IBM WebSphere Business Integration
Adapters

Adapter for ACORD XML ユーザーズ・ガイド

V1.0.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、 93 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 ACORD XML バージョン 1.0.0、および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

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

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

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

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

(URL は、変更になる場合があります)

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

原　典： IBM WebSphere Business Integration Adapters
Adapter for ACORD XML User Guide
V1.0.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 2004.1

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

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

目次

本書について

IBM(R) WebSphere(R) 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・プロセ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。

本書では、IBM WebSphere Business Integration Adapter for ACORD XML のインストール、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。

対象読者

本書は、WebSphere Business Integration システムをお客様のサイトでサポートおよび管理する、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 ACORD XML 仕様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。

関連資料

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。

本書には、次の 2 つの文書「*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)*」、および「*WebSphere InterChange Server インプリメンテーション・ガイド*」への参照が多数含まれています。本書を印刷される場合は、これらの文書も印刷されることをお勧めします。

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が必要な場合、アダプターを WebSphere Message Brokers (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、WebSphere Business Integration Message Broker) とともに使用する場合、およびアダプターを WebSphere Application Server とともに使用する場合は、以下のサイトを参照してください。

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

- アダプターを InterChange Server とともに使用する場合は、以下のサイト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

- Message Brokers (WebSphere MQ Integrator Broker、WebSphere MQ Integrator、および WebSphere Business Integration Message Broker) 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

これらのサイトには、資料をダウンロード、インストール、および表示するための簡単な指示が掲載されています。

表記上の規則

本書は下記の規則に従って編集されています。

`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。
斜体、*斜体*	変数名または相互参照を示します。
青のアウトライン	青のアウトラインは、マニュアルをオンラインで表示するときにのみ見られるもので、相互参照用のハイパーリンクを示します。 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、オプションのパラメーターです。
...	構文の記述行の場合、省略符号 ... は直前のパラメーターが繰り返されることを示します。例えば、`option[,...]` は複数のオプションをコンマで区切って入力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・パスに円記号 (¥) を使用します。UNIX システムの場合には、円記号 (¥) はスラッシュ (/) に置き換えてください。すべての製品のパス名は、使用システムで製品がインストールされたディレクトリーを基準とした相対パス名です。
%text% および *$text*	% 記号で囲まれたテキストは、Windows の `text` システム変数またはユーザー変数の値を示します。UNIX 環境での同等の表記は *$text* です。これは、UNIX 環境変数 *text* の値を示します。
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 1.0.x での新機能

Version 1.0.x は Adapter for ACORD の最初のリリースです。

第 1 章 概要

Connector for ACORD XML は、WebSphere Business Integration Adapter for ACORD XML のランタイム・コンポーネントです。コネクターを使用すると、WebSphere 統合ブローカーは、ACORD XML メッセージを送受信するアプリケーションとの間でビジネス・オブジェクトを交換することができます。この章では、コネクター・コンポーネントおよび関連するビジネス・インテグレーションのシステム・アーキテクチャーについて説明します。

コネクターは、アプリケーション固有のコンポーネントおよびコネクター・フレームワークから構成されます。アプリケーション固有のコンポーネントには、特定のアプリケーションに合わせて作成されたコードが含まれます。コネクター・フレームワークは統合ブローカーとアプリケーション固有のコンポーネントの間の仲介役として機能し、そのコードはどのコネクターにも共通です。コネクター・フレームワークは、統合ブローカーとアプリケーション固有のコンポーネントの間で、以下のサービスを提供します。

- ビジネス・オブジェクトの送受信
- 始動メッセージおよび管理メッセージの交換の管理

本書では、アプリケーション固有のコンポーネントおよびコネクター・フレームワークについて説明します。ここでは、これらの両方のコンポーネントを「コネクター」と呼んでいます。

統合ブローカーとコネクターの関係の詳細については、「*IBM WebSphere InterChange Server システム管理ガイド*」を参照してください。

注: WebSphere Business Integration Adapter は、いずれも統合ブローカーとともに動作します。ACORD XML 用のコネクターは、以下の統合ブローカーとともに動作します。

- InterChange Server 統合ブローカー。詳細については、「*テクニカル入門 (IBM WebSphere InterChange Server)*」を参照してください。
- WebSphere Application Server (WAS) 統合ブローカー。詳細は、「*アダプター実装ガイド (WebSphere Application Server)*」を参照してください。

ACORD 標準の概要

ACORD は、保険業界の一連の電子標準を管理する団体です。これらの標準は、生命保険、健康保険、年金保険、およびその他各種の保険業界のさまざまな側面を記述するXML バージョン 1.0 準拠のマークアップ言語です。

Adapter for ACORD XML 1.0.0 は、ACORD XML 標準の 2 つの主要部分である XML for Life および Property & Casualty (P&C) をサポートしています。 ACORD メッセージがここに記述されるスキーマで XML 文書として表示されるとき、そのメッセージは WebSphere Business Integration システムでビジネス・オブジェクトに変換され、ビジネス情報としてインターネットを介して、エンタープライズ内の各種ドメイン、および異種アプリケーション間で交換されます。

Adapter for ACORD XML は、WebSphere Business Integration システムと ACORD メッセージング・アプリケーション間でデータを移動させる手段を提供します。これについては、5 ページの『コネクター・アーキテクチャー』で説明します。

XML for Life

XML for Life は、異種プラットフォームのビジネス・パートナー間で、リアルタイムのメッセージと情報の共用を可能にする、生命保険業界の標準ファミリーです。XML for Life は、Life Data Model が持つ豊富なデータ語彙と XML のすべての利点を結びつけることで、これを実現しています。

XML for Life で表わされる ACORD メッセージは、プラットフォームにも言語にも依存しないので、システム、言語、通信方式およびメッセージング・インフラストラクチャーが異種であっても、メッセージを交換できます。

ACORD は、Life 標準をスキーマ・ファイルの形式で提供しています。 Life の最新バージョン 2.8.0 は、次の 3 つの主な仕様で構成されています。

- XMLife。Life Data Model の XML 版。
- TxLife。XMLife コンテンツのラッパー。
- XTbML。任意の次元の表データを作成する構造体。

注: 最新バージョンの Life 2.8.0 は、後方互換性があります。

注: ACORD XML for Life は、DTD をサポートしていません。

ACORD メッセージング・アプリケーションは、通常、ACORD スキーマ仕様に基づいて、XML メッセージを作成します。ACORD スキーマ仕様は、ACORD の Web サイトからダウンロードできます。これらのスキーマの実装は、クライアントによって異なる場合があります。

注: XMLifE と TXLifE は、後方互換性用に XML ネームスペースを 1 つだけ使用します。http://ACORD.org/Standards/Life/2

ACORD XML for Property & Casualty

ACORD XML for Property and Casualty (P&C) 標準では、課金、請求、個人用、業務用、特殊分野および保証の各トランザクション用の要求メッセージおよび応答メッセージの両方を含む、損害および傷害保険のトランザクションを定義しています。

Adapter for ACORD XML の環境

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。

- 『ブローカーとの互換性』
- 4 ページの『アダプターの規格』
- 4 ページの『アダプター・プラットフォーム』
- 4 ページの『アダプターの依存関係』
- 5 ページの『ロケール依存データ』

ブローカーとの互換性

アダプターが使用するアダプター・フレームワークは、アダプターと通信する統合ブローカーのバージョンとの互換性を備えている必要があります。ACORD XML のアダプターのバージョン 1.0 は、以下のアダプター・フレームワークと統合ブローカーでサポートされています。

- アダプター・フレームワーク: WebSphere Business Integration Adapter Framework、バージョン:
  - 2.1
  - 2.2
  - 2.3.x
  - 2.4
- 統合ブローカー:
  - WebSphere InterChange Server、バージョン:
    - 4.11
    - 4.2
    - 4.2.1
    - 4.2.x
  - WebSphere MQ Integration Broker、バージョン 2.1.0
  - WebSphere Business Integration Message Broker、バージョン 5.0
  - WebSphere Studio Application Developer Integration Edition バージョン 5.0.1 が組み込まれた WebSphere Application Server Enterprise バージョン 5.0.2

例外については、「リリース情報」を参照してください。

注: 統合ブローカーのインストールおよびその前提条件に関する説明については、以下の資料を参照してください。

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

WebSphere Message Brokers (WebSphere MQ Integrator Broker、WebSphere MQ Integrator、および WebSphere Business Integration Message Broker) 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*」、および Message Broker のインストール関連資料を参照してください。これらの資料には、 Web サイト: http://www.ibm.com/software/integration/mqfamily/library/manualsa/ から検索できるものもあります。 WebSphere Application Server については、「アダプター実装ガイド (WebSphere Application Server)」および http://www.ibm.com/software/webservers/appserv/library.html にある資料を参照してください。

アダプターの規格

アダプターは以下のバージョンの ACORD 標準に準拠して書かれています。

- XML for Life 2.8
- XML for Property & Casualty 1.4.1

アダプター・プラットフォーム

このアダプターは以下のプラットフォームでサポートされます。

- Windows 2000
- AIX 5.1、5.2
- Solaris 7、8
- HP-UX 11i

アダプターの依存関係

このアダプターには、以下のソフトウェア前提条件と、その他の依存関係があります。

- アダプターは、WebSphere MQ Integrator Broker バージョン 2.1 および WebSphere InterChange Server バージョン 4.2 でサポートされています。
- WebSphere Business Integration Message Broker バージョン 5.0 を使用している場合は、その統合ブローカーに加えて CSD02 をインストールする必要があります。
- コネクターは、WebSphere MQ または WebSphere MQ 5.1、5.2、[1]したがって、これらのソフトウェア・リリースのいずれかを、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

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、5.2.2 以上 (WebSphere MQ バージョン 5.2 の場合) とします。これにより、エンコードがサポートされていないことによるエラーを回避できる場合があります。

信に適した WebSphere MQ ソフトウェアのバージョンについては、ご使用のプラットフォーム (Windows または Unix) のインストール・ガイドを参照してください。

- さらに、IBM WebSphere MQ Java クライアント・ライブラリーも必要です。

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て、指定された言語でメッセージ・テキストを送れるように国際化されています。コネクターが、ある文字コード・セットを使用する場所から別の文字コード・セットを使用する場所にデータを転送する場合、文字変換を実行してデータの意味を保持します。

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 システムの大部分のコンポーネントは Java で作成されています。したがって、大部分のインテグレーション・コンポーネント間で行われるデータ転送には、文字変換は必要ありません。

エラー・メッセージや通知メッセージを個々の国や地域に合った適切な言語で記録するには、個々の環境に合わせて `Locale` 標準構成プロパティーを構成する必要があります。構成プロパティーの詳細については、55 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。

コネクター・アーキテクチャー

Connector for ACORD XML を使用すると、IBM WebSphere Business Integration Collaborations は、データの変更が生じた場合に ACORD XML メッセージを発行または受信するアプリケーションとの間で、ビジネス・オブジェクトを非同期に交換することができます。

コネクターは、MQ にインプリメントされている Java™ Message Service (JMS) 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、これにより保証付きイベント・デリバリーも可能になります。

Connector for ACORD XML

Connector for ACORD XML は、メタデータ主導型です。メッセージ・ルーティングおよびフォーマット変換は、イベント・ポーリング手法によって開始されます。

コネクターはキューから ACORD XML 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、その後それらのオブジェクトをコラボレーションに送達します。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・オブジェクトを受信し、同じデータ・ハンドラーを使用してそれらを ACORD XML メッセージに変換し、その後メッセージを ACORD メッセージング・アプリケーションに送達します。

Connector for ACORD XML は、ACORD XML メッセージを処理するのに XML データ・ハンドラーを使用します。Connector for ACORD XML で使用するこのデータ・ハンドラーの構成方法については、「*データ・ハンドラー・ガイド*」の第 3 章『XML data handler』を参照してください。

XML およびビジネス・オブジェクト

メッセージ処理に使用されるビジネス・オブジェクトと動詞のタイプは、ACORD XML メッセージ・ヘッダーに含まれる FORMAT フィールドを基本としています。コネクターはメタオブジェクト項目を使用して、ビジネス・オブジェクト名および動詞を決定します。各メッセージに対して、ACORD XML 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られた、ビジネス・オブジェクト名と動詞を保管するためのメタオブジェクトを作成します。

オプションで、コネクターに渡されるビジネス・オブジェクトに子として追加される動的なメタオブジェクトを構成することもできます。子メタオブジェクトの値は、コネクター全体を対象として指定されている静的なメタオブジェクトに指定されている値をオーバーライドします。子メタオブジェクトが定義されていない場合や、必要な変換プロパティーが定義されていない場合は、コネクターはデフォルトで、その値に対する静的なメタオブジェクトを調べます。単一の静的なコネクター・メタオブジェクトの代わりに、またはそれを補足するために、1 つ以上の動的な子メタオブジェクトを指定することができます。

コネクターは複数の入力キューのポーリングが可能で、各キューをラウンドロビン方式でポーリングして、各キューから指定された数のメッセージを検索します。ポーリング中に検索するメッセージごとに、コネクターは動的な子メタオブジェクト (ビジネス・オブジェクトに指定されている場合) を追加します。子メタオブジェクトの値はコネクターに対し、メッセージ・フォーマットとメッセージが検索された入力キューの名前を、ビジネス・オブジェクトの属性に取り込むよう指示できます。

入力キューからメッセージを検索する場合、コネクターは、メッセージ・ヘッダーに含まれる FORMAT フィールドに対応するビジネス・オブジェクト名を探します。次に、メッセージ本文が適切なビジネス・オブジェクトの新しいインスタンスとともにデータ・ハンドラーに渡されます。メッセージ・フォーマットに対応するビジネス・オブジェクト名が見つからない場合、メッセージ本文のみがデータ・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内容を正常に取り込むことができたら、コネクターはそのオブジェクトがサブスクライブされているかどうかを確認し、gotApplEvents() メソッドを使用して InterChange Server に配信します。

アプリケーションとコネクターの通信

コネクターは、IBM WebSphere MQ にインプリメントされている Java Message Service (JMS) 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ビジネス・データおよびイベントを非同期に送受信できるように設計されています。

メッセージ要求

図 1 にメッセージ要求通信を示します。doVerbFor() メソッドがコラボレーションから WebSphere Business Integration システムのビジネス・オブジェクトを受信すると、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・ハンドラーはビジネス・オブジェクトを XML に変換し、コネクターはそ

れをメッセージとしてキューに発行します。JMS 層は適切な呼び出しを行ってキュー・セッションを開き、メッセージを発送します。

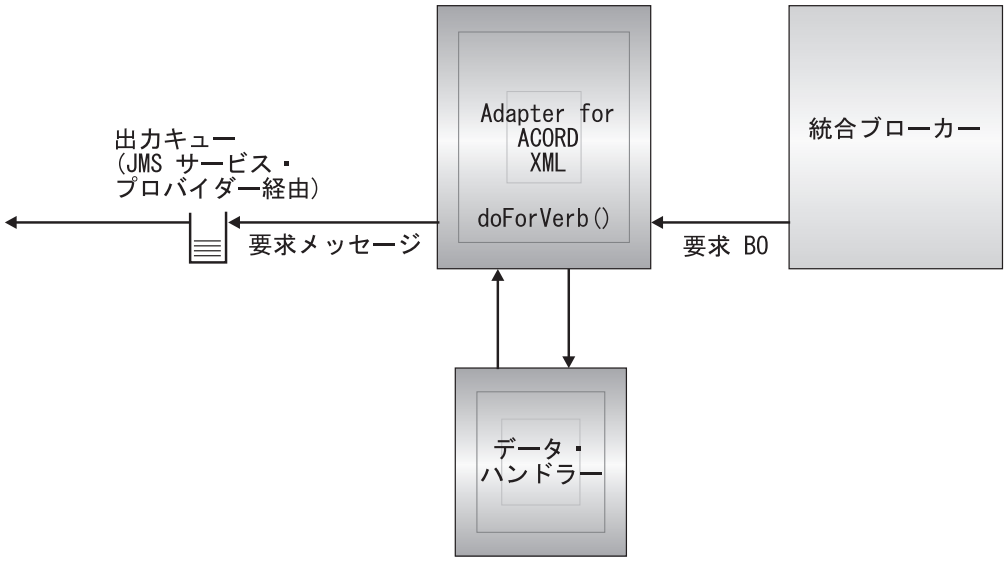

図 1. アプリケーションとコネクター間の通信方式: メッセージ要求

イベント・デリバリー

図 2に、イベント・デリバリーの方向を示します。`pollForEvents()` メソッドは、次の適用可能なメッセージを入力キューから取得します。メッセージは進行中のキューにステージングされて、処理が完了するまでそこに存在します。コネクターは、静的または動的メタオブジェクトを使用してメッセージ・タイプ (つまり XML) がサポートされているかどうかを判断します。コネクターはその後、メッセージを構成済みのデータ・ハンドラーに渡します。データ・ハンドラーは、メッセージを WebSphere Business Integration システムのビジネス・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・タイプに対して設定されている変換プロパティーが反映されます。コネクターは次に、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るかどうかを判断します。サブスクライブされる場合、`gotApplEvents()` メソッドはビジネス・オブジェクトを InterChange Server に配信し、メッセージは進行中のキューから除去されます。

