

**IBM WebSphere Business Integration
Adapters**



Adapter for eMatrix ユーザーズ・ガイド

V 2.1.0

お願い

本書および本書で紹介する製品をご使用になる前に、91ページの『特記事項』に記載されている情報をお読みください。

本書は、新しい版で明記されていない限り、Adapter for eMatrix バージョン 2.1.0 および以降のすべてのリリースおよびモディフィケーションに適用されます。

本マニュアルに関するご意見やご感想は、次の URL からお送りください。今後の参考にさせていただきます。

<http://www.ibm.com/jp/manuals/main/mail.html>

なお、日本 IBM 発行のマニュアルはインターネット経由でもご購入いただけます。詳しくは

<http://www.ibm.com/jp/manuals/> の「ご注文について」をご覧ください。

(URL は、変更になる場合があります)

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

原 典：	IBM WebSphere Business Integration Adapters Adapter for eMatrix User Guide V 2.1.0
発 行：	日本アイ・ビー・エム株式会社
担 当：	ナショナル・ランゲージ・サポート

第 1 刷 2004.1

この文書では、平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、平成角ゴシック体™W5、および平成角ゴシック体™W7を使用しています。この(書体*)は、(財)日本規格協会と使用契約を締結し使用しているものです。フォントとして無断複製することは禁止されています。

注* 平成明朝体™W3、平成明朝体™W9、平成角ゴシック体™W3、
平成角ゴシック体™W5、平成角ゴシック体™W7

© Copyright International Business Machines Corporation 2002, 2003. All rights reserved.

© Copyright IBM Japan 2004

目次

本書について	v
対象読者	v
関連文書	v
表記上の規則	vi
本リリースの新機能	vii
リリース 2.1.0 の新機能	vii
リリース 2.0.x の新機能	vii
リリース 1.0.x の新機能	vii
第 1 章 アダプターの概要	1
アダプター・アーキテクチャー	1
アダプターの動作方法	4
サンプル・シナリオ	6
第 2 章 アダプターのインストール	7
アダプター環境	7
インストール作業の概要	8
アダプターおよび関連ファイルのインストール	9
インストール済みファイルの構造	9
第 3 章 アダプターの構成	11
アダプターの構成	11
複数のコネクタ・インスタンスの作成	14
コネクタの始動	16
eMatrix アダプターの始動	17
コネクタの停止	17
ログ・ファイルとトレース・ファイルの使用	18
第 4 章 ビジネス・オブジェクトの理解	21
メタデータの定義	21
ビジネス・オブジェクト構造の概要	22
eMatrix のビジネス・オブジェクト	22
ビジネス・オブジェクトの生成	29
第 5 章 ビジネス・オブジェクト定義の生成	31
eMatrix 用の ODA の概要	31
ビジネス・オブジェクト定義の生成	32
ファイルのアップロード	38
第 6 章 トラブルシューティングおよびエラー処理	41
エラー処理	41
トラブルシューティング	42
付録 A. コネクタの標準構成プロパティ	47
新規プロパティと削除されたプロパティ	47
標準コネクタ・プロパティの構成	47
標準プロパティの要約	49
標準構成プロパティ	52
付録 B. Connector Configurator	65

Connector Configurator の概要	65
Connector Configurator の始動	66
System Manager からの Configurator の実行	67
コネクタ固有のプロパティ・テンプレートの作成	67
新しい構成ファイルの作成	70
既存ファイルの使用	71
構成ファイルの完成	72
構成ファイル・プロパティの設定	73
構成ファイルの保管	80
構成ファイルの変更	80
構成の完了	81
グローバル化環境における Connector Configurator の使用	81
付録 C. サンプル・シナリオ	83
概要	83
前提事項	83
シナリオのインストール	83
シナリオの実行	88
特記事項	91
プログラミング・インターフェース情報	92
商標	93

本書について

IBM[®] WebSphere[®] Business Integration Adapter ポートフォリオは、主要な e-business テクノロジー、エンタープライズ・アプリケーション、およびレガシー・システムとメインフレーム・システムに、統合接続性を提供します。製品セットには、ビジネス・プロセスの統合に向けてコンポーネントをカスタマイズ、作成、および管理するためのツールとテンプレートが含まれています。

本書では、IBM WebSphere Business Integration Adapter for eMatrix のインストール、構成、トラブルシューティング、およびビジネス・オブジェクト開発について説明します。

対象読者

本書は、お客様のサイトでアダプターを使用するコンサルタント、開発者、システム・インテグレーター、およびシステム管理者を対象としています。

関連文書

この製品に付属する資料の完全セットで、すべての WebSphere Business Integration Adapters のインストールに共通な機能とコンポーネントについて説明します。また、特定のコンポーネントに関する参考資料も含まれています。

以下のサイトから、関連資料をインストールすることができます。

- アダプターの一般情報、WebSphere Message Brokers (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、WebSphere Business Integration Message Broker) におけるアダプターの使用、および WebSphere Application Server におけるアダプターの使用については、次のアドレスの IBM WebSphere Business Integration Adapters InfoCenter を参照してください。
<http://www.ibm.com/websphere/integration/wbiadapters/infocenter>
- WebSphere InterChange Server におけるアダプターの使用については、次のアドレスの IBM WebSphere InterChange Server InfoCenters を参照してください。
<http://www.ibm.com/websphere/integration/wicsserver/infocenter>
<http://www.ibm.com/websphere/integration/wbicollaborations/infocenter>
- WebSphere Message Brokers の詳細については、以下を参照してください。
<http://www.ibm.com/software/integration/mqfamily/library/manualsa/>
- WebSphere Application Server の詳細については、以下を参照してください。
<http://www.ibm.com/software/webservers/appserv/library.html>

上記のサイトには資料のダウンロード、インストール、および表示に関する簡単な説明が記載されています。

表記上の規則

本書では、以下のような規則を使用しています。

<code>courier font</code>	コマンド名、ファイル名、入力情報、システムが画面に出力した情報など、記述されたとおりの値を示します。
イタリック 青のアウトライン	初出語、変数名、または相互参照を示します。 オンラインで表示したときのみ見られる青のアウトラインは、相互参照用のハイパーリンクです。アウトラインの内側をクリックすると、参照先オブジェクトにジャンプします。
{ }	構文の記述行の場合、中括弧 {} で囲まれた部分は、選択対象のオプションです。1 つのオプションのみを選択する必要があります。
	構文の記述行の場合、パイプで区切られた部分は、選択対象のオプションです。1 つのオプションだけを選択する必要があります。
[]	構文の記述行の場合、大括弧 [] で囲まれた部分は、オプションのパラメーターです。
...	構文の記述行の場合、省略符号 ... は直前のパラメーターが繰り返されることを示します。例えば、 <code>option[,...]</code> は、複数のオプションをコンマで区切って指定できることを意味します。
< >	1 つの名前の個々のエレメントを互いに区別するために、不等号括弧によって個々のエレメントが囲まれます。例えば、 <code><server_name><connector_name>tmp.log</code> のように使用します。
/, ¥	本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムの場合には、円記号をスラッシュ (/) に置き換えてください。すべての製品パス名は、システム上の、eMatrix 用のコネクタがインストールされているディレクトリーを基準とした相対パス名です。
%text% および \$text	% 記号内にあるテキストは、Windows の text システム変数またはユーザー変数の値を示します。UNIX 環境での同等の表記は \$text です。これは、UNIX 環境変数 text の値を示します。
<i>ProductDir</i>	製品のインストール先ディレクトリーを表します。IBM WebSphere InterChange Server の場合、デフォルトの製品ディレクトリーは IBM¥WebSphereICS です。IBM WebSphere Business Integration Adapters の場合、デフォルトの製品ディレクトリーは WebSphereAdapters です。

本リリースの新機能

リリース 2.1.0 の新機能

Adapter for eMatrix のバージョン 2.1.0 は、eMatrix Server および コラボレーション・サーバー、バージョン 10.0.0.0 をサポートします。

バージョン 2.1.0 以降、Adapter for eMatrix は、Microsoft Windows NT 上でサポートされなくなりました。

リリース 2.0.x の新機能

2003 年 7 月更新。Adapter for eMatrix は、WebSphere Application Server を統合ブローカーとして使用できるようになりました。

アダプターは以下のプラットフォーム上で稼働します。

- Solaris 7、8
- AIX 5.1
- HP-UX 11.i

リリース 1.0.x の新機能

2003 年 3 月更新。「CrossWorlds」という名前を使用してシステム全体を示したりコンポーネントまたはツールの名前を変更したりすることはなくなりましたが、その他の点では以前とほとんど同じです。例えば、「CrossWorlds System Manager」は現在では「System Manager」となり、「CrossWorlds InterChange Server」は「WebSphere InterChange Server」となっています。

eMatrix アダプターは国際化されましたが、メッセージは翻訳されていません。

第 1 章 アダプターの概要

本章では、IBM(R) WebSphere(R) Business Integration adapter for eMatrix(R) および関連するシステム・アーキテクチャーについて説明します。

MatrixOne eMatrix(TM) は、プロダクト・ライフサイクル・マネージメント・ソリューションです。これを使用すると、プロダクト・ライフサイクルに関連したすべての文書およびデータ・フローを中央制御し、それらに対するアクセスをモニターすることができます。Adapter for eMatrix は、eMatrix グローバル・コラボレーション・プラットフォームとその他のエンタープライズ・アプリケーション間の、リアルタイムで双方向の統合を可能にします。

アダプターは、アプリケーション自体とは別個にインストールされる、eMatrix コラボレーション・サーバーと通信します。eMatrix コラボレーション・サーバーのダウンロードおよびインストールの詳細については、eMatrix の資料を参照してください。

アダプターは、WebSphere Message Brokers、WebSphere InterChange Server (ICS)、および WebSphere Application Server (WAS) の 3 つの統合ブローカーと一緒に動作します。統合ブローカーとアダプターの関係の詳細については、「*IBM WebSphere Business Integration システム管理ガイド*」を参照してください。

本章の内容は、次のとおりです。

- 1 ページの『アダプター・アーキテクチャー』
- 2 ページの『アーキテクチャーの概要』
- 3 ページの『イベント通知』

アダプター・アーキテクチャー

Adapter for eMatrix は、コネクター、Java Program Object (JPO)、およびアダプター・フレームワークを持った Object Discovery agent (ODA) で構成されています。それらが一緒に動作することによって、eMatrix は、統合ブローカーと情報を交換できるようになります。その結果、会計および CRM のパッケージなどの外部アプリケーションとも通信できるようになります。

このアダプターには、以下の 3 つの基本コンポーネントが含まれます。

- **コネクター**

データの転送において、統合ブローカーとアプリケーションの間の中継を行います。コネクターはメタデータ主導型です。

- **Java Program Object (JPO)**

複数存在することもあります。eMatrix 内で動作し、eMatrix データベース内のイベントに関する情報を保管します。

- **Object Discovery Agent (ODA)**

設計時ツールであり、アプリケーションに照会し、eMatrix の既存ビジネス・オブジェクトのタイプおよび関係を確認します。これらが、WebSphere ビジネス・オ

プロジェクト定義の基礎になります。コネクタは、実行時にこれらの定義を使ってアプリケーション・データをビジネス・オブジェクトに変換します。

アダプターは、eMatrix ビジネス・オブジェクトおよび eMatrix 内の接続を操作します。MQL ステートメントを実行することもあります。MQL コマンドを使用することによって、アダプターは、eMatrix によってサポートされるほとんどすべてのアクションを実行できます。

アダプターは、eMatrix アダプター開発キット (ADK) と呼ばれる Java ベースの API および別個の eMatrix コラボレーション・サーバーを使用して、eMatrix アプリケーションと通信します。eMatrix ADK を使用すると、アダプターは、eMatrix コラボレーション・サーバーを介してアプリケーションに要求を渡すことにより、アプリケーション内のオブジェクトに照会したり、それら进行操作したりできます。

アーキテクチャーの概要

アダプターは、統合ブローカーおよびアプリケーションとの間で、データ・フローを取り次ぎます。フローは以下のとおりです。

- 要求処理。統合ブローカー (ICS、WebSphere Message Broker、または WAS) によって開始されます。
- イベント処理。eMatrix アプリケーションによって開始されます。

これらのフローについて、以下で詳しく説明します。

要求処理

要求処理は統合ブローカーによって開始されます。統合ブローカーはビジネス・オブジェクトを Adapter for eMatrix に渡します。アダプターは、Java ベースの eMatrix ADK を使用して eMatrix コラボレーション・サーバーと通信します。3 ページの図 1 は、データ・フローのアーキテクチャーの概要を示しています。

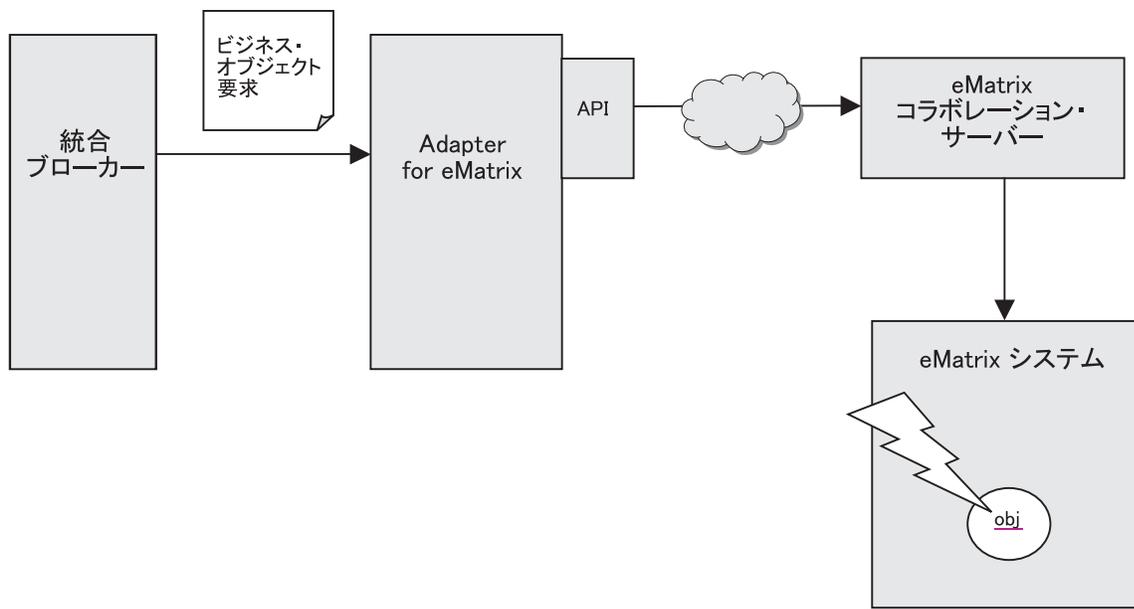


図 1. eMatrix アダプターにおける要求処理

要求処理は以下のように行われます。

1. 統合ブローカーは、ビジネス・オブジェクトの形でアダプターに要求を発行します。このオブジェクトには、タイプ情報およびビジネス・オブジェクトの動詞が含まれています。
2. アダプターは、該当する eMatrix ADK 呼び出しを行うことによって、ビジネス・オブジェクトを処理します。
3. eMatrix ADK 呼び出しが行われるたびに、コラボレーション・サーバーは、指定されたアクションを eMatrix システム内の該当するオブジェクトに対して実行し、結果をコネクターに戻します。
4. 要求の処理を完了すると、アダプターは、状況コードを統合ブローカーに戻します。必要な場合、変更結果を持ったビジネス・オブジェクトも戻します。

イベント通知

Adapter for eMatrix は、実行時にイベント・ポーリングを使用し、eMatrix システム内のイベントを検出します。4 ページの図 2 は、統合ブローカーがアダプターを介してアプリケーションをポーリングしたときに発生する、イベントの順序を示しています。

注: ポーリングを行うには、最初に、モニターしたいイベントの eMatrix トリガーを構成する必要があります。これは、初期セットアップ時に行われます。詳細については、11 ページの『アダプターの構成』を参照してください。

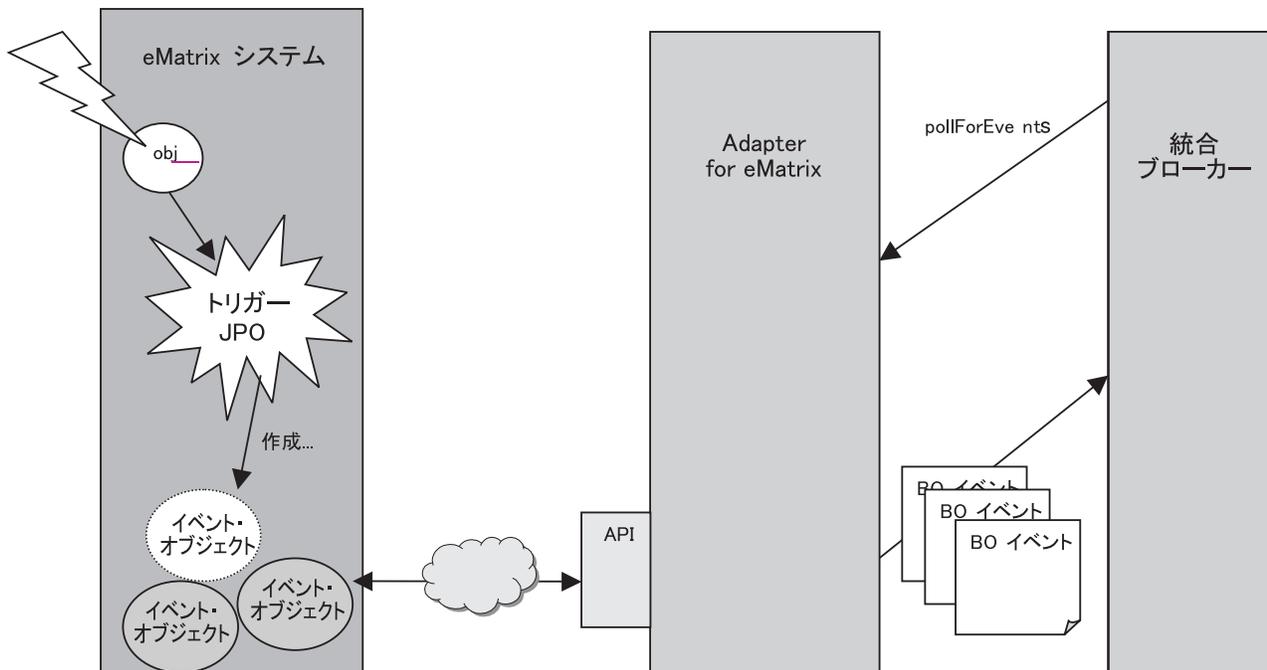


図 2. eMatrix アダプターにおけるイベント通知

イベント通知は以下のように行われます。

1. eMatrix システムでアクションが実行され、トリガーが起動します。
トリガーはさまざまな方法で起動できます。例えば、構成済みトリガーで eMatrix オブジェクトを変更するなどです。
2. トリガーが起動すると、eMatrix システムは、関連する Java Program Object (JPO) を実行します。これにより、eMatrix 内で新規イベント・オブジェクトが作成されます。
3. アダプターは、指定された間隔で eMatrix をポーリングし、JPO が作成した新規イベント・オブジェクトがないか調べます。
4. アダプターはイベント・オブジェクトを検索します。
5. アダプターは、次に、関連する eMatrix エンティティを検索し、それらをビジネス・オブジェクトに変換してから、統合ブローカーにパブリッシュします。アダプターは、eMatrix のビジネス・オブジェクトと関係の両方を検索できます。

アダプターの動作方法

Adapter for eMatrix は双方向に動作します。アダプターは、eMatrix システムで発生したイベントを処理し、統合ブローカーによって eMatrix システムに送信された要求を処理することができます。

イベント通知

アダプターは、ポーリング機構によってイベントを検出し、処理します。この機構を使用すると、ユーザーは、イベント優先順位を設定し、イベント検出頻度および

アダプターが処理できるイベント数を制御できます。このレベルの制御によって、大規模システム内でのアダプターの負荷のバランスを取り、重大イベントの経路を滑らかにできます。これらの値は、アダプターの構成可能プロパティーで設定されます (12 ページの『コネクターの構成』を参照)。

アダプターは、Java Program Object (JPO) および eMatrix ビジネス・オブジェクト定義を使用して、イベントに関する情報の保管および送信を行います。IBM では、選択して使用できる 2 つのデフォルト JPO を提供しています。この JPO は、以下のとおりです。

- **WBIEventLogger JPO**。イベントを記録するビジネス・オブジェクト・タイプを作成します。
- **InstallEventTables JPO**。イベントを保管するビジネス・オブジェクト・タイプを作成します。

この説明では、デフォルトの WBIEventLogger JPO の使用を想定しています。

eMatrix オブジェクトの構造の詳細については、21 ページの『第 4 章 ビジネス・オブジェクトの理解』を参照してください。

イベント通知においては、WBIEventLogger JPO が中核をなします。アクションが、特定のビジネス・オブジェクトまたは関係に関連するときは常に、トリガーを定義し、この JPO を呼び出すことができます。JPO は、オブジェクト ID および指定されたオプションを使用して、eMatrix 内で新規イベントのビジネス・オブジェクトまたは関係を作成します。

例えば、「Part Assembly」と呼ばれるオブジェクト・タイプに関連したイベントを追跡したい場合、eMatrix を構成し、そのようなアクションが発生したときに JPO を起動することができます。トリガーが起動すると、JPO は、PartAssembly ID およびイベント中に行われるアクション・タイプを記録します。

トリガー定義の詳細については、11 ページの『JPO の構成』を参照してください。

注: イベント通知は非同期処理です。アダプターは、同期イベント通知をサポートしていません。

イベント検索

アプリケーションをポーリングする場合、アダプターは、未処理イベントを eMatrix に照会します。アダプターは、構成プロパティーの EventStatus として **READY_FOR_POLL** の値を持っているすべてのイベントを検査します。アダプターは、WBIEventLogger JPO が作成したイベント・レコードを検査します。

すべてのイベントは同じボールドに存在します。アダプターは、始動時に EventVault プロパティーの値を読み取ります。

アダプターは、次に、関連した eMatrix オブジェクトを検索し、それを統合ブローカーに送信します。イベントが統合ブローカーにパブリッシュされたら、アダプターは、イベントを eMatrix に保存することもできます。

イベントに関する情報は、未処理イベントの場合、「<prefix>_Event」、処理済みイベントの場合、「<prefix>_Archived_Event」と呼ばれる eMatrix オブジェクトに保持されます。アダプターは、状況コードを使用してイベントの状態を追跡します。

予期せずにアダプターが終了すると、アダプターは、再始動時に「未確定」イベントを報告します。イベントが統合ブローカーに正常にパブリッシュされたかどうかをアダプターが判別できない場合、イベントには未確定というラベルが付けられます。アダプターは、通常のイベントを処理する前に、これらのイベントを処理する必要があります。アダプターが行うアクションの詳細については、41 ページの『エラー処理』を参照してください。

サンプル・シナリオ

Adapter for eMatrix には、アダプターがどのように要求処理およびイベント通知を行うかを示すサンプル・シナリオが含まれています。このシナリオのインストール、構成、および実行については、83 ページの『付録 C. サンプル・シナリオ』を参照してください。

第 2 章 アダプターのインストール

本章では、IBM WebSphere Business Integration Adapter for eMatrix のインストール方法を説明します。本章の内容は、次のとおりです。

- 『アダプター環境』
- 8 ページの『インストール作業の概要』
- 9 ページの『アダプターおよび関連ファイルのインストール』
- 9 ページの『インストール済みファイルの構造』

アダプター環境

アダプターをインストール、構成、使用する前に、環境要件を理解しておく必要があります。環境要件は、以下のセクションでリストされています。

- 『ブローカーの互換性』
- 8 ページの『アダプターのプラットフォーム』
- 8 ページの『アダプターの依存関係』
- 8 ページの『グローバリゼーション』

ブローカーの互換性

アダプターが使用するアダプター・フレームワークは、アダプターと通信する統合ブローカーのバージョンとの互換性を備えている必要があります。Adapter for eMatrix バージョン 2.1 は、以下のアダプター・フレームワークと統合ブローカーでサポートされています。

