IBM WebSphere Business Integration

Adapter for WebSphere Message Broker ユーザーズ・ガイド

Adapter バージョン 2.8.x

IBM WebSphere Business Integration

Adapter for WebSphere Message Broker ユーザーズ・ガイド

Adapter バージョン 2.8.x

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、 129 ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM WebSphere Business Integration Adapter for WebSphere Message Broker (5724-H37) バージョン 2.8.x 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
Adapter for WebSphere Message Broker User Guide
Adapter Version 2.8.x

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 2005.12

この文書では、平成明朝体™W3、平成明朝体™W7、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™W7、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

目次

本書について

IBM® WebSphere® Business Integration Adapter ポートフォリオは、優れた e-business テクノロジー、エンタープライズ・アプリケーション、レガシー・システム、およびメインフレーム・システムへの統合コネクティビティーを提供します。この製品セットには、ビジネス統合の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やテンプレートが含まれています。

本書の内容

本書では、この IBM WebSphere Business Integration アダプターのインストール、コネクター・プロパティー構成、ビジネス・オブジェクト開発、およびトラブルシューティングについて説明します。

本書に含まれないもの

本書では、サーバーのロード・バランシング、アダプター処理スレッドの数、最大および最小スループット、許容度しきい値などのデプロイメント・メトリックやキャパシティー・プランニングの問題は扱いません。

このような問題は、それぞれのお客様の開発に固有のものであり、アダプターが配置される実際の環境内またはそれに近い環境で測定される必要があります。お客様の構成は固有のものですので、配置サイトの構成や、この種のメトリックの計画と評価の詳細については、担当の IBM 技術員に連絡してください。

対象読者

本書は、お客様のサイトで WebSphere Business Integration 製品のサポートおよび管理を担当するコンサルタント、開発者、およびシステム管理者を対象としています。

本書の前提条件

本書の読者は、WebSphere Business Integration システム、ビジネス・オブジェクトとコラボレーションの開発、および WebSphere Integration Message Broker アプリケーションについて十分な知識と経験を持っている必要があります。

関連文書

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

関連資料は以下のサイトからインストールできます。

- アダプターの一般情報については、アダプターを WebSphere Message Broker (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、WebSphere

Business Integration Message Broker) と共に使用する場合、およびアダプターを WebSphere Application Server と共に使用する場合は、以下の IBM WebSphere Business Integration Adapters インフォメーション・センターを参照してください。 http://www.ibm.com/websphere/integration/wbiadapters/infocenter

- アダプターを WebSphere InterChange Server と共に使用する場合は、以下の IBM WebSphere InterChange Server インフォメーション・センターを参照してください。 http://www.ibm.com/websphere/integration/wicserver/infocenter
  http://www.ibm.com/websphere/integration/wbicollaborations/infocenter
- WebSphere メッセージ・ブローカーの詳細については、以下のサイト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、以下のサイトを参照してください。
  http://www.ibm.com/software/webservers/appserv/library.html

これらのサイトでは、資料のダウンロード、インストール、および表示方法を簡単に説明しています。

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。

`Courier フォント`	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。
太字	初出語を示します。
イタリック、*イタリック*	変数名または相互参照を示します。
青のアウトライン	オンラインで表示したときに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ことにより、参照先オブジェクトにジャンプできます。
{}	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、大括弧 [] で囲まれた部分は、オプション・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、`option[,...]` 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。
< >	命名規則では、不等号括弧は名前の個々の要素を囲み、各要素を区別します (例: `<server_name><*connector_name*>tmp.log`)。
/, ¥	本書では、ディレクトリー・パスの規則として円記号 (¥) を使用します。UNIX(R) 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。すべての製品のパス名は、ご使用のシステムで製品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 *$text*	% 記号で囲まれたテキストは、Windows(R) の text システム変数またはユーザー変数の値を示します。UNIX 環境での同等の表記は *$text* であり、UNIX の *text* 環境変数の値を示します。
ProductDir	製品のインストール先ディレクトリーを表します。

本リリースの新機能

リリース 2.8.x での新機能

このリリースでは、AIX 5.3 プラットフォームに対するサポートが追加されました。それについての詳細およびその他のハードウェアおよびソフトウェア要件の詳細は、http://www.ibm.com/support/docview.wss?uid=swg27006249 を参照してください。この URL で提供されている情報は、今回のリリースで本書から削除された『ブローカーの互換性』および『アダプターのプラットフォーム』セクションの情報に替わるものです。

アダプターは、IBM Tivoli License Manager (ITLM) をサポートするようになりました。

アダプターは、イベント・ポーリングで複数のスレッドを使用できるようになりました。スレッドの最大数は、Connector Configurator の WorkerThreadCount プロパティーで指定します。詳細については、 106 ページの『WorkerThreadCount』を参照してください。

コネクター固有の新規プロパティー DataHandlerPoolSize で、各データ・ハンドラー・タイプのインスタンス・プールを作成することができます。この機能拡張により、アダプターは、後で再利用するために構成済みデータ・ハンドラーのインスタンスをプールできます。詳細については、 104 ページの『DataHandlerPoolSize』を参照してください。

コネクター固有の 2 つの新規プロパティー SecurityExitClassName および SecurityExitInitParam を使用して、カスタム・セキュリティー出口を呼び出すようにアダプターを構成できるようになりました。詳細については、 106 ページの『SecurityExitClassName』および 106 ページの『SecurityExitInitParam』を参照してください。

テキストをサポートするアダプターのデータ・エンコードが、バイナリーと JMS オブジェクト・メッセージ・タイプもサポートするようになりました。メタオブジェクトの DataEncoding プロパティーで、テキスト (デフォルトのデータ・エンコード)、バイナリー、またはオブジェクトを指定します。詳細については、 58 ページの『バイナリー・メッセージとオブジェクト・メッセージの DataEncoding』を参照してください。

アダプターは、Windows 環境で動作しているときに、アラビア語とヘブライ語の双方向 (bi-di) スクリプト・データの処理を制限付きでサポートするようになりました。詳細については、 12 ページの『ロケール依存データ』を参照してください。

リリース 2.7.x での新機能

2004 年 9 月更新。アダプター、バージョン 2.7.x では、次の新規情報または訂正情報が本書に追加されました。

このリリースでは、次のプラットフォームおよびプラットフォーム・アップデートに対するサポートが追加されました。

- 2004 年 2 月 11 日以降の Solaris Patch Cluster を備えた Solaris 8 (2.8)
- 2004 年 2 月 11 日以降の Solaris Patch Cluster を備えた Solaris 9 (2.9)。このアダプターは、64 ビットのプラットフォームで 32 ビットの JVM をサポートします。
- 保守レベル 4 の AIX (R) 5.1
- 保守レベル 1 の AIX 5.2。このアダプターは、64 ビットのプラットフォームで 32 ビットの JVM をサポートします。
- Service Pack 4 を備えた Microsoft(R) Windows 2000 (Professional、Server、または Advanced Server)
- Microsoft Windows 2003 (Standard Edition または Enterprise Edition)
- Linux Red Hat AS 3.0 with Update 1、ES 3.0 with Update 1、および WS 3.0 with Update 1

  注: WebSphere Business Integration Adapter Framework V2.6 の Tivoli(R) Monitoring for Transaction Performance (TMTP) コンポーネントは、Linux Red Hat ではサポートされません。
- SUSE Linux Standard Server x86 8.1 with SP3 および Enterprise Server x86 8.1 with SP3
- HP-UX 11.i (11.11) with June 2003 GOLDBASE11i および June 2003 GOLDAPPS11i バンドル
- カスタム・アダプターのコンパイル用の Java™ コンパイラー IBM JDK 1.4.2 for Windows 2000

コネクター固有のプロパティー ReplyToQueuePollFrequency は文書化されています。

このリリースでは、アダプターによってキャッチされた例外で printStackTrace() をダンプするトレース・レベル 5 の使用をサポートします。

リリース 2.6.x での新機能

2 つのコネクター固有のプロパティー EnableMessageProducerCache および SessionPoolSizeForRequests が追加されました。詳細については、101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』を参照してください。

アダプターの名称が Adapter for WebSphere MQ Integrator Broker から Adapter for WebSphere Integration Message Broker へ変更されました。

バージョン 2.6.x から、アダプターは Solaris 7 でサポートされなくなりました。そのため、このプラットフォーム・バージョンに関する記述は本書から削除されました。

リリース 2.5.x での新機能

コネクターは、以下のプラットフォーム上で実行されます。

- Microsoft Windows 2000
- Solaris 7、8 または AIX 5.1、5.2 または HP UX 11.i

バージョン 2.5.0 からは、Adapter for WebSphere MQ は Microsoft Windows NT(R) ではサポートされなくなりました。

アダプターのインストール情報は本書から移動しました。この情報の新規掲載場所については 2 章を参照してください。

リリース 2.4.x での新機能

アダプターは、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。それについての詳細およびその他のハードウェアおよびソフトウェア要件の詳細は、http://www.ibm.com/support/docview.wss?uid=swg27006249 を参照してください。

コネクターは、以下のプラットフォーム上で実行されます。

- Microsoft Windows NT 4.0 Service Pack 6A または Windows 2000
- Solaris 7、8 または AIX 5.1、5.2 または HP UX 11.i

リリース 2.3.x での新機能

2003 年 3 月更新。「CrossWorlds(R)」という名前は、現在ではシステム全体を表したり、コンポーネント名やツール名を修飾するためには使用されなくなりました。コンポーネント名およびツール名自体は、以前とほとんど変わりません。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。

データ・ハンドラーを入力キューに関連付けることができるようになりました。詳細については、 60 ページの『データ・ハンドラーの入力キューへのマッピングの概要』を参照してください。

アダプターは、WebSphere MQ Integrator Broker と WebSphere InterChange Server の統合ブローカーのほかに、WebSphere MQ Event Broker をサポートするようになりました。

保証付きイベント・デリバリー機能が拡張されました。詳細については、 40 ページの『イベント・デリバリーの保証』を参照してください。

リリース 2.2.x での新機能

InProgress キューは不要になったため、使用不可にすることができます。

コネクターは、MQSeries(R) 5.1、5.2、および 5.3 を介したアプリケーションとのインターオペラビリティーをサポートします。詳細については、 12 ページの『アダプターの依存関係』を参照してください。

このリリースのコネクターには、ビジネス・オブジェクト処理のため UseDefaults プロパティーが用意されています。詳細については、 106 ページの『UseDefaults』を参照してください。

コネクターは、データ・ハンドラーがビジネス・オブジェクトに明示的にデフォルトの動詞を割り当てない場合、デフォルトの動詞を適用できるようになりました。詳細については、 104 ページの『DefaultVerb』を参照してください。

ReplyToQueue は、ReplyToQueue コネクター・プロパティーではなく、動的子メタオブジェクトを介して指示されるようになりました。詳細については、 63 ページの『JMS ヘッダーと動的子メタオブジェクトの属性』を参照してください。

メッセージ選択子を使用して識別やフィルター操作を行うほか、アダプターが特定の要求に対して応答メッセージを識別する方法を制御できます。この JMS 機能は同期要求処理にのみ適用されます。詳細については、 6 ページの『同期送達』を参照してください。

リリース 2.1.x での新機能

コネクターは国際化されました。詳細については、 1 ページの『作業ロードマップ』と 73 ページの『付録 A. コネクターの標準構成プロパティー』を参照してください。

本書では、このアダプターを InterChange Server と共に使用するための情報を提供します。

注: 保証付きイベント・デリバリー機能を使用するには、InterChange Server のリリース 4.1.1.2 をインストールする必要があります。

リリース 1.5.x での新機能

IBM WebSphere Business Integration Adapter for MQ Integrator には、MQ Integrator 用のコネクターが含まれます。このアダプターは、WebSphere InterChange Server (ICS) 統合ブローカーと共に動作します。統合ブローカーとは、異種のアプリケーション・セット間の統合を実行するアプリケーションです。統合ブローカーは、データ・ルーティングなどのサービスを提供します。アダプターには、以下の要素が含まれます。

- MQ Integrator に固有のアプリケーション・コンポーネント
- サンプル・ビジネス・オブジェクト
- IBM WebSphere Adapter Framework。コンポーネントは以下のとおりです。
  - コネクター・フレームワーク
  - 開発ツール (Business Object Designer と IBM CrossWorlds System Manager を含む)
  - API (CDK を含む)

このマニュアルでは、このアダプターを InterChange Server と共に使用するための情報を提供します。

重要: コネクターは国際化に対応していないため、ISO Latin-1 データのみが処理されることが確実である場合を除いて、コネクターと InterChange Server バージョン 4.1.1 を併用しないでください。

IBM CrossWorlds 4.1.x システムでコネクターが使用可能になりました。

リリース 1.4.x での新機能

本書の 1.4.x リリースには、欠陥を修正し IBM CrossWorlds インフラストラクチャー・リリース・バージョン 4.0.0 との互換性を保証するために、小さい変更が加えられています。

リリース 1.3.x での新機能

本書の 1.3.x リリースでは、以下の新機能と製品機能強化が追加されています。

- Create、Update、および Delete 操作を確認するための同期要求/応答処理のサポート。
- Retrieve、Retrieve By Content、および Exist 操作のサポート。
- 正常に処理されたメッセージ、アンサブスクライブされたメッセージ、およびエラーが発生したメッセージを含む、メッセージの全アーカイブ。
- 同じメッセージ・フォーマットを複数のビジネス・オブジェクトに割り当てる拡張機能。
- コネクターは、完全修飾 URI なしでローカル・キューを識別することが可能になりました。したがって、`InputQueueURI`、`InProgressQueueURI`、`UnsubscribedURI`、および `ErrorQueueURI` の各コネクター・プロパティーには、「URI」 サフィックスが含まれなくなりました。
- コネクター・メタオブジェクトにデフォルトの変換プロパティーが追加されました。したがって、コネクター・プロパティー `DefaultOutputQueueURI` が除去されました。

第 1 章 概要

この章では、知っておく必要がある用語について説明し、アダプターでの処理について解説して、全体像を示します。アダプターをインストール、構成、および使用する前にアダプターについて理解しておくことが重要です。

作業ロードマップ

Adapter for WebSphere Business Integration Message Broker を使用するには、表 1 に記載されている作業を行う必要があります。

表 1. アダプターの使用: 作業ロードマップ

作業	関連手順 (参照先)	詳細情報 (参照先)
コネクターのインストール	11 ページの『第 2 章 コネクターのインストール』	*WebSphere Business Integration Adapters のインストール*
ビジネス・オブジェクトとメタオブジェクトの構成	49 ページの『第 5 章 オブジェクトの作成』	「*ビジネス・オブジェクト開発ガイド*」
データ・ハンドラーの構成	67 ページの『第 6 章 データ・ハンドラーの構成』	「*データ・ハンドラー・ガイド*」
コネクターの構成	17 ページの『第 3 章 コネクターの構成』	73 ページの『付録 A. コネクターの標準構成プロパティー』、101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』、および「*コネクター開発ガイド*」
コネクターの実行	41 ページの『第 4 章 コネクターの実行』	
コネクターのトラブルシューティング	71 ページの『第 7 章 トラブルシューティング』	
チュートリアルの実行	107 ページの『付録 C. チュートリアル』	

用語

アダプターについて理解するには、以下の用語を理解する必要があります。

アダプター

統合ブローカーとアプリケーション (またはテクノロジー) との間の通信をサポートするコンポーネントを備えている、WebSphere Business Integration システムのコンポーネント。アダプターには、コネクター、メッセージ・ファイル、および構成ツールが必ず含まれています。また、Object Discovery

Agent (ODA) も含まれていることがあります。また、一部のアダプターではデータ・ハンドラーが必要な場合があります。

アダプター・フレームワーク
IBM が提供する、アダプターの構成と実行のためのソフトウェア。アダプター・フレームワークには、ランタイム・コンポーネントとして、Java ランタイム環境、コネクター・フレームワーク、および Object Discovery Agent (ODA) ランタイムが組み込まれています。コネクター・フレームワークには、コネクターを新規開発するときに必要となるコネクター・ライブラリー (C++ および Java) が含まれています。ODA ランタイムには、ODA を新規開発するときに必要となる Object Development Kit (ODK) のライブラリーが含まれています。構成用コンポーネントとしては、以下のツールが用意されています。

- Business Object Designer
- Connector Configurator
- Log Viewer
- System Manager
- Adapter Monitor
- Test Connector
- アダプターに関連付けられた Object Discovery Agent (ODA) (用意されていない場合もあります)

Adapter Development Kit (ADK)
アダプター開発用のサンプルをいくつか備えた開発キット。サンプルには、コネクターと Object Discovery Agent (ODA) のサンプルも含まれます。

コネクター
ビジネス・オブジェクトを使用して、統合ブローカーにイベント関連の情報を送信し (イベント通知)、統合ブローカーから要求関連の情報を受信する (要求処理)、アダプターのコンポーネント。コネクターは、コネクター・フレームワークと、コネクターのアプリケーション固有のコンポーネントで構成されています。

コネクター・フレームワーク
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統合ブローカーの間の相互作用を管理する、コネクターのコンポーネント。このコンポーネントは、必要な管理サービスをすべて備えており、コネクターが必要とするメタデータをリポジトリーから取得します。コネクター・フレームワークは Java で記述されており、C++ で記述されたアプリケーション固有のコンポーネントをサポートできるように C++ 拡張が組み込まれています。コネクター・フレームワークのコードはすべてのコネクターで共通です。

コネクター・コントローラー
コラボレーションと相互作用する、コネクター・フレームワークのサブコンポーネント。コネクター・コントローラーは、InterChange Server 内で動作するもので、アプリケーション固有のビジネス・オブジェクトと汎用ビジネス・オブジェクトの間のマッピングを開始し、コラボレーションのビジネス・オブジェクト定義に対するサブスクリプションを管理します。

統合ブローカー

異種のアプリケーションの間でデータを統合する、WebSphere Business Integration システムのコンポーネント。通常、統合ブローカーはさまざまなサービスを備えています。このサービスには、データをルーティングする機能、統合プロセスを決定する規則のリポジトリー、各種アプリケーションに接続する機能、および統合を容易にする管理機能が含まれます。統合ブローカーには、WebSphere Business Integration Message Broker や WebSphere Business InterChange Server などがあります。

WebSphere Business Integration システム

多様なソースの間で情報を移動してビジネス関連の情報を交換し、エンタープライズ環境内の異種のアプリケーションの間で情報の処理とルーティングを行う、エンタープライズ・ソリューション。このビジネス・インテグレーション・システムは、統合ブローカーと 1 つ以上のアダプターで構成されています。

WebSphere Business Integration Message Broker バージョン 2.2

WebSphere MQ キュー間でメッセージの変換とルーティングを行う、メッセージ・ブローカー製品。このテクノロジーにより、アプリケーションは、リモート・キューとの間でメッセージを送受信して非同期的に通信することができます。WebSphere Integration Message Broker での大きな変更点は、ユーザー定義のロジックに基づいてメッセージをフォーマット、格納、およびルーティングできる機能を追加するメッセージ・フローが追加されたことです。

アダプター環境の概要

図 1 に、WebSphere Business Integration システムでのアダプター、関連コンポーネント、およびこれらの間の関係を示します。この図は標準的な構成を示しています。アダプターはレガシー・アプリケーションとメッセージを交換するように構成されており、レガシー・アプリケーションのメッセージは WebSphere Business Integration Message Broker を通じて送受信されます。また、アダプターは、InterChange Server とビジネス・オブジェクトを交換するようにも構成されています。コネクターはメタデータ主導型です。メッセージのルーティングおよびフォーマット変換は、イベント・ポーリング技法によって開始されます。コネクターは、Java™ Message Service (JMS) の MQ インプリメンテーションを使用します。JMS は、エンタープライズ・メッセージング・システムにアクセスするための API です。

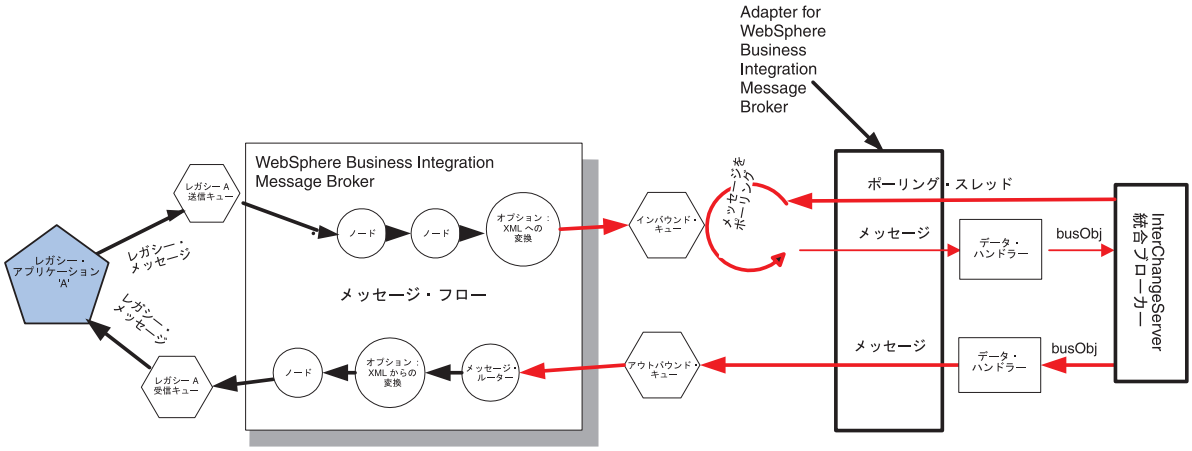

図 1. WebSphere Business Integration 環境でのアダプター

コネクターを使用すると、コラボレーションと、データの変更が発生したときに WebSphere MQ メッセージを送受信するアプリケーションとの間で、非同期的にビジネス・オブジェクトを交換できます。

コネクターはキューから WebSphere MQ メッセージを検索し、データ・ハンドラーを呼び出してメッセージを対応するビジネス・オブジェクトに変換し、コラボレーションにデリバリーします。反対方向の場合、コネクターはコラボレーションからビジネス・オブジェクトを受け取り、同じデータ・ハンドラーを使用して WebSphere MQ メッセージに変換し、WebSphere MQ キューにデリバリーします。

推奨: コネクターは、任意のデータ・ハンドラーを使用してメッセージを処理するように構成できます。ただし、WebSphere Integration Message Broker は構文解析可能なすべてのメッセージをオプションで XML 形式に変換できるため、すべてのメッセージを XML 形式でデリバリーするようにコネクターを構成することを強くお勧めします。これは、処理のために XML データ・ハンドラーを実装することを意味します。概要および手順については、67 ページの『データ・ハンドラーの構成の概要』を参照してください。

アダプターでの処理の概要

アダプターは、Java Message Service (JMS) の IBM による WebSphere MQ インプリメンテーションを使用します。 JMS は、エンタープライズ・メッセージング・システムにアクセスするためのオープン・スタンダード API です。JMS は、ビジネス・アプリケーションがデータとイベントを非同期的に送受信できるように設計されています。

アダプターは、WebSphere Business Integration Message Broker と直接相互作用しません。コネクターを構成する際には、WebSphere Business Integration Message Broker メッセージ・フローの入力ノードおよび出力ノードとして WebSphere MQ キューをセットアップします。アダプターは、このメッセージ・フローが配置されているメッセージ・ブローカーをホスティングしている WebSphere MQ キュー・マネージャーと通信します。

メッセージ要求

図 2 に、アダプターのランタイム・コンポーネントであるコネクターが行うメッセージ要求の通信を示します。doVerbFor() メソッドがコラボレーションからビジネス・オブジェクトを受け取ると、コネクターはそのビジネス・オブジェクトをデータ・ハンドラーに渡します。データ・ハンドラーはそのビジネス・オブジェクトを適したメッセージに変換し、キューに送ります。このとき、JMS 層は適切な呼び出しを実行してキュー・セッションを開き、メッセージの経路を指定します。コネクターは、要求を非同期的に送出する (応答不要送信を行う) ように構成できます。また、要求を同期的に処理できるようにコネクター・プロパティーを構成することもできます。

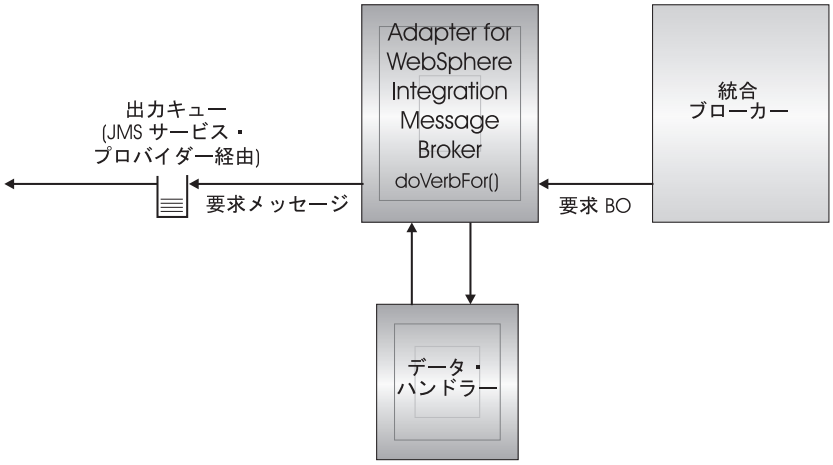

図 2. メッセージ要求の処理

コネクターは、コラボレーションから渡されたビジネス・オブジェクトを、各ビジネス・オブジェクトの動詞に基づいて処理します。コネクターはビジネス・オブジェクト・ハンドラーと doVerbFor() メソッドを使用して、コネクターがサポートするビジネス・オブジェクトを処理します。コネクターは、以下のビジネス・オブジェクトの動詞をサポートします。

- Create
- Update
- Delete
- Retrieve
- Exists
- Retrieve by Content

注: Create 動詞、Update 動詞、および Delete 動詞を持つビジネス・オブジェクトは、非同期的にも同期的にも送信できます。デフォルト・モードは非同期送信です。コネクターは、Retrieve 動詞、Exists 動詞、Retrieve by Content 動詞のビジネス・オブジェクトの非同期送信をサポートしません。したがって、Retrieve 動詞、Exists 動詞、または Retrieve by Content 動詞のデフォルト・モードは同期送信です。

Create、Update、および Delete

Create 動詞、Update 動詞、および Delete 動詞を持つビジネス・オブジェクトの処理は、ビジネス・オブジェクトが非同期的に送信されたか同期的に送信されたかによって決まります。

非同期送達: これは、Create 動詞、Update 動詞、および Delete 動詞を持つビジネス・オブジェクトのデフォルト送信モードです。データ・ハンドラーを使用して、ビジネス・オブジェクトからメッセージが作成され、出力キューに書き込まれます。メッセージがデリバリーされると、コネクターは SUCCESS を戻します。それ以外の場合は FAIL を戻します。

注: コネクターには、メッセージが受信されたかどうか、または、処置が行われたかどうかを確認する方法はありません。

同期送達: コネクター固有のプロパティーで ReplyToQueue が定義されており、かつビジネス・オブジェクトのメタオブジェクト変換プロパティーに responseTimeout が存在する場合、コネクターは同期モードで要求を送信します。続いて、コネクターは、受信側のアプリケーションで適切な処置が行われたかどうかを確認するために応答を待ちます。

WebSphere Integration Message Broker の場合、コネクターは最初に、表 2 に示すヘッダーを持つメッセージを発行します。

表 2. 要求メッセージ記述子ヘッダー (MQMD)

フィールド	説明	値
Format	フォーマット名	メタオブジェクト変換プロパティーに定義されている出力フォーマット。IBM の要件に合わせて、8 文字を超える部分が切り捨てられます (例 : MQSTR)
MsgType	メッセージ・タイプ	MQMT_DATAGRAM*
Report	要求されたレポート・メッセージのオプション	応答メッセージの返送が予測される場合、このフィールドには次の値が取り込まれます。処理が成功したときに肯定処理レポートが必要な場合は、MQRO_PAN*。処理が失敗したときに否定処理レポートが必要な場合は、MQRO_NAN*。生成されるレポートの相関 ID が最初に発行された要求のメッセージ ID と同じになる必要がある場合は、MQRO_COPY_MSG_ID_TO_CORREL_ID*。
ReplyToQ	応答キューの名前	応答メッセージの返送が予測される場合、このフィールドにはコネクター・プロパティー ReplyToQueue の値が取り込まれます。
Persistence	メッセージのパーシスタンス	MQPER_PERSISTENT*
Expiry	メッセージの存続時間	MQEI_UNLIMITED*

* は、IBM によって定義される定数を示します。

表 2 に示すメッセージ・ヘッダーの後に、メッセージ本文が続きます。メッセージの本文は、データ・ハンドラーを使用して直列化されたビジネス・オブジェクトです。

Report フィールドは、受信側アプリケーションから肯定処理レポートと否定処理レポートの両方の返送が予測されることを示すために設定されます。メッセージを発行したスレッドは、受信側アプリケーションが要求を処理できたかどうかを示す応答メッセージを待ちます。

