱い

ます。

- 左辺にプロパティ名（yyyy），右辺にプロパティ名に対するプロパティ ID の GUID 値を記述し，左辺と右辺の関連を「=」で記述してください。
- プロパティ名（yyyy）は 128 バイト以内で記述します。先頭が「**」で始まる名称は指定できません。
- プロパティ名（yyyy）には，「@」，「=」，「,」，「.」および半角空白文字は使用できません。

5.4.4 定義ファイルの記述例

定義ファイルの記述例を次に示します。

```
[ClassNameDefinition]
# Containerクラス
JIKEN=01234567-8901-2345-6789-012345678901
KIAN=01234567-8901-2345-6789-012345678902
# ConfigurationHistoryクラス
PAT-CONFIG=01234567-8901-2345-6789-012345678903
# DocVersionクラス
PAT-DOC=01234567-8901-2345-6789-012345678904
# DocVersionクラス
MEMO=01234567-8901-2345-6789-012345678905
# Independentクラス
I-DATA=01234567-8901-2345-6789-012345678906
# Structクラス
V-ARRAY=01234567-8901-2345-6789-012345678908
# ComponentDocVersionクラス
COMPONENT = bb6830ca-0bf0-11d2-9a68-0000e20838e7

[PropNameDefinition]
# フォルダ作成者，フォルダ名，作成日時，コメント
C-AUTHOR=01234567-8901-2345-6789-012345678911
C-TITLE=01234567-8901-2345-6789-012345678912
C-DATE=01234567-8901-2345-6789-012345678913
C-COMMENT=01234567-8901-2345-6789-012345678914

# 文書著者，文書名，作成日時，コメント，文書状態
D-AUTHOR=01234567-8901-2345-6789-012345678915
D-TITLE=01234567-8901-2345-6789-012345678916
D-DATE=01234567-8901-2345-6789-012345678917
D-COMMENT=01234567-8901-2345-6789-012345678918
D-STATE=01234567-8901-2345-6789-012345678919
# 作成者,データ
I-AUTHOR=01234567-8901-2345-6789-012345678920
I-ARRAY=01234567-8901-2345-6789-012345678907
# Struct型プロパティ
SET-INDEX=01234567-8901-2345-6789-012345678922
SET-DATA=01234567-8901-2345-6789-012345678923
# CSDVクラスのプロパティ
CSDV-P1=01234567-8901-2345-6789-012345678922
```

5.5 オブジェクト指定ファイル

ここでは，オブジェクト指定ファイルについて説明します。

DocumentBroker 文書空間から取得するオブジェクトの検索条件を指定するファイルです。オブジェクトエクスポートは，このファイルに指定したクラス中のプロパティ値が条件と一致するオブジェクトを取得します。

5.5.1 指定形式

オブジェクト指定ファイルの記述形式を次に示します。

```
Query/クラス名{/条件節並び}{/structサブクラス名}

条件節並び： 条件節1{論理演算子 条件節2}...{論理演算子 条件節5}
論理演算子： 論理積 | 論理和
論理積： &
論理和： |
条件節n： プロパティ名 比較演算子 プロパティ値
比較演算子： >|<|>=|<=|=|<>
structサブクラス名： structサブクラス名1 { struct 区切り文字 structサブクラス名n}
struct 区切り文字： 1文字以上の半角空白(0x20)
```

5.5.2 指定規則

オブジェクト指定ファイルの記述規則を次に示します。

規則

- プロパティ値に使用できる文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。
  文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。
- オブジェクト指定ファイル中に条件式が存在しない場合でもエラーにはなりません。
- クラス名，プロパティ名は，定義ファイルで定義した置き換え文字をそのまま記述します。
- struct サブクラス名（VariableArray 型のプロパティ名）は，定義ファイルの ClassNameDefinition セクションで定義された置き換え文字をそのまま記述します。
- 指定したクラス名，プロパティ名が定義ファイルに定義されていない場合はエラーメッセージ（KMBV11413-E）を出力してコマンド処理を終了します。指定したクラス名，プロパティ名に該当するものが DocumentBroker 文書空間に存在しない場合はエラーメッセージ（KMBV11021-E または KMBV11087-E）を出力してコマンド処理を終了します。

- struct サブクラス名は，クラス名で指定したクラスに属する struct サブクラスだけではなく，DCR でリンクした下位のクラスに属する struct サブクラスも指定します。指定した struct サブクラスが実際にクラスに属さない場合でもエラーにしないで，処理を続行します。
- struct サブクラス名を複数指定する場合は，半角空白（0x20）1 文字以上を区切り文字に指定してください。
- struct サブクラスに対しての対象指定はありません。全件を対象にします。
- クラス名に指定できるクラスの種別を次に示します。

  ・dmaClass_Container
  ・edmClass_ContainerVersion
  ・edmClass_IndependentPersistence
  ・dmaClass_ConfigurationHistory
  ・dmaClass_DocVersion

- 上に示したクラス種別以外のクラス種別が指定された場合はエラーメッセージ（KMBV11420-E）を出力してコマンド処理を終了します。
- 論理演算子は AND 条件と OR 条件を混在して使用することができます。
- 条件節は最大 5 個指定できます。5 個を超えて指定した場合はエラーメッセージ（KMBV11411-E）を出力してコマンド処理を終了します。複数指定する場合は論理演算子で条件節を連結します。例えば，次のように指定します。

  プロパティ ID1>= プロパティ値 1& プロパティ ID2<= プロパティ値 2

- 条件節並びを省略した場合は，条件式に指定したクラスのすべてのオブジェクトをオブジェクトローダ入力データファイルへの出力対象にします。
- 指定できるプロパティ値は，プロパティに対応するデータベースの型が文字列または整数のものです。それ以外の場合はエラーメッセージ（KMBV11415-E）を出力してコマンド処理を終了します。
- プロパティ値は，比較演算子と論理演算子または改行文字の前後の半角空白を除いた文字列をプロパティ値にします。プロパティ値が引用符で囲まれていない場合は，プロパティ値には「"」，「&」，「|」，「/」，「=」，「>」，「<」，半角空白が含まれる文字を指定することはできません。含まれていた場合はエラーメッセージ（KMBV11417-E）を出力してコマンド処理を終了します。
- 引用符をプロパティ値に含める場合は，プロパティ値を引用符で囲み，引用符を二つ続けて指定します。
- プロパティ値に NULL 値を指定する場合は，次のように引用符を二つ続けて記述します。

```
prop1=aabbcc          ← aabbcc というプロパティ値を指定する場合
prop2="aa&bb/cc"      ← aa&bb/cc というプロパティ値を指定する場合
```

```
prop3="aa""bb""cc"  ← aa"bb"cc というプロパティ値を指定する場合
prop4=""            ← プロパティ値に NULL を指定する場合
```

- プロパティ値に NULL を指定した場合，比較演算子には「=」と「<>」以外は指定できません。指定した場合はエラーメッセージ（KMBV11411-E）を出力してコマンド処理を終了します。
- 区切り文字を記述し，条件節並びを記述しない場合はエラーメッセージ（KMBV11411-E）を出力してコマンド処理を終了します。
- 比較演算子は「=」のほかに「>」，「<」，「>=」，「<=」，「<>」の指定が可能です。比較演算子の「<>」は否定を表します。
- 区切り文字（/），比較演算子（=，>，<，>=，<=，<>）の前後に半角空白（0x20）があってもかまいません。
- 引用符で囲まれているプロパティ値は文字列データとして扱います。したがって，条件節のプロパティのデータ型が数値型の場合に比較対象のプロパティ値が引用符で囲まれているときは，データ型不一致のため，エラーメッセージ（KMBV11417-E）を出力してコマンド処理を終了します。
- 条件節を指定しないで struct サブクラス名を指定する場合は，Query/ クラス名 // struct サブクラス名と記述します。
- 先頭 1 文字がシャープ（#）の行はコメント行とみなして処理の対象とはしません。
- 空白文字だけの行，または文字数がゼロの行は空行とみなして処理の対象とはしません。
- バージョン付き文書または文書だけをエクスポートする場合で，オブジェクト指定ファイルに指定するクラスに対して制御ファイルの DataMapping セクションに「**PROP_DCR**」が付加されているときは，「**PROP_DCR**」を削除する必要があります。

5.5.3 オブジェクト指定ファイルの記述例

オブジェクト指定ファイルの記述例を次に示します。

(a) "JIKEN" クラスの "DATE" プロパティに対し条件を満たすオブジェクトだけを生成対象とする指定

```
###コメント：この行の下の1行は空行です###

Query/JIKEN/DATE>=19990101&DATE<19990701
```

(b) "JIKEN" クラスの全オブジェクトを生成対象とする指定

```
Query/JIKEN
```

(c) "JIKEN" クラスと struct サブクラス "shinpan"，"yusen" の全オブジェクトを生成対象とする指定

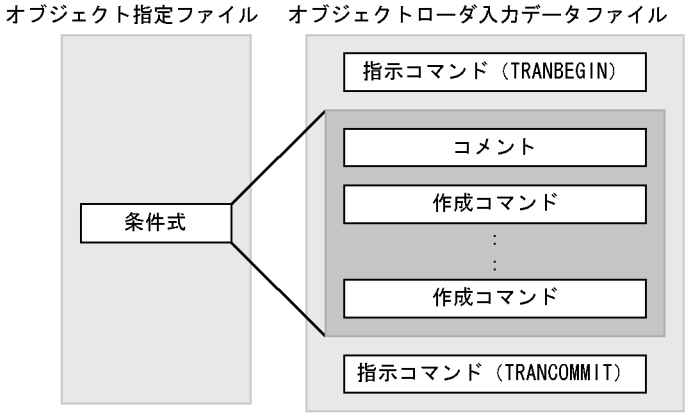
```
Query/JIKEN//shinpan yusen
```

5.6 オブジェクトローダ入力データファイル

ここでは，オブジェクトローダ入力データファイルについて説明します。

5.6.1 出力形式

オブジェクトエクスポートが出力するオブジェクトローダ入力データファイルの形式を次に示します。



5.6.2 出力内容

(1) ファイルの文字コード種別

オブジェクトローダ入力データファイルの文字コード種別は，文書空間で使用する文字コード種別または ASCII コードです。

文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けません。

(2) 指示コマンドの出力

- オブジェクトローダ入力データファイルの先頭行には指示コマンド「TRANBEGIN」文を出力し，最終行には指示コマンド「TRANCOMMIT」文を出力します。
- オブジェクト指定ファイルに指定した条件文に該当するオブジェクトが DocumentBroker 文書空間に存在しない場合は指示コマンドを出力しないで，エラーメッセージ（KMBV11412-E）を出力してプログラムを終了します。

(3) コメント行

処理対象のオブジェクト指定ファイルの条件式をコメントとして出力します。

(4) 作成コマンドの出力

- オブジェクト指定ファイルで指定された条件文に該当するオブジェクトの関連を解析

して作成コマンドを出力します。

- 作成コマンドの出力形式はカラム間を制御ファイルの Control セクションの ColumnSeparator エントリで指定された文字で区切り出力します。作成コマンドの各カラムの内容を次に示します。

表 5-8　入力データファイルのカラム

カラム位置	カラム名	説明
1 カラム目	コマンドカラム	作成コマンド名を出力する。
2 カラム目	ラベルカラム	ラベル名を出力する。
3 カラム目	クラス名カラム	出力するオブジェクトのクラス名を出力する。
4 カラム目以降	プロパティ値カラム	出力するオブジェクトのプロパティ値を DataMapping セクションの指定に従って出力する。

(a) コマンドカラム

オブジェクトエクスポートがコマンドカラムに出力するコマンドの一覧を次に示します。

表 5-9　作成されるコマンド一覧

生成コマンド名	説明
CREATE_RFCT	コンテナを作成する。
CREATE_DOC	文書を作成する。
CREATE_VRDOC	バージョン付き文書を作成する。
CREATE_DATA	独立オブジェクトを作成する。
CREATE_VARRAY	struct サブクラスを作成する。
CREATE_VRCV	構成管理の対象のバージョン付き文書を作成する。
CREATE_CV	構成管理コンテナを作成する。

(b) ラベル名カラム

- ラベル名は，各オブジェクトの OIID 値を使用します。ラベル名はオブジェクトローダ入力データファイル内でユニークになります。

(c) クラス名カラム

- オブジェクト指定ファイルで指定した条件に該当するオブジェクトのクラス ID に該当するクラス名を定義ファイルの ClassNameDefinition セクションから取得して出力します。
- 定義ファイルの ClassNameDefinition セクションに該当するクラス名が存在しない場合はエラーメッセージ（KMBV11416-E）を出力してコマンド処理を終了します。

(d) プロパティ値カラム

- 制御ファイルの DataMapping セクションの該当クラスエントリの右辺に定義しているプロパティの値を DocumentBroker 文書空間のオブジェクトから取得して出力しま

す。この時，制御ファイルの DataMapping セクションの該当クラスエントリの右辺に定義しているプロパティが文書空間に存在しない場合はエラーメッセージを出力してコマンド処理を終了します。

- オブジェクト間のクラスの関連を出力する場合，制御ファイルの DataMapping セクションの該当クラスエントリの右辺にシステム定義プロパティが定義されていない場合はエラーメッセージを出力してコマンド処理を終了します。例えば，次のような場合です。

  1. DocumentBroker 文書空間では，コンテナ B が直接型のコンテインメントでコンテナ A の Child の時にコンテナ B に **PROP_DCR** が指定されていない場合
  2. struct サブクラスの場合は，**PROP_OBJECT** が指定されていない場合

- プロパティ値を出力する場合にデータベースの列の型が文字列型（CHAR,MVARCHAR）の場合はデータの先頭および最終位置に引用符（"）を無条件に付加します。
- データに引用符（"）が含まれている場合は引用符（"）を二つ並べて（""）出力します。
- データに NULL 文字（0x00）が含まれている場合は NULL 文字を含めてデータ長分出力します（NULL 文字を終端文字として扱いません）。
- 取得したデータが NULL 値の場合は，データを出力しません。
- 文字列データの格納と出力結果を次に示します。「0x00」はバイナリコードのため，出力されたオブジェクトローダ入力データファイルの「0x00」は，テキストエディタ（vi など）では見えません。

表 5-10　文字列データの格納と出力結果

格納内容	出力結果
0x00 △△	"0x00 △△ "
A0x00 △	"A0x00 △ "
"0x00"	"""0x00"""
0x09 △△ （TAB）	"0x09 △△ "
NULL	出力なし
空文字（長さ 0 バイトの文字列）	"0x00"
0x00	"0x00"

（凡例）

△：半角空白

,：カラム間区切り文字

¥n：行間区切り文字

¥0：NULL 文字

- システムプロパティ値について

EDMLoad コマンドで必要なシステムプロパティが DataMapping セクションで指定されていない場合はエラーメッセージ（KMBV11049-E）を出力します。オブジェクトエクスポートが対象とするシステムプロパティを表 5-11 から表 5-14 に示します。

表 5-11　作成コマンドと DataMapping セクションに指定するプロパティ（1/4）

プロパティ種別	ユーザ定義	システム定義（**PROP_x** の x の部分）					
作成コマンド（CREATE_ は省略）		CT	RTYPE	CTYPE	DCR	DOC_CLASS	OBJECT
RFCT	○	×	×	×	○	×	×
DOC	○	●	●	○	○	×	×
VRDOC	○	●	●	○	○	●	×
DATA	○	×	×	×	×	×	×
VARRAY	○	×	×	×	×	×	●
VRCV	○	×	×	×	○	×	×
CV	○	×	×	×	○	×	×

（凡例）
●：必ず指定する　○：指定できる　×：エラー

表 5-12　作成コマンドと DataMapping セクションに指定するプロパティ（2/4）

プロパティ種別	システム定義（**PROP_x** の x の部分）		
作成コマンド（CREATE_ は省略）	VTMODE	CV_CLASS	VCR
RFCT	×	×	×
DOC	×	×	×
VRDOC	×	×	×
DATA	×	×	×
VARRAY	×	×	×
VRCV	×	●	×
CV	○※	×	○

（凡例）
●：必ず指定する　○：指定できる　×：エラー
注※　**PROP_VCR** プロパティが，DataMapping セクションに指定されているときに指定できます。**PROP_VCR** プロパティが，DataMapping セクションに指定されていない場合，エラーメッセージ（KMBV11049-E）を出力して処理を終了します。

表 5-13 作成コマンドと DataMapping セクションに指定するプロパティ（3/4）

プロパティ種別	システム定義（**PROP_x** の x の部分）				
作成コマンド（CREATE_ は省略）	ACL_OID	ACL_GID	ACL_OFLAG	ACL_GFLAG	ACL_EFLAG
RFCT	●	○	○	○	○
DOC	●	○	○	○	○
VRDOC	●	○	○	○	○
DATA	●	○	○	○	○
VARRAY	×	×	×	×	×
VRCV	●	○	○	○	○
CV	●	○	○	○	○

（凡例）
●：必ず指定する　○：指定できる　×：エラー

表 5-14 作成コマンドと DataMapping セクションに指定するプロパティ（4/4）

プロパティ種別	システム定義（**PROP_x** の x の部分）		
作成コマンド（CREATE_ は省略）	XML_MAP	XML_INDEX	PARSE_LEVEL
RFCT	×	×	×
DOC	○	○	○
VRDOC	○	○	○
DATA	×	×	×
VARRAY	×	×	×
VRCV	×	×	×
CV	×	×	×

（凡例）
○：指定できる　×：エラー

注　XML 文書の登録時，プロパティマッピングおよび構造指定全文検索ファイルの登録のどちらか，または両方を実行する場合，**PROP_XML_MAP** プロパティ，**PROP_XML_INDEX** プロパティおよび **PROP_PARSE_LEVEL** プロパティは，必ず三つ一組で制御ファイルに記述してください。一つまたは二つだけを指定した場合，エラーメッセージ（KMBV11049-E）を出力して，処理を終了します。

- **システムプロパティの出力内容**を次に示します。

表 5-15　システム定義プロパティの出力内容

システム定義プロパティ	出力内容
PROP_DCR	DCR を利用してリンクしている親コンテナをラベル名で出力する。DataMapping で指定されていて実際のリンクが存在しない場合はエラーメッセージを出力する。DataMapping で指定されていないが，実際にリンクが存在する場合は DCR プロパティ値を出力せず，リンク先のオブジェクトも出力しない。
PROP_CT	RetrievalName プロパティに格納されている文字列の前に制御ファイルの Export セクションに定義されているパス名＋生成したユニークな文字列を付加した文字列をプロパティ値として出力する。
PROP_RTYPE	Rendition オブジェクトに保持されている RenditionType（コンテントの表現形式を表す文字列）を出力する。
PROP_CTYPE	ContentTransfer または ContentReference オブジェクトに保持されているコンポーネントタイプ（コンテントを識別する文字列）を出力する。
PROP_DOC_CLASS	文書登録のためのオブジェクトを定義ファイルに定義した置き換え文字で出力する。
PROP_OBJECT	別表に登録する VariableArray 型のプロパティを持つオブジェクトのラベル名を出力する。
PROP_VTMODE	包含するオブジェクトの最新バージョンをトレースするモードを出力する（VTMode プロパティ）。 データベースの格納値 1：「FIX」を出力する。 2：「FLOAT」を出力する。
PROP_CV_CLASS	バージョン付き構成管理コンテナを作成するための ContainerVersion オブジェクトを，定義ファイルに定義した置き換え文字で出力する。
PROP_VCR	VersionTraceableContainmentRelationship オブジェクトを使用して構成管理コンテナに関連づけられているバージョン付き文書をラベル名で出力する。DataMapping で指定されたリンクが存在しない場合はエラーメッセージを出力する。DataMapping で指定されていないが，実際にはリンクが存在する場合は **PROP_VCR** プロパティの値を出力しない。また，リンク先のオブジェクトも出力しない。
PROP_ACL_OID	オブジェクトの所有者を出力する。
PROP_ACL_GID	オブジェクトのグループを出力する（NULL 値の場合は，「,,」を出力する）。
PROP_ACL_OFLAG	アクセス制御フラグ（ACFlag）の所有者のパーミッションを出力する。
PROP_ACL_GFLAG	アクセス制御フラグ（ACFlag）のグループのパーミッションを出力する。
PROP_ACL_EFLAG	アクセス制御フラグ（ACFlag）の全ユーザのパーミッションを出力する。

システム定義プロパティ	出力内容
PROP_XML_MAP	プロパティマッピングを行うかどうかを指定するコードを出力する。 ["NP"]（固定値）
PROP_XML_INDEX	インデクスファイルを作成するかどうかを指定するコードを出力する。 ContentIndexStatus プロパティの格納値 2：["OP"] を出力する。 2 以外：["NP"] を出力する。
PROP_PARSE_LEVEL	XML 文書の構文解析レベルを出力する。 「,0,」（固定値）