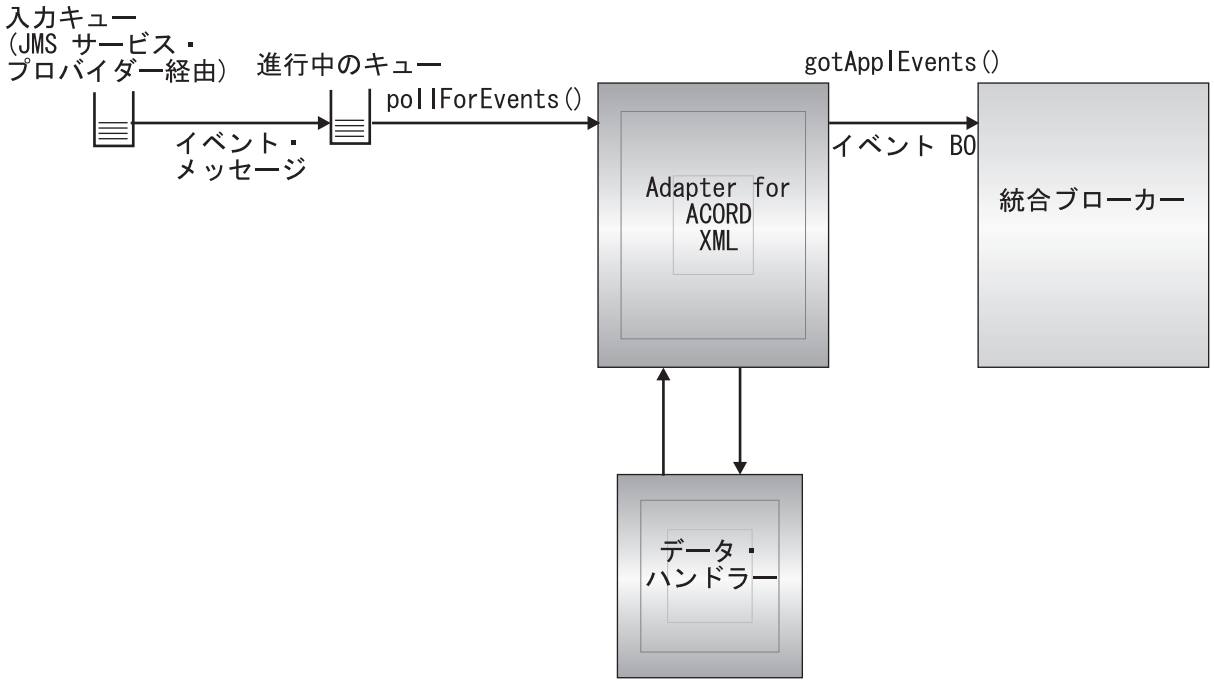

図 2. アプリケーションとコネクター間の通信方式: イベント・デリバリー

保証付きイベント・デリバリー

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、イベントが逸失したり、コネクターのイベント・ストア、JMS イベント・ストア、および宛先の JMS キューの間でイベントが 2 度送信されたりするのを防ぐことができます。JMS を有効にするには、コネクターの `DeliveryTransport` 標準プロパティーを JMS に設定する必要があります。この構成により、コネクターは JMS トランスポートを使用することになり、コネクターと統合ブローカーの間の後続の通信はすべて、このトランスポートを介して行われます。JMS トランスポートでは、メッセージがその宛先に最終的に到達することが保証されます。JMS トランスポートの役割は、トランザクションのキュー・セッションの開始後、コミットが発行されるまでメッセージが確実にキャッシュされるようにすることです。障害が起こったとき、あるいはロールバックが発行されたときには、メッセージは廃棄されます。

注: 保証付きイベント・デリバリー機能を使用していない場合は、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 pollForEvents() メソッド内で gotApplEvent() メソッドを呼び出した時点) から、コネクターがイベント・レコードを削除することによってイベント・ストアを更新するまで (あるいは、「イベント通知済み」状況に更新するまで) の間、小さいウィンドウが表示され、障害が発生した場合に通知されます。このウィンドウに障害が発生した場合は、イベントが送信されてもイベント・レコードが「進行中」の状況のままイベント・ストア内に残ります。コネクターは再始動後に、このイベント・レコードがまだイベント・ストア内に残っているのを検出し、結果的にイベントを再度送信することになります。

保証付きイベント・デリバリー機能は、JMS イベント・ストアを持つ JMS 対応コネクター用、あるいは JMS イベント・ストアを持たない JMS 対応コネクター用に構成できます。保証付きイベント・デリバリーを使用するため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・フレームワークは、ビジネス・オブジェクトを ICS 統合ブローカーに引き渡すことができない場合、オブジェクトを FaultQueue (UnsubscribedQueue や ErrorQueue ではなく) に入れ、状況表示および問題の記述を生成します。FaultQueue メッセージは、MQRFH2 形式で書き込まれます。

イベント処理

イベント通知では、コネクターはデータベース・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。アプリケーションまたはその他の ACORD ソフトウェアが ACORD XML メッセージを生成してそれらを MQ メッセージ・キューに格納すると、イベントが発生します。

検索

コネクターは `pollForEvents()` メソッドを使用して、一定の間隔で MQ キューをポーリングしてメッセージの有無を確認します。メッセージが見つかった場合、コネクターはメッセージを MQ キューから取り出して検討し、そのフォーマットを判断します。メッセージ・フォーマットがコネクターの静的オブジェクト内に定義されている場合、コネクターは、メッセージ本文とそのフォーマットに関連付けられたビジネス・オブジェクトの新しいインスタンスを構成済みのデータ・ハンドラーに渡します。この場合、データ・ハンドラーがビジネス・オブジェクトへのデータの取り込みと動詞の指定を行う必要があります。静的なメタオブジェクト内にメッセージ・フォーマットが定義されていない場合、コネクターはメッセージ本文のみをデータ・ハンドラーに渡します。この場合、データ・ハンドラーは、そのメッセージに応じた適切なビジネス・オブジェクトを決定および作成し、そこにデータを取り込む必要があります。イベント失敗のシナリオについては、51 ページの『エラー処理』を参照してください。

コネクターはメッセージを処理する際に、まず入力キューに対してトランザクションのセッションを開きます。このトランザクションのアプローチによって、ビジネス・オブジェクトがコラボレーションに 2 回配信される可能性があります。これは、コネクターがビジネス・オブジェクトのサブミットには成功するが、キュー内のトランザクションのコミットに失敗することが原因です。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のキューに移動します。メッセージは、処理が完了するまで進行中のキュー内に保留されます。コネクターが処理中に不意に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復元されずに進行中のキュー内に保留されます。

注: JMS サービス・プロバイダーを含むトランザクション・セッションでは、キュー上で要求されるすべてのアクションが、イベントがキューから除去される前に実行およびコミットされる必要があります。したがって、コネクターは、キューからメッセージを検索する際に以下の 3 つのことが発生するまでは検索にコミットしません。1) メッセージがビジネス・オブジェクトに変換される。2) ビジネス・オブジェクトが `gotApplEvents()` メソッドによって InterChange Server に配信される。3) 戻り値が受信される。

リカバリー

初期設定時に、コネクターは進行中のキューをチェックし、コネクターのシャットダウンが原因と考えられる、処理が不完全なメッセージの有無を確かめます。コネクター構成プロパティー InDoubtEvents を使用すると、このようなメッセージのリカバリー処理のための 4 つのオプション (始動時の失敗、再処理、無視、およびエラー・ログ記録) から 1 つを指定できます。

Fail on startup

FailOnStartup オプションを使用すると、初期設定中に進行中のキューからメッセージが検出された場合にコネクターはエラー・ログを記録し、すぐにシャットダウンします。検出されたメッセージを検討して適切な処置をとる、すなわち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のはユーザーまたはシステム管理者の責任です。

Reprocess

Reprocess オプションを使用すると、初期設定中に進行中のキューからメッセージが検出された場合に、コネクターは以降のポーリングでこれらのメッセージを最初に処理します。進行中のキューの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のメッセージの処理を開始します。

Ignore

Ignore オプションを使用すると、初期設定中に進行中のキューからメッセージが検出された場合に、コネクターはそれらを無視して、シャットダウンは行いません。

Log error

LogError オプションを使用すると、初期設定中に進行中のキューからメッセージが検出された場合に、コネクターはエラー・ログを記録しますが、シャットダウンはしません。

アーカイブ

コネクター・プロパティー ArchiveQueue が指定されて有効なキューを示す場合、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・キューに入れます。ArchiveQueue が定義されていない場合は、メッセージは処理後に廃棄されます。アンサブスクライブされたメッセージまたはエラーがあるメッセージのアーカイブの詳細については、49 ページの『第 3 章 ビジネス・オブジェクトの作成または変更』の 51 ページの『エラー処理』を参照してください。

注: JMS の規則によると、検索されたメッセージをすぐに別のキューに発行することはできません。メッセージのアーカイブおよび再デリバリーを可能にするためには、コネクターはまず、元のメッセージの本文とヘッダー (該当する場合) を複製した新規のメッセージを作成します。JMS サービス・プロバイダーとの矛盾を避けるために、JMS 必須フィールドのみを複製します。この結果、フォーマット・フィールドが、アーカイブまたは再デリバリーされるメッセージ用に複製された唯一の追加メッセージ・プロパティーです。

ビジネス・オブジェクト要求

ビジネス・オブジェクト要求は、InterChange Server がビジネス・オブジェクトを doVerbFor() メソッドに送信する時に処理されます。構成済みのデータ・ハンドラーを使用して、コネクターはビジネス・オブジェクトを WebSphere MQ メッセージに変換し、それを発行します。データ・ハンドラーの要件を除いては、処理されるビジネス・オブジェクトのタイプに関する要件はありません。

動詞の処理

コネクターは、各ビジネス・オブジェクトの動詞に基づいたコラボレーションによって、渡されるビジネス・オブジェクトを処理します。コネクターはビジネス・オブジェクト・ハンドラーと doForVerb() 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・オブジェクト動詞をサポートします。

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

注: Create、Update、および Delete の各動詞を含むビジネス・オブジェクトは、非同期的にも同期的にも発行できます。デフォルト・モードは非同期です。コネクターは、Retrieve、Exists、または Retrieve by content の各動詞を含むビジネス・オブジェクトについては、非同期のデリバリーをサポートしません。したがって、Retrieve、Exists、Retrieve by Content の各動詞のデフォルト・モード (唯一のモードでもある) は、同期です。

Create、Update、および Delete

Create、Update、および Delete の各動詞を含むビジネス・オブジェクトの処理は、オブジェクトが非同期または同期のどちらの方式で発行されているかによって異なります。

非同期デリバリー

Create、Update、および Delete の各動詞を含むビジネス・オブジェクトに関しては、これがデフォルトのデリバリー・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。メッセージが配信されると、コネクターは BON_SUCCESS、さもなければ BON_FAIL を戻します。

注: コネクターは、メッセージが受信されたかどうか、またはアクションが実行されたかどうかを確認することはできません。

同期デリバリー

コネクター・プロパティーに replyToQueue が定義されていて、ビジネス・オブジェクトの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

モードで要求を発行します。コネクターはその後で応答を待機して、受信アプリケーションによって適切なアクションが実行されることを確認します。

ACORD XML の場合、コネクターは最初に、次の表に示すヘッダーを持つメッセージを発行します。

フィールド	説明	値
Format	フォーマット名	変換プロパティーで定義された出力フォーマット。IBM 要件に合わせて 8 文字までに切り捨てられます (例: MQSTR)。
MessageType	メッセージ・タイプ	受信側アプリケーションからの応答が期待されない場合は、MQMT_DATAGRAM*。 MQMT_REQUEST* 応答を予期する場合
Report	必要なレポート・メッセージのオプション。	応答メッセージが期待される場合、このフィールドには次のように値が取り込まれます。処理が成功した場合の肯定処理レポートが必要なことを示す、MQRO_PAN* 。処理が失敗した場合の否定処理レポートが必要なことを示す、MQRO_NAN* 。生成されたレポートの相関 IDが初めに発行された要求のメッセージ ID と等しくなければならないことを示す、MQRO_COPY_MSG_ID_TO_CORREL_ID* 。
ReplyToQueue	応答キューの名前	応答メッセージが期待される場合は、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 (永続性)	MQPER_PERSISTENT*
Expiry	メッセージの存続時間	MQEI_UNLIMITED*

* は、IBM によって定義されている定数を示します。

上記の表に示したメッセージ・ヘッダーには、メッセージ本文が続きます。メッセージ本文は、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の肯定処理レポートと否定処理レポート両方の返送が期待されていることを示すよう、設定されます。メッセージを発行したスレッドは、受信アプリケーションが要求を処理できるかどうかを示す応答メッセージを待機します。

アプリケーションは、コネクターから同期要求を受信すると、ビジネス・オブジェクトを処理して、次の表に示すレポート・メッセージを発行します。

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

* は、IBM によって定義されている定数を示します。

動詞	フィードバック・フィールド	メッセージ本文
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

動詞	フィードバック・フィールド	メッセージ本文
	VALDUPES FAIL	(オプション) エラー・メッセージ

WebSphere MQ フィードバック・コード	該当する CrossWorlds 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
MQFB_APPL_FIRST + 8	APP_RESPONSE_TIMEOUT (コネクター・エージェントが即時終了します)

* 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。

ビジネス・オブジェクトが処理できた場合、アプリケーションはフィードバック・フィールドを MQFB_PAN (または特定の WebSphere Business Integration システムの値) に設定して、レポート・メッセージを作成します。オプションで、アプリケーションはメッセージ本体に変更箇所を含むビジネス・オブジェクトを直列化して取り込むことができます。ビジネス・オブジェクトが処理されなかった場合、アプリケーションはフィードバック・フィールドを MQFB_NAN (または特定の WebSphere Business Integration システムの値) に設定して、レポート・メッセージを作成します。その後、オプションでメッセージ本体にエラー・メッセージを組み込みます。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、replyTo 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、デフォルトでは、応答メッセージを検索する際に応答メッセージの correlationID と要求メッセージの messageID とを付き合わせます。次に、コネクターは要求を発行したスレッドを通知します。応答のフィードバック・フィールドによって、コネクターはメッセージ本体に、ビジネス・オブジェクトかエラー・メッセージのいずれかが含まれていることを、予測します。ビジネス・オブジェクトが要求されたがメッセージ本文に値が含まれていない場合、コネクターは単に、Request 操作で最初に InterChange Server によって発行されたものと同じビジネス・オブジェクトを戻します。エラー・メッセージが要求され、メッセージ本文に値が含まれていない場合、汎用エラー・メッセージが応答コードとともに InterChange Server に戻されます。ただし、メッセージ・セレクターを使用して、要求に対する応答メッセージの識別、フィルター操作、およびアダプターでの識別方法を制御することもできます。このメッセージ・セレクターの機能は JMS の機能です。これは、同期要求処理のみに適用されます (下記参照)。