アプリケーションがコネクターから同期要求を受け取ると、アプリケーションはビジネス・オブジェクトを処理し、表 3、4、および 5 に示すレポート・メッセージを発行します。

表 3. 応答メッセージ記述子ヘッダー (MQMD)

フィールド	説明	値
Format	フォーマット名	変換プロパティー内で定義された busObj の入力フォーマット
MsgType	メッセージ・タイプ	MQMT_REPORT*

* は、IBM によって定義される定数を示します。

表 4. 応答メッセージに含まれるデータ

動詞	Feedback フィールド	メッセージの本文
Create、Update、または Delete	SUCCESS VALCHANGE	(オプション) 変更を反映する、直列化されたビジネス・オブジェクト。
	VALDUPES FAIL	(オプション) エラー・メッセージ。

表 5. WebSphere Integration Message Broker フィードバック・コードと WebSphere Business Integration システムの応答値

WebSphere Integration Message Broker フィードバック・コード	対応する WebSphere Business Integration システムの応答*
MQFB_PAN または MQFB_APPL_FIRST	SUCCESS
MQFB_NAN または MQFB_APPL_FIRST + 1	FAIL
MQFB_APPL_FIRST + 2	VALCHANGE
MQFB_APPL_FIRST + 3	VALDUPES
MQFB_APPL_FIRST + 4	MULTIPLE_HITS
MQFB_APPL_FIRST + 5	FAIL_RETRIEVE_BY_CONTENT
MQFB_APPL_FIRST + 6	BO_DOES_NOT_EXIST
MQFB_APPL_FIRST + 7	UNABLE_TO_LOGIN (この応答後、コネクター・エージェントは即時に終了します)
MQFB_APPL_FIRST + 8	APP_RESPONSE_TIMEOUT

* 詳細については、「*コネクター開発ガイド*」を参照してください。

ビジネス・オブジェクトを処理できる場合、アプリケーションは、feedback フィールドが MQFB_PAN (または特定の WebSphere Business Integration システムの値) に設定されたレポート・メッセージを作成します。また、オプションで、アプリケーションはすべての変更を含む直列化されたビジネス・オブジェクトをメッセージ本文に取り込みます。ビジネス・オブジェクトを処理できない場合、アプリケーションは、feedback フィールドが MQFB_NAN (または特定の WebSphere Business Integration システムの値) に設定されたレポート・メッセージを作成します。オプションで、このレポート・メッセージの本文にエラー・メッセージを含めることもできます。どちらの場合でも、アプリケーションはメッセージの correlationID フィールドを

コネクター・メッセージの messageID に設定し、ReplyToQueue フィールドで指定されたキューにメッセージを発行します。

コネクターは、応答メッセージを取り出すと、デフォルトにより応答の correlationID を要求メッセージの messageID と突き合わせます。続いて、要求を発行したスレッドに通知を送信します。コネクターは、応答の feedback フィールドの設定によって、メッセージの本文にビジネス・オブジェクトとエラー・メッセージのどちらが含まれているかを予測します。ビジネス・オブジェクトが含まれていると予測したにもかかわらず、メッセージの本文にビジネス・オブジェクトが取り込まれていなかった場合、コネクターは InterChange Server が要求操作のために最初に発行したのと同じビジネス・オブジェクトを単純に返送します。エラー・メッセージが含まれていると予測したにもかかわらず、メッセージの本文にエラー・メッセージが取り込まれていなかった場合、InterChange Server には応答コードと汎用エラー・メッセージが返送されます。ただし、メッセージ選択子を使用して識別やフィルター操作を行うこともできます。あるいは、アダプターが特定の要求に対して応答メッセージを識別する方法を制御できます。このメッセージ選択子機能は JMS 機能です。この JMS 機能は同期要求処理にのみ適用されます。以下に詳細を説明します。

Retrieve、Exists、および Retrieve by content: Retrieve 動詞、Exists 動詞、および Retrieve By Content 動詞を持つビジネス・オブジェクトは、同期送信のみをサポートします。コネクターは、これらの動詞を持つビジネス・オブジェクトを、Create 動詞、Update 動詞、および Delete 動詞に対して定義されている同期送信と同様に処理します。ただし、Retrieve 動詞、Exists 動詞、および Retrieve By Content 動詞を使用する場合には、responseTimeout と replyToQueue が必要です。さらに、Retrieve By Content 動詞と Retrieve 動詞の場合、トランザクションを完了するためにはメッセージの本文に直列化されたビジネス・オブジェクトが取り込まれている必要があります。

表 6 に、これらの動詞に対応する応答メッセージを示します。

表 6. 応答メッセージへの取り込み

動詞	Feedback フィールド	メッセージの本文
Retrieve または RetrieveByContent	FAIL FAIL_RETRIEVE_BY_CONTENT	(オプション) エラー・メッセージ。
	MULTIPLE_HITS SUCCESS	直列化されたビジネス・オブジェクト。
Exist	FAIL	(オプション) エラー・メッセージ。
	SUCCESS	

イベント処理

図 3 に、コネクターでのイベント処理を示します。pollForEvents() メソッドは、次の該当するメッセージを入力キューから検索します。メッセージは実行中のキューに入れられ、処理が完了するまでキュー内に残ります。コネクターは最初に、コネクター・メタオブジェクトを使用して、そのメッセージ・タイプがサポートされているかどうかを調べます。サポートされている場合、コネクターは構成されているデータ・ハンドラーにメッセージを渡し、データ・ハンドラーがそのメッセージ

をビジネス・オブジェクトに変換します。設定される動詞には、そのメッセージ・タイプに対して定義されている変換プロパティーが反映されます。次に、コネクターは、そのビジネス・オブジェクトがコラボレーションによってサブスクライブされているかどうかを調べます。サブスクライブされている場合、gotApplEvents() メソッドがビジネス・オブジェクトを統合ブローカーにデリバリーし、実行中のキューからメッセージが削除されます。

コネクターは、イベント通知のために、データベース・トリガーではなくアプリケーションによってキューに書き込まれたイベントを検出します。イベントは、アプリケーションまたはその他の MQ 対応ソフトウェアが WebSphere MQ メッセージを生成して MQ メッセージ・キューに格納するときに発生します。

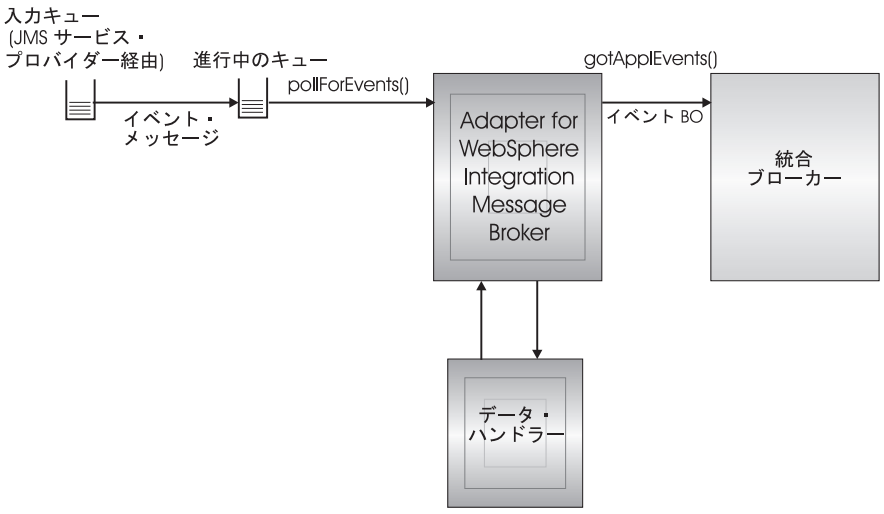

図 3. アプリケーションとコネクターの間の通信方法: メッセージ戻り

検索

コネクターは、pollForEvents() メソッドを使用して MQ キューからメッセージを定期的にポーリングします。メッセージを検出すると、コネクターはそれを MQ キューから検索して調べ、メッセージのフォーマットを判別します。フォーマットがコネクター・メタオブジェクト内で定義されている場合、コネクターはデータ・ハンドラーを使用して適切な動詞付きのビジネス・オブジェクトを生成します。イベント失敗のシナリオについては、45 ページの『エラー処理の概要』を参照してください。

コネクターは、最初に入力キューとのトランザクション・セッションを開いて、メッセージを処理します。このトランザクション・アプローチを使用すると、コネクターがビジネス・オブジェクトを正常にサブミットしたにもかかわらず、キューでトランザクションをコミットできなかった場合に、コラボレーションにビジネス・オブジェクトが 2 回デリバリーされるという可能性が少し残ります。この問題を回避するために、コネクターは、すべてのメッセージを進行中キューに移動します。メッセージは、処理が完了するまでキュー内に残ります。処理中にコネクターが予期しないエラーでシャットダウンした場合、メッセージは元の入力キューには戻されず、実行中のキュー内に残ります。

注: JMS サービス・プロバイダーとのトランザクション・セッションでは、キュー上の要求されたすべての処理が、キューからイベントが削除される前に実行され、コミットされる必要があります。したがって、コネクターがキューからメッセージを検索するときには、次の 3 つの処理が実行されるまでは検索がコミットされません。1) メッセージからビジネス・オブジェクトへの変換、2) `gotApplEvents()` メソッドによる、InterChange Server へのビジネス・オブジェクトのデリバリー、および 3) 戻り値の受信。

リカバリー

コネクターは初期化の際に実行中のキューを調べ、コネクターのシャットダウンが原因で未処理のまま残っているメッセージがないかどうかを調べます。コネクターの構成プロパティー `InDoubtEvents` を使用すると、そのようなメッセージのリカバリー処理に関する 4 つのオプション (fail on startup、reprocess、ignore、または log error) のうち、いずれかを指定できます。

Fail on startup: fail on startup オプションを指定した場合、コネクターが初期化の際に実行中のキュー内でメッセージを検出すると、コネクターはエラーを記録し、即時にシャットダウンします。ユーザーまたはシステム管理者は、検出されたメッセージを調べ、これらのメッセージを完全に削除するかまたは別のキューに移動するなどの適切な処置を取る必要があります。

Reprocess: reprocessing オプションを指定した場合、コネクターが初期化の際に実行中のキュー内でメッセージを検出すると、コネクターは以降のポーリングでそのメッセージを最初に処理します。実行中のキュー内にあったすべてのメッセージの処理が完了すると、コネクターは入力キューからのメッセージの処理を開始します。

Ignore: ignore オプションを指定した場合、初期化の際にコネクターが実行中のキュー内でメッセージを検出すると、コネクターはそれを無視しますが、シャットダウンはしません。

Log error: log error オプションを指定した場合、初期化の際にコネクターが実行中のキュー内でメッセージを検出すると、コネクターはエラーを記録しますが、シャットダウンはしません。

アーカイブ

コネクター固有のプロパティー `ArchiveQueue` が指定されており、かつ有効なキューを示している場合には、コネクターは正常に処理されたすべてのメッセージのコピーをアーカイブ・キューに格納します。 `ArchiveQueue` が未定義の場合、メッセージは処理後に破棄されます。アンサブスクライブされたメッセージまたはエラーを含むメッセージのアーカイブの詳細については、45 ページの『エラー処理の概要』を参照してください。

注: JMS 規則により、検索したメッセージを即時に別のキューに送信することはできません。メッセージをアーカイブして再デリバリーできるようにするために、コネクターは、オリジナルのメッセージから本文とヘッダー (該当する場合のみ) を複製した第 2 のメッセージを最初に生成します。JMS サービス・プロバイダーとの競合を避けるため、JMS に必須フィールドのみが複製されます。したがって、format フィールドは、アーカイブまたは再デリバリーされるメッセージにコピーされる唯一の追加メッセージ・プロパティーとなります。

第 2 章 コネクターのインストール

この章では、コネクターのインストール方法および構成方法と、メッセージ・フローをコネクターとともに動作させるための構成方法について説明します。

インストール作業の概要

Adapter for WebSphere Integration Message Broker をインストールするには、以下の作業を行う必要があります。

- **統合ブローカーのインストール**: この作業では、WebSphere Business Integration システムのインストールと統合ブローカーの始動を行います。作業の詳細については、使用するブローカーおよびオペレーティング・システムのインストール文書に説明があります。ブローカーおよびインストールの詳細については、http://www.ibm.com/support/docview.wss?uid=swg27006249 を参照してください。
- **アダプターおよび関連ファイルのインストール**: この作業では、アダプター用のファイルをソフトウェア・パッケージから使用システムにインストールします。13 ページの『アダプターと関連ファイルのインストール』を参照してください。
- **アダプターとアダプター・フレームワークのみのインストール**: 詳細については、次のサイトの WebSphere Business Integration Adapters インフォメーション・センターにある「*WebSphere Business Integration Adapters のインストール*」のガイドを参照してください。
  http://www.ibm.com/websphere/integration/wbiadapters/infocenter

アダプターをインストールする前に、アダプター環境について理解しておく必要があります。詳細については、 12 ページの『アダプター環境』を参照してください。

この章で説明する作業

この章で説明する作業は、以下のとおりです。

表 7. アダプターのインストール: 作業ロードマップ

作業	関連手順 (参照先)	詳細情報 (参照先)
アダプターのインストール	13 ページの『アダプターと関連ファイルのインストール』	*WebSphere Business Integration Adapters のインストール*
インストールの検証	14 ページの『インストールの検証』	

アダプター環境

アダプターをインストール、構成、および使用する前に、アダプターの環境要件を理解しておく必要があります。

- 『前提条件』
- 『アダプターの依存関係』
- 『ロケール依存データ』
- 13 ページの『Common Event Infrastructure』
- 13 ページの『Application Response Measurement』

前提条件

ブローカーの互換性やサポートされるプラットフォームなどについては、http://www.ibm.com/support/docview.wss?uid=swg27006249 を参照してください。

アダプターの依存関係

アダプターには次の依存関係があります。

- コネクターは、WebSphere MQ または WebSphere MQ 5.1、5.2[1]、および 5.3 を介したアプリケーションとのインターオペラビリティーをサポートします。したがって、これらのいずれかのソフトウェア・リリースをインストールする必要があります。

  注: このアダプターは、WebSphere MQ 5.3 環境で SSL (Secure Socket Layer) をサポートしていません。アダプター・フレームワークと統合ブローカーの通信にとって適切な WebSphere MQ ソフトウェア・バージョンについては、使用プラットフォーム (Windows または UNIX) のインストール・ガイドを参照してください。
- IBM WebSphere MQ Java クライアント・ライブラリー (Java および JMS 用の WebSphere MQ クラス) バージョン 5.x が必要です。

ロケール依存データ

コネクターは、2 バイト文字セットをサポートし、指定された言語でメッセージ・テキストを送達できるように国際化されています。コネクターは、1 つの文字コードを使用する場所から別のコード・セットを使用する場所にデータを転送するとき、データの意味を保存するように文字変換を実行します。

このアダプターは、Windows 環境で動作しているときに、アラビア語とヘブライ語の双方向 (bi-di) スクリプト・データの処理をサポートします。双方向処理は、Windows 以外の環境ではサポートされません。双方向機能を使用するには、双方向標準プロパティーを構成する必要があります。詳細については、 73 ページの『付録 A. コネクターの標準構成プロパティー』でコネクターの標準構成プロパティーを参照してください。

1. ご使用の環境に「get 時の変換」方式の文字セット変換が実装されている場合、IBM から最新の MA88 (JMS クラス) をダウンロードする必要があります。パッチ・レベルは最低でも 5.2.2 である必要があります (WebSphere MQ バージョン 5.2 の場合)。これにより、サポートされないエンコード・エラーを避けることができます。

Java 仮想マシン (JVM) 内での Java ランタイム環境は、Unicode 文字コード・セットでデータを表します。 Unicode には、ほとんどの既知の文字コード・セット (1 バイト系とマルチバイト系をいずれも含む) に対応できるエンコード方式が組み込まれています。WebSphere Business Integration システムのほとんどのコンポーネントは Java で記述されています。したがって、ほとんどのインテグレーション・コンポーネントの間でデータが転送されても、文字変換の必要はありません。

エラー・メッセージと通知メッセージを適切な言語で記録するには、該当する環境の `Locale` 標準構成プロパティーを設定します。構成プロパティーの詳細については、73 ページの『付録 A. コネクターの標準構成プロパティー』を参照してください。

Common Event Infrastructure

このアダプターは、その他の IBM WebSphere イベント生成アプリケーションとのインターオペラビリティーを可能にするイベント管理の標準である、IBM の Common Event Infrastructure との互換性があります。Common Event Infrastructure サポートが使用可能である場合は、アダプターによって生成されるイベントを、別の Common Event Infrastructure 互換アプリケーションが受信 (または使用) できます。

詳細については、本書の付録『Application Response Measurement』を参照してください。

Application Response Measurement

このアダプターは、アプリケーションの可用性、サービス・レベル・アグリーメント、およびキャパシティー・プランニングを管理するための Application Response Measurement (ARM) アプリケーション・プログラミング・インターフェース (API) に対応しています。ARM を備えたアプリケーションは、IBM Tivoli(R) Monitoring for Transaction Performance に参加して、トランザクション・メトリックに関するデータの収集と検討を実行できます。

詳細については、本書の付録『Application Response Measurement』を参照してください。

アダプターと関連ファイルのインストール

始める前に: インストールするアダプターに関して、要件、依存関係、およびブローカーの互換性を検討します。詳細については、12 ページの『アダプター環境』を参照してください。

WebSphere Business Integration Adapter 製品のインストールについては、「*WebSphere Business Integration Adapters のインストール*」のガイドを参照してください。この資料は、次の Web サイトの WebSphere Business Integration Adapters インフォメーション・センターにあります。

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

インストールの検証

以下のセクションでは、インストール後の製品のパスとファイル名およびアダプターのインストールの検証方法について説明します。

Windows システムでのインストールの検証の概要

始める前に: アダプターをインストールします。インストーラーは、アダプターに関連付けられた標準ファイルをご使用のシステムにコピーします。このユーティリティーは、コネクターを *ProductDir*¥connectors¥WebSphereBIMessageBroker (WBIMB) ディレクトリーにインストールし、そのコネクターへのショートカットを「スタート」メニューに追加します。

Windows システムでのインストールの検証のステップ

Windows システムでアダプターのインストールを検証するときは、次のステップを実行します。

- アダプターをインストールしたディレクトリー (*ProductDir*¥) に移動し、そのディレクトリーの内容を、表 8 に記載されている内容と比較します。

表 8 で、アダプターにより使用される Windows ファイル構造について説明し、インストーラーによるアダプターのインストールを選択したときに自動的にインストールされるファイルを示します。ただし、WebSphereBIMessageBrokerConnectorTemplate ファイルは repository¥WebSphereBIMessageBroker ディレクトリーに自動的にインストールされません。

表 8. アダプター用にインストールされた Windows ファイル構造

***ProductDir* のサブディレクトリー**	**説明**
connectors¥WebSphereBIMessageBroker¥CWWebSphereBIMessageBroker.jar	WebSphere Business Integration Message Broker コネクター・エージェントによってのみ使用されるクラスを含みます。
connectors¥WebSphereBIMessageBroker¥start_WebSphereBIMessageBroker.bat	コネクターの始動スクリプト
connectors¥messages¥WebSphereBIMessageBrokerConnector.txt	コネクターのメッセージ・ファイル
bin¥Data¥App¥WebSphereBIMessageBrokerConnectorTemplate	アダプター定義のテンプレート・ファイル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥WebSphereBIMessageBrokerConnector.cfg	WBIMB アダプター構成ファイルのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥PortConnector.cfg	ポート・コネクター構成ファイルのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥Sample_WebSphereBIMessageBroker_LegacyItem.xsd	サンプル・ビジネス・オブジェクト・リポジトリー・ファイル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥ Sample_WebSphereBIMessageBroker_LegacyItem_XMLDoc.xsd	サンプル・ビジネス・オブジェクト・リポジトリー・ファイル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥ Sample_WebSphereBIMessageBroker_MO_Config.xsd	コネクター・エージェント・メタオブジェクトのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥ Sample_WebSphereBIMessageBroker_MO_DataHandler.xsd	データ・ハンドラー用のトップレベル・メタオブジェクトのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥ Sample_WebSphereBIMessageBroker_MO_DataHandler_XMLConfig.xsd	XML データ・ハンドラーの構成に使用されるメタオブジェクトのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥LegacyItem.txt	サンプルを使用するための入力ファイルのサンプル

表 8. アダプター用にインストールされた Windows ファイル構造 (続き)

ProductDir のサブディレクトリー	説明
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥mqsiput.exe	サンプルとともに使用する WebSphere MQ 用のユーティリティー
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥Sample_WebSphereBIMessageBroker_Project¥.project connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥Sample_WebSphereBIMessageBroker_Project¥LocalDomain.configmgr connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥Sample_WBIMB_Project¥Sample_WebSphereBIMessageBroker_bar.bar	WBIMB プロジェクトのサンプル
connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥.project connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥ AdjustFormat.esql connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥ Check_Color.esql connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥ From_LegacyApplication_to_WBI.msgflow connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥ From_WBI_to_LegacyApplication.msgflow connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem ¥MSG_FLOW_RPOJECT¥ Loopback.msgflow	メッセージ・フロー・プロジェクトのサンプル

注: すべての製品のパス名は、ご使用のシステムで製品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。

UNIX システムでのインストールの検証の概要

始める前に: アダプターをインストールします。インストーラーは、アダプターに関連付けられた標準ファイルをご使用のシステムにコピーします。このユーティリティーは、コネクター・エージェントを *ProductDir*/connectors/WBIMB ディレクトリーにインストールします。

UNIX システムでのインストールの検証のステップ

UNIX システムでアダプターのインストールを検証するときは、次のステップを実行します。

- アダプターをインストールしたディレクトリー (*ProductDirl*) に移動し、そのディレクトリーの内容を、表 9 に記載されている内容と比較します。

表 9 で、アダプターにより使用される UNIX ファイル構造について説明し、インストーラーによるアダプターのインストールを選択したときに自動的にインストールされるファイルを示します。UNIX システムでは、次のファイルは自動的に適切な場所にインストールされないことに注意してください。

- WebSphereBIMessageBrokerConnectorTemplate
- mqsiput.exe

表 9. コネクター用にインストールされた UNIX ファイル構造

ProductDir のサブディレクトリー	説明
connectors/WebSphereBIMessageBroker/CWWebSphereBIMessageBroker.jar	WebSphere Business Integration Message Broker コネクター・エージェントによってのみ使用されるクラスを含みます。
connectors/WebSphereBIMessageBroker/start_WebSphereBIMessageBroker.sh	コネクターのシステム始動スクリプト

表 9. コネクター用にインストールされた UNIX ファイル構造 (続き)

***ProductDir* のサブディレクトリー**	**説明**
connectors/messages/WebSphereBIMessageBrokerConnector.txt	コネクターのメッセージ・ファイル
bin/Data/App/WebSphereBIMessageBrokerConnectorTemplate	アダプター定義のテンプレート・ファイル
connectors/WebSphereBIMessageBroker/samples/LegacyItem /WebSphereBIMessageBrokerConnector.cfg	WBIMB アダプター構成ファイルのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/PortConnector.cfg	ポート・コネクター構成ファイルのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_LegacyItem.xsd	サンプル・ビジネス・オブジェクト・リポジトリー・ファイル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_LegacyItem_XMLDoc.xsd	サンプル・ビジネス・オブジェクト・リポジトリー・ファイル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_MO_Config.xsd	コネクター・エージェント・メタオブジェクトのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_MO_DataHandler.xsd	データ・ハンドラー用のトップレベル・メタオブジェクトのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_MO_DataHandler_XMLConfig.xsd	XML データ・ハンドラーの構成に使用されるメタオブジェクトのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/LegacyItem.txt	サンプルを使用するための入力ファイルのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/mqsiput.exe	サンプルとともに使用する WebSphere MQ 用のユーティリティー
connectors/WebSphereBIMessageBroker/samples/LegacyItem/ Sample_WebSphereBIMessageBroker_Project/.project connectors/WebSphereBIMessageBroker/samples/LegacyItem /Sample_WebSphereBIMessageBroker_Project/LocalDomain.configmgr connectors/WebSphereBIMessageBroker/samples/LegacyItem /Sample_WebSphereBIMessageBroker_Project/Sample_WBIMB_bar.bar	WBIMB プロジェクトのサンプル
connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT/.project connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT¥ AdjustFormat.esql connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT/ Check_Color.esql connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT/ From_LegacyApplication_to_WBI.msgflow connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT/ From_WBI_to_LegacyApplication.msgflow connectors/WebSphereBIMessageBroker/samples/LegacyItem/MSG_FLOW_RPOJECT/ Loopback.msgflow	メッセージ・フロー・プロジェクトのサンプル

注: すべての製品のパス名は、ご使用のシステムで製品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。

第 3 章 コネクターの構成

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

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ー・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)

アダプターが DB2 Information Integrator をサポートしている場合は、WMQI オプションと DB2 II 標準プロパティーを使用します (付録『標準プロパティー』の「注」列を参照してください)。

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

- コネクターを構成するための**コネクター固有のプロパティー・テンプレート**を作成する。
- **コネクター構成ファイル**を作成する。インストールするコネクター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ーを設定する。
  場合によっては、コネクター・テンプレートでプロパティー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る場合があります。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 (19 ページの『スタンドアロン・モードでの Configurator の実行』を参照)。

コネクター構成プロパティーには、標準の構成プロパティー (すべてのコネクターがもつプロパティー) と、コネクター固有のプロパティー (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ーで必要なプロパティー) とが含まれます。

標準プロパティーはすべてのコネクターにより使用されるので、標準プロパティーを最初から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ー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ーの値を設定する必要があります。

標準プロパティーの範囲は、ブローカーと構成によって異なる可能性があります。特定のプロパティー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ーがあります。Connector Configurator の「標準のプロパティー」ウィンドウには、特定の構成で設定可能なプロパティーが表示されます。

ただし**コネクター固有プロパティー**の場合は、最初にプロパティーを定義し、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ー固有プロパティー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、20 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。

UNIX でのコネクターの実行

Connector Configurator は、Windows 環境内でのみ実行されます。UNIX 環境でコネクター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 内のプロパティーには、デフォルトで Windows 用のディレクトリー・パスの規則に設定されるディレクトリー・パスを使用するものもあります。UNIX 環境で構成ファイルを使用する場合は、それらのディレクトリー・パスに関する UNIX の規則に合うようにパスを変更してください。拡張検証で正しいオペレーティング・システム規則が使用されるように、ツールバーのドロップ・リストでターゲット・オペレーティング・システムを選択します。

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、System Manager を実行せずに Connector Configurator を実行し、コネクター構成ファイルを編集することができます。

これを行うには、以下の手順を実行します。

- 「**スタート**」>「**プログラム**」から、「**IBM WebSphere Business Integration Adapters**」>「**IBM WebSphere Business Integration Toolset**」>「**Connector Configurator**」をクリックします。
- 「**ファイル**」>「**新規**」>「**コネクター構成**」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 (26 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、以下の手順を実行します。

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、「**コネクター**」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、「**新規コネクター**」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、以下の手順を実行します。

- 「System Manager」ウィンドウの「コネクター」フォルダーで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。
- Connector Configurator で「**ファイル**」>「**開く**」を選択します。プロジェクトまたはプロジェクトが保管されているディレクトリーからコネクター構成ファイルを選択します。
- 「標準のプロパティー」タブをクリックし、この構成ファイルに含まれるプロパティーを確認します。

コネクター固有のプロパティー・テンプレートの作成

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

コネクター固有プロパティーのテンプレートを新規に作成するか、または既存のコネクター定義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、20 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、新しい名前でこのテンプレートを保管します。既存のテンプレートは ¥WebSphereAdapters¥bin¥Data¥App ディレクトリーにあります。

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ーを作成し、プロパティーの一般特性および値を定義し、プロパティー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、新規コネクター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。

Connector Configurator でテンプレートを作成するには、以下のステップを実行します。

1. 「**ファイル**」>「**新規**」>「**コネクター固有プロパティー・テンプレート (Connector-Specific Property Template)**」とクリックします。
2. 「**コネクター固有プロパティー・テンプレート**」 ダイアログ・ボックスが表示されます。
   - 「**新規テンプレート名を入力してください**」の下の「**名前**」フィールドに、新規テンプレート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。
   - テンプレートに含まれているコネクター固有のプロパティー定義を調べるには、「**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ー定義のリストが「**テンプレートのプレビュー**」表示に表示されます。
3. テンプレートを作成するときには、コネクターに必要なプロパティー定義に類似したプロパティー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。ご使用のコネクターで使用するコネクター固有のプロパティー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。
   - 既存のテンプレートを変更する場合には、「**変更する既存のテンプレートを選択してください: 検索テンプレート**」の下の「**テンプレート名**」テーブルのリストから、テンプレート名を選択します。
   - このテーブルには、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。テンプレートを検索することもできます。

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、「**プロパティー: コネクター固有プロパティー・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ーの「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:**
  プロパティー・タイプ