- アクセス制御フラグはデータベースに INTEGER 型で格納されていますが，それに対してオブジェクトエクスポートは，オブジェクトローダの指定形式に合わせてパーミッション文字列に変換して出力する必要があります。オブジェクトエクスポートでは基本パーミッション，FULL_CONTROL，NONE を使用してパーミッションを出力します。データベースから取得した値を基本パーミッション，FULL_CONTROL および NONE で AND をとり，各ビットに対応した文字列を「|」区切りで生成して，出力します。出力するパーミッション文字列を次に示します。オブジェクトエクスポートは基本パーミッションを使用して出力するため，オブジェクトローダを使用して初期ロードを行った時に指定したパーミッション文字列と異なる場合があります。

表 5-16　出力するパーミッション文字列

パーミッション文字列	パーミッション
PRIM_READ_PROPS	基本プロパティ参照権
PRIM_WRITE_PROPS	基本プロパティ更新権
PRIM_READ_CONTENTS	基本コンテント参照権
PRIM_WRITE_CONTENTS	基本コンテント更新権
PRIM_LINK	基本リンク権
PRIM_VERSION	基本バージョン権
PRIM_DELETE	基本削除権
NONE	権限なし
FULL_CONTROL	フルコントロール

（5）改行について

オブジェクトローダ入力データの行は，制御ファイルの Control セクションの RecordSeparator エントリで指定された文字を行の区切り文字として，データを切り出して出力します。

(6) エラー発生時のファイルの削除

オブジェクトローダ入力データファイルへの出力時にEレベルのエラーが発生した場合は，出力途中のオブジェクトローダ入力データファイルを削除します。出力途中のオブジェクトローダ入力データファイルは作成されません。

(7) ファイルの再作成範囲の軽減

オブジェクト指定ファイルに，一つの条件式に該当するオブジェクトが少なくなるような条件式を記述しておくと，オブジェクトエクスポートの出力量を複数回に分割できます。このように指定しておくことによって，オブジェクトローダ入力データファイルの作成に失敗したときのリカバリ作業（再実行で作成する範囲）を少なくすることができます。

(8) 運用時の注意

- オブジェクトエクスポートが出力するオブジェクトローダ入力データファイルのプロパティ値カラムは，取得したデータがNULL値の場合は，データを出力しません。HiRDBから取得したプロパティ値カラムがNULL値の場合，オブジェクトローダ入力データファイルには，区切り文字を続けて出力します。このオブジェクトローダ入力データファイルをオブジェクトローダでロードすると，オブジェクトローダではメタ情報を検索し，デフォルト値を取得してデータベース上に登録するので，データベースの値が変わる可能性があります。
- 予約語「P_CHID」，「P_CHID16」，または「P_CHID52」を指定してロードしたオブジェクトの場合，オブジェクトエクスポートが出力するオブジェクトローダ入力データファイルのプロパティ値カラムには，HiRDBから取得したデータがそのまま出力されます。
  このオブジェクトローダ入力データファイルをオブジェクトローダでロードすると，プロパティ値には，登録先クラスのオブジェクトと同時に作成する最上位クラスのオブジェクトのIDが設定されません。そのため，必要に応じて，オブジェクトローダ入力データファイルの指定を予約語に置き換える必要があります。

5.7 環境変数ファイル（Windows の場合）

Windows の場合，オブジェクトエクスポートが動作するために必要な情報を環境変数ファイル（NTconfig.ini）に設定します。

UNIX の場合，このファイルはありません。

5.7.1 記述形式

環境変数ファイルの記述形式を次に示します。

```
[ConfigObjLoader]
_HIEDMS_TRACE_DIR = トレースログファイルの
                    出力先ディレクトリのパス名 <改行コード>
_HIEDMS_TRACE_NUM = トレースログファイルの
                    ファイル数<改行コード>
_HIEDMS_TRACE_SIZE = トレースログファイルの
                     ファイルサイズ<改行コード>
_HIEDMS_TRACE_LEVEL = トレースレベル<改行コード>
```

5.7.2 記述規則

環境変数ファイルの記述規則を次に示します。

規則

- 文書空間で使用する文字コード種別が UTF-8 の場合は，印刷可能な ASCII コードで記述します。
- 各定義は 1 行で記述します。
- 1 文字目から記述します。
- 文字列の後ろに半角空白またはタブを含めることができます。
- 使用できる改行コードは（0x0d0a）です。
- 英字の大文字と小文字は区別されます。
- シャープ（#）で始まる行はコメント行になります。
- 文字列の途中に半角空白を含めることはできません。
- 同じエントリが複数存在する場合は最上位の指定が有効になります。また，ワーニングメッセージ（KMBV20101-W）が出力されます。
- [ConfigObjLoader] セクションのあとに記述されたエントリが有効になります。それ以外の場所に記述された文字列はコメントとして扱われます。
- 指定できるエントリ以外の文字列はコメントとして扱われます。
- NTconfig.ini にアクセス権限がない場合，NTconfig.ini ファイルがない場合，またはパス名が 256 バイト以上の場合は，エラーメッセージ（KMBV20102-E）を出力して処理を終了します。

注意事項

インストール直後の NTconfig.ini の初期状態では，パラメタの設定内容はコメント

行になっています。

5.7.3 環境変数ファイルに設定するパラメタ

環境変数ファイルに設定するパラメタを次に示します。必要なパラメタを NTconfig.ini ファイルに記述してください。

（1） _HIEDMS_TRACE_DIR

形式

```
_HIEDMS_TRACE_DIR = トレースログファイルの出力先ディレクトリのパス名
```

設定内容

- トレースログファイルの出力先ディレクトリのパス名を指定します。指定したディレクトリの下の ¥loader がトレースログファイルの出力先となります。
- このパラメタを省略した場合，インストールディレクトリ下の ¥spool¥loader が出力先ディレクトリになります。
- _HIEDMS_TRACE_DIR に指定したディレクトリが存在しない場合，または 256 文字以上の場合には，ワーニングメッセージ（KMBV20100-W）を出力し，インストールディレクトリ下の ¥spool¥loader を使用します。

（2） _HIEDMS_TRACE_NUM

形式

```
_HIEDMS_TRACE_NUM = トレースログファイルのファイル数
```

設定内容

- トレースの出力情報がトレースファイルのサイズの上限を超えた場合，トレースの出力先を別のファイルに切り替えられます。この場合の切り替えるファイル数を設定します。
- 設定できるファイルの数は 2 ～ 16 の値です。デフォルトは 2 です。
- 設定したファイル数までファイルが切り替わると，そのあとは最初に使用したファイルから順に上書きされます。

（3） _HIEDMS_TRACE_SIZE

形式

```
_HIEDMS_TRACE_SIZE = トレースログファイルのファイルサイズ
```

設定内容

トレースを出力するファイルのサイズを 4096 ～ 2147483647（バイト）で設定します。デフォルトは 1048576（1 メガバイト）です。

（4） _HIEDMS_TRACE_LEVEL

形式

```
_HIEDMS_TRACE_LEVEL = トレースレベル
```

設定内容

トレースの出力レベルを設定します。出力レベルには，0 または 10 が設定できます。デフォルトは 10 です。
それぞれを設定した場合に出力される情報を次に示します。

0 を設定した場合

- エラー情報
- サーバの開始と終了

10 を設定した場合

- トレースレベルが 0 の場合に出力される情報
- ユーザインターフェースの情報
- 他プログラムとのインターフェースの情報
- データベースへの接続と切断

6 コマンドリファレンス

この章では，DocumentBroker Object Loader で使用するコマンドの形式と文法について説明します。

実行コマンド一覧

DocumentBroker Object Loader が提供するコマンドの一覧を表 6-1 に示します。

表 6-1　DocumentBroker Object Loader が提供するコマンドの一覧

コマンド名	説明
EDMLodSetup（UNIX の場合）	DocumentBroker Object Loader の実行環境を作成する。または，DocumentBroker Object Loader の実行環境を削除する。
EDMLoad	オブジェクトローダユティリティを実行する。
EDMCrtLDF	定義ファイルを生成する。
EDMCrtLCF	制御ファイルを生成する。
EDMExport	オブジェクトエクスポートユティリティを実行する。

コマンドの形式

DocumentBroker Object Loader で使用するコマンドの入力形式，使用方法および注意事項について説明します。

入力形式

コマンドの入力形式を次に示します。

コマンド名 〔**オプション**…〕

コマンド名

コマンド名は，実行するコマンドのファイル名です。

オプション

オプションの入力形式の規則を次に示します。なお，説明文で使用する「$」はシェルのプロンプトまたはコマンドプロンプト，「cmd」はコマンド名を表します。

オプションの形式

オプションはハイフン（-）で始まる文字列で，フラグ引数を指定しない場合と，1 個のフラグ引数を指定する形式があります。

- 形式 1：- オプションフラグ
- 形式 2：- オプションフラグ＜空白またはタブ＞フラグ引数

（凡例）

オプションフラグ：1 文字の英数字（英大文字・小文字は区別される）

フラグ引数：オプションフラグに対する引数

オプションの指定規則

- 大文字と小文字を区別してください。
- オプションが複数指定できる場合，オプションの指定順序は任意です。
- フラグ引数を指定しないオプションフラグは，一つのハイフン（-）のあとにまとめて指定できません。
  誤った指定例：$ cmd -abc
  正しい指定例：$ cmd -a -b -c
- フラグ引数を必要とするオプションフラグのフラグ引数は省略できません。
- オプションフラグとフラグ引数の間には空白またはタブが必要です。
  誤った指定例：$ cmd -afile
  正しい指定例：$ cmd -a file
- 同じオプションフラグを 2 回以上指定できません。例えば，「$ cmd -a 1 -a 2」とは入力できません。同じオプションフラグを 2 回以上指定した場合は，エラーメッセージを出力してプログラムを終了します。
- ハイフンだけのオプションは入力できません。例えば，「$ cmd -」と入力すると「-」はコマンド引数とみなされます。

- 指定できないオプションを指定した場合は，エラーメッセージを出力してプログラムを終了します。

入出力

入力

入力は，すべてコマンドのオプションおよび引数の並びです。

出力

- コマンド処理が正常に終了したときの出力は，すべて標準出力に対して行います。
- 終了コードの一覧を表 6-2 に示します。

表 6-2　終了コード一覧

終了コード	意味
0	正常終了
1	警告付き正常終了
2	引数エラー
上記以外	そのほかのエラー

- コマンド処理がエラーになったとき（終了コマンドが「0」以外）は，すべて標準エラー出力に対して出力します。
- 引数エラー（終了コードが「2」）の場合は，標準エラー出力にコマンドの使用方法（USAGE）が出力されます。出力形式は次のとおりです。

出力形式

Usage:xxxx yyyy

- xxxx：コマンド名称が出力されます。
- yyyy：コマンドの指定形式が出力されます。

出力例

```
Usage:command -a -b option_arg_1 〔-c〕〔-d option_arg_2〕…
        {-e | -f option_arg_3}
```

EDMLodSetup（DocumentBroker Object Loader の実行環境作成・削除）（UNIX の場合）

機能

UNIX の場合，指定したディレクトリの下に DocumentBroker Object Loader の**実行環境を作成**します。また，DocumentBroker Object Loader の**実行環境を削除**します。

形式

```
EDMLodSetup  [-m { create | delete } ]  -d ディレクトリ名
```

オプション

-m { create | delete }

実行するモードとして，次のどちらかの文字列を指定します。オプションの省略時は，「-m create」が仮定されます。

create

DocumentBroker Object Loader の実行環境を作成する場合に指定します。

delete

DocumentBroker Object Loader の実行環境を削除する場合に指定します。

-d ディレクトリ名

- 実行環境を作成または削除するディレクトリを絶対パスで指定します。パスの指定方法については，「4.2.2(3)　パスの指定方法」を参照してください。
- 指定するディレクトリは必ず存在するディレクトリでなければなりません。また，インストールディレクトリを指定した場合は，エラーになります。
- 実行環境を作成する場合と削除する場合とでは，次のように処理の内容が異なります。

実行環境を作成する場合

1. -d オプションで指定したディレクトリの下に「bin」「lib」「adm」「spool」というディレクトリを作成します。
2. インストールディレクトリ（/opt/HiEDMS_Lod）の下の「bin」および「lib」ディレクトリに存在するファイルへのリンクを，1. で作成した 「bin」および「lib」の下に作成します。
3. インストールディレクトリ（/opt/HiEDMS_Lod）の下の「adm」ディレクトリに存在するファイルを，-d オプションで指定したディレクトリ下の「adm」の下にコピーします。

実行環境を削除する場合

-d オプションで指定したディレクトリの下から，「bin」「lib」「adm」「spool」ディレクトリを削除します。

注意事項

- 実行環境を作成する場合，このコマンドの実行前に実行環境を作成するディレクトリを作成しておく必要があります。
- -d オプションで指定するディレクトリは，DocumentBroker の他プログラムの実行環境とは別のディレクトリを指定してください。
- このコマンドは作成するすべてのディレクトリおよびファイルに対して，コマンドを実行したユーザおよびグループにアクセス権を与えます。
- 実行環境の作成中に，次に示す状態を検知した場合はエラーメッセージを出力して処理を終了します。
  - **実行環境を作成するディレクトリが存在しない。**
  - **実行環境を作成するディレクトリにアクセス権限がない。**
- この場合，処理が終了するまでの間（先に説明した状態が検知されるまでの間）に作成されたディレクトリやファイルは削除されません。したがって，「-m delete」を指定して，コマンドを再実行したあとに，実行環境を作成し直してください。
- 実行環境を削除する場合，-d で指定したディレクトリ下のファイルは，このコマンドが作成したものかどうかに関係なく削除されます。削除するディレクトリ下には不用意にファイルを作成しないでください。
- 実行環境を削除する場合は，-d で指定したディレクトリがあるかどうかはチェックされません。例えば，指定したディレクトリ下に「bin」ディレクトリだけ存在している場合は，そのディレクトリを削除して正常終了とします。ただし，削除対象となるディレクトリやファイルが一つも存在しない場合はエラーになります。
- 実行環境作成後，EDMLoad コマンドを動作させるために，環境変数（OBJLOADERDIR）に実行環境ディレクトリを指定してください。
- このコマンドで作成したディレクトリは移動しないでください。また，EDMLodSetup -m delete 以外の方法で，削除しないでください。
- このコマンドで作成したディレクトリには，別のディスクへのリンクを作成しないでください。
- 一つの文書空間に対して複数の DocumentBroker の実行環境を構築している場合，このコマンドを実行する前に，実行環境識別子が 0 の実行環境であることを確認してください。
  実行環境識別子が 0 以外の実行環境であっても，EDMLodSetup コマンドは正常終了します。ただし，作成した DocumentBroker Object Loader の実行環境では，EDMLodSetup 以外のコマンドを使用できません。この場合，DocumentBroker Object Loader の実行環境を削除してください。その後，実行環境識別子が 0 の環境に DocumentBroker Object Loader の実行環境を作成して使用してください。

EDMLoad（オブジェクトローダの実行）

機能

DocumentBroker サーバに**オブジェクトを登録**します。

形式

High-end Option をインストールしていない場合

```
EDMLoad　入力データファイル [-c 制御ファイル] [-d 定義ファイル] [-A] [-M]
```

High-end Option をインストールしている場合

```
EDMLoad　入力データファイル [-c 制御ファイル] [-d 定義ファイル] [-A] [-S | -M]
```

オプション

入力データファイル

入力データファイルのパス名を指定します。

-c 制御ファイル

制御ファイルのパス名を指定します。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_control.txt

Windows の場合：.¥EDM_lod_control.txt

-d 定義ファイル

定義ファイルのパス名を指定します。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_define.txt

Windows の場合：.¥EDM_lod_define.txt

-A

アクセス制御情報（ACL）データの登録機能を使用する場合に指定します。

-S

High-end Option をインストールしている場合で，EDMLoad コマンドを非同時実行モードで，単独で実行するときに指定します。一つだけの EDMLoad コマンドを実行する場合，"-S" オプションを指定することをお勧めします。

High-end Option をインストールしている場合で，EDMLoad コマンドの並列実行をするときには，"-S" オプションを省略してください。

"-S" オプションおよび High-end Option の詳細については，「付録 E　High-end Option」を参照してください。同時実行モードおよび非同時実行モードの詳細については，「付録 F　同時実行機能の実行環境と運用上の注意事項」を参照してくださ

い。
High-end Option をインストールしていない場合，このオプションを指定するとエラーメッセージ（KMBV11002-E）を出力してプログラムを終了します。

-M

EDMLoad コマンドを同時実行モードで実行するときに指定します。このオプションを指定すると，DocumentBroker サーバが起動しているかどうかに関係なく同時実行モードで動作します。ただし，非同時実行モードと同時実行モードの並列実行はできません。同時実行モードおよび非同時実行モードの詳細については，「付録 F 同時実行機能の実行環境と運用上の注意事項」を参照してください。

パスの指定方法

パスの指定方法については，「4.2.2(3) パスの指定方法」を参照してください。

注意事項

- コマンド実行時には，データベースを起動しておく必要があります。
- 一つの文書空間に対して複数の DocumentBroker の実行環境を構築している場合，このコマンドを実行する前に，実行環境識別子が 0 の実行環境であることを確認してください。
  実行環境識別子が 0 以外の実行環境で，このコマンドを実行すると，エラーメッセージ（KMBV11037-E）を出力し，処理を終了します。

EDMCrtLDF（定義ファイル生成）

機能

DocumentBroker Object Loader の**定義ファイルを生成**します。

形式

```
EDMCrtLDF ユーザID [-d 定義ファイル]
```

オプション

ユーザ ID

HiRDB にテーブルを作成した認可識別子を指定します。

-d 定義ファイル

生成する定義ファイルのパス名を指定します。パスの指定方法については，「4.2.2(3)　パスの指定方法」を参照してください。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_define.txt

Windows の場合：.¥EDM_lod_define.txt

実行条件

- このコマンドを実行できるユーザは，DocumentBroker サーバのシステム管理者です。
- このコマンドを実行する前に DocumentBroker サーバの動作環境が整っている必要があります。
- このコマンドを実行するときに，データベースを起動しておく必要があります。

注意事項

- データベースに登録されている DocumentBroker のメタ情報から取得する，このコマンドの出力結果の定義ファイルのクラス名およびプロパティ名を修正する場合は，規則に従って修正してください。定義ファイルの各セクションの指定形式および指定規則については，「4.4　定義ファイル」を参照してください。
- 一つの文書空間に対して複数の DocumentBroker の実行環境を構築している場合，このコマンドを実行する前に，実行環境識別子が 0 の実行環境であることを確認してください。

  実行環境識別子が 0 以外の実行環境で，このコマンドを実行すると，エラーメッセージ（KMBV11037-E）を出力し，処理を終了します。

EDMCrtLCF（制御ファイル生成）

機能

DocumentBroker Object Loader の制御ファイルと入力データファイルを生成します。

形式

```
EDMCrtLCF ユーザID クラス関連ファイル [-c 制御ファイル]
          [-i 入力データファイル] -L
```

オプション

ユーザ ID

HiRDB にテーブルを作成した認可識別子を指定します。

クラス関連ファイル

クラス関連ファイルのパスを指定してください。ファイルの定義内容は，このコマンドの「クラス関連ファイルの作成方法」を参照してください。

-c 制御ファイル

生成する制御ファイルのパスを指定してください。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_control.txt

Windows の場合：.¥EDM_lod_control.txt

-i 入力データファイル

生成する入力データファイルのパスを指定してください。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_input.txt

Windows の場合：.¥EDM_lod_input.txt

なお，生成する入力データファイルは，オブジェクトローダで使用する入力データファイルの記述形式に従って編集する必要があります。

-L

制御ファイルおよび入力データファイルを生成するためのオプションです。このオプションは省略できません。生成する制御ファイルの詳細については，「付録 B　制御ファイルの状態表」を参照してください。

なお，DocumentBroker サーバ環境がアクセス制御情報（ACL）対応の場合は，（**PROP_ACL_OID**）プロパティを制御ファイルの DataMapping セクションに出力します。

パスの指定方法

パスの指定方法については，「4.2.2(3)　パスの指定方法」を参照してください。

クラス関連ファイルの作成方法

文書クラスに対して最上位クラスと登録先クラスの関連を定義するファイルがクラス関連ファイルです。登録先クラスが保有するユーザプロパティを，制御ファイルの DataMapping セクションに出力する場合に定義します。