メッセージ選択子を使用した応答メッセージのフィルター操作: コネクターは、同期要求処理の対象となるビジネス・オブジェクトを受信するときに、動詞のアプリ

ケーション固有情報に response_selector ストリングが存在するかどうかを検査します。response_selector が定義されていない場合は、コネクターは、上記の相関 ID を使用して応答メッセージを識別します。

response_selector が定義されている場合は、名前と値の組が以下の構文で格納されている必要があります。

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

メッセージ・セレクター・ストリングは応答を一意的に識別しなければならず、値は以下のように単一引用符で囲まれていなければなりません。

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

上記の例では、要求メッセージを発行した後、アダプターは correlationID が「Oshkosh」に等しい応答メッセージがあるかどうか ReplyToQueue をモニターします。アダプターはこのメッセージ・セレクターに一致する最初のメッセージを検索し、応答としてディスパッチします。

オプションで、アダプターは実行時置換を行い、要求ごとに固有のメッセージ・セレクターを生成できるようにします。メッセージ・セレクターの代わりに、整数を中括弧で囲んだ形式 (例: '{1}') でプレースホルダーを指定します。この後にコロンを置き、置換に使用する属性をコンマで区切ってリストします。プレースホルダーの整数は、置換に使用する属性に対する指標として機能します。例えば、以下のメッセージ・セレクターでは、

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

{1} をセレクターの最初の属性 (この例では子オブジェクト MyDynamicMO の属性 CorrelationId) の値で置換するようにアダプターに指示します。属性 CorrelationID の値が 123ABC の場合は、アダプターは以下の基準で作成されたメッセージ・セレクターを生成し、使用して

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

応答メッセージを識別します。

以下のように複数の置換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} をトップレベルのビジネス・オブジェクトの属性 PrimaryId の値で置換し、{2} を子コンテナー・オブジェクト Address の 5 番目の AddressId の値で置換します。この方法では、応答メッセージ・セレクターでビジネス・オブジェクトおよびメタオブジェクトの任意の属性を参照できます。Address[4].AddressId を使用した深い検索の方法については、JCDK API のマニュアル (getAttribute メソッド) を参照してください。

以下の場合は、実行時にエラーが報告されます。

- {} 記号の間に整数以外の値を指定した場合
- 属性が定義されていない指標を指定した場合

- 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、リテラル値「{」または「}」をメッセージ・セレクターに含める場合は、それぞれ「{{」または「{}」を使用してください。また、これらの文字を属性値に入れることもできますが、この場合は最初の「{」は不要です。エスケープ文字の使用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・セレクターを以下のように解決します。

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

コネクターは、属性値で「{」、「}」、「:」、「;」などの特殊文字を検出した場合は、それらの文字を照会ストリングに直接挿入します。これにより、アプリケーション固有情報の区切り文字としても機能する特殊文字を照会ストリングに含めることができます。

属性値からリテラル・ストリング置換を抽出する方法を次の例に示します。

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

`MyDynamicMO.CorrelationID` に値 `{A:B}C;D` が含まれている場合は、コネクターは、このメッセージ・セレクターを `JMSCorrelationID LIKE '{A:B}C;D' and CompanyName='A{P'` と解決します。

応答セレクター・コードの詳細については、JMS 1.0.1 仕様を参照してください。

カスタム・フィードバック・コードの作成: コネクター・プロパティー FeedbackCodeMappingMO を指定して、WebSphere MQ のフィードバック・コードを拡張し、デフォルトの解釈をオーバーライドできます。このプロパティーを使用すると、WebSphere Business Integration システム固有の戻り状況値が WebSphere MQ のフィードバック・コードにマップされるメタオブジェクトを作成できます。(メタオブジェクトを使用して) フィードバック・コードに割り当てられた戻り状況は、InterChange Server に渡されます。詳しくは、26 ページの『FeedbackCodeMappingMO』を参照してください。

Retrieve、Exists、および Retrieve By Content

Retrieve、Exists、および Retrieve By Content 動詞を含むビジネス・オブジェクトは、同期デリバリーのみをサポートします。コネクターはこれらの動詞を含むビジネス・オブジェクトを、create、update、および delete 用に定義された同期デリバリーの場合と同様に処理します。ただし、Retrieve、Exists、および Retrieve By Content 動詞を使用している場合、`responseTimeout` と `replyToQueue` が必要になります。さらに、Retrieve By Content および Retrieve 動詞の場合は、トランザクションを完了するには、メッセージ本文に直列化ビジネス・オブジェクトが取り込まれる必要があります。

次の表に、これらの動詞の応答メッセージを示します。

動詞	フィードバック・フィールド	メッセージ本文
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	

ロケール依存データの処理

コネクターは、2 バイト文字セットをサポートして、指定された言語でメッセージ・テキストを送れるように国際化されています。ある文字コードを使用する場所から別の文字コード・セットを使用する場所へデータを転送する場合、コネクターは、そのデータの意味が伝わるように文字変換を実行します。

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、最もよく知られた文字コード・セット (単一バイトとマルチバイトの両方) の文字エンコードが含まれています。WebSphere Business Integration システムの大部分のコンポーネントは Java で作成されています。したがって、大部分のインテグレーション・コンポーネント間で行われるデータ転送には、文字変換は必要ありません。

エラー・メッセージや通知メッセージを個々の国や地域に合った適切な言語で記録するには、個々の環境に合わせて `Locale` 標準構成プロパティーを構成する必要があります。構成プロパティーの詳細については、55 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。

共通の構成タスク

インストールが終了したら、コネクターを開始する前に構成する必要があります。このセクションでは、多くの開発者が実行する必要がある、構成タスクと開始タスクの一部について概説します。

アダプターのインストール

インストールするコンポーネントとインストール先の場所については、21 ページの『第 2 章 アダプターのインストールと構成』を参照してください。

コネクター・プロパティーの構成

コネクターの構成プロパティーには、標準構成プロパティーとコネクター固有の構成プロパティーという 2 つのタイプがあります。これらのプロパティーの一部は、デフォルト値を持っており、変更する必要はありません。コネクターを実行する前に、これらのプロパティーの一部の値を設定する必要があります。詳しくは、21 ページの『第 2 章 アダプターのインストールと構成』を参照してください。

Adapter for ACORD XML のコネクター・プロパティーを構成する際は、以下のことに注意してください。

- コネクター・プロパティー HostName に指定した値が、WebSphere MQ サーバーのホストの値と一致すること。
- コネクター・プロパティー Port に指定した値が、キュー・マネージャーのリスナーのポートの値と一致すること。
- コネクター・プロパティー Channel に指定した値が、キュー・マネージャーのサーバー接続チャネルと一致すること。
- コネクター・プロパティー InputQueue、InProgressQueue、ArchiveQueue、ErrorQueue、UnsubscribeQueue の各キューの URI が有効で、実際に存在すること。

通知なしで要求を送信するためのコネクターの構成

通知なしで要求を送信するように (デフォルトの非同期モード、別名「fire and forget」とも呼ぶ) コネクターを構成するには、以下のようにします。

- 送信する要求を表し、コネクター用に構成したデータ・ハンドラーと互換性もある、ビジネス・オブジェクトを作成します。
- 静的または動的メタオブジェクトのいずれかを使用して、ターゲット・キューとフォーマットを指定します。静的および動的メタオブジェクトの詳細については、31 ページの『静的メタオブジェクト』 と 37 ページの『動的な子メタオブジェクト』を参照してください。
- (静的または動的) メタオブジェクトのプロパティー ResponseTimeout を -1 に設定します。これにより、コネクターは応答をチェックせずに、ビジネス・オブジェクトを発行するようになります。
- 詳しくは、11 ページの『Create、Update、および Delete』、30 ページの『メタオブジェクト属性の構成』、および 49 ページの『第 3 章 ビジネス・オブジェクトの作成または変更』を参照してください。

要求を送信して通知を取得するためのコネクターの構成

要求を送信し通知を取得するように (同期イベント処理)、コネクターを構成するには以下のようにします。

- 『通知なしで要求を送信するためのコネクターの構成』で説明されている手順に従いますが、コネクターが応答を待機する時間を示すために正の ResponseTimeout 値を指定する点は異なります。
- コネクターが待つ応答メッセージの具体的な内容については、11 ページの『Create、Update、および Delete』を参照してください。提示されている要件を応答メッセージが満たしていない場合、コネクターがエラーを報告するか、または応答メッセージを認識できないことがあります。30 ページの『メタオブジェクト属性の構成』と 49 ページの『第 3 章 ビジネス・オブジェクトの作成または変更』のセクションも参照してください。

静的メタオブジェクトの構成

静的メタオブジェクトには、ビジネス・オブジェクトについて指定したアプリケーション固有情報と、コネクターがビジネス・オブジェクトを処理する方法が格納さ

れます。静的メタオブジェクトは、コネクターの始動時に、ビジネス・オブジェクトの処理にコネクターが必要とするすべての情報をコネクターに提供します。

インプリメント時にビジネス・オブジェクトごとに送信先キューがわかっている場合は、静的メタオブジェクトを使用します。静的メタオブジェクトを作成して構成するには、以下のようにします。

- 31 ページの『静的メタオブジェクト』の手順に従います。
- コネクター固有プロパティー ConfigurationMetaObject に静的メタオブジェクトの名前を指定して、コネクターが静的メタオブジェクトにサブスクライブするようにします。詳しくは、24 ページの『コネクター固有のプロパティー』を参照してください。

動的メタオブジェクトの構成

コネクターが、シナリオに応じて、ビジネス・オブジェクトの処理を変える必要がある場合は、動的メタオブジェクトを使用します。これは、ビジネス・オブジェクトに追加する子オブジェクトです。動的メタオブジェクトは、(実行時に) コネクターに要求の処理方法を知らせます。ビジネス・オブジェクトの処理に必要な情報をすべてコネクターに提供する静的メタオブジェクトとは違い、 動的メタオブジェクトは特定のシナリオの処理に必要なロジックの追加情報だけを提供します。動的メタオブジェクトを作成して構成するには、以下のようにします。

- 動的メタオブジェクトを作成し、それを子として要求ビジネス・オブジェクトに追加します。
- 動的メタオブジェクトをコネクターに発行する前に、ターゲット・キュー、メッセージ・フォーマットなどの情報を動的メタオブジェクトに取り込む追加ロジックで、コラボレーションをプログラムします。

コネクターは動的メタオブジェクトをチェックして、動的メタオブジェクトの情報を使用してビジネス・オブジェクトの処理方法を決定します。詳しくは、37 ページの『動的な子メタオブジェクト』を参照してください。

MQMD フォーマットの構成

MQMD は、メッセージ記述子です。MQMD には、メッセージがアプリケーション間で送信される場合に、アプリケーション・データに添付される制御情報が格納されます。使用する静的または動的メタオブジェクトのいずれかに、MQMD 属性 OutputFormat の値を指定する必要があります。詳しくは、11 ページの『Create、Update、および Delete』を参照してください。

キュー URI の構成

Adapter for ACORD XML で使用するキューを構成するには、次のようにします。

- すべてのキューを Uniform Resource Identifier (URI) として指定します。構文は次のとおりです。

  queue://<キュー・マネージャーの名前>/<実際のキューe>

- コネクター固有の構成プロパティーに、キュー・マネージャーのホストを指定します。
- ターゲット・アプリケーションが MQMD ヘッダーだけを予期し、JMS クライアントが使用する 拡張 MQRFH2 ヘッダーを処理できない場合は、キュー URI に

?targetClient=1 を追加します。詳細については、29 ページの『キューの Uniform Resource Identifier (URI)』 および WebSphere MQ のプログラミング・ガイドを参照してください。

データ・ハンドラーの構成

データ・ハンドラーの構成方法は、2つあります。

- コネクター固有プロパティー DataHandlerClassName にデータ・ハンドラーのクラス名を指定します。詳しくは、24 ページの『コネクター固有のプロパティー』を参照してください。
- あるいは、コネクター固有プロパティー DataHandlerMimeType に MIME タイプを、DataHandlerConfigMO にその MIME タイプの構成を定義するデータ・ハンドラー・メタオブジェクトを、それぞれ指定します。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。

始動スクリプトの変更

コネクターの始動方法の詳細については、 21 ページの『第 2 章 アダプターのインストールと構成』 を参照してください。コネクターの始動前に、コネクターのプロパティーを構成する必要があります。また、始動ファイルを変更する必要もあります。

- クライアント・ライブラリーの場所を指すように、start_connector スクリプトを必ず変更してください。クライアント・ライブラリーのバージョンを複数インストールしたり、ご使用の WebSphere MQ サーバーに対応していないバージョンをインストールしないでください。詳しくは、47 ページの『始動ファイルの構成』を参照してください。

第 2 章 アダプターのインストールと構成

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

Connector for ACORD XML をインストールするには、以下の作業を実行する必要があります。

- **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。
- **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。『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。

アダプターと関連ファイルのインストール

WebSphere Business Integration adapter 製品のインストールの詳細については、以下の WebSphere Business Integration Adapters Infocenter のサイトにある「*WebSphere Business Integration Adapters インストール・ガイド*」を参照してください。

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

インストール済みファイルの構造

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について説明します。

注: 通常、Windows 環境でも UNIX 環境でも、WebSphere MQ と JMS は別のディレクトリーにインストールされます。例えば AIX システムの場合、WebSphere MQ はデフォルトでは `/var/mqm/` にインストールされますが、JMS は `/usr/mqm/java/lib` にインストールされます。ルーチン `/usr` 関連のシステム管理タスクによる削除を回避するために、JMS のインストール先を `/var/mqm/java/lib` に変更することをお勧めします。 同様に Windows でも、

通常、WebSphere MQ は ¥Program Files¥WebSphere MQ にインストールされ、JMS は ¥Program Files¥IBM¥MQSerires¥Java にインストールされます。WebSphere MQ コネクターの始動スクリプト内のクラスパスを、適切に更新してください。

Windows のファイル構造

インストーラーは、コネク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¥WebSphereMQConnector ディレクトリーにインストールして、「スタート」メニューにコネクターへのショートカットを追加します。

次の表に、コネクターが使用する Windows のファイル構造の説明と、インストーラーによるコネクターのインストールを選択した場合に自動的にインストールされるファイルを示します。

ProductDir のサブディレクトリー	説明
connectors¥WebSphereMQ¥CWWebSphereMQ.jar	WebSphere MQ コネクターに使用されるクラスだけを含みます。
connectors¥WebSphereMQConnector¥start_WebSphereMQ.bat	コネクターの始動スクリプト (Windows 2000)。
connectors¥messages¥WebSphereMQConnector.txt	コネクターのメッセージ・ファイル。
repository¥WebSphereMQ¥CN_WebSphereMQ.txt	コネクターのリポジトリー定義。
connectors¥WebSphereMQ¥samples¥LegacyContact¥WebSphereMQConnector.cfg	WebSphere MQ 構成ファイルのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥PortConnector.cfg	ポート・コネクター構成ファイルのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥Sample_WebSphereMQ _LegacyContact.xsd	スキーマのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥Sample_WebSphereMQ _MO_Config.xsd	メタオブジェクトのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥Sample_WebSphereMQ _MO_DataHandler.xsd	データ・ハンドラー・メタオブジェクトのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥Sample_WebSphereMQ _MO_DataHandler_DelimitedConfig.xsd	区切りデータ・ハンドラー・メタオブジェクトのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥Sample_WebSphereMQ _DynMO_Config.xsd	動的なメタオブジェクトのサンプル
connectors¥WebSphereMQ¥samples¥LegacyContact¥JMSPropertyPairs.xsd	JMS プロパティーのサンプル