プロパティー・サブタイプ
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

「**プロパティー・サブタイプ (Property Subtype)**」は、「**プロパティー・タイプ**」が String である場合に選択できます。これは、構成ファイルを保管する際に構文検査を提供するオプションの値です。デフォルトではブランクのスペースです。これは、プロパティーにサブタイプがないことを意味します。

プロパティー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、プロパティーの最大長、最大複数値、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、以下の手順を実行します。

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. 「**プロパティーを編集**」表示でプロパティーの名前を選択します。
3. 「**最大長**」および「**最大複数値**」のフィールドに値を入力します。

新規プロパティー値を作成するには、以下のステップを実行します。

1. 「値」列見出しの左方にある正方形を右クリックします。
2. ポップアップ・メニューから、「**追加**」を選択して「プロパティー値」ダイアログ・ボックスを表示します。プロパティー・タイプによっては、ダイアログ・ボックスに値、または値と範囲の両方を入力できます。
3. 新規プロパティー値を入力し、「OK」をクリックします。右側の「**値**」パネルに値が表示されます。

「**値**」パネルには、3 つの列からなるテーブルが表示されます。

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

「**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、「**プロパティー値**」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ー・テンプレート**」ダイアログ・ボックスが表示されます。

依存プロパティーは、別のプロパティー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ーです。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。プロパティーを依存プロパティーとして指定し、依存する条件を設定するには、以下の手順を実行します。

1. 「**使用可能なプロパティー**」表示で、依存プロパティーとして指定するプロパティーを選択します。
2. 「**プロパティーを選択**」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ーを選択します。
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、依存プロパティー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ー**」表示で強調表示された依存プロパティーで、矢印をクリックし、「**依存プロパティー**」表示に移動させます。
6. 「**完了 (Finish)**」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。

パス名の設定

パス名の設定に関するいくつかの一般規則を次に示します。

- Windows および UNIX では、ファイル名の最大長は 255 文字です。
- Windows では、絶対パス名は [Drive:][Directory]¥filename の形式に従っている必要があります。例えば、C:¥WebSphereAdapters¥bin¥Data¥Std¥StdConnProps.xml です。
  UNIX では、先頭文字は / である必要があります。
- キュー名には、先行スペースまたは埋め込みスペースは使用できません。

新規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、構成ファイルの名前を指定し、統合ブローカーを選択する必要があります。

また、ファイルでの拡張検証用のオペレーティング・システムを選択します。ツールバーには「**ターゲット・システム**」というドロップ・リストがあります。これを使用すると、プロパティーの拡張検証用のターゲット・オペレーティング・システムを選択できます。選択可能なオプションは、「Windows」、「UNIX」、「その他」 (Windows または UNIX 以外の場合)、および「なし - 拡張検証なし (None-no extended validation)」(拡張検証をオフにします) です。始動時のデフォルトは Windows です。

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

- 「System Manager」ウィンドウで、「**ツール**」メニューから「**Connector Configurator**」を選択します。「Connector Configurator」が開きます。
- スタンドアロン・モードで、Connector Configurator を起動します。

構成ファイルの拡張検証用にオペレーティング・システムを設定するには、次の手順を実行します。

- メニュー・バーで、「**ターゲット・システム:**」ドロップ・リストをプルダウンします。
- 実行しているオペレーティング・システムを選択します。

次に、「**ファイル**」>「**新規**」>「**コネクター構成**」を選択します。「新規コネクター」ウィンドウで、新規コネクターの名前を入力します。

また、統合ブローカーも選択する必要があります。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ーが決まります。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。

- 「**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を完了します。

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、その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. メニュー・バーの「**ターゲット・システム:**」ドロップ・リストを使用して、構成ファイルの拡張検証用のオペレーティング・システムを設定します (前述の『新規構成ファイルの作成』を参照してください)。
2. 「**ファイル**」>「**新規**」>「**コネクター構成**」をクリックします。
3. 以下のフィールドを含む「**新規コネクター**」ダイアログ・ボックスが表示されます。
   - **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。

- **「コネクター固有プロパティー・テンプレート (Connector-Specific Property Template)」を選択します。**

  ご使用のコネクター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、「**プロパティー・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ー固有のプロパティーが表示されます。

  使用するテンプレートを選択し、「**OK**」をクリックします。

4. 構成しているコネクター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ー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。
5. 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・コネクターを保管**」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、コネクターの始動ファイルで指定するコネクター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。
6. 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、コネクター定義を完了します。

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 つ以上の形式になります。

- コネクター定義ファイル。
  コネクター定義ファイルは、特定のコネクターのプロパティーと、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。コネクターの配布パッケー

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 コネクターの場合は CN_XML.txt です)。

- ICS リポジトリー・ファイル。
  コネクターの以前の ICS インプリメンテーションで使用した定義は、そのコネクターの構成で使用されたリポジトリー・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。
- コネクター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、コネクターのコネクター固有プロパティーのほとんど、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、コネクター構成ファイルは、ファイルを開いて、プロパティー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ーを構成するには、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。

以下の手順を実行して、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**」とクリックします。
2. 「**ファイル・コネクターを開く**」ダイアログ内で、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 リポジトリー (*.in、*.out)

     ICS 環境でのコネクターの構成にリポジトリー・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。リポジトリー・ファイルに複数のコネクター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 (*.*)

     コネクターのアダプター・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、適切なコネクター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ー構成を開くには、以下の手順を実行します。

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**」とクリックします。

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ーを開くと、「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ー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、コネクター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、以下の手順を実行します。

1. 「**標準のプロパティー**」タブで、BrokerType プロパティー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コネクター・プロパティーが「標準のプロパティー」タブに表示されます。テーブルには、「**プロパティー名**」、「**値**」、「**タイプ**」、「**サブタイプ**」 (タイプが String である場合)、「**説明**」、および「**更新メソッド**」が表示されます。
3. ここでファイルを保管するか、または 30 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。
4. 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。

   複数のコネクター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、コネクター構成をすべて 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。

   構成ファイルを作成する前に、「**ターゲット・システム**」ドロップ・リストを使用します。これを使用すると、プロパティーの拡張検証用のターゲット・オペレーティング・システムを選択できます。

   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ーすべてに値が設定されているかどうかが確認されます。必須の標準プロパティー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、そのプロパティーの値を指定する必要があります。

   「**ターゲット・システム**」ドロップ・リストから「Windows」、「UNIX」、または「その他」の値を選択して拡張検証機能を使用することを選択した場合、システムはタイプだけでなくプロパティー・サブタイプも検証し、検証が失敗した場合は警告メッセージを表示します。

構成ファイル・プロパティーの設定

新規のコネクター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ー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ー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ーで、以下のカテゴリーのプロパティーに値が設定されている必要があります。

- 標準のプロパティー
- コネクター固有のプロパティー
- 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ーの場合に該当する)

注: JMS メッセージングを使用するコネクターの場合は、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ー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ーの場合、以下のプロパティーの値も設定されている必要があります。

- 関連マップ
- リソース
- メッセージング (該当する場合)
- セキュリティー

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ー値も設定可能です。ただし、標準のプロパティーおよびコネクター固有プロパティー、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。

標準プロパティーとコネクター固有プロパティーの違いは、以下のとおりです。

- コネクターの標準プロパティーは、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用されます。すべてのコネクターが同じ標準プロパティーのセットを使用します。これらのプロパティーの説明は、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プロパティーは、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ーがあります。これらのプロパティーには、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ーおよび推奨値が記述されています。

「**標準のプロパティー**」と「**コネクター固有プロパティー (Connector-Specific Properties)**」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、標準のプロパティーを表します。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ー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ーを表します。これらのプロパティーは、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ーの除去も可能です。
- 「値」フィールドは構成できます。
- プロパティーごとに「**更新メソッド**」フィールドが表示されます。これは、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。この設定を構成することはできません。

標準コネクター・プロパティーの設定

標準のプロパティーの値を変更するには、以下の手順を実行します。

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る場合にはメニューから値を選択します。

   注: プロパティーのタイプが String である場合は、「サブタイプ」列にサブタイプ値がある場合があります。このサブタイプは、プロパティーの拡張検証に使用されます。
3. 標準のプロパティーの値をすべて入力すると、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。「**標準のプロパティー**」 (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きに、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。

特定の標準プロパティーの詳細については、「標準プロパティー」タブのシートで、そのプロパティーの「説明」列内のエントリーを左クリックします。全般ヘルプをインストールしてある場合は、右側に矢印ボタンが表示されます。ボタンをクリックすると、「ヘルプ」ウィンドウが開かれ、その標準プロパティーの詳細が表示されます。

注: ホット・ボタンが表示されない場合、そのプロパティーの全般ヘルプはありません。

インストールされている場合、全般ヘルプ・ファイルは
<*ProductDir*>¥bin¥Data¥Std¥Help¥<*RegionalSetting*>¥ にあります。

コネクター固有の構成プロパティーの設定

コネクター固有の構成プロパティーの場合、プロパティー名の追加または変更、値の構成、プロパティーの削除、およびプロパティーの暗号化が可能です。プロパティー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。プロパティーを追加するときは「**追加**」をクリックします。子プロパティーを追加するには、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. プロパティーまたは子プロパティーの値を入力します。

   注: プロパティーのタイプが String である場合は、「サブタイプ」ドロップ・リストからサブタイプを選択できます。このサブタイプは、プロパティーの拡張検証に使用されます。
3. プロパティーを暗号化するには、「**暗号化**」ボックスを選択します。
4. 特定のプロパティーの詳細については、そのプロパティーの「説明」列内のエントリーを左クリックします。全般ヘルプをインストールしてある場合は、ホット・ボタンが表示されます。ホット・ボタンをクリックすると、「ヘルプ」ウィンドウが開かれ、その標準プロパティーの詳細が表示されます。

   注: ホット・ボタンが表示されない場合、そのプロパティーの全般ヘルプはありません。
5. 28 ページの『標準コネクター・プロパティーの設定』で説明したように、変更内容を保管するかまたは破棄するかを選択します。

全般ヘルプ・ファイルがインストールされ、AdapterHelpName プロパティーがブランクである場合、Connector Configurator は
<*ProductDir*>¥bin¥Data¥App¥Help¥<*RegionalSetting*>¥ にあるアダプター固有の全般ヘルプ・ファイルを指します。それ以外の場合、Connector Configurator は、
<*ProductDir*>¥bin¥Data¥App¥Help¥<*AdapterHelpName*>¥<*RegionalSetting*>¥ にあるアダプター固有の全般ヘルプ・ファイルを指します。付録『標準プロパティー』で説明している AdapterHelpName プロパティーを参照してください。

各プロパティーごとに表示される「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ー・プロパティー名を変更すると、コネクターに障害が発生する可能性があります。コネクター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、特定のプロパティー名が必要である場合があります。

コネクター・プロパティーの暗号化

「コネクター固有プロパティー」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ー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、暗号化が解除された値が表示されます。

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

プロパティーに複数の値がある場合には、プロパティーの最初の値に「**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、そのプロパティーのすべての値が暗号化されます。プロパティーの複数の値を暗号化解除するには、そのプロパティーの最初の値の「**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、すべての複数値が暗号化解除されます。

更新メソッド

74 ページの『構成プロパティー値の概要』の下の付録『標準プロパティー』で、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

コネクター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: コネクターによっては、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。

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ー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ー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、以下の手順を実行します。

1. 「**ビジネス・オブジェクト名**」リストの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・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ー定義が、System Manager の ICL (Integration Component Library) 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、以下の手順を実行します。

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、「**プロジェクトに保管**」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、コネクター定義が変更され、削除されたビジネス・オブジェクトはコネクターのこのインプリメンテーションで使用不可になります。コネクター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・オブジェクトにエージェント・サポートがある場合、システムは、コネクター・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・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。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 を選択する場合、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できるビジネス・オブジェクトのリストが示されます。このリストから目的のビジネス・オブジェクトを選択します。

関連付けられたマップ (ICS)

各コネクターは、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ライブ・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、このコネクター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、サポートされるビジネス・オブジェクトを追加指定した場合、それらの内容は、「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、コネクターの、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。
- **明示的バインディング**

  場合によっては、関連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、コネクターごとに、サポートされる各ビジネス・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 つのマップを見付けて、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。

  以下の手順を実行して、マップを明示的にバインドします。

  1. 「**明示 (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することができます。

すべてのコネクターでこの機能がサポートされるわけではありません。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**」タブを使用すると、メッセージング・プロパティーを構成できます。メッセージング・プロパティーは、`DeliveryTransport` 標準プロパティーの値として 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ーは、コネクターによるキューの使用方法に影響します。

メッセージ・キューの検証

メッセージ・キューを検証する前に、次の手順を実行する必要があります。

- WebSphere MQ Series がインストールされていることを確認します。
- ホスト・マシン上のチャネルとポートでメッセージ・キューを作成します。
- ホスト・マシンへの接続をセットアップします。

キューを検証するには、「メッセージング」タブの「メッセージング・タイプ」および「ホスト名」フィールドの右方にある「検証 (Validate)」ボタンを使用します。

セキュリティー (ICS)

Connector Configurator で「**セキュリティー**」タブを使用して、メッセージに対しさまざまなプライバシー・レベルを設定することができます。この機能を使用できるのは、DeliveryTransport プロパティーが JMS に設定されている場合だけです。

デフォルトでは、プライバシーはオフになっています。使用可能にするには、「**プライバシー**」ボックスにチェック・マークを付けます。

鍵ストア・ターゲット・システム絶対パス名を次に示します。

- Windows の場合:
  `<ProductDir>¥connectors¥security¥<connectorname>.jks`
- UNIX の場合:
  `opt/IBM/WebSphereAdapters/connectors/security/<connectorname>.jks`

このパスとファイルは、コネクター、つまりターゲット・システムを開始することを計画しているシステム上にある必要があります。

右方の「ブラウズ」ボタンを使用できるのは、ターゲット・システムが現在実行中のシステムである場合だけです。「**プライバシー**」が使用可能になっていて、メニュー・バーの「**ターゲット・システム**」が Windows に設定されていない限り、これはグレー表示されます。

「**メッセージ・プライバシー・レベル (Message Privacy Level)**」は、3 つのメッセージ・カテゴリー (すべてのメッセージ、すべての管理メッセージ、およびすべてのビジネス・オブジェクト・メッセージ) で次のように設定できます。

- “” はデフォルトであり、メッセージ・カテゴリーに対してプライバシー・レベルが設定されていない場合に使用されます。
- なし
  デフォルトとは異なります。これは、メッセージ・カテゴリーに対して意図的に「なし」のプライバシー・レベルを設定する場合に使用します。
- integrity
- privacy
- integrity_plus_privacy

「**キー保守 (Key Maintenance)**」機能を使用すると、サーバーとアダプター用の公開鍵を生成、インポート、およびエクスポートすることができます。

- 「**キーの生成 (Generate Keys)**」を選択すると、「キーの生成 (Generate Keys)」ダイアログ・ボックスが表示され、キーを生成するキー・ツールのデフォルトが示されます。
- 鍵ストア値は、デフォルトで、「セキュリティー」タブの「**鍵ストア・ターゲット・システム絶対パス名 (Keystore Target System Absolute Pathname)**」に入力した値になります。
- 「OK」を選択すると、エントリーが検証され、キー証明書が生成され、出力が「Connector Configurator ログ (Connector Configurator log)」ウィンドウに送信されます。

アダプター鍵ストアへの証明書をインポートするには、まずサーバー鍵ストアからそれをエクスポートする必要があります。「**アダプター公開鍵のエクスポート (Export Adapter Public Key)**」 を選択すると、「アダプター公開鍵のエクスポート (Export Adapter Public Key)」ダイアログ・ボックスが表示されます。

- エクスポート証明書は、デフォルトでは、ファイル拡張子が <filename>.cer であることを除き鍵ストアと同じ値です。

「**サーバー公開鍵のインポート (Import Server Public Key)**」 を選択すると、「サーバー公開鍵のインポート (Import Server Public Key)」ダイアログ・ボックスが表示されます。

- インポート証明書は、デフォルトでは、<*ProductDir*>¥bin¥ics.cer (ファイルがシステム上にある場合) です。
- import Certificate Association がサーバー名になります。サーバーが登録されている場合は、それをドロップ・リストから選択できます。

「**アダプター・アクセス制御 (Adapter Access Control)**」機能を使用できるのは、DeliveryTransport の値が IDL である場合だけです。デフォルトでは、アダプターはゲスト ID でログインします。「**ゲスト ID を使用 (Use guest identity)**」ボックスにチェック・マークが付けられていない場合は、「**アダプター ID (Adapter Identity)**」フィールドと「**アダプター・パスワード (Adapter Password)**」フィールドを使用できます。

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、以下の手順を実行します。

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、指定する格納場所に移動し、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このデータ・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. コネクターの標準構成プロパティー』にある `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するか、あるいは
- 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。デフォルトでは、このファイルは `¥WebSphereAdapters¥bin¥Data¥App` に保管されます。
- WebSphere Application Server プロジェクトをセットアップしている場合には、このファイルを WebSphere Application Server プロジェクトに保管することもできます。

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 システム・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、以下の手順を実行します (オプション)。

- Connector Configurator で既存の構成ファイルを開きます。
- 「**標準のプロパティー**」タブを選択します。
- 「標準のプロパティー」タブの「**ブローカー・タイプ**」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、プロパティー・ウィンドウの利用可能なタブおよびフィールド選択がただちに変更され、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

コネクターの構成ファイルを作成し、そのファイルを変更した後で、コネクターの始動時にコネクター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、コネクターが使用する始動ファイルを開き、コネクター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ー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ーのドロップ・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、製品ディレクトリーの ¥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 プロパティー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、以下に太字で示される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>
```

構成例の検証

構成例を検証するには、次のようにします。

- 『付録 C. チュートリアル』を参照してください。このチュートリアルに従うと、アダプターに付属しているサンプルを使用してサンプル・シナリオを簡単に構成し実行することができます。

キューの Uniform Resource Identifier (URI) の設定

Uniform Resource Identifier を使用して、キューの指定や、指定したキューの値の設定または変更を行うことができます。これらは、キューを定義するコネクター固有のプロパティーの値を指定するときに行います。

キューの URI を設定するには、次のようにします。

- Connector Configurator を使用して、コネクター固有のキューを次の構文で指定します。
  キューの URI の構文: キューの URI は、シーケンス queue:// を先頭とし、その後に以下の要素が続きます。
  - キューが存在しているキュー・マネージャーの名前
  - 別の /
  - キューの名前
  - (オプション) 残りのキュー・プロパティーの、名前と値のペアのリスト

例えば、次の URI を指定した場合、キュー・マネージャー crossworlds.queue.manager に存在するキュー IN に接続し、すべてのメッセージが優先順位 5 の WebSphere MQ メッセージとして送信されます。

```
queue://crossworlds.queue.manager/MQCONN.IN?targetClient=1&priority=5
```

以下の表に、キュー URI のプロパティー名を示します。

プロパティー名	説明	値
expiry	メッセージの存続時間 (ミリ秒単位)	0 = 無制限 正の整数 = タイムアウト (ミリ秒単位)
priority	メッセージの優先順位	0 から 9 で、1 が最高の優先順位。値 -1 は、このプロパティーがキューの構成によって決定されることを意味します。値 -2 は、コネクター自身のデフォルト値を使用できるよう指定します。
persistence	メッセージをディスクに「ハード化」するかどうか	1 = 非永続 2 = 永続値 -1 は、このプロパティーがキューの構成によって決定されることを意味します。値 -2 は、コネクター自身のデフォルト値を使用できるよう指定します。
CCSID	アウトバウンド・メッセージの文字セット・エンコード	整数: WebSphere MQ の資料にリストされている有効な値。この値は、CCSID コネクター固有の構成プロパティーの値と一致する必要があります。 103 ページの『CCSID』を参照してください。
targetClient	受信側アプリケーションが JMS 準拠であるかどうか	0 = JMS (MQRFH2 ヘッダー) 1 = MQ (MQMD ヘッダーのみ)
encoding	数値フィールドの表示方法	基本的な WebSphere MQ 資料に記載されている整数値。

注: アダプターは、MQMessage 内のデータの文字セット (CCSID) またはエンコード属性を制御できません。データ変換はデータがメッセージ・バッファーから検索されるかメッセージ・バッファーにデリバリーされるときに行われるため、コネクターは JMS の IBM WebSphere MQ インプリメンテーションに依存してデータ変換を行います (IBM WebSphere MQ Java クライアント・ライブラリーの資料を参照してください)。したがって、これらの変換は、ネイティブ WebSphere MQ API がオプション MQGMO_CONVERT を使用して実行する変換と双方向で等しくなければなりません。コネクターは、変換プロセスにおける差異または失敗を制御できません。コネクターは、特別な変更を必要とせずに、WebSphere MQ によってサポートされるすべての CCSID またはエンコードのメッセージ・データを検索できます。特定の CCSID またはエンコードのメッセージを送信するには、出力キューが完全修飾 URI であり、CCSID および encoding の値を指定している必要があります。コネクターはこの情報を WebSphere MQ に渡し、WebSphere MQ は MQMessage のデリバリーのためにデータをエンコードするときに (JMS API を介して) この情報を使用します。CCSID およびエンコードがサポートされていなくても、IBM の Web サイトから最新バージョンの IBM WebSphere MQ Java クライアント・ライブラリーをダウンロードすることによって、多くの場合解決できます。それでも CCSID お

よびエンコードに関する問題が解消されない場合は、IBM ソフトウェア・サポートに連絡し、代替の Java 仮想マシンを使用してコネクターを実行することを検討してください。

イベント・デリバリーの保証

保証付きイベント・デリバリー機能により、コネクター・フレームワークは、コネクターのイベント・ストアから、JMS イベント・ストア、そして宛先の JMS キューまでの間で、イベントを失うことなく、また 2 度送りすることなく、確実に送達することができます。コネクターを JMS 対応にするには、connectorDeliveryTransport 標準プロパティーを JMS に構成する必要があります。このように構成されたコネクターは、JMS トランスポートを使用し、コネクターと統合ブローカー間の以降の通信は、このトランスポートを介して行われます。JMS トランスポートにより、メッセージは最終的には確実に宛先に配達されます。このトランスポートの役割は、一度トランザクション・キュー・セッションが開始すると、メッセージはコミットが発行されるまで、確実にそのトランスポートのキャッシュに保持されます。障害が発生するかロールバックが発行されると、メッセージは破棄されます。

注: 保証付きイベント・デリバリー機能が使用されない場合、コネクターがイベントをパブリッシュして (コネクターがその pollForEvents() メソッド内から gotApplEvent() メソッドを呼び出して) から、コネクターがイベント・レコードを削除してイベント・ストアを更新する (またはイベント・ストアを「イベント通知済み」状況に更新する) まで、障害が起こる可能性のある短い時間枠が存在します。この間に障害が発生すると、イベントは送信されますが、そのイベント・レコードは「進行中」の状況でイベント・ストアに残ります。コネクターが再始動すると、イベント・ストアにイベント・レコードが残っているため、これを送信する結果、イベントが 2 回送信されることになります。

JMS イベント・ストアを使用する JMS 対応コネクター用、または JMS イベント・ストアを使用しない JMS 対応コネクター用の保証付きイベント・デリバリー機能を構成することができます。保証付きイベント・デリバリーを行うようにコネクターを構成するには、「コネクター開発ガイド (Java 用)」の説明を参照してください。

コネクター・フレームワークがビジネス・オブジェクトを WebSphere InterChange Server 統合ブローカーに配送できない場合、オブジェクトは (UnsubscribedQueue と ErrorQueue ではなく) FaultQueue に配置されて、状況表示と問題の説明を生成します。FaultQueue メッセージは、MQRFH2 フォーマットで書き込まれます。

第 4 章 コネクターの実行

この章では、コネクター始動ファイルの構成方法と、コネクターの始動と停止の方法について説明します。

コネクターの実行の概要

コネクターは、以下の作業が完了している場合に実行できます。

- コネクターと関連ファイルのインストール
- ビジネス・オブジェクトとメタオブジェクトの構成
- データ・ハンドラーの構成
- コネクターの構成

この章で説明する作業

この章では、以下の作業について説明します。

表 10. アダプターの実行: 作業ロードマップ

作業	関連手順 (参照先)	詳細情報 (参照先)
コネクターの始動	『コネクターの始動』	
コネクターの停止	43 ページの『コネクターの停止』	60 ページの『データ・ハンドラーの入力キューへのマッピングの概要』
アダプターの複数インスタンスの実行	44 ページの『アダプターの複数インスタンスの実行』	

コネクターの始動

コネクターは、**コネクター始動スクリプト**を使用して明示的に始動する必要があります。Windows システムでは、始動スクリプトは、次に示すようなコネクターのランタイム・ディレクトリーに存在していなければなりません。

ProductDir¥connectors¥*connName*

ここで、*connName* はコネクターを示します。

UNIX システムでは、始動スクリプトは、*UNIX ProductDir*/bin ディレクトリーに存在していなければなりません。

始動スクリプトの名前は、表 11 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 11. コネクターの始動スクリプト

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`
Windows	`start_connName.bat`

始動スクリプトは、実行時に、デフォルトでは *Productdir* に構成ファイルがあると予期します (後述のコマンドを参照してください)。これは、構成ファイルを置く場所です。

注: アダプターが JMS トランスポートを使用している場合は、ローカル構成ファイルが必要です。

コネクター始動スクリプトは、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**」>「**コネクター**」 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、ご使用のコネクターへのデスクトップ・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 -start connName brokerName [-cconfigFile ]`

  ここで、*connName* はコネクターの名前であり、*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ー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。Adapter Monitor は、WebSphere Application Server または InterChange Server ブローカーで実行する System Manager の開始時に起動されます。

  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。

- System Manager から (すべてのブローカーで使用可能)。

  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。

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。
  - InterChange Server を使用している場合、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* はコネクターの名前です。
- 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。
- 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 システムのシャットダウン時に、コネクターは停止します。

アダプターの複数インスタンスの実行

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。

コネクターに、コネクター固有のメタオブジェクトがある場合、コネクター・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、次のディレクトリーを作成して、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

各コネクター・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. 初期コネクター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。初期コネクター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. 初期コネクターのファイルは、次のディレクトリーに入っていなければなりません。

   ```
   ProductDir¥repository¥initialConnectorInstance
   ```

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

コネクター定義の作成

Connector Configurator 内で、コネクター・インスタンスの構成ファイル (コネクター定義) を作成します。これを行うには、以下の手順を実行します。

1. 初期コネクターの構成ファイル (コネクター定義) をコピーし、名前変更をします。
2. 各コネクター・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、コネクター・プロパティーをカスタマイズします。

始動スクリプトの作成

始動スクリプトを作成するには、以下のようにします。

1. 初期コネクターの始動スクリプトをコピーし、以下のコネクター・ディレクトリー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、 44 ページの『新規ディレクトリーの作成』で作成したコネクター・ディレクトリーに格納します。
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. 初期コネクターのショートカット・テキストをコピーし、新規コネクター・インスタンスの名前に一致するように (コマンド行で) 初期コネクター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ー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・コネクター作成の詳細については、「*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。

エラー処理の概要

コネクターによって生成されたすべてのエラー・メッセージは、`MQSIV2Connector.txt` という名前のメッセージ・ファイルに格納されます。(このファイルの名前は、コネクター構成標準プロパティー `LogFileName` によって決定されます。) 各エラーはエラー番号が付けられ、その後にエラー・メッセージが表示されます。