表 6-3 に，クラス関連ファイルに定義が必要な文書クラスおよび文書クラスに対する最上位クラスと登録先クラスを示します。

注意

最上位クラスと登録先クラスの関連を定義する必要がない場合は，空ファイルをクラス関連ファイルとして EDMCrtLCF に指定する必要があります。

表 6-3　クラスの関連を定義する文書とクラスの関係

文書	最上位クラス	登録先クラス
バージョン付き文書	VRDOC の dmaClass_ConfigurationHistory	dmaClass_DocVersion
構成管理用バージョン付き構成管理コンテナ	VRCV の dmaClass_ConfigurationHistory	edmClass_ContainerVersion

クラス関連ファイルの記述形式を示します。

```
[VRDOC]
VRDOCのConfigurationHistoryクラスのエントリ名/DocVersionクラスのエントリ名

[VRCV]
VRCVのConfigurationHistoryクラスのエントリ名/ContainerVersionクラスのエントリ名
```

セクションの記述方法

- CREATE_xxx のコマンド名の xxx を角括弧（[ ]）で囲みます。
- 構成管理用（CREATE_VRCV 用）のユーザクラスを生成したい場合は，VRCV セクションを定義し，最上位クラスを必ず指定してください。なお，登録先クラスにユーザプロパティを指定しない場合は，登録先クラスを省略できます。

クラスの関連の記述方法

- 該当するセクションにスラント（/）区切りでエントリ名を記述します。エントリ名は，定義ファイル生成コマンドで生成された ClassNameDefinition セクションのエントリ名を指定してください。
- システム標準のクラスを使用している場合は，システムクラスのエントリ名を指定します。システムクラスとエントリ名の対応を表 6-4 に示します。なお，全文検索機能付き DocVersion は必ずユーザクラスなのでシステムクラスのエントリ名はありません。
- エントリ名に「[」，「]」および「/」は指定できません。

表 6-4 システムクラスとエントリ名の対応

システムクラス	定義ファイルで生成するエントリ名
VRDOC の dmaClass_ConfigurationHistory	dmaClass_CH_vrdoc
dmaClass_DocVersion	dmaClass_DV
edmClass_ComponentDocVersion	dmaClass_CD
edmClass_ContainerVersion	edmClass_CV
構成管理用の dmaClass_ConfigurationHistory	dmaClass_VRCH

実行条件

- コマンドを実行するためにはクラス関連ファイルを作成しておく必要があります。
- このコマンドを実行できるユーザは，DocumentBroker サーバのシステム管理者です。
- このコマンドを実行する前に DocumentBroker サーバの動作環境が整っている必要があります。
- このコマンドを実行するときに，データベースを起動しておく必要があります。

注意事項

- 「-L」オプションは必ず指定してください。このオプションを指定しないでコマンドを実行するとエラーになります。
- DocumentBroker サーバ環境が ACL 対応の時，出力した制御ファイルを使用して EDMLoad コマンドを実行するとエラーが発生して，処理が中断するので注意してください。
- 制御ファイルの各セクションの指定形式および指定規則については「4.3　制御ファイル」を参照してください。
- 一つの文書空間に対して複数の DocumentBroker の実行環境を構築している場合，このコマンドを実行する前に，実行環境識別子が 0 の実行環境であることを確認してください。
  実行環境識別子が 0 以外の実行環境で，このコマンドを実行すると，エラーメッセージ（KMBV11037-E）を出力し，処理を終了します。

EDMExport（オブジェクトエクスポートの実行）

機能

データベースからオブジェクトを抽出し，オブジェクトローダに入力する**入力データファイルを生成**します。

形式

```
EDMExport  オブジェクト指定ファイル
           -o オブジェクトローダ入力データファイル
           [-c 制御ファイル] [-d 定義ファイル]
```

オプション

オブジェクト指定ファイル

オブジェクト指定ファイルのパス名を指定します。

-o オブジェクトローダ入力データファイル

オブジェクトローダ入力データファイルのパス名を指定します。

-c 制御ファイル

制御ファイルのパス名を指定します。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_control.txt

Windows の場合：.¥EDM_lod_control.txt

-d 定義ファイル

定義ファイルのパス名を指定します。オプションの省略時は，次の値が仮定されます。

UNIX の場合：./EDM_lod_define.txt

Windows の場合：.¥EDM_lod_define.txt

パスの指定方法

パスの指定方法については，「4.2.2(3)　パスの指定方法」を参照してください。

注意事項

- 一つの文書空間に対して複数の DocumentBroker の実行環境を構築している場合，このコマンドを実行する前に，実行環境識別子が 0 の実行環境であることを確認してください。
  実行環境識別子が 0 以外の実行環境で，このコマンドを実行すると，エラーメッセージ（KMBV11037-E）を出力し，処理を終了します。

7 トラブルシュート機能

この章では，DocumentBroker Object Loader を保守運用するためのトラブルシュート機能とエラーメッセージについて説明します。

7.1 トラブルシュート機能の概要

ここでは，DocumentBroker Object Loader を保守運用するために必要なトラブルシュート機能の概要について説明します。

(1) メッセージ

障害対応の出力メッセージのうち重要度の高いものは，UNIX の場合，syslog ファイルに，Windows の場合，イベントビューアのアプリケーションログに出力されます。

また，コマンド関連のメッセージは，このほかに，標準出力や標準エラー出力にも出力されます。

DocumentBroker Object Loader のメッセージプリフィクスは，「KMBV」です。

(2) トレース出力

トラブルの切り分け，原因究明の補助のため，実行中の重要な情報はトレース情報としてファイルに出力します。出力量によってトレースレベルを設定できます。

トレースレベルについては，DocumentBroker Object Loader の起動ごとに設定を変更できます。

(3) エラーデータファイル

エラーが発生したトランザクションを，入力データファイルと同形式でファイル出力します。

(4) エラーログファイル

オブジェクトローダまたはオブジェクトエクスポート実行時のエラー，ワーニング，インフォメーション情報を出力します。制御ファイルの ErrorLog エントリにエラーログファイルの出力先を指定した場合だけ出力します。

7.2 エラーデータファイルの内容

ここでは，エラーデータファイルの内容について説明します。

- エラーデータファイルには，オブジェクトの登録に失敗した場合，失敗した入力データファイルの定義が入力データファイルと同一形式で出力されます。
- 出力されたエラーデータファイルは，エラーを修正後，DocumentBroker Object Loader 再実行時の入力データファイルとして使用できます。この場合，親オブジェクトを [Label/ ラベル名] で指定して，親オブジェクトのラベルがなくなった個所は，直前に「SELECT_OBJECT」で親オブジェクトをラベル指定するコマンドを追加します。
- オブジェクトローダの場合，TRANBEGIN と TRANCOMMIT で囲まれたオブジェクトでエラーが発生した場合は，TRANBEGIN から TRANCOMMIT までをコメント行や空行も含めて出力します。TRANBEGIN と TRANCOMMIT で囲まれていない場合は，オブジェクトだけを出力して，前の行のコメント行や空行は出力しません。
- オブジェクトエクスポートの場合，エラーが発生したオブジェクト指定ファイルの条件式だけを出力します。

7.3 エラーログファイルの内容

ここでは，エラーログファイルの内容について説明します。

- 制御ファイルの ErrorLog エントリでエラーログファイルの出力を指定した場合，オブジェクトロードまたはオブジェクトエクスポート実行時のエラー，ワーニング，インフォメーション情報をエラーログファイルに出力します。
- エラーログファイルに出力される入力データに関するエラー情報とエラーデータファイルの内容を突き合わせて，入力データのエラー状況を確認できます。
- UNIX の場合，エラーログファイルの出力先に，/dev/console を指定すると，/dev/console がロックされた状態になり，そのあとの出力の際，出力側でフリーズする場合があるので注意してください。

7.4　トレースログファイルの内容

ここでは，トレースログファイルの内容について説明します。

DocumentBroker Object Loader 実行中の重要な保守情報はトレースログファイルに出力されます。このファイルは主に DocumentBroker Object Loader の保守に使用します。

(1) トレースログファイルの出力形式

トレースログファイルの出力形式を図 7-1 に示します。

図 7-1　トレースログファイルの出力形式

```
     yyyy/mm/dd hh:mm:ss.sss                pid      tid      message-id  message(LANG=ja_JP.SJIS)
0760 1999/03/01 18:29:17.433 DocumentBroker 0000741E 00000001 KMBV11000-I オブジェクトローダの処理が終了しました。
登録：２件
エラー：１件
実行行番号：４
```

トレースログの表示内容を表 7-1 に示します。

表 7-1　トレースログの表示内容

項目	表示内容
−	行番号
yyyy/mm/dd hh:mm:ss.sss	実行日付および時刻
−	製品名称
pid	プロセス ID
tid	スレッド ID
message-id	メッセージ ID
message(LANG=ja_JP.SJIS)	メッセージ

(2) 出力先ディレクトリ

トレースフログファイルは，UNIX の場合は環境変数「_HIEDMS_TRACE_DIR」，Windows の場合は環境変数ファイルのパラメタ「_HIEDMS_TRACE_DIR」にフルパス形式で指定したディレクトリに出力されます。デフォルトの出力先ディレクトリは，次のとおりです。

トレースログファイルの出力先ディレクトリ

UNIX の場合

- $OBJLOADERDIR/spool/loader
- $OBJLOADERDIR/spool/loader/export（オブジェクトエクスポートの場合）

Windows の場合

- （DocumentBroker Object Loader のインストールディレクトリ）¥spool¥loader
- （DocumentBroker Object Loader のインストールディレクトリ）¥spool¥loader¥export（オブジェクトエクスポートの場合）

注意事項

「_HIEDMS_TRACE_DIR」に指定する出力先ディレクトリは相対パス指定はできません。ただし，指定が「.」のときだけ，カレントディレクトリの指定であるとみなします。

(3) 出力ファイル名

トレースログファイルの出力ファイル名は，環境変数（UNIX の場合）および環境変数ファイルのパラメタ（Windows の場合）に指定できません。次のファイル名の形式で出力されます。

トレースログファイルの出力ファイル名

```
EDMRasTrace"PID"_"NO".log
```

（PID：プロセス ID，NO：ファイル番号）

(4) 切り替えファイル数の指定

トレースの出力情報がトレースファイルのサイズの上限を超えた場合に，出力先を切り替えるファイルの数を指定できます。UNIX の場合は環境変数「_HIEDMS_TRACE_NUM」，Windows の場合は環境変数ファイルのパラメタ「_HIEDMS_TRACE_NUM」に，切り替えることができるファイルの数を 2 ～ 16 の値で指定します。デフォルトは 2 です。

トレースの出力処理で切り替ファイルの数だけファイルが切り替わると，そのあとは最初に使用したファイルから順に上書きされ再利用されます。

(5) ファイルサイズの指定

トレースを出力するファイルのサイズを，UNIX の場合は環境変数「_HIEDMS_TRACE_SIZE」，Windows の場合は環境変数ファイルのパラメタ「_HIEDMS_TRACE_SIZE」に 4096 ～ 2147483647（バイト）の値で指定します。デフォルトは 1048576（1 メガバイト）です。

(6) トレースの出力レベルの指定

トレースの出力レベルを変更できます。UNIX の場合は環境変数「_HIEDMS_TRACE_LEVEL」，Windows の場合は環境変数ファイルのパラメタ「_HIEDMS_TRACE_LEVEL」にトレースレベルを指定します。デフォルトは 10 です。トレースレベルと出力情報を表 7-2 に示します。

表 7-2 トレースレベルと出力情報

トレースレベル	出力情報
0	エラー情報 サーバの開始 / 終了
10（デフォルト）	トレースレベルが 0 の場合に出力される情報 ユーザインターフェースの情報 他プログラムとのインターフェースの情報 データベースへの接続 / 切断

7.5 メッセージの形式

ここでは DocumentBroker Object Loader のメッセージについて説明します。

7.5.1 メッセージの出力言語の設定

DocumentBroker Object Loader のメッセージの出力言語には，日本語と英語があります。

日本語で出力するか，英語で出力するかを設定する方法を次に示します。

AIX の場合

LANG 環境変数で設定した値に従います。
設定値が「Ja_JP」の場合，日本語のメッセージを出力します。
設定値が「Ja_JP」以外の場合は，英語のメッセージを出力します。
LANG 環境変数の設定については，「2.3(1) 環境変数の設定」を参照してください。

Windows の場合

インストールした OS に従います。
日本語版 OS をインストールした場合，日本語のメッセージを出力します。
日本語版以外の OS をインストールした場合，英語のメッセージを出力します。
なお，日本語のメッセージの文字コード種別は Shift-JIS です。

7.5.2 メッセージの出力形式

DocumentBroker Object Loader のメッセージの出力形式は次のとおりです。

```
メッセージID　メッセージテキスト
```

7.5.3 メッセージの記述形式

(1) 記述形式

「7.6　メッセージの詳細」でのメッセージの記述形式は次のとおりです。

```
KMBVnnnnn-i 〈Y〉
日本語のメッセージテキスト
英語のメッセージテキスト
        （必要に応じて）出力されるメッセージテキストに対する補足説明
        (S)
            DocumentBroker Object Loaderの動作（処理）
        (O)
            ユーザの対処方法
```

(2) メッセージ ID の記号の説明

メッセージ ID の記号の意味を説明します。

KMBV

メッセージを出力したプログラム（オブジェクトローダ）を表します。

nnnnn

メッセージの番号です。それぞれのメッセージに 5 けたの固有の番号が付いています。

i

メッセージの種類を表します。

I

システムの動作を通知します。

W

処理は続行しますが，障害が発生したので警告します。

E

障害が発生したので，処理を中断します。

<Y>

メッセージの出力先を表します。

C

syslog ファイル（UNIX の場合）
イベントビューア（Windows の場合）

P

標準エラー出力

F

トレースログファイル

U

エラーログファイル

注意

この記号はメッセージ ID には付加されていません。このマニュアル内で使用している付加情報です。

（3） メッセージテキストの説明

出力されるメッセージのテキストを示します。なお，メッセージテキストの %n（n は挿入句の順番）などの文字は，メッセージが出力される状況によって変わる値です。

また，文書空間で使用する文字コード種別が UTF-8 の場合，メッセージテキストの %n に UTF-8 の文字列が出力されることがあります。UTF-8 の文字列の内容を確認するには，UTF-8 に対応したエディタなどで参照してください。

7.6 メッセージの詳細

KMBV11000-I <C,P,F,U>

オブジェクトローダの処理が終了しました。

登録：%1 **件**

エラー：%2 **件**

実行行番号：%3

EDMLoad ended.

Entry : %1

Error : %2

Execute Line : %3

入力データファイルの %3 行まで実行しました。その結果，%1 件が登録でき，エラーによって %2 件が登録できませんでした。

（S）

処理を終了します。

（O）

―

KMBV11001-E <P,F>

他のプログラムが実行中のため，コマンドが実行できません。

Unable to execute this command because other program is executing.

DocumentBroker サーバ環境下で，DocumentBroker サーバ開始中に本機能を実行しようとしたか，本機能を複数実行しようとしています。または，High-end Option がインストールされている状態で同時実行機能を使用する EDMLoad と，使用しない EDMLoad での複数実行をしようとしています。

または，DocumentBroker Library の環境で，ライブラリを利用した UAP 実行中に本機能を実行しようとしたか，本機能を複数実行しようとしています。

（S）

処理を終了します。

（O）

DocumentBroker サーバを停止後，または実行中の処理が終了してから再度実行してください。

KMBV11002-E <P>

指定したコマンドの形式に誤りがあります。コマンドの形式は次のとおりです。

EDMLoad **入力データファイル** 〔-c **制御ファイル**〕〔-d **定義ファイル**〕〔-A〕〔-M〕

Error exists in command line, usage is as follows

Usage: EDMLoad InputDataFile 〔-c ControlFile〕〔-d DefineFile〕〔-A〕〔-M〕

指定したコマンド形式に誤りがあります。

(S)

処理を終了します。

(O)

正しいコマンド形式を指定して再度実行してください。

KMBV11003-E <P,F>

環境変数の指定に誤りがあります。

環境変数名：%1

Environment variable is invalid.

Environment variable name : %1

環境変数（%1）が定義されていないか，指定値が誤っているため，コマンドが実行できません。オブジェクトローダの実行には，次の環境変数を正しく定義する必要があります。

AIX の場合

- LANG
- LIBPATH
- OBJLOADERDIR
- DOCBROKERDIR

Windows の場合

- PATH
- DOCBROKERDIR

(S)

処理を終了します。

(O)

環境変数を定義後，再度実行してください。

KMBV11004-E <C,P,F>

アクセス制御情報機能付きで，コマンドを実行できません。

Unable to execute this Command with Access Control List function.

docaccess.ini または docspace.ini ファイルの参照権限がないか，または DocumentBroker サーバの文書空間がアクセス制御（ACL）対応の環境になっていないので，アクセス制御情報機能付きで EDMLoad コマンドを実行できません。

（S）

処理を終了します。

（O）

docaccess.ini または docspace.ini ファイルに対して参照権限があるか確認後，再実行してください。

KMBV11005-I　<C,P,F,U>

オブジェクトローダの処理が終了しました。

正常：%1 **件**

エラー：%2 **件**

実行行番号：%3

OIID **使用数：**%4

EDMLoad ended.

Normal : %1

Error : %2

Execute Line : %3

OIID_UseCount : %4

入力データファイルの %3 行まで実行しました。その結果，%1 件が正常で，%2 件がエラーでした。
また，正常に実行できたオブジェクトは OIID を %4 個使用しました。

（S）

処理を終了します。

（O）

―

KMBV11006-I　<C,P,F,U>

OIID_Count **の算出処理が終了し，オブジェクトの登録を開始します。**

Calculation processing of OIID_Count is finished and starts registration of an object.

OIID_Count 自動算出の処理が終了し，オブジェクトの登録が開始されます。

（S）

処理を続行します。

（O）

―

KMBV11007-I <P,F,U>

%1 行の処理を終了しました。

%1 Line ended.

%1 行の処理が終了しました。

（S）

処理を続行します。

（O）

―

KMBV11008-E <C,P,F>

環境定義ファイル [docaccess.ini] **に指定したパーミッションの値が不正です。**

エントリ：%1

パーミッション：%2

Invalid property Permission value exists environment file [docaccess.ini].

entry : %1

Permission : %2

DocumentBroker サーバ が提供する docaccess.ini ファイルのアクセス制御フラグに指定しているパーミッションが不正のため，アクセス制御情報機能付きで EDMLoad コマンドを実行できません。

（S）

処理を終了します。

（O）

docaccess.ini ファイルのアクセス制御フラグに指定しているパーミッションを正しい値に変更してから，再度実行してください。

KMBV11009-E <C,P,F>

コマンドオプションに「-A」が指定されていないので，コマンドを実行できません。

Unable to execute this Command due to no [-A] Option.

DocumentBroker サーバの文書空間がアクセス制御（ACL）対応の環境になっています。

アクセス情報機能付きでコマンドを実行してください。

（S）

処理を終了します。

（O）

コマンドオプションに「-A」を付加したあと，再度実行してください。

KMBV11010-E <P>

指定したコマンドの形式に誤りがあります。コマンドの形式は次のとおりです。

EDMLoad **入力データファイル** [-c **制御ファイル**] [-d **定義ファイル**] [-A] [-S | -M]

Error exists in command line, usage is as follows

Usage: EDMLoad InputDataFile [-c ControlFile] [-d DefineFile] [-A] [-S | -M]

指定したコマンド形式に誤りがあります。

（S）

処理を終了します。

（O）

正しいコマンド形式を指定して再度実行してください。

KMBV11011-E <C,P,F>

入力データファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of InputData file.

Reason code : %1

入力データファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	入力データファイルが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルに参照権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11012-E <C,P,F>

制御ファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of Control file.

Reason code : %1

制御ファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	制御ファイルが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルに参照権限がありません。

(S)

処理を終了します。

(O)

エラーの要因を取り除いてください。

KMBV11013-E <C,P,F>

エラーログファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of ErrorLog file.

Reason code : %1

エラーログファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	制御ファイルの ErrorLog エントリに指定したパスが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ディレクトリにファイルの作成権限がありません。
4	ファイルに書き込み権限がありません。

(S)

処理を終了します。

(O)

エラーの要因を取り除いてください。

KMBV11014-I <P,F,U>

%1 行の処理が終了しました。

%1 Line ended.

%1 行の処理が終了しました。

（S）

処理を続行します。

（O）

―

KMBV11015-E <C,P,F,U>

エラーデータファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of ErrorData file.