注: すべての製品のパス名は、使用システムで製品がインストールされたディレクトリーを基準とした相対パス名です。

UNIX のファイル構造

インストーラーは、コネク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/WebSphereMQConnector ディレクトリーにインストールします。

次の表に、コネクターが使用する UNIX のファイル構造の説明と、インストーラーによるコネクターのインストールを選択した場合に自動的にインストールされるファイルを示します。

ProductDir のサブディレクトリー	説明
connectors/WebSphereMQ/CWWebSphere MQ.jar	Connector for ACORD XML のみに使用されるクラスを含みます。

ProductDir のサブディレクトリー	説明
connectors/WebSphereMQ/start_WebSphereMQ.sh	コネクターのシステム始動スクリプト。このスクリプトは、汎用のコネクター・マネージャー・スクリプトから呼び出されます。System Manager の「コネクター構成」画面をクリックすると、インストーラーは、このコネクター・マネージャー・スクリプト用にカスタマイズされたラッパーを作成します。コネクターの始動および停止には、このカスタマイズされたラッパーを使用してください。
connectors/messages/WebSphereMQ/Connector.txt	コネクターのメッセージ・ファイル。
repository/WebSphereMQ/CN_WebSphere MQ.txt	コネクターのリポジトリー定義。
connectors/WebSphereMQ/samples/LegacyContact/WebSphereMQConnector.cfg	WebSphere MQ 構成ファイルのサンプル
connectors/WebSphereMQ/samples/LegacyContact/PortConnector.cfg	ポート・コネクター構成ファイルのサンプル
connectors/WebSphereMQ/samples/LegacyContact/Sample_WebSphereMQ _LegacyContact.xsd	スキーマのサンプル
connectors/WebSphereMQ/samples/LegacyContact/Sample_WebSphereMQ _MO_Config.xsd	メタオブジェクトのサンプル
connectors/WebSphereMQ/samples/LegacyContact/Sample_WebSphereMQ _MO_DataHandler.xsd	データ・ハンドラー・メタオブジェクトのサンプル
connectors/WebSphereMQ/samples/LegacyContact/Sample_WebSphereMQ _MO_DataHandler_DelimitedConfig.xsd	区切りデータ・ハンドラー・メタオブジェクトのサンプル
connectors/WebSphereMQ/samples/LegacyContact/Sample_WebSphereMQ _DynMO_Config.xsd	動的なメタオブジェクトのサンプル
connectors/WebSphereMQ/samples/LegacyContact/JMSPropertyPairs.xsd	JMS プロパティーのサンプル

注: すべての製品のパス名は、使用システムで製品がインストールされたディレクトリーを基準とした相対パス名です。

コネクター構成

コネクターの構成プロパティーには、標準構成プロパティーとアダプター固有の構成プロパティーという 2 つのタイプがあります。アダプターを実行する前に、これらのプロパティーの値を設定する必要があります。

コネクターのプロパティーを構成するには、Connector Configurator を使用します。

- Connector Configurator の説明と段階的な手順については、73 ページの『付録 B. Connector Configurator』を参照してください。
- 標準コネクター・プロパティーの説明については、24 ページの『標準コネクター・プロパティー』、および 55 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。
- コネクター固有のプロパティーの詳細については、24 ページの『コネクター固有のプロパティー』を参照してください。

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・プロパティーの値の変更が必要になることがあります。
AgentTraceLevel など一部のコネクター構成プロパティーへの変更は、即時に有効になります。その他のコネクター・プロパティーへの変更を有効にするには、変更後にコンポーネントまたはシステムを再始動する必要があります。あるプロパティーが動的 (即時に有効になる) か静的 (コネクター・コンポーネントまたはシステムを再始動する必要がある) かを判別するには、Connector Configurator の「コネクター・プロパティー」ウィンドウ内の「更新メソッド」列を参照してください。

標準コネクター・プロパティー

標準構成プロパティーにより、すべてのコネクターによって使用される情報が提供されます。標準構成プロパティーの資料については、55 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。

注: Connector Configurator で構成プロパティーを設定するときは、BrokerType プロパティーで使用するブローカーを指定します。このプロパティーの値を設定すると、使用するブローカーに関連するプロパティーが「Connector Configurator」ウィンドウに表示されます。

コネクター固有のプロパティー

コネクター固有の構成プロパティーは、コネクター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。

次の表に、アダプターに対するコネクター固有の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。

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.queuemanager/MQCONN.ARCHIVE	いいえ
CCSID	*キュー・マネージャーの接続に使用する文字セット*	null	いいえ
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	いいえ
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	いいえ
DefaultVerb	*コネクターによってサポートされる動詞*	Create	
ErrorQueue	*未処理のメッセージ・キュー*	queue://crossworlds.queuemanager/MQCONN.ERROR	いいえ
FeedbackCodeMappingMO	*フィードバック・コード・メタオブジェクト*		いいえ
HostName	*WebSphere MQ サーバー*		はい
InDoubtEvents	FailOnStartup Reprocess IgnoreLogError	Reprocess	いいえ
InputQueue	*ポーリング・キュー*	queue://crossworlds.queuemanager/MQCONN.IN	いいえ
InProgressQueue	*進行中のイベント・キュー*	queue://crossworlds.queuemanager/MQCONN.IN_PROGRESS	いいえ
PollQuantity	*InputQueue プロパティーで指定した各キューから取得するメッセージの数*	1	いいえ

名前	指定可能な値	デフォルト値	必須
Port	*WebSphere MQ リスナー用に設定されるポート*		はい
ReplyToQueue	コネクターからの要求発行時に応答メッセージが配信されるキュー	queue://crossworlds.queuemanager/MQCONN.REPLYTO	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.queuemanager/MQCONN.UNSUBSCRIBE	いいえ
UseDefaults	true または false	false	

ApplicationPassword
WebSphere MQ へのログインに、UserID と使用されるパスワード。

デフォルト = なし。

ApplicationPassword がブランクのままか、除去された場合は、コネクターは WebSphere MQ が提供するデフォルト・パスワードを使用します。*

ApplicationUserName
WebSphere MQ へのログインに Password と使用されるユーザー ID。

デフォルト = なし。

ApplicationUserName がブランクのままか、除去された場合は、コネクターは WebSphere MQ が提供するデフォルト・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/MQCONN.ARCHIVE

CCSID
キュー・マネージャーの接続に使用する文字セット。このプロパティーの値は、キュー URI の CCSID プロパティーの値と一致する必要があります。29 ページの『キューの Uniform Resource Identifier (URI)』を参照してください。

デフォルト = null

Channel
コネクターが WebSphere MQ 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 = なし。

Channel がブランクのままか、除去された場合は、コネクターは WebSphere MQ が提供す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静的なメタオブジェクトの名前です。

デフォルト = 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。

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

DefaultVerb
ポーリング中にデータ・ハンドラーが動詞を設定しなかった場合に、着信ビジネス・オブジェクト内に設定される動詞を指定します。

デフォルト= Create

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/MQCONN.ERROR

FeedbackCodeMappingMO
メッセージの受信を InterChange Server に同期的に知らせるのに使用される、デフォルトのフィードバック・コードをオーバーライドして再割り当てすることができます。このプロパティーを使用すると、それぞれの属性名がフィードバック・コードを表すと解釈されるメタオブジェクトを指定できます。フィードバック・コードに対応する値は、InterChange Server に渡される戻り状況値です。デフォルトのフィードバック・コードのリストについては、11 ページの『同期デリバリー』を参照してください。コネクターは、WebSphere MQ 固有のフィードバック・コードを表す、次の属性値を受け入れます。

- MQFB_APPL_FIRST
- MQFB_APPL_FIRST_OFFSET_*N*、 *N* は整数 (MQFB_APPL_FIRST + *N* の値として解釈される)
- MQFB_NONE
- MQFB_PAN
- MQFB_NAN

コネクターは、次の WebSphere Business Integration システム固有の状況コードを、メタオブジェクトの属性値として受け入れます。

- SUCCESS
- FAIL
- APP_RESPONSE_TIMEOUT

- MULTIPLE_HITS
- UNABLE_TO_LOGIN
- VALCHANGE
- VALDUPES

次の表に、メタオブジェクトのサンプルを示します。

属性名	デフォルト値
MQFB_APPL_FIRST	SUCCESS
MQFB_APPL_FIRST + 1	FAIL
MQFB_APPL_FIRST + 2	UNABLE_TO_LOGIN

デフォルト = なし。

HostName
WebSphere MQ をホストしているサーバーの名前。

デフォルト = なし。

InDoubtEvents
コネクターの予期しないシャットダウンのために、処理が完了していない進行中イベントの処理方法を指定します。初期化中に進行中のキューにイベントが見つかった場合に実行するアクションを、以下の 4 つから選択してください。

- FailOnStartup。 エラーをログに記録し、ただちにシャットダウンします。
- Reprocess。 残りのイベントを先に処理してから、入力キューのメッセージを処理します。
- Ignore。 進行中のキューのメッセージをすべて無視します。
- LogError。 エラーをログに記録しますが、シャットダウンしません。

デフォルト = Reprocess

InputQueue
コネクターが新規のメッセージの有無を確認するためにポーリングするメッセージ・キューです。コネクターは、セミコロンで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 つのキューに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 とします。

InputQueue プロパティーが指定されていない場合、コネクターは正常に始動して警告メッセージを印刷し、要求処理のみを実行します。この場合はイベント処理は実行しません。

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て、各キューから最大 pollQuantity 個のメッセージを検索します。例えば、*pollQuantity* が 2 であり、MyQueueA に 2 件のメッセージがあり、MyQueueB に 1 件のメッセージがあり、MyQueueC に 5 件のメッセージがある場合は、コネクターは以下のようにメッセージを取得します。

PollQuantity の値は 2 のため、コネクターは pollForEvents への呼び出しを行うたびに各キューから多くても 2 つのメッセージを検索します。最初のサイクル (2 回のうちの 1 回目) では、コネクターは、MyQueueA、MyQueueB、および MyQueueC の各キューの 1 番目のメッセージを検索します。これで 1 回目のポーリングは完了し、PollQuantity の値を 1 に設定している場合、コネクターは停止します。ここでは PollQuantity を 2 に設定しているため、コネクターは 2 回目のポーリング (2 回のうちの 2 回目) を開始し、MyQueueA と MyQueueC からそれぞれ 1 つずつメッセージを検索します。MqQueueB は空になっているのでスキップされます。すべてのキューを 2 回ずつポーリングしたら、メソッド pollForEvents への呼び出しは完了します。以下に、メッセージ検索の順序を示します。

1. MyQueueA から 1 件のメッセージ
2. MyQueueB から 1 件のメッセージ
3. MyQueueC から 1 件のメッセージ
4. MyQueueA から 1 件のメッセージ
5. 空になったため、MyQueueB をスキップ
6. MyQueueC から 1 件のメッセージ

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

InProgressQueue

処理中にメッセージが保留されるメッセージ・キューです。 System Manager を使用してデフォルトの InProgressQueue 名をコネクター固有のプロパティーから除去することにより、このキューなしで動作するようにコネクターを構成できます。このようにすると、始動時にイベントが保留されているときにコネクターをシャットダウンするとイベント・デリバリーで問題が発生する場合があることを示す警告メッセージが出されます。

デフォルト = queue://crossworlds.queue.manager/MQCONN.IN_PROGRESS

PollQuantity

pollForEvents スキャン中に、InputQueue プロパティーで指定した各キューから取得するメッセージの数です。

デフォルト =1

Port

WebSphere MQ リスナー用に設定されるポート。

デフォルト = なし。

ReplyToQueue

コネクターからの要求発行時に応答メッセージが配信されるキューです。子動的メタオブジェクトの属性を使用して応答を無視することもできます。これらの属性の詳細については、41 ページの『JMS ヘッダー、WebSphere MQ メッセージ・プロパティー、および動的子メタオブジェクトの属性』を参照してください。

デフォルト = queue://crossworlds.queue.manager/MQCONN.REPLYTO

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/MQCONN.UNSUBSCRIBED

注: *WebSphere MQ が提供する値が誤っていたり、不明である可能性があるので、必ずこれらの値をチェックしてください。値が誤っていたり不明の場合は、値を暗黙的に指定してください。

UseDefaults
Create 操作の場合、UseDefaults を true に設定すると、コネクターは、各 isRequired ビジネス・オブジェクト属性に有効値またはデフォルト値が指定されているかどうかをチェックします。値が指定されている場合、Create 操作は成功します。このパラメーターを false に設定すると、コネクターは有効値の有無だけをチェックし、有効値が指定されていない場合、Create 操作は失敗します。デフォルトは false です。

キューの Uniform Resource Identifier (URI)
キューの URI は、シーケンス queue:// で始まり、それに以下の項目が続きます。

- 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
- 残りのキュー・プロパティーを設定する、名前と値のペアのリスト (オプション)