```
Message number
Message text
```

コネクターは、以降のセクションで説明する方法で特定のエラーを処理します。

アプリケーション・タイムアウト

エラー・メッセージ ABON_APPRESPONSETIMEOUT は、以下の場合に戻されます。

- コネクターは、メッセージの検索中に JMS サービス・プロバイダーとの接続を確立できませんでした。
- コネクターはビジネス・オブジェクトをメッセージに正常に変換しましたが、接続切断が原因でメッセージを出力キューにデリバリーできませんでした。
- コネクターはメッセージを発行しましたが、変換プロパティー `TimeoutFatal` が `True` であるビジネス・オブジェクトに対する応答の待機中にタイムアウトが発生しました。
- コネクターは、戻りコードが APP_RESPONSE_TIMEOUT または UNABLE_TO_LOGIN の応答メッセージを受信しました。

アンサブスクライブされたビジネス・オブジェクト

コネクターは、次の場合に、UnsubscribedQueue プロパティーで指定されたキューにメッセージをデリバリーします。

- アンサブスクライブされたビジネス・オブジェクトに関連付けられているメッセージをコネクターが検索した場合、または gotApplEvent() メソッドによって NO_SUBSCRIPTION_FOUND コードが戻された場合。
- コネクターがメッセージを検索したが、FORMAT フィールドのテキストをビジネス・オブジェクト名に関連付けることができない場合。

注: UnsubscribedQueue が定義されていない場合、アンサブスクライブされたメッセージは破棄されます。

コネクターがアクティブでない

gotApplEvent() メソッドが CONNECTOR_NOT_ACTIVE コードを戻すと、pollForEvents() メソッドは APP_RESPONSE_TIMEOUT コードを戻し、イベントは InProgress キューに置かれたままになります。

データ・ハンドラーによる変換

データ・ハンドラーがメッセージからビジネス・オブジェクトへの変換に失敗した場合、または、(JMS プロバイダーではなく) ビジネス・オブジェクトに固有の処理エラーが発生した場合、メッセージは ErrorQueue で指定されたキューにデリバリーされます。ErrorQueue が定義されていない場合、エラーが原因で処理できなかったメッセージは破棄されます。

データ・ハンドラーがビジネス・オブジェクトからメッセージへの変換に失敗した場合、BON_FAIL が戻されます。

JMS プロパティー

メッセージに対して JMS プロパティーの読み取りまたは書き込みを実行できない場合、コネクターはエラーをログに記録し、要求またはイベントは失敗します。ReplyToQueue が未指定またはアクセス不可の場合、コネクターはエラーをログに記録し、要求は失敗します。CorrelationID が無効であるか設定できない場合、コネクターはエラーをログに記録し、要求は失敗します。

いずれの場合も、ログに記録されるメッセージはコネクター・メッセージ・ファイルから読み込まれます。

入力フォーマットの多重定義

コネクターは通常、メッセージを検索するときに、ビジネス・オブジェクトと動詞の特定の 1 つの組み合わせと入力フォーマットを突き合わせます。そして、コネクターはビジネス・オブジェクト名とメッセージの内容をデータ・ハンドラーに渡します。これにより、データ・ハンドラーは、メッセージの内容がユーザーの予期するビジネス・オブジェクトに対応していることを確認できます。

しかし、複数のビジネス・オブジェクトに対して同じ入力フォーマットが定義されている場合、コネクターは、データ・ハンドラーにデータを渡す前に、データがどのビジネス・オブジェクトを表すのかを判別できません。そのような場合、コネク

ターはデータ・ハンドラーにメッセージの内容のみを渡し、生成されるビジネス・オブジェクトに基づいて変換プロパティーを調べます。したがって、データ・ハンドラーは、メッセージの内容のみに基づいてビジネス・オブジェクトを決定する必要があります。

生成されたビジネス・オブジェクトに動詞が設定されていない場合、コネクターはビジネス・オブジェクトの動詞の種類を限定せずに、このビジネス・オブジェクトに定義されている変換プロパティーを検索します。変換プロパティーが 1 セットのみ検出された場合は、コネクターは特定の動詞を割り当てます。複数のプロパティーが検出された場合は、コネクターは動詞を判別できないため、メッセージの処理が失敗します。

トレースの概要

トレースはオプションのデバッグ機能で、オンにするとコネクターの動作を詳細にトレースできます。デフォルトでは、トレース・メッセージは STDOUT に書き込まれます。トレース・メッセージの構成の詳細については、*第 2 章*のコネクター構成プロパティーについての説明を参照してください。トレースの使用可能化や設定方法などの詳細については、「*コネクター開発ガイド (Java 用)*」を参照してください。

コネクターのトレース・メッセージに推奨される内容を以下に示します。

レベル 0 コネクターのバージョンを確認するトレース・メッセージに使用します。

レベル 1 処理される各ビジネス・オブジェクトについての重要な情報を提供するトレース・メッセージや、ポーリング・スレッドが入力キュー内で新しいメッセージを検出するたびに記録されるトレース・メッセージに使用します。

レベル 2 ビジネス・オブジェクトが `gotApplEvent()` または `executeCollaboration()` から InterChange Server に通知されるたびに記録されるトレース・メッセージに使用します。

レベル 3 データ・フォーマット (例えば XML) からビジネス・オブジェクトへの変換およびビジネス・オブジェクトからデータ・フォーマットへの変換に関する情報を提供するトレース・メッセージや、出力キューへのメッセージのデリバリーに関する情報を提供するトレース・メッセージに使用します。

レベル 4 コネクターが動作を開始または終了した時間を記録するトレース・メッセージに使用します。

レベル 5 コネクターの初期化を示すトレース・メッセージ、アプリケーション内で実行されるステートメントを表すトレース・メッセージ、メッセージが除去されるかキューに送出されるたびに記録されるトレース・メッセージ、または、ビジネス・オブジェクトのダンプを記録するトレース・メッセージに使用します。

このレベルを使用して、アダプターによってキャッチされた例外で printStackTrace() をダンプします。

第 5 章 オブジェクトの作成

- 『オブジェクトの作成の概要』
- 50 ページの『ビジネス・オブジェクトの作成』
- 51 ページの『ビジネス・オブジェクトの変更』
- 53 ページの『メタオブジェクトの作成』

この章では、ビジネス・オブジェクトとメタオブジェクトの作成および変更に必要な作業について説明します。コネクターを正常に機能させるには、これらの 2 種類のオブジェクトの両方が必要です。

オブジェクトの作成の概要

以下の 2 種類のオブジェクトの定義を作成する必要があります。

- **ビジネス・オブジェクト** ビジネス・エンティティー (従業員やカタログ項目など) またはデータ・トランザクション (作成、更新、削除など) を表現します。データ処理に関するアプリケーション固有の指示も含まれています。コネクターにはビジネス・オブジェクトのサンプルのみが付属しています。これらのサンプルはチュートリアルで使用します。ご使用のコネクターに適したビジネス・オブジェクト定義をビルドする必要があります。
- **メタオブジェクト** 構成メタオブジェクトは、データ・フォーマット、キュー、マッピング、メッセージ・ヘッダー情報、および応答時間を定義します。これらの情報は、コネクターがビジネス・オブジェクトとメッセージを交換するときの動作を左右します。コネクターには、メタオブジェクトのサンプルのみが付属しています。このサンプルを変更してアダプターで使用できるようにしても、独自のメタオブジェクトを作成してもかまいません。メタオブジェクトは、プロパティーを使用して構成する必要があります。この構成を行わないと、コネクターが正常に機能しません。

この章で説明する作業

この章で説明する作業は、以下のとおりです。

表 12. オブジェクトの作成: 作業ロードマップ

作業	関連手順 (参照先)	詳細情報 (参照先)
ビジネス・オブジェクトの作成	50 ページの『ビジネス・オブジェクトの作成』	「*ビジネス・オブジェクト開発ガイド*」
ビジネス・オブジェクトの変更	51 ページの『ビジネス・オブジェクトの変更』	「*ビジネス・オブジェクト開発ガイド*」
メタオブジェクトの作成	53 ページの『メタオブジェクトの作成』	59 ページの『静的メタオブジェクトの作成の概要』。 61 ページの『動的子メタオブジェクトの作成の概要』
データ・ハンドラーの入力キューへのマッピング	61 ページの『データ・ハンドラーの入力キューへのマッピングのステップ』	60 ページの『データ・ハンドラーの入力キューへのマッピングの概要』

ビジネス・オブジェクトの作成

このトピックでは、ビジネス・オブジェクト定義の作成の概要と手順を示します。

ビジネス・オブジェクトの作成の概要

コネクターをインストールした後は、ビジネス・オブジェクト定義を作成する必要があります。構成するデータ・ハンドラーやメタオブジェクトの属性に応じて求められる要件以外に、ビジネス・オブジェクトの構造に関する要件はありません。コネクターで使用するデータ・フォーマットが決定したら、適切なデータ・ハンドラーを構成し、そのフォーマットに準拠したビジネス・オブジェクトを生成します。

WebSphere Business Integration Message Broker は XML メッセージをネイティブにサポートしています。そのため、Adapter for WebSphere Business Integration Message Broker は、デフォルトでは XML データ・ハンドラーに対応するように構成されます。つまり、このアダプターは、ビジネス・オブジェクトからメッセージへの変換およびメッセージからビジネス・オブジェクトへの変換に XML データ・ハンドラーを使用します。

注: メッセージやビジネス・オブジェクトの要件をサポートするために必要な場合には、XML データ・ハンドラー以外の使用可能なデータ・ハンドラーを使用したり、カスタム・データ・ハンドラーを作成したりすることができます。この場合、そのデータ・ハンドラーをコネクターで使用できるように構成し、メッセージ・フローをそれに応じて変更する必要があります。使用可能なデータ・ハンドラーとカスタム・データ・ハンドラーの詳細については、「データ・ハンドラー・ガイド」を参照してください。データ・ハンドラーをアダプターで使用できるように構成する方法と、メッセージ・フローを変更する方法の詳細については、67 ページの『第 6 章 データ・ハンドラーの構成』を参照してください。

XML データ・ハンドラーとともに使用するビジネス・オブジェクトを構成するには、XML Object Discovery Agent (ODA) を使用します。XML ODA は、XML 文書の DTD またはスキーマ文書に基づいて、その XML 文書に対応するビジネス・オブジェクト定義を生成します。構成されたビジネス・オブジェクトは、アダプターで使用できます。WebSphere Business Integration Message Broker アダプターに付属しているサンプルでは、ビジネス・オブジェクトの変換に XML データ・ハンドラーを使用するように構成されているファイルが提供されます。

注: コネクターが処理するビジネス・オブジェクトは、統合ブローカーによって許可されている任意の名前を持つことができます。InterChange Server で許可されている命名規則の詳細については、「*Naming IBM WebSphere InterChange Server Components*」を参照してください。この資料は、次のサイトで入手できます。
http://www-306.ibm.com/software/integration/wbiadapters/library/infocenter/index.html

ビジネス・オブジェクトの作成のステップ

始める前に: コネクターが交換するメッセージとビジネス・オブジェクトのデータ・フォーマット (XML など) を確認する必要があります。

コネクターに適したビジネス・オブジェクトを作成するには、次のステップを実行します。

1. **Object Discovery Agent (ODA)** を使用してビジネス・オブジェクト定義を生成します (ODA が使用可能な場合)。例えば、XML メッセージを交換するように構成されているコネクターに適したビジネス・オブジェクトを作成するには、XML ODA を使用します。詳細については、「*データ・ハンドラー・ガイド*」を参照してください。
2. 必要に応じて、**Business Object Designer** を使用して、ODA が生成した情報に変更や追加情報を加えます。また、Business Object Designer を使用して XML ODA を実行および構成することもできます。詳細については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

注: 使用するデータ・フォーマットに対応する ODA がない場合には、Business Object Designer を使用してビジネス・オブジェクト定義を作成することができます。詳細については、「*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・オブジェクト定義を作成した後は、**Connector Configurator** を使用して、それらのビジネス・オブジェクト定義を、コネクターがサポートするビジネス・オブジェクト定義のリストに追加します。Connector Configurator の使用についての詳細は、67 ページの『第 6 章 データ・ハンドラーの構成』を参照してください。

ビジネス・オブジェクトの変更

ビジネス・オブジェクトを変更すると、各種処理機能を活用することができます。ビジネス・オブジェクトを変更するときは、Business Object Designer を使用します。詳細については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

このトピックでは、ビジネス・オブジェクト定義を変更して同期要求処理機能を使用可能にする作業の概要と手順を示します。

メッセージ選択子による応答メッセージのフィルター操作の概要

同期要求処理の場合にビジネス・オブジェクトを受信すると、コネクターは動詞のアプリケーション固有情報に `response_selector` ストリングが含まれているかどうかを検査します。`response_selector` が未定義の場合、コネクターは相関 ID を使用して応答メッセージを識別します (8 ページの『Retrieve、Exists、および Retrieve by content』を参照)。

`response_selector` が定義されている場合、コネクターは以下の構文を持つ名前と値のペアを期待します。

```
response_selector=JMSCorrelationID LIKE 'selectorstring'
```

メッセージ選択子 `selectorstring` は応答を一意的に識別する必要があります。また、値は以下に示すように単一引用符で囲む必要があります。

```
response_selector=JMSCorrelationID LIKE 'Oshkosh'
```

上記の例では、要求メッセージの発行後に、アダプターは「Oshkosh」と等しい correlationID を持つ応答メッセージの ReplyToQueue をモニターします。アダプターはこのメッセージ選択子に一致する最初のメッセージを検索し、これを応答としてディスパッチします。

メッセージ選択子による応答メッセージのフィルター操作のステップ

メッセージ選択子を使用して応答メッセージをフィルター操作するには、以下の手順を実行します。

1. **Business Object Designer** を起動します。
2. 同期メッセージ・デリバリーの対象として変換されるようにするビジネス・オブジェクトを開きます。
3. 動詞のアプリケーション固有情報に、「response_selector=JMSCorrelationID LIKE '*selectorstring*'」と指定します。
   selectorstring は、応答を一意的に識別するためのストリングです。*selectorstring* に指定されたストリングを相関 ID とする応答がアプリケーションから戻されると、コネクターはその応答を同期要求の応答として識別します。コネクターは、その後、データ・ハンドラーを呼び出してその応答を応答ビジネス・オブジェクトに変換し、要求元のコラボレーションに戻します。

   注: *selectorstring* には複数のストリングを指定することができます。また、特殊文字を指定したり、静的メタオブジェクトや動的メタオブジェクトの属性を参照したり、*selectorstring* に指定したストリングを置換できるようにしたりすることもできます。詳細については、以下の例を参照してください。

例: 必要に応じて、アダプターによる実行時置換を実行して、それぞれの要求ごとに固有のメッセージ選択子を生成することもできます。メッセージ選択子の代わりに、整数を中括弧で囲んだ形式でプレースホルダーを指定することもできます。例えば、'{1}' のようになります。次に、後ろにコロンを付け、置換に使用する属性をコンマで区切ってリストします。プレースホルダー内の整数は、置換に使用する属性の索引の役割を持ちます。以下のメッセージ選択子を例にとります。

```
response_selector=JMSCorrelationID LIKE '{1}': MyDynamicMO.CorrelationID
```

このメッセージ選択子は、アダプターに {1} を選択子の後ろの最初の属性の値 (この場合は、子オブジェクト MyDynamicMO の CorrelationId という属性) に置換するよう通知します。属性 CorrelationID の値が 123ABC であった場合、アダプターは以下の基準によって作成されたメッセージ選択子を生成および使用します。

```
JMSCorrelation LIKE '123ABC'
```

これによって応答メッセージを識別します。

また、以下のように複数の置換対象を指定することもできます。

```
response_selector=PrimaryId LIKE '{1}' AND AddressId LIKE '{2}' :
PrimaryId, Address[4].AddressId
```

この例では、アダプターは {1} をトップレベル・ビジネス・オブジェクトの PrimaryId 属性の値に置換し、{2} を子コンテナー・オブジェクト Address の 5 番目の位置の AddressId の値に置換します。この方法を使用すると、応答メッセージ選択子内にあるビジネス・オブジェクトおよびメタオブジェクトのどの属性でも参照できます。Address[4].AddressId を使用した詳細検索の実行方法に関する詳細は、JCDK API のマニュアル (getAttribute メソッド) を参照してください。

以下のいずれかの状況が発生した場合、実行時にエラーが報告されます。

- 「{}」シンボルの間に非整数を指定した場合
- 属性が定義されていない索引を指定した場合
- 指定された属性がビジネス・オブジェクトまたはメタオブジェクトに存在しない場合
- 属性パスの構文が誤っている場合

例えば、メッセージ選択子にリテラル値「{」または「}」を組み込む場合は、それぞれ「{{」または「{}」を使用できます。また、これらの文字を属性値に組み込むこともできます。その場合、最初の「{」は不要です。エスケープ文字を使用した例を以下に示します。

```
response_selector=JMSCorrelation LIKE '{1}' and CompanyName='A{{P':
MyDynamicMO.CorrelationID
```

コネクターはこのメッセージ選択子を以下のように解決します。

```
  JMSCorrelationID LIKE '123ABC' and CompanyName='A{P'
```

コネクターが属性値内で特殊文字 (「{」、「}」、「:」、または「;」など) を検出すると、これらの文字は照会ストリングに直接挿入されます。これにより、アプリケーション固有情報の区切り文字の役割も持つ特殊文字を照会ストリングに組み込むことができます。

次の例は、リテラル・ストリングの置換値が属性値からどのように抽出されるかを示します。

```
response_selector=JMSCorrelation LIKE '{1}' and CompanyName='A{{P':
MyDynamicMO.CorrelationID
```

MyDynamicMO.CorrelationID に値 {A:B}C;D が含まれていた場合、コネクターはメッセージ選択子を以下のように解決します。

```
JMSCorrelationID LIKE '{A:B}C;D' and CompanyName='A{P'
```

応答選択子コードに関する詳細は、JMS 1.0.1 仕様を参照してください。

メタオブジェクトの作成

このトピックでは、メタオブジェクトの作成の概要と手順を示します。

コネクターは、メタオブジェクト・エントリーを使用して、メッセージに関連付けるビジネス・オブジェクトを決定します。メッセージの処理に使用されるビジネス・オブジェクトのタイプと動詞は、WebSphere MQ メッセージのヘッダーに含まれる FORMAT フィールドによって決定されます。ビジネス・オブジェクト名と動詞

を格納するメタオブジェクト属性を構成し、WebSphere MQ メッセージ・ヘッダーの FORMAT フィールドのテキストに関連付けます。メタオブジェクト属性には、メッセージ処理のガイドラインも含まれます。

入力キューからメッセージが検索されると、コネクターは、FORMAT テキスト・フィールドに関連付けられているビジネス・オブジェクト名を調べます。次に、ビジネス・オブジェクト名とともに、メッセージがデータ・ハンドラーに渡されます。ビジネス・オブジェクトにメッセージの内容が正常に取り込まれると、コネクターはそのビジネス・オブジェクトがサブスクライブされているかどうかをチェックしてから、gotApplEvents() メソッドを使用して統合ブローカーにデリバリーします。

WebSphere Business Integration Message Broker 対応コネクターは、次の 2 種類のメタオブジェクトを認識および読み取りできます。

- 静的コネクター・メタオブジェクト
- 動的子メタオブジェクト

動的子メタオブジェクトの属性値は、静的メタオブジェクトの属性値を複写およびオーバーライドします。

ご自分の実装にはどちらのメタオブジェクトが最適であるかを判断する際には、以下のことを考慮してください。

- **静的メタオブジェクト**
  - 各種メッセージのメタデータがすべて固定されており、かつ構成時に指定可能である場合に役立ちます。
  - ビジネス・オブジェクト・タイプごとにしか値を指定できません。例えば、Customer タイプのオブジェクトのすべてが同一の宛先に送られます。
- **動的メタオブジェクト**
  - ビジネス・プロセスからメッセージ・ヘッダー内の情報にアクセスできます。
  - ビジネス・オブジェクト・タイプに関係なく、実行時にビジネス・プロセスでメッセージの処理を変更できます。例えば、動的メタオブジェクトを使用すると、アダプターに送信された Customer タイプのオブジェクトのそれぞれに、別々の宛先を指定することができます。
  - サポートするビジネス・オブジェクトの構造を変更する必要があります。この変更によって、マップとビジネス・プロセスの変更が必要になる場合があります。
  - カスタム・データ・ハンドラーに変更を加える必要があります。

メタオブジェクト・プロパティー

表 13 に、メタオブジェクトでサポートされるプロパティーをすべて含むリストを示します。メタオブジェクトを実装するときには、これらのプロパティーの説明を参照してください。メタオブジェクトには、表 13 に示すプロパティーが 1 つ以上含まれていなければなりません。

一部のプロパティーは、静的メタオブジェクトと動的メタオブジェクトのいずれかでしか使用できません。また、メッセージ・ヘッダーからの読み取りやメッセージ・ヘッダーへの書き込みが不可能なプロパティーもあります。特定のプロパティ

ーについて、コネクターでどのように解釈および使用されるかを判断するには、 1 ページの『第 1 章 概要』のイベント処理と要求処理に関する適切なセクションを参照してください。

表 13. Adapter for WebSphere Business Integration Message Broker のメタオブジェクト・プロパティー

プロパティー名	静的メタオブジェクトで定義可能か	動的メタオブジェクトで定義可能か	説明
DataHandlerConfigMO	はい	はい	構成情報を提供するためにデータ・ハンドラーに渡されるメタオブジェクトです。静的メタオブジェクトに指定されている場合、この値は DataHandlerConfigMO コネクター・プロパティーに指定されている値をオーバーライドします。このメタオブジェクト・プロパティーは、さまざまなタイプのビジネス・オブジェクトを処理するために複数の異なるデータ・ハンドラーが必要な場合に使用します。データ・フォーマットが実際のビジネス・データに依存する可能性がある場合は、要求処理に動的子メタオブジェクトを使用してください。指定するビジネス・オブジェクトは、コネクター・エージェントでサポートされるものでなければなりません。 101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』の説明を参照してください。
DataHandlerMimeType	はい	はい	使用すると、特定の MIME タイプに基づいたデータ・ハンドラーを要求できます。メタオブジェクトに指定されている場合、この値は DataHandlerMimeType コネクター・プロパティーに指定されている値をオーバーライドします。このメタオブジェクト・プロパティーは、さまざまなタイプのビジネス・オブジェクトを処理するために複数の異なるデータ・ハンドラーが必要な場合に使用します。データ・フォーマットが実際のビジネス・データに依存する可能性がある場合は、要求処理に動的子メタオブジェクトを使用してください。DataHandlerConfigMO に指定されているビジネス・オブジェクトに、このプロパティーの値に対応する属性が含まれていることが必要です。 101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』の説明を参照してください。
DataHandlerClassName	はい	はい	101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』の説明を参照してください。

表 13. Adapter for WebSphere Business Integration Message Broker のメタオブジェクト・プロパティー (続き)

プロパティー名	静的メタオブジェクトで定義可能か	動的メタオブジェクトで定義可能か	説明
InputFormat	はい	はい	特定のビジネス・オブジェクトに関連付けるインバウンド (イベント) メッセージのフォーマットまたはタイプです。この値は、メッセージ内容の識別に役立つものであり、メッセージを生成するアプリケーションによって決まります。検索されたメッセージがこのフォーマットである場合、そのメッセージは (可能であれば) 特定のビジネス・オブジェクトに変換されます。ビジネス・オブジェクトにこのフォーマットが指定されていない場合、コネクターは特定のビジネス・オブジェクトのサブスクリプション・デリバリーを処理しません。このプロパティーを設定するときは、デフォルトのメタオブジェクト変換プロパティーを使用しないでください。デフォルトのメタオブジェクト変換プロパティーの値は、着信メッセージをビジネス・オブジェクトに一致させるために使用されます。コネクターがメッセージのフォーマットを定義していると見なすフィールドは、コネクター固有のプロパティー MessageFormatProperty を使用してユーザーが定義できます。
OutputFormat	はい	はい	アウトバウンド・メッセージに取り込まれるフォーマットです。OutputFormat が指定されていない場合は、入力フォーマットが (利用可能であれば) 使用されます。
InputQueue	はい	はい	新規メッセージを検出するためにコネクターがポーリングする入力キューです。このプロパティーは、着信メッセージとビジネス・オブジェクトを一致させる目的でのみ使用されます。このプロパティーを設定するときは、デフォルトの変換プロパティーを使用しないでください。デフォルトの変換プロパティーの値は、着信メッセージをビジネス・オブジェクトに一致させるために使用されます。 **注:** コネクター固有のプロパティーにある InputQueue プロパティーは、アダプターがポーリングするキューを定義します。これは、アダプターがポーリングするキューを決定するのに使用する唯一のプロパティーです。静的 MO では、InputQueue プロパティーおよび InputFormat プロパティーは、アダプターが指定されたメッセージを特定のビジネス・オブジェクトにマップする条件として使用できます。この機能を実装するには、コネクター固有のプロパティーを使用して複数の入力宛先を構成し、さらに、必要に応じて、着信メッセージの入力フォーマットを基に各入力宛先に別個のデータ・ハンドラーをマップします。詳細については、60 ページの『データ・ハンドラーの入力キューへのマッピングの概要』を参照してください。
OutputQueue	はい	はい	特定のビジネス・オブジェクトから派生したメッセージが送信されるキューです。

表 13. Adapter for WebSphere Business Integration Message Broker のメタオブジェクト・プロパティー (続き)

プロパティー名	静的メタオブジェクトで定義可能か	動的メタオブジェクトで定義可能か	説明
ResponseTimeout	はい	はい	同期要求処理で応答を待機するときにタイムアウトとするまでの待機時間を、ミリ秒単位で示します。このプロパティーが未定義のままか、またはゼロよりも小さい値に設定されている場合、コネクターは応答を待機せず、SUCCESS を即時に戻します。
TimeoutFatal	はい	はい	同期要求処理で、応答の受信がないためコネクターからエラー・メッセージを戻す動作が開始されるときに使用されます。このプロパティーの値が True の場合、コネクターは、ResponseTimeout に指定されている時間内に応答の受信がなければ、APPRESPONSETIMEOUT をブローカーに戻します。このプロパティーが未定義の場合、または False に設定されている場合、コネクターは応答タイムアウトが発生すると要求を失敗させます。ただし、終了させることはありません。デフォルト値は False です。
DataEncoding	はい	はい	メッセージのタイプと、アダプターがビジネス・オブジェクト conversions.Possible の値 (text、binary、または object) に対して使用するエンコードを指定します。デフォルト = text。この属性の値のフォーマットは、messageType[:enc] です。例えば、Text:ISO8859_1、Text:UnicodeLittle、Text、Binary、または Object のようになります。このプロパティーは内部的に InputFormat プロパティーに関連します。InputFormat ごとに 1 つの DataEncoding のみを指定します。
以下に示すのは、JMS メッセージ・ヘッダーのみにマップされるフィールドです。具体的な説明や値の解釈などの詳細情報については、JMS API の仕様を参照してください。フィールドによっては、JMS プロバイダー間で解釈が異なることがあります。ご使用の JMS プロバイダーの資料を参照して、解釈の違いがないかどうかを確認してください。			
ReplyToDestination		はい	要求に対する応答メッセージの送信先となる宛先です。
Type		はい	メッセージのタイプです。通常はユーザーが定義できます (JMS プロバイダーによって異なります)。
MessageID		はい	メッセージの固有 ID です (特定の JMS プロバイダーでのみ有効)。
CorrelationID	はい	はい	応答メッセージで、その応答の開始理由にあたる要求メッセージの ID を示すために使用されます。
Delivery Mode	はい	はい	メッセージを MOM システム内で永続化するかどうかを指定します。許容値は次のとおりです。 1 = 非永続 2 = 永続 その他の値も使用できる場合があります (JMS プロバイダーによって異なります)。
Priority		はい	メッセージの優先順位を数値で表現したものです。許容値は 0 から 9 です (数値が大きいほど優先順位が高まります)。
Destination		はい	MOM システム内でのメッセージの現在の位置 (除去済みの場合は除去直前の位置) です。
Expiration		はい	メッセージの存続時間です。

表 13. Adapter for WebSphere Business Integration Message Broker のメタオブジェクト・プロパティー (続き)

プロパティー名	静的メタオブジェクトで定義可能か	動的メタオブジェクトで定義可能か	説明
Redelivered		はい	以前に JMS プロバイダーからクライアントに対してメッセージのデリバリーが試行されたが、受取の確認がなかった可能性が高いことを示すものです。
Timestamp		はい	メッセージが JMS プロバイダーに渡された時刻です。
UserID		はい	メッセージを送信したユーザーの ID です。
AppID		はい	メッセージを送信したアプリケーションの ID です。
DeliveryCount		はい	デリバリーを試行した回数です。
GroupID		はい	メッセージ・グループの ID です。
GroupSeq		はい	GroupID に指定されたメッセージ・グループにおける順位です。
JMSProperties		はい	63 ページの『JMS プロパティー』を参照してください。

バイナリー・メッセージとオブジェクト・メッセージの DataEncoding

メタオブジェクト・プロパティー DataEncoding を使用して、メッセージ・タイプを変更することができます。このプロパティーは、`text`、`binary`、または `object` の 3 つの値のいずれかを受け入れます。

デフォルトでは、アダプターは、すべてのメッセージがタイプ `text` であると想定します。バイナリー・メッセージを受信すると、アダプターは Java 仮想マシン (JVM) のデフォルト・エンコードを使用してバイナリー・コンテンツをテキストに変換してから、構成されたデータ・ハンドラーにコンテンツを渡します。メタオブジェクトの DataEncoding プロパティーで `binary` または `object` メッセージ・タイプを明示的に指定すると、この振る舞いは変わります。

- メタオブジェクトで DataEncoding プロパティーを使用して `binary` を指定すると、アダプターは次のように動作します。
  1. 要求処理中に、アダプターは、データ・ハンドラーのバイナリー・メソッドにビジネス・オブジェクトを渡し、バイト・メッセージを送信します。
  2. イベント通知中に、アダプターは、バイナリー・メッセージからバイトを受信し、それらを Java InputStream インスタンス (バイト) としてデータ・ハンドラーに渡します。
  3. テキスト・メッセージを受信すると、アダプターは JVM のデフォルト・エンコードを使用してテキスト本文をバイナリーに変換してから、データ・ハンドラーにコンテンツを渡します。
- メタオブジェクトで `object` DataEncoding 値を指定した場合、アダプターは次のように動作します。
  1. 要求処理中に、アダプターは、データ・ハンドラーの getStreamFromBO() メソッドにビジネス・オブジェクトを渡し、ObjectInputStream を取得します。
  2. イベント通知中に、アダプターは、オブジェクト・メッセージから Java オブジェクトを受信し、それらを Java ObjectInputStream インスタンス (バイト) としてデータ・ハンドラーに渡します。

注: すべてのデータ・ハンドラーが binary および object データをサポートしているわけではありません。構成したデータ・ハンドラーのサポートを確認してください。

静的メタオブジェクトの作成の概要

WebSphere Business Integration Message Broker の構成メタオブジェクトは、さまざまなビジネス・オブジェクトに定義される変換プロパティーのリストで構成されます。静的メタオブジェクトのサンプルを参照するには、Business Object Designer を起動し、アダプターに付属しているサンプル connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥Sample_WBIMB_MO_Config.xsd を開きます。

コネクターがサポートする静的メタオブジェクトは、常に 1 つだけです。静的メタオブジェクトを実装するには、そのメタオブジェクトの名前を、コネクター・プロパティー ConfigurationMetaObject に指定します。

静的メタオブジェクトは、各属性が、それぞれ 1 つのビジネス・オブジェクトと動詞の組み合わせを、そのオブジェクトの処理に関連するメタデータのすべてとともに示す構造になっています。各属性の名前は、Customer_Create のように、ビジネス・オブジェクト・タイプと動詞の間を下線で区切った名前にする必要があります。属性のアプリケーション固有情報には、このオブジェクトと動詞の固有の組み合わせに対して指定するメタデータ・プロパティーを表す名前と値のペアを、セミコロンで区切って 1 つ以上含める必要があります。

表 14. 静的メタオブジェクト構造

属性名	アプリケーション固有のテキスト
<business object type>_<verb>	property=value;property=value;...
<business object type>_<verb>	property=value;property=value;...

例えば、次のようなメタオブジェクトがあるとします。

表 15. 静的メタオブジェクト構造のサンプル

属性名	アプリケーション固有の情報
Customer_Create	OutputFormat=CUST;OutputDestination=QueueA
Customer_Update	OutputFormat=CUST;OutputDestination=QueueB
Order_Create	OutputFormat=ORDER;OutputDestination=QueueC

このサンプルのメタオブジェクトは、タイプが Customer で動詞が Create の要求ビジネス・オブジェクトを受信したときは、そのオブジェクトをフォーマットが CUST のメッセージに変換して宛先 QueueA に入れる必要があることを、コネクターに対して通知します。Customer タイプのオブジェクトの動詞が Update である場合、変換後のメッセージは QueueB に入れられます。オブジェクトのタイプが Order で動詞が Create である場合には、コネクターはそのオブジェクトをフォーマットが ORDER のメッセージに変換し、QueueC にデリバリーします。それ以外のビジネス・オブジェクトがコネクターに渡された場合には、サブスクライブされていないものとして処理されます。

また、Default 属性を 1 つ指定して、アプリケーション固有情報 (ASI) のプロパティーを 1 つ以上割り当てることもできます。このデフォルト属性のプロパティーと、オブジェクトと動詞の組み合わせごとの属性のプロパティーが結合されて、メタオブジェクトの属性の最終的なプロパティーになります。オブジェクトと動詞の組み合わせに関係なく汎用的に適用するプロパティーが 1 つ以上ある場合には、このデフォルト属性を使用すると便利です。次の例の場合、コネクターは、オブジェクトと動詞の組み合わせ Customer_Create と Order_Create については、それぞれに個別に指定されているメタオブジェクト・プロパティーに加えて OutputDestination=QueueA が指定されていると見なします。

表 16. 静的メタオブジェクト構造のサンプル

属性名	アプリケーション固有の情報
Default	OutputDestination=QueueA
Customer_Update	OutputFormat=CUST
Order_Create	OutputFormat=ORDER

静的メタオブジェクトにアプリケーション固有情報として指定できるプロパティーは、55 ページの表 13 に記載されています。

注: 静的メタオブジェクトが指定されていない場合、コネクターはポーリング時にメッセージ・フォーマットを特定のビジネス・オブジェクト・タイプにマップできません。そのような場合、コネクターは、ビジネス・オブジェクトを指定せずに、構成されているデータ・ハンドラーにメッセージ・テキストを渡します。データ・ハンドラーがテキストのみに基づいてビジネス・オブジェクトを作成することができない場合、コネクターはこのメッセージ・フォーマットを認識できないことを示すエラーを報告します。

静的メタオブジェクトの作成のステップ

静的メタオブジェクトを実装するには、以下の手順を実行します。

1. Business Object Designer を起動します。詳細については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。
2. メタオブジェクトのサンプル

   connectors¥WebSphereBIMessageBroker¥samples¥LegacyItem¥Sample_WBIMB_MO_Config.xsd

   を開きます。
3. 必要な要件が反映されるように属性と ASI を編集して (55 ページの表 13 を参照)、メタオブジェクト・ファイルを保管します。
4. 保管したメタオブジェクト・ファイルの名前を、コネクター固有のプロパティー ConfigurationMetaObject の値に指定します。

データ・ハンドラーの入力キューへのマッピングの概要

静的メタオブジェクトのアプリケーション固有情報内の InputQueue プロパティーを使用して、データ・ハンドラーを入力キューに関連付けることができます。この機能は、フォーマットや型変換の要件が異なる複数の取引先との取引を行なう際に役立ちます。

データ・ハンドラーの入力キューへのマッピングのステップ

データ・ハンドラーを InputQueue にマップするには、以下の手順を実行します。

1. コネクター固有のプロパティー (105 ページの『InputQueue』を参照) を使用して、1 つ以上の入力キューを構成します。
2. **Business Object Designer** で、適切な静的メタオブジェクトを開きます。
3. 静的メタオブジェクトのそれぞれの入力キューについて、アプリケーション固有情報に、キュー・マネージャー、入力キュー名、データ・ハンドラーのクラス名、および MIME タイプを指定します。

例えば、次に示す静的メタオブジェクト内の属性は、データ・ハンドラーを CompReceipts という名前の InputQueue に関連付けます。