Reason code : %1

エラーデータファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	制御ファイルの ErrorDataFile エントリに指定したパスが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ディレクトリにファイルの作成権限がありません。
4	ファイルに書き込み権限がありません。
5	ファイルの作成を続行できないエラーが発生しました。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。要因コードが 5 の場合，エラーデータファイルの作成先ディレクトリに出力された一時ファイル（tmpXX）をエラーデータファイルとして利用してください。

KMBV11016-E <P>

プロダクト情報ファイルが見付かりません。

The product information file was not found.

プロダクト情報ファイルが見つからないため，処理を実行できませんでした。

（S）

処理を中止します。

（O）

次のファイルが存在するか確認してください。存在しない場合は，該当する製品を再インストールしてください。

DocumentBroker Object Loader **の場合**

/opt/HiEDMS_Lod/adm/productdef.ini

DocumentBroker Server **の場合**

/opt/HiEDMS/adm/productdef.ini

KMBV11017-E　<C,P,F>

制御ファイルに ErrorDataFile **エントリが指定されていません。**

ErrorDataFile entry does not exist in Control file.

制御ファイルの ErrorDataFile エントリにエラーデータファイルのパスが設定されていないため，プログラムが実行できません。

（S）

処理を終了します。

（O）

制御ファイルの ErrorDataFile エントリを指定してください。

KMBV11018-E　<C,P,F,U>

StopCount **エントリに指定した値が不正です。**

Invalid value exists in StopCount entry.

制御ファイルの StopCount エントリに指定した値が，数値ではないか，指定範囲（0 ～ 2147483647）外の値です。

（S）

処理を終了します。

（O）

制御ファイルの StopCount エントリに指定した値を確認してください。

KMBV11019-E　<C,P,F,U>

StartPoint **エントリに指定した値が不正です。**

Invalid value exists in StartPoint entry.

制御ファイルの StartPoint エントリに指定した値が，数値でないか，指定範囲（1 ～

2147483647）外の値です。

（S）

処理を終了します。

（O）

制御ファイルの StartPoint エントリに指定した値を確認してください。

KMBV11020-E　<C,P,F,U>

%1 エントリに指定した文字が不正です。

Invalid character exists in %1 entry.

制御ファイルの %1 エントリに指定した文字が引用符（"）で囲まれていないか，指定できない文字です。または ColumnSeparator と RecordSeparator と LineContinue のうちどれか二つ以上に同一文字が指定されています。

（S）

処理を終了します。

（O）

制御ファイルの %1 エントリに指定した値を確認してください。

KMBV11021-E　<C,P,F,U>

指定されたクラス名（%1），プロパティ名（%2）はデータベースにありません。

Class name(%1),Property name(%2) not found in Database.

指定されたクラス名（%1），プロパティ名（%2）はデータベースにありません。

（S）

処理を終了します。

（O）

データベースの構築が正しいかを確認してください。

KMBV11022-E　<C,P,F,U>

%1 エントリの GUID 値が不正です。

Invalid GUID value exists in %1 entry.

定義ファイルの [ClassNameDefinition] または [PropNameDefinition] に指定したエントリ名（%1）の GUID 値が，長さが 36 文字でないか，指定できる文字以外を指定しているか，"-" の位置が不正です。

（S）

処理を終了します。

（O）

定義ファイルを修正してください。

KMBV11023-E　<C,P,F>

定義ファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred opening of Define file.

Reason code : %1

定義ファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	定義ファイルが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルに参照権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11024-E　<C,P,F,U>

指定されたクラス名（%1）はデータベースにありません。

Class name(%1) not found in Database.

指定されたクラス名（%1）はデータベースにありません。

（S）

処理を終了します。

（O）

データベースの構築が正しいかを確認してください。

KMBV11025-E　<C,P,F,U>

%1 **エントリに指定したディレクトリの作成でエラーが発生しました。**

要因コード：%2

An error occurred making of directory in %1 entry.

Reason code : %2

制御ファイルの %1 エントリに指定したディレクトリの作成時にエラーが発生しました。
要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	指定値が誤っています。
2	パスの長さが制限値（128 バイト）を超えています。
3	ディレクトリに書き込み権限がありません。
4	同一名称のディレクトリまたはファイルが存在します。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11026-E　<C,P,F,U>

%1 エントリに指定したファイルの作成でエラーが発生しました。

要因コード : %2

An error occurred making of file in %1 entry.

Reason code : %2

制御ファイルの %1 エントリに指定したディレクトリの作成時にエラーが発生しました。
要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	文字が引用符（"）で囲まれていないか指定できない文字です。
2	ファイルの長さが制限値（36 バイト）を超えています。
3	ディレクトリにファイルの作成権限がありません。
4	ファイルに書き込み権限がありません。
5	ほかのエントリと同一の値は指定できません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11027-E <C,P,F,U>

SortBuffSize エントリに指定した値が不正です。

Invalid value exists in SortBuffSize entry.

制御ファイルの SortBuffSize エントリに指定した値が，数値でないか，指定範囲（128 ～ 2097152）外の値です。

（S）

処理を終了します。

（O）

制御ファイルの SortBuffSize エントリに指定した値を確認してください。

KMBV11028-E <C,P,F,U>

OIID_Count エントリに指定した値が不正です。

Invalid value exists in OIID_Count entry.

制御ファイルの OIID_Count エントリに指定した値が，数値でないか，指定範囲（0 ～ 2147483647）外の値です。

（S）

処理を終了します。

（O）

制御ファイルの OIID_Count エントリに指定した値を確認してください。

KMBV11029-E <C,P,F,U>

システムの表または列がデータベースにありません。

表名：%1

列名：%2

System Table or Column not found in Database.

Table name : %1

Column name : %2

DocumentBroker が定義するクラスおよびプロパティに対応する表または列がデータベースの定義に存在しません。表名（%1），列名（%2）のデータベース定義が不足しています。

DocumentBroker Object Loader は，DocumentBroker が定義するすべてのクラス，プロパティに対応する表および列がデータベースに存在することが前提になります。

（S）

処理を終了します。

（O）

データベースに DocumentBroker が定義するすべての表および列が存在するかを確認してください。

KMBV11030-E <C,P,F,U>

INTEGER_NullValue エントリに指定した値が不正です。

Invalid value exists in INTEGER_NullValue entry.

制御ファイルの INTEGER_NullValue エントリに指定した値が，数値でないか，指定範囲（-2147483648 ～ 2147483647）外の値です。

（S）

処理を終了します。

（O）

制御ファイルの INTEGER_NullValue エントリに指定した値を確認してください。

KMBV11031-E <C,P,F,U>

%1 エントリに指定したディレクトリの削除でエラーが発生しました。

要因コード : %2

An error occurred removing of directory in %1 entry.

Reason code : %2

制御ファイルの %1 エントリに指定したディレクトリの削除時にエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	削除権限がないファイルか，ディレクトリが存在しています。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11032-E <C,P,F,U>

エントリが指定されていません。

エントリ名：%1

セクション名：%2

Entry does not exist.

Entry name : %1

Section name : %2

セクション（%2）にエントリ（%1）が指定されていません。または，必須のエントリが指定されていません。

（S）

処理を終了します。

（O）

エントリを確認してください。

KMBV11034-W　<F,U>

入力データファイルに指定したディレクトリが存在しません。

行番号：%1

ディレクトリ名：%2

Specified object in InputData file does not exist.

Line : %1

Directory : %2

入力データファイルの %1 行目に指定されたディレクトリ（%2）が存在しません。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11035-W　<F,U>

入力データファイルに指定したファイルが登録可能なサイズを超えました。

行番号：%1

サイズ：%2

レコード：%3

ファイル：%4

Invalid file size exists in InputData file is over Size.

Line : %1

Size : %2

Record : %3

File : %4

入力データファイルの %1 行目に指定したファイルが登録可能なサイズ %2 を超えました。%3 レコード目のファイル %4 です。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11036-W <F,U>

マルチファイルのファイル数とコンポーネントタイプの数が一致しません。

行番号：%1

Number of multi files does not much Number of ComponentTypes.

Line : %1

入力データファイルの %1 行目に指定したコンポーネントタイプの数と，マルチファイル文書で管理されているファイルの数が一致しません。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11037-E <C,P,F>

実行環境 ID が不正のため，コマンドを終了します。

Unable to execute this command because the execution environment-id is invalid.

実行環境 ID（実行環境識別子）が 0 以外であるため，コマンドを終了します。

（S）

処理を終了します。

（O）

実行環境 ID（実行環境識別子）が 0 の環境で実行してください。

KMBV11041-W <F,U>

入力データファイル中の "TRANBEGIN" に対する "TRANCOMMIT" がありません。

行番号：%1

There is no "TRANCOMMIT" for "TRANBEGIN" in InputData file.

Line : %1

入力データファイルの %1 行目の "TRANBEGIN" に対応する "TRANCOMMIT" がありません。

(S)

エラーの発生したトランザクションを無視して処理を続行します。

(O)

入力データファイルを修正してください。

KMBV11042-W <F,U>

入力データファイル中に不正な %1 があります。

行番号：%2

Invalid %1 exists in InputData file.

Line : %2

入力データファイルの %2 行目に不正な %1（"TRANBEGIN" または "TRANCOMMIT"）が出現しました。"TRANBEGIN" と "TRANCOMMIT" が対で指定されていません。

(S)

"TRANBEGIN" または "TRANCOMMIT" を無視して処理を続行します。

(O)

入力データファイルを修正してください。

KMBV11043-W <F,U>

入力データファイル中に不正な文字列があります。

行番号：%1

Invalid string exists in InputData file.

Line : %1

入力データファイルの %1 行目の文字列中に，指定できない文字を指定している個所があるか，引用符（"）の指定が誤っている個所があります。または，ラベル名の長さが制限値（16 バイト）を超えているか，クラス名の長さが制限値（128 バイト）を超えています。

(S)

エラーが発生したオブジェクトを無視して処理を続行します。

(O)

入力データファイルを修正してください。

KMBV11044-W <F,U>

入力データファイル中の列数が制御ファイルの DataMapping **エントリに定義した列数と一致しません。**

行番号：%1

Number of columns in InputData file not equal to DataMapping entry in Control file.

Line : %1

%1 行目の入力データファイルの列の個数と制御ファイルの DataMapping エントリに定義した列の個数が一致していません。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルまたは制御ファイルを修正してください。

KMBV11045-E <C,P,F,U>

データベースへの接続でエラーが発生しました。

An error occurred in connecting database.

データベースに接続できません。データベースサーバが起動していないか，接続情報の定義ファイル（docspace.ini）が存在しないか，定義内容が誤っている可能性があります。

（S）

処理を終了します。

（O）

データベースの起動または接続情報の定義を確認してください。

KMBV11046-E <C,P,F,U>

OIID **の取得でエラーが発生しました。**

要因コード：%1

プロセス ID**：**%2

An error occurred in getting OIID.

Reason code : %1

ProcessID : %2

データベースから OIID が取得できませんでした。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	データベースサーバが起動していないか，データベースが正しく構築されていない可能性があります。
XX	OIID 通番管理オブジェクトのメソッドコールでエラーが発生しました。このメッセージの前に，プロセス ID：%2 で出力された KMBR のメッセージを基に対策を実施してください。

（凡例）

XX：OIID 通番管理オブジェクトのメソッドから返却されたリターンコード

（S）

処理を終了します。

（O）

要因コード 1：データベースサーバの起動またはデータベースを確認してください。

要因コード XX：エラー内容に従う処置をとってください。

KMBV11047-E　<C,P,F,U>

セクションに指定したエントリ名称が不正です。

セクション名：%1

エントリ名；%2

Invalid entry name exists in section.

Section name : %1

Entry name : %2

定義ファイルの %1（ClassNameDefinition セクションまたは PropNameDefinition セクションまたは DataMapping セクション）にエントリ名称（%2）が存在しません。または，エントリ名称に " ＊＊ " で始まる名称が指定されているか，長さの制限値（128 バイト）を超えた名称が指定されています。

（S）

処理を終了します。

（O）

定義ファイルの定義を修正してください。

KMBV11048-E　<C,P,F,U>

DataMapping **セクションのエントリに指定したエントリ名称が不正です。**

エントリ：%1

エントリ名称：%2

Invalid entry name exists in entry of DataMapping section.

Entry : %1

Entry name : %2

制御ファイルの DataMapping セクションの %1 エントリ中に指定されているエントリ名称（%2）は ClassNameDefinition または PropNameDefinition セクションに存在しません。または，エントリ名称に " ＊＊ " で始まる名称が指定されているか，長さの制限値（128 バイト）を超えた名称が指定されています。

（S）

処理を終了します。

（O）

制御ファイルの定義を修正してください。

KMBV11049-E　<C,P,F,U>

DataMapping **セクションに指定したエントリが不正です。**

エントリ：%1

Invalid entry exists in DataMapping section.

Entry : %1

制御ファイルの DataMapping セクションの %1 エントリに指定できないシステム定義プロパティがあるか，2 個以上指定できないプロパティを 2 個以上指定しているか，必須のシステム定義プロパティがありません。
または，プロパティに指定した登録先クラスは指定できないクラスです。

（S）

処理を終了します。

（O）

制御ファイルの定義を修正してください。

KMBV11050-W　<F,U>

入力データファイルに指定したオブジェクトは存在しません。

行番号：%1

オブジェクト：%2

Specified object in InputData file does not exist.

Line : %1

Object : %2

入力データファイルの %1 行目に指定したオブジェクト（%2）は存在しません。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11051-W <F,U>

入力データファイルに指定したオブジェクトは複数存在します。

行番号：%1

オブジェクト：%2

Specified object in InputData file exist more than one.

Line : %1

Object : %2

入力データファイルの %1 行目に指定したオブジェクト（%2）は複数存在します。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。またはデータベースの登録内容を確認してください。

KMBV11052-W <F,U>

入力データファイルに指定したプロパティの値が不正です。

行番号：%1

プロパティ値：%2

Invalid property value exists in InputData file.

Line : %1

Property value : %2

入力データファイルの %1 行目に指定したプロパティ名 %2 の入力値は，記述形式が誤っているか，入力可能な範囲外の値を指定しています。または，データベースの型に使用できない型を定義しています。

なお，VariableArray 型のプロパティ，またはマルチファイル文書，パブリック ACL，もしくはローカル ACL のシステムプロパティの場合は，入力データファイルに記述した内容がそのまま出力されないことがあります。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データを修正するか，またはデータベースの定義内容を確認してください。

KMBV11053-E　<C,P,F,U>

StartPoint エントリで指定した行は入力データファイルに存在しません。

Specified line in StartPoint entry does not exist in InputData file.

制御ファイルの StartPoint エントリで指定した行が入力データファイルに存在しません。

（S）

処理を終了します。

（O）

入力データファイルまたは制御ファイルを修正してください。

KMBV11054-E　<C,P,F,U>

HiRDB の制限のため処理が続行できません。

Unable to continue processing due to a limit of HiRDB.

HiRDB の制限のため処理が続行できません。HiRDB をアクセスするための SQL 領域が制限値（2 メガバイト）を超えたか，1 テーブルの列数が制限数（4,000）を超えました。

（S）

処理を終了します。

（O）

データベースのテーブルや列の名称を短くするか，1 テーブルの列数を制限内にしてください。

KMBV11055-W　<F,U>

入力データファイルに指定した文書のオープンでエラーが発生しました。

行番号：%1

An error occurred in opening specified document in InputData file.

Line : %1

入力データファイルの %1 行目に指定した文書がないか，パスの長さが制限値を超えています。または，ファイルに参照権限がありません。

パスの長さの制限値は，文書空間で使用する文字コード種別が Shift-JIS の場合，UTF-8

の場合共に，255 バイトです。

（S）

エラーの発生したトランザクションを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11056-W <F,U>

入力データファイルに指定した検索条件のプロパティは検索できないプロパティです。

行番号：%1

Invalid Property exists in InputData File.

Line : %1

入力データファイルの %1 行目に指定した検索条件のプロパティは検索できないプロパティです。検索条件のプロパティに指定できるプロパティは，プロパティに対応するデータベースの型が文字列または整数です。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正するか，データベースの定義内容を確認ください。

KMBV11057-W <F,U>

入力データファイルに指定したラベル名は指定できないオブジェクトのラベルです。

行番号：%1

ラベル名：%2

Invalid Label exists in InputData File.

Line : %1

Label name : %2

入力データファイルの %1 行目に指定したラベル名（%2）は指定できないオブジェクトのラベルです。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11058-I <F,U>

エラーデータをエラーデータファイルに出力しました。

行番号：%1

Error data outputted for ErrorData file.

Line : %1

エラーデータファイルの %1 行目にエラーデータを出力しました。

（S）

処理を続行します。

（O）

―

KMBV11059-E <C,P,F,U>

入力データファイルに指定したオブジェクトは存在しません。

行番号：%1

オブジェクト：%2

Specified object in InputData file does not exist.

Line : %1

Object : %2

入力データファイルの %1 行目に指定したオブジェクト（%2）は存在しません。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11060-E <C,P,F,U>

入力データファイルに指定したプロパティの値が不正です。

行番号：%1

プロパティ値：%2

Invalid property value exists in InputData file.

Line : %1

Property value : %2

入力データファイルの %1 行目に指定したプロパティ名 %2 の入力値は，記述形式が誤っ

ているか，入力可能な範囲外の値を指定しています。または，データベースの型に使用できない型を定義しています。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データを修正するか，またはデータベースの定義内容を確認してください。

KMBV11061-E　<C,P,F,U>

入力データファイルに指定した文書のオープンでエラーが発生しました。

行番号：%1

An error occurred in opening specified document in InputData file.

Line : %1

入力データファイルの %1 行目に指定した文書の，パスの長さが制限値（255 バイト）を超えています。または，ファイルに参照権限がありません。

（S）

エラーの発生したトランザクションを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11062-E　<C,P,F,U>

入力データファイルに指定したラベル名は指定できないオブジェクトのラベルです。

行番号：%1

ラベル名：%2

Invalid Label exists in InputData File.

Line : %1

Label name : %2

入力データファイルの %1 行目に指定したラベル名（%2）は指定できないオブジェクトのラベルです。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11065-E <C,P,F,U>

入力データファイルに指定したプロパティの値が最大要素数を超えています。

行番号：%1

プロパティ値：%2

最大要素数：%3

Invalid property value exists in InputData file is over Max Element.

Line : %1

Property value : %2

Max Element : %3

入力データファイルの %1 行目に指定した先頭が %2 で始まるプロパティ値の要素数がデータベースで定義した最大要素数を超えています。

（S）

処理を終了します。

（O）

入力データを修正するか，またはデータベースの定義内容を確認してください。

KMBV11066-W <F,U>

入力データファイルに指定したプロパティの値が最大要素数を超えています。

行番号：%1

プロパティ値：%2

最大要素数：%3

Invalid property value exists in InputData file is over Max Element.

Line : %1

Property value : %2

Max Element : %3

入力データファイルの %1 行目に指定した先頭が %2 で始まるプロパティ値の要素数がデータベースで定義した最大要素数を超えています。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データを修正するか，またはデータベースの定義内容を確認してください。

KMBV11067-E <C,P,F,U>

入力データファイルに指定した作成コマンドに実行できないクラス名が指定されています。

行番号：%1

作成コマンド名：%2

クラス名：%3

Invalid Class name exists Command in InputData file.

Line : %1

Command name : %2

Class name : %3

入力データファイルの %1 行目に指定した作成コマンド %2 とクラス名 %3 のスーパークラスが一致していません。作成コマンドを見直すか，クラス名を見直してください。

(S)

処理を終了します。

(O)

入力データファイルを修正後，再度実行してください。

KMBV11069-W <F,U>

入力データファイルに指定したパーミッションの値が不正です。

行番号：%1

パーミッション：%2

Invalid property Permission value exists in Input file.

Line : %1

Permission : %2

入力データファイルの %1 行目に指定したパーミッション値 %2 は，記述形式が誤っているか，指定できない組み合わせを指定しています。

(S)

エラーが発生したオブジェクトを無視して処理を続行します。

(O)

入力データファイルを修正してください。

KMBV11071-W <F,U>

入力データファイルに XML **文書解析に必要な** XML **ファイルのパスが指定されていません。**

行番号：%1

要因コード：%2

Not Specified XML file path that analyze XML in InputData file.