- **アダプター・フレームワーク:**
 - WebSphere Business Integration Adapter Framework、バージョン 2.1、2.2、2.3.x、および 2.4
- **統合ブローカー:**
 - WebSphere InterChange Server、バージョン 4.1.1、4.2、4.2.1、4.2.2
 - WebSphere MQ Integrator、バージョン 2.1.0
 - WebSphere MQ Integrator Broker、バージョン 2.1.0
 - WebSphere Business Integration Message Broker、バージョン 5.0
 - WebSphere Application Server Enterprise、バージョン 5.0.2
(Studio Application Developer Integration Edition、バージョン 5.0.1 と併用)

例外については、「リリース情報」を参照してください。

注: 統合ブローカーおよびその前提条件のインストールに関する説明については、以下のガイドを参照してください。

WebSphere InterChange Server (ICS) については、「*IBM WebSphere InterChange Server システム・インストール・ガイド (UNIX 版)*」または「*IBM WebSphere InterChange Server システム・インストール・ガイド (Windows 版)*」を参照してください。

WebSphere Message Broker については、「*WebSphere Message Brokers 使用ア*

「アダプター・インプリメンテーション・ガイド」を参照してください。
WebSphere Application Server については、「アダプター実装ガイド (WebSphere Application Server)」を参照してください。

アダプターのプラットフォーム

このアダプターは、次のソフトウェアでサポートされています。

オペレーティング・システム:

- AIX 4.3.3, AIX 5.1, AIX 5.2
- Solaris 7.0, Solaris 8.0
- HP UX 11.0, HP UX 11i
- Windows 2000

データベース:

- DB2 7.2
- Oracle 8.1.7.4, Oracle 9.2

サード・パーティー・ソフトウェア:

- eMatrix Server、バージョン 9.6.01、10.0.0.0
- eMatrix コラボレーション・サーバー、バージョン 9.6.01、10.0.0.0

アダプターの依存関係

アダプター・クライアントのライブラリー/API は、eMatrixServletRMI.jar、バージョン 10.0.0.0 にあります。

アダプターは、実行時に eMatrix ADK にアクセスする必要があります。ADK は、eMatrix コラボレーション・サーバーによって分散され、独立型バージョンになっています。

eMatrix コラボレーション・サーバーと同じマシン上にアダプターをインストールしている場合、ADK の独立型バージョンをダウンロードする必要はありません。しかし、アダプターとコラボレーション・サーバーが 2 つの異なるマシン上にある場合、アダプターと同じマシン上に独立型 ADK をインストールする必要があります。

グローバル化

このアダプターは DBCS (2 バイト文字セット) で使用可能であり、翻訳されています。

インストール作業の概要

Adapter for eMatrix をインストールするには、次のステップに従います。

1. 統合ブローカーがインストールされていることを確認します。
2. 必要なソフトウェアがすべてインストールされていることを確認します。
3. リモート eMatrix Server の名前およびロケーションを確認します。
4. Adapter for eMatrix および関連ファイルをインストールします。

5. eMatrix 用アダプターを構成します。

本書のインストールの説明では、ステップ 4 および 5 を対象としています。

アダプターおよび関連ファイルのインストール

WebSphere Business Integration アダプター製品のインストールについては、次のサイトで WebSphere Business Integration Adapters Infocenter にある「*WebSphere Business Integration Adapters* インストール・ガイド」を参照してください。

<http://www.ibm.com/websphere/integration/wbiadapters/infocenter>

インストール済みファイルの構造

表 1 に、アダプターで使用されるファイル構造、およびシステムにインストールされるファイルを示します。

注:

1. 本書では、ディレクトリー・パスとして円記号 (¥) を使用しています。UNIX システムの場合には、円記号をスラッシュ (/) に置き換えてください。
2. すべての製品パス名は、システム上の、製品のインストール先ディレクトリーを基準とした相対パス名です。

表 1. システムにインストールされたアダプターのファイル

%ProductDir% のサブディレクトリー	説明
¥bin¥Data¥App¥eMatrixConnectorTemplate	eMatrix Connector のテンプレート
connectors¥eMatrix	内容: <ul style="list-style-type: none">• CWeMatrix.jar• start_eMatrix.bat
connectors¥eMatrix¥dependencies	以下のデフォルトの JPO が入っています。 <ul style="list-style-type: none">• InstallEventTables.java• WBIEventLogger.java
connectors¥eMatrix¥samples	以下のサンプルのコネクター構成ファイルが入っています。 <ul style="list-style-type: none">• EmatrixConnector.cfg• PortConnector.cfg• repos• sound_card_update.bo

表 1. システムにインストールされたアダプターのファイル (続き)

%ProductDir% のサブディレクトリー	説明
connectors¥eMatrix¥samples¥repos	<p>以下のサンプルのビジネス・オブジェクトおよび関係オブジェクトが入っています。</p> <ul style="list-style-type: none"> • wbi_bundle.xsd • wbi_computer.xsd • wbi_computer_bundle_to_computer.xsd • wbi_computer_computer_to_pci_component.xsd • wbi_computer_to_pci_component.xsd • wbi_cpu.xsd • wbi_cw_event.xsd • wbi_expand_bo.xsd • wbi_input.xsd • wbi_network_card.xsd • wbi_sound_card.xsd • wbi_sound_card_sound_card_to_sound_chip.xsd • wbi_sound_chip.xsd • wbi_usb_2_0_card.xsd
connectors¥messages	<p>エラー・メッセージなどのメッセージがリストされた、eMatrixConnector.txt ファイルが入っています。</p>
ODA¥eMatrix	<p>内容:</p> <ul style="list-style-type: none"> • eMatrixODA.jar ODA コードを持った Object Discovery Agent • start_eMatrixODA.bat
ODA¥messages	<p>エラー・メッセージなどのメッセージがリストされた、eMatrixODAAgent.txt ファイルが入っています。</p>

第 3 章 アダプターの構成

この章では、IBM WebSphere Business Integration Adapter for eMatrix の構成および始動方法を説明します。本章の内容は、次のとおりです。

- 『アダプターの構成』
- 14 ページの『複数のコネクタ・インスタンスの作成』
- 16 ページの『コネクタの始動』
- 17 ページの『eMatrix アダプターの始動』
- 17 ページの『コネクタの停止』
- 18 ページの『ログ・ファイルとトレース・ファイルの使用』

アダプターの構成

インストールしたら、始動する前に、以下のようにして、アダプター・コンポーネントを構成する必要があります。

JPO の構成

JPO トリガーが動作するには、前もって特定の構成プロパティが設定されている必要があります。これらの構成プロパティを設定するには、各 JPO のソース・コードを編集する必要があります。

Adapter for eMatrix と一緒にインストールされる JPO は、WBIEventLogger および InstallEventTables です。それらをインストールしたら、eMatrix Business Modeler でそれぞれの JPO を探し、ソース・コードを編集する必要があります。

編集する必要のあるフィールドには、わかりやすいマークが付いています。それらの名前およびデフォルト値は、11 ページの表 2 にリストされています。

表 2. JPO 構成プロパティ

プロパティ名	説明	デフォルト値
wbiPrefix	eMatrix 内で、すべての WebSphere Business Integration eMatrix メタエンティティ (タイプ、属性、ポリシー) にそれら独自のネーム・スペースを提供するために使用される接頭部。WBIPrefix コネクタ構成プロパティに指定された値と一致する必要があります。	wbi_
vault	新規イベントが作成される eMatrix ポールの名前。EventVault コネクタ構成プロパティに指定された値に一致する必要があります。	WBI_Events
adapterUser	eMatrix システムで要求を行うときにアダプターが使用する ID。ApplicationUserName コネクタ構成プロパティに指定された値に一致する必要があります。 「ピンポン」を防止するのに重要です (下記参照)。	adapter

ピンポン作用の防止

トリガーが割り当てられている eMatrix システム内のオブジェクトをアダプターが変更すると、そのトリガーが新しいイベントを生成し、それがブローカーに戻されてパブリッシュされる場合があります。次の仕組みによって、このことが起こらないようにできます。

JPO は、呼び出されると、特にそれを呼び出したユーザーに関連付けられたユーザー ID を含む、コンテキスト・オブジェクトを受け取ります。このユーザー ID は、JPO プロパティの `adapterUser` で設定されます。WBIEventLogger JPO は、起動されると、eMatrix コンテキスト・オブジェクトから受け取ったユーザー ID を検査します。

ユーザーが Adapter for eMatrix の場合、JPO の `adapterUser` プロパティの値は、コネクターの `ApplicationUserName` プロパティの値に一致します。この場合、JPO はそのイベントを無視し、イベントはログに記録されません。ユーザー ID が一致しない場合、JPO は新しいイベントを作成します。

注: JPO の `adapterUser` プロパティとアダプターの `ApplicationUserName` プロパティを同じ値に設定することが重要です。そうしない場合は、ピンポン作用が発生します。

コネクターの構成

アダプターのコネクタ・コンポーネントには、2 種類の構成プロパティがあります。1 つは標準の構成プロパティで、ほとんどのアダプターに適用されます。もう 1 つはアプリケーション固有の構成プロパティで、ご使用のアダプターにのみ適用されます。コネクターを実行する前に、必ずこれらのプロパティの値を設定してください。

標準構成プロパティ

標準コネクタ構成プロパティを構成するには、Connector Configurator ツールを使用します。詳細は、65 ページの『付録 B. Connector Configurator』に記載されています。このツールは、コネクターを構成するためのグラフィカル・ユーザー・インターフェースを提供します。「**標準構成プロパティ (Standard Config Properties)**」タブをクリックして、構成プロパティを追加または変更します。

コネクターの構成プロパティへの値の指定が終了すると、Connector Configurator は値をアダプター・リポジトリに保管する (ICS の場合) か、構成ファイルを生成してアダプターのローカル・リポジトリに配置 (WebSphere Message Brokers または WAS の場合) します。

コネクターは、始動時に構成値を取得します。実行時のセッション中に、1 つ以上のコネクタ・プロパティの値を変更できます。

- `AgentTraceLevel` など、一部の構成プロパティへの変更は動的であり、即時に有効になります。
- それ以外のコネクタ・プロパティへの変更は静的であり、変更後にコンポーネントまたはシステムの再始動が必要です。

プロパティが動的と静的のいずれであるかを判別するには、Connector Configurator の「更新メソッド」欄を参照してください。

アプリケーション固有の構成プロパティ

アプリケーション固有のコネクタ構成プロパティは、アプリケーションに関連し、実行時にコネクタによって必要とされる情報を提供します。また、コネクタ内の静的情報やロジックを、再コーディングや再ビルドを行わずに変更する方法も提供します。

これらのプロパティを構成するには、Connector Configurator を使用します。構成プロパティを追加または変更するには、「アプリケーション構成プロパティ」タブをクリックします。詳細については、65 ページの『付録 B. Connector Configurator』を参照してください。

表 3 には、コネクタのアプリケーション固有の構成プロパティが、説明および指定可能な値とともにリストされています。

表 3. eMatrix 用のアプリケーション固有の構成プロパティ

プロパティ	説明	指定可能な値	デフォルト値	必須
ApplicationUserName	eMatrix システムに接続するときに使用される名前。	<任意> adapterUser JPO 構成プロパティに設定された値に一致する必要がある。	WBI	はい
ApplicationPassword	eMatrix システムに接続するときに使用されるパスワード。	<任意>	WBI	はい
EventVault	アダプターがイベントを検索するポールド。	<任意> ポールド JPO 構成プロパティに設定された値に一致する必要がある。	WBIEventVault	はい
inDoubtEvents	未確定イベントのリカバリー方針。値が指定されないと、アダプターは開始時に失敗する。	以下のいずれか: <ul style="list-style-type: none"> • FailOnStartup • Reprocess • LogError • Ignore 	Reprocess	はい
ArchiveProcessed	True の場合、アダプターはすべてのイベントを保存する。False または未定義の場合、アダプターは、イベントが処理された後に、それらを eMatrix から削除する。	true または false	false	
DefaultVault	アダプターが新規オブジェクトを作成し、既存オブジェクトを操作するデフォルト・ポールド。	<任意>	DefaultWBIVault	
DefaultPolicy	eMatrix に新規オブジェクトを作成するときに、アダプターが使用するデフォルト・ポリシー。	<任意>	DefaultWBIPolicy	

表 3. eMatrix 用のアプリケーション固有の構成プロパティ (続き)

プロパティ	説明	指定可能な値	デフォルト値	必須
KeepRelations	true の場合、アダプターは、ビジネス・オブジェクトの更新時に関係を維持する。false の場合、アダプターは、更新要求で未定義のすべての関係を破棄する。	true または false	false	
UseDefaults	UseDefaults が true に設定される場合、アダプターは、ビジネス・オブジェクト定義でデフォルト設定を使用する。	true または false	true	
PollQuantity	各ポーリング呼び出しで、アダプターが処理できる最大イベント数。	<任意の正の整数>	1	
WBIPrefix	eMatrix システムで、イベント通知に関連付けられたすべてのビジネス・オブジェクトおよび属性の接頭部。 注: 既存のビジネス・オブジェクトまたは属性は、この値で開始することはできない。	<任意> JPO wbiPrefix 構成プロパティに指定された値に一致する必要がある。	wbi_	
EmatrixServer	リモート・ビジネス・オブジェクト・サーバーの名前。	<任意>	:bos	
Hostname	コラボレーション・サーバーの URL。	<任意>	localhost:1099	

複数のコネクター・インスタンスの作成

コネクターの複数のインスタンスを作成する作業は、いろいろな意味で、カスタム・コネクターの作成と同じです。以下に示すステップを実行することによって、コネクターの複数のインスタンスを作成して実行するように、ご使用のシステムを設定することができます。次のようにする必要があります。

- コネクター・インスタンス用に新規ディレクトリーを作成します。
- 必要なビジネス・オブジェクト定義が設定されていることを確認します。
- 新規コネクター定義ファイルを作成します。
- 新規始動スクリプトを作成します。

新規ディレクトリーの作成

それぞれのコネクター・インスタンスごとにコネクター・ディレクトリーを作成する必要があります。このコネクター・ディレクトリーには、次の名前を付けなければなりません。

ProductDir%connectors%connectorInstance

ここで connectorInstance は、コネクター・インスタンスを一意的に示します。

コネクタに、コネクタ固有のメタオブジェクトがある場合、コネクタ・インスタンス用のメタオブジェクトを作成する必要があります。メタオブジェクトをファイルとして保管する場合は、次のディレクトリを作成して、ファイルをそこに格納します。

```
ProductDir¥repository¥connectorInstance
```

ビジネス・オブジェクト定義の作成

各コネクタ・インスタンスのビジネス・オブジェクト定義がプロジェクト内にまだ存在しない場合は、それらを作成する必要があります。

1. 初期コネクタに関連付けられているビジネス・オブジェクト定義を変更する必要がある場合は、適切なファイルをコピーし、**Business Object Designer** を使用してそれらのファイルをインポートします。初期コネクタの任意のファイルをコピーできます。変更を加えた場合は、名前を変更してください。
2. 初期コネクタのファイルは、次のディレクトリに入っていない限りません。

```
ProductDir¥repository¥initialConnectorInstance
```

作成した追加ファイルは、`ProductDir¥repository` の適切な `connectorInstance` サブディレクトリ内に存在している必要があります。

コネクタ定義の作成

Connector Configurator 内で、コネクタ・インスタンスの構成ファイル (コネクタ定義) を作成します。これを行うには、以下のステップを実行します。

1. 初期コネクタの構成ファイル (コネクタ定義) をコピーし、名前変更します。
2. 各コネクタ・インスタンスが、サポートされるビジネス・オブジェクト (および関連メタオブジェクト) を正しくリストしていることを確認します。
3. 必要に応じて、コネクタ・プロパティをカスタマイズします。

始動スクリプトの作成

始動スクリプトは以下のように作成します。

1. 初期コネクタの始動スクリプトをコピーし、コネクタ・ディレクトリの名前を含む名前を付けます。

```
dirname
```

2. この始動スクリプトを、14 ページの『新規ディレクトリの作成』で作成したコネクタ・ディレクトリに格納します。
3. 始動スクリプトのショートカットを作成します (Windows のみ)。
4. 初期コネクタのショートカット・テキストをコピーし、新規コネクタ・インスタンスの名前に一致するように (コマンド行で) 初期コネクタの名前を変更します。

これで、ご使用の統合サーバー上でコネクタの両方のインスタンスを同時に実行することができます。

カスタム・コネクタ作成の詳細については、「コネクタ開発ガイド (C++ 用)」または「コネクタ開発ガイド (Java 用)」を参照してください。

コネクタの始動

コネクタは、**コネクタ始動スクリプト**を使用して明示的に始動する必要があります。始動スクリプトは、次に示すようなコネクタのランタイム・ディレクトリに存在していなければなりません。

```
ProductDir¥connectors¥connName
```

ここで、*connName* はコネクタを示します。始動スクリプトの名前は、表 4 に示すように、オペレーティング・システム・プラットフォームによって異なります。

表 4. コネクタの始動スクリプト

オペレーティング・システム	始動スクリプト
UNIX ベースのシステム	connector_manager_connName
Windows	start_connName.bat

コネクタ始動スクリプトは、以下に示すいずれかの方法で起動することができます。

- Windows システムで「スタート」メニューから。
「プログラム」>「IBM WebSphere Business Integration Adapters」>「アダプター」>「コネクタ」を選択します。デフォルトでは、プログラム名は「IBM WebSphere Business Integration Adapters」となっています。ただし、これはカスタマイズすることができます。あるいは、ご使用のコネクタへのデスクトップ・ショートカットを作成することもできます。
- コマンド行から。
 - Windows システム:
start_connName connName brokerName [-cconfigFile]
 - UNIX ベースのシステム:
connector_manager_connName -start

ここで、*connName* はコネクタの名前であり、*brokerName* は以下のようにご使用の統合ブローカーを表します。

- WebSphere InterChange Server の場合は、*brokerName* に ICS インスタンスの名前を指定します。
- WebSphere Message Brokers (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、または WebSphere Business Integration Message Broker) または WebSphere Application Server の場合は、*brokerName* にブローカーを示すストリングを指定します。

注: Windows システム上の WebSphere Message Broker または WebSphere Application Server の場合は、*-c* オプションに続いてコネクタ構成ファイルの名前を指定しなければなりません。ICS の場合は、*-c* はオプションです。

- Adapter Monitor から (WebSphere Business Integration Adapters 製品のみ)。
Adapter Monitor は System Manager 始動時に起動されます。
このツールを使用して、コネクタのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- System Monitor から (WebSphere InterChange Server 製品のみ)。このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。
- Windows システムでは、Windows サービスとして始動するようにコネクターを構成することができます。この場合、Windows システムがブートしたとき (自動サービスの場合)、または Windows サービス・ウィンドウを通じてサービスを始動したとき (手動サービスの場合) に、コネクターが始動します。

コマンド行の始動オプションなどのコネクターの始動方法の詳細については、以下の資料のいずれかを参照してください。

- WebSphere InterChange Server については、「システム管理ガイド」を参照してください。
- WebSphere Message Brokers については、「WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド」を参照してください。
- WebSphere Application Server については、「アダプター実装ガイド (WebSphere Application Server)」を参照してください。

eMatrix アダプターの始動

start_eMatrix.bat スクリプトまたは start_eMatrix.sh スクリプトが実行されると、コネクターが始動します。これらの始動ファイルには、アダプターの実行前に設定しておく必要のある 2 つの変数が含まれています。変数は以下のとおりです。

- EMADK: eMatrix ADK .jar ファイルのロケーションに設定する必要があります。
- EM_LIB: eMatrix 共有ライブラリー・ファイルが格納されたディレクトリーのパスに設定する必要があります。

例えば、次のようになります。

```
set EMADK=c:%ematrix96%java%lib%eMatrixServletRMI.jar
set EM_LIB=c:%ematrix96%bin%winnt
```

注: 別個のマシン上で eMatrix コラボレーション・サーバーを稼働し、独立型 eMatrix ADK .jar ファイルをそのマシンにインストール済みの場合、EM_LIB を設定する必要はありません。

コネクターの停止

コネクターを停止する方法は、以下に示すように、コネクターが始動された方法によって異なります。

- コマンド行からコネクターを始動した場合は、コネクター始動スクリプトを用いて、以下の操作を実行します。
 - Windows システムでは、始動スクリプトを起動すると、そのコネクター用の別個の「コンソール」ウィンドウが作成されます。このウィンドウで、「Q」と入力して Enter キーを押すと、コネクターが停止します。
 - UNIX ベースのシステムでは、コネクターはバックグラウンドで実行されるため、別ウィンドウはありません。代わりに、次のコマンドを実行してコネクターを停止します。

```
connector_manager_connName -stop
```

ここで、*connName* はコネクタの名前です。

- Adapter Monitor から (WebSphere Business Integration Adapters 製品のみ)。Adapter Monitor は System Manager 始動時に起動されます。
このツールを使用して、コネクタのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。
- System Monitor から (WebSphere InterChange Server 製品のみ)
このツールを使用して、コネクタのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。
- Windows システムでは、Windows サービスとして始動するようにコネクタを構成することができます。この場合、Windows システムのシャットダウン時に、コネクタは停止します。

ログ・ファイルとトレース・ファイルの使用

アダプター・コンポーネントは、いくつかのレベルのメッセージのロギングとトレースを提供します。

コネクタはエラー・メッセージ、通知メッセージ、およびトレース・メッセージを記録するために、アダプター・フレームワークを使用します。エラー・メッセージと通知メッセージは外部ログ・ファイルに記録され、トレース・メッセージとトレース・レベル (0 から 5) はトレース・ファイルに記録されます。デフォルトのメッセージ・ファイル名は `eMatrixConnector.txt` です。

表 5 には、各トレース・レベルで記録されるメッセージのタイプが記載されています。

表 5. トレース・ファイルのトレース・レベル

トレース・レベル	説明
0	<ul style="list-style-type: none">• eMatrix ODA からのエラーと致命的エラーを記録します。• システム管理者の対応が必要な警告を記録します。• アプリケーションからの重要なメッセージを記録します。
1	メソッドのすべての開始メッセージおよび終了メッセージをトレースします。
2	ODA のプロパティとそれらの値をトレースします。
3	すべてのビジネス・オブジェクトの名前をトレースします。
4	ビジネス・オブジェクトのプロパティと、受信した値をトレースします。
5	<ul style="list-style-type: none">• すべての ODA プロパティの ODA 初期化値を示します。• ビジネス・オブジェクト定義のダンプをトレースします。

ログ・ファイルとトレース・ファイルの名前、およびトレース・レベルを Connector Configurator で構成してください。詳細については、65 ページの『付録 B. Connector Configurator』を参照してください。

ODA にはロギング機能はなく、エラー・メッセージは直接ユーザー・インターフェースに送られます。トレース・ファイルとトレース・レベルは、**Business Object Designer** で構成されます。このプロセスについては、33 ページの『エージェントの構成』を参照してください。ODA トレース・レベルはコネクター・トレース・レベルと同一であり、上記の 18 ページの表 5 に記載されています。

エラー処理とトラブルシューティングの詳細については、41 ページの『第 6 章 トラブルシューティングおよびエラー処理』を参照してください。

第 4 章 ビジネス・オブジェクトの理解

この章では、ビジネス・オブジェクトの構造、アダプターがビジネス・オブジェクトを処理する仕組み、およびビジネス・オブジェクトに関するアダプターの前提事項について説明します。

本章の内容は、次のとおりです。

- 21 ページの『メタデータの定義』
- 22 ページの『ビジネス・オブジェクト構造の概要』
- 22 ページの『eMatrix のビジネス・オブジェクト』
- 29 ページの『ビジネス・オブジェクトの生成』

メタデータの定義

eMatrix のアダプターはメタデータ主導型です。WebSphere Business Integration システム内では、メタデータはデータ構造を記述するアプリケーション固有情報であると定義されます。メタデータは、コネクターがビジネス・オブジェクト作成のために実行時に使用する、ビジネス・オブジェクト定義と関係定義を構成するために使用します。

メタデータ主導型アダプターは、サポートする各ビジネス・オブジェクトを処理する際に、ビジネス・オブジェクト定義にエンコードされたメタデータに従って処理を行います。これによりアダプターは、コードを変更することなく、新規または変更されたビジネス・オブジェクト定義を処理できます。ビジネス・オブジェクト定義を変更するには、Business Object Designer で編集します。