```
[Attribute]
Name = Cust_Create
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo =
InputQueue=//queue.manager/CompReceipts;DataHandlerClassName=
com.crossworlds.DataHandlers.WBIMB.disposition_notification;DataHandlerMime
Type=
message/
disposition_notification
IsRequiredServerBound = false
[End]
```

動的子メタオブジェクトの作成の概要

静的メタオブジェクトを使用して必要なメタデータを指定することが困難または実行不可能な場合には、実行時に、ビジネス・オブジェクト・インスタンスごとにデリバリーされるメタデータをコネクターで受信することもできます。

動的メタオブジェクトを使用すると、要求処理では、コネクターがビジネス・オブジェクトの処理に使用するメタデータを要求ごとに変更できるようになります。また、イベント処理では、イベント・メッセージに関する情報の検索が可能になります。

コネクターは、コネクターに渡されるトップレベルのビジネス・オブジェクトに子として追加された動的メタオブジェクトから、変換プロパティーを認識して読み取ります。動的子メタオブジェクトの属性値は、コネクターの構成に使用される静的メタオブジェクトによって指定できる変換プロパティーを複写します。

動的子メタオブジェクト・プロパティーは静的メタオブジェクト・プロパティーの値をオーバーライドするため、動的子メタオブジェクトを指定する場合は、静的メタオブジェクトを指定するコネクター・プロパティーを含める必要はありません。したがって、動的子メタオブジェクトは静的メタオブジェクトから独立して使用でき、また、静的メタオブジェクトは動的子メタオブジェクトから独立して使用できます。

動的メタオブジェクトにアプリケーション固有情報として指定できるプロパティーは、 55 ページの表 13 に記載されています。

以下の属性は JMS および WebSphere MQ ヘッダー・プロパティーを反映しており、動的メタオブジェクトで認識されます。

表 17. 動的メタオブジェクト・ヘッダー属性

ヘッダー属性名	モード	対応する JMS ヘッダー
CorrelationID	読み取り/書き込み	JMSCorrelationID
ReplyToQueue	読み取り/書き込み	JMSReplyTo
DeliveryMode	読み取り/書き込み	JMSDeliveryMode
Priority	読み取り/書き込み	JMSPriority
Destination	読み取り	JMSDestination
Expiration	読み取り	JMSExpiration
MessageID	読み取り	JMSMessageID
Redelivered	読み取り	JMSRedelivered
TimeStamp	読み取り	JMSTimeStamp
Type	読み取り	JMSType
UserID	読み取り	JMSXUserID
AppID	読み取り	JMSXAppID
DeliveryCount	読み取り	JMSXDeliveryCount
GroupID	読み取り	JMSXGroupID
GroupSeq	読み取り	JMSXGroupSeq
JMSProperties	読み取り/書き込み	

読み取り専用属性は、イベント通知中にメッセージ・ヘッダーから読み取られ、動的メタオブジェクトに書き込まれます。これらのプロパティーは、要求処理中に応答メッセージが発行されたときに動的メタオブジェクトも設定します。読み取り/書き込み属性は、要求処理中に作成されるメッセージ・ヘッダーで設定されます。イベント通知中は、読み取り/書き込み属性はメッセージ・ヘッダーから読み取られ、動的メタオブジェクトを設定します。

動的メタオブジェクトは、各属性がそれぞれ 1 つのメタデータ・プロパティーと値を meta-object property name =meta-object property value の形式で表す構造になっています。

注: IBM WebSphere の標準のデータ・ハンドラーは、いずれも、cw_mo_ タグを認識すると、その後に指定されている動的メタオブジェクトを表す属性をビジネス・データ用の属性として処理しないように設計されています。アダプターで使用するカスタム・データ・ハンドラーを開発するときは、同様に設計する必要があります。

ポーリング時の動的子メタオブジェクトの取り込み

ポーリング時に検索されたメッセージに関してより多くの情報を持つコラボレーションを提供するために、コネクターは動的メタオブジェクトの特定の属性を (作成されるビジネス・オブジェクトにすでに定義されていれば) 取り込みます。

表 18 に、動的子メタオブジェクトがポーリングのためにどのように構成されるかを示します。

表 18. ポーリング用の動的子メタオブジェクトの構造

プロパティー名	サンプル値
InputFormat	CUST_IN
InputQueue	MYInputQueue
OutputFormat	CxIgnore
OutputQueue	CxIgnore
ResponseTimeout	CxIgnore
TimeoutFatal	CxIgnore

表 18 に示すように、動的子メタオブジェクトに、追加属性 InputFormat および InputQueue を定義することができます。InputFormat には検索されたメッセージのフォーマットが取り込まれ、InputQueue 属性には特定のメッセージが検索されたキューの名前が含まれます。これらのプロパティーが子メタオブジェクトで定義されていない場合は、これらのプロパティーは取り込まれません。

シナリオの例:

- コネクターは、フォーマット CUST_IN のメッセージをキュー MyInputQueue から検索します。
- コネクターはこのメッセージを Customer ビジネス・オブジェクトに変換し、アプリケーション固有のテキストをチェックして、メタオブジェクトが定義されているかどうかを判別します。
- メタオブジェクトが定義されている場合、コネクターはそのメタオブジェクトのインスタンスを作成し、それに従って InputQueue 属性と InputFormat 属性を取り込み、利用可能なコラボレーションにビジネス・オブジェクトを発行します。

JMS ヘッダーと動的子メタオブジェクトの属性

動的メタオブジェクトに属性を追加すると、メッセージ・トランスポートに関する詳細情報を取得したり、メッセージ・トランスポートの管理を強化したりできます。このセクションでは、これらの属性、および属性がイベント通知と要求処理に及ぼす影響について説明します。

JMS プロパティー: 動的メタオブジェクトの他の属性とは異なり、JMSProperties には単一カーディナリティーの子オブジェクトを定義しなければなりません。この子オブジェクトの各属性には、JMS メッセージ・ヘッダーの可変部分を読み取り元/書き込み先とするプロパティーを、以下の規則に従って 1 つずつ定義する必要があります。

1. 属性名にはセマンティック値を含めないこと。
2. 属性の型は、JMS プロパティーの型に関係なく、必ず String にすること。
3. 属性のアプリケーション固有情報には、その属性をマップする JMS メッセージ・プロパティーの名前とフォーマットを定義した 2 組の名前と値のペアを含めること。名前はユーザーが定義できます。値の型は次のいずれかでなければなりません。
   - Boolean
   - String
   - Int
   - Float

- Double
- Long
- Short
- Byte

次の表に、JMSProperties オブジェクトの各属性に定義する必要があるアプリケーション固有情報のプロパティーを示します。

表 19. JMS プロパティー属性のアプリケーション固有情報

属性	指定可能な値	ASI	コメント
Name	任意の有効な JMS プロパティー名 (有効とは、プロパティーの型と ASI で定義した型が矛盾しないこと)	name=<*JMS プロパティー名*>;type=<*JMS プロパティーの型*>	一部のベンダーは、特定のプロパティーを機能拡張のために予約しています。一般に、ユーザーは、そのようなベンダー固有の機能を使用する場合を除いて、JMS で始まる名前のカスタム・プロパティーを定義してはいけません。
Type	String	type=<*コメントを参照*>	JMS プロパティーの型です。JMS API には、JMS メッセージ内の値を設定するためのメソッドがいくつか用意されています (setIntProperty 、setLongProperty、setStringProperty など)。ここで指定する JMS プロパティーの型は、メッセージ内のそのプロパティーの値を設定するときにこれらのメソッドのうちのどれを使用すればよいかを示します。

次の例では、Customer オブジェクトに JMSProperties 子オブジェクトを定義して、メッセージ・ヘッダーのユーザー定義フィールドにアクセスできるようにしています。

```
Customer (ASI = cw_mo_conn=MetaData)
  |-- Id

  |-- FirstName

  |-- LastName
```

```
|-- ContactInfo
|-- MetaData
      |-- OutputFormat = CUST
      |-- OutputDestination = QueueA
      |-- JMSProperties
           |-- RoutingCode = 123 (ASI= name=RoutingCode;type=Int)
           |-- Dept = FD (ASI= name=RoutingDept;type=String)
```

もう 1 つの例として、図 4 に、動的メタオブジェクトに含まれる属性 JMSProperties と、JMS メッセージ・ヘッダーの 4 つのプロパティー (ID、GID、RESPONSE、および RESPONSE_PERSIST) の定義を示します。これらの属性のアプリケーション固有情報には、名前と型がそれぞれ定義されています。例えば、属性 ID は、型が String の JMS プロパティー ID にマップされます。

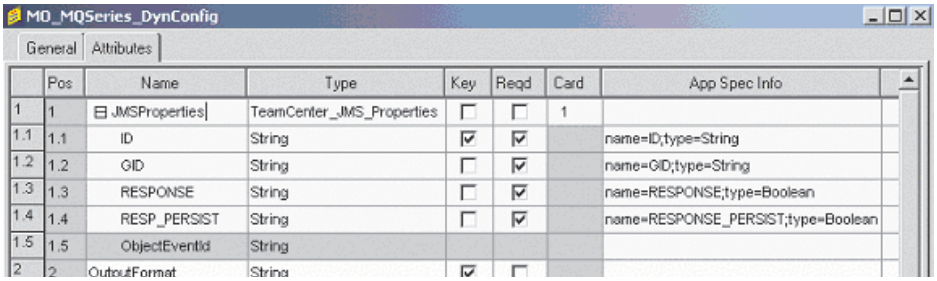

図 4. 動的メタオブジェクトの JMS プロパティー属性

動的メタオブジェクトの作成のステップ

動的メタオブジェクトを実装するには、以下の手順を実行します。

1. **Business Object Designer** を起動します。詳細については、「ビジネス・オブジェクト開発ガイド」を参照してください。
2. 処理するときに動的メタオブジェクトを作用させる必要があるトップレベル・ビジネス・オブジェクトを開きます。
3. 開いたトップレベル・オブジェクトに動的メタオブジェクトを子オブジェクトとして追加し、名前と値のペア cw_mo_conn=*<MO attribute>* をそのトップレベル・オブジェクトの ASI に追加します。ここで、*<MO attribute>* は、そのトップレベル・オブジェクトに含まれる、動的メタオブジェクトを表現する属性の名前です。例えば、次のようにします。

   ```
   Customer (ASI = cw_mo_conn=MetaData)
     |-- Id
     |-- FirstName
     |-- LastName
     |-- ContactInfo
   ```