Line : %1

Reason code : %2

入力データファイルの %1 行目に XML 文書解析に必要な XML ファイルのパスが指定されていません。

要因コード	エラー内容
1	XML ファイルが存在しません。
2	XML ファイルのパスの長さが制限値を超えています。 パスの長さの制限値は，文書空間で使用する文字コード種別が Shift-JIS の場合，UTF-8 の場合共に，255 バイトです。
3	XML ファイルに参照権限がありません。
4	XML ファイルが 0 バイトです。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11073-W <F,U>

入力データファイルにプロパティマッピングに必要なデータが指定されていません。

行番号：%1

要因コード：%2

Not Specified Data that required Property Mapping in InputData file.

Line : %1

Reason code : %2

入力データファイルの %1 行目のデータにプロパティマッピングに必要なデータが指定されていないか，指定が不正です。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	マッピング定義名が指定されていません。
2	マッピング定義名の長さが制限値（214 バイト）を超えています。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

エラーの要因を取り除いてください。

KMBV11074-W <F,U>

入力データファイルに指定したプロパティマッピングに必要なファイルのオープンでエラーが発生しました。

行番号：%1

要因コード：%2

An error occurred in opening specified file for Property Mapping in InputData file.

Line : %1

Reason code : %2

入力データファイルの %1 行目に指定したプロパティマッピングに必要なファイルのオープンでエラーが発生しました。

要因コード	エラー内容
1	XMS ファイルが存在しません。
2	XMS ファイルのパスの長さが制限値（255 バイト）を超えています。
3	XMS ファイルに参照権限がありません。
4	XMS ファイルが 0 バイトです。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

エラーの要因を取り除いてください。

KMBV11075-W <F,U>

プロパティマッピング実行時にエラーが発生しました。

行番号：%1

詳細情報：%2

Exists Invalid Data in specified file that required Property Mapping in InputData file.

Line : %1

Information %2

入力データファイルの %1 行目でプロパティマッピング実行時にエラーが発生しました。エラーの詳細は %2 です。

詳細情報	エラー内容
HiRDB Adapter for XML loading failed.	次のエラー内容が考えられます。 • HiRDB Adapter for XML がインストールされていません。 • 環境変数 LIBPATH（AIX の場合）にパスが設定されていない可能性があります。
XX	次のエラー内容が考えられます。 • プロパティマッピングの取得処理に失敗しました。 • 環境変数 XMLBRKDIR にパスが設定されていない可能性があります。
Boolean data bad value.	Boolean データとして，0，1，2 以外が指定されています。

（凡例）

XX：HiRDB Adapter for XML から返却されたエラーメッセージテキスト

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

%2 に示される詳細情報を基に障害の要因を取り除いてください。

KMBV11076-W <F,U>

入力データファイルで指定した XML 文書登録クラス（%1）は処理対象外のクラスです。

要因コード：%2

Invalid Class in Class(%1).that specified in InputData file for setting XML file.

Reason code : %2

指定した処理対象文書クラス（%1）は処理対象外です。要因コード（%2）エラーの内容を確認してください。

要因コード	エラー内容
1	指定したクラスは全文検索対象クラスではありません。 または，対象外の全文検索インデクス用プロパティが定義されています。対象プロパティは，次のとおりです。 • edmProp_Content • edmProp_StIndex • edmProp_ConceptStIndex

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

メタ情報およびデータベース定義を正しく設定してください。

KMBV11077-W <F,U>

入力データファイルに指定したフィルタリング定義ファイルのオープンでエラーが発生しました。

行番号：%1

要因コード：%2

An error occurred in opening specified definition file for filtering in Input Data file.

Line : %1

Reason code : %2

入力データファイルの %1 行目に指定したフィルタリング定義ファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	ファイルが存在しません。
2	ファイルのパスの長さが制限値（255 バイト）を超えています。
3	ファイルに参照権限がありません。
4	ファイルが 0 バイトです。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

エラーの要因を取り除いてください。

KMBV11078-W <F,U>

構造指定全文検索ファイル作成時にエラーが発生しました。

行番号：%1

詳細情報：%2

An error occurred making struct all sentence search file.

Line : %1

Information %2

入力データファイルの %1 行目で構文指定全文検索ファイル作成時にエラーが発生しました。エラーの詳細は %2 です。詳細情報でエラー内容を確認してください。

詳細情報	エラー内容
TmpFile create error	TMPDIR 下に全文検索インデクス一時保管用ファイルの作成に失敗しました。
Preprocessing Library for Text Search loading failed.	Preprocessing Library for Text Search がインストールされていません。
XX	全文検索インデクスファイルの作成処理に失敗しました。

（凡例）

XX：Preprocessing Library for Text Search から返却されたエラーメッセージテキスト

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

エラーの要因を取り除いてください。

KMBV11079-W <F,U>

入力データファイルに指定したプロパティマッピング用のプロパティ値に P_XML が指定されていません。

行番号：%1

プロパティ名：%2

No "P_XML" in specified definition file for filtering in InputData file.

Line : %1

Property: %2

入力データファイルの %1 行目に指定したプロパティマッピング用のプロパティ値に P_XML が指定されていません。プロパティ値に P_XML を指定してください。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11081-E <C,P,F>

メモリが不足しているため，処理を続行できません。

There is insufficient memory.

メモリが不足しました。

（S）

処理を終了します。

（O）

不要なプログラムを終了させて再度実行してください。

KMBV11082-E　<C,P,F>

内部矛盾が発生しました。

エラー情報：%1

Internal error occurred to ObjectLoader.

Error information：%1

オブジェクトローダで内部矛盾が発生しました。
%1：エラー情報

（S）

処理を終了します。

（O）

トレースログを採取して保守員に連絡してください。

KMBV11083-E　<C,P,F,U>

データベースでエラーが発生しました。

エラー情報：%1

要因コード：%2

An error occurred in database.

Error information : %1

Reason code : %2

データベースへのアクセスでエラーが発生しました。
%1：データベースのエラーメッセージ
%2：データベースのエラーコード
エラー情報に表示されるメッセージ ID でエラー内容を確認してください。

メッセージ ID	エラー内容
KFPA11205-E	制御ファイルの DataMapping セクションの記述または定義ファイルの GUID 値に誤りがあります。見直してください。
KFPA11561-E KFPA11703-E	接続情報の定義ファイル（docspace.ini）の記述に誤りがあります。見直してください。
その他	マニュアル「HiRDB　メッセージ」を参照してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11084-E <C,P,F>

システムコールでエラーが発生しました。

関数名：%1

要因コード：%2

An error occurred on the system call function.

Function Name : %1

Reason Code : %2

システムコール（%1）でエラーが発生しました。

%1：エラーの発生した関数名

%2：システムコールで返された errno 値

（S）

処理を終了します。

（O）

%2 に出力される要因コードを基に障害の要因を取り除いてください。

KMBV11085-E <P,F>

DocumentBroker **サーバのバージョンが不正なためコマンドを終了します。**

Unable to execute this command because version of DocumentBroker Server is invalid.

DocumentBroker サーバのバージョンが 03-00 よりも前のため，EDMLoad コマンドを実行できません。

（S）

処理を終了します。

（O）

DocumentBroker サーバのバージョンを 03-00 以降にしてください。

KMBV11086-E <P>

DocumentBroker **サーバのバージョン取得時にエラーが発生しました。**

An error occurred in obtaining of Version information for DocumentBroker Server.

DocumentBroker サーバのバージョン情報を取得できないため，EDMLoad コマンドを実行できません。

（S）

処理を終了します。

（O）

DocumentBroker サーバが正しくインストールされているかを確認してください。

KMBV11087-E　<C,P,F,U>

メタ情報の取得でエラーが発生しました。

GUID **値：**%1

An error occurred in obtaining of Meta information.

GUID value : %1

GUID 値（%1）のメタ情報の取得でエラーが発生しました。

（S）

処理を終了します。

（O）

GUID 値（%1）のメタ情報が正しく登録されているかを確認してください。

KMBV11090-E　<C,F>

メモリが不足しているため，処理が続行できません。

領域名：%1

領域長：%2

Memory became insufficient.

Area Name : %1

Area Size : %2

領域長 %2 のサイズのメモリを確保できなかったため，処理が続行できません。

（S）

処理を続行します（エラーを返却）。

（O）

メモリを増設，またはメモリを多量に消費しているアプリケーションを停止させ，処理を再実行してください。

KMBV11091-E　<C,F>

メモリが不足しているため，処理が続行できません。

クラス名：%1

Memory became insufficient.

Class Name : %1

クラス %1 のインスタンスを生成するためのメモリを確保できなかったため，処理が続行できません。

(S)

処理を続行します（エラーを返却)。

(O)

メモリを増設，またはメモリを多量に消費しているアプリケーションを停止させ，処理を再実行してください。

KMBV11092-E <C,F>

ファイルシステムに入出力エラーが発生しました。

対象：%1

操作：%2

要因コード：%3

付加情報：%4

An I/O error occurred in the file system.

Object : %1

Operation : %2

Reason Code : %3

Additional Information : %4

ファイルシステム中の %1 のオブジェクトに %2 の操作を行う際にエラーが発生したため，処理が続行できません。

(S)

処理を続行します（エラーを返却)。

(O)

%3 に出力される要因コードを基に障害の要因を取り除き，処理を再実行してください。

KMBV11093-E <C,F>

システムコールでエラーが発生しました。

関数名：%1

要因コード：%2

付加情報：%3

An error occurred on the system call function.

Function Name : %1

Reason Code : %2

Additional Information : %3

システムコール %1 でエラーが発生したため，処理が続行できません。

(S)

処理を続行します（エラーを返却)。

(O)

%2 に出力される要因コードを基に障害の要因を取り除き，処理を再実行してください。

KMBV11094-E <C,P,F>

環境定義ファイル読み込み中にエラーを検出しました。

詳細：%1

位置：[%2]%3

An error occurred on the setting of environment.

Detail : %1

Position : [%2]%3

環境定義ファイル読み込み中に %1 に示すエラーを検出したため，サーバを起動できません。

(S)

サーバが停止しています。

(O)

%1 に示される詳細を基に障害の要因を取り除き，サーバを再起動してください。

詳細

- docspace.ini does not found.
  docspace.ini ファイルが存在しない。
- docspace.def does not found.
- docspace.def uncorrect
  インストールが正しく行われていない。
- parameter is not omissible.
  [%2]%3 で示されるエントリは docspace.ini から省略することはできない。
- parameter is uncorrect.
  [%2]%3 で示されるエントリの docspace.ini 中の設定が誤っている。

KMBV11095-E <C,P,F>

言語の指定に誤りがあります。

言語名：%1

An error occurred on the setting of locale.

Locale name : %1

（S）

コマンドを実行した環境下の言語の指定が不正であるため，処理を中断しました。

（O）

言語を正しく（ja_JP.SJIS または C）設定してください。

KMBV11096-E <C,P,F>

ディスク容量が不足しているため，処理を続行できません。

There is insufficient space of disk.

ディスク容量が不足しました。

（S）

処理を終了します。

（O）

ディスクに空きがあることを確認して再度実行してください。

KMBV11097-E <C,P,F,U>

システム定義プロパティに指定したクラス名が不正です。

行番号：%1

システムプロパティ名：%2

クラス名：%3

Invalid Class name to System property.

Line : %1

System property name : %2

Class name : %3

システムプロパティ %2 に指定したクラス名 %3 はスーパークラスが一致していません。クラス名を見直してください。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正後，再度実行してください。

KMBV11098-W　<F,U>

全文検索インデクスファイルの登録時にエラーが発生しました。

行番号：%1

要因コード：%2

Error Occurred in entry Text Search Index file.

Line : %1

Reason code : %2

全文検索インデクスファイルの登録時にエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	全文検索インデクスファイルのサイズが 0 バイトです。
2	全文検索インデクスファイルのサイズが 5 メガバイトを超えています。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

XML 文書の内容を確認してから再度実行してください。

KMBV11099-W　<F,U>

マッピング定義 ID に定義されていないユーザプロパティに P_XML が指定されています。

行番号：%1

クラス名：%2

プロパティ名：%3

Invalid "P_XML" exists User Property is not define MappingID.

Line : %1

Class name : %2

Property name : %3

システム定義プロパティ（**PROP_XML_MAP**）に指定したマッピング定義名に定義されていないユーザプロパティに対して，予約語（P_XML）を指定しています。

（S）

エラーが発生したオブジェクトを無視して処理を続行します。

（O）

入力データファイルを修正してください。

KMBV11100-W <F,U>

ファイル名が重複しています。

行番号：%1

ファイル名：%2

The file name is duplicated.

Line : %1

File name : %2

入力データ %1 行目のシステム定義プロパティ（**PROP_CT_PATH**）に定義されているディレクトリ下に同じファイル名 %2 のファイルがあります。

（S）

ワーニングが発生したオブジェクトを無視して処理を続行します。

（O）

PROP_CT_PATH に定義されているディレクトリ下で重複しているファイル名 %2 を変更してください。

KMBV11101-W <F,U>

マッピング定義 ID **に定義されていないユーザプロパティに** P_XML **が指定されているか，** HiRDB Adapter for XML **から値が返却されません。**

行番号：%1

クラス名：%2

プロパティ名：%3

Invalid "P_XML" exists User Property is not define MappingID, or a value is not returned from HiRDB Adapter for XML.

Line : %1

Class name : %2

Property name : %3

システム定義プロパティ（**PROP_XML_MAP**）に指定したマッピング ID に定義されていないユーザプロパティに対して，予約語（P_XML）を指定しています。または，HiRDB Adapter for XML からプロパティマッピングに使用するプロパティデータが返却

されません。

(S)

エラーが発生したオブジェクトを無視して処理を続行します。

(O)

入力データファイルを修正してください。または，XML 文書の内容を確認してください。

KMBV11102-W <F,U>

入力データファイルに指定したサブジェクトの値が不正です。

行番号：%1

サブジェクト：%2

Invalid subject value exists in InputData file.

Line : %1

Subject value: %2

システム定義プロパティ（**PROP_ACL_SUBJECT**）に指定したサブジェクトの値がシステムサブジェクトではありません。

(S)

エラーが発生したオブジェクトを無視して処理を続行します。

(O)

入力データファイルを修正してください。

KMBV11103-W <F,U>

リファレンスファイル文書のコンテント操作でエラーが発生しました。

行番号：%1

詳細 1：%2

詳細 2：%3

詳細 3：%4

An error occurred in content operation of a reference file document.

Line : %1

Detail1: %2

Detail2: %3

Detail3: %4

入力データファイルの %1 行目の処理中に，リファレンスファイル文書のコンテント操

作でエラーが発生しました。

（S）

エラーが発生したオブジェクトを無視して処理続行します。

（O）

%2 に出力される詳細別のエラー内容および対処の一覧を参照して，エラーの要因を取り除いてください。%2 に出力される詳細別のエラー内容および対処の一覧を次に示します。

詳細（%2）	エラー内容	対処
File open / Directory make / Change permission failed.	次のエラー内容が考えられます。 • 関数 open または関数 fopen でファイルのオープンに失敗しました。 • 関数 mkdir でディレクトリの作成に失敗しました。 • アクセス権の変更に失敗しました。	コンテント格納先ベースパス以下のディレクトリのアクセス権限を見直してください。 詳細 2 に示されるエラー番号※1 から errno.h の内容を参照して，対処してください。 詳細 3 に，該当するパスの情報が出力されます。
File write failed.	関数 fwrite でファイルの書き込みに失敗しました。	詳細 2 に示されるエラー番号※1 から errno.h の内容を参照して，対処してください。 詳細 3 に，該当するパスの情報が出力されます。
File close failed.	関数 close または関数 fclose でファイルのクローズに失敗しました。	詳細 2 に示されるエラー番号※1 から errno.h の内容を参照して，対処してください。 詳細 3 に，該当するパスの情報が出力されます。
FilePath is too long.	コンテントの格納先パスが長過ぎます。	コンテント格納先のファイル名のパス※2 の長さが，指定できる範囲を超えています。 文字コード種別が，Shift-JIS および UTF-8 のどちらの場合も，コンテント格納先のファイル名の長さが次の範囲内になるように，「コンテント格納先ベースパス」，「コンテント格納先パス」，および「コンテントのファイル名」を指定してください。 AIX の場合 1 ～ 1,022 バイト Windows の場合 1 ～ 259 バイト※3 詳細 3 に，該当するパスの情報が出力されます。

詳細（%2）	エラー内容	対処
Reference path invalid.	コンテント格納先ベースパスの指定が不正です。	コンテント格納先ベースパスの指定を見直してください。 詳細 2 に示されるエラー番号※1 から errno.h の内容を参照して，対処してください。詳細 3 に，該当するパスの情報が出力されます。
Reference path is not directory.	コンテント格納先ベースパスの指定がディレクトリではありません。	コンテント格納先ベースパスの指定を見直してください。詳細 3 に，該当するパスの情報が出力されます。

注※ 1

エラー番号については，errno.h もしくは Win32(R) API のエラーコード，または OS のマニュアルを参照してください。

注※ 2

コンテント格納先のファイル名のフルパスは次のようになります（条件によってフルパスが異なります）。

● DocumentSpace 構成定義ファイル（docspace.ini）の ReferenceStorageMode エントリに Origin を指定した場合，または DocumentBroker サーバのバージョンが 03-11 以前の場合

```
aaaa/bbbb/cccc/dddd
```

aaaa：コンテント格納先ベースパス

bbbb：コンテント格納先パス

cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（33 バイト）

dddd：**PROP_CT** に指定したコンテントのファイル名（拡張子を含む）

● DocumentSpace 構成定義ファイル（docspace.ini）の ReferenceStorageMode エントリに Divide を指定した場合

```
aaaa/bbbb/cccc/ddddeeee
```

aaaa：コンテント格納先ベースパス

bbbb：コンテント格納先パス

cccc：DocumentBroker サーバがコンテントを管理するためのディレクトリ（4 バイト）

dddd：DocumentBroker サーバがコンテントを管理するためのファイル名（42 バイト）

eeee：**PROP_CT** に指定したコンテントのファイル名の拡張子（ピリオドを含めて 0 ～ 17 バイト）

注※ 3

指定できるパスの長さは，文書空間で使用する文字コード種別が Shift-JIS の場合，UTF-8 の場合共に，1 ～ 259 バイトです。

KMBV11107-I　<F,U>

入力データファイルに指定したファイルが存在しないためリファレンスファイル管理機能のパス操作のモードを NONE として登録します。

行番号：%1

ファイル名：%2

The mode of path operation of a reference file function is registered as NONE because a specified file does not exist.

Line : %1

File name : %2

入力データファイルの %1 行目に指定した %2 ファイルがないため，リファレンスファイル文書のパス操作のモードを NONE として登録します。

（S）

処理を続行します。

（O）

―

KMBV11110-E　<C,P,F,U>

文書空間文字コード種別の取得でエラーが発生しました。

要因コード：%1

詳細：%2

An error occurred in getting docspace character-code type.

Reason Code : %1

Detail : %2

DocumentBroker サーバの文書空間の文字コード種別を取得する処理でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容	対処
3	fopen 関数で DocumentBroker サーバのメタ情報ファイル（edmsys.ini）のオープンに失敗しました。	詳細に出力されるエラー番号※を基に，障害の要因を取り除いてください。 なお，詳細に示されるエラー番号※が 2 の場合，環境変数 DOCBROKERDIR の指定値が誤っているか，DocumentBroker サーバの環境が構築されていません。 再度，正しく環境設定をしてください。
4	DocumentBroker サーバのメタ情報ファイル（edmsys.ini）の読み込みに失敗しました。	詳細に示されるエラー番号※を基に，障害の要因を取り除いてください。
5	DocumentBroker サーバのメタ情報ファイル（edmsys.ini）が不正です。	DocumentBroker サーバの環境を再構築してください。

注※

エラー番号については，errno.h もしくは Win32 API のエラーコード，または OS のマニュアルを参照してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11111-E　<C,P,F,U>

リファレンスファイル管理機能のコンテント格納方式の取得でエラーが発生しました。

要因：%1

詳細 1：%2

詳細 2：%3

An error occurred in getting storage method of a reference file document.

Reason: %1

Detail1: %2

Detail2: %3

DocumentBroker サーバのリファレンスファイル管理機能のコンテント格納方式の取得処理でエラーが発生しました。エラーの要因を確認してください。

要因	要因の内容	対処
File is not accessible.	DocumentSpace 構成定義ファイル（docspace.ini）のオープンに失敗しました。または，DocumentSpace 構成定義ファイルの参照権限がありません。	DocumentSpace 構成定義ファイル（docspace.ini）のアクセス権を見直してください。 または，詳細（%2）に示されるエラー番号※を基に，errno.h の内容を参照して対処してください。
Parameter is incorrect.	DocumentSpace 構成定義ファイル（docspace.ini）のセクション（%2）のエントリ（%3）の設定が不正です。	DocumentSpace 構成定義ファイル（docspace.ini）のセクション（%2）のエントリ（%3）の内容を見直してください。

注※

エラー番号については，errno.h もしくは Win32 API のエラーコード，または OS のマニュアルを参照してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11309-I <C,P,F>

実行環境を作成中です。

It is creating the environment of execute.