アプリケーション固有のメタデータは、ビジネス・オブジェクト・タイプ、関係、それらのプロパティ、または属性を表すことができます。各ビジネス・オブジェクトの実際のデータ値は、実行時に送信されます。データ値は、アダプターとブローカーの間で渡されるビジネス・オブジェクトにカプセル化されます。

アダプターには、サポートするビジネス・オブジェクトの構造、親と子のビジネス・オブジェクト間の関係、およびデータのフォーマットに関する前提事項があります。したがって、ビジネス・オブジェクトの構造が eMatrix 内の対応するオブジェクトに定義されている構造と正確に一致することは重要であり、一致しない場合はアダプターがビジネス・オブジェクトを正しく処理できません。

注: ビジネス・オブジェクト構造を変更する必要がある場合、ODA を使用して、eMatrix 内の対応するオブジェクトに変更を加えることをお勧めします。詳細については、31 ページの『第 5 章 ビジネス・オブジェクト定義の生成』を参照してください。

ビジネス・オブジェクト定義の変更の詳細については、「*WebSphere Business Integration Adapters* ビジネス・オブジェクト開発ガイド」を参照してください。

ビジネス・オブジェクト構造の概要

WebSphere Business Integration システムでは、ビジネス・オブジェクト定義の構成は以下のとおりです。

- タイプ名
- サポートされる動詞
- 属性
- アプリケーション固有情報 (ASI)

アプリケーション固有のビジネス・オブジェクト定義は、特殊なタイプのビジネス・オブジェクト定義です。これは、個々のアプリケーションのデータ構造および属性プロパティを反映します。

一部の属性は、データを格納する代わりに、子ビジネス・オブジェクト、または子ビジネス・オブジェクト配列を指しています。例えば属性は、関係を表す WebSphere Business Integration 子ビジネス・オブジェクトを参照することができます。

アダプターのビジネス・オブジェクトにはフラットなものと同層構造のものがあります。フラット・ビジネス・オブジェクトには、単純属性、つまり、単一の値 (ストリングなど) を表す属性だけを格納できます。これは、子ビジネス・オブジェクトを指しません。階層ビジネス・オブジェクトには、単純属性と、子ビジネス・オブジェクトまたは子ビジネス・オブジェクト配列の両方が格納されます。

階層ビジネス・オブジェクトには、単一カーディナリティーまたは複数カーディナリティーを持つ子ビジネス・オブジェクトを含めることができます。子オブジェクトは、親の属性の中に格納されます。複数カーディナリティーを持つ子ビジネス・オブジェクトを含む属性は、ビジネス・オブジェクトの配列です。単一カーディナリティーを持つ子ビジネス・オブジェクトを表す属性は、このようなオブジェクトを 1 つのみ含むことができます。

eMatrix のビジネス・オブジェクト

WebSphere Business Integration システム内の有効なアプリケーション固有ビジネス・オブジェクトによって、eMatrix システムの必須のコンポーネントがモデル化されます。以下の eMatrix コンポーネントは、WebSphere Business Integration ビジネス・オブジェクトとしてモデル化することができます。

- **ビジネス・オブジェクトのタイプ:**
情報を編成するために使用されます。ビジネス・オブジェクトのインスタンスが情報を保管します。
- **関係:** 2 つのビジネス・オブジェクト間の関係を定義します。
- **コマンド:**
カスタム操作の実行または eMatrix プログラムの起動のために使用されます。

WBIA ビジネス・オブジェクト定義は、ビジネス・オブジェクト・レベルで定義されたアプリケーション固有情報 (ASI) の `ematrix_class` の値で表されているコンポーネントを指定します。上述の eMatrix コンポーネントに対応する 3 つの有効な `ematrix_class` の値については、次のセクションで説明します。

ビジネス・オブジェクトのタイプ

WBIA システムでは、eMatrix ビジネス・オブジェクト・タイプは、値が `business_object` の `ematrix_class` を指定するビジネス・オブジェクト・レベルの ASI を含むビジネス・オブジェクト定義によって示されます。また、ASI には、この特定の WBIA ビジネス・オブジェクトがモデルとなる eMatrix ビジネス・オブジェクト・タイプの名前を指定する `object_type` パラメーターが含まれていなければなりません。例えば、次のようになります。

```
ematrix_class=business_object; object_type=<name of eMatrix business object type>
```

eMatrix ビジネス・オブジェクト・タイプには、2 セットのフィールドが含まれています (従って WBIA システム内のそれらのビジネス・オブジェクト定義にも含まれます)。フィールド・セットは以下のとおりです。

- プロパティ。固定されており、すべての eMatrix ビジネス・オブジェクト・タイプが共有します。ビジネス・オブジェクト所有者、ポリシー、および改訂などがあります。
- 属性。ユーザーが定義します (WBIA システム属性と混同しないでください)。

ビジネス・オブジェクトを通じて提供される eMatrix プロパティが、表 6 にリストされています。

注: 「string」タイプの属性を使用して日付を表します。サポートされる日付形式については、eMatrix の資料を参照してください。

表 6. eMatrix ビジネス・オブジェクト・プロパティ

プロパティ	説明	予期される属性タイプ
object_id	eMatrix オブジェクトの一意の ID	string
Name	eMatrix オブジェクトの名前	string
Revision	eMatrix オブジェクト改訂番号	string
Policy	eMatrix オブジェクトの制御ポリシー	string
Owner	eMatrix オブジェクトの所有者	string
State	eMatrix オブジェクトの現在の状態	string
Vault	このオブジェクトを格納する eMatrix ボールト	string
to_relationship	このオブジェクトがサポートする 1 つ以上の「関係先」 eMatrix 関係	eMatrix 関係をモデル化する子オブジェクト Cardinality = 1...n
from_relationship	このオブジェクトがサポートする 1 つ以上の「関係元」 eMatrix 関係	eMatrix 関係をモデル化する子オブジェクト Cardinality = 1...n

eMatrix では、オブジェクト ID を通じて、またはタイプ、名前、および改訂の組み合わせを通じて、ユーザーがオブジェクトを識別します。そのため、すべての WBIA ビジネス・オブジェクトは、以下のいずれかを含んでいなければなりません。

- プロパティ `object_id` にマップされる属性
- プロパティ `name` および `revision` のそれぞれにマップされる 2 つの属性

これらは、ビジネス・オブジェクト定義内のキー値となります。例については、24 ページの『ビジネス・オブジェクト定義の例』を参照してください。

特定の WBIA ビジネス・オブジェクト属性が eMatrix 属性にマップされるように指定するためには、以下を実行します。

1. 属性を、ストリングまたは整数などの単純タイプとして定義します。
2. ビジネス・オブジェクト定義属性レベル ASI に、

```
attr=<eMatrix attribute name>
```

と記述します。

ビジネス・オブジェクト定義の例

24 ページの表 7 に、PartNumber および Size の 2 つのユーザー定義 eMatrix 属性を含み、ASI プロパティ object_id にマップされるビジネス・オブジェクト定義を示します。

表 7. ビジネス・オブジェクト定義 #1

属性名	タイプ	デフォルト値	アプリケーション固有情報	IsKey?
ObjectId	ストリング		prop=object_id	はい
State	ストリング		prop=state	いいえ
PartNumber	ストリング	ASX31	attr=PartNumber	いいえ
Size	整数	15	attr=Size	いいえ

24 ページの表 8 に示すのは同じビジネス・オブジェクト定義ですが、eMatrix プロパティの name および revision にマップされています。

表 8. ビジネス・オブジェクト定義 #2

属性名	タイプ	デフォルト値	アプリケーション固有情報	IsKey?
Name	ストリング	AllSeasonX31	prop=name	あり
Revision	ストリング	1	prop=revision	あり
State	ストリング		prop=state	なし
PartNumber	ストリング	ASX31	attr=PartNumber	なし
Size	整数	15	attr=Size	なし

関係

WBIA システムでは、eMatrix 関係は、値が relationship の ematrix_class を指定するビジネス・オブジェクト・レベルの ASI を含むビジネス・オブジェクト定義によって示されます。また、ASI には、この特定の WBIA ビジネス・オブジェクトがモデルとなる eMatrix 関係の名前を指定する object_type パラメーターが含まれていなければなりません。例えば、次のようになります。

```
ematrix_class=relationship; object_type=<name of eMatrix relationship type>
```

前述のビジネス・オブジェクト定義と同様に、eMatrix 関係を反映するビジネス・オブジェクト定義には、eMatrix プロパティ (固定) および属性 (ユーザー定義) の組み合わせを含めることができます。

25 ページの表 9 に、関係プロパティをリストします。

表9. eMatrix 関係オブジェクト・プロパティ

プロパティ	説明	予期される属性タイプ
relationship_id	eMatrix 関係の一意の ID	ストリング
to_object	この関係の関係先の eMatrix オブジェクト	eMatrix ビジネス・オブジェクトをモデル化する子オブジェクト Cardinality = 1
from_object	この関係の関係元の eMatrix オブジェクト	eMatrix ビジネス・オブジェクトをモデル化する子オブジェクト Cardinality = 1

すべての関係ビジネス・オブジェクト定義には、以下のデータが含まれていなければなりません。

- プロパティ relationship_id にマップする属性
- to_object または from_object 属性のインスタンス (下記の注を参照)
- ビジネス・オブジェクトを to_object または from_object 子ビジネス・オブジェクト・コンテナとして分類するアプリケーション固有情報を持つ属性

注: 単一の関係が多くのオブジェクト・タイプをサポートできるため、関係オブジェクトには、to_object または from_object 属性のインスタンスが多く含まれています。しかし、関係が、親ビジネス・オブジェクトの to_relationship として定義される場合、関係オブジェクトは to_object の ID のみを持つことができます。同様に、親ビジネス・オブジェクトで from_relationship が定義される場合には、関係オブジェクトは from_object の ID のみを持つことができます。

関係定義は、以下のいずれかの書式で記述します。

```
--Business_Object_A
| -- to_relationship = Relationship_A_to_B
| -- to_object = Business_Object_B
```

または

```
--Business_Object_A
| -- from_relationship = Relationship_A_to_B
| -- from_object = Business_Object_B
```

例については、26 ページの『関係の例』を参照してください。

関係の定義

アダプターは、ビジネス・オブジェクト定義で明示的にその関係を定義した場合のみ、異なるオブジェクト間の関係を認識できます。ビジネス・オブジェクト同士が関連する場合、アダプターは、ビジネス・オブジェクトを接続するために関係オブジェクトを必要とします。ビジネス・オブジェクト同士が関係オブジェクトによって接続されていない場合、関連しないものと見なされます。

例えば、次のようなビジネス・オブジェクトを定義します。

```
--Business_Object_A
| -- to_relationship = Relationship_A_to_B
| -- to_object = Business_Object_B
```

そして親ビジネス・オブジェクト (オブジェクト A) で Retrieve を実行した場合、アダプターは A を取り込み、また、A から B へ の関係を通じて A に関係付けられているタイプ B のオブジェクトのみを取り込みます。

ただし、次のビジネス・オブジェクトでは異なる結果になります。

```
--Business_Object_D
  | -- Business_Object_E
```

このとき、親ビジネス・オブジェクト (オブジェクト D) で Retrieve を実行すると、アダプターは D を取り込み、また、D に関連するかどうかにかかわらずタイプ E のオブジェクトを取り込みます。この場合、ビジネス・オブジェクト E は、有効な検索基準を含む属性を持っていないければなりません。持たない場合はエラーとなります。

ビジネス・シナリオおよび関係に応じて、関連するビジネス・オブジェクトと関連のないビジネス・オブジェクトを定義中で結合することができます。例えば、以下のようなビジネス・オブジェクトを定義することができます。

```
--Business_Object_A
  | -- to_relationship = Relationship_A_to_B
    | -- to_object = Business_Object_B
    | -- Business_Object_D
```

ここで、ビジネス・オブジェクト A および B は互いに関連していますが、ビジネス・オブジェクト D はそのどちらにも関連していません。

関係の例

以下の 2 つの例では、それぞれのビジネス・オブジェクト定義内で関係の定義は異なっていますが、同じ方法で関連付けられています。

```
--Car
  | -- to_relationship = CarToTireRelationship
    | -- to_object = Tire

--Tire
  | -- from_relationship = CarToTireRelationship
    | -- from_object = Car
```

コマンド・オブジェクト

WBIA システムでは、eMatrix コマンドは、値が command の ematrix_class を指定するビジネス・オブジェクト・レベルの ASI を含むビジネス・オブジェクト定義によって示されます。例えば、次のようになります。

```
ematrix_class=command
```

コマンド・ビジネス・オブジェクト定義は、以下のような構造になっています。

```
-- Command_object
  | -- Input_Object
  | -- Output_Object
```

入力オブジェクトと出力オブジェクトは、トップレベル・コマンドの子オブジェクトです。入力オブジェクト定義 (コマンド・ストリングとそのパラメーターをカプセル化したもの) は必須です。ただし、出力オブジェクト定義 (コマンドが戻した出力をカプセル化したもの) は必須ではありません。

入力子オブジェクトには、以下の属性が含まれていないければなりません。

- type=input

これにより、アダプターに対して子オブジェクトが入力として定義されます。

- command=<command string to run in eMatrix>

属性名をパーセント記号で囲むと (% <name>%)、コマンド・ストリングを使用して子オブジェクト内の属性を参照することができます。下記の例を参照してください。

コマンドが結果を戻す場合、オブジェクトでは、データを受け取るために出力属性を定義する必要があります。出力属性は、ストリングまたは子オブジェクトです。出力属性として子オブジェクトを使用する場合、アダプターが着信データをマップできるようにするため、データ・ハンドラーを指定する必要があります。以下の属性のペアのいずれかを使用して、データ・ハンドラーを指定することができます。

- dh_mo=<data handler meta-object name>;dh_mimetype=<mimetype>
または
- dh_classname=<data handler classname>

例については、『コマンドの例』を参照してください。

コマンド・オブジェクト階層は、eMatrix ビジネス・オブジェクトと eMatrix 関係の集合を含む階層を内包することも、そのような階層に内包されることもできません。コマンド・オブジェクト階層は独立していなければなりません。

注: コマンド・ビジネス・オブジェクトでサポートされる動詞は Create のみです。

コマンドの例

コマンド・オブジェクト定義の例を以下に示します。

```
--Command_Object
| -- Input_Object (ASI = 'type=input;command=approve %ObjectId% signature
%signature% comment %Comment%'
| -- ObjectId = 1000
| -- Signature = Loan Approved
| -- Comment = Approved up to $20,000
```

アダプターが置換を行った後に eMatrix で処理される実際のコマンドは、次のようになります。

```
approve 1000 signature Loan Approved comment Approve up to $20,000
```

XML を戻すプログラムを実行するコマンド・オブジェクトの例を以下に示します。

```
--Command_Object
| -- Input_Object (ASI = 'type=input;command=exec program BuildReport
%ObjectId%'
| -- ObjectId = 1000
| -- Output_Object (ASI =
'type=output;dh_mo=Default_DataHandler_M0:dh_mimetype=text/xml'
| -- AttrA
| -- AttrB
```

ビジネス・オブジェクト動詞

すべてのビジネス・オブジェクトは動詞を含んでおり、この動詞は、アプリケーション固有のビジネス・オブジェクト内のデータを eMatrix アダプターが処理する方法を記述しています。

アダプターは、一意のオブジェクト ID を用いるか、またはタイプ、名前、および改訂の組み合わせを用いて、ビジネス・オブジェクトを識別します。これらはすべてユーザー定義のプロパティです。詳細については、23 ページの『ビジネス・オブジェクトのタイプ』を参照してください。

アダプターは、eMatrix オブジェクトのクラスが異なる場合、同じ動詞について異なる解釈をします。アダプターは、ビジネス・オブジェクト定義内の `ematrix_class` の値をチェックしてから、ビジネス・オブジェクトを処理します。

28 ページの表 10 に、Adapter for eMatrix で使用可能なビジネス・オブジェクト動詞をリストし、それぞれの動詞でアダプターが実行するアクションを説明します。

表 10. ビジネス・オブジェクト動詞とアクション

動詞	eMatrix クラス	アクション
Create	ビジネス・オブジェクト	eMatrix 内で、1 つ以上の新規ビジネス・オブジェクトを関係と共に作成します。要求で指定された名前、タイプ、および改訂を使用します。ポルトとポリシーが指定されていればそれらを使用し、指定されていなければデフォルトを使用します。指定された子オブジェクトと関係を作成します。
	関係	すべての子オブジェクトは親オブジェクトと同じ動詞を持つ必要があるため、関係をそれぞれ独立して作成することはできません。すでに関係が存在しているオブジェクト間に新規の関係を作成するには、Update を使用します。
	コマンド	eMatrix システムでコマンドを実行し、要求オブジェクトに結果をすべて戻します。
Update	ビジネス・オブジェクトおよび関係	1 つ以上の指定されたビジネス・オブジェクトおよびそれらの関係を更新します。例については後述します。
Delete	ビジネス・オブジェクト	指定されたビジネス・オブジェクトを削除し、それに結び付けられた関係を自動的に削除します。すべての削除は物理的削除です。
	関係	関係 ID が示す関係を削除します。その関係に結び付くビジネス・オブジェクトの接続を解除しますが、ビジネス・オブジェクトは削除しません。
Retrieve	ビジネス・オブジェクトおよび関係	関係と、WBIA ビジネス・オブジェクト内の属性に対応する eMatrix ビジネス・オブジェクトを含む、ビジネス・オブジェクト全体を、eMatrix から取り込みます。統合ブローカーに戻されるビジネス・オブジェクトは、ビジネス・オブジェクト定義内の構造を正確に反映していなければなりません。
RetrieveBy Content	ビジネス・オブジェクトおよび関係	キー情報または非キー情報、あるいはその両方を使用して、eMatrix からビジネス・オブジェクト全体を取り込みます。一致が複数検出された場合は、最初の一致を返し、「複数のヒット」を知らせるメッセージを送信します。
Exists	ビジネス・オブジェクトおよび関係	トップレベル・ビジネス・オブジェクトが eMatrix 内に存在するかどうかを確認します。

動詞は、アプリケーション固有のビジネス・オブジェクト内部に含まれています。ビジネス・オブジェクトを作成するブローカーによって値が設定されます。アダプターは、動詞およびビジネス・オブジェクトの内容に応じて、eMatrix アプリケーション上で異なる操作を実行します。

アダプターは、ビジネス・オブジェクトを処理し、eMatrix システム内で必要なアクションを起動すると、状況コードを戻します。また、アプリケーションの状態に関する情報を含む新規ビジネス・オブジェクトを統合ブローカーに戻すこともあります。

属性プロパティ

WBIA ビジネス・オブジェクト定義には、それらのビジネス・オブジェクト属性で設定可能な、さまざまなプロパティがあります。以下の表に属性とその値を示します。

表 11 には単純属性をリストしています。

表 11. 単純属性

属性	プロパティ
Name	ビジネス・オブジェクトのフィールド名を指定します。
Type	ビジネス・オブジェクトのフィールド・タイプを指定します。
MaxLength	デフォルトは 255 です。
IsKey	各ビジネス・オブジェクトは、少なくとも 1 つのキー属性を持つ必要があります。キー属性は、属性のキー・プロパティを true に設定することで指定します。
IsForeignKey	使用されません。
Is Required	使用されません。
AppSpecInfo	attr=<attribute name> などの、アプリケーション固有情報 (ASI) を格納します。
DefaultValue	必要な属性が設定されていない場合に、コネクタがインバウンド・ビジネス・オブジェクト内の単純属性用に使用するデフォルト値を指定します。

アプリケーション固有情報

アプリケーション固有情報 (ASI) は、ビジネス・オブジェクトの処理方法に関するアプリケーション固有の指示を含むコネクタを提供します。ビジネス・オブジェクト定義を拡張または変更する場合は、定義内のアプリケーション固有情報が、コネクタが期待する構文と一致することを確認する必要があります。

アプリケーション固有情報は、ビジネス・オブジェクトおよび各ビジネス・オブジェクト属性で指定することができ、各動詞ごとに指定することもできます。単純属性の場合、ASI は eMatrix 属性のタイプまたは eMatrix プロパティ (名前、改訂、ポールトなど) の名前のいずれかを指定します。子ビジネス・オブジェクトの場合、ASI は (eMatrix ビジネス・オブジェクトおよび関係を表すオブジェクト階層の) 関係の方向を指定します。

ビジネス・オブジェクトの生成

Business Object Designer の提供するグラフィカル・インターフェースを使用すると、実行時に使用するビジネス・オブジェクト定義を生成できます。詳細については、31 ページの『第 5 章 ビジネス・オブジェクト定義の生成』を参照してください。

eMatrix 内のオブジェクト・モデルが変更された場合は、Business Object Designer の ODA を使用して、更新を反映するようにビジネス・オブジェクト定義を変更するか、または新規の定義を作成します。

第 5 章 ビジネス・オブジェクト定義の生成

この章では、Object Discovery Agent (ODA) for eMatrix の詳細と、それを使用して IBM WebSphere Business Integration Adapter for eMatrix のビジネス・オブジェクト定義を生成する方法について説明します。

本章の内容は、次のとおりです。

- 31 ページの『eMatrix 用の ODA の概要』
- 32 ページの『ビジネス・オブジェクト定義の生成』
- 38 ページの『ファイルのアップロード』

eMatrix 用の ODA の概要

ODA (Object Discovery Agent) を使用すると、アプリケーション固有のビジネス・オブジェクト定義を生成できます。ビジネス・オブジェクト定義は、ビジネス・オブジェクトのテンプレートです。ODA は、指定されたアプリケーション・オブジェクトを調べ、ビジネス・オブジェクト属性に対応するオブジェクトの要素を「ディスカバー」し、情報を表すビジネス・オブジェクト定義を生成します。Business Object Designer には、Object Discovery Agent にアクセスし、それを対話式に操作するためのグラフィカル・インターフェースが用意されています。

Object Discovery Agent (ODA) for eMatrix は、ビジネス・オブジェクトおよび関係タイプの両方をサポートします。ODA は、Business Object Designer で指定する初期設定プロパティを使用して、eMatrix データベースに接続します。

ODA は次に、eMatrix データベースのすべてのビジネス・オブジェクト・タイプ名を検索し、Business Object Designer にそれらを表示します。任意のビジネス・オブジェクト・タイプを展開し、それらに関連した関係を見ることができます。また、関係を展開することもできます。次に、1 つ以上のビジネス・オブジェクト・タイプ (親または子) および関係タイプを選択し、そのビジネス・オブジェクト定義を生成することができます。

Business Object Designer ウィザードは、ODA によって検索された eMatrix ビジネス・オブジェクト属性を使用して、定義作成プロセスを自動化します。子ビジネス・オブジェクトおよび関係タイプ用に作成されたビジネス・オブジェクト定義は、親ビジネス・オブジェクト定義に関連付けられます。ビジネス・オブジェクト定義はサーバーに保管する前に表示または修正できます。

ビジネス・オブジェクト定義を生成するために、ODA を次の 2 つの段階で使用できます。

1. **システムが最初にセットアップされる時:** eMatrix 内で定義されるすべてのイベントについて、一組のビジネス・オブジェクト定義を作成する必要があります。
2. **イベントで使用されるオブジェクトまたは関係タイプが eMatrix で変更される時 (常に):** Business Object Designer で ODA を実行し、変更されたオブジェ

クトのメタデータを選び出し、新しいビジネス・オブジェクト定義をブローカー・リポジトリにエクスポートします。

ビジネス・オブジェクト定義の生成

このセクションでは、Business Object Designer で eMatrix ODA を使用してビジネス・オブジェクト定義を生成する方法について説明します。Business Object Designer の起動方法と使用方法の詳細については、「*IBM WebSphere Business Integration Adapters* ビジネス・オブジェクト開発ガイド」を参照してください。

始める前に

eMatrix ODA を実行する前に、eMatrix RMI コラボレーション・サーバーをインストールする必要があります。コラボレーション・サーバーがインストールされると、ODA 用の RMI デーモンを実行して、サーバーに接続する必要があります。

ODA とサーバーは、異なるマシン上で実行できます。その場合、リモート・マシン上のコラボレーション・サーバーのホスト名、ポート、および名前を知っている必要があります。

ODA の始動

eMatrix 用の ODA のデフォルト名は eMatrixODA です。この名前は、開始スクリプト内の AGENTNAME 変数の値を変えることで変更できます。