```
|-- MetaData
      |-- OutputFormat = CUST
      |-- OutputDestination = QueueA
```

コネクターは、上の定義にあてはまる要求を受信すると、その要求 (Customer オブジェクト) をフォーマットが CUST のメッセージに変換し、キュー QueueA にそのメッセージを入れます。

4. トップレベル・ビジネス・オブジェクトを保管します。

注: 複数のビジネス・オブジェクトで同じ動的メタオブジェクトを使用することも、それぞれで異なる動的メタオブジェクトを使用することもできます。また、動的メタオブジェクトを一切使用しなくてもかまいません。

第 6 章 データ・ハンドラーの構成

- 『データ・ハンドラーの構成の概要』
- 『データ・ハンドラーの指定』
- 68 ページの『メッセージ・フローの変更』

この章では、データ・ハンドラーの構成方法について説明します。

データ・ハンドラーの構成の概要

データ・ハンドラーはコネクターの中心的なコンポーネントです。コネクターは、データ・ハンドラーを呼び出して、ビジネス・オブジェクトからメッセージへの変換とメッセージからビジネス・オブジェクトへの変換を行います。

コネクター固有のデータ・ハンドラー・プロパティーに含まれる情報は、これらの変換において重要な役割を果たします。この情報の構成は、製品ファイルをインストールした後、アダプターを始動する前に行います。

注: 構成したデータ・ハンドラーは、入力キューにマップすることもできます。詳細と手順については、 60 ページの『データ・ハンドラーの入力キューへのマッピングの概要』を参照してください。

この章で説明する作業

この章で説明する作業は、以下のとおりです。

表 20. データ・ハンドラーの構成: 作業ロードマップ

作業	関連手順 (参照先)	詳細情報 (参照先)
データ・ハンドラーの指定	『データ・ハンドラーの指定』	
メッセージ・フローの変更	68 ページの『メッセージ・フローの変更』	ご使用のメッセージ・ブローカーの資料

データ・ハンドラーの指定

このトピックでは、データ・ハンドラーの指定の概要と手順を示します。

データ・ハンドラーの指定の概要

データ・ハンドラーを構成するには、以下のコネクター固有の構成プロパティーに値を指定します。

- DataHandlerClassName
- DataHandlerConfigMO
- DataHandlerMimeType

これらのプロパティーの詳細については、 101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』を参照してください。

データ・ハンドラーの指定のステップ

始める前に: XML データ・ハンドラーを使用する場合は、データ・ハンドラー関連のコネクター構成プロパティーのデフォルト値をそのまま使用するだけで済みます。該当する 3 つのプロパティー (DataHandlerClassName、DataHandlerConfigMO、および DataHandlerMimeType) は、いずれもデフォルトで XML データ・ハンドラー用に構成されています。

データ・ハンドラーを構成するには、以下の手順を実行します。

1. **Connector Configurator** で、「**コネクター固有プロパティー**」タブをクリックします。
2. DataHandlerClassName に、構成するデータ・ハンドラーに対応するプロパティー値を指定します。
3. DataHandlerConfigMO に、構成するデータ・ハンドラーに対応するプロパティー値を指定します。
4. DataHandlerMimeType に、構成するデータ・ハンドラーに対応するプロパティー値を指定します。
5. Connector Configurator でプロパティーを適用します。

注: 静的メタオブジェクトまたは動的メタオブジェクトで上記のコネクター固有のプロパティーと同名のプロパティーに値を指定すると、上記のプロパティーに指定した値よりもそれらの値が優先されます。詳細については、54 ページの『メタオブジェクト・プロパティー』を参照してください。

メッセージ・フローの変更

このトピックでは、メッセージ・フローの変更の概要と手順を示します。

メッセージ・フローの変更の概要

WebSphere Business Integration Adapter は、メッセージ・ブローカーを使用する場合、WebSphere MQ メッセージ・フローを使用してデータの処理とルーティングを行います。キューごとに 1 つずつ定義されるメッセージ・フローは、対応するキューに入れられたメッセージのすべてを処理します。Message Brokers Toolkit を使用すると、メッセージ・フローを構成して、処理する予定のメッセージのタイプごとに異なる処理ステップを指定することができます。

各着信メッセージが構成されたデータ・ハンドラーに対応するフォーマットに変換されるように、メッセージ・フローを変更する必要があります。この変換は、メッセージがコネクターの入力キューに送信される前に行われる必要があります。下記の手順は、メッセージ・フローを XML 対応の構成に変更する方法を示しています。メッセージ・フローを他のデータ・フォーマットに対応させるために変更する場合は、ステップ 2 の 3 番目の箇条書き項目の `XML` を、そのフォーマットを表す MIME タイプで置き換えてください。

メッセージ・フローを XML 対応の構成に変更するステップ

始める前に: XML データ・ハンドラーを構成します。

メッセージ・フローを変更して XML 対応にするには、以下の手順を実行します。

1. **Message Brokers Toolkit** を起動します。
2. メッセージ・フローの終わりに計算ノードを追加します。
3. ESQL テキスト領域に、以下のようにフィールドを入力します。

   ```
   Set OutputRoot = InputRoot;
   ```

   これにより、メッセージが出力のためにコピーされます。

   ```
   Set OutputRoot.MQHRF2.Format = 'SO-CR';
   ```

   これにより、コネクターがこのフォーマットをチェックして、メッセージを適切に変換することが保証されます。

   ```
   SET OutputRoot.Properties.MessageFormat = 'XML';
   ```

   これは、メッセージがデリバリー時に XML に変換される必要があることを WebSphere Business Integration Message Broker に示します。
4. 「**適用**」をクリックして計算ノードを使用可能にします。

図 5 に、着信メッセージをコネクターが認識できるフォーマットに変換するように構成された計算ノードのサンプル・ビューを示します。この計算ノードが使用可能になった後は、オリジナルのメッセージを表す XML 文書がコネクターの入力キューに送信されます。

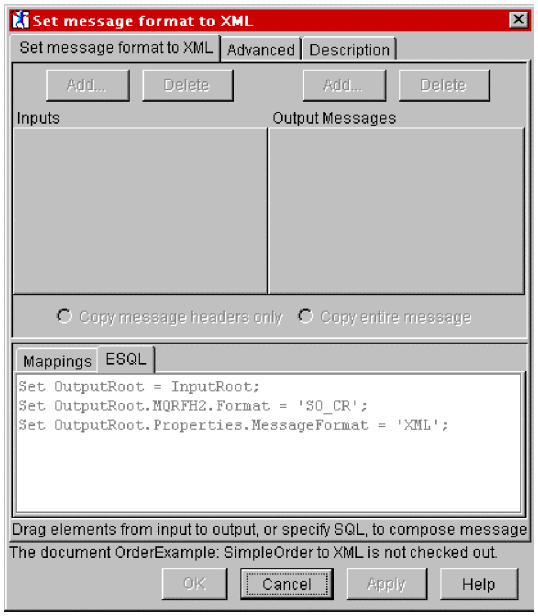

図 5. メッセージ・フォーマットを XML に設定

注: WebSphere Business Integration Message Broker Repository Manager 内でカスタム・フォーマットを定義している場合は、メッセージ・フォーマットを XML に設定するだけで既存のフォーマットを XML に変換できます。このフォーマットは、MQHRF2 とは異なります。`OutputRoot.Properties.MessageFormat` は MRM に関するものですが、`OutputRoot.MQHRF2.Format` はメッセージを受信するアプリケーションのメッセージ・フォーマットを指定するために使用されます。

第 7 章 トラブルシューティング

この章では、コネクターを始動または実行するときに発生する可能性がある問題について説明します。また、IBM のサポートを受ける方法についても説明します。

始動時の問題のトラブルシューティング

問題	考えられる処置/説明
初期化中にコネクターが予期しないエラーでシャットダウンし、次のメッセージが報告されました: `Exception in thread "main" java.lang.NoClassDefFoundError: javax/jms/JMSException...`	コネクターは、WebSphere MQ Java クライアント・ライブラリーからファイル `jms.jar` を見つけることができません。`start_connector.bat` 内の変数 `MQSERIES_JAVA_LIB` が WebSphere MQ Java クライアント・ライブラリー・フォルダーを指していることを確認します。
初期化中にコネクターが予期しないエラーでシャットダウンし、次のメッセージが報告されました: `Exception in thread "main" java.lang.NoClassDefFoundError: com/ibm/mq/jms/MQConnectionFactory...`	コネクターは、WebSphere MQ Java クライアント・ライブラリーからファイル `com.ibm.mqjms.jar` を見つけることができません。`start_connector.bat` 内の変数 `MQSERIES_JAVA_LIB` が WebSphere MQ Java クライアント・ライブラリー・フォルダーを指していることを確認します。
初期化中にコネクターが予期しないエラーでシャットダウンし、次のメッセージが報告されました: `Exception in thread "main" java.lang.NoClassDefFoundError: javax/naming/Referenceable...`	コネクターは、WebSphere MQ Java クライアント・ライブラリーからファイル `jndi.jar` を見つけることができません。`start_connector.bat` 内の変数 `MQSERIES_JAVA_LIB` が WebSphere MQ Java クライアント・ライブラリー・フォルダーを指していることを確認します。
初期化中にコネクターが予期しないエラーでシャットダウンし、次の例外が報告されました: `java.lang.UnsatisfiedLinkError: no mqjbnd01 in shared library path`	コネクターは、IBM MQSeries Java クライアント・ライブラリーから必要なランタイム・ライブラリー (`mqjbnd01.dll` [Windows] または `libmqjbnd01.so` [UNIX]) を見つけることができません。パスに WebSphere MQ Java クライアント・ライブラリーのフォルダーが含まれていることを確認します。
コネクターから次の例外が報告されました: `MQJMS2005: failed to create MQQueueManager for ':'`	次のプロパティーの値を明示的に設定します: `HostName`、`Channel`、および `Port`。

イベント処理のトラブルシューティング

問題	考えられる処置/説明
コネクターは、MQRFH2 ヘッダーを持つすべてのメッセージをデリバリーします。	MQMD WebSphere MQ ヘッダーを持つメッセージのみをデリバリーするには、出力キューの URI の名前に `?targetClient=1` を付加します。例えば、メッセージをキュー `queue: //my.queue.manager/OUT` に出力する場合は、URI を `queue: //my.queue.manager/OUT?targetClient=1` に変更します。詳細は、11 ページの『第 2 章 コネクターのインストール』を参照してください。
コネクターは、コネクター・メタオブジェクト内でのメッセージ・フォーマットの定義にかかわらず、デリバリー時にすべてのメッセージ・フォーマットの 8 文字を超える部分を切り捨てます。	これは WebSphere MQ MQMD メッセージ・ヘッダーの制限であり、コネクターの制限ではありません。
pollForEvents 時に、次の JMS 例外が報告された後でコネクターがシャットダウンしました: `MQJMS1000: Failed to create JMS message`	このエラーは MQ Java API に関するエラーであり、コネクター自身のエラーではありません。このエラーは、多くの場合、WebSphere Business Integration Message Broker 製品自体と同じマシン上でコネクターを実行しているときに発生します。この問題を解決するには、以下の手順を実行する必要があります。 1. ファイル `Product_Dir¥Dependencies¥JRE_122_4.zip` をフォルダー *Product_Dir*`¥connectors¥WBIMB¥Dependencies¥jre_122_Rel4` に unzip します。 2. Open `Product_Dir¥connectors¥WBIMB¥start_MQSIV2.bat` を開き、次の 2 つの行のコメントをはずします。`oset PATH=%CONNDIR%¥Dependencies¥jre_122_Rel4¥bin...oset JAVA=%CONNDIR%¥Dependencies¥jre_122_Rel4¥lib¥rt.jar` 3. コネクターを再始動します。
`pollForEvents` 時に、次の JMS 例外が報告された後でコネクターがシャットダウンしました: `MQJMS1052: Unrecognised [sic] JMS Message class`	このエラーは、JMS 準拠のアプリケーションから発生したメッセージを WebSphere Business Integration Message Broker が変更したことによって発生することがあります。このエラーを訂正するには、WebSphere Business Integration Message Broker 計算ノードに次の SQL ステートメントを追加することにより、問題が発生したメッセージから残りの JMS 情報を除去します: `SET OutputRoot.MQRFH2.jms = null;`

サポートを受ける方法

始める前に: 本書の発行後に公開されたテクニカル・サポートの技術情報や速報に、本書の対象製品に関する重要な情報が記載されている場合があります。これらの技術情報や速報は、WebSphere Business Integration のサポート Web サイトで参照できます。

WebSphere Business Integration のサポート Web サイトを利用するには、以下の手順を実行します。

1. http://www.ibm.com/software/integration/websphere/support/ にアクセスします。
2. 適切なコンポーネント領域を選択し、「Technotes」セクションと「Flashes」セクションをブラウズするか検索します。

付録 A. コネクターの標準構成プロパティー

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ー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 (Connector Configurator では WMQI) と総称)
- Information Integrator (II)
- WebSphere Application Server (WAS)

ご使用のアダプターが DB2 Information Integrator をサポートしている場合は、WMQI オプションおよび DB2 II 標準プロパティーを使用します (75 ページの表 21 の「注」列を参照してください)。

アダプターに対して設定するプロパティーは、どの統合ブローカーを使用するかによって異なります。統合ブローカーは Connector Configurator を使用して選択します。ブローカーを選択すると、Connector Configurator は、アダプターに対して構成する必要のある標準プロパティーをリストします。

このコネクター固有のプロパティーの詳細については、本書の該当するセクションを参照してください。

新規プロパティー

このリリースでは、次の標準プロパティーが追加されました。

- BOTrace

標準コネクター・プロパティーの概要

コネクターには 2 つのタイプの構成プロパティーがあります。

- フレームワークによって使用される標準構成プロパティー
- エージェントによって使用されるアプリケーション、またはコネクター固有の構成プロパティー

これらのプロパティーによって、アダプターのフレームワークとエージェントの実行時の振る舞いが決まります。

このセクションでは、Connector Configurator の開始方法とすべてのプロパティーに共通する特性を説明します。コネクター固有の構成プロパティーの詳細は、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 の始動

Connector Configurator からコネクター・プロパティーを構成します。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、本書の Connector Configurator に関するセクションを参照してください。

Connector Configurator と System Manager は、Windows システム上でのみ動作します。コネクターが UNIX システム上で動作している場合は、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ーのコネクター・プロパティーを設定する場合は、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、コネクター用の Connector Configurator を開く必要があります。

構成プロパティー値の概要

コネクターは、以下の順序に従ってプロパティーの値を決定します。

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

プロパティーのフィールド長のデフォルトは 255 文字です。 STRING プロパティー・タイプの長さには制限がありません。INTEGER タイプの長さは、アダプターが稼働しているサーバーによって決まります。

コネクターは、始動時に構成値を取得します。実行時セッション中に 1 つ以上のコネクター・プロパティーの値を変更する場合は、プロパティー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。

プロパティーの更新特性、つまり、コネクター・プロパティーへの変更がいつどのように有効になるかは、プロパティーの性質によって決まります。

標準コネクター・プロパティーには、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、新規値が即時に有効になります。ただし、例えば WebSphere Message Broker で稼働している場合など、コネクターがスタンドアロン・モードである (System Manager から独立している) 場合は、構成ファイルでのみプロパティーを変更できます。この場合、動的更新は実行できません。
- **エージェント再始動 (ICS のみ)**
  コネクター・エージェントを停止して再始動しなければ、新規値が有効になりません。
- **コンポーネント再始動**
  System Manager でコネクターを停止してから再始動しなければ、新規値が有効になりません。エージェントまたはサーバー・プロセスを停止、再始動する必要はありません。

- **システム再始動**
  コネクター・エージェントおよびサーバーを停止して再始動しなければ、新規値が有効になりません。

特定のプロパティー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、75 ページの表 21 の「更新メソッド」列を参照してください。

標準プロパティーが常駐できるロケーションは 3 つあります。複数のロケーションに常駐できるプロパティーもあります。

- **ReposController**
  プロパティーはコネクター・コントローラーに常駐し、そこでのみ有効になります。エージェント側で値を変更した場合、コントローラーには影響しません。
- **ReposAgent**
  プロパティーはエージェントに常駐し、そこでのみ有効になります。プロパティーによっては、ローカル構成がこの値よりも優先されます。
- **LocalConfig**
  プロパティーはコネクターの構成ファイルに常駐し、構成ファイルを通じてのみ動作できます。コントローラーは、プロパティーの値を変更することができず、システムを再配置してコントローラーを明示的に更新するまで、構成ファイルに加えられた変更にも気付きません。

標準プロパティーのクイック・リファレンス

表 21 は、標準コネクター構成プロパティーのクイック・リファレンスです。すべてのコネクターがこれらのプロパティーのすべてを必要とするわけではありません。また、使用する統合ブローカーによってプロパティーの設定が異なる可能性があります。

各プロパティーの説明は、表の後のセクションを参照してください。

注: 表 21の「注」列にある「RepositoryDirectory が <REMOTE> に設定され」という句は、ブローカーが InterChange Server であることを示しています。ブローカーが WMQI または WAS の場合には、リポジトリー・ディレクトリーは <*ProductDir*>¥repository に設定されます。

表 21. 標準構成プロパティーの要約

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
AdapterHelpName	有効な <*RegionalSetting*> ディレクトリーを含む <*ProductDir*>¥bin¥Data¥App¥Help¥ 内の有効なサブディレクトリーの 1 つ	テンプレート名 (有効な場合)、またはブランク・フィールド	コンポーネント再始動	サポートされる地域設定。chs_chn、cht_twn、deu_deu、esn_esp、fra_fra、ita_ita、jpn_jpn、kor_kor、ptb_bra、および enu_usa (デフォルト) が含まれます。
AdminInQueue	有効な JMS キュー名	*<CONNECTORNAME>*/ADMININQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
AdminOutQueue	有効な JMS キュー名	*<CONNECTORNAME>* /ADMINOUTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
AgentConnections	1 から 4	1	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が MQ または IDL で、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
AgentTraceLevel	0 から 5	0	ブローカーが ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
ApplicationName	アプリケーション名	コネクター・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	
BiDi.Application	次の双方向属性の有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポーネント再始動	このプロパティーは、BiDi.Transformation の値が true である場合のみ有効です。
BiDi.Broker	次の双方向属性の有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポーネント再始動	このプロパティーは、BiDi.Transformation の値が true である場合のみ有効です。BrokerType の値が ICS である場合、プロパティーは読み取り専用です。
BiDi.Metadata	次の双方向属性の有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポーネント再始動	このプロパティーは、BiDi.Transformation の値が true である場合のみ有効です。
BiDi.Transformation	true または false	false	コンポーネント再始動	このプロパティーは、BrokerType の値が WAS ではない場合のみ有効です。
BOTrace	none または keys または full	none	エージェント再始動	このプロパティーは、AgentTraceLevel の値が 5 より低い場合のみ有効です。
BrokerType	ICS、WMQI、WAS	ICS	コンポーネント再始動	

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
CharacterEncoding	サポートされる任意のコード。リストには次のサブセットが表示されます。ascii7、ascii8、SJIS、Cp949、GBK、Big5、Cp297、Cp273、Cp280、Cp284、Cp037、Cp437	ascii7	コンポーネント再始動	このプロパティーは、C++ コネクターに対してのみ有効です。
CommonEventInfrastructure	true または false	false	コンポーネント再始動	
CommonEventInfrastructureURL	例えば corbaloc:iiop:host:2809 などの URL ストリング	デフォルト値なし。	コンポーネント再始動	このプロパティーは、CommonEvent Infrastructure の値が true である場合のみ有効です。
ConcurrentEventTriggeredFlows	1 から 32,767	1	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ContainerManagedEvents	ブランクまたは JMS	ブランク	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
ControllerEventSequencing	true または false	true	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ControllerStoreAndForwardMode	true または false	true	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ControllerTraceLevel	0 から 5	0	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
DeliveryQueue	任意の有効な JMS キュー名	*<CONNECTORNAME>*/DELIVERYQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
DeliveryTransport	MQ、IDL、または JMS	RepositoryDirectory の値が <REMOTE> である場合は IDL で、それ以外の場合は JMS	コンポーネント再始動	RepositoryDirectory の値が <REMOTE> ではない場合、このプロパティーで有効な値は JMS だけです。
DuplicateEventElimination	true または false	false	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
EnableOidForFlowMonitoring	true または false	false	コンポーネント再始動	このプロパティーは、BrokerType の値が ICS である場合のみ有効です。

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
FaultQueue	任意の有効なキュー名。	*<CONNECTORNAME>* /FAULTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.FactoryClassName	CxCommon.Messaging.jms .IBMMQSeriesFactory、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	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.ListenerConcurrency	1 から 32767	1	コンポーネント再始動	このプロパティーは、jms.TransportOptimized の値が true である場合のみ有効です。
jms.MessageBrokerName	jms.FactoryClassName の値が IBM である場合は、crossworlds.queue. manager を使用します。	crossworlds.queue. manager	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.NumConcurrent Requests	正整数	10	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.Password	任意の有効なパスワード		コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.TransportOptimized	true または false	false	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS で BrokerType の値が ICS である場合のみ有効です。
jms.UserName	任意の有効な名前		コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
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	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
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	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
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	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ListenerConcurrency	1 から 100	1	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が MQ である場合のみ有効です。

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
ロケール	これは、サポートされるロケールの一部です。 en_US、ja_JP、ko_KR、zh_CN、zh_TW、fr_FR、de_DE、it_IT、es_ES、pt_BR	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	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
MaxEventCapacity	1 から 2147483647	2147483647	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
MessageFileName	有効なファイル名	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	このプロパティーは、DuplicateEventElimination の値が true で ContainerManagedEvents に値がない場合のみ有効です。
OADAutoRestartAgent	true または false	false	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
OADMaxNumRetry	正整数	1000	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
OADRetryTimeInterval	正整数 (単位: 分)	10	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
PollEndTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒)	10000	ブローカーが ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
PollQuantity	1 から 500	1	エージェント再始動	このプロパティーは、ContainerManagedEvents の値が JMS である場合のみ有効です。
PollStartTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
RepositoryDirectory	ブローカーが ICS である場合は <REMOTE> で、それ以外の場合は有効な任意のローカル・ディレクトリー	ICS の場合、値は <REMOTE> に設定されます。 WMQI および WAS の場合、値は *<ProductDir*¥repository です。	エージェント再始動	
RequestQueue	有効な JMS キュー名	*<CONNECTORNAME>*/REQUESTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
ResponseQueue	有効な JMS キュー名	*<CONNECTORNAME>*/RESPONSEQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
RestartRetryCount	0 から 99	7	ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
RestartRetryInterval	1 から 2147483647 までの分単位の値。	1	ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
ResultsSetEnabled	true または false	false	コンポーネント再始動	DB2II をサポートするコネクターによってのみ使用されます。 このプロパティーは、DeliveryTransport の値が JMS で BrokerType の値が WMQI である場合のみ有効です。
ResultsSetSize	正整数	0 (結果セットのサイズが無制限であることを意味します)	コンポーネント再始動	DB2II をサポートするコネクターによってのみ使用されます。このプロパティーは、ResultsSetEnabled の値が true である場合のみ有効です。
RHF2MessageDomain	mrm または xml	mrm	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS で WireFormat の値が CwXML である場合のみ有効です。
SourceQueue	有効な WebSphere MQ キュー名	*<CONNECTORNAME>*/SOURCEQUEUE	エージェント再始動	このプロパティーは、ContainerManagedEvents の値が JMS である場合のみ有効です。
SynchronousRequest Queue	任意の有効なキュー名。	*<CONNECTORNAME>*/SYNCHRONOUSREQUEST QUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。

表 21. 標準構成プロパティーの要約 (続き)

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
SynchronousRequest Timeout	0 から任意の数 (ミリ秒)	0	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
SynchronousResponse Queue	任意の有効なキュー名	*<CONNECTORNAME>* /SYNCHRONOUSRESPONSE QUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。
TivoliMonitorTransaction Performance	true または false	false	コンポーネント再始動	
WireFormat	CwXML または CwBO	CwXML	エージェント再始動	RepositoryDirectory の値が <REMOTE> に設定されていない場合、このプロパティーの値は CwXML である必要があります。 RepositoryDirectory の値が <REMOTE> に設定されている場合、値は CwBO である必要があります。
WsifSynchronousRequest Timeout	0 から任意の数 (ミリ秒)	0	コンポーネント再始動	このプロパティーは、BrokerType の値が WAS である場合のみ有効です。
XMLNameSpaceFormat	short または long または no	short	エージェント再始動	このプロパティーは、BrokerType の値が WMQI または WAS である場合のみ有効です。

標準プロパティー

このセクションでは、標準コネクター構成プロパティーについて説明します。

AdapterHelpName

AdapterHelpName プロパティーは、コネクター固有の全般ヘルプ・ファイルがあるディレクトリーの名前です。ディレクトリーは *<ProductDir>*¥bin¥Data¥App¥Help にある必要があります。また、少なくとも言語ディレクトリー enu_usa を含んでいる必要があります。ロケールによっては、その他のディレクトリーを組み込むこともできます。

デフォルト値は、有効な場合はテンプレート名、そうでなければブランクです。

AdminInQueue

AdminInQueue プロパティーは、統合ブローカーからコネクタ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を指定します。

デフォルト値は *<CONNECTORNAME>*/ADMININQUEUE です。

AdminOutQueue

AdminOutQueue プロパティーは、コネクター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を指定します。

デフォルト値は *<CONNECTORNAME>*/ADMINOUTQUEUE です。

AgentConnections

AgentConnections プロパティーは、ORB が初期設定する場合、ORB (オブジェクト・リクエスト・ブローカー) 接続の数を制御します。

これは、RepositoryDirectory が <REMOTE> に設定され、DeliveryTransport プロパティーの値が MQ または IDL である場合のみ有効です。

このプロパティーのデフォルト値は 1 です。

AgentTraceLevel

AgentTraceLevel プロパティーは、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を設定します。コネクターは、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。

デフォルト値は 0 です。

ApplicationName

ApplicationName プロパティーは、コネクター・アプリケーションの名前を一意的に識別します。この名前は、システム管理者が統合環境をモニターするために使用します。コネクターを実行する前に、このプロパティーに値を指定する必要があります。

デフォルトはコネクターの名前です。

BiDi.Application

BiDi.Application プロパティーは、外部アプリケーションからアダプターに入ってくる、アダプターによってサポートされるビジネス・オブジェクトの形式のデータに対して、双方向フォーマットを指定します。プロパティーは、アプリケーション・データの双方向属性を定義します。属性は次のとおりです。

- テキストのタイプ: 暗黙またはビジュアル (I または V)
- テキストの方向: 左から右または右から左 (L または R)
- 対称スワッピング: オン/オフ (Y または N)
- シェーピング (アラビア語): オン/オフ (S または N)
- 数値シェーピング (アラビア語): ヒンディ語、コンテキスト、または公称 (H、C、または N)

このプロパティーは、BiDi.Transformation プロパティーが true に設定されている場合のみ有効です。

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Broker

BiDi.Broker プロパティーは、サポートされるビジネス・オブジェクトの形式でアダプターから統合ブローカーへ送信されるデータの双方向スクリプト形式を指定します。これは、前述の BiDi.Application の下にリストされるデータの双方向属性を定義します。

このプロパティーは、BiDi.Transformation プロパティーが true に設定されている場合のみ有効です。BrokerType プロパティーが ICS である場合、プロパティー値は読み取り専用です。

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Metadata

BiDi.Metadata プロパティーは、メタデータの双方向フォーマットまたは属性を定義します。これは、外部アプリケーションへのリンクを確立または保守するためにコネクターによって使用されます。属性設定は、双方向機能を使用する各アダプターに固有です。アダプターが双方向処理をサポートしている場合、詳細については、アダプター固有のプロパティーに関するセクションを参照してください。

このプロパティーは、BiDi.Transformation プロパティーが true に設定されている場合のみ有効です。

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Transformation

BiDi.Transformation プロパティーは、システムが実行時に双方向変換を実行するかどうかを定義します。

プロパティー値が true に設定されている場合は、BiDi.Application、BiDi.Broker、および BiDi.Metadata プロパティーを使用できます。プロパティー値が false に設定されている場合、それらは非表示になります。

デフォルト値は false です。

BOTrace

BOTrace プロパティーは、ビジネス・オブジェクト・トレース・メッセージを実行時に使用可能にするかどうかを指定します。

注: これは、AgentTraceLevel プロパティーが 5 より小さく設定されている場合に適用されます。

トレース・レベルが 5 より小さく設定されている場合は、次のコマンド行パラメーターを使用して BOTrace の値をリセットできます。

- ビジネス・オブジェクトの属性をすべてダンプする場合は、-xBOTrace=Full と入力します。
- ビジネス・オブジェクトのキーだけをダンプする場合は、-xBOTrace=Keys と入力します。

- ビジネス・オブジェクト属性のダンプを使用不可にする場合は、-xBOTrace=None と入力します。

デフォルト値は false です。

BrokerType

BrokerType プロパティーは、使用する統合ブローカー・タイプを指定します。可能な値は、ICS、WMQI (WMQI、WMQIB または WBIMB の場合)、または WAS です。

CharacterEncoding

CharacterEncoding プロパティーは、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 ベースのコネクターでは、このプロパティーは使用しません。 C++ コネクターは、このプロパティーに対して値 ascii7 を使用します。

デフォルトでは、サポートされる文字エンコードのサブセットだけが表示されます。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 (<*ProductDir*>)。詳細については、本書の Connector Configurator の付録を参照してください。

CommonEventInfrastructure

Common Event Infrastructure (CEI) は、生成されたイベントを処理する単純なイベント管理機能です。CommonEventInfrastructure プロパティーは、実行時に CEI を起動するかどうかを指定します。

デフォルト値は false です。

CommonEventInfrastructureContextURL

CommonEventInfrastructureContextURL は、Common Event Infrastructure (CEI) サーバー・アプリケーションを実行する WAS サーバーへのアクセスを獲得するために使用します。このプロパティーは、使用する URL を指定します。

このプロパティーは、CommonEventInfrastructure の値が true に設定されている場合のみ有効です。

デフォルト値はブランク・フィールドです。

ConcurrentEventTriggeredFlows

ConcurrentEventTriggeredFlows プロパティーは、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されるビジネス・オブジェクトの数に設定します。例えば、このプロパティーの値を 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。

このプロパティーを 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、次のプロパティーを構成する必要があります。

- `Maximum number of concurrent events` プロパティーの値を増やして、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ように構成する必要があります。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 構成プロパティーは、1 より大きい値に設定する必要があります。

ConcurrentEventTriggeredFlows プロパティーは、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。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です。

デフォルト値は `1` です。

ContainerManagedEvents

ContainerManagedEvents プロパティーにより、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、1 つの JMS トランザクションとして宛先キューに配置されます。

このプロパティーが `JMS` に設定されている場合は、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ーも構成する必要があります。

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

また、MimeType および DHClass (データ・ハンドラー・クラス) プロパティーを設定したデータ・ハンドラーも構成する必要があります。DataHandlerConfigMOName (オプションのメタオブジェクト名) を追加することもできます。これらのプロパティー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。

これらのプロパティーはアダプターに固有ですが、いくつかの値の例を次に示します。

- MimeType = `text¥xml`
- DHClass = `com.crossworlds.DataHandlers.text.xml`
- DataHandlerConfigMOName = `MO_DataHandler_Default`

「**データ・ハンドラー**」タブのこれらの値のフィールドは、ContainerManagedEvents プロパティーが値 JMS に設定されている場合のみ表示されます。

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。

ContainerManagedEvents プロパティーは、DeliveryTransport プロパティーの値が JMS に設定されている場合のみ有効です。

デフォルト値はありません。

ControllerEventSequencing

ControllerEventSequencing プロパティーは、コネクター・コントローラーでイベントの順序付けを使用可能にします。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

デフォルト値は true です。

ControllerStoreAndForwardMode

ControllerStoreAndForwardMode プロパティーは、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コネクター・コントローラーが検出した場合に、コネクター・コントローラーが実行する動作を設定します。

このプロパティーを 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、コネクター・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、コネクター・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、コネクター・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、コネクター・コントローラーはその要求を失敗させます。

このプロパティーを false に設定した場合、コネクター・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。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType プロパティーは ICS)。

デフォルト値は true です。

ControllerTraceLevel

ControllerTraceLevel プロパティーは、コネクター・コントローラーのトレース・メッセージのレベルを設定します。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です。

デフォルト値は 0 です。

DeliveryQueue

DeliveryQueue プロパティーは、コネクター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を定義します。

このプロパティーは、DeliveryTransport プロパティーの値が JMS に設定されている場合のみ有効です。

デフォルト値は *<CONNECTORNAME>*/DELIVERYQUEUE です。

DeliveryTransport

DeliveryTransport プロパティーは、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。

- RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合、DeliveryTransport プロパティーの値は MQ、IDL、または JMS で、デフォルトは IDL です。
- RepositoryDirectory プロパティーの値がローカル・ディレクトリーである場合、指定できる値は JMS のみです。

RepositoryDirectory プロパティーの値が MQ または IDL である場合、コネクターは、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。

DeliveryTransport プロパティーの値が MQ である場合は、アダプター開始スクリプトでコマンド行パラメーター WhenServerAbsent を設定して、InterChange Server がシャットダウンされているときにアダプターを一時停止するかあるいはシャットダウンするかを示すことができます。

- ICS が使用不可である場合にアダプターを一時停止する場合は、WhenServerAbsent=pause と入力します。
- ICS が使用不可である場合にアダプターをシャットダウンする場合は、WhenServerAbsent=shutdown と入力します。

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ー・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ー・データベースに書き込むことがなくなります。

- 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、コネクター・ポーリング・スレッドは、イベントを選出した後、コネクター・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 よりも高速です。IDL の場合、コネクターのポーリング・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ー・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

JMS トランスポート・メカニズムは、Java Messaging Service (JMS) による、コネクターとクライアント・コネクター・フレームワーク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`、および `jms.UserName` などの追加の JMS プロパティーが Connector Configurator 内にリストされます。プロパティー `jms.MessageBrokerName` および `jms.FactoryClassName` は、このトランスポートで必須です。

以下の環境では、コネクターに JMS トランスポート機構を使用すると、メモリー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

この環境では、WebSphere MQ クライアント内でメモリーが使用されるため、(サーバー側の) コネクター・コントローラーと (クライアント側の) コネクター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 768MB 未満である場合は、次の変数とプロパティーを設定します。

- 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 (<*ProductDir*>) の下の ¥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 プロパティーの値を 11 または 12 に設定します。このプロパティーの詳細については、「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

このプロパティーの値が `true` である場合は、JMS 対応コネクターによるデリバリー・キューへの重複イベントのデリバリーが防止されます。コネクターの開発時にこの機能を使用するには、コネクターに対し、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。

注: このプロパティーの値が `true` である場合、保証付きイベント・デリバリーを提供するため、MonitorQueue プロパティーが使用可能になっている必要があります。

デフォルト値は `false` です。

EnableOidForFlowMonitoring

このプロパティーの値が `true` である場合は、アダプター・ランタイムによって着信 ObjectEventID にフロー・モニターの外部キーのマークが付けられます。

このプロパティーは、BrokerType プロパティーが ICS に設定されている場合のみ有効です。

デフォルト値は `false` です。

FaultQueue

コネクターでメッセージを処理中にエラーが発生すると、コネクターは、そのメッセージ (および状況標識と問題説明) を FaultQueue プロパティーで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。

jms.FactoryClassName

jms.FactoryClassName JMS プロパティーは、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。DeliveryTransport プロパティーの値が `JMS` である場合は、このプロパティーを設定する必要があります。

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

jms.ListenerConcurrency

jms.ListenerConcurrency プロパティーは、JMS コントローラーの並行リスナーの数を指定します。これは、コントローラー内でメッセージを並行して取り出して処理するスレッドの数を指定します。

このプロパティーは、jms.OptimizedTransport プロパティーの値が `true` である場合のみ有効です。

デフォルト値は `1` です。

jms.MessageBrokerName

jms.MessageBrokerName は、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として指定する場合 (DeliveryTransport プロパティーで) は、このコネクター・プロパティーを設定する必要があります。

リモート・メッセージ・ブローカーに接続する場合、このプロパティーには次の値を指定する必要があります。
QueueMgrName:*Channel*:*HostName*:*PortNumber*
各変数の意味は、以下のとおりです。
QueueMgrName は、キュー・マネージャー名です。
Channel は、クライアントが使用するチャネルです。
HostName は、キュー・マネージャーの配置先のマシン名です。
PortNumber は、キュー・マネージャーが listen に使用するポートの番号です。

例えば、次のようにします。
```
jms.MessageBrokerName = WBIMB.Queue.Manager:CHANNEL1:RemoteMachine:1456
```

デフォルト値は crossworlds.queue.manager です。ローカル・メッセージ・ブローカーに接続する場合は、デフォルト値を使用します。

jms.NumConcurrentRequests

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

デフォルト値はありません。

jms.TransportOptimized

jms.TransportOptimized プロパティーは、WIP (処理中の作業) を最適化するかどうかを決定します。WIP を最適化する WebSphere MQ プロバイダーが必要です。最適化された WIP を操作するには、メッセージング・プロバイダーが次のことを実行できる必要があります。

1. メッセージをキューから削除せずに読み取る
2. メッセージ全体を受信側のメモリー・スペースに転送せずに、特定の ID を持つメッセージを削除する
3. 特定の ID を使用してメッセージを読み取る (リカバリーに必要)
4. 読み取られていないイベントが表示されるポイントを追跡する

JMS API は上記の条件 2 と 4 に合わないので、最適化された WIP に使用することはできませんが、MQ Java API は 4 つの条件すべてを満たしているので、最適化された WIP に必要です。

このプロパティーは、DeliveryTransport の値が JMS で BrokerType の値が ICS である場合のみ有効です。

デフォルト値は false です。

jms.UserName

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

デフォルト値はありません。

JvmMaxHeapSize

JvmMaxHeapSize プロパティーは、エージェントの最大ヒープ・サイズ (メガバイト単位) を指定します。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です。

デフォルト値は 128M です。

JvmMaxNativeStackSize

JvmMaxNativeStackSize プロパティーは、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) を指定します。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です。

デフォルト値は 128K です。

JvmMinHeapSize

JvmMinHeapSize プロパティーは、エージェントの最小ヒープ・サイズ (メガバイト単位) を指定します。

このプロパティーは、RepositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です。

デフォルト値は 1M です。

ListenerConcurrency

ListenerConcurrency プロパティーは、統合ブローカーとして ICS を使用する場合の WebSphere MQ Listener でのマルチスレッド化をサポートしています。このプロパティーにより、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。

このプロパティーは、MQ トランスポートを使用するコネクターでのみ有効です。DeliveryTransport プロパティーの値は、MQ にしてください。

デフォルト値は 1 です。

Locale

Locale プロパティーは、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。このプロパティーの値は、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

各変数の意味は、以下のとおりです。
ll は、2 文字の言語コード (小文字) です。
TT は、2 文字の国または地域コード (大文字) です。
codeset は、関連文字コード・セットの名前です。

デフォルトでは、サポートされるロケールのサブセットだけがリストされます。サポートされる他の値をリストに追加するには、<*ProductDir*>¥bin ディレクトリーにある ¥Data¥Std¥stdConnProps.xml ファイルを変更します。詳細については、本書の Connector Configurator の付録を参照してください。

コネクターが国際化に対応していない場合、このプロパティーの有効な値は en_US のみです。特定のコネクター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、そのアダプターのユーザー・ガイドを参照してください。

デフォルト値は en_US です。

LogAtInterchangeEnd

LogAtInterchangeEnd プロパティーは、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。

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 ファイルの 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 の値が true である場合にコネクター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。

このプロパティーは、RespositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

デフォルト値は false です。

MaxEventCapacity

MaxEventCapacity プロパティーは、コントローラー・バッファー内のイベントの最大数を指定します。このプロパティーは、フロー制御フィーチャーによって使用されます。

このプロパティーは、RespositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

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

MessageFileName プロパティーは、コネクター・メッセージ・ファイルの名前を指定します。メッセージ・ファイルの標準位置は、製品ディレクトリーの ¥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。

コネクター・メッセージ・ファイルが存在しない場合は、コネクターは `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: コネクターについて、コネクター独自のメッセージ・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。

デフォルト値は `InterchangeSystem.txt` です。

MonitorQueue

MonitorQueue プロパティーは、コネクター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を指定します。

これは DeliveryTransport プロパティーの値が `JMS` で DuplicateEventElimination の値が `true` である場合のみ有効です。

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

OADAutoRestartAgent

OADAutoRestartAgent プロパティーは、コネクター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、WebSphere 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ーを再始動したり、System Monitor からリモート・コネクター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、このプロパティーを `true` に設定する必要があります。WebSphere 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。

このプロパティーは、RespositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

デフォルト値は `false` です。

OADMaxNumRetry

OADMaxNumRetry プロパティーは、異常シャットダウンの後で WebSphere MQ により起動される Object Activation Daemon (OAD) がコネクター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。このプロパティーを有効にするには、OADAutoRestartAgent プロパティーを `true` に設定する必要があります。

このプロパティーは、RespositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

デフォルト値は `1000` です。

OADRetryTimeInterval

OADRetryTimeInterval プロパティーは、WebSphere MQ により起動される Object Activation Daemon (OAD) の再試行時間間隔の分数を指定します。コネクター・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、コネクター・コントローラーはコネクター・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 プロパティーで指定された回数だけ繰り返します。このプロパティーを有効にするには、OADAutoRestartAgent プロパティーを `true` に設定する必要があります。

このプロパティーは、RespositoryDirectory プロパティーの値が <REMOTE> に設定されている場合のみ有効です (BrokerType は ICS)。

デフォルト値は `10` です。

PollEndTime

PollEndTime プロパティーは、イベント・キューのポーリングを停止する時刻を指定します。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。

このプロパティー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、値が指定されていないので、これは変更する必要があります。

アダプター・ランタイムが次のことを検出した場合:

- PollStartTime が設定され PollEndTime が設定されていない、または、
- PollEndTime が設定され PollStartTime が設定されていない

ポーリングは PollFrequency プロパティーで構成された値を使用して実行されます。

PollFrequency

PollFrequency プロパティーは、あるポーリング・アクションの終了から次のポーリング・アクションの開始までの時間をミリ秒単位で指定します。これはポーリング・アクション間の間隔ではありません。この論理を次に説明します。

- ポーリングし、PollQuantity プロパティーの値により指定される数のオブジェクトを取得します。

- これらのオブジェクトを処理します。一部のコネクターでは、これは個別のスレッドで部分的に実行されます。これにより、次のポーリング・アクションまで処理が非同期に実行されます。
- PollFrequency プロパティーで指定された間隔にわたって遅延します。
- このサイクルを繰り返します。

このプロパティーで有効な値は次のとおりです。

- ポーリング・アクション間のミリ秒数 (正整数)。
- ワード `no`。コネクター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。
- ワード `key`。コネクターは、コネクターのコマンド・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。

デフォルト値は `10000` です。

重要: 一部のコネクターでは、このプロパティーの使用が制限されています。このようなコネクターが存在する場合には、アダプターのインストールと構成に関する章で制約事項が説明されています。

PollQuantity

PollQuantity プロパティーは、コネクター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ー固有のポーリング数設定プロパティーがある場合、標準プロパティーの値は、このコネクター固有のプロパティーの設定値によりオーバーライドされます。

このプロパティーは、`DeliveryTransport` プロパティーの値が `JMS` で `ContainerManagedEvents` プロパティーに値がある場合のみ有効です。

電子メール・メッセージもイベントと見なされます。コネクターは、電子メールに関するポーリングを受けたときには次のように動作します。

- 一度ポーリングされると、コネクターはメッセージの本文を検出し、それを添付ファイルとして読み取ります。本文の MIME タイプにはデータ・ハンドラーが指定されていないので、コネクターはメッセージを無視します。
- コネクターは最初の BO 添付を処理します。この添付の MIME タイプには対応するデータ・ハンドラーがあるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。
- 2 回目にポーリングされると、コネクターは 2 番目の BO 添付を処理します。この添付の MIME タイプには対応するデータ・ハンドラーがあるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。
- 3 番目の BO 添付は、受け入れられると、送信されます。

PollStartTime

PollStartTime プロパティーは、イベント・キューのポーリングを開始する時刻を指定します。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。

このプロパティー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、値が指定されていないので、これは変更する必要があります。

アダプター・ランタイムが次のことを検出した場合:

- PollStartTime が設定され PollEndTime が設定されていない、または、
- PollEndTime が設定され PollStartTime が設定されていない

ポーリングは PollFrequency プロパティーで構成された値を使用して実行されます。

RepositoryDirectory

RepositoryDirectory プロパティーは、コネクターが XML スキーマ文書を読み取るリポジトリーの場所です。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、コネクターが InterChange Server リポジトリー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 である場合、この値はデフォルトで *<ProductDir>*¥repository に設定されます。ただし、任意の有効なディレクトリー名に設定することもできます。

RequestQueue

RequestQueue プロパティーは、統合ブローカーからコネクターへビジネス・オブジェクトが送信されるときに使用されるキューを指定します。

このプロパティーは、DeliveryTransport プロパティーの値が JMS である場合のみ有効です。

デフォルト値は *<CONNECTORNAME>*/REQUESTQUEUE です。

ResponseQueue

ResponseQueue プロパティーは、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ー・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。

このプロパティーは、DeliveryTransport プロパティーの値が JMS である場合のみ有効です。

デフォルト値は *<CONNECTORNAME>*/RESPONSEQUEUE です。

RestartRetryCount

RestartRetryCount プロパティーは、コネクターによるコネクター自体の再始動の試行回数を指定します。このプロパティーを並列に接続されたコネクターに対して使用すると、コネクター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クライアント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 7 です。

RestartRetryInterval

RestartRetryInterval プロパティーは、コネクターによるコネクター自体の再始動の試行間隔を分単位で指定します。このプロパティーを並列にリンクされたコネクターに対して使用すると、コネクター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クライアント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。

このプロパティーで指定できる値は、1 から 2147483647 です。

デフォルト値は 1 です。

ResultsSetEnabled

ResultsSetEnabled プロパティーは、Information Integrator がアクティブである場合に、結果セット・サポートを使用可能または使用不可にします。このプロパティーを使用できるのは、アダプターが DB2 Information Integrator をサポートしている場合だけです。

このプロパティーは、DeliveryTransport プロパティーの値が JMS で BrokerType の値が WMQI である場合のみ有効です。

デフォルト値は false です。

ResultsSetSize

ResultsSetSize プロパティーは、Information Integrator へ戻すことができるビジネス・オブジェクトの最大数を定義します。このプロパティーを使用できるのは、アダプターが DB2 Information Integrator をサポートしている場合だけです。

このプロパティーは、ResultsSetEnabled プロパティーの値が true である場合のみ有効です。

デフォルト値は 0 です。これは、結果セットのサイズが無限であることを意味します。

RHF2MessageDomain

RHF2MessageDomain プロパティーにより、JMS ヘッダーのドメイン名フィールドの値を構成できます。JMS トランスポートを介してデータを WebSphere メッセージ・ブローカーに送信するときに、アダプター・フレームワークにより JMS ヘッダー情報、ドメイン名、および固定値 mrm が書き込まれます。構成可能ドメイン・ネームを使用すると、WebSphere メッセージ・ブローカーがメッセージ・データをどのように処理するかを追跡できます。

次に、ヘッダーの例を示します。