例えば、次の URI を指定すると、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
```

次の表に、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	ミリ秒で表した、メッセージの存続時間	0 = 無制限。正整数 = タイムアウト (ミリ秒)。
priority	メッセージの優先順位	0-9。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、コネクターが独自のデフォルト値を使用できることを示します。
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、コネクターが独自のデフォルト値を使用できることを示します。

プロパティー名	説明	値
CCSID	アウトバウンド・メッセージをエンコードする文字セット	整数 - 有効値は、WebSphere MQ の基本資料にリストされています。この値は、コネクター固有構成プロパティーの CCSID の値と一致する必要があります。 25 ページの『CCSID』を参照してください。
targetClient	受信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	WebSphere MQ の基本資料に記載されている整数値。

注: アダプターは、MQMessages 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、あるいはメッセージ・バッファーに送達されるときに行われるため、コネクターはデータ変換を、IBM WebSphere MQ にインプリメントされている JMS に依存します (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、ネイティブの 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 がサポートする、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達するには、出力キューが完全修飾の URI で、CCSID と 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 を送達するためにデータをエンコードするときに、この情報を使用します (JMS API を介して)。多くの場合、CCSID およびエンコードのサポートの欠如は、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で、解決できます。CCSID およびエンコードに固有の問題が解決されない場合は、WebSphere Business Integration システム・テクニカル・サポートに連絡して、代替の Java 仮想マシンを使用したコネクターの実行について相談してください。

メタオブジェクト属性の構成

Connector for ACORD XML は、2 種類のメタオブジェクトを認識および読み取ることができます。

- 静的なコネクター・メタオブジェクト
- 動的な子メタオブジェクト

動的な子メタオブジェクトの属性値は、静的なメタオブジェクトの属性値と重複し、それらをオーバーライドします。

静的メタオブジェクト

ACORD XML の静的メタオブジェクトは、ビジネス・オブジェクトごとに定義された変換プロパティーのリストで構成されています。ビジネス・オブジェクトの変換プロパティーを定義するには、ストリング属性を作成し、構文 busObj_verb を使用してそれを命名します。例えば、動詞 Create を含む Customer オブジェクトの変換プロパティーを定義するには、Customer_Create という名前の属性を作成します。属性のアプリケーション固有テキストには、実際の変換プロパティーを指定します。

注: 静的な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中にある特定のメッセージ・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。この場合、コネクターはビジネス・オブジェクトを指定せずに、メッセージ・テキストを構成済みのデータ・ハンドラーに渡します。データ・ハンドラーがテキストのみに基づいたビジネス・オブジェクトを作成できない場合、コネクターはこのメッセージ・フォーマットが認識されていないことを表すエラーを報告します。

次の表に、メタオブジェクトのプロパティーを示します。

プロパティー名	説明
CorrelationID	このプロパティーは、要求処理中のアダプターの動作にのみ影響し、動的メタオブジェクトの CorrelationID プロパティーと同じように処理されます。詳しくは、45 ページの『非同期要求の処理』を参照してください。
CollaborationName	CollaborationName は、ビジネス・オブジェクトと動詞の組み合わせに対する属性の、アプリケーション固有テキスト内で指定される必要があります。例えば、ユーザーが動詞 Create 付きのビジネス・オブジェクト Customer の同期要求を処理しようとしている場合、静的メタデータ・オブジェクトは Customer_Create という名前の属性を含んでいる必要があります。 Customer_Create 属性には、名前と値のペアを含む、アプリケーション固有テキストが入っていなければなりません。例えば、CollaborationName=MyCustomerProcessingCollab です。構文の詳細については、34 ページの『アプリケーション固有の情報』の節を参照してください。 この条件が満たされていない場合は、コネクターが Customer ビジネス・オブジェクトに関する要求を同期処理しようとすると、ランタイム・エラーが発生します。 **注:** このプロパティーは、同期要求にのみ有効です。

プロパティー名	説明
DataEncoding	DataEncoding は、メッセージの読み書きに使用されるエンコードです。このプロパティーが静的メタオブジェクトに指定されていない場合、コネクターは特定のエンコードを使用せずに、メッセージを読もうとします。動的な子メタオブジェクトに定義された DataEncoding は、静的なメタオブジェクトに定義された値をオーバーライドします。デフォルト値は Text です。この属性の値のフォーマットは、messageType[:*enc*] です。つまり、Text:ISO8859_1 、Text:UnicodeLittle、Text、または Binary のようになります。このプロパティーは、内部的に InputFormat プロパティーと関連しています。InputFormat プロパティーごとに DataEncoding を 1 つだけ指定します。
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。静的なメタオブジェクトに指定された場合、この値は DataHandlerConfigMO コネクター・プロパティーに指定された値をオーバーライドします。さまざまなビジネス・オブジェクト・タイプを処理するために各種のデータ・ハンドラーが必要な場合は、この静的なメタオブジェクトのプロパティーを使用します。動的な子メタオブジェクトに指定された場合、このプロパティーはコネクター・プロパティーと静的なメタオブジェクトのプロパティーをオーバーライドします。データ形式が実際のビジネス・データに依存する可能性がある場合は、要求処理には動的な子メタオブジェクトを使用します。指定されたビジネス・オブジェクトはコネクターによりサポートされていることが必要です。
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。静的なメタオブジェクトに指定された場合、この値は DataHandlerMimeType コネクター・プロパティーに指定された値をオーバーライドします。さまざまなビジネス・オブジェクト・タイプを処理するために各種のデータ・ハンドラーが必要な場合は、この静的なメタオブジェクトのプロパティーを使用します。動的な子メタオブジェクトに指定された場合、このプロパティーはコネクター・プロパティーと静的なメタオブジェクトのプロパティーをオーバーライドします。データ形式が実際のビジネス・データに依存する可能性がある場合は、要求処理には動的な子メタオブジェクトを使用します。DataHandlerConfigMO に指定されたビジネス・オブジェクトは、このプロパティーの値に対応する属性を含める必要があります。

プロパティー名	説明
DoNotReportBusObj	オプションで、ユーザーは DoNotReportBusObj プロパティーを含めることができます。このプロパティーを true に設定すると、発行されるすべての PAN レポート・メッセージのメッセージ本体がブランクになります。要求側が要求が正常に処理されたことは確認したいが、ビジネス・オブジェクトへの変更に関する通知は必要としない場合に、このように設定することをお勧めします。このプロパティーは、NAN レポートには影響しません。 静的メタオブジェクトでこのプロパティーが見つからない場合、コネクターはこれを false にデフォルト設定し、メッセージ・レポートにビジネス・オブジェクトを取り込みます。 **注:** このプロパティーは、同期要求にのみ有効です。
InputFormat	InputFormat は、特定のビジネス・オブジェクトと関連付けるメッセージ・フォーマットです。検索されたメッセージがこのフォーマットである場合、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。このプロパティーは、内部的に DataEncoding プロパティーと関連しています。InputFormat プロパティーごとに DataEncoding を 1 つだけ指定します。 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。
OutputFormat	OutputFormat は、指定されたビジネス・オブジェクトから作成されたメッセージで設定されます。OutputFormat が指定されていない場合、使用可能であれば入力フォーマットが使用されます。動的な子メタオブジェクトに定義された OutputFormat は、静的なメタオブジェクトに定義された値をオーバーライドします。
InputQueue	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。ACORD XML アダプターについては、この機能は必要ない可能性があります。
OutputQueue	OutputQueue は、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配信される出力キューです。動的な子メタオブジェクトに定義された OutputQueue は、静的なメタオブジェクトに定義された値をオーバーライドします。
ResponseTimeout	応答を待機した状態で、タイムアウトになるまでの時間をミリ秒で表します。このプロパティーが定義されていないかまたはゼロより小さい値が設定されている場合、コネクターは応答を待機せずにすぐに SUCCESS を戻します。動的な子メタオブジェクトに定義された ResponseTimeout は、静的なメタオブジェクトに定義された値をオーバーライドします。

プロパティー名	説明
TimeoutFatal	このプロパティーが定義されていて、値 True を含む場合、ResponseTimeout に指定された時間内に応答を受信しなければ、コネクターは APP_RESPONSE_TIMEOUT を戻します。応答メッセージを待機中のその他すべてのスレッドは、InterChange Server にすぐに APP_RESPONSE_TIMEOUT を戻します。これにより、InterChange Server はコネクターを終了します。動的な子メタオブジェクトに定義された TimeoutFatal は、静的なメタオブジェクトに定義された値をオーバーライドします。

さらに、Default という名前の予約済みプロパティー名を、メタオブジェクトに定義することもできます。このプロパティーが存在する場合、そのアプリケーション固有情報によって、ビジネス・オブジェクトのすべての変換プロパティーに対しデフォルト値が指定されます。

以下に、メタオブジェクトのサンプルを示します。

プロパティー名	アプリケーション固有のテキスト
デフォルト	DataEncoding=Text:UnicodeLittle; OutputFormat=CUST_OUT; OutputQueue=QueueA;ResponseTimeout=10000; TimeoutFatal=False

アプリケーション固有の情報

アプリケーション固有の情報は、名前と値のペアで構成され、それらはセミコロンで区切られています。以下に例を示します。

```
InputFormat=CUST_IN;OutputFormat=CUST_OUT
```

データ・ハンドラーの InputQueues へのマッピング

静的メタオブジェクトのアプリケーション固有情報で InputQueue プロパティーを使用することにより、データ・ハンドラーと入力キューを関連付けることができます。この機能は、異なる書式や変換要件を持つ複数の取引先と取り引きする場合に役立ちます。それには、以下の作業を行う必要があります。

1. コネクター固有プロパティー (27 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する。
2. それぞれの入力キューごとに、キュー・マネージャーおよび入力キュー名を指定し、またアプリケーション固有情報にデータ・ハンドラーのクラス名および MIME タイプを指定する。

例えば、次に示す静的メタオブジェクトの属性は、データ・ハンドラーと、CompReceipts という名前の InputQueue を関連付けています。

```
[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.MQ.disposition_notification;DataHandlerMimeType=
message/
disposition_notification
IsRequiredServerBound = false
[End]
```

入力フォーマットの多重定義

コネクターは通常、メッセージ検索時に入力フォーマットを特定のビジネス・オブジェクトと動詞の組み合わせと付き合わせます。次に、コネクターはそのビジネス・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要求するビジネス・オブジェクトと対応しているかどうかを確認できます。

ただし、2 つ以上のビジネス・オブジェクトに同一の入力フォーマットが定義されている場合は、コネクターはデータ・ハンドラーにデータを渡す前にそのデータが表すビジネス・オブジェクトを判別することはできません。このような場合、コネクターはメッセージ内容のみをデータ・ハンドラーに渡してから、生成されるビジネス・オブジェクトに基づいた変換プロパティーを検索します。したがって、データ・ハンドラーはメッセージ内容のみに基づいてビジネス・オブジェクトを判別する必要があります。

生成されるビジネス・オブジェクトの動詞が設定されていない場合、コネクターはなんらかの動詞を含む同じビジネス・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。変換プロパティーのセットが 1 つだけ検出された場合、コネクターは特定の動詞を割り当てます。複数の変換プロパティーが検出された場合、コネクターは動詞を区別できないため、メッセージは失敗します。

サンプル・メタオブジェクト

以下に示す静的なメタオブジェクトは、Create、Update、Delete、および Retrieve の各動詞を使用して Customer ビジネス・オブジェクトを変換するようにコネクターを構成します。属性 `Default` はメタオブジェクトで定義されます。コネクターは以下の属性を持つ変換プロパティーを使用します。

```
OutputQueue=CustomerQueue1;ResponseTimeout=5000;TimeoutFatal=true
```

この属性は、その他すべての変換プロパティーのデフォルト値として使用されます。したがって、ある属性によって別の指定をされたり動的な子メタオブジェクト値によってオーバーライドされる場合を除いて、コネクターはすべてのビジネス・オブジェクトをキュー `CustomerQueue1` に発行し、その後応答メッセージを待機します。5000 ミリ秒内に応答が到着しない場合、コネクター はすぐに終了します。

動詞 Create 付き Customer オブジェクト: 属性 `Customer_Create` は、フォーマット `NEW` のメッセージはすべて、動詞 Create を含む Customer ビジネス・オブジェクトに変換する必要があることをコネクターに示します。出力フォーマットは定義されていないため、コネクターは入力用に定義されたフォーマット (この場合は `NEW`) を使用して、このオブジェクトと動詞の組み合わせを表すメッセージを送信します。

動詞 Update および Delete 付き Customer オブジェクト: 入力フォーマット `MODIFY` は多重定義されます。すなわち、動詞 Update を含む Customer ビジネス・オブジェクトと動詞 Delete を含む Customer ビジネス・オブジェクトの両方に定義

されます。このフォーマットを持つ検索されたメッセージを正常に処理するには、データ・ハンドラーが識別できるように、ビジネス・オブジェクト名と (該当する場合) 動詞をメッセージ内容に含める必要があります (35 ページの『入力フォーマットの多重定義』を参照してください)。要求処理操作では、出力フォーマットは定義されていないため、コネクターはどちらの動詞のメッセージも入力フォーマット MODIFY を使用して送信します。

動詞 Retrieve 付き Customer オブジェクト: 属性 Customer_Retrieve は、動詞 Retrieve を含むタイプ Customer のビジネス・オブジェクトが、フォーマット Retrieve を持つメッセージとして送信される必要があることを示します。デフォルトの応答時間は、コネクターがタイムアウトまでに最大 10000 ミリ秒待機するようにオーバーライドされているので注意してください (応答が受信されない場合も終了します)。

```
[ReposCopy]
Version = 3.1.0
Repositories = 1cHyILNuPTc=
[End]
[BusinessObjectDefinition]
Name = Sample_MO
Version = 1.0.0

[Attribute]
Name = Default
Type = String
Cardinality = 1
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = false
AppSpecificInfo = OutputQueue=CustomerQueue1;ResponseTimeout=5000;TimeoutFatal=true
IsRequiredServerBound = false
[End]
[Attribute]
Name = Customer_Create
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo = InputFormat=NEW
IsRequiredServerBound = false
[End]
[Attribute]
Name = Customer_Update
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo = InputFormat=MODIFY
IsRequiredServerBound = false
[End]
[Attribute]
Name = Customer_Delete
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
```

```
AppSpecificInfo = InputFormat=MODIFY
IsRequiredServerBound = false
[End]
[Attribute]
Name = Customer_Retrieve
Type = String
Cardinality = 1
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
AppSpecificInfo = OutputFormat=RETRIEVE;ResponseTimeout=10000
IsRequiredServerBound = false
[End]
[Attribute]
Name = ObjectEventId
Type = String
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]

[Verb]
Name = Create
[End]

[Verb]
Name = Delete
[End]

[Verb]
Name = Retrieve
[End]

[Verb]
Name = Update
[End]
[End]
```

動的な子メタオブジェクト

静的メタオブジェクトに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、オプションで、コネクターが実行時にビジネス・オブジェクト・インスタンスごとに指定されたメタデータを受け入れることができます。

コネクターは、コネクターに渡されるトップレベル・ビジネス・オブジェクトに子として追加される動的なメタオブジェクトから、変換プロパティーを認識し、読み取ります。この動的な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なメタオブジェクトに指定可能であった変換プロパティーと重複します。

動的な子メタオブジェクトのプロパティーは静的なメタオブジェクトから検出されるプロパティーをオーバーライドするため、動的な子メタオブジェクトを指定する場合は、静的なメタオブジェクトを指定するコネクター・プロパティーを組み込む必要はありません。したがって、動的な子メタオブジェクトは、静的なメタオブジェクトとは無関係に使用することができ、その逆もまた同様です。

注: コネクターは、同期イベント・デリバリーの、動的な子メタオブジェクトを使用したコラボレーション名の提供をサポートしていません。

ビジネス・オブジェクト Customer_Create の静的メタオブジェクトのサンプルは前セクションの表に示しました。次の表には、動的な子メタオブジェクトのサンプルを示します。アプリケーション固有の情報は名前と値のペアで構成され、それぞれのペアはセミコロンで区切られています。

プロパティー名	値
DataEncoding	Text:UnicodeLittle
DataHandlerMimeType*	text/delimited
OutputFormat	CUST_OUT
OutputQueue	QueueA
ResponseTimeout	10000
TimeoutFatal	False

*DataHandlerConfigMO は、コネクター構成プロパティーか静的なメタオブジェクトのいずれかに指定されていると想定します。

コネクターは、受信されたトップレベル・ビジネス・オブジェクトのアプリケーション固有の情報を調べて、タグ cw_mo_conn が子メタオブジェクトを指定しているかどうかを判断します。子メタオブジェクトが指定されている場合、動的な子メタオブジェクトの値が静的なメタオブジェクトに指定された値をオーバーライドします。

ポーリング時の動的子メタオブジェクトの取り込み

ポーリング中に検索されたメッセージについてさらに詳しい情報をコラボレーションに提供するため、コネクターは、作成されたビジネス・オブジェクトに動的なメタオブジェクトが定義済みである場合、その特定の属性に値を取り込みます。

次の表に、動的な子メタオブジェクトをポーリング用に構成する例を示します。

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

上の表に示すように、動的な子メタオブジェクトで、追加の属性 InputQueue を定義できます。この属性には特定のメッセージが検索されるキューの名前が含まれます。子メタオブジェクト内にこのプロパティーが定義されていない場合、これらには値が取り込まれません。

シナリオ例:

- コネクターは、キュー MyInputQueue からフォーマット CUST_IN でメッセージを取得します。
- 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調べてメタオブジェクトが定義されているかどうかを判断します。

- メタオブジェクトが定義されている場合、コネクターはこのメタオブジェクトのインスタンスを作成し、定義に基づいて InputQueue および InputFormat 属性に値を取り込んで、ビジネス・オブジェクトを使用可能なコラボレーションにパブリッシュします。

動的子メタオブジェクトのサンプル

```
[BusinessObjectDefinition]
Name = MO_Sample_Config
Version = 1.0.0

[Attribute]
Name = OutputFormat
Type = String
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = false
DefaultValue = CUST
IsRequiredServerBound = false
[End]
[Attribute]
Name = OutputQueue
Type = String
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
DefaultValue = OUT
IsRequiredServerBound = false
[End]
[Attribute]
Name = ResponseTimeout
Type = String
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
DefaultValue = -1
IsRequiredServerBound = false
[End]
[Attribute]
Name = TimeoutFatal
Type = String
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
DefaultValue = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = InputFormat
Type = String
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = InputQueue
Type = String
MaxLength = 1
IsKey = false
IsForeignKey = false
```

```
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = ObjectEventId
Type = String
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]

[Verb]
Name = Create
[End]

[Verb]
Name = Delete
[End]

[Verb]
Name = Retrieve
[End]