- Windows から ODA を始動するには、次のように選択します。
「スタート」>「プログラム」>「IBM WebSphere Business Integration Adapters」
「アダプター」>「Object Discovery Agent」>「eMatrix Object Discovery Agent」
- このコマンド行から ODA を始動するには、次のコマンドを実行します。
`start_eMatrixODA`

Business Object Designer の実行

Business Object Designer には、ODA を使用してビジネス・オブジェクト定義を生成する手順を示すウィザードが用意されています。手順は以下のとおりです。

Business Object Designer の始動

Windows から Business Object Designer を実行するには、以下のようになります。

1. 「スタート」>「プログラム」>「IBM WebSphere Business Integration Adapters」>「ツール」>「Business Object Designer」
2. 「Business Object Designer」メイン・ウィンドウが開きます。
3. ODA を選択するには、「ファイル」>「ODA を使用して新規作成」を選択します。「ビジネス・オブジェクト・ウィザード - ステップ 1/6 - エージェントの選択」画面が表示されます (33 ページの図 3 を参照)。

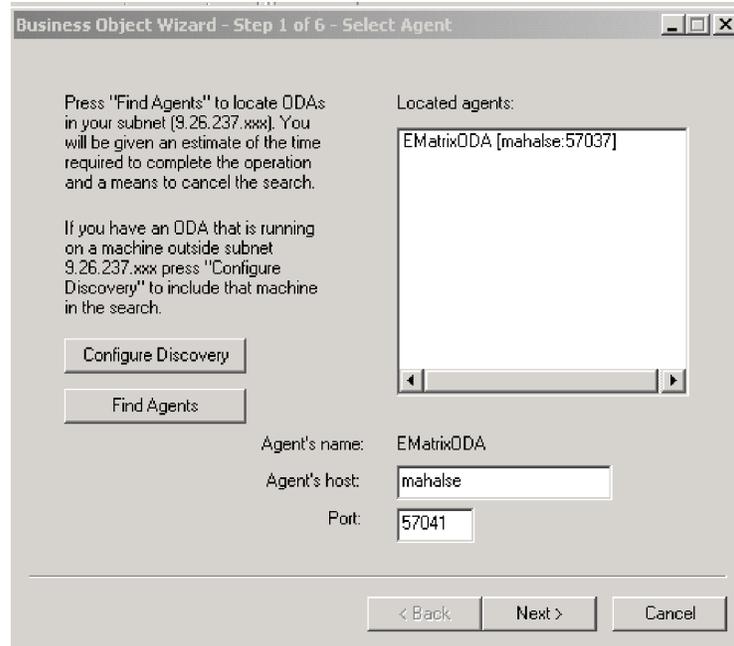


図 3. 「エージェントの選択」画面

4. 「エージェントの検索」をクリックします。eMatrixODA を含め、システムで実行されている ODA が検索され、「検索されたエージェント:」ウィンドウに表示されます。
5. eMatrixODA を選択して、「次へ」をクリックします。

エージェントの構成

「ビジネス・オブジェクト・ウィザード - ステップ 2/6 - エージェントの構成」画面が表示されます。

この画面で、eMatrix データベースと通信するコンテキストを設定するのに ODA が必要とする情報を入力します。必要なプロパティおよびそれらの値が 34 ページの表 12 に示されています。

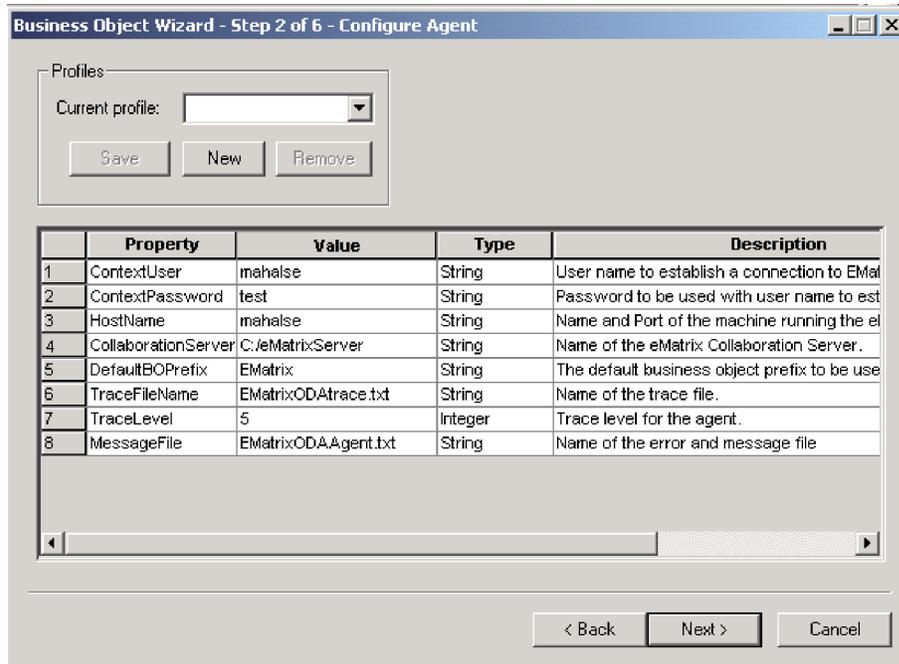


図4. 「エージェントの構成」画面

1. 「プロファイル」メニューを使用して新規プロファイルを作成します。既存プロファイルを選択することもできます。
2. 各プロパティの名前、値、タイプおよび説明を入力します。

注: プロファイルを使用すると、プロパティ値が入力されます。

表 12. ODA 構成用のプロパティ値

プロパティ名	プロパティ・タイプ	デフォルト値	説明	必須
ContextUser	ストリング	Creator	eMatrix データベースへ接続するコンテキスト・オブジェクトの設定に使用されるユーザー名。	はい
ContextPassword	ストリング	なし	eMatrix データベースへ接続するコンテキスト・オブジェクトの設定に使用されるパスワード。空にしておくこともできます。	はい
HostName	ストリング		eMatrix コラボレーション・サーバーのホスト名およびポート。katari:1099 など。	はい
CollaborationServer	ストリング		eMatrix コラボレーション・サーバーの名前。:bos など。	はい
DefaultBOPrefix	ストリング	EMatrix	ビジネス・オブジェクト定義のデフォルトの接頭部。	いいえ
TraceFileName	ストリング	<agentname> Trace.txt	トレース・ファイルの名前。	いいえ
TraceLevel	整数	5	ODA に適用されるトレース・レベル。	はい
MessageFile	ストリング	<agentname> Agent.txt	エラー・ファイルおよびメッセージ・ファイルの名前。	はい

eMatrix ODA によって表示されるすべてのメッセージは、標準メッセージ・ファイル・フォーマットの eMatrixODAAgent.txt ファイルに表示されます。

注: メッセージ・ファイルの名前が正しく指定されていない場合には、ODA はメッセージなしで実行されます。

プロファイルの作成

この画面に入力した値すべてをプロファイルに保管することもできます。ODA を次回に実行する際に、ドロップダウン・メニューからプロファイルを選択して保管しておいた値を使用すれば、すべてのデータを再入力する必要はありません。

同じプロパティに異なる値が指定された、複数のプロファイルを保管することができます。

入力が完了したら、「次へ」をクリックします。

ビジネス・オブジェクトの選択

「ビジネス・オブジェクト・ウィザード - ステップ 3/6 - ソースの選択」画面が表示されます。この画面を使用して、ODA がビジネス・オブジェクト定義を生成する、任意の数のビジネス・オブジェクトまたは関係タイプを選択します。

画面には、eMatrix で定義されているタイプがリストされています。eMatrix が初期化されていない場合には、リストは空になります。

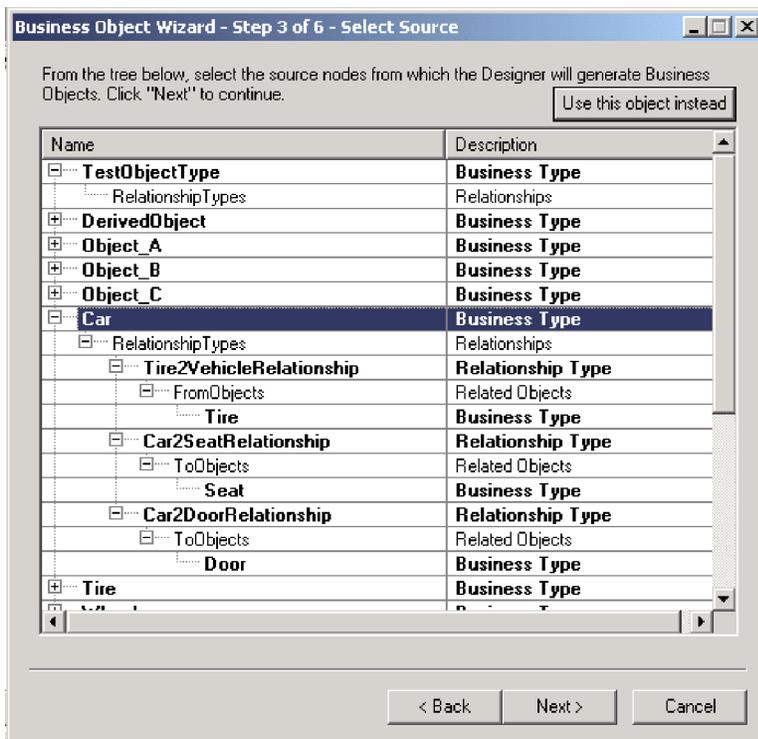


図 5. 「ソースの選択」画面

- ・ 親ビジネス・オブジェクトを展開して、その関係タイプのリストを見ます。

- 関係タイプを展開して、関連しているビジネス・オブジェクトのリストを見ます。
1. 使用するビジネス・オブジェクトのタイプを選択します。
 2. 「次へ」をクリックします。

選択の確認

「ビジネス・オブジェクト・ウィザード - ステップ 4/6 - ビジネス・オブジェクト定義のソース・ノードの確認」画面が表示されます。ここでは、選択したソース・ノードが表示されます。

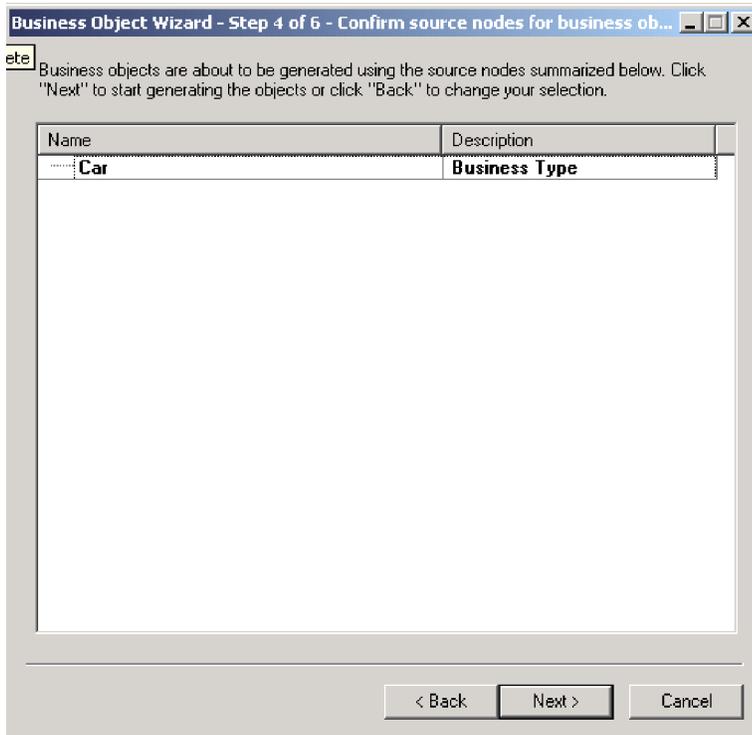


図 6. ソース・ノード画面を確認します。

「戻る」をクリックして変更するか、または「次へ」をクリックしてリストが正しいことを確認します。

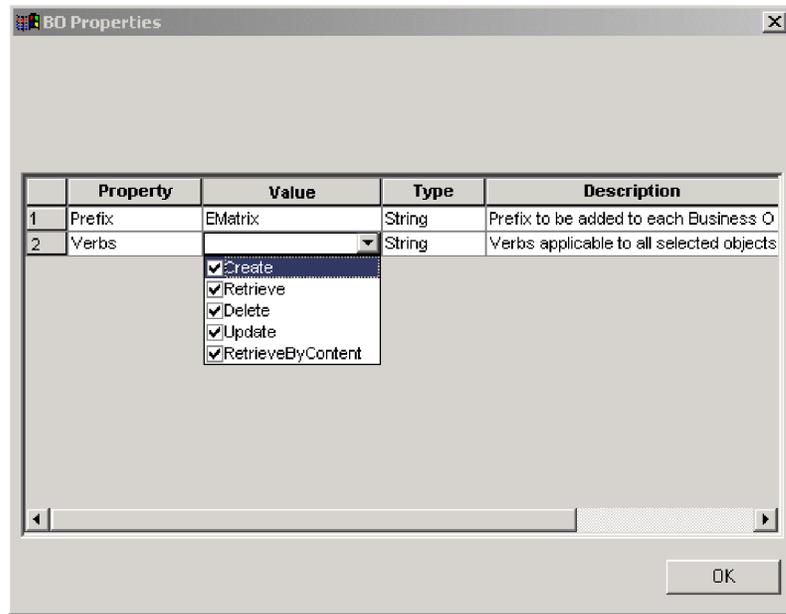


図7. 「ビジネス・オブジェクトの生成中」画面

次の画面で、ビジネス・オブジェクト・プロパティを設定する必要があります。ここで、ビジネス・オブジェクトが使用する、ビジネス・オブジェクトの接頭部および動詞を選択できます。複数の動詞を選択することもできます。ビジネス・オブジェクトに使用させたくない動詞の値を選択解除できます。

「OK」をクリックし、続けます。

ビジネス・オブジェクト定義の保管

「ビジネス・オブジェクト・ウィザード - ステップ 5/6 - ビジネス・オブジェクトの保管」画面が表示されます。

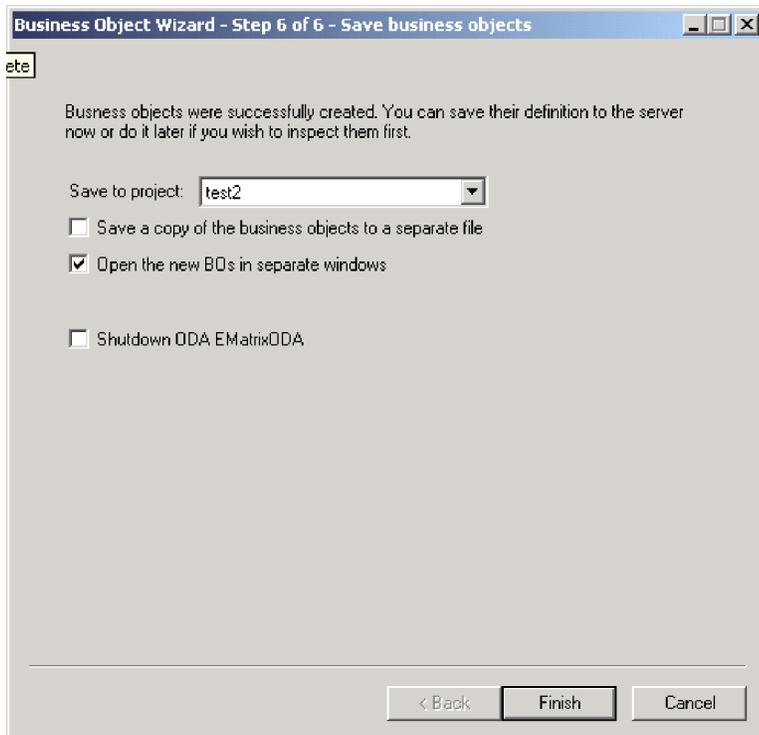


図 8. 「ビジネス・オブジェクトの保管」画面

ここで、ビジネス・オブジェクト定義を保管するロケーションを選択します。

生成されたビジネス・オブジェクト定義をオプションでファイルに保存できます。これを行うには、以下のステップを実行します。

1. 「ビジネス・オブジェクトをファイルに保管」にチェックマークを付けます。ダイアログ・ボックスが表示されます。
2. 保管する新規ビジネス・オブジェクト定義のコピーを格納する場所を入力します。

Business Object Designer によって、指定した場所にファイルが保管されます。

注: eMatrix 内で新規のカスタム・オブジェクトを作成した場合には、本章で説明されているように、ODA を使用してそのオブジェクトに新規のビジネス・オブジェクト定義を追加する必要があります。

ODA での作業が終了したら、「ODA: eMatrix ODA をシャットダウン (Shutdown ODA: eMatrix ODA)」にチェックマークを付け、「完了」をクリックします。

ファイルのアップロード

新規作成したビジネス・オブジェクト定義ファイルは、作成後すぐに統合ブローカーにアップロードする必要があります。このプロセスは、ICS、WebSphere Message Broker、または WAS のいずれを稼働しているかによって異なります。

ICS の場合: ビジネス・オブジェクト定義ファイルをローカル・マシンに保管しており、サーバーのリポジトリにアップロードする必要がある場合は、「*WebSphere InterChange Server* インプリメンテーション・ガイド」を参照してください。

WebSphere Message Broker の場合: ビジネス・オブジェクト定義を Business Object Designer からエクスポートし、統合ブローカーにインポートする必要があります。詳細については、「*WebSphere MQ Integrator Broker* 用インプリメンテーション・ガイド」を参照してください。

WAS の場合: Business Object Designer から統合ブローカーにビジネス・オブジェクト定義をエクスポートする必要があります。詳細については、「アダプター実装ガイド (*WebSphere Application Server*)」を参照してください。

第 6 章 トラブルシューティングおよびエラー処理

この章では、アダプターが共通エラーを処理する方法について説明し、共通問題のトラブルシューティングについて説明します。

本章の内容は、次のとおりです。

- 41 ページの『エラー処理』
- 42 ページの『トラブルシューティング』

エラー処理

一般に、アダプターはエラーを検出すると常に、コネクタ API によって、エラー・メッセージをログに記録します。

すべてのエラー・メッセージは、18 ページの『ログ・ファイルとトレース・ファイルの使用』で説明したように、外部メッセージ・ファイルに保管されます。

すべての操作に関して、eMatrix への接続が確立できなかった場合、または処理中に接続が失われた場合、アダプターは、コード APPRESPONSETIMEOUT を戻し、致命的エラーをログに記録します。これにより、コネクタは終了し、電子メール通知が管理者に送信されます。

未解決のイベント

アダプターは、構成プロパティの `inDoubtEvents` の値に応じて、未解決または未完了のイベントからリカバリーするためのアクションを実行します。

表 13 にオプションがリストされています。

表 13. 未確定イベントの処理

「inDoubtEvents」の値	実行するアクション
FailOnStartUp	致命的エラーをログに記録し、終了します。
Reprocess	通常のイベントと同じように、イベントをブローカーにパブリッシュします。
LogError	イベントを無視し、エラーをログに記録します。
Ignore	イベントを無視します。

ポーリング・エラー

ポーリング処理でいくつかのエラーが発生する場合があります。表 14 に最も一般的なものがリストされています。

表 14. ポーリング・エラーの処理

エラー	アクション
サブスクリプションされていないため、イベントをパブリッシュできません。	エラー・メッセージをログに記録し、場合によりイベントを保存し、イベントの処理を続けます。

表 14. ポーリング・エラーの処理 (続き)

エラー	アクション
コネクタ・コントローラーがアクティブでないため、イベントをパブリッシュできません。	イベント状況を 0 に設定し、SUCCESS コードを戻します。
エラーがこのイベントに固有のもので、イベントに無効な値が指定された場合などです。	エラー・メッセージをログに記録し、場合によりイベントを保存し、イベントの処理を続けます。
ブローカーは、エラー・メッセージを出してイベントを拒否します。	エラー・メッセージをログに記録し、場合によりイベントおよびエラー・メッセージを保存し、FAIL コードを戻します。
eMatrix と通信できません。その他のリカバリー不能エラーです。	致命エラー・メッセージをログに記録し、APPRESPONSETIMEOUT コードを戻します。

動詞処理エラー

アダプターは、ビジネス・オブジェクトを識別するために、固有のビジネス・オブジェクト ID または値の組み合わせ (タイプ、名前、改訂) を必要とします。指定された値が誤っている場合、または既存ビジネス・オブジェクトに一致しない場合、エラーが報告されます。

最も一般的な動詞処理エラーのいくつかを以下に示します。完全なリストについては、「*IBM WebSphere Business Integration Adapters コネクタ開発ガイド (Java 用)*」または「*IBM WebSphere Business Integration Adapters コネクタ開発ガイド (C++ 用)*」を参照してください。

- ビジネス・オブジェクトを作成する場合にアダプターが正常なときは、アダプターは、新しく作成したビジネス・オブジェクトを統合ブローカーに送り、値 SUCCEED を戻します。
指定された値を持ったビジネス・オブジェクトがすでに存在する場合、アダプターは VALDUPES を戻し、新しいビジネス・オブジェクトを作成しません。
- ビジネス・オブジェクトを更新する場合にアダプターが正常なときは、アダプターは、新しく更新したビジネス・オブジェクトを統合ブローカーに送り、値 VALCHANGE を戻します。
- 処理中にエラーが発生する場合、アダプターはブローカーに FAIL コードを戻し、説明エラー・メッセージをログに記録します。

トラブルシューティング

このセクションでは、Adapter for eMatrix の使用時に発生する共通問題に関して、いくつかの簡単なトラブルシューティングの手法を説明します。

イベント通知

このセクションでは、イベント通知中に発生する可能性のある 4 つの共通問題について説明します。共通問題は以下のとおりです。

- イベント生成の失敗
- eMatrix イベント・ビジネス・オブジェクト名の誤り
- イベント・ボルトの未検出
- ビジネス・オブジェクト階層の誤り

イベント生成の検査

イベント通知機能をデバッグする場合、まず、イベントが正常に生成されているかどうかを検査します。これを行うには、以下のステップを実行します。

1. WBIEventLogger を呼び出すトリガーを起動するアクションを実行します。
2. 次に、以下のような照会を実行し、イベントが生成されたかどうかを確認します。

```
temp query bus wbi_Event **;
```

3. イベントが生成されていない場合、WBIEventLogger を呼び出すトリガーがインストールされていることを確認します。

eMatrix Business Modeler の該当するビジネス・オブジェクトのタイプまたは関係定義を開くことによって、トリガーがインストールされているかどうかを検査することができます。イベント通知トリガーのインストール例については、83 ページの『付録 C. サンプル・シナリオ』を参照してください。

注: トリガーを呼び出すアクションを行うユーザーのユーザー名が、

WBIEventLogger で指定された adapterUser 定数に一致する場合、イベントは生成されません。adapterUser 定数は、アダプターが eMatrix システムにログインするとき使用するユーザー名を指定します。

そのようなイベントを無視する上で大事な点は、アダプターが eMatrix システムのエントティティを変更することによって、イベントがログに記録されるようになることを防ぐことにあります。ここでは、アダプターはそれ自身の活動を認識しているため、それらをログに記録する必要がないことが前提になっています。テストを行う場合はこのことに注意してください。

4. WBIEventLogger を実行するトリガーが呼び出されているかどうかをテストする場合、matrix.ini ファイルに次のプロパティを設定します。

```
MX_MQL_TRACE=TRUE
```

トリガーを呼び出すアクションを実行すると、次のようなメッセージが表示されます。

```
18:15:00.826 MQL t@2900 Session: mx3f20221364f87ac7
18:15:00.826 MQL t@2900 Program: WBIEventLogger
18:15:00.826 MQL t@2900 args: -method recordEvent ${OBJECTID} n=wbi_cpu
v=Update p=1
18:15:08.417 MQL t@2900 End Program:MQL<3>
```

このメッセージは、トリガーが、上にリストされた引き数で、WBIEventLogger を実行したことを示しています。

eMatrix イベント・ビジネス・オブジェクト名の検査

イベントが生成されていることがわかっている場合でも、eMatrix アプリケーションをポーリングするたびにエラーが発生するときは、イベント・ビジネス・オブジェクトの名前を詳しく調べてください。ビジネス・オブジェクト名は、次のようになっている必要があります。

```
wbi_22032.41802.50936.705_1058897313350_1892833611
```