```
<mcd><Msd>mrm</Msd><Set>3</Set><Type>
Retek_POPhyDesc</Type><Fmt>CwXML</Fmt></mcd>
```

このプロパティーは、BrokerType の値が WMQI または WAS である場合のみ有効です。また、これは DeliveryTransport プロパティーの値が JMS で WireFormat プロパティーの値が CwXML である場合のみ有効です。

指定できる値は mrm と xml です。デフォルト値は mrm です。

SourceQueue

SourceQueue プロパティーは、JMS イベント・ストアを使用する JMS 対応コネクター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ー・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、85 ページの『ContainerManagedEvents』を参照してください。

このプロパティーは、DeliveryTransport の値が JMS で ContainerManagedEvents の値が指定されている場合のみ有効です。

デフォルト値は *<CONNECTORNAME>*/SOURCEQUEUE です。

SynchronousRequestQueue

SynchronousRequestQueue プロパティーは、同期応答を要求する要求メッセージを、コネクター・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、コネクター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、コネクター・フレームワークは、同期要求キューにメッセージを送信し、同期要求キューでブローカーからの応答を待ちます。コネクター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。

このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。

デフォルト値は *<CONNECTORNAME>*/SYNCHRONOUSREQUESTQUEUE です。

SynchronousRequestTimeout

SynchronousRequestTimeout プロパティーは、コネクターが同期要求への応答を待機する時間をミリ秒単位で指定します。コネクターは、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 (およびエラー・メッセージ) を障害キューに移動します。

このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。

デフォルト値は 0 です。

SynchronousResponseQueue

SynchronousResponseQueue プロパティーは、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ー・フレームワークに配信します。このキューは、コネクターが同期実行を使用する場合にのみ必要です。

このプロパティーは、DeliveryTransport の値が JMS である場合のみ有効です。

デフォルトは *<CONNECTORNAME>*/SYNCHRONOUSRESPONSEQUEUE です。

TivoliMonitorTransactionPerformance

TivoliMonitorTransactionPerformance プロパティーは、実行時に IBM Tivoli Monitoring for Transaction Performance (ITMTP) を起動するかどうかを指定します。

デフォルト値は `false` です。

WireFormat

WireFormat プロパティーは、トランスポートでのメッセージ・フォーマットを指定します。

- RepositoryDirectory プロパティーの値がローカル・ディレクトリーである場合、値は `CwXML` です。
- RepositoryDirectory プロパティーの値がリモート・ディレクトリーである場合、値は `CwBO` です。

WsifSynchronousRequestTimeout

WsifSynchronousRequestTimeout プロパティーは、コネクターが同期要求への応答を待機する時間をミリ秒単位で指定します。コネクターは、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 (およびエラー・メッセージ) を障害キューに移動します。

このプロパティーは、BrokerType の値が `WAS` である場合のみ有効です。

デフォルト値は 0 です。

XMLNameSpaceFormat

XMLNameSpaceFormat プロパティーは、ビジネス・オブジェクト定義の XML フォーマットで短いネーム・スペースまたは長いネーム・スペースを指定します。

このプロパティーは、BrokerType の値が WMQI または WAS に設定されている場合のみ有効です。

デフォルト値は `short` です。

付録 B. このアダプターのコネクター固有のプロパティー

この付録では、Adapter for WebSphere Business Integration Message Broker のコネクター固有のプロパティーについて説明します。Connector Configurator を使用してアダプターを構成するときには、これらのプロパティーの説明を参照してください。

コネクター固有のプロパティーの概要

コネクター固有の構成プロパティーは、コネクター・エージェントが実行時に必要とする情報を提供します。また、コネクター固有の構成プロパティーを使用すると、コネクター・エージェントのコード変更や再ビルドを行わなくても、エージェント内の静的情報またはロジックを変更できます。

次のプロパティーは、アダプターと WebSphere Business Integration Message Broker の間の通信を判別します。

- ArchiveQueue
- ErrorQueue
- InputQueue
- InProgressQueue
- ReplyToQueue
- UnsubscribedQueue

InterChange Server 統合ブローカーとアダプターの間の通信を判別するプロパティーは、73 ページの『付録 A. コネクターの標準構成プロパティー』にあります。

コネクター固有のプロパティー

表 22 に、このコネクターに固有な構成プロパティーを示します。プロパティーの説明については、以下の各セクションを参照してください。

表 22. コネクター固有の構成プロパティー

名前	指定可能な値	デフォルト値	必須
ApplicationPassword	*ログイン・パスワード*		いいえ
ApplicationUserName	*ログイン・ユーザー ID*		いいえ
ArchiveQueue	*正常に処理されたメッセージのコピーが送信されるキュー*	queue://crossworlds.queue.manager/WBIMBConnector/ARCHIVE	いいえ
CCSID	*キュー・マネージャー接続用文字セット*		はい
Channel	*MQ サーバー・コネクター・チャネル*		はい
ConfigurationMetaObject	*構成メタオブジェクトの名前*		はい
DataHandlerClassName	*データ・ハンドラー・クラス名*	com.crossworlds.DataHandlers.text.xml	はい
DataHandlerConfigMO	*データ・ハンドラー・メタオブジェクト*	MO_DataHandler_Default	はい
DataHandlerMimeType	*ファイルの MIME タイプ*	text/xml	はい
DataHandlerPoolSize	*再利用のためキャッシュに入れられるデータ・ハンドラー・インスタンスの数*	30	いいえ
DefaultVerb	*コネクターがサポートしている任意の動詞*		
EnableMessageProducerCache	true または false	true	いいえ
ErrorQueue	*未処理のメッセージのキュー*	queue://crossworlds.queue.manager/WBIMBConnector/ERROR	いいえ
HostName	*WebSphere MQ サーバー*	バインディング・モードでローカル・キュー・マネージャーに接続します。	いいえ
InputQueue	*ポーリング・キュー*	queue://crossworlds.queue.manager/WBIMBConnector/IN	はい
PollQuantity	*InputQueue プロパティー内で指定された各キューから検索されるメッセージの数*	1	いいえ
Port	*WebSphere MQ リスナー用に設定されたポート*	1414	いいえ
ReplyToQueue	*コネクターからの要求発行時に応答メッセージがデリバリーされるキュー*	queue://crossworlds.queue.manager/WBIMBConnector/REPLY	いいえ
ReplyToQueuePollFrequency	*同期要求処理中の受信側のポーリング間隔 (ミリ秒単位)*		いいえ
SecurityExitClassName	*使用中のセキュリティー出口の完全修飾クラス名*		いいえ
SecurityExitInitParam	*セキュア出口を呼び出すために使用する初期化ストリングで構成する値を指定します。*		いいえ
SessionPoolSizeForRequests	*要求処理中に使用されるセッションをキャッシュする最大プール・サイズ*	10	いいえ
UnsubscribedQueue	*アンサブスクライブされたメッセージが送信されるキュー*	queue://crossworlds.queue.manager/WBIMBConnector/UNSUBSCRIBED	いいえ
UseDefaults	true または false	false	
WorkerThreadCount	*ポーリングの並列スレッドの最大数*	1	いいえ

ApplicationPassword

WebSphere Business Integration Message Broker にログインするために、UserID とともに使用されるパスワード。

デフォルト = 設定値なし

ApplicationPassword の値がブランクのままか、または除去された場合、コネクターは WebSphere Business Integration Message Broker によって提供されるデフォルトのパスワードを使用します。

ApplicationUserName

WebSphere Business Integration Message Broker にログインするために、Password とともに使用されるユーザー ID。

デフォルト = 設定値なし

ApplicationUserName の値がブランクのままか、または除去された場合、コネクターは WebSphere Business Integration Message Broker によって提供されるデフォルトのユーザー ID を使用します。

ArchiveQueue

正常に処理されたメッセージのコピーが送信されるキューです。

デフォルト = queue://crossworlds.queue.manager/WBIMBConnector/ARCHIVE

CCSID

キュー・マネージャー接続用文字セット。このプロパティーの値は、キュー URI 内の CCSID プロパティーの値と一致していることが必要です。 38 ページの『キューの Uniform Resource Identifier (URI) の設定』を参照してください。

デフォルト = 設定値なし

Channel

コネクターが WebSphere Business Integration Message Broker と通信するときに使用する MQ サーバー・コネクター・チャネルです。

デフォルト = 設定値なし

Channel の値がブランクのままか、または除去された場合、コネクターは WebSphere Business Integration Message Broker によって提供されるデフォルトのサーバー・チャネルを使用します。

ConfigurationMetaObject

コネクターの構成情報を含むメタオブジェクトの名前です。

デフォルト = 設定値なし

DataHandlerClassName

ビジネス・オブジェクトとの間でのメッセージ変換に使用するデータ・ハンドラー・クラスです。

デフォルト = com.crossworlds.DataHandlers.text.xml

DataHandlerConfigMO

構成情報を提供するためにデータ・ハンドラーに渡されるメタオブジェクトです。

デフォルト = MO_DataHandler_Default

DataHandlerMimeType

使用すると、特定の MIME タイプに基づいたデータ・ハンドラーを要求できます。

デフォルト = text/xml

DataHandlerPoolSize

特定のタイプのデータ・ハンドラー用にキャッシュに入れるデータ・ハンドラー・インスタンスの最大数を指定できます。

デフォルト = 30

DefaultVerb

ポーリング時にデータ・ハンドラーによって動詞が設定されなかった場合、着信ビジネス・オブジェクトの内部に設定する動詞を指定します。

デフォルト = 設定値なし

EnableMessageProducerCache

要求メッセージを送信するために、アダプターがメッセージ・プロデューサーのキャッシュを有効にすることを指定する boolean プロパティー。

デフォルト = true

ErrorQueue

処理されなかったメッセージが送信されるキューです。

デフォルト = queue://crossworlds.queue.manager/WBIMBConnector/ERROR

HostName

WebSphere Business Integration Message Broker をホスティングしているサーバーの名前です。

デフォルト = 値が指定されない場合、アダプターはバインディング・モードでローカル・キュー・マネージャーに接続します。

InputQueue

コネクターが新規メッセージをポーリングするメッセージ・キューです。コネクターは、セミコロンによって区切られた複数のキュー名を受け入れます。例えば、MyQueueA、MyQueueB、および MyQueueC の 3 つのキューをポーリングするには、コネクター構成プロパティー *InputQueue* の値を MyQueueA;MyQueueB;MyQueueC にします。

コネクターはラウンドロビン方式でキューをポーリングし、各キューから pollQuantity で指定された値を最大数とするメッセージを検索します。例えば、*pollQuantity* の値が 2 で、MyQueueA 内に 2 つ、MyQueueB 内に 1 つ、MyQueueC 内に 5 つのメッセージがそれぞれ格納されている場合、コネクターは次のようにメッセージを検索します。

pollQuanity が 2 に設定されているため、コネクターは、pollForEvents への 1 回の呼び出しごとに各キューからそれぞれ最大 2 つのメッセージを検索します。最初のサイクル (2 サイクルのうちの 1 サイクル目) では、コネクターは MyQueueA、MyQueueB、および MyQueueC の各キューからそれぞれ 1 番目のメッセージを検索します。これによって、ポーリングの第 1 ラウンドが完了します。pollQuantity が 1 に設定されている場合、コネクターはこの時点で停止します。この例では pollQuanity が 2 に設定されているため、コネクターは第 2 ラウンドのポーリングを開始し、MyQueueA と MyQueueC の各キューからそれぞれ 1 つずつのメッセージを検索します。このとき、MqQueueB は空になっているためスキップされます。すべてのキューを 2 回ずつポーリングすると、メソッド pollForEvents への呼び出しは完了します。このメッセージ検索の順序を以下に示します。

1. MyQueueA から 1 つのメッセージ
2. MyQueueB から 1 つのメッセージ
3. MyQueueC から 1 つのメッセージ
4. MyQueueA から 1 つのメッセージ
5. MyQueueB は空なのでスキップ
6. MyQueueC から 1 つのメッセージ

デフォルト = queue://crossworlds.queue.manager/WBIMBConnector/IN

PollQuantity

pollForEvents スキャン中に InputQueue プロパティー内で指定された各キューから検索されるメッセージの数。

デフォルト = 1

Port

WebSphere Business Integration Message Broker リスナー用に設定されたポートです。

デフォルト = WebSphere MQ 環境のデフォルト・ポート。これは、1414 です。

ReplyToQueue

コネクターからの要求発行時に応答メッセージがデリバリーされるキューです。

デフォルト = queue://crossworlds.queue.manager/WBIMBConnector/REPLY

ReplyToQueuePollFrequency

同期要求処理中の受信側のポーリング間隔を指定します。値はミリ秒単位です。

デフォルト = 設定値なし

SecurityExitClassName

使用中のセキュリティー出口の完全修飾クラス名。

デフォルト = 設定値なし

SecurityExitInitParam

セキュア出口を呼び出すために使用する初期化ストリングで構成する値を指定します。

デフォルト = 設定値なし

SessionPoolSizeForRequests

要求処理中に使用されるセッションをキャッシュする最大プール・サイズ。

デフォルト = 10

UnsubscribedQueue

サブスクライブされていないメッセージが送信されるキューです。

デフォルト = queue://crossworlds.queue.manager/WBIMBConnector/UNSUBSCRIBED

注: *WebSphere Business Integration Message Broker によって提供される値は誤っていたり不明である可能性があるため、常にチェックする必要があります。値が誤っていたり不明な場合は、値を暗黙的に指定してください。

UseDefaults

Create 操作において、UseDefaults に true が設定されている場合、コネクターは、各 isRequired ビジネス・オブジェクト属性に対して、デフォルト値または有効値が指定されているかどうかをチェックします。デフォルト値または有効値が指定されている場合は、Create 操作は成功します。このパラメーターに false が設定されている場合は、コネクターは、有効値のみをチェックします。有効値が指定されていない場合は、Create 操作は失敗します。デフォルト値は false です。

WorkerThreadCount

ポーリングの並列スレッドの最大数。イベントの並行処理中、アダプターがブローカーに、イベントを受信した順序で送信することはできません。順序を守る必要がある場合は、WorkerThreadCount を必ず 1 に設定してください。

付録 C. チュートリアル

このチュートリアルは、以下のことを示すことを目的として記述されています。

- Adapter for WebSphere Business Integration Message Broker を WebSphere Business Integration Message Broker のメッセージ・フローと統合するにはどうすればよいか。
- アダプターはどのようにしてレガシー・アプリケーションとの間でメッセージを送受信するのか。

このチュートリアルのシナリオは、アダプターの機能の基本を示すことを目的としています。

表記規則のガイドについては、本書のまえがきを参照してください。

チュートリアルの概要

このチュートリアルでは、メッセージ・ブローカー経由でアダプターに XML メッセージを送信するレガシー・アプリケーション (ユーティリティーによりシミュレートしたもの) を使用します。メッセージにはさまざまなカタログ項目の変更が含まれます。このレガシー・メッセージは、WebSphere Business Integration Message Broker に関連付けられているデータウェアハウス・フローとメッセージ変換によって処理されるのではなく、単純なカラー・チェック・フローによって処理されます。処理されたメッセージは、アダプターに送信されます。

アダプターでは、カラー処理されたメッセージを WebSphere Business Integration Message Broker から受け取ります。アダプターは XML データ・ハンドラーを使用してそのメッセージをビジネス・オブジェクト `Sample_WBIMB_LegacyItem_XMLDoc` に変換し、さらに統合ブローカーにこのオブジェクトを渡します。このチュートリアルでは、Visual Test Connector を使用してポート・コネクターをシミュレートし、新規作成されたビジネス・オブジェクトを取得してその内容を確認します。

また、逆に Visual Test Connector からビジネス・オブジェクトを発行します。発行されたオブジェクトは、Adapter for WebSphere Business Integration Message Broker でレガシー・メッセージに変換されます。このメッセージはアダプターから WebSphere Business Integration Message Broker アプリケーションにデリバリーされ、ここで処理された後 (メッセージ・フォーマットが `LI_UP` に更新されます)、レガシー・アプリケーション (シミュレートされたもの) に転送されます。このシナリオについては、WMQI に対応するように構成されたアダプターを使用して説明します。

このシナリオを実行するために必要な条件は、アダプターとアダプター・フレームワークに加えて以下のものがインストールされていることだけです。

- メッセージ・フローのサンプル、ドメイン構成マネージャー・ファイルのサンプル、および配置可能な bar ファイルが含まれる WebSphere Business Integration Message Broker プロジェクト (メッセージ・フロー・プロジェクトおよびサーバー・プロジェクト)。
- ポート・コネクターのリポジトリーと Visual Test Connector (ADK のインストールに組み込まれています)。ポート・コネクターは、特定のコードを基礎としないアダプター定義で構成されており、そのためシミュレーション・シナリオに適しています。Visual Test Connector は、インターフェースをテストするためのツールです。

この付録で説明する作業

この付録では、以下の作業について説明します。

表 23. アダプターのデモンストレーション: 作業ロードマップ

作業	関連手順 (参照先)
インストール内容のチェックリストの確認	『始める前に』
環境のセットアップ	109 ページの『環境のセットアップ』
シナリオの実行	112 ページの『シナリオの実行』

始める前に

このチュートリアルを開始する前に、以下のことを確認してください。

- IBM WebSphere 製品がインストールされており、かつこの製品の操作に慣れていること。
- WebSphere MQ 5.1 以降がインストールされていること。
- WebSphere Business Integration Message Broker 5.0 以降がインストールされていること。
- WebSphere MQ クライアント・ライブラリー (Java 用) がインストールされていること。
- Adapter for WebSphere Business Integration Message Broker がインストールされていること (構成方法については、このチュートリアルで説明します)。
- WebSphere MQ のアダプター・キュー・マネージャーに `crossworlds.queue.manager` という名前 (インストール時に設定されるデフォルト値) が付けられていること。これ以外の名前が付けられている場合は、本書に記載されている「`crossworlds.queue.manager`」を、そのキュー・マネージャー名に読み替えてください。また、以下の手順を実行する必要があります。
  1. Message Brokers Toolkit を開き、メッセージ・フローのサンプルに含まれるすべての MQOutput ノードの QueueManagerName 属性の値を、`crossworlds.queue.manager` に変更します。
  2. サーバー・プロジェクト (アダプターとともにインストールされたサンプルに含まれています) 内の `LocalDomain.configmgr` ファイルを開き、キュー・マネージャー名 (ブローカーが対応するキュー・マネージャーの名前) を `crossworlds.queue.manager` に変更します。

3. Business Object Designer を開き、メタオブジェクトのサンプルである Sample_WBIMB_MO_Config に指定されている出力キュー・マネージャーを crossworlds.queue.manager に変更します。

- WebSphere MQ Explorer を含めて、WebSphere MQ の (基本インストールではなく) フルインストールが行われていること。WebSphere MQ Explorer は必須ではありません。ただし、このツールを使用すると、キューの設定とメッセージの検証を容易に実行できます。
- WebSphere MQ クライアント・ライブラリー (Java 用) がインストールされていること。

環境のセットアップ

このトピックでは、環境のセットアップの概要と手順を示します。

環境のセットアップの概要

このセクションでは、このチュートリアルを実行するために必要な環境を準備する方法について説明します。なお、この後の説明の *sample_folder* という変数は、インストール済みのファイル構造に含まれている ConnName/sample/LegacyItem フォルダーを表しています。詳細については、14 ページの『インストールの検証』を参照してください。ビジネス・オブジェクト・リポジトリーは、*sample_folder* 内に .xsd ファイルとして用意されています。

このチュートリアルでは、Adapter for WebSphere Business Integration Message Broker と Visual Test Connector の間での単純なビジネス・オブジェクト交換の例を示します。この交換は、WebSphere Business Integration Message Broker 環境内で行われます。

環境のセットアップのステップ

このチュートリアルで使用する環境をセットアップするには、以下の構成作業を行う必要があります。

1. **キューの定義**: このチュートリアルでは、キュー・マネージャーに 8 つのキューが定義されていることが必要です。必要なキューを作成するには、コマンド行から「RUNMQSC」と入力し、以下のコマンドを実行します。
   - DEFINE QL('Samples/WBIMB/Item/LegacyApp')
   - DEFINE QL('Samples/WBIMB/Item/WBIMBConnector')
   - DEFINE QL('Samples/LegacyApp/Item/WBIMB')
   - DEFINE QL('Samples/WBIMBConnector/Item/WBIMB')
   - DEFINE QL('Samples/WBIMB/FAIL')
   - DEFINE QL('Samples/WBIMBConnector/UNSUBSCRIBED')
   - DEFINE QL('Samples/WBIMBConnector/ERROR')
   - DEFINE QL('Samples/WBIMBConnector/ARCHIVE')

   次に、WebSphere Business Integration Message Broker を構成するために、Adapter for Business Integration Message Broker とポート・コネクターが必要とするキューを以下のように定義します。
   - DEFINE QL('WBIMBConnector/ADMININQUEUE')

- DEFINE QL('WBIMBConnector/ADMINOUTQUEUE')
- DEFINE QL('WBIMBConnector/DELIVERYQUEUE')
- DEFINE QL('WBIMBConnector/FAULTQUEUE')
- DEFINE QL('WBIMBConnector/REQUESTQUEUE')
- DEFINE QL('WBIMBConnector/RESPONSEQUEUE')
- DEFINE QL('WBIMBConnector/SYNCHRONOUSREQUESTQUEUE')
- DEFINE QL('WBIMBConnectorSYNCHRONOUSRESPONSEQUEUE')
- DEFINE QL('WBIMBConnectorMONITORQUEUE')
- DEFINE QL('PortConnector/ADMININQUEUE')
- DEFINE QL('PortConnector/ADMINOUTQUEUE')
- DEFINE QL('PortConnector/DELIVERYQUEUE')
- DEFINE QL('PortConnector/FAULTQUEUE')
- DEFINE QL('PortConnector/REQUESTQUEUE')
- DEFINE QL('PortConnector/RESPONSEQUEUE')
- DEFINE QL('PortConnector/SYNCHRONOUSREQUESTQUEUE')
- DEFINE QL('PortConnector/SYNCHRONOUSRESPONSEQUEUE')

2. **アダプターの構成**: **Connector Configurator** で「**ファイル**」->「**開く**」->「**ファイルから**」を選択し、*sample_folder* にある WBIMBConnector.cfg をロードします。アダプター構成プロパティーの値が以下に示す値と一致していることを確認します。または、一致するように変更します。Connector Configurator の使用方法の詳細については、67 ページの『第 6 章 データ・ハンドラーの構成』を参照してください。コネクター固有のプロパティーの詳細については、101 ページの『付録 B. このアダプターのコネクター固有のプロパティー』を参照してください。
   標準プロパティーに関しては、以下のプロパティーを設定する必要があります。
   - BrokerType: WMQI に設定します。
   - RepositoryDirectory: *sample_folder* ディレクトリーに設定します。
   - DuplicateEventElimination: true に設定します。
   - MonitorQueue: WBIMBConnector/MONITORQUEUE に設定します。

   . 以下のコネクター固有プロパティーを設定します。
   - ConfigurationMetaObject: Sample_WBIMB_MO_Config に設定します。
   - DataHandlerConfigMO: Sample_WBIMB_MO_DataHandler に設定します。
   - DataHandlerMimeType: text/xml に設定します。
   - ErrorQueue: queue://crossworlds.queue.manager/Samples/WBIMBConnector/ERROR に設定します。
   - InputQueue: queue://crossworlds.queue.manager/Samples/WBIMB/Item/WBIMBConnector に設定します。
   - UnsubscribedQueue: queue://crossworlds.queue.manager/Samples/WBIMBConnector/UNSUBSCRIBED に設定します。

- ArchiveQueue: queue://crossworlds.queue.manager/Samples/WBIMBConnector/ARCHIVE に設定します。

3. **ポート・コネクターの構成**: **Connector Configurator** を使用して、以下の標準プロパティーを設定します。
   - BrokerType: WMQI に設定します。
   - RepositoryDirectory: *sample_folder* ディレクトリーに設定します。
   - RequestQueue: WBIMBConnector/DELIVERYQUEUE (Adapter for WebSphere Business Integration Message Broker の DeliveryQueue プロパティーの値) に設定します。
   - DeliveryQueue: WBIMBConnector/REQUESTQUEUE (Adapter for WebSphere Business Integration Message Broker の RequestQueue プロパティーの値) に設定します。
4. **サポートするビジネス・オブジェクトの構成**: ビジネス・オブジェクトを使用するには、まずアダプターでそのビジネス・オブジェクトがサポートされていることが必要です。**Connector Configurator** で、Adapter for WebSphere Business Integration Message Broker の「**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、表 24 のビジネス・オブジェクトを追加します。さらに、「**メッセージ・セット ID**」を、サポートするビジネス・オブジェクトのそれぞれに固有な値に設定します。

表 24. JMS アダプターでサポートするサンプル・ビジネス・オブジェクト

ビジネス・オブジェクト名	メッセージ ID
Sample_WBIMB_MO_Config	1
Sample_WBIMB_MO_DataHandler	2
Sample_WBIMB_LegacyItem	3
Sample_WBIMB_LegacyItem_XMLDoc	4

Connector Configurator で、*sample_folder* にあるポート・コネクターの定義 PortConnector.cfg を開き、表 25 のサポートするビジネス・オブジェクトとメッセージ ID を追加します。

表 25. ポート・コネクターでサポートするサンプル・ビジネス・オブジェクト

ビジネス・オブジェクト名	メッセージ ID
Sample_WBIMB_LegacyItem	1
Sample_WBIMB_LegacyItem_XMLDoc	2

5. **メッセージ・フロー・プロジェクトの新規作成**
   a. **Message Brokers Toolkit** を開き、**メッセージ・フロー・プロジェクト**を新規作成します。作成したメッセージ・フロー・プロジェクトに、*sample_folder*/MSG_FLOW_PROJECT ディレクトリーのメッセージ・フローと ESQL ファイルをすべてインポートします。
   b. **サーバー・プロジェクト**を新規作成し、作成したサーバー・プロジェクトに *sample_folder*/Sample_WBIMB_Project のファイルをすべてインポートします。
   c. 「**ブローカー管理 (Broker Administration)**」パースペクティブの「**ドメイン (Domains)**」ビューの下に表示されたドメインに接続します。bar ファイル

Sample_WBIMB_bar.bar を、「**ブローカー管理 (Broker Administration)**」 パースペクティブの「**ブローカー管理ナビゲーター (Broker Administration Navigator)**」パネルから、ブローカーのデフォルトの実行グループに配置します。

6. **コネクター始動スクリプトの構成**

   Windows:

   a. Adapter for WebSphere Business Integration Message Broker のショートカットのプロパティーを開きます。

   b. リンク先の最後の引数として、-c に *<WBIMBConnector.cfg ファイルの絶対パスとファイル名>* を続けたものを追加します。例えば、次のようにします。

   ```
   -cProduct_Dir¥connectors¥WBIMB¥LegacyItem¥
   WBIMBConnector.cfg
   ```

   UNIX:

   a. ファイル *Product_Dir*/bin/connector_manager_WebSphereBIMessageBroker を開きます。AGENTCONFIG_FILE プロパティーの値として、-c に *<WBIMBConnector.cfg ファイルの絶対パスとファイル名>* を続けたものを設定します。例えば、次のようにします。

   ```
   AGENTCONFIG_FILE=Product_Dir/connectors/
   WebSphereBusinessIntegrationMessageBroker/
   samples/LegacyItem/WBIMBConnector.cfg
   ```

シナリオの実行

このトピックでは、シナリオの実行の概要と手順を示します。

シナリオの実行の概要

このチュートリアルには、要求処理シナリオとイベント処理シナリオがあります。これらのシナリオを実行するには、109 ページの『環境のセットアップ』に説明されているとおりにサンプルを構成する必要があります。

シナリオの実行のステップ

シナリオを実行するには、以下の手順を実行します。

1. **Adapter for WebSphere Business Integration Message Broker を始動します** (このアダプターがまだ実行されていない場合)。詳細については、41 ページの『コネクターの始動』を参照してください。
2. **Visual Test Connector を始動します** (このツールがまだ実行されていない場合)。詳細については、「*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。
3. **WebSphere Business Integration Message Broker アプリケーション・ブローカーを始動します** (このブローカーがまだ実行されていない場合)。詳細については、「*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。
4. **ポート・コネクターをシミュレートします。Visual Test Connector** を使用して、以下の手順でポート・コネクターのプロファイルを定義してください。

a. 「**Visual Test Connector**」メニューから「**ファイル**」->「**プロファイルを作成/選択**」を選択し、次に「**コネクター・プロファイル**」メニューから「**ファイル**」->「**新規プロファイル**」を選択します。

b. *sample_folder* にあるポート・コネクターの構成ファイル `PortConnector.cfg` を選択し、**コネクター名**と**ブローカー・タイプ**を構成してから 「**OK**」をクリックします。

c. 作成したプロファイルを選択し、「**OK**」をクリックします。

5. **要求処理シナリオをテストします。Visual Test Connector** を使用して、以下の手順でビジネス・オブジェクトのインスタンスを新規作成し、送信してください。

a. ビジネス・オブジェクト `Sample_WBIMB_LegacyItem_XMLDoc` のインスタンスを新規作成します。これを行うには、「**ビジネス・オブジェクト・タイプ**」ドロップダウン・ボックスでこのビジネス・オブジェクトを選択し、次に「**BO インスタンス**」で「**作成**」を選択します。このビジネス・オブジェクトは、XML データ・ハンドラーが必要とする XML ラッパーでしかありません。実際のデータを格納するには、このテスト用オブジェクトの内部に、子オブジェクト **LegacyItem** のインスタンスを新規作成する必要があります。デフォルト値は、必要に応じて変更してください。

b. 「**ビジネス・オブジェクトを送信**」をクリックしてメッセージを送信します。

6. **メッセージのデリバリーを確認します。**WebSphere MQ Explorer または同様のアプリケーションを使用してキュー `queue://crossworlds.queue.manager/Samples/WBIMB/Item/LegacyApp` を開き、フォーマットが **LI_UP** の新しいレガシー項目メッセージがアダプターから届いているかどうかを確認します。届いていれば、ビジネス・オブジェクトのペイロードが、WebSphere Business Integration Message Broker アプリケーション経由でレガシー・メッセージとして正常にデリバリーされていることになります。メッセージが見つからない場合は、キュー `queue://crossworlds.queue.manager/Samples/WBIMB/FAIL` を調べて、WebSphere Business Integration Message Broker アプリケーションがメッセージを処理できなかったのかどうかを確認します。処理されていないときは、「*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、トレースを使用可能にします。これで、WebSphere Business Integration Message Broker アプリケーションのどの部分でエラーが発生したのかを判別できるようになります。

7. **イベント処理シナリオをテストします。**ユーティリティー *sample_folder*¥`mqsiput.exe` を使用して、レガシー項目メッセージを WebSphere Business Integration Message Broker アプリケーション経由でアダプターにデリバリーします。`mqsiput` ユーティリティーの構文は次のとおりです。
`mqsiput [queue] [queue manager] < [message file]`
デリバリー・メッセージのサンプル `LegacyItem.txt` は、*sample_folder* ディレクトリーにあります。このチュートリアルでは、次のように引数を指定する必要があります。
```
mqsiput Samples/LegacyApp/Item/WBIMB crossworlds.queue.manager <
LegacyItem.txt
```
コマンド行でこのとおりに入力してください。正常に実行されると、項目メッセ

ージのサンプルが、レガシー・アプリケーションをシミュレートしている WebSphere Business Integration Message Broker アプリケーションにデリバリーされます。

注: ブローカーの実行グループには、ループバック・メッセージ・フローを追加することもできます。(このフローは、IBM Adapter for WebSphere Business Integration Message Broker のサンプル・メッセージ・フローとともに提供されています。) ループバック・メッセージ・フローを追加すると、アダプターから送信されたどのメッセージも、WebSphere Business Integration Message Broker アプリケーションで処理された後、アダプターに戻されるようになります。

WebSphere Business Integration Message Broker アプリケーションがメッセージを入力キューにデリバリーすると、アダプターはそのメッセージを取得して Sample_WBIMB_LegacyItem_XMLDoc ビジネス・オブジェクトに変換しようとします。アダプターにメッセージをポーリングさせる上で重要なことは、メッセージ・フォーマットを、Sample_WBIMB_LegacyItem_XMLDoc ビジネス・オブジェクトに関連付けられているフォーマット値 (メタオブジェクト Sample_WBIMB_MO_Config に指定されています) と確実に一致させることです。アダプターは、メッセージのフォーマットが LI_UP であることを識別すると、データ・ハンドラーを呼び出して、そのメッセージを、動詞が Update のビジネス・オブジェクト Sample_WBIMB_LegacyItem_XMLDoc に変換します。新しく作成されたビジネス・オブジェクトは、続いて WebSphere Business Integration Broker にデリバリーされ、その後 Visual Test Connector に再デリバリーされます。

8. **メッセージのデリバリーを確認します。**これまでに説明したステップをすべて完了すると、シナリオが実践されます。その結果、Adapter for WebSphere Business Integration Message Broker ではメッセージを取得して Sample_WBIMB_LegacyContact ビジネス・オブジェクトに変換できるようになり、また Sample_WBIMB_LegacyContact ビジネス・オブジェクトをメッセージに変換できるようになります。

付録 D. Common Event Infrastructure

WebSphere Business Integration Server Foundation には、Common Event Infrastructure の稼働に必要な Common Event Infrastructure Server Application が含まれています。WebSphere Application Server Foundation は、任意のシステムにインストールできます (アダプターがインストールされているマシンである必要はありません)。

WebSphere Application Server Application Client には、アダプターと Common Event Infrastructure Server Application との対話に必要なライブラリーがあります。WebSphere Application Server Application Client は、アダプターと同じシステムにインストールする必要があります。アダプターは、構成可能 URL を使用して WebSphere Application Server (WebSphere Business Integration Server Foundation 内の) に接続します。

Common Event Infrastructure サポートは、このリリースでサポートされる統合ブローカーを使用して利用することができます。

必須のソフトウェア

Common Event Infrastructure が動作するには、アダプターに必要なソフトウェア前提条件に加えて、次のものがインストールされている必要があります。

- WebSphere Business Integration Server Foundation 5.1.1
- WebSphere Application Server Application Client 5.0.2、5.1、または 5.1.1

  (WebSphere Application Server Application Client 5.1.1 は、WebSphere Business Integration Server Foundation 5.1.1 と共に提供されています。)

注: Common Event Infrastructure は、HP-UX または Linux プラットフォームではサポートされていません。

Common Event Infrastructure の使用可能化

Common Event Infrastructure 機能は、標準プロパティー `CommonEventInfrastructure` および `CommonEventInfrastructureContextURL` で使用可能にし、Connector Configurator で構成されます。デフォルトでは、Common Event Infrastructure は使用不可になっています。`CommonEventInfrastructureContextURL` プロパティーを使用すると、Common Event Infrastructure サーバーの URL を構成できます (詳細については、本書の付録『標準プロパティー』を参照してください)。

Common Event Infrastructure アダプター・イベントの取得

Common Event Infrastructure が使用可能になっている場合、アダプターは、次のアダプター・イベントへマップする Common Event Infrastructure イベントを生成します。

- アダプターの開始

- アダプターの停止
- アダプター・エージェントからのタイムアウトに対するアプリケーションの応答
- アダプター・エージェントから発行された doVerbFor 呼び出し
- アダプター・エージェントからの gotApplEvent 呼び出し

アダプターによって生成された Common Event Infrastructure イベントを別のアプリケーション (「コンシューマー・アプリケーション」)で受信するには、そのアプリケーションが Common Event Infrastructure イベント・カタログを使用してアプリケーション・イベントとそのプロパティーの定義を判別する必要があります。コンシューマー・アプリケーションが送信側のアプリケーションのイベントを利用するには、イベントがイベント・カタログ内で定義されている必要があります。

本書の付録『Common Event Infrastructure イベント・カタログ定義』には、WebSphere Business Information アダプターの場合に、コンシューマー・アプリケーションが検索すべきイベント記述子とプロパティーを示す、XML 形式のメタデータが含まれています。

関連情報

Common Event Infrastructure の詳細については、次の URL で入手できる WebSphere Business Integration Server Foundation の資料で Common Event Infrastructure の情報を参照してください。

http://publib.boulder.ibm.com/infocenter/ws51help

コンシューマー・アプリケーションが検索すべきアダプター生成イベント記述子およびプロパティーを示すサンプル XML メタデータは、『Common Event Infrastructure イベント・カタログ定義』を参照してください。

Common Event Infrastructure イベント・カタログ定義

Common Event Infrastructure イベント・カタログには、その他のアプリケーションで照会できるイベント定義が含まれています。次に、XML メタデータを使用した、通常のアダプター・イベントのイベント定義のサンプルを示します。別のアプリケーションを記述している場合は、イベント・カタログ・インターフェースを使用してイベント定義を照会することができます。イベント定義とそれらを照会する方法については、オンラインの IBM WebSphere Server Foundation インフォメーション・センターから入手できる Common Event Infrastructure の資料を参照してください。

WebSphere Business Integration アダプターの場合、イベント・カタログで定義する必要のある拡張データ・エレメントは、ビジネス・オブジェクトのキーです。それぞれのビジネス・オブジェクト・キーにイベント定義が必要です。したがって、どのアダプターでも、start adapter、stop adapter、timeout adapter、およびあらゆる doVerbFor イベント (例えば create、update、または delete) など、さまざまなイベントが、イベント・カタログ内に対応するイベント定義を持っている必要があります。

次のセクションで、start adapter、stop adapter、およびイベント request または delivery 用の XML メタデータの例を示します。

“start adapter” メタデータの XML 形式