（S）

DocumentBroker Object Loader の実行環境を作成中です。

（O）

―

KMBV11310-I <C,P,F>

実行環境を作成しました。

It is success to create the environment of execute.

（S）

DocumentBroker Object Loader の実行環境の作成が完了しました。

（O）

―

KMBV11311-I <C,P,F>

実行環境を削除中です。

It is deleting the environment of execute.

（S）

DocumentBroker Object Loader の実行環境を削除中です。

（O）

―

KMBV11312-I <C,P,F>

実行環境を削除しました。

It is success to delete the environment of execute.

（S）

DocumentBroker Object Loader の実行環境の削除が完了しました。

（O）

―

KMBV11313-W <C,P,F>

ファイル（%1）はすでに実行環境に存在するため，処理をスキップします。

The operation skipped about file (%1), because it already exists.

（S）

%1 に示したファイルはすでに指定した実行環境に存在しているので処理をスキップしました。

（O）

―

KMBV11314-E <C,P,F>

ファイル（%1）がすでに実行環境に存在するため，処理が続行できません。

The operation can not continue, because the file (%1) already exists.

（S）

%1 に示したディレクトリを作成しようとしましたが，すでにファイルとして存在したため処理を中断しました。

（O）

ファイルを削除してください。または，別の実行環境を指定してコマンドを再実行してください。

KMBV11315-E <C,P,F>

ファイル（%1）がインストールディレクトリ下に存在するため，処理が続行できません。

The operation can not continue, because the file (%1) exists under install directory.

（S）

インストールディレクトリ下に存在するファイル %1 の open システムコールに失敗したため，処理を中断しました。

（O）

この前に出力された open システムコールのエラーメッセージを基にエラーの要因を取り除き，再実行してください。

KMBV11320-W <C,P,F>

他のプログラムが起動されています。

Other program was started.

（S）

指定したディレクトリでほかのプログラムが起動しています。

（O）

―

KMBV11321-E　<C,P,F>

環境作成先のディレクトリもしくはインストールディレクトリが存在しません。

位置：%1

Directory for making environment or install directory is not exist.

Position : %1

環境作成先ディレクトリまたはインストールディレクトリが存在しません。

（S）

DocumentBroker Object Loader の実行環境の作成を中止します。

（O）

%1 に必要なディレクトリが存在するかどうか確認してください。

KMBV11322-E　<C,P,F>

環境作成先ディレクトリにアクセス権限がありません。

位置：%1

You don't have permission of directory for making environment.

Position : %1

環境作成先ディレクトリにアクセス権限がありません。

（S）

DocumentBroker Object Loader の実行環境の作成を中止します。

（O）

%1 のディレクトリにアクセス権限があるかどうか確認してください。

KMBV11324-E　<C,P,F>

インストールディレクトリ下を指定することはできません。

Can't designate install directory.

インストールディレクトリ下へ環境を作成したり，インストールディレクトリ下の環境を削除することはできません。

（S）

DocumentBroker Object Loader の実行環境の作成を中止します。

（O）

指定したディレクトリがインストールディレクトリ下かどうかを確認してください。

KMBV11325-E　<P>

処理できない障害が発生したため，コマンドの実行を中止します。

This command is terminated abnormally due to an error which prevents processing.

処理できない障害が発生したため，EDMLodSetup を中止しました。

（S）

DocumentBroker Object Loader の実行環境の作成を中止します。

（O）

この前に出力されたエラーメッセージを基に障害を取り除き，EDMLodSetup を再実行してください。

KMBV11400-I　<C,P,F,U>

オブジェクトエクスポートの処理が終了しました。

出力オブジェクト数：%1 **件**

EDMExport ended.

Object : %1

オブジェクト指定ファイルの実行の結果，%1 件のオブジェクトを出力できました。

（S）

処理を終了します。

（O）

-

KMBV11401-E　<P>

指定したコマンドの形式に誤りがあります。コマンドの形式は次のとおりです。

EDMExport **オブジェクト指定ファイル** -o **オブジェクトローダ入力データファイル** [-c **制御ファイル**] [-d **定義ファイル**]

Error exists in command line, usage is as follows

Usage: EDMExport SelectDataFile -o ObjectLoaderInputFile [-c ControlFile] [-d DefineFile]

指定したコマンド形式に誤りがあります。

（S）

処理を終了します。

（O）

正しいコマンド形式を指定して再度実行してください。

KMBV11402-E <C,P,F>

オブジェクトエクスポートで内部矛盾が発生しました。（%1）

Logical inconsistency occurred to ObjectExport.（%1）

オブジェクトエクスポートで内部矛盾が発生しました。（%1）

（S）

処理を終了します。

（O）

システム管理者に連絡してください。

KMBV11403-E <C,P,F,U>

DataMapping **セクションに（%1）が定義されていません。**

%1 is not defined in DataMapping section.

定義ファイルの DataMapping セクションにクラスまたはプロパティ（%1）が定義されていません。

（S）

処理を終了します。

（O）

制御ファイルの DataMapping セクションの定義を見直してください。

KMBV11404-E <C,P,F,U>

オブジェクトの矛盾を検知しました。

オブジェクトの This **ポインタ：**%1

Invalid status found in object.

This pointer of object : %1

クラスのテーブルに出力対象のオブジェクトが存在しません。This ポインタで示されたオブジェクトを正しく修正するか，削除してください。オブジェクトまたは関連するオブジェクトが削除された可能性があります。This ポインタの先頭 36 バイトがクラスの GUID 値を表します。この GUID 値を定義ファイルで検索することによってオブジェクトのクラスを特定できます。This ポインタの残りの 16 バイトはオブジェクトの OIID を表します。この OIID によってオブジェクトを特定することができます。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11405-E　<C,P,F,U>

オブジェクト指定ファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of SelectData file.

Reason code : %1

オブジェクト指定ファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	ファイルが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルに参照権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11406-E　<C,P,F>

オブジェクトローダ入力データファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening specified document in ObjectLoaderInput file.

Reason code : %1

オブジェクトローダ入力データファイルのオープンでエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	パスが正しくありません。
2	パスの長さが制限値（255 バイト）を超えています。
3	パス中のディレクトリに更新権限がありません。

要因コード	エラー内容
4	ファイルがすでに存在します。存在しないファイル名を指定してください。また，同一ファイル名で実行中の EDMExport が存在する可能性があります。ほかの EDMExport で出力しているファイル名と重複しないファイル名を指定してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11407-E　<C,P,F,U>

クラスの定義に誤りがあります。

Invalid definition of class exists in Define File.

クラスの定義に誤り（クラス名の長さが上限を超えている場合）があります。制御ファイル，定義ファイルの内容を確認してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11408-E　<C,P,F,U>

出力対象のオブジェクトが属するクラスが定義ファイルに定義されていません。

Class of object for output not found in Define File.

出力対象のオブジェクトが属するクラスが定義ファイルに定義されていません。定義ファイルの内容を確認してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11410-E　<C,P,F,U>

オブジェクト指定ファイル中に不正な文字列があります。

Invalid string exists in SelectData file.

オブジェクト指定ファイルの文字列中に，指定できない文字を指定している個所があるか，引用符（"）の指定が誤っている個所があります。または，クラス名の長さが制限値

（128 バイト）を超えています。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルを修正してください。

KMBV11411-E　<C,P,F,U>

オブジェクト指定ファイル中の条件式の構文が不正です。

Number of columns in SelectData file not equal to DataMapping entry in Control file.

オブジェクト指定ファイル中の条件式の構文が不正です。条件式が構文規則に合っているか確認してください。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルまたは制御ファイルを修正してください。

KMBV11412-E　<C,P,F,U>

オブジェクト指定ファイルに指定したオブジェクトは存在しません。

Specified object in SelectData file does not exist.

オブジェクト指定ファイルに指定したオブジェクトは存在しません。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルを修正してください。

KMBV11413-E　<C,P,F,U>

オブジェクト指定ファイルに指定したクラス，またはプロパティが不正です。

名称：%1

Invalid class or property name exists in SelectData file.

Property value : %1

オブジェクト指定ファイルに指定したクラス名またはプロパティ名 %1 の入力値は，記述形式が誤っているか，定義ファイルに定義されていません。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルの内容を修正するか，またはデータベースの定義内容を確認してください。

KMBV11414-E　<C,P,F,U>

%1 に指定した値が有効範囲内ではありません。

Specified value %1 is out of range.

%1 に指定した値が有効範囲内ではありません。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルを修正してください。

KMBV11415-E　<C,P,F,U>

オブジェクト指定ファイルに指定した条件式のプロパティはデータ型が不正です。

Invalid Property exists in SelectData File.

オブジェクト指定ファイルに指定した検索条件のプロパティは検索できないプロパティです。検索条件のプロパティに指定できるプロパティは，プロパティに対応するデータベースの型が文字列（255 バイト以下の MVARCHAR）または整数（INTEGER）です。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルを修正するか，データベースの定義内容を確認してください。

KMBV11416-E　<C,P,F,U>

オブジェクトのクラスが定義ファイルに定義されていません。

Class name of selected object is undefined in Definition file.

オブジェクトローダ入力データファイルの出力対象のオブジェクトのクラスの定義が定義ファイル中にありません。

（S）

処理を終了します。

（O）

定義ファイルの内容を確認してください。

KMBV11417-E　<C,P,F,U>

オブジェクト指定ファイルに指定したプロパティの値が不正です。

プロパティ値：%1

Invalid property value exists in SelectData file.

Property value : %1

オブジェクト指定ファイルに指定したプロパティ名 %1 の入力値は，記述形式が誤っているか，入力可能な範囲外の値を指定しています。または，データベースの型に使用できない型を定義しています。

(S)

処理を終了します。

(O)

オブジェクト指定ファイルの内容を修正するか，またはデータベースの定義内容を確認してください。

KMBV11418-E　<C,P,F,U>

ラベル名の上限を超えました。

Over the limit of label name.

ラベル名の上限を超えました。

(S)

処理を終了します。

(O)

オブジェクト指定ファイルを分割して処理単位を少なくしてください。

KMBV11419-E　<C,P,F,U>

オブジェクト指定ファイルに指定したラベル名は指定できないオブジェクトのラベルです。

ラベル名：%1

Invalid Label exists in SelectData File.

Label name : %1

オブジェクト指定ファイルに指定したラベル名（%1）は指定できないオブジェクトのラベルです。

(S)

処理を終了します。

(O)

オブジェクト指定ファイルを修正してください。

KMBV11420-E <C,P,F,U>

オブジェクト指定ファイルに指定した条件式に指定できないクラスが指定されています。

Invalid Class exists in SelectData File.

オブジェクト指定ファイルに指定した条件式に指定できないクラスが指定されている。

（S）

処理を終了します。

（O）

オブジェクト指定ファイルを修正してください。

KMBV11421-E <C,P,F,U>

エクスポートセクションの文書のコンテンツを格納するディレクトリパスの指定が不正です。

要因コード：%1

The designation of the directory pass which the contents of the document of ExportSection are stored in is illegal.

Reason code : %1

文書のコンテンツを格納するディレクトリパスの指定でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	制御ファイルに DocDir エントリが指定されていません。
2	ディレクトリパスが正しくありません。
3	ディレクトリパスの長さが制限値（255 バイト）を超えています。
4	ディレクトリがすでに存在します。存在しないディレクトリ名を指定してください。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11422-E <C,P,F,U>

DataMapping **セクションのエントリに指定した登録先クラス名が不正です。**

エントリ：%1

登録先クラス名：%2

Invalid ClassName exists entry of DataMapping section.

Entry : %1

Class name:%2

制御ファイルの DataMapping セクションの %1 エントリ中に指定している登録先クラス名（%2）は，スーパークラスが dmaClass_DocVersion 以外のため指定できません。

（S）

処理を終了します。

（O）

制御ファイルの定義を修正してください。

KMBV11423-W　<F,U>

出力対象のオブジェクトが属するクラスに繰り返しデータ型プロパティが含まれています。

クラス名：%1

Class of Object contains HiRDB Array Property.

Class name : %1

出力対象のオブジェクトが属するクラス（%1）には，繰り返しデータ型（HiRDB Array 列型）プロパティが含まれています。
このメッセージが出力された入力データファイルを使用してオブジェクトローダを実行する場合は，入力データファイルに，エラーログファイルの KMBV11424-W のメッセージに表示された繰り返しデータ型（HiRDB Array 列型）のプロパティに対して，オブジェクローダの記述形式に従って繰り返しデータ（HiRDB Array 列）を指定してください。

（S）

処理を続行します。

（O）

―

KMBV11424-W　<F,U>

出力対象のオブジェクトが属するクラスに繰り返しデータ型プロパティが含まれています。

クラス名：%1

プロパティ名：%2

行番号：%3

Class of Object contains HiRDB Array Property.

Class name : %1

Property name : %2

Line : %3

出力対象のオブジェクトが属するクラス（%1）に，繰り返しデータ型（HiRDB Array 列型）プロパティが含まれています。
オブジェクトエクスポート終了後，生成したオブジェクトローダ入力データファイルの（%3）行目の繰り返しデータ型（HiRDB Array 列型）プロパティ（%2）に対して，必要に応じて繰り返しデータ（HiRDB Array 列）を追加指定してください。

（S）

処理を続行します。

（O）

―

KMBV11425-W　<F,U>

出力対象外のオブジェクトが存在します。

クラス名：%1

Object of can not output contains.

Class name : %1

オブジェクト指定ファイルに指定したクラス（%1）に，全文検索用オブジェクトが含まれています。全文検索用オブジェクトは出力対象外のためスキップします。

（S）

処理を続行します。

（O）

―

KMBV11426-W　<C,P,F,U>

コンテナバージョンと関連付けされている ConfigurationHistory **クラスに** **PROP_DCR** **が指定されています。**

エントリ：%1

System property **PROP_DCR** found in ConfigurationHistory Class with ContainerVersion.

Entry : %1

VersionTraceableContainmentRelationship オブジェクトによってバージョン付き構成管理コンテナに関連づけられた ConfigurationHistory クラスに，**PROP_DCR** が指定されています。不要な **PROP_DCR** の指定がある場合は，オブジェクトローダ入力データファイルが正しく生成されていません。

（S）

処理を続行します。

（O）

制御ファイルの DataMapping セクションを見直して，不要な **PROP_DCR** の指定を削除してから再実行してください。

KMBV11427-I <P,F,U>

文書オブジェクトの出力をスキップしました。

文書コード：%1

スキップしたオブジェクト数：%2

Document Object output processing was Skipped.

Document Code : %1

Skipped Object : %2

文書オブジェクトの出力をスキップしました。スキップされた文書の内容を文書コードで確認してください。

文書コード	文書の内容
0	マルチファイル文書，リファレンスファイル文書以外の文書
1	マルチファイル文書
2	リファレンスファイル文書

（S）

処理を続行します。

（O）

―

KMBV11900-I <C,P,F>

%1 コマンドを実行中です。

It is executing %1.

%1 コマンドを実行中です。

（S）

%1 コマンドを実行中です。

（O）

―

KMBV11901-I <C,P,F>

%1 コマンドの処理が終了しました。

It is success to execute %1.

%1 コマンドの処理が終了しました。

（S）

%1 コマンドの処理を終了します。

（O）

―

KMBV11902-E <C,P,F>

入力データファイルの生成でエラーが発生しました。

要因コード：%1

An error occurred in creating of InputData file.

Reason code : %1

入力データファイルの生成でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	パスが存在しないかまたは同一名称のディレクトリが存在します。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルへのアクセス権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11903-E <C,P,F>

制御ファイルの生成でエラーが発生しました。

要因コード：%1

An error occurred in creating of Control file.

Reason code : %1

制御ファイルの生成でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	パスが存在しないかまたは同一名称のディレクトリが存在します。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルへのアクセス権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11904-E　<C,P,F>

定義ファイルの生成でエラーが発生しました。

要因コード：%1

An error occurred in creating of Define file.

Reason code : %1

定義ファイルの生成でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	パスが存在しないかまたは同一名称のディレクトリが存在します。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルへのアクセス権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11905-E　<C,P,F>

データベースへの接続でエラーが発生しました。

An error occurred in connecting database.

データベースに接続できません。データベースサーバが起動していないか，接続情報の定義ファイル（docspace.ini）が存在しないか，定義内容が誤っている可能性があります。

（S）

処理を終了します。

（O）

データベースの起動または接続情報の定義を確認してください。

KMBV11906-E <C,P,F>

生成対象のテーブルがありません。

There is no table.

コマンドの引数に指定したユーザ ID のスキーマに，DocumentBroker サーバが提供するテーブルがありませんでした。

（S）

処理を終了します。

（O）

指定したユーザ ID を見直してください。

KMBV11907-E <P>

他のプログラムが実行中のため，コマンドが実行できません。

Unable to execute this command because other program is executing.

DocumentBroker サーバ開始中に本機能を実行しようとしたか，または本機能を複数実行しようとしています。

（S）

処理を終了します。

（O）

DocumentBroker サーバを停止後，または実行中の処理が終了してから再度実行してください。

KMBV11909-E <P>

指定したコマンドの形式に誤りがあります。コマンドの形式は次のとおりです。

EDMCrtLDF **ユーザ** ID [-d **定義ファイル**]

Error exists in command line, usage is as follows

Usage: EDMCrtLDF USER_ID [-d DefineFile]

指定したコマンド形式に誤りがあります。

（S）

処理を終了します。

（O）

正しいコマンド形式を指定して再度実行してください。

KMBV11910-W <C,P,F>

メタ情報（%1）に不正な定義があるため，定義ファイルに出力できませんでした。

Invalid definition in META file (%1)

メタ情報（%1）は，DisplayName と GUID 値の数が合っていないため処理できません。

（S）

エラーが発生したメタ情報を無視して処理を続行します。

（O）

メタ情報ファイルの定義内容，およびデータベースの定義を見直してください。

KMBV11912-E <C,P,F>

クラス関連ファイルのオープンでエラーが発生しました。

要因コード：%1

An error occurred in opening of ClassRelationFile.

Reason code : %1

クラス関連ファイルの生成でエラーが発生しました。要因コードでエラー内容を確認してください。

要因コード	エラー内容
1	クラス関連ファイルが存在しません。
2	パスの長さが制限値（255 バイト）を超えています。
3	ファイルへのアクセス権限がありません。

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11913-W <C,P,F>

クラス関連ファイルに指定したクラス名が不正です。

クラス名：%1

Invalid Class name exists in ClassRelationFile.

Class name : %1

クラス関連ファイルに指定されているクラス名（%1）は，メタ情報にありません。

（S）

エラーが発生したクラス名を無視して処理を続行します。

（O）

クラス関連ファイルの定義内容を見直してください。

KMBV11914-W <C,P,F>

クラス関連ファイルに指定したセクション名が不正です。

セクション名：%1

Invalid Section name exists in ClassRelationFile.

Section name : %1

クラス関連ファイルに指定されているセクション名（%1）は，CREATE_%1 に相当するコマンドが存在しないので処理できません。

（S）

エラーが発生したクラス名を無視して処理を続行します。

（O）

クラス関連ファイルの定義内容を見直してください。

KMBV11916-E <P>

指定したコマンドの形式に誤りがあります。コマンドの形式は次のとおりです。

EDMCrtLCF **ユーザ** ID **クラス関連ファイル** [-c **制御ファイル**] [-i **入力データファイル**] -L

Error exists in command line, usage is as follows

Usage: EDMCrtLCF USER_ID ClassRelationFile [-c ControlFile] [-i InputDataFile] -L

指定したコマンド形式に誤りがあります。

（S）

処理を終了します。

（O）

正しいコマンド形式を指定して再度実行してください。

KMBV11990-E <C,P,F>

メモリが不足しているため，処理を続行できません。

There is insufficient memory.

メモリが不足しました。

（S）

処理を終了します。

（O）

不要なプログラムを終了させて再度実行してください。

KMBV11991-E　<C,P,F>

ディスク容量が不足しているため，処理を続行できません。

There is insufficient space of disk.

ディスク容量が不足しました。

（S）

処理を終了します。

（O）

ディスクに空きがあることを確認してから再度実行してください。

KMBV11992-E　<C,P,F>

%1 で内部矛盾が発生しました。

エラー情報：%2

Internal error occurred to %1.