ビジネス・オブジェクト名に FROMOBJECT のような大文字がある場合、引き数として無効な MARCO 値をトリガーに指定した可能性があります。これを解決するには、以下のようにします。

1. Business Modeler のすべてのビジネス・オブジェクト・タイプ定義を検査します。
2. それらが WBIEventLogger を呼び出すときに使用するトリガーを調べ、引き数として指定した MARCO 値がすべて正しいことを確認します。

イベント・ボールの検索

イベント・ポーリングのときに次のエラーを受け取る場合、アダプターが、コネクタ・プロパティーで参照される EventVault を見つけられなかったことを意味します。エラーは次のとおりです。

```
System Error: #1600039: vault 'WBI_Events' does not exist
```

このエラーは、以下のいずれかの理由で発生します。

- プロパティーで誤ったボールトを参照している。
- ボールトが作成されていない。

InstallEventTables を実行して、ボールトが作成されていることを確認し、コネクタ・プロパティーの EventVault の名前を検査してください。

ビジネス・オブジェクト階層の検査

イベント通知に使用されるビジネス・オブジェクト階層を作成する場合、階層に、親も eMatrix ビジネス・オブジェクト・タイプを表すような、eMatrix ビジネス・オブジェクト・タイプを表す子ビジネス・オブジェクト定義が含まれないようにしてください。

そのようなオブジェクトが存在すると、アダプターはポーリング時に失敗し、次のようなメッセージをログに記録します。

```
An error occurred while trying to validate your WBI business object instance hierarchy.
```

このエラーは、eMatrix ビジネス・オブジェクトを表すビジネス・オブジェクトが、ヌルでない eMatrix ビジネス・オブジェクトも表す属性を含むことができないために発生します。

要求処理

このセクションでは、要求処理中に発生する可能性のある 2 つの共通問題について説明します。共通問題は以下のとおりです。

- コマンドにおけるストリング値の誤り
- 階層ビジネス・オブジェクトの値の誤り

コマンドにおけるストリングの処理

eMatrix コマンドを表すビジネス・オブジェクトのストリング型属性の値は、コマンド処理前のコマンド・テンプレートへの挿入時に、引用符で囲まれません。したがって、引用符で囲まれていない空白文字を含む属性値は、2 語の MQL コマンドとして解釈されます。

ビジネス・オブジェクト階層の値

要求を送信する場合、階層内に、eMatrix ビジネス・オブジェクトを表す子があり、その親オブジェクトも eMatrix ビジネス・オブジェクトを表している場合があります。

す。子ビジネス・オブジェクトがヌルでないことを確認してください。そのような階層が検出されると、アダプターはエラーをスローします。

一般的なヒント

2 つの一般的な問題およびそれらの解決法を以下に示します。

サポートされないビジネス・オブジェクト

ログまたはトレース・ファイルで、次のようなメッセージを受け取る場合があります。

Unable to find version “*.*.*” of business object definition
“wbi_computer_bundle_to_computer”

これを解決するには、メッセージに示されたビジネス・オブジェクトを、Connector Configurator でサポートされるビジネス・オブジェクトのリストに追加する必要があります。

不完全なビジネス・オブジェクト関係

ODA を使用する場合、2 つのビジネス・オブジェクト・タイプとそれらの接続関係をモデル化したいときがあります。図 9 に示すように、必ず、階層関係にある 3 つすべてを選択してください。

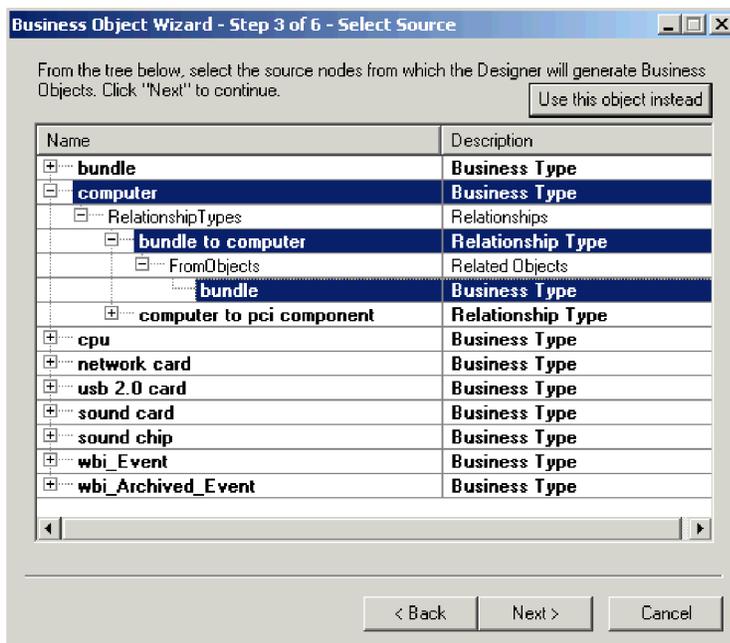


図 9. ビジネス・オブジェクト関係の選択

これを実行しないと、ODA は、タイプとそれらの共用関係が不完全なモデルを作成してしまいます。

付録 A. コネクターの標準構成プロパティ

この付録では、WebSphere Business Integration Adapter のコネクター・コンポーネントの標準構成プロパティについて説明します。この付録の内容は、以下の統合ブローカーで実行されるコネクターを対象としています。

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (WebSphere Message Brokers (WMQI) と総称)
- WebSphere Application Server (WAS)

コネクターによっては、一部の標準プロパティが使用されないことがあります。Connector Configurator から統合ブローカーを選択するときには、そのブローカーで実行されるアダプターについて構成する必要のある標準プロパティのリストが表示されます。

コネクター固有のプロパティの詳細については、該当するアダプターのユーザーズ・ガイドを参照してください。

注: 本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムを使用している場合は、円記号をスラッシュ (/) に置き換えてください。また、各オペレーティング・システムの規則に従ってください。

新規プロパティと削除されたプロパティ

以下の標準プロパティは、本リリースで追加されました。

新規プロパティ

- XMLNamespaceFormat

削除されたプロパティ

- RestartCount
- RHF2MessageDomain

標準コネクター・プロパティの構成

Adapter コネクターには 2 つのタイプの構成プロパティがあります。

- 標準構成プロパティ
- コネクター固有のプロパティ

このセクションでは、標準構成プロパティについて説明します。コネクター固有の構成プロパティについては、該当するアダプターのユーザーズ・ガイドを参照してください。

Connector Configurator の使用

Connector Configurator からコネクタ・プロパティを構成します。Connector Configurator には、System Manager からアクセスします。Connector Configurator の使用法の詳細については、付録の『Connector Configurator』を参照してください。

注: Connector Configurator と System Manager は、Windows システム上でのみ動作します。コネクタを UNIX システム上で稼働している場合でも、これらのツールがインストールされた Windows マシンが必要です。UNIX 上で動作するコネクタのコネクタ・プロパティを設定する場合は、Windows マシン上で System Manager を起動し、UNIX の統合ブローカーに接続してから、コネクタ一用の Connector Configurator を開く必要があります。

プロパティ値の設定と更新

プロパティ・フィールドのデフォルトの長さは 255 文字です。

コネクタは、以下の順序に従ってプロパティの値を決定します (最も番号の大きい項目が他の項目よりも優先されます)。

1. デフォルト
2. リポジトリ (WebSphere InterChange Server が統合ブローカーである場合のみ)
3. ローカル構成ファイル
4. コマンド行

コネクタは、始動時に構成値を取得します。実行時セッション中に 1 つ以上のコネクタ・プロパティの値を変更する場合は、プロパティの**更新メソッド**によって、変更を有効にする方法が決定されます。標準コネクタ・プロパティには、以下の 4 種類の更新メソッドがあります。

• 動的

変更を System Manager に保管すると、変更が即時に有効になります。コネクタが System Manager から独立してスタンドアロン・モードで稼働している場合 (例えば、いずれかの WebSphere Message Brokers と連携している場合) は、構成ファイルでのみプロパティを変更できます。この場合、動的更新は実行できません。

• コンポーネント再始動

System Manager でコネクタを停止してから再始動しなければ、変更が有効になりません。アプリケーション固有コンポーネントまたは統合ブローカーを停止、再始動する必要はありません。

• サーバー再始動

アプリケーション固有のコンポーネントおよび統合ブローカーを停止して再始動しなければ、変更が有効になりません。

• エージェント再始動 (ICS のみ)

アプリケーション固有のコンポーネントを停止して再始動しなければ、変更が有効になりません。

特定のプロパティの更新方法を確認するには、「Connector Configurator」ウィンドウ内の「更新メソッド」列を参照するか、次に示すプロパティの要約の表の「更新メソッド」列を参照してください。

標準プロパティの要約

表 15 は、標準コネクタ構成プロパティの早見表です。標準プロパティの依存関係は RepositoryDirectory に基づいているため、コネクタによっては使用されないプロパティがあり、使用する統合ブローカーによってプロパティの設定が異なる可能性があります。

コネクタを実行する前に、これらのプロパティの一部の値を設定する必要があります。各プロパティの詳細については、次のセクションを参照してください。

表 15. 標準構成プロパティの要約

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
AdminInQueue	有効な JMS キュー名	CONNECTORNAME/ADMININQUEUE	コンポーネント再始動	Delivery Transport は JMS
AdminOutQueue	有効な JMS キュー名	CONNECTORNAME/ADMINOUTQUEUE	コンポーネント再始動	Delivery Transport は JMS
AgentConnections	1 から 4	1	コンポーネント再始動	Delivery Transport は MQ および IDL: Repository Directory は <REMOTE>
AgentTraceLevel	0 から 5	0	動的	
ApplicationName	アプリケーション名	コネクタ・アプリケーション名として指定された値	コンポーネント再始動	
BrokerType	ICS、WMQI、WAS			
CharacterEncoding	ascii7、ascii8、SJIS、Cp949、GBK、Big5、Cp297、Cp273、Cp280、Cp284、Cp037、Cp437 注: これは、サポートされる値の一部です。	ascii7	コンポーネント再始動	
ConcurrentEventTriggeredFlows	1 から 32,767	1	コンポーネント再始動	Repository Directory は <REMOTE>
ContainerManagedEvents	値なしまたは JMS	値なし	コンポーネント再始動	Delivery Transport は JMS
ControllerStoreAndForwardMode	true または false	True	動的	Repository Directory は <REMOTE>
ControllerTraceLevel	0 から 5	0	動的	Repository Directory は <REMOTE>
DeliveryQueue		CONNECTORNAME/DELIVERYQUEUE	コンポーネント再始動	JMS トランスポートのみ
DeliveryTransport	MQ、IDL、または JMS	JMS	コンポーネント再始動	Repository Directory がローカルの場合は、値は JMS のみ

表 15. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
DuplicateEventElimination	True または False	False	コンポーネント再始動	JMS トランスポートのみ、Container Managed Events は <NONE> でなければならない
FaultQueue		CONNECTORNAME/FAULTQUEUE	コンポーネント再始動	JMS トランスポートのみ
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory または CxCommon.Messaging.jms.SonicMQFactory または任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	コンポーネント再始動	JMS トランスポートのみ
jms.MessageBrokerName	FactoryClassName が IBM の場合は crossworlds.queue.manager を使用。FactoryClassName が Sonic の場合は localhost:2506 を使用。	crossworlds.queue.manager	コンポーネント再始動	JMS トランスポートのみ
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	JMS トランスポートのみ
jms.Password	任意の有効なパスワード		コンポーネント再始動	JMS トランスポートのみ
jms.UserName	任意の有効な名前		コンポーネント再始動	JMS トランスポートのみ
JvmMaxHeapSize	ヒープ・サイズ (メガバイト単位)	128m	コンポーネント再始動	Repository Directory は <REMOTE>
JvmMaxNativeStackSize	スタックのサイズ (キロバイト単位)	128k	コンポーネント再始動	Repository Directory は <REMOTE>
JvmMinHeapSize	ヒープ・サイズ (メガバイト単位)	1m	コンポーネント再始動	Repository Directory は <REMOTE>
ListenerConcurrency	1 から 100	1	コンポーネント再始動	Delivery Transport は MQ でなければならない
Locale	en_US、ja_JP、ko_KR、zh_CN、zh_TW、fr_FR、de_DE、it_IT、es_ES、pt_BR 注: これは、サポートされるロケールの一部です。	en_US	コンポーネント再始動	
LogAtInterchangeEnd	True または False	False	コンポーネント再始動	Repository Directory は <REMOTE> でなければならない

表 15. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
MaxEventCapacity	1 から 2147483647	2147483647	動的	Repository Directory は <REMOTE> でなければならぬ
MessageFileName	パスまたはファイル名	InterchangeSystem.txt	コンポーネント再始動	
MonitorQueue	任意の有効なキュー名	CONNECTORNAME/MONITORQUEUE	コンポーネント再始動	JMS トランスポートのみ: DuplicateEvent Elimination は True でなければならぬ
OADAutoRestartAgent	True または False	False	動的	Repository Directory は <REMOTE> でなければならぬ
OADMaxNumRetry	正数	1000	動的	Repository Directory は <REMOTE> でなければならぬ
OADRetryTimeInterval	正数 (単位: 分)	10	動的	Repository Directory は <REMOTE> でなければならぬ
PollEndTime	HH:MM	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒) no (ポーリングを使用不可にする) key (コネクタのコマンド・プロンプト・ウィンドウで文字 p が入力された場合にのみポーリングする)	10000	動的	
PollQuantity	1 から 500	1	エージェント再始動	JMS トランスポートのみ: Container Managed Events を指定
PollStartTime	HH:MM (HH は 0 から 23、MM は 0 から 59)	HH:MM	コンポーネント再始動	
RepositoryDirectory	メタデータ・リポジトリの場所		エージェント再始動	ICS の場合は <REMOTE> に設定する。 WebSphere MQ Message Brokers および WAS の場合: C:\crossworlds¥repository に設定する

表 15. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
RequestQueue	有効な JMS キュー名	CONNECTORNAME/REQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
ResponseQueue	有効な JMS キュー名	CONNECTORNAME/RESPONSEQUEUE	コンポーネント再始動	Delivery Transport が JMS の場合: Repository Directory が <REMOTE> の場合のみ必要
RestartRetryCount	0 から 99	3	動的	
RestartRetryInterval	適切な正数 (単位: 分): 1 から 2147483547	1	動的	
SourceQueue	有効な WebSphere MQ 名	CONNECTORNAME/SOURCEQUEUE	エージェント再始動	Delivery Transport が JMS であり、かつ Container Managed Events が指定されている場合のみ
SynchronousRequestQueue		CONNECTORNAME/ SYNCHRONOUSREQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
SynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	Delivery Transport は JMS
SynchronousResponseQueue		CONNECTORNAME/ SYNCHRONOUSRESPONSEQUEUE	コンポーネント再始動	Delivery Transport は JMS
WireFormat	CwXML、CwBO	CwXML	エージェント再始動	Repository Directory が <REMOTE> でない場合は CwXML: Repository Directory が <REMOTE> の場合は CwBO
WsifSynchronousRequest Timeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	WAS のみ
XMLNamespaceFormat	short、long	short	エージェント再始動	WebSphere MQ Message Brokers および WAS のみ

標準構成プロパティ

このセクションでは、各標準コネクタ構成プロパティの定義を示します。

AdminInQueue

統合ブローカーからコネクタへ管理メッセージが送信されるときに使用されるキューです。

デフォルト値は CONNECTORNAME/ADMININQUEUE です。

AdminOutQueue

コネクタから統合ブローカーへ管理メッセージが送信されるときに使用されるキューです。

デフォルト値は `CONNECTORNAME/ADMINOUTQUEUE` です。

AgentConnections

`RepositoryDirectory` が `<REMOTE>` の場合のみ適用されます。

`AgentConnections` プロパティは、`orb.init[]` により開かれる ORB 接続の数を制御します。

デフォルトでは、このプロパティの値は 1 に設定されます。このデフォルト値を変更する必要はありません。

AgentTraceLevel

アプリケーション固有のコンポーネントのトレース・メッセージのレベルです。デフォルト値は 0 です。コネクタは、設定されたトレース・レベル以下の該当するトレース・メッセージをすべてデリバリーします。

ApplicationName

コネクタのアプリケーションを一意的に特定する名前です。この名前は、システム管理者が WebSphere Business Integration システム環境をモニターするために使用されます。コネクタを実行する前に、このプロパティに値を指定する必要があります。

BrokerType

使用する統合ブローカー・タイプを指定します。オプションは ICS、WebSphere Message Brokers (WMQI、WMQIB または WBIMB) または WAS です。

CharacterEncoding

文字 (アルファベットの文字、数値表現、句読記号など) から数値へのマッピングに使用する文字コード・セットを指定します。

注: Java ベースのコネクタでは、このプロパティは使用しません。C++ ベースのコネクタでは、現在、このプロパティに `ascii7` という値が使用されています。

デフォルトでは、ドロップ・リストには、サポートされる文字エンコードの一部のみが表示されます。ドロップ・リストに、サポートされる他の値を追加するには、製品ディレクトリーにある `¥Data¥Std¥stdConnProps.xml` ファイルを手動で変更する必要があります。詳細については、Connector Configurator に関する付録を参照してください。

ConcurrentEventTriggeredFlows

`RepositoryDirectory` が `<REMOTE>` の場合のみ適用されます。

コネクターがイベントのデリバリー時に並行処理できるビジネス・オブジェクトの数を決定します。この属性の値を、並行してマップおよびデリバリーできるビジネス・オブジェクトの数に設定します。例えば、この属性の値を 5 に設定すると、5 個のビジネス・オブジェクトが並行して処理されます。デフォルト値は 1 です。

このプロパティを 1 よりも大きい値に設定すると、ソース・アプリケーションのコネクターが、複数のイベント・ビジネス・オブジェクトを同時にマップして、複数のコラボレーション・インスタンスにそれらのビジネス・オブジェクトを同時にデリバリーすることができます。これにより、統合ブローカーへのビジネス・オブジェクトのデリバリーにかかる時間、特にビジネス・オブジェクトが複雑なマップを使用している場合のデリバリー時間が短縮されます。ビジネス・オブジェクトのコラボレーションに到達する速度を増大させると、システム全体のパフォーマンスを向上させることができます。

ソース・アプリケーションから宛先アプリケーションまでのフロー全体に並行処理を実装するには、次のようにする必要があります。

- **Maximum number of concurrent events** プロパティの値を増加して、コラボレーションが複数のスレッドを使用できるように構成します。
- 宛先アプリケーションのアプリケーション固有コンポーネントが複数の要求を並行して実行できることを確認します。つまり、このコンポーネントがマルチスレッド化されているか、またはコネクター・エージェント並列処理を使用でき、複数プロセスに対応するよう構成されている必要があります。Parallel Process Degree 構成プロパティに、1 より大きい値を設定します。

ConcurrentEventTriggeredFlows プロパティは、順次に実行される単一スレッド処理であるコネクターのポーリングでは無効です。

ContainerManagedEvents

このプロパティにより、JMS イベント・ストアを使用する JMS 対応コネクターが、保証付きイベント・デリバリーを提供できるようになります。保証付きイベント・デリバリーでは、イベントはソース・キューから除去され、単一 JMS トランザクションとして宛先キューに配置されます。

デフォルト値は No value です。

ContainerManagedEvents を JMS に設定した場合には、保証付きイベント・デリバリーを使用できるように次のプロパティも構成する必要があります。

- PollQuantity = 1 から 500
- SourceQueue = CONNECTORNAME/SOURCEQUEUE

また、MimeType、DHClass、および DataHandlerConfigMOName (オプション) プロパティを設定したデータ・ハンドラーも構成する必要があります。これらのプロパティの値を設定するには、Connector Configurator の「データ・ハンドラー」タブを使用します。「データ・ハンドラー」タブの値のフィールドは、ContainerManagedEvents を JMS に設定した場合にのみ表示されます。

注: ContainerManagedEvents を JMS に設定した場合、コネクターはその pollForEvents() メソッドを呼び出さなくなるため、そのメソッドの機能は使用できなくなります。

このプロパティは、DeliveryTransport プロパティが値 JMS に設定されている場合にのみ表示されます。

ControllerStoreAndForwardMode

RepositoryDirectory が <REMOTE> の場合のみ適用されます。

宛先側のアプリケーション固有のコンポーネントが使用不可であることをコネクタ・コントローラーが検出した場合に、コネクタ・コントローラーが実行する動作を設定します。

このプロパティを true に設定した場合、イベントが ICS に到達したときに宛先側のアプリケーション固有のコンポーネントが使用不可であれば、コネクタ・コントローラーはそのアプリケーション固有のコンポーネントへの要求をブロックします。アプリケーション固有のコンポーネントが作動可能になると、コネクタ・コントローラーはアプリケーション固有のコンポーネントにその要求を転送します。

ただし、コネクタ・コントローラーが宛先側のアプリケーション固有のコンポーネントにサービス呼び出し要求を転送した後でこのコンポーネントが使用不可になった場合、コネクタ・コントローラーはその要求を失敗させます。

このプロパティを false に設定した場合、コネクタ・コントローラーは、宛先側のアプリケーション固有のコンポーネントが使用不可であることを検出すると、ただちにすべてのサービス呼び出し要求を失敗させます。

デフォルト値は true です。

ControllerTraceLevel

RepositoryDirectory が <REMOTE> の場合のみ適用されます。

コネクタ・コントローラーのトレース・メッセージのレベルです。デフォルト値は 0 です。

DeliveryQueue

DeliveryTransport が JMS の場合のみ適用されます。

コネクタから統合ブローカーへビジネス・オブジェクトが送信されるときに使用されるキューです。

デフォルト値は CONNECTORNAME/DELIVERYQUEUE です。

DeliveryTransport

イベントのデリバリーのためのトランスポート機構を指定します。指定可能な値は、WebSphere MQ の MQ、CORBA IIOP の IDL、Java Messaging Service の JMS です。

- ICS がブローカー・タイプの場合は、DeliveryTransport プロパティの指定可能な値は MQ、IDL、または JMS であり、デフォルトは IDL になります。

- `RepositoryDirectory` がローカル・ディレクトリーの場合は、指定可能な値は `JMS` のみです。

`DeliveryTransport` プロパティに指定されている値が、MQ または IDL である場合、コネクタは、CORBA IIOP を使用してサービス呼び出し要求と管理メッセージを送信します。

WebSphere MQ および IDL

イベントのデリバリー・トランスポートには、IDL ではなく WebSphere MQ を使用してください (1 種類の製品だけを使用する必要がある場合を除きます)。

WebSphere MQ が IDL よりも優れている点は以下のとおりです。

- 非同期 (ASYNC) 通信:
WebSphere MQ を使用すると、アプリケーション固有のコンポーネントは、サーバーが利用不能である場合でも、イベントをポーリングして永続的に格納することができます。
- サーバー・サイド・パフォーマンス:
WebSphere MQ を使用すると、サーバー・サイドのパフォーマンスが向上します。最適化モードでは、WebSphere MQ はイベントへのポインターのみをリポジトリ・データベースに格納するので、実際のイベントは WebSphere MQ キュー内に残ります。これにより、サイズが大きい可能性のあるイベントをリポジトリ・データベースに書き込む必要がありません。
- エージェント・サイド・パフォーマンス:
WebSphere MQ を使用すると、アプリケーション固有のコンポーネント側のパフォーマンスが向上します。WebSphere MQ を使用すると、コネクタのポーリング・スレッドは、イベントを選出した後、コネクタのキューにそのイベントを入れ、次のイベントを選出します。この方法は IDL よりも高速で、IDL の場合、コネクタのポーリング・スレッドは、イベントを選出した後、ネットワーク経由でサーバー・プロセスにアクセスしてそのイベントをリポジトリ・データベースに永続的に格納してから、次のイベントを選出する必要があります。

JMS

Java Messaging Service (JMS) を使用しての、コネクタとクライアント・コネクタ・フレームワークとの間の通信を可能にします。

JMS をデリバリー・トランスポートとして選択した場合は、`jms.MessageBrokerName`、`jms.FactoryClassName`、`jms.Password`、`jms.UserName` などの追加の JMS プロパティが `Connector Configurator` 内に表示されます。このうち最初の 2 つは、このトランスポートの必須プロパティです。