```
<eventDefinition name="startADAPTER"
      parent="event">
    <property name ="creationTime" //Comment: example value would be
 "2004-05-13T17:00:16.319Z"
         required="true" />
    <property name="globalInstanceId" //Comment: Automatically generated
 by Common Event Infrastructure
         required="true"/>
    <property name="sequenceNumber"    //Comment: Source defined number
for messages to be sent/sorted logically
         required="false"/>
    <property name="version"    //Comment: Version of the event
         required="false"
         defaultValue="1.0.1"/>
    <property name="sourceComponentId"
         path="sourceComponentId"
         required="true"/>
    <property name="application"    //Comment: The name#version of the
source application generating the event. Example is "SampleConnector#3.0.0"
         path="sourceComponentId/application"         required="false"/>
    <property name="component"   //Comment: This will be the name#version
 of the source component.
         path="sourceComponentId/component"
         required="true"
         defaultValue="ConnectorFrameWorkVersion#4.2.2"/>
    <property name="componentIdType"    //Comment: specifies the format
and meaning of the component
          path="sourceComponentId/componentIdType"
          required="true"
          defaultValue="Application"/>
    <property name="executionEnvironment"
 //Comment: Identifies the environment the application is running
 in...example is "Windows 2000#5.0"
          path="sourceComponentId/executionEnvironment"
          required="false" />
     <property name="location"    //Comment: The value of this is the
 server name...example is "WQMI"
          path="sourceComponentId/location"
          required="true"/>
    <property name="locationType"   //Comment specifies the format and
     meaning of the location
          path="sourceComponentId/locationType"
          required="true"
          defaultValue="Hostname"/>
     <property name="subComponent"     //Comment:further distinction
of the logical component
          path="sourceComponentId/subComponent"
          required="true"
          defaultValue="AppSide_Connector.AgentBusinessObjectManager"/>
     <property name="componentType"      //Comment: well-defined name
used to characterize all instances of this component
          path="sourceComponentId/componentType"
          required="true"
          defaultValue="ADAPTER"/>
     <property name="situation"   //Comment: Defines the type of
 situation that caused the event to be reported
          path="situation"
          required="true"/>
     <property name="categoryName="    //Comment: Specifies the type
of situation for the event
         path="situation/categoryName"
         required="true"
         defaultValue="StartSituation"/>
    <property name="situationType"    //Comment: Specifies the type
of situation and disposition of the event
```

```
        path="situation/situationType"
        required="true"
   <property name="reasoningScope" //Comment: Specifies the scope
 of the impact of the event
        path="situation/situationType/reasoningScope"
        required="true"
        permittedValue="INTERNAL"
        permittedValue="EXTERNAL"/>
   <property name="successDisposition" //Comment: Specifies the
 success of event
        path="situation/situationType/successDisposition"
        required="true"
        permittedValue="SUCCESSFUL"
        permittedValue="UNSUCCESSFUL" />
   <property name="situationQualifier"  //Comment: Specifies the
 situation qualifiers for this event
        path="situation/situationType/situationQualifier"
        required="true"
        permittedValue="START_INITIATED"
        permittedValue="RESTART_INITIATED"
        permittedValue="START_COMPLETED" />
</eventDefinition>
```

″stop adapter″ メタデータの XML 形式

“stop adapter” のメタデータは、次の場合を除き、“start adapter” のメタデータと同じです。

- categoryName プロパティーのデフォルト値は、StopSituation です。

```
<property name="categoryName="
 //Comment: Specifies the type
 of situation for the event
             path="situation/categoryName"
             required="true"
             defaultValue="StopSituation"/>
```

- situationQualifier プロパティーで許可される値が異なります。“stop adapter” では、次のようになります。

```
<property name="situationQualifier"
 //Comment: Specifies the situation qualifiers for this event
          path="situation/situationType/situationQualifier"
          required="true"
          permittedValue="STOP_INITIATED"
          permittedValue="ABORT_INITIATED"
          permittedValue="PAUSE_INITIATED"
          permittedValue="STOP_COMPLETED"
 />
```

“timeout adapter” メタデータの XML 形式

“timeout adapter” のメタデータは、次の場合を除き、“start adapter” および “stop adapter” のメタデータと同じです。

- categoryName プロパティーのデフォルト値は、ConnectSituation です。

```
<property name="categoryName="
 //Comment: Specifies the type
 of situation for the event
             path="situation/categoryName"
             required="true"
             defaultValue="ConnectSituation"/>
```

- situationQualifier プロパティーで許可される値が異なります。“timeout adapter” では、次のようになります。

```
<property name="situationQualifier"  //Comment: Specifies
 the situation qualifiers for this event
          path="situation/situationType/situationQualifier"
          required="true"
          permittedValue="IN_USE"
          permittedValue="FREED"
          permittedValue="CLOSED"
          permittedValue="AVAILABLE"
/>
```

″request″ または ″delivery″ メタデータの XML 形式

この XML 形式の最後には、拡張データ・エレメントがあります。adapter request および delivery イベントの拡張データ・エレメントは、処理中のビジネス・オブジェクトからのデータを表します。このデータには、ビジネス・オブジェクトの名前、ビジネス・オブジェクトのキー (外部またはローカル)、親ビジネス・オブジェクトの子であるビジネス・オブジェクトが含まれます。次に、子ビジネス・オブジェクトが親と同じデータ (名前、キー、およびその子ビジネス・オブジェクト) に分割されます。このデータは、イベント定義の拡張データ・エレメントで示されます。このデータは、どのビジネス・オブジェクト、キー、および子ビジネス・オブジェクトが処理されているかによって変わります。このイベント定義の拡張データは、1 つの例ですが、キー EmployeeId を持つ Employee というビジネス・オブジェクトと、キー EmployeeId を持つ子ビジネス・オブジェクト EmployeeAddress を示しています。このパターンを使用して、特定のビジネス・オブジェクトに存在するすべてのデータを表すことができます。

```
<eventDefinition name="createEmployee"     //Comment: This
 extension name is always the business object verb followed by the business
 object name
          parent="event">
    <property name ="creationTime"  //Comment: example value would be
"2004-05-13T17:00:16.319Z"
         required="true" />
    <property name="globalInstanceId" //Comment: Automatically generated
 by Common Event Infrastructure
         required="true"/>
    <property name="localInstanceId"    //Comment: Value is business
 object verb+business object name+#+app name+ business object identifier
         required="false"/>
    <property name="sequenceNumber"    //Comment: Source defined number
for messages to be sent/sorted logically
         required="false"/>
    <property name="version"  //Comment: Version of the event...value is
 set to 1.0.1
         required="false"
         defaultValue="1.0.1"/>
    <property name="sourceComponentId"
         path="sourceComponentId"
         required="true"/>
    <property name="application"    //Comment: The name#version of the
 source application generating the event...example is
 "SampleConnector#3.0.0"
         path="sourceComponentId/application"
         required="false"/>
    <property name="component"   //Comment: This will be the name#version
of the source component.
         path="sourceComponentId/component"
         required="true"
```

```
        defaultValue="ConnectorFrameWorkVersion#4.2.2"/>
    <property name="componentIdType"     //Comment: specifies the format
 and meaning of the component
        path="sourceComponentId/componentIdType"
        required="true"
        defaultValue="Application"/>
    <property name="executionEnvironment" //Comment: Identifies the
 environment#version the app is running in...example is "Windows 2000#5.0"
        path="sourceComponentId/executionEnvironment"
        required="false" />
    <property name="instanceId" //Comment: Value is business object
  verb+business object name+#+app name+ business object identifier
         path="sourceComponentId/instanceId"
         required="false"
    <property name="location"   //Comment: The value of this is the
server name...example is "WQMI"
         path="sourceComponentId/location"
         required="true"/>
     <property name="locationType" //Comment specifies the format and
 meaning of the location
         path="sourceComponentId/locationType"
         required="true"
         defaultValue="Hostname"/>
     <property name="subComponent"  //Comment:further distinction of the
 logical component-in this case the value is the name of the business
 object
         path="sourceComponentId/subComponent"
         required="true"/>
     <property name="componentType"      //Comment: well-defined name used
 to characterize all instances of this component
         path="sourceComponentId/componentType"
         required="true"
         defaultValue="ADAPTER"/>
     <property name="situation" //Comment: Defines the type of
situation that caused the event to be reported
         path="situation"
         required="true"/>
    <property name="categoryName"    //Comment: Specifies the type
 of situation for the event
        path="situation/categoryName"
        required="true"
        permittedValue="CreateSituation"
        permittedValue="DestroySituation"
        permittedValue="OtherSituation" />
    <property name="situationType"    //Comment: Specifies the type
of situation and disposition of the event
        path="situation/situationType"
        required="true"
    <property name="reasoningScope" //Comment: Specifies the scope
of the impact of the event
        path="situation/situationType/reasoningScope"
        required="true"
        permittedValue="INTERNAL"
        permittedValue="EXTERNAL"/>
    <property name="successDisposition" //Comment: Specifies the
 success of event
        path="situation/situationType/successDisposition"
        required="true"
        permittedValue="SUCCESSFUL"
        permittedValue="UNSUCCESSFUL" />
    <extendedDataElements name="Employee" //Comment: name of business
 object itself
            type="noValue"
            <children name="EmployeeId"
                 type="string"/>  //Comment: type is one of the
 permitted values within Common Event Infrastructure documentation
            <children name="EmployeeAddress"
```

```
                    type="noValue"/>
                       <children name="EmployeeId"
                           type="string"/>
                        -
                        -
                        -
    </extendedDataElements
</eventDefinition>
```

付録 E. Application Response Measurement

このアダプターは、アプリケーションの可用性、サービス・レベル・アグリーメント、およびキャパシティー・プランニングを管理するための Application Response Measurement アプリケーション・プログラミング・インターフェース (API) に対応しています。ARM を備えたアプリケーションは、IBM Tivoli Monitoring for Transaction Performance に参加して、トランザクション・メトリックに関するデータの収集と検討を実行できます。

Application Response Measurement 計測サポート

このアダプターは、アプリケーションの可用性、サービス・レベル・アグリーメント、およびキャパシティー・プランニングを管理するための Application Response Measurement アプリケーション・プログラミング・インターフェース (API) に対応しています。ARM を備えたアプリケーションは、IBM Tivoli Monitoring for Transaction Performance に参加して、トランザクション・メトリックに関するデータの収集と検討を実行できます。

必須のソフトウェア

ARM が動作するには、アダプターに必要なソフトウェア前提条件に加えて、次のものがインストールされている必要があります。

- WebSphere Application Server 5.0.1 (IBM Tivoli Monitoring for Transaction Performance サーバーを含む)。これは、アダプターと同じシステムにインストールする必要はありません。
- IBM Tivoli Monitoring for Transaction Performance v. 5.2 Fixpack 1。これは、IBM Tivoli Monitoring for Transaction Performance サーバーがあるシステムを指すため、アダプターがインストールされ構成されているシステムにインストールする必要があります。

Application Response Measurement サポートは、このリリースでサポートされる統合ブローカーを使用して利用することができます。

注: Application Response Measurement 計測は、HP-UX (すべてのバージョン) および Red Hat Linux 3.0 を除き、この IBM WebSphere Business Integration Adapters リリースでサポートされるすべてのオペレーティング・システムでサポートされます。

Application Response Measurement の使用可能化

ARM 計測は、Connector Configurator の標準プロパティー `TivoliMonitorTransactionPerformance` を “True” に設定することによって使用できます。デフォルトでは、ARM サポートは使用不可になっています (詳細については、本書の付録『標準プロパティー』を参照してください)。

トランザクション・モニター

ARM が使用可能になっている場合、モニターされるトランザクションは、サービス要求とイベント・デリバリーです。トランザクションは、サービス要求またはイベント・デリバリーの開始から終わりまで測定されます。Tivoli Monitoring for Transaction Performance コンソールに表示されるトランザクションの名前は、SERVICE REQUEST または EVENT DELIVERY で始まります。名前の次の部分は、ビジネス・オブジェクト動詞 (CREATE、RETRIEVE、UPDATE または DELETE など) です。名前の最後の部分は、“EMPLOYEE” などのビジネス・オブジェクト名です。例えば、従業員の作成用のイベント・デリバリーのトランザクション名は、EVENT DELIVERY CREATE EMPLOYEE のようになります。あるいは、SERVICE REQUEST UPDATE ORDER などです。

デフォルトでは、サービス要求またはイベント・デリバリーのタイプごとに次の測定基準が収集されます。

- 最小トランザクション時間
- 最大トランザクション時間
- 平均トランザクション時間
- 合計トランザクション実行

ユーザー (または WebSphere Application Server のシステム管理者) は、Tivoli Monitoring for Transaction Performance コンソール内から特定のトランザクションのディスカバリー・ポリシーとリスナー・ポリシーを構成することによって、どのアダプター・イベントに対してどの測定基準を表示するかを選択できます (『関連情報』を参照してください)。

関連情報

詳細については、IBM Tivoli Monitoring for Transaction Performance の資料を参照してください。特に、アダプターによって生成される測定基準のモニターと管理については、「*IBM Tivoli Monitoring for Transaction Performance User's Guide*」を参照してください。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

[ハ行]

[マ行]

[ヤ行]

[ラ行]

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

W

X

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

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

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

〒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で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

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

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

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

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

IBM Corporation
577 Airport Blvd., Suite 800
Burlingame, CA 94010
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すが、有償の場合もあります。

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

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証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

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、単に目標を示しているものです。

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。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

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。

汎用プログラミング・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。

i5/OS
IBM
IBM ロゴ
AIX
AIX 5L
CICS
CrossWorlds
DB2
DB2 Universal Database
HelpNow
IMS
Informix
iSeries
Lotus
Lotus Domino
Lotus Notes
MQIntegrator
MQSeries
MVS
OS/400
Passport Advantage
pSeries
Redbooks
SupportPac
WebSphere
z/OS

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。

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。

Intel、Intel (ロゴ)、Intel Inside、Intel Inside (ロゴ)、Intel Centrino、Intel Centrino (ロゴ)、Celeron、Intel Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation 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。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

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。

この製品には、Eclipse Project (http://www.eclipse.org/) により開発されたソフトウェアが含まれています。

WebSphere Business Integration Adapter Framework バージョン 2.6.0.3

Printed in Japan

日本アイ・ビー・エム株式会社
〒106-8711　東京都港区六本木3-2-12