[Verb]
Name = Update
[End]
[End]
[BusinessObjectDefinition]
Name = Customer
Version = 1.0.0
AppSpecificInfo = cw_mo_conn=MyConfig

[Attribute]
Name = FirstName
Type = String
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = LastName
Type = String
MaxLength = 1
IsKey = true
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = Telephone
Type = String
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = MyConfig
Type = MO_Sample_Config
ContainedObjectVersion = 1.0.0
Relationship = Containment
Cardinality = 1
```

```
MaxLength = 1
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Attribute]
Name = ObjectEventId
Type = String
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]

[Verb]
Name = Create
[End]

[Verb]
Name = Delete
[End]

[Verb]
Name = Retrieve
[End]

[Verb]
Name = Update
[End]
[End]
```

JMS ヘッダー、WebSphere MQ メッセージ・プロパティー、および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・トランスポートの詳細情報を取得したりメッセージ・トランスポートを詳細に制御したりすることができます。このような属性を追加すると、JMS プロパティーを変更し、(アダプター・プロパティーで指定されたデフォルト ReplyToQueue を使用せずに) 要求ごとに ReplyToQueue を制御したり、メッセージの CorrelationID を再ターゲットしたりすることができます。このセクションでは、これらの属性、および同期モードと非同期モードの両方におけるイベント通知と要求処理に対する影響について説明します。

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 1. 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

表 1. 動的メタオブジェクト・ヘッダー属性 (続き)

ヘッダー属性名	モード	対応する JMS ヘッダー
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。

以下のセクションでは、これらの属性の解釈および使用について説明します。

注: 上記の属性はいずれも必須ではありません。ビジネス・プロセスに関連する動的メタオブジェクトには任意の属性を追加できます。

JMS プロパティー: 動的メタオブジェクトの他の属性と異なり、JMSProperties は単一カーディナリティー子オブジェクトを定義する必要があります。この子オブジェクトの各属性は、以下のように JMS メッセージ・ヘッダーの可変部分で読み取り/書き込みを行う単一プロパティーを定義する必要があります。

1. 属性の名前はセマンティック値を持ちません。
2. 属性のタイプは、JMS プロパティー・タイプに無関係に必ず String でなければなりません。
3. 属性のアプリケーション固有情報は、属性をマップする JMS メッセージ・プロパティーの名前と形式を定義する 2 つの名前と値の組を含まなければなりません。

以下の表に、JMSProperties オブジェクトの属性に対して定義する必要があるアプリケーション固有情報プロパティーを示します。

表 2. JMS プロパティー属性のアプリケーション固有情報

名前	指定可能な値	コメント
名前	任意の有効な JMS プロパティー名	これは JMS プロパティーの名前です。ベンダーによっては、拡張機能を提供するために特定のプロパティーを予約している場合があります。一般に、ユーザーはベンダー固有の機能にアクセスする場合以外は、JMS で開始するカスタム・プロパティーを定義してはなりません。
Type	String、Int、Boolean、Float、Double、Long、Short	これは JMS プロパティーのタイプです。JMS API は、JMS メッセージに値を設定するための多くのメソッドを提供します (例: setIntProperty、setLongProperty、setStringProperty)。ここで指定する JMS プロパティーのタイプによって、どのメソッドを使用してメッセージのプロパティー値を設定するかが決まります。

以下の図に、動的メタオブジェクトの属性 JMSProperties および 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。属性のアプリケーション固有情報はそれぞれの名前およびタイプを定義します。例えば、属性 ID はタイプ String の JMS プロパティー ID にマップされます。

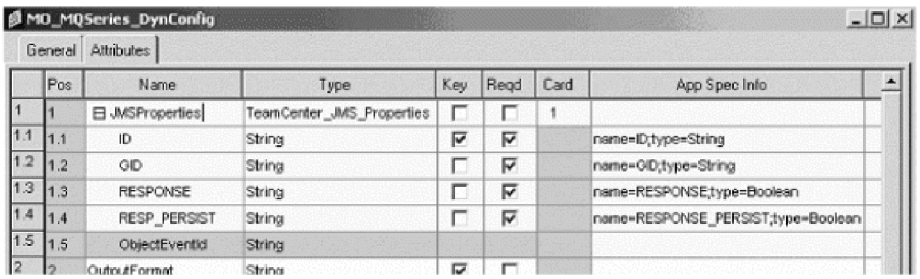

図 3. 動的メタオブジェクトの JMS プロパティー属性

非同期イベント通知: ヘッダー属性を持つ動的メタオブジェクトがイベント・ビジネス・オブジェクトに存在する場合は、コネクターは、(メタオブジェクトにトランスポート関連のデータを設定するほかに) 以下のステップを実行します。

1. メタオブジェクトの CorrelationId 属性に、メッセージの JMSCorrelationID ヘッダー・フィールドで指定された値を設定します。
2. メタオブジェクトの ReplyToQueue 属性に、メッセージの JMSReplyTo ヘッダー・フィールドで指定されたキューを設定します。このヘッダー・フィールド

はメッセージの Java オブジェクトによって表されるため、この属性にはキューの名前 (多くの場合は URI) が設定されます。

3. メタオブジェクトの DeliveryMode 属性に、メッセージの JMSDeliveryMode ヘッダー・フィールドで指定された値を設定します。
4. メタオブジェクトの Priority 属性に、メッセージの JMSPriority ヘッダー・フィールドを設定します。
5. メタオブジェクトの Destination 属性に、メッセージの JMSDestination ヘッダー・フィールドの名前を設定します。Destination はオブジェクトによって表されるため、この属性には Destination オブジェクトの名前が設定されます。
6. メタオブジェクトの Expiration 属性に、メッセージの JMSExpiration ヘッダー・フィールドの値を設定します。
7. メタオブジェクトの MessageID 属性に、メッセージの JMSMessageID ヘッダー・フィールドの値を設定します。
8. メタオブジェクトの Redelivered 属性に、メッセージの JMSRedelivered ヘッダー・フィールドの値を設定します。
9. メタオブジェクトの TimeStamp 属性に、メッセージの JMSTimeStamp ヘッダー・フィールドの値を設定します。
10. メタオブジェクトの Type 属性に、メッセージの JMSType ヘッダー・フィールドの値を設定します。
11. メタオブジェクトの UserID 属性に、メッセージの JMSXUserID プロパティー・フィールドの値を設定します。
12. メタオブジェクトの AppID 属性に、メッセージの JMSXAppID プロパティー・フィールドの値を設定します。
13. メタオブジェクトの DeliveryCount 属性に、メッセージの JMSXDeliveryCount プロパティー・フィールドの値を設定します。
14. メタオブジェクトの GroupID 属性に、メッセージの JMSXGroupID プロパティー・フィールドの値を設定します。
15. メタオブジェクトの GroupSeq 属性に、メッセージの JMSXGroupSeq プロパティー・フィールドの値を設定します。
16. メタオブジェクトの JMSProperties 属性に定義されたオブジェクトを検証します。アダプターは、メッセージの対応するプロパティーの値をこのオブジェクトの各属性に設定します。特定のプロパティーがメッセージで定義されていない場合は、アダプターはその属性の値を CxBlank に設定します。

同期イベント通知: 同期イベント処理の場合は、アダプターはイベントを通知し、統合ブローカーからの応答を待った後、アプリケーションに応答メッセージを送信します。ビジネス・データに対する変更は、戻される応答メッセージに反映されます。イベントを通知する前に、アダプターは、非同期イベント通知の場合と同様に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に設定される値は、以下のように応答発行ヘッダーに反映されます (動的メタオブジェクトの他の読み取り専用属性は無視されます)。

- **CorrelationID** 動的メタオブジェクトが属性 CorrelationId を含む場合は、発信元アプリケーションが必要とする値に設定する必要があります。アプリケーションは、CorrelationID を使用してコネクターから戻されたメッセージと元の要求

を突き合わせます。CorrelationID が予期しない値または無効値の場合は、問題が発生します。これは、この属性を使用する前にアプリケーションが相関する要求および応答メッセージを処理する方法を判別するのに役立ちます。同期要求で CorrelationID を設定するには 4 つの方法があります。

1. 値を変更しない。応答メッセージの CorrelationID は、要求メッセージの CorrelationID と同じになります。これは、WebSphere MQ オプション MQRO_PASS_CORREL_ID と同等です。
2. 値を CxIgnore に変更する。コネクターは、デフォルトで要求のメッセージ ID を応答の CorrelationID にコピーします。これは、WebSphere MQ オプション MQRO_COPY_MSG_ID_TO_CORREL_ID と同等です。
3. 値を CxBlank に変更する。コネクターは応答メッセージの CorrelationID を設定しません。
4. 値をカスタム値に変更する。これを行うには、応答を処理するアプリケーションでカスタム値を認識する必要があります。

メタオブジェクトで属性 CorrelationID を定義しない場合は、コネクターは自動的に CorrelationID を処理します。

- **ReplyToQueue** 属性 ReplyToQueue に別のキューを指定することによって動的メタオブジェクトを更新する場合は、コネクターは、応答メッセージを指定のキューに送信します。これはお勧めしません。コネクターに対して応答メッセージを別のキューに送信させると、応答メッセージで特定の応答キューを設定するアプリケーションはそのキューで応答を待つと想定されるため、通信に干渉する場合があります。
- **JMS プロパティー** 更新されたビジネス・オブジェクトがコネクターに戻されるときに動的メタオブジェクトの JMS プロパティー属性に設定された値が応答メッセージに設定されます。

非同期要求の処理: コネクターは、動的メタオブジェクト (存在する場合) を使用して要求メッセージを設定してから発行します。コネクターは、以下のステップを実行してから要求メッセージを送信します。

1. 属性 CorrelationID が動的メタオブジェクトに存在する場合は、コネクターは、アウトバウンド要求メッセージの CorrelationID をこの値に設定します。
2. 属性 ReplyToQueue が動的メタオブジェクトで指定されている場合は、コネクターは、応答メッセージでこのキューを渡し、このキューで応答を待ちます。これにより、コネクター構成プロパティーで指定されている ReplyToQueue 値をオーバーライドできます。さらに負の ResponseTimeout (コネクターが応答を待たないことを示す) を指定した場合は、コネクターは実際には応答を待ちませんが、応答メッセージで ReplyToQueue が設定されます。
3. 属性 DeliveryMode を 2 に設定すると、メッセージは永続的に送信されます。DeliveryMode を 1 に設定すると、メッセージは永続的に送信されません。その他の値を設定すると、コネクターに障害が発生します。MO にDeliveryMode を指定しないと、JMS プロバイダーが永続設定を確立します。
4. 属性 Priority を指定すると、コネクターが発信要求に値を設定します。Priority 属性には 0 から 9 までの値を設定できます。その他の値を指定すると、コネクターが終了します。

5. 動的メタオブジェクトで属性 JMSProperties を指定した場合は、コネクターによって送信されるアウトバウンド・メッセージに、子動的メタオブジェクトで指定された対応する JMS プロパティーが設定されます。

注: 動的メタオブジェクトのヘッダー属性が定義されていない場合または CxIgnore を指定した場合は、コネクターはデフォルト設定に従います。

同期要求の処理: コネクターは、動的メタオブジェクト (存在する場合) を使用して要求メッセージを設定してから発行します。動的メタオブジェクトがヘッダー属性を含む場合は、コネクターは、応答メッセージで検出された対応する新しい値をそのヘッダー属性に設定します。コネクターは、応答メッセージを受信した後、(メタオブジェクトにトランスポート関連のデータを設定するほかに) 以下のステップを実行します。

1. 属性 CorrelationID が動的メタオブジェクトに存在する場合は、アダプターは、応答メッセージで指定された JMSCorrelationID でこの属性を更新します。
2. 属性 ReplyToQueue が動的メタオブジェクトで定義されている場合は、アダプターは、応答メッセージで指定された JMSReplyTo の名前でこの属性を更新します。
3. 属性 DeliveryMode が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSDeliveryMode ヘッダー・フィールドの値でこの属性を更新します。
4. 属性 Priority が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSPriority ヘッダー・フィールドの値でこの属性を更新します。
5. 属性 Destination が動的メタオブジェクトで定義されている場合は、アダプターは、応答メッセージで指定された JMSDestination の名前でこの属性を更新します。
6. 属性 Expiration が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSExpiration ヘッダー・フィールドの値でこの属性を更新します。
7. 属性 MessageID が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSMessageID ヘッダー・フィールドの値でこの属性を更新します。
8. 属性 Redelivered が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSRedelivered ヘッダー・フィールドの値でこの属性を更新します。
9. 属性 TimeStamp が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSTimeStamp ヘッダー・フィールドの値でこの属性を更新します。
10. 属性 Type が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSType ヘッダー・フィールドの値でこの属性を更新します。
11. 属性 UserID が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSXUserID ヘッダー・フィールドの値でこの属性を更新します。
12. 属性 AppID が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSXAppID プロパティー・フィールドの値でこの属性を更新します。

13. 属性 DeliveryCount が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSXDeliveryCount ヘッダー・フィールドの値でこの属性を更新します。
14. 属性 GroupID が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSXGroupID ヘッダー・フィールドの値でこの属性を更新します。
15. 属性 GroupSeq が動的メタオブジェクトに存在する場合は、アダプターは、メッセージの JMSXGroupSeq ヘッダー・フィールドの値でこの属性を更新します。
16. 属性 JMSProperties が動的メタオブジェクトで定義されている場合は、アダプターは、子オブジェクトで定義されているすべてのプロパティーを、応答メッセージで検出された値で更新します。子オブジェクトで定義されているプロパティーがメッセージに存在しない場合は、値は CxBlank に設定されます。

注: 動的メタオブジェクトを使用して要求メッセージで設定された CorrelationID を変更しても、アダプターが応答メッセージを識別する方法には影響しません。アダプターは、デフォルトですべての応答メッセージの CorrelationID がアダプターによって送信された要求のメッセージ ID に等しいことを要求します。

エラー処理: JMS プロパティーをメッセージから読み取れない場合、またはメッセージに書き込めない場合は、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。ユーザー指定の ReplyToQueue が存在しないかアクセスできない場合は、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は、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、ログに記録されたメッセージはコネクターのメッセージ・ファイルからのものです。

始動ファイルの構成

Connector for ACORD XML を始動する前に、始動ファイルを構成する必要があります。

Windows

Windows プラットフォームのコネクターの構成を完了するには、start_WebSphereMQ.bat ファイルを修正する必要があります。

1. start_WebSphereMQ.bat ファイルを開きます。
2. 「Set the directory containing your WebSphere MQ Java client libraries,」で始まるセクションまでスクロールし、WebSphere MQ Java クライアント・ライブラリーの場所を指定します。

UNIX

UNIX プラットフォームのコネクターの構成を完了するには、start_WebSphereMQ.sh ファイルを修正する必要があります。

1. start_WebSphereMQ.sh ファイルを開きます。
2. 「Set the directory containing your WebSphere MQ Java client libraries,」で始まるセクションまでスクロールし、WebSphere MQ Java クライアント・ライブラリーの場所を指定します。

始動

コネクターの始動と停止、およびコネクターの一時始動ログ・ファイルの詳細については、ご使用のプラットフォーム用の「*システム・インストール・ガイド*」の始動に関する章を参照してください。

第 3 章 ビジネス・オブジェクトの作成または変更

- 『アダプター・ビジネス・オブジェクトの構造』
- 51 ページの『エラー処理』
- 52 ページの『トレース』

コネクターとともに提供されるのは、サンプルのビジネス・オブジェクトのみです。システム・インテグレーター、コンサルタント、またはカスタマーは、ビジネス・オブジェクトをビルドする必要があります。

コネクターは、メタデータ主導型コネクターです。WebSphere Business Integration システムのビジネス・オブジェクトにおいては、メタデータはアプリケーションに関するデータであり、ビジネス・オブジェクト定義に格納され、コネクターとアプリケーションの間の通信を支援します。メタデータ主導型コネクターは、コネクターにハードコーディングされている命令ではなく、ビジネス・オブジェクト定義にエンコード されているメタデータに基づいて、サポートする各ビジネス・オブジェクトを処理します。

ビジネス・オブジェクトのメタデータには、ビジネス・オブジェクトの構造、その属性プロパティーの設定、およびアプリケーション固有情報の内容が含まれます。コネクターはメタデータ主導型のため、コネクターのコーディングを変更しなくても、新規ビジネス・オブジェクトや変更されたビジネス・オブジェクトを処理できます。ただし、コネクターの構成済みのデータ・ハンドラーは、コネクターのビジネス・オブジェクトの構造、オブジェクトのカーディナリティー、アプリケーション固有情報のフォーマット、およびビジネス・オブジェクトのデータベース表現についてさまざまな想定を行います。したがって、ACORD XML 用のビジネス・オブジェクトを作成または変更する場合は、コネクターがそれに従うように設計されている規則に準拠して変更を行う必要があります。そうしないと、コネクターは新規または変更されたビジネス・オブジェクトを適切に処理できません。

この章では、コネクターがビジネス・オブジェクトを処理する方法と、コネクターが行うさまざまな想定について説明します。これらの情報は、新規のビジネス・オブジェクトをインプリメントするための手引きとして役立ちます。

アダプター・ビジネス・オブジェクトの構造

アダプターのインストールが完了したら、ビジネス・オブジェクトを作成する必要があります。ビジネス・オブジェクトの構造については、構成済みのデータ・ハンドラーによって定められている以外の要件はありません。コネクターが処理するビジネス・オブジェクトには、InterChange Server によって許可されている任意の名前を命名できます。

アダプターはキューからメッセージを検索し、ビジネス・オブジェクト (メタオブジェクトによって定義されたもの) にメッセージの内容を取り込もうとします。厳密に言うと、コネクターはビジネス・オブジェクト構造を制御したり、それに影響を及ぼしたりはしません。そのような働きは、メタオブジェクト定義とコネクターのデータ・ハンドラー要件の機能です。実際に、ビジネス・オブジェクト・レベル

のアプリケーション情報は存在しません。むしろ、ビジネス・オブジェクトの検索や受け渡しの際のコネクターの主な役割は、ビジネス・オブジェクトへのメッセージ (逆もまた同様) の処理をモニターしてエラーの有無を確認することです。

ビジネス・オブジェクト・プロパティーのサンプル

このセクションでは、XML データ・ハンドラーを含む ACORD XML コネクターのサンプル・ビジネス・オブジェクト・プロパティーについて説明します。