重要: 以下の環境では、コネクタに JMS トランスポート機構を使用すると、メモリ制限が発生することもあります。

- AIX 5.0
- WebSphere MQ 5.3.0.1
- ICS が統合ブローカーの場合

この環境では、WebSphere MQ クライアント内でメモリが使用されるため、(サーバー側の) コネクタ・コントローラーと (クライアント側の) コネクタの両方を

始動するのは困難な場合があります。ご使用のシステムのプロセス・ヒープ・サイズが 768M 未満である場合には、次のように設定することをお勧めします。

- CWSHaredEnv.sh スクリプト内で LDR_CNTRL 環境変数を設定する。

このスクリプトは、製品ディレクトリー配下の %bin ディレクトリーにあります。テキスト・エディターを使用して、CWSHaredEnv.sh スクリプトの最初の行として次の行を追加します。

```
export LDR_CNTRL=MAXDATA=0x30000000
```

この行は、ヒープ・メモリーの使用量を最大 768 MB (3 セグメント * 256 MB) に制限します。プロセス・メモリーがこの制限値を超えると、ページ・スワッピングが発生し、システムのパフォーマンスに悪影響を与える場合があります。

- IPCCBaseAddress プロパティの値を 11 または 12 に設定する。このプロパティの詳細については、「システム・インストール・ガイド (UNIX 版)」を参照してください。

DuplicateEventElimination

このプロパティを true に設定すると、JMS 対応コネクタによるデリバリー・キューへの重複イベントのデリバリーが防止されます。この機能を使用するには、コネクタに対し、アプリケーション固有のコード内でビジネス・オブジェクトの **ObjectEventId** 属性として一意のイベント ID が設定されている必要があります。これはコネクタ開発時に設定されます。

このプロパティは、false に設定することもできます。

注: DuplicateEventElimination を true に設定する際は、MonitorQueue プロパティを構成して保証付きイベント・デリバリーを使用可能にする必要があります。

FaultQueue

コネクタでメッセージを処理中にエラーが発生すると、コネクタは、そのメッセージを状況表示および問題説明とともにこのプロパティに指定されているキューに移動します。

デフォルト値は CONNECTORNAME/FAULTQUEUE です。

JvmMaxHeapSize

エージェントの最大ヒープ・サイズ (メガバイト単位)。このプロパティは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 128M です。

JvmMaxNativeStackSize

エージェントの最大ネイティブ・スタック・サイズ (キロバイト単位)。このプロパティは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 128K です。

JvmMinHeapSize

エージェントの最小ヒープ・サイズ (メガバイト単位)。このプロパティは、RepositoryDirectory の値が <REMOTE> の場合にのみ適用されます。

デフォルト値は 1M です。

jms.FactoryClassName

JMS プロバイダーのためにインスタンスを生成するクラス名を指定します。JMS をデリバリー・トランスポート機構 (DeliveryTransport) として選択する際は、このコネクター・プロパティを必ず設定してください。

デフォルト値は CxCommon.Messaging.jms.IBMMQSeriesFactory です。

jms.MessageBrokerName

JMS プロバイダーのために使用するブローカー名を指定します。JMS をデリバリー・トランスポート機構 (DeliveryTransport) として選択する際は、このコネクター・プロパティを必ず設定してください。

デフォルト値は crossworlds.queue.manager です。

jms.NumConcurrentRequests

コネクターに対して同時に送信することができる並行サービス呼び出し要求の数 (最大値) を指定します。この最大値に達した場合、新規のサービス呼び出し要求はブロックされ、既存のいずれかの要求が完了した後で処理されます。

デフォルト値は 10 です。

jms.Password

JMS プロバイダーのためのパスワードを指定します。このプロパティの値はオプションです。

デフォルトはありません。

jms.UserName

JMS プロバイダーのためのユーザー名を指定します。このプロパティの値はオプションです。

デフォルトはありません。

ListenerConcurrency

このプロパティは、統合ブローカーとして ICS を使用する場合の MQ Listener でのマルチスレッド化をサポートしています。このプロパティにより、データベースへの複数イベントの書き込み操作をバッチ処理できるので、システム・パフォーマンスが向上します。デフォルト値は 1 です。

このプロパティは、MQ トランスポートを使用するコネクターにのみ適用されます。DeliveryTransport プロパティには MQ を設定してください。

Locale

言語コード、国または地域、および、希望する場合には、関連した文字コード・セットを指定します。このプロパティの値は、データの照合やソート順、日付と時刻の形式、通貨記号などの国/地域別情報を決定します。

ロケール名は、次の書式で指定します。

`ll_TT.codeset`

ここで、以下のように説明されます。

<code>ll</code>	2 文字の言語コード (普通は小文字)
<code>TT</code>	2 文字の国または地域コード (普通は大文字)
<code>codeset</code>	関連文字コード・セットの名前。名前のこの部分は、通常、オプションです。

デフォルトでは、ドロップ・リストには、サポートされるロケールの一部のみが表示されます。ドロップ・リストに、サポートされる他の値を追加するには、製品ディレクトリーにある `¥Data¥Std¥stdConnProps.xml` ファイルを手動で変更する必要があります。詳細については、Connector Configurator に関する付録を参照してください。

デフォルト値は `en_US` です。コネクタがグローバル化に対応していない場合、このプロパティの有効な値は `en_US` のみです。特定のコネクタがグローバル化に対応しているかどうかを判別するには、以下の Web サイトにあるコネクタのバージョン・リストを参照してください。

<http://www.ibm.com/software/websphere/wbiadapters/infocenter>、または
<http://www.ibm.com/websphere/integration/wicsserver/infocenter>

LogAtInterchangeEnd

`RepositoryDirectory` が `<REMOTE>` の場合のみ適用されます。

統合ブローカーのログ宛先にエラーを記録するかどうかを指定します。ブローカーのログ宛先にログを記録すると、電子メール通知もオンになります。これにより、エラーまたは致命的エラーが発生すると、`InterchangeSystem.cfg` ファイルに指定された `MESSAGE_RECIPIENT` に対する電子メール・メッセージが生成されます。

例えば、`LogAtInterChangeEnd` を `true` に設定した場合にコネクタからアプリケーションへの接続が失われると、指定されたメッセージ宛先に、電子メール・メッセージが送信されます。デフォルト値は `false` です。

MaxEventCapacity

コントローラー・バッファ内のイベントの最大数。このプロパティはフロー制御が使用し、`RepositoryDirectory` プロパティの値が `<REMOTE>` の場合のみ適用されます。

値は 1 から 2147483647 の間の正整数です。デフォルト値は 2147483647 です。

MessageFileName

コネクタ・メッセージ・ファイルの名前です。メッセージ・ファイルの標準位置は `¥connectors¥messages` です。メッセージ・ファイルが標準位置に格納されていない場合は、メッセージ・ファイル名を絶対パスで指定します。

コネクタ・メッセージ・ファイルが存在しない場合は、コネクタは `InterchangeSystem.txt` をメッセージ・ファイルとして使用します。このファイルは、製品ディレクトリーに格納されています。

注: 特定のコネクタについて、コネクタ独自のメッセージ・ファイルがあるかどうかを判別するには、該当するアダプターのユーザーズ・ガイドを参照してください。

MonitorQueue

コネクタが重複イベントをモニターするために使用する論理キューです。このプロパティは、`DeliveryTransport` プロパティ値が `JMS` であり、かつ `DuplicateEventElimination` が `TRUE` に設定されている場合にのみ使用されます。

デフォルト値は `CONNECTORNAME/MONITORQUEUE` です。

OADAutoRestartAgent

`RepositoryDirectory` が `<REMOTE>` の場合のみ有効です。

コネクタが自動再始動機能およびリモート再始動機能を使用するかどうかを指定します。この機能では、MQ により起動される Object Activation Daemon (OAD) を使用して、異常シャットダウン後にコネクタを再始動したり、System Monitor からリモート・コネクタを始動したりします。

自動再始動機能およびリモート再始動機能を使用可能にするには、このプロパティを `true` に設定する必要があります。MQ により起動される OAD 機能の構成方法については、「システム・インストール・ガイド (Windows 版)」または「システム・インストール・ガイド (UNIX 版)」を参照してください。

デフォルト値は `false` です。

OADMaxNumRetry

`RepositoryDirectory` が `<REMOTE>` の場合のみ有効です。

異常シャットダウンの後で MQ により起動される OAD がコネクタの再始動を自動的に試行する回数の最大数を指定します。このプロパティを有効にするには、`OADAutoRestartAgent` プロパティを `true` に設定する必要があります。

デフォルト値は `1000` です。

OADRetryTimeInterval

`RepositoryDirectory` が `<REMOTE>` の場合のみ有効です。

MQ により起動される OAD の再試行時間間隔の分数を指定します。コネクタ・エージェントがこの再試行時間間隔内に再始動しない場合は、コネクタ・コント

ローラーはコネクタ・エージェントを再び再始動するように OAD に要求します。OAD はこの再試行プロセスを OADMaxNumRetry プロパティで指定された回数だけ繰り返します。このプロパティを有効にするには、OADAutoRestartAgent プロパティを true に設定する必要があります。

デフォルト値は 10 です。

PollEndTime

イベント・キューのポーリングを停止する時刻です。形式は HH:MM です。ここで、HH は 0 から 23 時を表し、MM は 0 から 59 分を表します。

このプロパティには必ず有効な値を指定してください。デフォルト値は HH:MM ですが、この値は必ず変更する必要があります。

PollFrequency

ポーリング・アクション間の時間の長さです。PollFrequency は以下の値のいずれかに設定します。

- ポーリング・アクション間のミリ秒数。
- ワード key。コネクタは、コネクタのコマンド・プロンプト・ウィンドウで文字 p が入力されたときにのみポーリングを実行します。このワードは小文字で入力します。
- ワード no。コネクタはポーリングを実行しません。このワードは小文字で入力します。

デフォルト値は 10000 です。

重要: 一部のコネクタでは、このプロパティの使用が制限されています。このプロパティが使用されるかどうかを特定のコネクタについて判別するには、該当するアダプター・ガイドのインストールと構成についての章を参照してください。

PollQuantity

コネクタがアプリケーションからポーリングする項目の数を指定します。アダプターにコネクタ固有のポーリング数設定プロパティがある場合、標準プロパティの値は、このコネクタ固有のプロパティの設定値によりオーバーライドされます。

PollStartTime

イベント・キューのポーリングを開始する時刻です。形式は HH:MM です。ここで、HH は 0 から 23 時を表し、MM は 0 から 59 分を表します。

このプロパティには必ず有効な値を指定してください。デフォルト値は HH:MM ですが、この値は必ず変更する必要があります。

RequestQueue

統合ブローカーが、ビジネス・オブジェクトをコネクタに送信するときに使用されるキューです。

デフォルト値は CONNECTOR/REQUESTQUEUE です。

RepositoryDirectory

コネクタが XML スキーマ文書を読み取るリポジトリの場所です。この XML スキーマ文書には、ビジネス・オブジェクト定義のメタデータが含まれています。

統合ブローカーが ICS の場合はこの値を <REMOTE> に設定する必要があります。これは、コネクタが InterChange Server リポジトリからこの情報を取得するためです。

統合ブローカーが WebSphere Message Broker または WAS の場合には、この値を <local directory>に設定する必要があります。

ResponseQueue

DeliveryTransport が JMS の場合のみ適用され、RepositoryDirectory が <REMOTE> の場合のみ必要です。

JMS 応答キューを指定します。JMS 応答キューは、応答メッセージをコネクタ・フレームワークから統合ブローカーへデリバリーします。統合ブローカーが ICS の場合、サーバーは要求を送信し、JMS 応答キューの応答メッセージを待ちます。

RestartRetryCount

コネクタによるコネクタ自体の再始動の試行回数を指定します。このプロパティを並列コネクタに対して使用する場合、コネクタのマスター側のアプリケーション固有のコンポーネントがスレーブ側のアプリケーション固有のコンポーネントの再始動を試行する回数が指定されます。

デフォルト値は 3 です。

RestartRetryInterval

コネクタによるコネクタ自体の再始動の試行間隔を分単位で指定します。このプロパティを並列コネクタに対して使用する場合、コネクタのマスター側のアプリケーション固有のコンポーネントがスレーブ側のアプリケーション固有のコンポーネントの再始動を試行する間隔が指定されます。指定可能な値の範囲は 1 から 2147483647 です。

デフォルト値は 1 です。

SourceQueue

DeliveryTransport が JMS で、ContainerManagedEvents が指定されている場合のみ適用されます。

JMS イベント・ストアを使用する JMS 対応コネクタでの保証付きイベント・デリバリーをサポートするコネクタ・フレームワークに、JMS ソース・キューを指定します。詳細については、54 ページの『ContainerManagedEvents』を参照してください。

デフォルト値は CONNECTOR/SOURCEQUEUE です。

SynchronousRequestQueue

DeliveryTransport が JMS の場合のみ適用されます。

同期応答を要求する要求メッセージを、コネクタ・フレームワークからブローカーに配信します。このキューは、コネクタが同期実行を使用する場合にのみ必要です。同期実行の場合、コネクタ・フレームワークは、SynchronousRequestQueue にメッセージを送信し、SynchronousResponseQueue でブローカーから戻される応答を待機します。コネクタに送信される応答メッセージには、元のメッセージの ID を指定する 相関 ID が含まれています。

デフォルトは CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE です。

SynchronousResponseQueue

DeliveryTransport が JMS の場合のみ適用されます。

同期要求に対する応答として送信される応答メッセージを、ブローカーからコネクタ・フレームワークに配信します。このキューは、コネクタが同期実行を使用する場合にのみ必要です。

デフォルトは CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE です。

SynchronousRequestTimeout

DeliveryTransport が JMS の場合のみ適用されます。

コネクタが同期要求への応答を待機する時間を分単位で指定します。コネクタは、指定された時間内に応答を受信できなかった場合、元の同期要求メッセージをエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

WireFormat

トランスポートのメッセージ・フォーマットです。

- RepositoryDirectory がローカル・ディレクトリーの場合は、設定は CwXML になります。
- RepositoryDirectory の値が <REMOTE> の場合は、設定は CwB0 になります。

WsifSynchronousRequest Timeout

WAS 統合ブローカーでのみ使用されます。

コネクタが同期要求への応答を待機する時間を分単位で指定します。コネクタは、指定された時間内に応答を受信できなかった場合、元の同期要求メッセージをエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

XMLNamespaceFormat

WebSphere Message Brokers および WAS 統合ブローカーでのみ使用されます。

ビジネス・オブジェクト定義の XML 形式でネーム・スペースを short と long のどちらにするかをユーザーが指定できるようにするための、強力なプロパティです。

デフォルト値は short です。

付録 B. Connector Configurator

この付録では、Connector Configurator を使用してアダプターの構成プロパティ値を設定する方法について説明します。

Connector Configurator を使用して次の作業を行います。

- コネクタを構成するためのコネクタ固有のプロパティ・テンプレートを作成する
- 構成ファイルを作成する
- 構成ファイル内のプロパティを設定する

注:

本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムを使用している場合は、円記号をスラッシュ (/) に置き換えてください。また、各オペレーティング・システムの規則に従ってください。

この付録では、次のトピックについて説明します。

- 65 ページの『Connector Configurator の概要』
- 66 ページの『Connector Configurator の始動』
- 67 ページの『コネクタ固有のプロパティ・テンプレートの作成』
- 70 ページの『新しい構成ファイルの作成』
- 73 ページの『構成ファイル・プロパティの設定』
- 81 ページの『グローバル化環境における Connector Configurator の使用』

Connector Configurator の概要

Connector Configurator では、次の統合ブローカーで使用するアダプターのコネクタ・コンポーネントを構成できます。

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (WebSphere Message Brokers (WMQI) と総称)
- WebSphere Application Server (WAS)

Connector Configurator を使用して次の作業を行います。

- コネクタを構成するためのコネクタ固有のプロパティ・テンプレートを作成します。
- **コネクタ構成ファイル**を作成します。インストールするコネクタごとに構成ファイルを 1 つ作成する必要があります。
- 構成ファイル内のプロパティを設定します。
場合によっては、コネクタ・テンプレートでプロパティに対して設定されているデフォルト値を変更する必要があります。また、サポートされるビジネス・オブジェクト定義と、ICS の場合はコラボレーションとともに使用するマップを

指定し、必要に応じてメッセージング、ロギング、トレース、およびデータ・ハンドラー・パラメーターを指定する必要があります。

Connector Configurator の実行モードと使用する構成ファイルのタイプは、実行する統合ブローカーによって異なります。例えば、使用している統合ブローカーが WMQI の場合、Connector Configurator を System Manager から実行するのではなく、直接実行します (66 ページの『スタンドアロン・モードでの Configurator の実行』を参照)。

コネクタ構成プロパティには、標準の構成プロパティ (すべてのコネクタにもつプロパティ) と、コネクタ固有のプロパティ (特定のアプリケーションまたはテクノロジーのためにコネクタに必要なプロパティ) とが含まれます。

標準プロパティはすべてのコネクタにより使用されるので、標準プロパティを新規に定義する必要はありません。ファイルを作成すると、Connector Configurator により標準プロパティがこの構成ファイルに挿入されます。ただし、Connector Configurator で各標準プロパティの値を設定する必要があります。

標準プロパティの範囲は、ブローカーと構成によって異なる可能性があります。特定のプロパティに特定の値が設定されている場合にのみ使用できるプロパティがあります。Connector Configurator の「標準のプロパティ」ウィンドウには、特定の構成で設定可能なプロパティが表示されます。

ただし**コネクタ固有プロパティ**の場合は、最初にプロパティを定義し、その値を設定する必要があります。このため、特定のアダプターのコネクタ固有プロパティのテンプレートを作成します。システム内で既にテンプレートが作成されている場合には、作成されているテンプレートを使用します。システム内でまだテンプレートが作成されていない場合には、67 ページの『新規テンプレートの作成』のステップに従い、テンプレートを新規に作成します。

注: Connector Configurator は、Windows 環境内でのみ実行されます。UNIX 環境でコネクタを実行する場合には、Windows で Connector Configurator を使用して構成ファイルを変更し、このファイルを UNIX 環境へコピーします。

Connector Configurator の始動

以下の 2 種類のモードで Connector Configurator を開始および実行できます。

- スタンドアロン・モードで個別に実行
- System Manager から

スタンドアロン・モードでの Configurator の実行

どのブローカーを実行している場合にも、Connector Configurator を個別に実行し、コネクタ構成ファイルを編集できます。

これを行うには、以下のステップを実行します。

- 「スタート」>「プログラム」から、「**IBM WebSphere InterChange Server**」>「**IBM WebSphere Business Integration Toolset**」>「**開発**」>「**Connector Configurator**」をクリックします。
- 「ファイル」>「新規」>「構成ファイル」を選択します。

- 「システム接続: Integration Broker」の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、または WAS を選択します。

Connector Configurator を個別に実行して構成ファイルを生成してから、System Manager に接続してこの構成ファイルを System Manager プロジェクトに保存することもできます (72 ページの『構成ファイルの完成』を参照)。

System Manager からの Configurator の実行

System Manager から Connector Configurator を実行できます。

Connector Configurator を実行するには、以下のステップを実行します。

1. System Manager を開きます。
2. 「System Manager」ウィンドウで、「統合コンポーネント・ライブラリー」アイコンを展開し、「コネクタ」を強調表示します。
3. System Manager メニュー・バーから、「ツール」>「**Connector Configurator**」をクリックします。「Connector Configurator」ウィンドウが開き、「新規コネクタ」ダイアログ・ボックスが表示されます。
4. 「システム接続: Integration Broker」の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、または WAS を選択します。

既存の構成ファイルを編集するには、以下のステップを実行します。

1. 「System Manager」ウィンドウの「コネクタ」フォルダーでいずれかの構成ファイルを選択し、右クリックします。Connector Configurator が開き、この構成ファイルの統合ブローカー・タイプおよびファイル名が上部に表示されます。
2. 「標準のプロパティ」タブをクリックし、この構成ファイルに含まれているプロパティを確認します。

コネクタ固有のプロパティ・テンプレートの作成

コネクタの構成ファイルを作成するには、コネクタ固有プロパティのテンプレートとシステム提供の標準プロパティが必要です。

コネクタ固有プロパティのテンプレートを新規に作成するか、または既存のファイルをテンプレートとして使用します。

- テンプレートの新規作成については、67 ページの『新規テンプレートの作成』を参照してください。
- 既存のファイルを使用する場合には、既存のテンプレートを変更し、新しい名前でのこのテンプレートを保管します。

新規テンプレートの作成

このセクションでは、テンプレートでプロパティを作成し、プロパティの一般特性および値を定義し、プロパティ間の依存関係を指定する方法について説明します。次にそのテンプレートを保管し、新規コネクタ構成ファイルを作成するためのベースとして使用します。

テンプレートは以下のように作成します。

1. 「ファイル」>「新規」>「コネクタ固有プロパティ・テンプレート」をクリックします。
2. 以下のフィールドを含む「コネクタ固有プロパティ・テンプレート」ダイアログ・ボックスが表示されます。
 - 「テンプレート」、「名前」
このテンプレートが使用されるコネクタ（またはコネクタのタイプ）を表す固有の名前を入力します。テンプレートから新規構成ファイルを作成するためのダイアログ・ボックスを開くと、この名前が再度表示されます。
 - 「旧テンプレート」、「変更する既存のテンプレートを選択してください」
「テンプレート名」表示に、現在使用可能なすべてのテンプレートの名前が表示されます。
 - テンプレートに含まれているコネクタ固有のプロパティ定義を調べるには、「テンプレート名」表示でそのテンプレートの名前を選択します。そのテンプレートに含まれているプロパティ定義のリストが「テンプレートのプレビュー」表示に表示されます。テンプレートを作成するときには、ご使用のコネクタに必要なプロパティ定義に類似したプロパティ定義が含まれている既存のテンプレートを使用できます。
3. 「テンプレート名」表示からテンプレートを選択し、その名前を「名前の検索」フィールドに入力し（または「テンプレート名」で自分の選択項目を強調表示し）、「次へ」をクリックします。

ご使用のコネクタで使用するコネクタ固有のプロパティが表示されるテンプレートが見つからない場合は、自分で作成する必要があります。

一般特性の指定

「次へ」をクリックしてテンプレートを選択すると、「プロパティ: コネクタ固有プロパティ・テンプレート」ダイアログ・ボックスが表示されます。このダイアログ・ボックスには、定義済みプロパティの「一般」特性のタブと「値」の制限のタブがあります。「一般」表示には以下のフィールドがあります。

- **一般:**
 - プロパティ・タイプ
 - 更新されたメソッド
 - 説明
- **フラグ**
 - 標準フラグ
- **カスタム・フラグ**
 - フラグ

プロパティの一般特性の選択を終えたら、「値」タブをクリックします。

値の指定

「値」タブを使用すると、プロパティの最大長、最大複数值、デフォルト値、または値の範囲を設定できます。編集可能な値も許可されます。これを行うには、以下のステップを実行します。

1. 「値」タブをクリックします。「一般」のパネルに代わって「値」の表示パネルが表示されます。
2. 「プロパティを編集」表示でプロパティの名前を選択します。
3. 「最大長」および「最大複数値」のフィールドで、変更を行います。次のステップで説明するように、プロパティの「プロパティ値」ダイアログ・ボックスを開かない限り、そのプロパティの変更内容は受け入れられませんので、注意してください。
4. 値テーブルの左上の隅にあるボックスを右マウス・ボタンでクリックしてから、「追加」をクリックします。「プロパティ値」ダイアログ・ボックスが表示されます。このダイアログ・ボックスではプロパティのタイプに応じて、値だけを入力できる場合と、値と範囲の両方を入力できる場合があります。適切な値または範囲を入力し、「OK」をクリックします。
5. 「値」パネルが最新表示され、「最大長」および「最大複数値」で行った変更が表示されます。以下のような 3 つの列があるテーブルが表示されます。

「値」の列には、「プロパティ値」ダイアログ・ボックスで入力した値と、以前に作成した値が表示されます。

「デフォルト値」の列では、値のいずれかをデフォルトとして指定することができます。

「値の範囲」の列には、「プロパティ値」ダイアログ・ボックスで入力した範囲が表示されます。

値が作成されて、グリッドに表示されると、そのテーブルの表示内から編集できるようになります。テーブルにある既存の値の変更を行うには、その行の行番号をクリックして行全体を選択します。次に「値」フィールドを右マウス・ボタンでクリックし、「値の編集 (Edit Value)」をクリックします。