Error information : %2

%1 で内部矛盾が発生しました。

%2：エラー情報

（S）

処理を終了します。

（O）

トレースログを採取して保守員に連絡してください。

KMBV11993-E　<C,P,F>

データベースでエラーが発生しました。

エラー情報：%1

要因コード：%2

An error occurred in database.

Error information : %1

Reason code : %2

データベースへのアクセスでエラーが発生しました。

%1：データベースのエラーメッセージ

%2：データベースのエラーコード

（S）

処理を終了します。

（O）

エラーの要因を取り除いてください。

KMBV11994-E <C,P,F>

システムコールでエラーが発生しました。

関数名：%1

要因コード：%2

An error occurred on the system call function.

Function Name : %1

Reason Code : %2

システムコール（%1）でエラーが発生しました。

%1：エラーの発生した関数名

%2：システムコールで返された errno 値

（S）

処理を終了します。

（O）

%2 に出力される要因コードを基に障害の要因を取り除いてください。

KMBV20100-W <C,P>

NTconfig.ini のエントリ %1 に指定したディレクトリが不正です。デフォルトディレクトリを使用します。

Invalid directory that specified %1 entry in NTconfig.ini. This Program uses default directory.

NTconfig.ini のエントリ %1 に指定したディレクトリが不正です。デフォルトディレクトリを使用します。

（S）

処理を続行します。

（O）

―

KMBV20101-W <C,P>

NTconfig.ini にエントリ %1 が複数存在します。最上位の指定を有効にします。

Same entry %1 exists in NTconfig.ini. First entry is used.

NTconfig.ini にエントリ %1 が複数存在します。最上位の指定を有効にします。

（S）

処理を続行します。

（O）

―

KMBV20102-E <C,P>

NTconfig.ini ファイル（%1）のオープンでエラーが発生しました。

要因コード：%2

An error occurred in opening of NTconfig.ini file (%1).

Reason code : %2

NTconfig.ini ファイル（%1）のオープンでエラーが発生しました。
要因コード（%2）でエラー内容を確認してください。

要因コード	エラー内容
1	アクセス権限がありません。
2	ファイルが存在しないか，環境変数 OBJLOADERDIR が不正です。
3	パス名が 255 バイトを超えています。

（S）

処理を終了します。

（O）

%2 に出力される要因コードを基にエラーの要因を取り除いてください。

付録

付録 A　作成コマンドと対象テーブルの関連

オブジェクトローダで提供する作成コマンドと，コマンドを実行した時にデータのインサート対象となるテーブルの関連を次に示します。

表 A-1　作成コマンドと対象テーブルの関連

作成コマンド名	対象テーブル	備考
CREATE_RFCT	Container	—
	DirectContainmentRelationship	**PROP_DCR** の場合
	ReferentialContainmentRelationship	**PROP_RCR** の場合
CREATE_DOC	DocVersion	—
	全文検索機能付き DocVersion	XML 文書の場合
	ContentTransfer	**PROP_CT** の場合
	ContentTransfers	マルチファイル文書の場合
	ContentReference(edm)	リファレンスファイル文書の場合
CREATE_VRDOC	ConfigurationHistory	—
	VersionSeries	—
	VersionDescription	—
	DocVersion	—
	全文検索機能付き DocVersion	XML 文書の場合
	ContentTransfer	**PROP_CT** の場合
	ContentTransfers	マルチファイル文書の場合
	ContentReference(edm)	リファレンスファイル文書の場合
CREATE_DATA	IndependentPersistence	—
CREATE_VARRAY	STRUCT	—
CREATE_VRCV	DirectContainmentRelationship	**PROP_DCR** の場合
	ReferentialContainmentRelationship	**PROP_RCR** の場合
	ConfigurationHistory	—
	VersionSeries	—
	VersionDescription	—
	ContainerVersion	—
	VersionTraceableContainmentRelationship	**PROP_VCR** の場合
CREATE_CV	DirectContainmentRelationship	**PROP_DCR** の場合
	ReferentialContainmentRelationship	**PROP_RCR** の場合

作成コマンド名	対象テーブル	備考
	ContainerVersion	—
	VersionTraceableContainmentRelationship	**PROP_VCR** の場合
CREATE_PACL	PublicACL	—

付録 B　制御ファイルの状態表

EDMCrtLCF 実行時に指定する制御ファイルの出力内容の状態表を次に示します。

表 B-1　制御ファイルの状態表

セクション	DocumentBroker サーバが ACL 非対応の場合	DocumentBroker サーバが ACL 対応の場合（ACL 対応の場合に追加出力されるプロパティ）
[Control]	○	○
ErrorLog	ErrorLog	ErrorLog
ErrorDataFile	ErrorDataFile	ErrorDataFile
StopCount	1	1
ColumnSeparator	","	","
RecordSeparator	"¥n"	"¥n"
StartPoint	1	1
OIID_Count	0	0
MsgInterval	10000	10000
XmlBroker	HAX	HAX
[DataMapping]	○	○
Container	出力プロパティなし	**PROP_ACL_OID**
DocVersion	**PROP_RTYPE**, **PROP_CT**	**PROP_ACL_OID**
ConfigurationHistory (CREATE_VRDOC 用)	**PROP_RTYPE**, **PROP_DOC_CLASS**, **PROP_CT**	**PROP_ACL_OID**
IndependentPersistence	出力プロパティなし	**PROP_ACL_OID**
Struct	**PROP_OBJECT**	出力プロパティなし
ConfigurationHistory (CREATE_VRCV 用)	**PROP_CV_CLASS**	**PROP_ACL_OID**
ContainerVersion	出力プロパティなし	**PROP_ACL_OID**
PublicACL	対象外※	**PROP_ACL_SUBJECT**, **PROP_ACL_STYPE**, **PROP_ACL_PERM**, **PROP_ACL_OID**
[Export]	○	○
DocDir	UNIX の場合 /tmp/ObjectExport Windows の場合 < システムドライブ >:¥tmp¥ObjectExport	UNIX の場合 /tmp/ObjectExport Windows の場合 < システムドライブ >:¥tmp¥ObjectExport

（凡例）

○：出力される

注※

DocumentBroker サーバの設定でアクセス制御機能に対応していない場合，エントリ自体が出力されません。

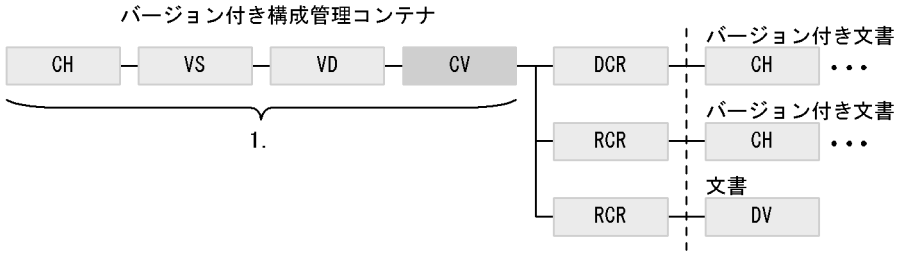
付録 C　構成管理コンテナの登録方法

ここでは，オブジェクトローダが提供する構成管理機能のオブジェクトモデルおよびその登録方法について説明します。

(1) オブジェクトローダが提供する構成管理機能のオブジェクトモデル

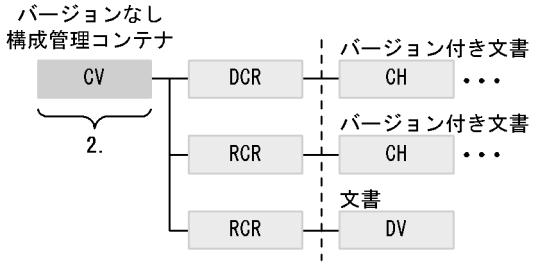
構成管理機能を実現するためには，構成管理コンテナの作成が必要になります。バージョン付き構成管理コンテナのオブジェクトモデルを図 C-1 に，バージョンなし構成管理コンテナのオブジェクトモデルを図 C-2 に示します。

図 C-1　バージョン付き構成管理コンテナのオブジェクトモデル



（凡例）
CH: ConfigurationHistoryオブジェクト　DCR: DirectContainmentRelationshipオブジェクト
VS: VersionSeriesオブジェクト　RCR: ReferentialContainmentRelationshipオブジェクト
VD: VersionDescriptionオブジェクト　DV: DocVersionオブジェクト
CV: ContainerVersionオブジェクト

図 C-2　バージョンなし構成管理コンテナのオブジェクトモデル



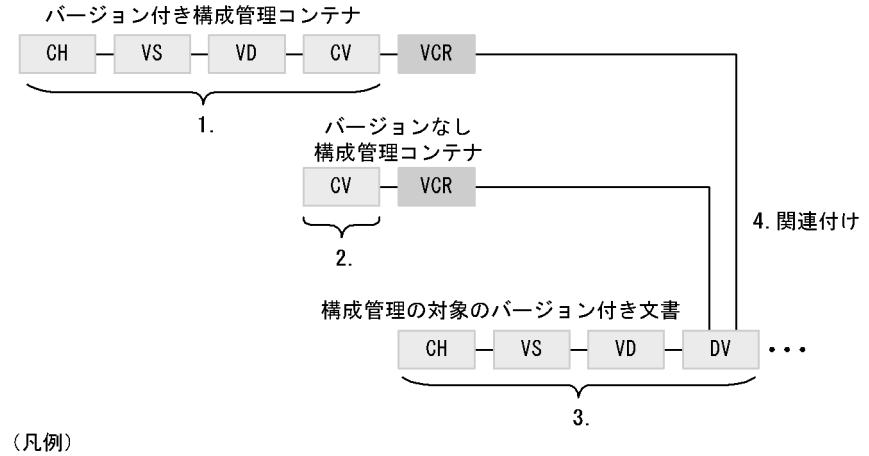
（凡例）
CV:　ContainerVersionオブジェクト
DCR: DirectContainmentRelationshipオブジェクト
RCR: ReferentialContainmentRelationshipオブジェクト
CH:　ConfigurationHistoryオブジェクト
DV:　DocVersionオブジェクト

オブジェクトローダでは図 C-1 の 1.（バージョン付き構成管理コンテナ）を作成するために CREATE_VRCV コマンドを，図 C-2 の 2.（バージョンなし構成管理コンテナ）を作成するために CREATE_CV コマンドを提供します。

次に，構成管理コンテナを使用した構成管理機能を実現するためのオブジェクトモデル

を次に示します。

図 C-3　構成管理コンテナを使用した構成管理機能のオブジェクトモデル



（凡例）
CH: ConfigurationHistoryオブジェクト
VS: VersionSeriesオブジェクト
VD: VersionDescriptionオブジェクト
CV: ContainerVersionオブジェクト
VCR: VersionTraceableContainmentRelationshipオブジェクト
DV: DocVersionオブジェクト

図 C-3 のように，VersionTraceableContainmentRelationship オブジェクトで，構成管理コンテナと文書オブジェクトを関連づけます。

(2) 構成管理のためのオブジェクトの登録方法

オブジェクトローダで構成管理のためのオブジェクトを登録するためには，まず構成管理の対象である文書オブジェクト（図 C-3 の 3.）を作成します。次に，構成管理コンテナ（バージョン付きまたはバージョンなし）と，作成した文書オブジェクトを関連づけます（図 C-3 の 4.）。

構成管理の対象の文書オブジェクトの作成

構成管理の対象の文書オブジェクトは，提供コマンド CREATE_VRDOC を使用して作成します。システム定義プロパティ（**PROP_DOC_CLASS**）には，スーパークラスが edmClass_VersionTracedDocVersion，または edmClass_VersionTracedComponentDocVersion であるユーザクラス（サブクラス）を指定してください。

構成管理コンテナと文書オブジェクトの関連づけ

制御ファイルの DataMapping セクションに，図 C-3 の 1. または 2. で使用するコマンド（CREATE_VRCV または CREATE_CV）用のユーザクラスに対して，システム定義プロパティ（**PROP_VCR**）をセットして，図 C-3 の 3. の CREATE_VRDOC 実行時に指定したラベルを指定します。

次に，構成管理オブジェクトの制御ファイルおよび入力データファイルの記述例（Windows の場合）を次に示します。UNIX の場合，ディレクトリの区切り文字が「/」になります。

制御ファイルの記述例

```
[DataMapping]
USER_CH=**PROP_DOC_CLASS**,**PROP_RTYPE**,**PROP_CT**
USER_VRCH=**PROP_CV_CLASS**,**PROP_VCR**,**PROP_VTMODE**
```

入力データファイルの記述例

```
#構成管理の対象のバージョン付文書を作成する
CREATE_VRDOC,LABEL_DOC,USER_CH,[USER_VTDoc],"MIME::text/
ms-word","C:¥tmp¥Document.doc"
#VCRを使用して構成管理コンテナと関連づける
CREATE_VRCV,,USER_VRCH,[USER_CV],[Label/LABEL_DOC],FIX
```

付録 D　ConfigurationHistory クラスのエクスポート

ここでは，異なるクラスのオブジェクト（バージョン付き文書，構成管理コンテナ）が混在する ConfigurationHistory クラスに対してオブジェクトエクスポートを実行する場合の，実行方法および注意事項について説明します。

クラスが混在している場合の例を次に示します。

図 D-1　異なるクラスのオブジェクトが混在する ConfigurationHistory クラスの例

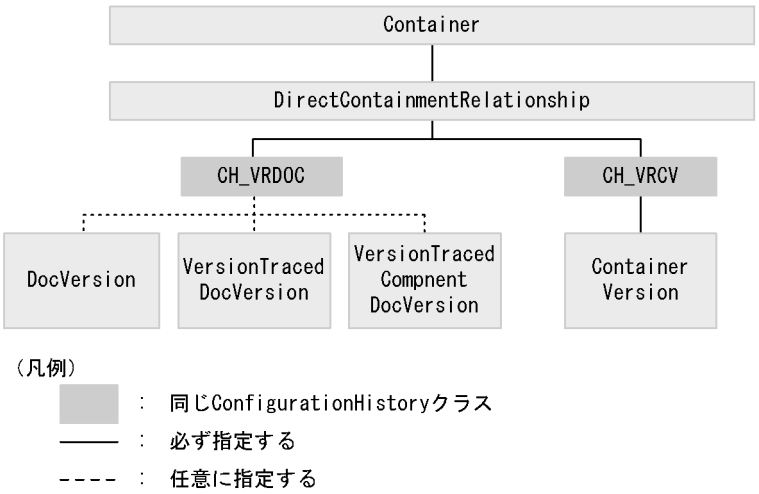


図 D-1 のように，同じ ConfigurationHistory クラスに，異なるクラスのオブジェクト（バージョン付き文書，構成管理コンテナ）が混在する場合，次のどちらかの方法でオブジェクトエクスポートを実行してください。

- ConfigurationHistory に関連づけられた Container クラスをエクスポートする
  詳細については，「(1)　ConfigurationHistory に関連づけられた Container クラスのエクスポート」を参照してください。
- 検索条件で特定の ConfigurationHistory クラスを指定する
  詳細については，「(2)　オブジェクト指定ファイルに ConfigurationHistory クラスを指定したエクスポート」を参照してください。

これらの場合，定義ファイルの ClassNameDefinition セクションおよび制御ファイルの DataMapping セクションには，異なるクラスの数だけ，ConfigurationHistory クラスを異なるクラス名で指定してください。指定例を次に示します。

定義ファイルの ClassNameDefinition セクションの指定例

```
#バージョン付き文書用のConfigurationHistoryクラス
CH_VRDOC=01234567-8901-2345-6789-012345678903
#構成管理コンテナ用のConfigurationHistoryクラス
CH_VRCV=01234567-8901-2345-6789-012345678903
```

制御ファイルの DataMapping セクションの指定例

```
CH_VRDOC =
**PROP_DCR**,**PROP_RTYPE**,**PROP_CT**,**PROP_DOC_CLASS**
CH_VRCV = **PROP_DCR**,**PROP_CV_CLASS**
```

オブジェクトエクスポートで使用する制御ファイルの詳細については，「5.3　制御ファイル」を参照してください。オブジェクトエクスポートで使用する定義ファイルの詳細については，「5.4　定義ファイル」を参照してください。

注意事項

- 定義ファイルの ClassNameDefinition セクションと制御ファイルの DataMapping セクションでは，記述順序を合わせてください。
- 制御ファイルの DataMapping セクションのクラス名が同一の定義が複数記述されている場合は，最初に指定したものだけが対象になります。
- 同一の ConfigurationHistory クラスで異なる文書クラスを使用している場合，ConfigurationHistory クラスをエクスポートできません。文書クラスが混在した Configuration クラスの詳細については，「(3)　異なる文書クラスが混在した ConfigurationHistory クラス」を参照してください。

（1）ConfigurationHistory に関連づけられた Container クラスのエクスポート

図 D-1 のモデルの文書空間で，関連づけられた Container クラスに対してオブジェクトエクスポートを実行することで，ConfigurationHistory クラスを一度にエクスポートできます。

Container クラスに対してオブジェクトエクスポートを実行する場合の注意事項を次に示します。

- オブジェクト指定ファイルには Container クラスを指定してください。
- 制御ファイルの DataMapping セクションには，Container クラス，バージョン付き文書の ConfigurationHistory クラス，および構成管理コンテナの ConfigurationHistory クラスを記述してください。
- 制御ファイルの DataMapping セクションの ConfigurationHistory クラスに **PROP_DCR** を指定してください。**PROP_DCR** を指定しない場合は，エラーメッセージ（KMBV11049-E）が出力されます。

オブジェクト指定ファイル，および制御ファイルの DataMapping セクションの記述例を次に示します。

オブジェクト指定ファイルの記述例

```
Query/Container
```

制御ファイルの DataMapping セクションの記述例

```
Container =
CH_VRDOC =
```

```
**PROP_DCR**,**PROP_RTYPE**,**PROP_CT**,**PROP_DOC_CLASS**
CH_VRCV = **PROP_DCR**,**PROP_CV_CLASS**
```

（2） オブジェクト指定ファイルに ConfigurationHistory クラスを指定したエクスポート

図 D-1 のモデルの文書空間で，検索条件で該当するクラスを対象とする条件が指定できる場合，オブジェクト指定ファイルに直接 ConfigurationHistory を指定してオブジェクトエクスポートを実行できます。

オブジェクト指定ファイルに直接 ConfigurationHistory を指定してオブジェクトエクスポートを実行するときの注意事項を次に示します。

- クラスを対象とする条件が指定できない場合，エラーメッセージ（KMBV11049-E）が出力されます。
- オブジェクト指定ファイルには，エクスポートする ConfigurationHistory クラスを指定してください。
- 制御ファイルの DataMapping セクションには，エクスポートする ConfigurationHistory クラス（オブジェクト指定ファイルに指定したクラス）を記述してください。
- 制御ファイルの DataMapping セクションの ConfigurationHistory クラスには，**PROP_DCR** を指定しないでください。**PROP_DCR** を指定した場合は，エラーメッセージ（KMBV11049-E）が出力されます。

オブジェクト指定ファイル，および制御ファイルの DataMapping セクションの記述例を次に示します。

オブジェクト指定ファイルの記述例

```
Query/CH_VRDOC/propA=VRDOC
Query/CH_VRCV/propA=VRCV
```

制御ファイルの DataMapping セクションの記述例

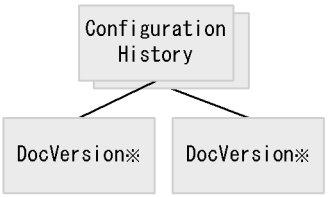
```
CH_VRDOC = **PROP_RTYPE**,**PROP_CT**,**PROP_DOC_CLASS**,propA
CH_VRCV = **PROP_CV_CLASS**,propA
```

（3） 異なる文書クラスが混在した ConfigurationHistory クラス

同一の ConfigurationHistory クラスで異なる文書クラスを使用している場合，ConfigurationHistory クラスをエクスポートできません。実行した場合，エラーメッセージ（KMBV11404-E）が出力されます。

エクスポートできる ConfigurationHistory クラスの例を次に示します。

図 D-2　エクスポートできる ConfigurationHistory クラスの例

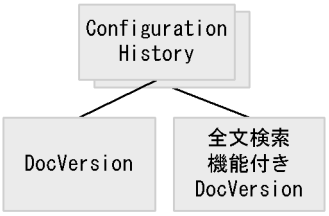


注※　同じDocVersionクラス

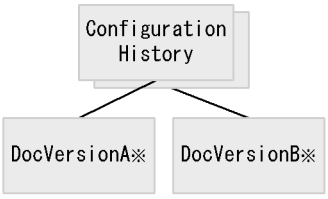
エクスポートできない ConfigurationHistory クラスの例を次に示します。