```
[BusinessObjectDefinition]
Name = ACORD_TXLife_VendorName
Version = 1.0.0
AppSpecificInfo = target_ns=http://ACORD.org/Standards/Life/2;
  elem_fd=unqualified;attr_fd=unqualified
[Attribute]
Name = VendorCode
Type = String
Cardinality = 1
MaxLength = 255
IsKey = true
IsForeignKey = false
IsRequired = true
AppSpecificInfo = attr_name=VendorCode;type=attribute
IsRequiredServerBound = false
[End]
[Attribute]
Name = VendorName
Type = String
Cardinality = 1
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = true
AppSpecificInfo = elem_name=VendorName;type=pcdata;notag
IsRequiredServerBound = false
[End]
[Attribute]
Name = ObjectEventId
Type = String
Cardinality = 1
MaxLength = 255
IsKey = false
IsForeignKey = false
IsRequired = false
IsRequiredServerBound = false
[End]
[Verb]
Name = Create
[End]
[Verb]
Name = Retrieve
[End]
[Verb]
Name = Delete
[End]
[Verb]
Name = Update
[End]
[End]
```

エラー処理

コネクターが生成するすべてのエラー・メッセージは、WebSphere MQConnector.txt という名前のメッセージ・ファイルに保管されます (ファイル名は、LogFileName 標準コネクター構成プロパティーによって決定されます。) それぞれのエラー・メッセージの前にはエラー番号が付けられています。