依存関係の設定

「一般」タブと「値」タブで変更を行ったら、「次へ」をクリックします。「依存関係: コネクター固有プロパティ・テンプレート」ダイアログ・ボックスが表示されます。

依存プロパティは、別のプロパティの値が特定の条件に合致する場合にのみ、テンプレートに組み込まれて、構成ファイルで使用されるプロパティです。例えば、テンプレートに PollQuantity が表示されるのは、トランスポート機構が JMS であり、DuplicateEventElimination が True に設定されている場合のみです。プロパティを依存プロパティとして指定し、依存する条件を設定するには、以下のステップを実行します。

1. 「使用可能なプロパティ」表示で、依存プロパティとして指定するプロパティを選択します。
2. 「プロパティを選択」フィールドで、ドロップダウン・メニューを使用して、条件値を持たせるプロパティを選択します。
3. 「条件演算子」フィールドで以下のいずれかを選択します。
 - == (等しい)
 - != (等しくない)
 - > (より大)
 - < (より小)

>= (より大か等しい)

<= (より小か等しい)

4. 「条件値」フィールドで、依存プロパティをテンプレートに組み込むために必要な値を入力します。
5. 「使用可能なプロパティ」表示で依存プロパティを強調表示させて矢印をクリックし、「依存プロパティ」表示に移動させます。
6. 「完了」をクリックします。Connector Configurator により、XML 文書として入力した情報が、Connector Configurator がインストールされている %bin ディレクトリーの %data%app の下に保管されます。

新しい構成ファイルの作成

構成ファイルを新規に作成するには、最初に統合ブローカーを選択します。選択したブローカーによって、構成ファイルに記述されるプロパティが決まります。

ブローカーを選択するには、以下のステップを実行します。

- Connector Configurator のホーム・メニューで、「ファイル」>「新規」>「コネクター構成」をクリックします。「新規コネクター」ダイアログ・ボックスが表示されます。
- 「Integration Broker」フィールドで、ICS 接続、WebSphere Message Brokers 接続、WAS 接続のいずれかを選択します。
- この章で後述する説明に従って「新規コネクター」ウィンドウの残りのフィールドに入力します。

また、以下の作業も実行できます。

- 「System Manager」ウィンドウで「コネクター」フォルダーを右クリックし、「新規コネクターの作成」を選択します。Connector Configurator が開き、「新規コネクター」ダイアログ・ボックスが表示されます。

コネクター固有のテンプレートからの構成ファイルの作成

コネクター固有のテンプレートを作成すると、テンプレートを使用して構成ファイルを作成できます。

1. 「ファイル」>「新規」>「コネクター構成」をクリックします。
2. 以下のフィールドを含む「新規コネクター」ダイアログ・ボックス表示されず。

- **名前**

コネクターの名前を入力します。名前では大文字と小文字が区別されます。入力する名前は、システムにインストールされているコネクターのファイル名に対応した一意の名前でなければなりません。

重要: Connector Configurator では、入力された名前のスペルはチェックされません。名前が正しいことを確認してください。

- **システム接続**

ICS 接続、WebSphere Message Brokers 接続、WAS のいずれかをクリックします。

- 「コネクタ固有プロパティ・テンプレート」を選択します。

ご使用のコネクタ用に設計したテンプレートの名前を入力します。「テンプレート名」表示に、使用可能なテンプレートが表示されます。「テンプレート名」表示で名前を選択すると、「プロパティ・テンプレートのプレビュー」表示に、そのテンプレートで定義されているコネクタ固有のプロパティが表示されます。

使用するテンプレートを選択し、「OK」をクリックします。

3. 構成しているコネクタの構成画面が表示されます。タイトル・バーに統合ブローカーとコネクタの名前が表示されます。ここですべてのフィールドに値を入力して定義を完了するか、ファイルを保管して後でフィールドに値を入力するかを選択できます。
4. ファイルを保管するには、「ファイル」>「保管」>「ファイルに」をクリックするか、「ファイル」>「保管」>「プロジェクトに」をクリックします。プロジェクトに保管するには、System Manager が実行中でなければなりません。ファイルとして保管する場合は、「ファイル・コネクタを保管」ダイアログ・ボックスが表示されます。*.cfg をファイル・タイプとして選択し、「ファイル名」フィールド内に名前が正しいスペル (大文字と小文字の区別を含む) で表示されていることを確認してから、ファイルを保管するディレクトリーにナビゲートし、「保管」をクリックします。Connector Configurator のメッセージ・パネルの状況表示に、構成ファイルが正常に作成されたことが示されます。

重要: ここで設定するディレクトリー・パスおよび名前は、コネクタの始動ファイルで指定するコネクタ構成ファイルのパスおよび名前に一致している必要があります。

5. この章で後述する手順に従って、「Connector Configurator」ウィンドウの各タブにあるフィールドに値を入力し、コネクタ定義を完了します。

既存ファイルの使用

使用可能な既存ファイルは、以下の 1 つまたは複数の形式になります。

- コネクタ定義ファイル。
コネクタ定義ファイルは、特定のコネクタのプロパティと、適用可能なデフォルト値がリストされたテキスト・ファイルです。コネクタの配布パッケージの `¥repository` ディレクトリー内には、このようなファイルが格納されていることがあります (通常、このファイルの拡張子は `.txt` です。例えば、XML コネクタの場合は `CN_XML.txt` です)。
- ICS リポジトリー・ファイル。
コネクタの以前の ICS インプリメンテーションで使用した定義は、そのコネクタの構成で使用されたりポジトリー・ファイルで使用可能になります。そのようなファイルの拡張子は、通常 `.in` または `.out` です。
- コネクタの以前の構成ファイル。
これらのファイルの拡張子は、通常 `*.cfg` です。

これらのいずれのファイル・ソースにも、コネクタのコネクタ固有プロパティのほとんど、あるいはすべてが含まれますが、この章内の後で説明するように、コネクタ構成ファイルは、ファイルを開いて、プロパティを設定しない限り完成しません。

既存ファイルを使用してコネクタを構成するには、Connector Configurator でそのファイルを開き、構成を修正し、そのファイルを再度保管する必要があります。

以下のステップを実行して、ディレクトリーから *.txt、*.cfg、または *.in ファイルを開きます。

1. Connector Configurator 内で、「ファイル」>「開く」>「ファイルから」をクリックします。
2. 「ファイル・コネクタを開く」ダイアログ・ボックス内で、以下のいずれかのファイル・タイプを選択して、使用可能なファイルを調べます。
 - 構成 (*.cfg)
 - ICS リポジトリ (*.in、*.out)
ICS 環境でのコネクタの構成にリポジトリ・ファイルが使用された場合には、このオプションを選択します。リポジトリ・ファイルに複数のコネクタ定義が含まれている場合は、ファイルを開くとすべての定義が表示されます。
 - すべてのファイル (*.*)
コネクタのアダプター・パッケージに *.txt ファイルが付属していた場合、または別の拡張子で定義ファイルが使用可能である場合は、このオプションを選択します。
3. ディレクトリー表示内で、適切なコネクタ定義ファイルへ移動し、ファイルを選択し、「開く」をクリックします。

System Manager プロジェクトからコネクタ構成を開くには、以下のステップを実行します。

1. System Manager を始動します。System Manager が開始されている場合にのみ、構成を System Manager から開いたり、System Manager に保管したりできます。
2. Connector Configurator を始動します。
3. 「ファイル」>「開く」>「プロジェクトから」をクリックします。

構成ファイルの完成

構成ファイルを開くか、プロジェクトからコネクタを開くと、「Connector Configurator」ウィンドウに構成画面が表示されます。この画面には、現在の属性と値が表示されます。

構成画面のタイトルには、ファイル内で指定された統合ブローカーとコネクタの名前が表示されます。正しいブローカーが設定されていることを確認してください。正しいブローカーが設定されていない場合、コネクタを構成する前にブローカー値を変更してください。これを行うには、以下のステップを実行します。

1. 「標準のプロパティ」タブで、BrokerType プロパティの値フィールドを選択します。ドロップダウン・メニューで、値 ICS、WMQI、または WAS を選択します。
2. 選択したブローカーに関連付けられているプロパティが「標準のプロパティ」タブに表示されます。ここでファイルを保管するか、または 75 ページの

『サポートされるビジネス・オブジェクト定義の指定』の説明に従い残りの構成フィールドに値を入力することができます。

3. 構成が完了したら、「ファイル」>「保管」>「プロジェクトに」を選択するか、または「ファイル」>「保管」>「ファイルに」を選択します。

ファイルに保管する場合は、*.cfg を拡張子として選択し、ファイルの正しい格納場所を選択して、「保管」をクリックします。

複数のコネクタ構成を開いている場合、構成をすべてファイルに保管するには「すべてファイルに保管」を選択し、コネクタ構成をすべて System Manager プロジェクトに保管するには「すべてプロジェクトに保管」をクリックします。

Connector Configurator では、ファイルを保管する前に、必須の標準プロパティすべてに値が設定されているかどうかを確認されます。必須の標準プロパティに値が設定されていない場合、Connector Configurator は、検証が失敗したというメッセージを表示します。構成ファイルを保管するには、そのプロパティの値を指定する必要があります。

構成ファイル・プロパティの設定

新規のコネクタ構成ファイルを作成して名前を付けるとき、または既存のコネクタ構成ファイルを開くときには、Connector Configurator によって構成画面が表示されます。構成画面には、必要な構成値のカテゴリに対応する複数のタブがあります。

Connector Configurator では、すべてのブローカーで実行されているコネクタで、以下のカテゴリのプロパティに値が設定されている必要があります。

- 標準のプロパティ
- コネクタ固有のプロパティ
- サポートされるビジネス・オブジェクト
- トレース/ログ・ファイルの値
- データ・ハンドラー (保証付きイベント・デリバリーで JMS メッセージングを使用するコネクタの場合に該当する)

注: JMS メッセージングを使用するコネクタの場合は、データをビジネス・オブジェクトに変換するデータ・ハンドラーの構成に関して追加のカテゴリが表示される場合があります。

ICS で実行されているコネクタの場合、以下のプロパティの値も設定されている必要があります。

- 関連付けられたマップ
- リソース
- メッセージング (該当する場合)

重要: Connector Configurator では、英語文字セットまたは英語以外の文字セットのいずれのプロパティ値も設定可能です。ただし、標準のプロパティおよびコネクタ固有プロパティ、およびサポートされるビジネス・オブジェクトの名前では、英語文字セットのみを使用する必要があります。

標準プロパティとコネクタ固有プロパティの違いは、以下のとおりです。

- コネクターの標準プロパティは、コネクターのアプリケーション固有のコンポーネントとブローカー・コンポーネントの両方によって共有されます。すべてのコネクターが同じ標準プロパティのセットを使用します。これらのプロパティの説明は、各アダプター・ガイドの付録 A にあります。変更できるのはこれらの値の一部のみです。
- アプリケーション固有のプロパティは、コネクターのアプリケーション固有コンポーネント (アプリケーションと直接対話するコンポーネント) のみに適用されます。各コネクターには、そのコネクターのアプリケーションだけで使用されるアプリケーション固有のプロパティがあります。これらのプロパティには、デフォルト値が用意されているものもあれば、そうでないものもあります。また、一部のデフォルト値は変更することができます。各アダプター・ガイドのインストールおよび構成の章に、アプリケーション固有のプロパティおよび推奨値が記述されています。

「標準プロパティ」と「コネクター固有プロパティ」のフィールドは、どのフィールドが構成可能であるかを示すために色分けされています。

- 背景がグレーのフィールドは、標準のプロパティを表します。値を変更することはできますが、名前の変更およびプロパティの除去はできません。
- 背景が白のフィールドは、アプリケーション固有のプロパティを表します。これらのプロパティは、アプリケーションまたはコネクターの特定のニーズによって異なります。値の変更も、これらのプロパティの除去も可能です。
- 「値」フィールドは構成できます。
- 「更新メソッド」フィールドは通知用であり、構成できません。このフィールドは、値が変更されたプロパティをアクティブにするために必要なアクションを示します。

標準コネクター・プロパティの設定

標準のプロパティの値を変更するには、以下のステップを実行します。

1. 値を設定するフィールド内でクリックします。
2. 値を入力するか、ドロップダウン・メニューが表示された場合にはメニューから値を選択します。
3. 標準のプロパティの値をすべて入力後、以下のいずれかを実行することができます。
 - 変更内容を破棄し、元の値を保持したままで Connector Configurator を終了するには、「ファイル」>「終了」をクリックし (またはウィンドウを閉じ)、変更内容を保管するかどうかを確認するプロンプトが出されたら「いいえ」をクリックします。
 - Connector Configurator 内の他のカテゴリーの値を入力するには、そのカテゴリーのタブを選択します。「標準のプロパティ」(またはその他のカテゴリー) で入力した値は、次のカテゴリーに移動しても保持されます。ウィンドウを閉じると、すべてのカテゴリーで入力した値を一括して保管するかまたは破棄するかを確認するプロンプトが出されます。
 - 修正した値を保管するには、「ファイル」>「終了」をクリックし (またはウィンドウを閉じ)、変更内容を保管するかどうかを確認するプロンプトが出されたら「はい」をクリックします。「ファイル」メニューまたはツールバーから「保管」>「ファイルに」をクリックする方法もあります。

アプリケーション固有の構成プロパティの設定

アプリケーション固有の構成プロパティの場合、プロパティ名の追加または変更、値の構成、プロパティの削除、およびプロパティの暗号化が可能です。プロパティのデフォルトの長さは 255 文字です。

1. グリッドの左上端の部分で右マウス・ボタンをクリックします。ポップアップ・メニュー・バーが表示されます。プロパティを追加するときは「追加」をクリックします。子プロパティを追加するには、親の行番号で右マウス・ボタンをクリックし、「子を追加」をクリックします。
2. プロパティまたは子プロパティの値を入力します。
3. プロパティを暗号化するには、「暗号化」ボックスを選択します。
4. 74 ページの『標準コネクタ・プロパティの設定』の説明に従い、変更内容を保管するかまたは破棄するかを選択します。

各プロパティごとに表示される「更新メソッド」は、変更された値をアクティブにするためにコンポーネントまたはエージェントの再始動が必要かどうかを示します。

重要: 事前設定のアプリケーション固有のコネクタ・プロパティ名を変更すると、コネクタに障害が発生する可能性があります。コネクタをアプリケーションに接続したり正常に実行したりするために、特定のプロパティ名が必要である場合があります。

コネクタ・プロパティの暗号化

「プロパティを編集」ウィンドウの「暗号化」チェック・ボックスにチェックマークを付けると、アプリケーション固有のプロパティを暗号化することができます。値の暗号化を解除するには、「暗号化」チェック・ボックスをクリックしてチェックマークを外し、「検証」ダイアログ・ボックスに正しい値を入力し、「OK」をクリックします。入力された値が正しい場合は、暗号化解除された値が表示されます。

各プロパティとそのデフォルト値のリストおよび説明は、各コネクタのアダプター・ユーザーズ・ガイドにあります。

プロパティに複数の値がある場合には、プロパティの最初の値に「暗号化」チェック・ボックスが表示されます。「暗号化」を選択すると、そのプロパティのすべての値が暗号化されます。プロパティの複数の値を暗号化解除するには、そのプロパティの最初の値の「暗号化」チェック・ボックスをクリックしてチェックマークを外してから、「検証」ダイアログ・ボックスで新規の値を入力します。入力値が一致すれば、すべての複数值が暗号化解除されます。

更新メソッド

付録『コネクタの標準構成プロパティ』の 48 ページの『プロパティ値の設定と更新』にある更新メソッドの説明を参照してください。

サポートされるビジネス・オブジェクト定義の指定

コネクタで使用するビジネス・オブジェクトを指定するには、Connector Configurator の「サポートされているビジネス・オブジェクト」タブを使用します。

汎用ビジネス・オブジェクトと、アプリケーション固有のビジネス・オブジェクトの両方を指定する必要があり、またそれらのビジネス・オブジェクト間のマップの関連を指定することが必要です。

注: コネクタによっては、アプリケーションでイベント通知や (メタオブジェクトを使用した) 追加の構成を実行するために、特定のビジネス・オブジェクトをサポートされているものとして指定することが必要な場合もあります。詳細は、「コネクタ開発ガイド (C++ 用)」または「コネクタ開発ガイド (Java 用)」を参照してください。

ご使用のブローカーが ICS の場合

ビジネス・オブジェクト定義がコネクタでサポートされることを指定する場合や、既存のビジネス・オブジェクト定義のサポート設定を変更する場合は、「サポートされているビジネス・オブジェクト」タブをクリックし、以下のフィールドを使用してください。

ビジネス・オブジェクト名: ビジネス・オブジェクト定義がコネクタによってサポートされることを指定するには、System Manager を実行し、以下のステップを実行します。

1. 「ビジネス・オブジェクト名」リストで空のフィールドをクリックします。
System Manager プロジェクトに存在するすべてのビジネス・オブジェクト定義を示すドロップダウン・リストが表示されます。
2. 追加するビジネス・オブジェクトをクリックします。
3. ビジネス・オブジェクトの「エージェント・サポート」(以下で説明) を設定します。
4. 「Connector Configurator」ウィンドウの「ファイル」メニューで、「プロジェクトに保管」をクリックします。追加したビジネス・オブジェクト定義に指定されたサポートを含む、変更されたコネクタ定義が、System Manager のプロジェクトに保管されます。

サポートされるリストからビジネス・オブジェクトを削除する場合は、以下のステップを実行します。

1. ビジネス・オブジェクト・フィールドを選択するため、そのビジネス・オブジェクトの左側の番号をクリックします。
2. 「Connector Configurator」ウィンドウの「編集」メニューから、「行を削除」をクリックします。リスト表示からビジネス・オブジェクトが除去されます。
3. 「ファイル」メニューから、「プロジェクトの保管」をクリックします。

サポートされるリストからビジネス・オブジェクトを削除すると、コネクタ定義が変更され、削除されたビジネス・オブジェクトはコネクタのこのインプリメンテーションで使用不可になります。コネクタのコードに影響したり、そのビジネス・オブジェクト定義そのものが System Manager から削除されることはありません。

エージェント・サポート: ビジネス・オブジェクトがエージェント・サポートを備えている場合、システムは、コネクタ・エージェントを介してアプリケーションにデータを配布する際にそのビジネス・オブジェクトの使用を試みます。

一般に、コネクターのアプリケーション固有ビジネス・オブジェクトは、そのコネクターのエージェントによってサポートされますが、汎用ビジネス・オブジェクトはサポートされません。

ビジネス・オブジェクトがコネクター・エージェントによってサポートされるよう指定するには、「エージェント・サポート」ボックスにチェックマークを付けます。「Connector Configurator」ウィンドウでは「エージェント・サポート」の選択の妥当性は検査されません。

最大トランザクション・レベル: コネクターの最大トランザクション・レベルは、そのコネクターがサポートする最大のトランザクション・レベルです。

ほとんどのコネクターの場合、選択可能な項目は「最大限の努力」のみです。

トランザクション・レベルの変更を有効にするには、サーバーを再始動する必要があります。

ご使用のブローカーが WebSphere Message Broker の場合

スタンドアロン・モードで作業している (System Manager に接続していない) 場合、手動でビジネス名を入力する必要があります。

System Manager を実行している場合、「サポートされているビジネス・オブジェクト」タブの「ビジネス・オブジェクト名」列の下にある空のボックスを選択できます。コンボ・ボックスが表示され、コネクターが属する統合コンポーネント・ライブラリー・プロジェクトから選択可能なビジネス・オブジェクトのリストが示されます。リストから必要なビジネス・オブジェクトを選択します。

「メッセージ・セット ID」は、WebSphere Business Integration Message Broker 5.0 のオプションのフィールドです。この ID が提供される場合、一意である必要はありません。ただし、WebSphere MQ Integrator および Integrator Broker 2.1 の場合は、一意の ID を提供する必要があります。

ご使用のブローカーが WAS の場合

使用するブローカー・タイプとして WebSphere Application Server を選択した場合、Connector Configurator にメッセージ・セット ID は必要ありません。「サポートされているビジネス・オブジェクト」タブには、サポートされるビジネス・オブジェクトの「ビジネス・オブジェクト名」列のみが表示されます。

スタンドアロン・モードで作業している (System Manager に接続していない) 場合、手動でビジネス・オブジェクト名を入力する必要があります。

System Manager を実行している場合、「サポートされているビジネス・オブジェクト」タブの「ビジネス・オブジェクト名」列の下にある空のボックスを選択できます。コンボ・ボックスが表示され、コネクターが属する統合コンポーネント・ライブラリー・プロジェクトから選択可能なビジネス・オブジェクトのリストが示されます。このリストから必要なビジネス・オブジェクトを選択します。

関連付けられているマップ (ICS のみ)

各コネクタは、現在 WebSphere InterChange Server でアクティブなビジネス・オブジェクト定義、およびそれらの関連付けられたマップのリストをサポートします。このリストは、「**関連付けられたマップ**」タブを選択すると表示されます。

ビジネス・オブジェクトのリストには、エージェントでサポートされるアプリケーション固有のビジネス・オブジェクトと、コントローラーがサブスクリプト・コラボレーションに送信する、対応する汎用オブジェクトが含まれます。マップの関連によって、アプリケーション固有のビジネス・オブジェクトを汎用ビジネス・オブジェクトに変換したり、汎用ビジネス・オブジェクトをアプリケーション固有のビジネス・オブジェクトに変換したりするときに、どのマップを使用するかが決定されます。

特定のソースおよび宛先ビジネス・オブジェクトについて一意的に定義されたマップを使用する場合、表示を開くと、マップは常にそれらの該当するビジネス・オブジェクトに関連付けられます。ユーザーがそれらを変更する必要はありません (変更できません)。

サポートされるビジネス・オブジェクトで使用可能なマップが複数ある場合は、そのビジネス・オブジェクトを、使用する必要のあるマップに明示的にバインドすることが必要になります。

「**関連付けられたマップ**」タブには以下のフィールドが表示されます。

- **ビジネス・オブジェクト名**

これらは、「**サポートされているビジネス・オブジェクト**」タブで指定した、このコネクタでサポートされるビジネス・オブジェクトです。「**サポートされているビジネス・オブジェクト**」タブでビジネス・オブジェクトを追加指定した場合、その内容は、「Connector Configurator」ウィンドウの「**ファイル**」メニューから「**プロジェクトに保管**」を選択して、変更を保管した後に、このリストに反映されます。

- **関連付けられたマップ**

この表示には、コネクタの、サポートされるビジネス・オブジェクトでの使用のためにシステムにインストールされたすべてのマップが示されます。各マップのソース・ビジネス・オブジェクトは、「**ビジネス・オブジェクト名**」表示でマップ名の左側に表示されます。

- **明示的**

場合によっては、関連付けられたマップを明示的にバインドすることが必要になります。

明示的バインディングが必要なのは、特定のサポートされるビジネス・オブジェクトに複数のマップが存在する場合のみです。ICS は、ブート時、各コネクタでサポートされるそれぞれのビジネス・オブジェクトにマップを自動的にバインドしようとします。複数のマップでその入力データとして同一のビジネス・オブジェクトが使用されている場合、サーバーは、他のマップのスーパーセットである 1 つのマップを見つけて、バインドしようとします。

他のマップのスーパーセットであるマップがないと、サーバーは、ビジネス・オブジェクトを単一のマップにバインドすることができないため、バインディングを明示的に設定することが必要になります。

以下のステップを実行して、マップを明示的にバインドします。

1. 「明示的 (Explicit)」列で、バインドするマップのチェック・ボックスにチェックマークを付けます。
2. ビジネス・オブジェクトに関連付けるマップを選択します。
3. 「Connector Configurator」ウィンドウの「ファイル」メニューで、「プロジェクトに保管」をクリックします。
4. プロジェクトを ICS に配置します。
5. 変更を有効にするため、サーバーをリブートします。

リソース (ICS)

「リソース」タブでは、コネクター・エージェントが、コネクター・エージェント並列処理を使用して同時に複数のプロセスを処理するかどうか、またどの程度処理するかを決定する値を設定できます。