図 D-3　エクスポートできない ConfigurationHistory クラス

例1：DocVersionクラスと全文検索機能付き
DcoVersionクラスの混在



例2：異なるDocVersionクラスの混在



注※　異なるDocVersionクラス

付録 E　High-end Option

ここでは，High-end Option を使用して，複数のオブジェクトローダを同時に実行する方法について説明します。

付録 E.1　High-end Option の概要

High-end Option は，複数のオブジェクトローダを同時に実行できるようにするプログラムです。High-end Option を使用すると，同時に複数のオブジェクトローダを実行できるため，登録時間を短縮できます。

なお，以降の説明で，オブジェクトローダを一つだけ実行することを「**単一実行**」，同時に複数のオブジェクトローダを実行することを「**並列実行**」と呼びます。

付録 E.2　High-end Option を使用する環境の設定

ここでは，High-end Option を使用するための環境を設定する方法について説明します。

（1）High-end Option の前提環境

High-end Option は，DocumentBroker Object Loader と同一のサーバ上で使用してください。

High-end Option を使用できる DocumentBroker Object Loader の前提バージョンを次に示します。

- R-1M95E-43　uCosminexus DocumentBroker Object Loader Version 3　03-60 以降（適用 OS：AIX 5L V5.1，AIX 5L V5.2，AIX 5L V5.3）
- R-1595E-43　uCosminexus DocumentBroker Object Loader Version 3　03-60 以降（適用 OS：Windows Server 2003，Windows Server 2003 R2，Windows Server 2003 R2（x64），Windows Server 2008 x86）

（2）High-end Option のインストール

ここでは，High-end Option のインストールについて説明します。インストール方法は UNIX の場合と Windows の場合で異なります。

(a) UNIX の場合

UNIX の場合の High-end Option のインストール方法について説明します。

UNIX の場合の High-end Option のインストールディレクトリは /opt/HiEDMS_LodH です。デバイススペシャルファイル名や CD-ROM のマウントディレクトリは使用する環境によって異なります。詳細は，OS のマニュアルを参照してください。また，インストールの手順は，プログラムの提供媒体によって異なります。

DAT の場合

提供媒体が DAT の場合のインストール方法について次に示します。

＜操作＞

1. root 権限でログインします。
2. High-end Option のテープを DAT ドライブにセットします。
3. tar xf /dev/rmt/0m を実行します。
   「/dev/rmt/0m」の部分は使用環境によって異なります。使用環境に合わせてファイル名を変更してください。
4. /etc/hitachi_setup -i /dev/rmt/0m を実行します。
   初期画面が表示されます。
5. 「I）Install Software」を選択します。
6. インストールするプログラムにカーソルを移動させて，スペースキーで選択します。
   選択したプログラムの左側に＜@＞が付きます。
7. 「I）Install」を選択してから，最下行に表示されるメッセージに対して y または Y を入力します。
   インストールが始まります。
8. インストールが終了したら，「Q）Quit」を選択して終了します。

CD-ROM の場合

提供媒体が CD-ROM の場合のインストール方法について次に示します。

＜操作＞

1. root 権限でログインします。
2. マウント用の /cdrom ディレクトリを作成します。mkdir /cdrom を実行します。
   「/cdrom」の部分には，CD-ROM をファイルシステムとしてマウントするディレクトリ名を指定してください。
3. High-end Option の CD-ROM を CD-ROM ドライブにセットします。
4. CD-ROM をマウントします。
   AIX の場合，mount -r -v cdrfs /dev/cd0 /cdrom を実行します。
   「/dev/cd0 /cdrom」の部分は使用環境によって異なります。使用する CD-ROM デバイススペシャルファイル名を指定してください。
5. セットアッププログラムを実行します。
   AIX の場合，/cdrom/aix/setup /cdrom を実行します。
   すでに CD-ROM セットアッププログラムを実行し，日立 PP インストーラがハードディスク上にある場合は，直接日立 PP インストーラ「/etc/hitachi_setup -i /cdrom」を起動してください。
   なお，「/cdrom」の部分は使用環境によって異なります。使用する CD-ROM のマウントディレクトリ名を指定してください。
   初期画面が表示されます。
6. 「I）Install Software」を選択します。

7. **インストールするプログラムにカーソルを移動させて，スペースキーで選択します。**
   選択したプログラムの左側に＜@＞が付きます。
8. **「I）Install」を選択してから，最下行に表示されるメッセージに対して y または Y を入力します。**
   インストールが始まります。
9. **インストールが終了したら，「Q）Quit」を選択して終了します。**

High-end Option のインストールが終了したあとに，次のファイルが正しく作成されているか確認してください。

/opt/HiEDMS_LodH/adm/.edmload_highend.lck

（b）Windows の場合

Windows の場合の High-end Option のインストール方法について説明します。

＜操作＞

1. Administrators **グループのユーザでログインします。**
2. **インストールを実行する前にすべての** Windows **アプリケーションを終了させてください。**
3. **統合** CD-ROM **を** CD-ROM **ドライブにセットし，統合** CD-ROM **の手順に従ってインストールしてください。**
   インストール説明のダイアログが表示されます。
4. **「次へ」ボタンをクリックします。**
   ユーザ情報として「名前」と「会社名」を入力するダイアログが表示されます。上書きインストールの場合には，上書き確認ダイアログが表示されます。ここで「はい」ボタンをクリックすると「9. プログラムフォルダを指定します」から実行されます。
5. **「名前」と「会社名」を入力します。**
6. **「次へ」ボタンをクリックします。**
   インストール先のディレクトリを指定するダイアログが表示されます。
7. **インストール先ディレクトリを指定します。**
   指定したディレクトリの下にインストールされます。デフォルトのインストール先は，C:¥Program files¥HITACHI¥DocBroker_LodH¥ です。C:¥ は，OS がインストールされているドライブ名です。
8. **「次へ」ボタンをクリックします。**
   プログラムフォルダを指定するダイアログが表示されます。
9. **プログラムフォルダを指定します。**
   指定したプログラムフォルダにプログラムアイコンが追加されます。
10. **「次へ」ボタンをクリックします。**
    現在の設定（ユーザ情報，インストール先のディレクトリ，プログラムフォルダ）確認のダイアログが表示されます。
11. **設定を確認して，「次へ」ボタンをクリックします。**

インストールが開始されます。
インストールが終了すると，インストールが終了したことを通知するダイアログが表示されます。

12.**「完了」ボタンをクリックします。**
インストールプログラムを終了します。

注意事項

文書空間の文字コード種別が UTF-8 の場合，インストール先ディレクトリには，ASCII コードで表せるパスを指定してください。

（3）アンインストール

High-end Option のアンインストールの方法について説明します。アンインストール方法は UNIX の場合と Windows の場合で異なります。

（a）UNIX の場合

UNIX の場合のアンインストールの方法について説明します。アンインストールを実行すると，インストールしたディレクトリおよびファイルがすべて削除されます。

＜操作＞

1. root **権限でログインします。**
2. /etc/hitachi_setup **を実行します。**
   初期画面が表示されます。
3. **「D）Delete」を選択します。**
4. **アンインストールするプログラム（**DocumentBroker Object Loader High-end Option**）にカーソルを移動させて，スペースキーで選択します。**
   選択したプログラムの左側に＜@＞が付きます。
5. **「D）Delete」を選択してから，最下行に表示されるメッセージに対して** y **または** Y **を入力します。**
   アンインストールが始まります。
6. **アンインストールが終了したら，「**Q）Quit**」を選択して終了します。**

（b）Windows の場合

Windows の場合のアンインストールの方法について説明します。アンインストールを実行すると，インストールしたディレクトリおよびファイルがすべて削除されます。

アンインストールは Windows「コントロールパネル」の「アプリケーションの追加と削除」を使用して行います。

＜操作＞

1. Administrators **グループのユーザでログインします。**
2. **「コントロールパネル」の「アプリケーションの追加と削除」を実行します。**
   アプリケーションの追加と削除のプロパティが表示されます。
3. **インストールと削除タブでアンインストールするプログラム（**DocumentBroker

Object Loader High-end Option）を選択して「追加と削除」ボタンをクリックします。

付録 E.3　High-end Option と EDMLoad コマンドの関係

ここでは，High-end Option と EDMLoad コマンドの関係について説明します。

High-end Option および EDMLoad コマンドのオプションと EDMLoad の動作の関係を次に示します。なお，High-end Option と同時実行機能との関係については，「付録 F 同時実行機能の実行環境と運用上の注意事項」を参照してください。

表 E-1　High-end Option と EDMLoad コマンドの関係

EDMLoad コマンドのオプション	High-end Option	
	未インストール	インストール済み
"-S" を指定しない	単一実行	並列実行
"-S" を指定する	エラー※	単一実行

注※
コマンド形式が誤っているため実行できません。

High-end Option をインストールした環境で単一実行をさせる場合，EDMLoad コマンドに "-S" オプションを指定して実行してください。一つだけのオブジェクトローダを実行する場合，"-S" オプションを指定して単一実行にした方が，登録時間を短縮できます。

オブジェクトローダの並列実行をしているときに，"-S" を指定した EDMLoad コマンドを実行すると，"-S" を指定した EDMLoad コマンドの処理が終了するまで単一実行になります。"-S" を指定した EDMLoad コマンドの処理の終了後は，並列実行になります。

付録 E.4　High-end Option に関する注意事項

- オブジェクトローダの並列実行をする場合，それぞれの EDMLoad コマンドに対して制御ファイルを作成してください。また，制御ファイルの Control セクションの ErrorLog エントリおよび ErrorDataFile エントリには，制御ファイルごとに異なるパスを指定してください。同じパスを指定すると，オブジェクトローダの実行ごとに上書きされるため，正しいエラー情報を得ることができなくなります。
- オブジェクトローダの並列実行をしたときに発生するエラー情報やトレース情報は，オブジェクトローダが出力する情報を参照してください。
- SELECT_OBJECT を使用する場合，SELECT_OBJECT で指定しているクラスに登録する EDMLoad コマンドを同時に実行しないでください。エラーメッセージ（KMBV11083-E）が出力されて，実行できない場合があります。

付録 F　同時実行機能の実行環境と運用上の注意事項

ここでは，DocumentBroker サーバとオブジェクトローダを同時に実行できる**同時実行機能**を使用する場合の実行環境および運用上の注意事項について説明します。同時実行機能を使用して EDMLoad コマンドを実行することを**同時実行モード**，同時実行機能を使用しないで EDMLoad コマンドを実行することを**非同時実行モード**といいます。

(1) 実行環境

同時実行機能を使用できるのは，"-M" オプションを指定した場合です。

実行環境による EDMLoad コマンドの動作の違いを次に示します。

表 F-1　実行環境による EDMLoad コマンドの動作の違い

High-end Option	コマンドのオプション	サーバの状態	
		停止中	起動中
未インストール	"-M" を指定する	同時実行	同時実行
	"-M" を指定しない	非同時実行	エラー
インストール済み	"-M" を指定する	同時実行	同時実行
	"-S" を指定する	非同時実行	エラー
	"-M" および "-S" を指定しない	非同時実行	エラー

(2) 運用上の注意事項

同時実行機能を使用するときの注意事項について次に示します。

- 同時実行モードでオブジェクトローダ実行中に，DocumentBroker サーバを停止した場合でも，同時実行モードとしての処理が続行されます。
- 同時実行モードでオブジェクトローダ実行中に，DocumentBroker サーバを起動，停止および再起動してもかまいません。
- 非同時実行モードでオブジェクトローダ実行中に，DocumentBroker サーバを起動することはできません。
- 環境変数の指定に誤りがあり，同時実行機能を使用するためのライブラリがロードできない場合は，エラーになります。
- 同時実行時に TRANBEGIN と TRANCOMMIT の指定をして複数のコマンドを 1 トランザクションで登録すると，複数のコマンド単位の排他資源が必要になります。

付録 G　文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能およびファイル

ここでは，文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能およびファイルについて説明します。なお，文書空間で使用する文字コード種別が Shift-JIS の場合は，すべての機能およびファイルを使用できます。

(1) 文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能

文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能を表 G-1 に示します。

表 G-1　文書空間で使用する文字コード種別が UTF-8 の場合に使用できる機能

分類	機能名	使用可否
オブジェクトローダ	文書およびコンテナに対する関連オブジェクト	○
	バージョン付き文書およびコンテナに対する関連オブジェクト	○
	コンテナおよびコンテナ間の関連を示す関連オブジェクト	○
	独立永続化クラス	○
	可変長配列	○
	オブジェクトごとのアクセス制御情報（ACL）データの登録	○
	マルチレンディション文書の登録	○
	構成管理コンテナ	○
	XML 文書	－
	マルチファイル文書の登録	○
	リファレンスファイル文書の登録	○
	同時実行機能	○
オブジェクトエクスポート	文書およびコンテナに対する関連オブジェクト（DCR だけ）	○
	バージョン付き文書およびコンテナに対する関連オブジェクト（DCR だけ）	○
	コンテナおよびコンテナ間の関連を示す関連オブジェクト（DCR だけ）	○
	独立永続化クラス	○
	可変長配列	○
	オブジェクトごとのアクセス制御情報（ACL）データの抽出	○

分類	機能名	使用可否
	構成管理コンテナ	○
	XML 文書	－

（凡例）

○：使用できます。

－：使用できません。

（2）文書空間で使用する文字コード種別が UTF-8 の場合に使用できるファイル

ユーザが作成するファイルの文字コード種別には，文書空間で使用する文字コード種別および ASCII コードを使用するか，ASCII コードだけを使用します。そのため，文書空間で使用する文字コード種別が UTF-8 の場合，UTF-8 が使用できないファイルのときは，必ず ASCII コードで記述してください。

また，ファイルの文字コード種別として UTF-8 を使用する場合，ファイルには BOM を付けないでください。

(a) オブジェクトローダで使用するファイル

文書空間で使用する文字コード種別が UTF-8 の場合にオブジェクトローダで使用できるファイルを表 G-2 に示します。

表 G-2　文書空間で使用する文字コード種別が UTF-8 の場合にオブジェクトローダで使用できるファイル

ファイル	使用可否
制御ファイル	－
定義ファイル	－
入力データファイル	○※
Windows の場合 環境変数ファイル（NTconfig.ini）	－

（凡例）

○：使用できます。

－：使用できません。

注※

カラムによって UTF-8 を使用できるかどうかが異なります。文字コード種別として UTF-8 を使用できるカラムについては，表 G-3 を参照してください。

表 G-3　文字コード種別として UTF-8 を使用できるカラム（入力データファイルの場合）

コマンド分類	カラム位置	カラムの意味	使用可否
作成コマンド	1	コマンドカラム	－
	2	ラベルカラム	－
	3	クラス名カラム	－
	4 以降	プロパティ値カラム	○※1
選択コマンド	1	コマンドカラム	－
	2	ラベルカラム	－
	3	パラメタカラム	○※2
指示コマンド	1	コマンドカラム	－

（凡例）

○：使用できます。

－：使用できません。

注※ 1

次に示す項目の最大長は 255 バイトになります。なお，バイト数は，文字列を UTF-8 で表現した場合の長さです。

- コンテントのファイル名
- ファイル名から生成されるプロパティ値
- コンテントの絶対パス

また，次に示す項目は印刷可能な ASCII コードで記述してください。

- 文書オブジェクトのコンポーネントタイプ
- コンテント格納先ベースパス
- コンテント格納先パス

注※ 2

検索条件として指定するプロパティ値にだけ，UTF-8 を使用できます。

(b) オブジェクトエクスポートで使用するファイル

文書空間で使用する文字コード種別が UTF-8 の場合にオブジェクトエクスポートで使用できるファイルを表 G-4 に示します。

表 G-4　文書空間で使用する文字コード種別が UTF-8 の場合にオブジェクトエクスポートで使用できるファイル

ファイル	使用可否
制御ファイル	○※1
定義ファイル	－
オブジェクト指定ファイル	○※2

ファイル	使用可否
オブジェクトローダ入力データファイル	○※3
Windows の場合 環境変数ファイル（NTconfig.ini）	－

（凡例）

○：使用できます。

－：使用できません。

注※ 1

Export セクションの DocDir エントリにだけ，UTF-8 を使用できます。

注※ 2

オブジェクト指定ファイルに記述する項目によって，UTF-8 を使用できるかどうかが異なります。UTF-8 を使用できる項目を表 G-5 に示します。

注※ 3

次に示す項目の最大長は 255 バイトになります。なお，バイト数は，文字列を UTF-8 で表現した場合の長さです。

- コンテントのファイル名
- ファイル名から生成されるプロパティ値
- コンテントの絶対パス

表 G-5　文字コード種別として UTF-8 を使用できる項目（オブジェクト指定ファイルの場合）

項目		使用可否
クラス名		－
条件節	プロパティ名	－
	プロパティ値	○
struct サブクラス名		－

（凡例）

○：使用できます。

－：使用できません。

索引

記号

B

C

D

E

G

H

K

L

M

N

O

P

R

S

T

U

X

あ

い

う

え

お

か

き

く

け

こ

さ

し

す

せ

た

て

と

に

は

ひ

ふ

へ

ま

め

も

ゆ

よ

ら

り

ソフトウェアマニュアルのサービス ご案内

１．マニュアル情報ホームページ

ソフトウェアマニュアルの情報をインターネットで公開しています。

URL　http://www.hitachi.co.jp/soft/manual/

ホームページのメニューは次のとおりです。

■マニュアル一覧	日立コンピュータ製品マニュアルを製品カテゴリ，マニュアル名称，資料番号のいずれかから検索できます。
■CD-ROMマニュアル	日立ソフトウェアマニュアルと製品群別CD-ROMマニュアルの仕様について記載しています。
■マニュアルのご購入	マニュアルご購入時のお申し込み方法を記載しています。
■オンラインマニュアル	一部製品のマニュアルをインターネットで公開しています。
■サポートサービス	ソフトウェアサポートサービスお客様向けページでのマニュアル公開サービスを記載しています。
■ご意見・お問い合わせ	マニュアルに関するご意見，ご要望をお寄せください。

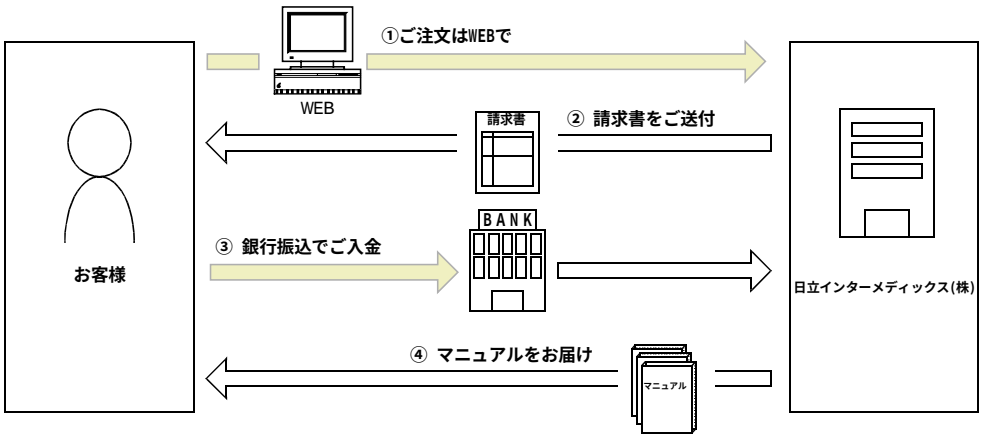
２．インターネットでのマニュアル公開

2 種類のマニュアル公開サービスを実施しています。

(1) マニュアル情報ホームページ「オンラインマニュアル」での公開
　製品をよりご理解いただくためのご参考として，一部製品のマニュアルを公開しています。

(2) ソフトウェアサポートサービスお客様向けページでのマニュアル公開
　ソフトウェアサポートサービスご契約のお客様向けにマニュアルを公開しています。公開しているマニュアルの一覧，本サービスの対象となる契約の種別などはマニュアル情報ホームページの「サポートサービス」をご参照ください。

３．マニュアルのご注文



① マニュアル情報ホームページの「マニュアルのご購入」にアクセスし，お申し込み方法をご確認のうえ WEB からご注文ください。ご注文先は日立インターメディックス(株)となります。

② ご注文いただいたマニュアルについて請求書をお送りします。

③ 請求書の金額を指定銀行へ振り込んでください。

④ 入金確認後 7 日以内にお届けします。在庫切れの場合は，納期を別途ご案内いたします。