すべてのコネクターがこの機能をサポートしていません。複数のプロセスを使用するよりも複数のスレッドを使用する方が通常は効率的であるため、Java でマルチスレッドとして設計されたコネクター・エージェントを実行している場合、この機能を使用することはお勧めできません。

メッセージング (ICS)

メッセージング・プロパティは、DeliveryTransport 標準プロパティの値として MQ を設定し、ブローカー・タイプとして ICS を設定した場合にのみ、使用可能です。これらのプロパティは、コネクターによるキューの使用方法に影響します。

トレース/ログ・ファイル値の設定

コネクター構成ファイルまたはコネクター定義ファイルを開くと、Connector Configurator は、そのファイルのログおよびトレースの値をデフォルト値として使用します。Connector Configurator 内でこれらの値を変更できます。

ログとトレースの値を変更するには、以下のステップを実行します。

1. 「トレース/ログ・ファイル」タブをクリックします。
2. ログとトレースのどちらでも、以下のいずれかまたは両方へのメッセージの書き込みを選択できます。
 - コンソールに (STDOUT):
ログ・メッセージまたはトレース・メッセージを STDOUT ディスプレイに書き込みます。

注: STDOUT オプションは、Windows プラットフォームで実行しているコネクターの「トレース/ログ・ファイル」タブでのみ使用できます。

- ファイルに:
ログ・メッセージまたはトレース・メッセージを指定されたファイルに書き込みます。ファイルを指定するには、ディレクトリー・ボタン (省略符号) をクリックし、指定する格納場所へ移動し、ファイル名を指定し、「保管」をクリックします。ログ・メッセージまたはトレース・メッセージは、指定した場所の指定したファイルに書き込まれます。

注: ログ・ファイルとトレース・ファイルはどちらも単純なテキスト・ファイルです。任意のファイル拡張子を使用してこれらのファイル名を設定できます。ただし、トレース・ファイルの場合、拡張子として `.trc` ではなく `.trace` を使用することをお勧めします。これは、システム内に存在する可能性がある他のファイルとの混同を避けるためです。ログ・ファイルの場合、通常使用されるファイル拡張子は `.log` および `.txt` です。

データ・ハンドラー

データ・ハンドラー・セクションの構成が使用可能となるのは、`DeliveryTransport` の値に `JMS` を、また `ContainerManagedEvents` の値に `JMS` を指定した場合のみです。すべてのアダプターでデータ・ハンドラーを使用できるわけではありません。

これらのプロパティーに使用する値については、付録 A の『コネクターの標準構成プロパティー』の `ContainerManagedEvents` の下の説明を参照してください。その他の詳細は、「コネクター開発ガイド (C++ 用)」または「コネクター開発ガイド (Java 用)」を参照してください。

構成ファイルの保管

コネクターの構成が完了したら、コネクター構成ファイルを保管します。Connector Configurator では、構成中に選択したブローカー・モードで構成ファイルを保管します。Connector Configurator のタイトル・バーには現在のブローカー・モード (ICS、WMQI、または WAS) が常に表示されます。

ファイルは XML 文書として保管されます。XML 文書は次の 3 通りの方法で保管できます。

- System Manager から統合コンポーネント・ライブラリーに `*.con` 拡張子付きファイルとして保管します。
- 指定したディレクトリーに保管します。
- スタンドアロン・モードで、ディレクトリー・フォルダーに `*.cfg` 拡張子付きファイルとして保管します。

System Manager でのプロジェクトの使用法、および配置の詳細については、以下のインプリメンテーション・ガイドを参照してください。

- ICS: 「*WebSphere InterChange Server* インプリメンテーション・ガイド」
- WebSphere Message Brokers: 「*WebSphere Message Brokers* 使用アダプター・インプリメンテーション・ガイド」
- WAS: 「アダプター実装ガイド (*WebSphere Application Server*)」

構成ファイルの変更

既存の構成ファイルの統合ブローカー設定を変更できます。これにより、他のブローカーで使用する構成ファイルを新規に作成するときに、このファイルをテンプレートとして使用できます。

注: 統合ブローカーを切り替える場合には、ブローカー・モード・プロパティーと同様に他の構成プロパティーも変更する必要があります。

既存の構成ファイルでのブローカーの選択を変更するには、以下のステップを実行します (オプション)。

- Connector Configurator で既存の構成ファイルを開きます。
- 「標準のプロパティ」タブを選択します。
- 「標準のプロパティ」タブの「BrokerType」フィールドで、ご使用のブローカーに合った値を選択します。
現行値を変更すると、プロパティ画面の利用可能なタブおよびフィールド選択がただちに更改され、選択した新規ブローカーに適したタブとフィールドのみが表示されます。

構成の完了

コネクタの構成ファイルを作成し、そのファイルを変更した後で、コネクタの始動時にコネクタが構成ファイルの位置を特定できるかどうかを確認してください。

これを行うには、コネクタが使用する始動ファイルを開き、コネクタ構成ファイルに使用されている格納場所とファイル名が、ファイルに対して指定した名前およびファイルを格納したディレクトリまたはパスと正確に一致しているかどうかを検証します。

グローバル化環境における Connector Configurator の使用

Connector Configurator はグローバル化され、構成ファイルと統合ブローカー間の文字変換を処理できます。Connector Configurator では、ネイティブなエンコード方式を使用しています。構成ファイルに書き込む場合は UTF-8 エンコード方式を使用します。

Connector Configurator は、以下の場所で英語以外の文字をサポートします。

- すべての値のフィールド
- ログ・ファイルおよびトレース・ファイル・パス (「トレース/ログ・ファイル」タブで指定)

CharacterEncoding および Locale 標準構成プロパティのドロップ・リストに表示されるのは、サポートされる値の一部のみです。ドロップ・リストに、サポートされる他の値を追加するには、製品ディレクトリの %Data%Std%stdConnProps.xml ファイルを手動で変更する必要があります。

例えば、Locale プロパティの値のリストにロケール en_GB を追加するには、stdConnProps.xml ファイルを開き、以下に太文字で示した行を追加してください。

```
<Property name="Locale"
isRequired="true"
updateMethod="component restart">
  <ValidType>String</ValidType>
  <ValidValues>
    <Value>ja_JP</Value>
    <Value>ko_KR</Value>
    <Value>zh_CN</Value>
    <Value>zh_TW</Value>
    <Value>fr_FR</Value>
    <Value>de_DE</Value>
    <Value>it_IT</Value>
```

```
<Value>es_ES</Value>
<Value>pt_BR</Value>
<Value>en_US</Value>
<Value>en_GB</Value>
    <DefaultValue>en_US</DefaultValue>
</ValidValues>
</Property>
```

付録 C. サンプル・シナリオ

この付録では、eMatrix の有効なサンプル・シナリオについて説明します。このサンプル・シナリオは、Adapter for eMatrix を使用して、eMatrix を、WebSphere Business Integration アダプターがサポートする他のアプリケーションおよびテクノロジーと統合する方法を示します。

この付録では、次のトピックについて説明します。

- 83 ページの『概要』
- 83 ページの『前提事項』
- 83 ページの『シナリオのインストール』
- 88 ページの『シナリオの実行』

概要

この単純な例では、eMatrix アダプターは Visual Test Connector と情報を交換します。Visual Test Connector がコンピューター・パーツ調達システムになっているものとします。パーツが利用可能になると、それがコンピューター・メーカーに伝達されます。コンピューター・メーカーは eMatrix アダプターを使用して自動的に eMatrix システムを更新します。コンピューター・メーカーのパーツの要件に変更があると、自動的にその変更が検出され、調達システムに送信されます。

前提事項

前提事項は以下のとおりです。

1. サポートされる統合ブローカーのいずれかをインストール済みで、その運用経験があること (詳細については第 2 章参照)。
2. eMatrix バージョン 9.6.0.1 およびそのコラボレーション・サーバーのいずれかをインストール済みで、その運用経験があること。

シナリオのインストール

最初に、eMatrix をセットアップし、eMatrix ビジネス・オブジェクト・タイプを作成します。

注: %SAMPLE_FOLDER% は、本書が格納されているフォルダーを参照します。
%WBIA% は、現行の WebSphere Business Integration Adapters がインストールされているフォルダーを参照します。

eMatrix ビジネス・オブジェクト・タイプの定義

最初に、コンピューター・システムのパーツを表すタイプを定義します。これを行うには、以下のステップを実行します。

1. eMatrix MQL シェルを開きます。
2. 以下のコマンドを、シェルにカット・アンド・ペーストします。

```

#BEGIN
#1. create the attributes

add attribute 'component cost'           type real;
add attribute 'in stock'                 type boolean;
add attribute 'date available'           type date;
add attribute 'component manufacturer'   type string;
add attribute megahertz                   type integer;
add attribute slot                        type integer;

#2. create the types

add type bundle;
add type computer;
add type cpu attribute 'component cost' attribute 'in stock' attribute
'date available'
attribute 'component manufacturer' attribute megahertz;
add type 'network card' attribute 'component cost' attribute 'in stock'
attribute 'date available' attribute 'component manufacturer';
add type 'usb 2.0 card' attribute 'component cost' attribute 'in stock'
attribute 'date available' attribute 'component manufacturer';
add type 'sound card' attribute 'component cost' attribute 'in stock'
attribute 'date available' attribute 'component manufacturer';
add type 'sound chip' attribute 'component cost' attribute 'in stock'
attribute 'date available' attribute 'component manufacturer'
attribute megahertz;

#3. create the relationships
add relationship 'bundle to computer' to type computer cardinality n from
type bundle cardinality n;
add relationship 'computer to pci component' to type 'network card',
'sound card', 'usb 2.0 card' cardinality n from type computer
cardinality 1 attribute slot;
add relationship 'sound card to sound chip' to type 'sound chip'
cardinality 1 from type 'sound card' cardinality 1;

#4. create policy

add policy 'computer manufacturing' type bundle, computer, cpu,
'network card', 'usb 2.0 card', 'sound card', 'sound chip' state
'pre production' state 'ready for production';

#5. create vault

add vault 'wbi computer manufacturer'

#6. create the 'adapter' person

add person adapter password wbia type business, system access all admin all;

#END

```

上述のステップ 6 では、システム内のすべての操作を実行する権限をユーザー 'adapter' に与えています。この例を実行するために、これらの特権がすべて必要というわけではありません。以下の MQL コマンドは、必要な特権をすべて提供します。

```

add person adapter password wbia type business access all
admin none;

```

ただし、これらのサンプルで取り上げられていない作業のために eMatrix アダプターを使用する場合、ステップ 6 で提供される特権が必要となることがあります。

eMatrix ビジネス・オブジェクト・タイプの作成

次に、仮定上のコンピューター・メーカーのコンピューター・システムを表すデータを作成します。これを行うには、以下のステップを実行します。

1. eMatrix MQL シェルを開きます。
2. 以下のコマンドを、シェルにカット・アンド・ペーストします。

```
#business objects
add bus bundle 'home office and entertainment' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer';
add bus computer 'gamer deluxe' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer';
add bus cpu 'max processor' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer'
  megahertz 2000
  'in stock' true
  'date available' 9/30/01
  'component manufacturer' acetech
  'component cost' 67;
add bus 'sound card' 'super sonic' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer'
  'in stock' false
  'date available' 6/12/03
  'component cost' 50
  'component manufacturer' 'feedback inc.';
add bus 'sound chip' 'super sound chip' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer'
  'component manufacturer' ChipsWeMake
  megahertz 100;
add bus 'usb 2.0 card' 'usb enabler' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer'
  'in stock' true
  'date available' 2/10/02
  'component manufacturer' usbworks
  'component cost' 12;
add bus 'network card' 'net connector' 1
  policy 'computer manufacturing'
  vault 'wbi computer manufacturer'
  'in stock' true
  'date available' 8/12/02
  'component cost' 10
  'component manufacturer' 'tcp specialists';
#relationships
connect bus bundle 'home office and entertainment' 1 relationship
'bundle to computer' to computer 'gamer deluxe' 1;
connect bus computer 'gamer deluxe' 1 relationship 'computer to pci component'
to 'sound card' 'super sonic' 1
slot 1;
connect bus computer 'gamer deluxe' 1 relationship 'computer to pci component'
to 'usb 2.0 card' 'usb enabler' 1
slot 2;
connect bus 'sound card' 'super sonic' 1 relationship 'sound card to
sound chip' to 'sound chip' 'super sound chip' 1;
```

JPO のインストール

2 つの JPO をインストールする必要があります。1 つは WBIA イベントを作成し、イベント・タイプを保存するためのもので、もう 1 つはイベント検出時にトリガーとして動作するものです。

1. eMatrix Business Modeler を開き、プログラムを作成する権限を持つユーザーとしてログインします。
2. 「オブジェクト」>「新規」>「プログラム (program)」を選択します。
3. 「名前」というタイトルのテキスト・ボックスに **WBIEventLogger** と入力します。
4. 「タイプ」は「Java」を選択します。
5. 「コード」タブをクリックします。
6. コードを
`%WBIA%\connector\%eMatrix\dependencies\WBIEventLogger.java` ファイルからこのタブのテキスト・ボックスへカット・アンド・ペーストします。
ソース・コードで `wbiPrefix` および `vault` の値の設定が必要な場合もあります。詳細については、コード中のコメントを調べてください。
7. 「作成」をクリックします。

同じステップを実行して、`InstallEventTables` プログラムを実行します。ただし以下の点が異なります。

- プログラムに `InstallEventTables` という名前を付けます。
- コードを `%WBIA%\connector\%eMatrix\dependencies\InstallEventTables.java` ファイルからカット・アンド・ペーストします。

トリガーのインストール

このセクションでは、仮定上のコンピューター設計に変更があったこと検出するトリガーを作成します。'computer to pci component' 関係においてトリガーを作成します。これを行うには、以下のステップを実行します。

1. eMatrix Business Modeler 内で、ウィンドウの左上の角にある双眼鏡をクリックします。
2. 「関係」を選択し、「検索」をクリックします。
3. 'computer to pci component' ビジネス・オブジェクトをダブルクリックします。
4. 新規ウィンドウで、「トリガー (Triggers)」タブを選択し、「追加」をクリックします。
5. トリガーの「作成」を選択し、「OK」をクリックします。
6. 新規ウィンドウで、「アクション」テキスト・ボックスに **WBIEventLogger** と入力します。
7. 「入力」テキスト・ボックスに次のように入力します。
`-method recordEvent ${FROMOBJECTID} n=wbi_computer_poll v=Update`
8. 「OK」をクリックしてから、「編集」をクリックして、トリガーの作成を終了します。

イベント・ビジネス・オブジェクトのインストール

シナリオのインストールの最後のステップは、イベント・ビジネス・オブジェクトのインストールです。これを行うためには、MQL コマンド行から次のコマンドを実行します。

```
exec program InstallEventTables;
```

eMatrix アダプターの構成

最初に、eMatrix Connector を構成します。これを行うには、以下のステップを実行します。

1. Connector Configurator を実行します。
2. %SAMPLE_FOLDER%\EmatrixConnector.cfg を開き、以下のプロパティを設定します。

```
RepositoryDirectory property = %SAMPLE_FOLDER%\repos  
PollFrequency property = key
```
3. ApplicationUserName および ApplicationPassword プロパティを、コネクタ実行の対象となるユーザーに設定します。
4. eMatrixServer プロパティを、接続先の eMatrix Server の名前に設定します。
5. HostName プロパティを、アダプターが通信する eMatrix コラボレーション・サーバーの URL (ポートを含む) に設定します。
6. EventVault プロパティを、イベントを保管するポートの名前に設定します。イベントを保管する新規のポートを作成することができます。これを行うためには、MQL eMatrix クライアントを実行し、ウィンドウ下部のテキスト・ボックスに 'add vault <vaultname>' と入力します。この値が WBIEventLogger JPO 内の **vault** スtringの値と一致することを確認してください。デフォルトでは、どちらの値も 'WBI_Events' となります。
7. KeepRelations プロパティを 'true' に設定します。
8. %SAMPLE_FOLDER%\PortConnector.cfg を開き、以下のプロパティを設定します。

```
RepositoryDirectory = %SAMPLE_FOLDER%\repos
```

ビジネス・オブジェクトのサポート

アダプターがビジネス・オブジェクトを使用するためには、まずビジネス・オブジェクトをサポートする必要があります。使用するビジネス・オブジェクトをサポートするには、以下のステップを実行します。

1. Connector Configurator が開いた状態で、「サポートされているビジネス・オブジェクト」タブを選択して、以下のビジネス・オブジェクトを追加します。
 - wbi_computer
 - wbi_computer_poll
2. 各ビジネス・オブジェクトの横にある「サポートされているエージェント (Agent supported)」ボックスにチェックマークを付けます。
3. ポート・コネクタで上述のステップを繰り返します。

キューの作成

このサンプル・シナリオでは、ご使用のキュー・マネージャー内でいくつかのキューを定義する必要があります。必要なキューを作成するには、コマンド行から

```
RUNMQSC crosswor1ds.queue.manager
```

と入力し、以下のコマンドを発行します。

```
DEFINE QL('ADMININQUEUE')
DEFINE QL('ADMINOUTQUEUE')
DEFINE QL('DELIVERYQUEUE')
DEFINE QL('FAULTQUEUE')
DEFINE QL('REQUESTQUEUE')
DEFINE QL('RESPONSEQUEUE')
DEFINE QL('EMATRIXCONNECTOR/RESPONSEQUEUE')
DEFINE QL('SYNCHRONOUSREQUESTQUEUE')
DEFINE QL('SYNCHRONOUSRESPONSEQUEUE')
DEFINE QL('PORTCONNECTOR/SYNCHRONOUSREQUESTQUEUE')
DEFINE QL('PORTCONNECTOR/SYNCHRONOUSRESPONSEQUEUE')
```

start_eMatrix.bat または start_eMatrix.sh ファイル (プラットフォームによって異なる) を開き、これらのファイル中で EMADK および EM_LIB 変数を設定して、始動スクリプトを更新します。

シナリオの実行

この例は、要求処理およびイベント通知という 2 つのアクティビティについてシナリオを提供します。

シナリオ 1: 要求処理

要求処理のシナリオでは、設計で使用されるコンポーネントの 1 つが利用可能になったことを、調達システムが eMatrix システムに通知します。システムは、更新されたシステム・コンポーネントを示すビジネス・オブジェクトを送信します。

Visual Test Connector の始動

1. 「スタート」>「プログラム」>「WebSphere Business Integration Adapters」>「ツール」>「Visual Test Connector」を選択します。コネクタが始動します。
2. コネクタ・ウィンドウで、「ファイル」>「プロファイルを作成/選択」を選択します。「コネクタ・プロファイル」ウィンドウが表示されます。
3. まだプロファイルを作成していなければ、以下のようにして、プロファイル・ウィンドウでポート・コネクタをエミュレートするプロファイルを作成します。
 - 「ファイル」>「新規プロファイル」を選択します。
 - 「参照」をクリックし、%SAMPLE_FOLDER%\PortConnector.cfg ファイルを選択します。
 - 「コネクタ名」テキスト・ボックスに、Port Connector と入力します。「ブローカー・タイプ」ドロップダウン・メニューから、WMQI を選択します。
 - 「OK」をクリックします。
4. すでにプロファイルを作成済みの場合は、それを選択して、「OK」をクリックします。
5. プロファイルが準備されたら、「ファイル」>「接続」を選択します。

Adapter for eMatrix の始動

次のコマンドを実行して、Adapter for eMatrix を始動します。

```
%WBIA%\connectors\eMatrix\start_eMatrix.bat eMatrix wmqi  
-c%SAMPLE_FOLDER%\EmatrixConnector.cfg
```

“wmqi” の値を、英数字の任意の組み合わせ (先頭は英字) で置き換えることができます。

eMatrix コンポーネントの始動

eMatrix コラボレーション・サーバーを始動します。eMatrix システムが存在する Oracle のインスタンスが稼働していることを確認します。次に、「MQL コマンド (MQL Command)」ウィンドウを開きます。

eMatrix の現在の状況のチェック

MQL コマンド行で、次のコマンドを入力します。

```
expand bus 'computer' 'gamer deluxe' 1 select bus  
'attribute[in stock]';
```

超音波サウンド・カードは在庫がないことに注意してください。

要求の送信

これを行うには、以下のステップを実行します。

1. Visual Test Connector で、「編集」>「ビジネス・オブジェクトをロード」を選択します。
2. ファイル %SAMPLE_FOLDER%\sound_card_update.bo を選択します。
3. ポップアップ・ウィンドウに任意の名前を入力し、「OK」をクリックします。
4. Visual Test Connector の左上隅の「ビジネス・オブジェクトを送信」をクリックします。アイコンには、黄色の矢印が 1 つ含まれています。
Adapter for eMatrix を含むウィンドウに、テキスト・メッセージが表示されます。これらのメッセージは、要求の処理中であることを示します。

eMatrix の更新された状況のチェック

MQL コマンド行で、次のように入力します。

```
expand bus 'computer' 'gamer deluxe' 1 select bus  
'attribute[in stock]';
```

ここで超音波サウンド・カードの在庫があることに注意してください。

データベース内の状況のリセット

このシナリオを再度実行するには、超音波サウンド・カードをリセットして、在庫がないようにする必要があります。これを行うには、次の MQL コマンドを実行します。

```
modify bus 'sound card' 'super sonic' 1 'in stock' false;
```

シナリオ 2: イベント通知

このシナリオでは、コンピューター・メーカーが、コンピューター・システムの 1 つの構成を変更します。この変更がトリガーとなって、調達システムへ通知が転送されます。

1. 前述のシナリオ 1 の、ステップ 1 から 3 までを繰り返します。
2. 以下のステップに従ってください。

コンピューター構成の更新

コンピューター構成が変更されました。コンピューター・メーカーはコンピューターにネットワーク・カードを追加します。MQL コマンド行で、次のコマンドを実行します。

```
connect bus 'computer' 'gamer deluxe' 1 relationship  
'computer to pci component' to 'network card' 'net  
connector' 1 slot 3;
```

イベントのポーリング

Adapter for eMatrix を含むコマンド・ウィンドウに移動します。ウィンドウの下部で、'p' と入力します。これにより、アダプターがポーリングを行います。

イベントの表示

アダプターは、新規のコンピューター構成を記録し、調達システム (この場合は Visual Test Connector) に転送します。

Visual Test Connector の右側のウィンドウ・ペインを調べます。テキスト 'wbi_computer.Update' を含む行が表示されています。これは、WebSphere ビジネス・オブジェクトとして表される新規のコンピューター構成です。ここでダブルクリックすると、その内容を見ることができます。

コンピューター構成のリセット

このシナリオを再度実行するには、eMatrix ビジネス・オブジェクトを切断する必要があります。これを行うには、次のコマンドを実行します。

```
disconnect bus 'computer' 'gamer deluxe' 1 relationship  
'computer to pci component' to 'network card'  
'net connector' 1;
```

これにより、サンプル・シナリオのインストールおよびテストが完了します。上述のすべてのステップが正常に実行されると、Adapter for eMatrix を使用可能にする適切なサンプルができます。

特記事項

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-0032
東京都港区六本木 3-2-31
IBM World Trade Asia Corporation
Licensing

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM RTP Laboratory

3039 Cornwallis Road
P.O. BOX 12195
Raleigh, NC 27709-2195
U.S.A

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができませんが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

プログラミング・インターフェース情報

プログラミング・インターフェース情報は、プログラムを使用してアプリケーション・ソフトウェアを作成する際に役立ちます。

一般使用プログラミング・インターフェースにより、お客様はこのプログラム・ツール・サービスを含むアプリケーション・ソフトウェアを書くことができます。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

警告: 診断、修正、調整情報は、変更される場合がありますので、プログラミング・インターフェースとしては使用しないでください。

商標

以下は、IBM Corporation の商標です。

IBM
IBM ロゴ
AIX
CrossWorlds
DB2
DB2 Universal Database
MQIntegrator
MQSeries
Tivoli
WebSphere

Domino、Lotus、Lotus Notes および Notes Mail は、IBM Corporation の商標です。

Microsoft、Windows、Windows NT および Windows ロゴは、Microsoft Corporation の米国およびその他の国における商標です。

MMX、Pentium および ProShare は、Intel Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。



IBM WebSphere Business Integration Adapter Framework V2.4.0