```
Message number
Message text
```

コネクターは、以下の各セクションで説明するような特定のエラーを処理します。

アプリケーションのタイムアウト

以下のような場合には、エラー・メッセージ ABON_APPRESPONSETIMEOUT が戻されます。

- メッセージ検索中に、コネクターが JMS サービス・プロバイダーとの接続を確立できない。
- コネクターはビジネス・オブジェクトを正常にメッセージに変換したが、接続が切断されたためにメッセージを出力キューに配信できない。
- コネクターはメッセージを発行したが、変換プロパティー TimeoutFatal の値が True であるビジネス・オブジェクトの応答待ちがタイムアウトになった。
- コネクターが戻りコード APP_RESPONSE_TIMEOUT または UNABLE_TO_LOGIN を含む応答メッセージを受信した。

アンサブスクライブされたビジネス・オブジェクト

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するメッセージを検索する場合、コネクターは、UnsubscribedQueue プロパティーに指定されたキューにメッセージを配信します。

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

gotApplEvent() メソッドによって NO_SUBSCRIPTION_FOUND コードが戻された場合、コネクターはそのメッセージを UnsubscribedQueue プロパティーで指定されているキューに送信し、他のイベントの処理を続けます。

アクティブでないコネクター

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・ハンドラーの変換

データ・ハンドラーがメッセージのビジネス・オブジェクトへの変換に失敗した場合、または、ビジネス・オブジェクトに特有の (JMS プロバイダーとは反対の) 処理エラーが発生した場合、メッセージは ErrorQueue に指定されたキューに配信されます。ErrorQueue が定義されていない場合、エラーが原因で処理できないメッセージは廃棄されます。

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

トレース

トレースはオプションのデバッグ機能であり、この機能をオンにするとコネクターの動作を密着して追跡できます。トレース・メッセージは、デフォルトでは 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、21 ページの『第 2 章 アダプターのインストールと構成』 に記載されている『コネクター構成プロパティー』を参照してください。トレースの有効化および設定の方法など、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。

以下に、コネクターのトレース・メッセージの推奨レベルを示します。

レベル 0　このレベルは、コネクターのバージョンを示すトレース・メッセージに使用します。

レベル 1　このレベルは、処理される各ビジネス・オブジェクトに関するキー情報を提供したり、ポーリング・スレッドが入力キューに新規のメッセージを検出するたびに記録したりするトレース・メッセージに使用します。

レベル 2　このレベルは、ビジネス・オブジェクトが `gotApplEvent()` または `executeCollaboration()` のいずれかから InterChange Server に通知されるたびにログを記録するトレース・メッセージに使用します。

レベル 3　このレベルは、メッセージからビジネス・オブジェクトおよびその反対の変換についての情報を提供したり、メッセージの出力キューへのデリバリーについての情報を提供したりするトレース・メッセージに使用します。

レベル 4　このレベルは、コネクターがある関数を入力または出力する場合を示すトレース・メッセージに使用します。

レベル 5　このレベルは、コネクターの初期化の通知、アプリケーション内で実行されるステートメントの表現、メッセージがキューから出し入れされる際の通知、またはビジネス・オブジェクト・ダンプの記録をするトレース・メッセージに使用します。

第 4 章 トラブルシューティング

この章では、コネクターの始動時および実行時に発生する可能性がある問題について説明します。

始動時の問題

問題	可能性のある解決方法/説明
コネクターが初期設定中に不意にシャットダウンして、「Exception in thread "main" java.lang.NoClassDefFoundError: javax/jms/JMSException…」というメッセージがレポートされた。	コネクターが IBM 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 の変数 WebSphere MQ_JAVA_LIB が、IBM 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てください。
コネクターが初期設定中に不意にシャットダウンして、「Exception in thread "main" java.lang.NoClassDefFoundError: com/ibm/mq/jms/MQConnectionFactory...」というメッセージがレポートされた。	コネクターが IBM 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 の変数 WebSphere MQ_JAVA_LIB が、IBM 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てください。
コネクターが初期設定中に不意にシャットダウンして、「Exception in thread "main" java.lang.NoClassDefFoundError: javax/naming/Referenceable...」というメッセージがレポートされた。	コネクターが IBM 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 の変数 WebSphere MQ_JAVA_LIB が、IBM 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てください。
コネクターが初期設定中に不意にシャットダウンして、「java.lang.UnsatisfiedLinkError: no mqjbnd01 in shared library path 」という例外がレポートされた。	コネクターが IBM WebSphere MQ 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 [NT] または libmqjbnd01.so [Solaris]) を見つけることができません。パスに IBM WebSphere MQ Java クライアント・ライブラリー・フォルダーが含まれているか確認してください。
コネクターが、「MQJMS2005: failed to create MQQueueManager for ':'」とレポートした。	次のプロパティーに値を明示的に設定してください。HostName、Channel、Port

イベント処理

問題	可能性のある解決方法/説明
コネクターが、MQRFH2 ヘッダーの付いたすべてのメッセージを送達する。	MQMD WebSphere MQ ヘッダーのみの付いたメッセージを送達するには、?targetClient=1 を出力キュー URI の名前に追加します。例えば、メッセージをキュー queue://my.queue.manager/OUT に出力するには、URI を queue://my.queue.manager/OUT?targetClient=1 に変更します。詳しくは、21 ページの『第 2 章 アダプターのインストールと構成』を参照してください。

問題	可能性のある解決方法/説明
コネクターが、コネクター・メタオブジェクトでのフォーマットの定義に関係なく、すべてのメッセージ・フォーマットを送達時に 8 文字に切り捨てる。	これは、コネクターではなく、WebSphere MQ MQMD メッセージ・ヘッダーの制約です。

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 コネクター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 Broker (WMQI)
- WebSphere Application Server (WAS)

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

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

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

新規プロパティーと削除されたプロパティー

アダプターのリリース 2.3 で追加された標準プロパティーと削除された標準プロパティーは、以下の通りです。

新規プロパティー

- RFH2Message Domain
- ListenerConcurrency
- RestartCount
- WsifSynchronousRequestTimeout

削除されたプロパティー

なし

標準コネクター・プロパティーの構成

アダプター・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。

- 標準構成プロパティー
- コネクター固有のプロパティー

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

Connector Configurator の使用

Connector Configurator からコネクター・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、付録の『Connector Configurator』を参照してください。

注: Connector Configurator と System Manager は、Windows システム上でのみ動作します。コネクターを UNIX システム上で稼動している場合でも、これらのツールがインストールされた Windows マシンが必要です。UNIX 上で動作するコネクターのコネクター・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。

プロパティー値の設定と更新

プロパティー・フィールドのデフォルトの長さは 255 文字です。

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 (最も番号の大きい項目が他の項目よりも優先されます)。

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

コネクターは、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。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、変更が即時に有効になります。コネクターがスタンドアロン・モードで (System Manager から独立して) 稼動している場合、例えば WebSphere MQ Integrator Broker で稼動している場合などには、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。
- 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、変更が有効になりません。アプリケーション固有コンポーネントまたは統合ブローカーを停止、再始動する必要はありません。
- サーバー再始動
  アプリケーション固有のコンポーネントおよび統合ブローカーを停止して再始動しなければ、変更が有効になりません。
- エージェント再始動 (ICS のみ)
  アプリケーション固有のコンポーネントを停止して再始動しなければ、変更が有効になりません。

特定のプロパティ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、次に示すプロパティーの要約の表の「更新メソッド」列を参照してください。

標準プロパティーの要約

表 3 は、標準コネクター構成プロパティーの早見表です。コネクターによっては使用されないプロパティーがあります。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。

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

表 3. 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
AdminInQueue	*有効な JMS キュー名*	*CONNECTORNAME* /ADMININQUEUE		Delivery Transport は JMS
AdminOutQueue	*有効な JMS キュー名*	*CONNECTORNAME*/ADMINOUTQUEUE		Delivery Transport は JMS
AgentConnections	1 から 4	1	コンポーネント再始動	ICS: Delivery Transport は MQ または IDL
AgentTraceLevel	0 から 5	0	動的	
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	必須値
BrokerType	ICS、WMQI、WAS			
CharacterEncoding	ascii7、ascii8、SJIS、Cp949、GBK、Big5、Cp297、Cp273、Cp280、Cp284、Cp037、Cp437 **注:** これは、サポートされる値の一部です。	ascii7	コンポーネント再始動	
ConcurrentEventTriggeredFlows	1 から 32,767	値なし	コンポーネント再始動	
ContainerManagedEvents	値なし、または JMS	JMS		保証付きイベント・デリバリー
ControllerStoreAndForwardMode	true または false	True	動的	ICS のみ
ControllerTraceLevel	0 から 5	0	動的	ICS のみ
DeliveryQueue		*CONNECTORNAME*/DELIVERYQUEUE	コンポーネント再始動	JMS トランスポートのみ
DeliveryTransport	MQ、IDL、または JMS	JMS	コンポーネント再始動	WAS または WMQI: JMS のみ
DuplicateEventElimination	True/False	False	コンポーネント再始動	JMS トランスポートのみ、 Container Managed Events は <NONE> でなければならない
FaultQueue		*CONNECTORNAME*/FAULTQUEUE	コンポーネント再始動	
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory または CxCommon.Messaging.jms.SonicMQFactory または任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	サーバー再始動	JMS トランスポートのみ

表 3. 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
jms.MessageBrokerName	FactoryClassName が IBM の場合は crossworlds.queue.manager を使用。FactoryClassName が Sonic の場合 localhost:2506 を使用。	crossworlds.queue.manager	サーバー再始動	JMS トランスポートのみ
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	JMS トランスポートのみ
jms.Password	任意の有効なパスワード		サーバー再始動	JMS トランスポートのみ
jms.UserName	任意の有効な名前		サーバー再始動	JMS トランスポートのみ
JvmMaxHeapSize	ヒープ・サイズ (メガバイト単位)	128M	コンポーネント再始動	ICS のみ
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	ICS のみ
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	ICS のみ
ListenerConcurrency	1 から 100	1	コンポーネント再始動	ICS のみ: Delivery Transport は MQ でなければならない
Locale	en_US、ja_JP、ko_KR、zh_C、zh_T、fr_F、de_D、it_I、es_E、pt_BR **注:** これは、サポートされるロケールの一部です。	en_US	コンポーネント再始動	
LogAtInterchangeEnd	True または False	False	コンポーネント再始動	ICS のみ
MaxEventCapacity	1 から 2147483647	2147483647	動的	ICS: Repository Directory は <REMOTE> でなければならない
MessageFileName	パス/ファイル名	*Connectorname*.txt あるいは InterchangeSystem.txt	コンポーネント再始動	
MonitorQueue	任意の有効なキュー名	*CONNECTORNAME*/MONITORQUEUE	コンポーネント再始動	JMS トランスポートのみ: DuplicateEvent Elimination は True でなければならない
OADAutoRestartAgent	True または False	False	動的	ICS のみ: Repository Directory は <REMOTE> でなければならない

表 3. 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
OADMaxNumRetry	*正数*	1000	動的	ICS のみ: Repository Directory は <REMOTE> でなければならない
OADRetryTimeInterval	*正数 (単位: 分)*	10	動的	ICS のみ: Repository Directory は <REMOTE> でなければならない
PollEndTime	HH:MM	HH:MM	コンポーネント再始動	
PollFrequency	*正整数 (単位: ミリ秒)* no (ポーリングを使用不可にする) key (コネクターのコマンド・プロンプト・ウィンドウで文字 p が入力された場合にのみポーリングする)	10000	動的	
PollQuantity	1 から 500	1	コンポーネント再始動	JMS トランスポートのみ: DuplicateEvent Elimination は True でなければならない
PollStartTime	HH:MM *(HH は 0 から 23、MM は 0 から 59)*	HH:MM	コンポーネント再始動	
RepositoryDirectory	メタデータの場所 repository		コンポーネント再始動	ICS の場合は <REMOTE> に設定する。WMQI または WAS の場合は *<local directory>* に設定する。
RequestQueue	*有効な JMS キュー名*	*CONNECTORNAME*/REQUESTQUEUE	コンポーネント再始動	
ResponseQueue	*有効な JMS キュー名*	*CONNECTORNAME*/RESPONSEQUEUE	コンポーネント再始動	
RestartCount	0 から 100		動的	コネクターはポーリング・モードで稼動している必要がある
RestartRetryCount	0 から 99	3	動的	
RestartRetryInterval	*適切な正数 (単位: 分)*	1	動的	
RHF2MessageDomain	mrm, xml	mrm	コンポーネント再始動	Delivery Transport が JMS であり、かつ WireFormat が CwXML である

表 3. 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
SourceQueue	*有効な WebSphere MQ 名*	*CONNECTORNAME*/SOURCEQUEUE	コンポーネント再始動	Delivery Transport が JMS であり、かつ Container Managed Events が指定されている場合のみ
SynchronousRequestQueue		*CONNECTORNAME*/SYNCHRONOUSREQUESTQUEUE	コンポーネント再始動	
SynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	
SynchronousResponseQueue		*CONNECTORNAME*/SYNCHRONOUSRESPONSEQUEUE	コンポーネント再始動	
WireFormat	CwXML、CwBO	CwXML	コンポーネント再始動	WMQI および WAS の場合は CwXML。Repository Directory が <REMOTE> の場合は CwBO (ICS)。
WisfSynchronousRequest Timeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	WAS のみ

標準構成プロパティー

このセクションでは、各標準コネクター構成プロパティーの定義を示します。

AdminInQueue

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です。

デフォルト値は CONNECTORNAME/ADMININQUEUE です。

AdminOutQueue

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です。

デフォルト値は CONNECTORNAME/ADMINOUTQUEUE です。

AgentConnections

WebSphere ICS でのみ使用されます。

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

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

AgentTraceLevel

アプリケーション固有のコンポーネントのトレース・メッセージのレベルです。デフォルト値は 0 です。コネクターは、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。

ApplicationName

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

BrokerType

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

CharacterEncoding

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

注: Java ベースのコネクターでは、このプロパティーは使用しません。C++ ベースのコネクターでは、現在、このプロパティーに ASCII という値が使用されています。このプロパティーの値を ascii7 または ascii8 に設定している場合、ASCII またはサポートされる他のいずれかの値に設定して、コネクターを構成し直す必要があります。

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

デフォルト値は ascii です。

ConcurrentEventTriggeredFlows

WebSphere ICS でのみ使用されます。

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

このプロパティーを 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、次のようにする必要があります。

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

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。

ContainerManagedEvents

この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、単一 JMS トランザクションとして宛先キューに配置されます。

デフォルト値は JMS です。また、値なしに設定することもできます。

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

- PollQuantity = 1 から 500
- SourceQueue = SOURCEQUEUE

また、MimeType、DHClass、および DataHandlerConfigMOName (オプショナル) 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。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。「データ・ハンドラー」タブの値のフィールドは、ContainerManagedEvents を JMS に設定した場合にのみ表示されます。

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

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

ControllerStoreAndForwardMode

WebSphere ICS でのみ使用されます。

宛先側のアプリケーション固有のコンポーネントが使用不可であることをコネクター・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。

デフォルト値は true です。

ControllerTraceLevel

WebSphere ICS でのみ使用されます。

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です。デフォルト値は 0 です。

DeliveryQueue

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です。

デフォルト値は DELIVERYQUEUE です。

DeliveryTransport

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

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

DeliveryTransport プロパティーに指定されている値が、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。

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 キュー内に保持されます。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必要がありません。
- エージェント・サイド・パフォーマンス:
  WebSphere MQ を使用すると、アプリケーション固有のコンポーネント側のパフォーマンスが向上します。WebSphere MQ を使用すると、コネクターのポーリング・スレッドは、イベントを選出した後、コネクターのキューにそのイベントを入れ、次のイベントを選出します。この方法は IDL よりも高速で、IDL の場合、コネクターのポーリング・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

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

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

重要: 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 が統合ブローカーの場合

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 768M 未満である場合には、次のように設定することをお勧めします。

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

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

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

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

- IPCCBaseAddress プロパティーの値を 11 または 12 に設定する。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

このプロパティーを true に設定すると、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。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。これはコネクター開発時に設定されます。

このプロパティーは、false に設定することもできます。

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

FaultQueue

コネクタ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、そのメッセージを状況表示および問題説明とともにこのプロパティーに指定されているキューに移動します。

デフォルト値は CONNECTORNAME/FAULTQUEUE です。

JvmMaxHeapSize

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

デフォルト値は 128M です。

JvmMaxNativeStackSize

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

デフォルト値は 128K です。

JvmMinHeapSize

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

デフォルト値は 1M です。

jms.FactoryClassName

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

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

jms.MessageBrokerName

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

デフォルト値は crossworlds.queue.manager です。

jms.NumConcurrentRequests

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

デフォルト値は 10 です。

jms.Password

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

デフォルトはありません。

jms.UserName

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

デフォルトはありません。

ListenerConcurrency

このプロパティーは、統合ブローカーとして ICS を使用する場合の MQ Listener でのマルチスレッド化をサポートしています。このプロパティーにより、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。デフォルト値は 1 です。

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にのみ適用されます。DeliveryTransport プロパティーには MQ を設定してください。

Locale

言語コード、国または地域、および、希望する場合には、関連した文字コード・セットを指定します。このプロパティーの値は、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

ここで、以下のように説明されます。

ll	2 文字の言語コード (普通は小文字)
TT	2 文字の国または地域コード (普通は大文字)

codeset　　関連文字コード・セットの名前。名前のこの部分は、通常、オプションです。

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

デフォルト値は en_US です。コネクターがグローバル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、以下の Web サイトにあるコネクターのバージョン・リストを参照してください。

http://www.ibm.com/software/websphere/wbiadapters/infocenter、または
http://www.ibm.com/websphere/integration/wicserver/infocenter

LogAtInterchangeEnd

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

例えば、LogAtInterChangeEnd を true に設定した場合にコネクタ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。デフォルト値は false です。

MaxEventCapacity

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

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

MessageFileName

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

コネクター・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: 特定の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。

MonitorQueue

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です。このプロパティーは、DeliveryTransport プロパティー値が JMS であり、かつ DuplicateEventElimination が TRUE に設定されている場合にのみ使用されます。

デフォルト値は CONNECTORNAME/MONITORQUEUE です。

OADAutoRestartAgent

統合ブローカーが ICS で、Repository Directory が <REMOTE> の場合にのみ有効です。

異常シャットダウンの後、Object Activation Daemon (OAD) がアプリケーション固有のコンポーネントの再始動を自動的に試行するかどうかを指定します。このプロパティーは自動再始動を要求します。

デフォルト値は false です。

OADMaxNumRetry

統合ブローカーが ICS で、Repository Directory が <REMOTE> の場合にのみ有効です。

異常シャットダウンの後で OAD がアプリケーション固有のコンポーネント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。

デフォルト値は 1000 です。

OADRetryTimeInterval

統合ブローカーが ICS で、Repository Directory が <REMOTE> の場合にのみ有効です。

異常シャットダウンの後で OAD がアプリケーション固有のコンポーネントの再始動を自動的に試行する間隔 (分単位) を指定します。アプリケーション固有のコンポーネントが指定された間隔以内に始動しない場合、OAD は、『OADMaxNumRetry』で指定された回数だけ試行を繰り返します。

デフォルト値は 10 です。

PollEndTime

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

このプロパティ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 ですが、この値は必ず変更する必要があります。

PollFrequency

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

- ポーリング・アクション間のミリ秒数。

- ワード key。コネクターは、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。
- 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。

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。このプロパティーが使用されるかどうかを特定のコネクターについて判別するには、該当するアダプター・ガイドのインストールと構成についての章を参照してください。

PollQuantity

コネクタ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。

PollStartTime

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

このプロパティ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 ですが、この値は必ず変更する必要があります。

RequestQueue

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

デフォルト値は REQUESTQUEUE です。

RepositoryDirectory

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 WMQI または WAS の場合には、この値を *<local directory>* に設定する必要があります。

ResponseQueue

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。

RestartCount

設定されている数のイベントの処理が完了すると、コネクターが自動的にシャットダウンしてから再始動します。イベントの数を RestartCount に設定します。このプロパティーを有効にするには、コネクターがポーリング・モードで稼動している必要があります (PollFrequency を「p」に設定します)。

要求処理時に設定されたイベント数を超えると、コネクターがシャットダウンし、次回ポーリングするときに再始動します。

RestartRetryCount

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コネクターに対して使用する場合、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スレーブ側のアプリケーション固有のコンポーネントの再始動を試行する回数が指定されます。

デフォルト値は 3 です。

RestartRetryInterval

コネクタ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コネクターに対して使用する場合、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スレーブ側のアプリケーション固有のコンポーネントの再始動を試行する間隔が指定されます。

デフォルト値は 1 です。

RHF2MessageDomain

WebSphere MQ Integrator Broker でのみ使用されます。

この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 MQ Integrator Broker に送信するときに、コネクター・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。この構成可能なドメイン名により、ユーザーは WebSphere MQ Integrator Broker によるメッセージ・データの処理方法を追跡できます。

サンプル・ヘッダーを以下に示します。

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

デフォルト値は mrm ですが、このプロパティーには xml も設定できます。このプロパティーは、DeliveryTransport が JMS に設定されており、かつ WireFormat が CwXML に設定されている場合にのみ表示されます。

SourceQueue

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、62 ページの『ContainerManagedEvents』を参照してください。

デフォルト値は SOURCEQUEUE です。

SynchronousRequestQueue

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、SynchronousRequestQueue にメッセージを送信し、SynchronousResponseQueue でブローカーから戻される応答を待機します。コネクタ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。

SynchronousResponseQueue

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。

SynchronousRequestTimeout

コネクターが同期要求への応答を待機する時間を分単位で指定します。コネクターは、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をエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

WireFormat

トランスポートのメッセージ・フォーマットです。

- 統合ブローカーが WMQI または WAS の場合には、設定値は CwXML です。
- 統合ブローカーが ICS であり、RepositoryDirectory の値が <REMOTE> の場合には、設定値は CwBO です。

WisfSynchronousRequest Timeout

WAS 統合ブローカーでのみ使用されます。

コネクターが同期要求への応答を待機する時間を分単位で指定します。コネクターは、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をエラー・メッセージとともに障害キューに移動します。

デフォルト値は 0 です。

付録 B. Connector Configurator

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

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

- コネクタ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
- 構成ファイルを作成する
- 構成ファイル内のプロパティーを設定する

注:

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

この付録では、次のトピックについて説明します。

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 Broker (WMQI)
- WebSphere Application Server (WAS)

Connector Configurator の実行モードと使用する構成ファイルのタイプは、実行する統合ブローカーによって異なります。例えば、使用している統合ブローカーが WMQI の場合、Connector Configurator を System Manager から実行するのではなく、直接実行します (74 ページの『スタンドアロン・モードでの Configurator の実行』を参照)。

新しいアダプターをインストールする度に、コネクターの**構成ファイル**を作成する必要があります。この構成ファイルでは次の項目が設定されます。

- コネクターの標準プロパティーとアプリケーション固有のプロパティーが設定される。
- サポートするビジネス・オブジェクトとメタオブジェクトが指定される。

- コネクターが実行時に使用するロギング値とトレース値が設定される。
- アダプターのメッセージング・ハンドラーおよびデータ・ハンドラーにより使用されるプロパティー値が設定される。
- 既存のコネクターのコネクター・プロパティーを変更できるようになる。

この構成ファイルを作成し、構成ファイルの設定値を変更するには、Connector Configurator を使用します。

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新規に定義する必要はありません。ファイルを作成すると、Connector Configurator により標準プロパティ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、76 ページの『新規テンプレートの作成』のステップに従い、テンプレートを新規に作成します。

注: Connector Configurator は、Windows 環境内でのみ実行されます。UNIX 環境でコネクタ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 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 (すべてのブローカー)
- System Manager から実行 (ICS および WAS のみ)

スタンドアロン・モードでの Configurator の実行

どのブローカーを実行している場合にも、Connector Configurator を個別に実行し、コネクター構成ファイルを編集できます。ただし、統合ブローカーとして WMQI を使用している場合には、Connector Configurator はスタンドアロン・モードでのみ実行できます。

これを行うには、以下のステップを実行します。

- **「スタート」>「プログラム」**から、**「IBM WebSphere InterChange Server」>「IBM WebSphere Business Integration Toolset」>「開発」>「Connector Configurator」**をクリックします。

- **「ファイル」>「新規」>「構成ファイル」**を選択します。
- **「システム接続: Integration Broker」**の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS 接続、WMQI 接続、WAS 接続のいずれかを選択します。

ブローカーとして ICS または WAS を使用するときに ICS または WAS で使用する構成ファイルを作成する場合には、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方法が便利です (81 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

統合ブローカーとして ICS または WAS を使用している場合には、System Manager から Connector Configurator を実行できます。

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

1. System Manager を開きます。
2. 「System Manager」ウィンドウで、**「統合コンポーネント・ライブラリー」**アイコンを展開し、**「コネクター」**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 接続または WAS 接続のいずれかを選択します。

既存の構成ファイルを編集するには、以下のステップを実行します。

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

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、または既存のファイルをテンプレートとして使用します。

- テンプレートの新規作成については、76 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。

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。

テンプレートは以下のように作成します。

1. **「ファイル」>「新規」>「コネクター固有プロパティー・テンプレート」**をクリックします。
2. 以下のフィールドを含む**「コネクタ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」**、**「名前」**

     このテンプレートが使用されるコネクター (またはコネクターのタイプ) を表す固有の名前を入力します。テンプレートから新規構成ファイルを作成するためのダイアログ・ボックスを開くと、この名前が再度表示されます。
   - **「旧テンプレート」**、**「変更する既存のテンプレートを選択してください」**

     「テンプレート名」表示に、現在使用可能なすべてのテンプレートの名前が表示されます。
   - テンプレートに含まれているコネクタ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。テンプレートを作成するときには、ご使用の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。
3. **「テンプレート名」**表示からテンプレートを選択し、その名前を**「名前の検索」**フィールドに入力し (または**「テンプレート名」**で自分の選択項目を強調表示し)、**「次へ」**をクリックします。

ご使用のコネクタ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。Connector Configurator には、デフォルト選択として、なにもプロパティー定義が含まれていない、**「None」**という名前のテンプレートが用意されています。

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。

- **プロパティーを編集**

  用意されているボタンを使用して (または**「プロパティーを編集」**表示内で右マウス・ボタンをクリックして)、テンプレートに新規プロパティーを追加したり、既存のプロパティーの編集や削除を行なったり、既存のプロパティーに子プロパティーを追加したりします。

  子プロパティーとは、別のプロパティー (親プロパティー) の属性です。親プロパティーには、単純値、子プロパティー、またはそれら両方を取り込むことができます。親プロパティーと子プロパティーは階層の関係にあります。これらのプロ

パティーから構成ファイルを作成すると、Connector Configurator は階層プロパティー・セットを、親プロパティーの左側にボックスで囲んだ正符号 (+) を付けて識別します。

- **プロパティー・タイプ**

  Boolean、String、Integer、または Time のいずれかのプロパティー・タイプを選択します。

- フラグ

  このプロパティーに適用する、**「標準フラグ」** (IsRequired、IsDeprecated、IsOverridden) または**「カスタム・フラグ」** (ブール演算子の場合) を設定できます。

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。これを行うには、以下のステップを実行します。

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

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

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

   「値の範囲」の列には、**「プロパティー値」**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」**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、以下のステップを実行します。

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」**表示で依存プロパティーを強調表示させて矢印をクリックし、**「依存プロパティー」**表示に移動させます。
6. **「完了」**をクリックします。Connector Configurator により、XML 文書として入力した情報が、Connector Configurator がインストールされている ¥bin ディレクトリーの ¥data¥app の下に保管されます。

新しい構成ファイルを作成

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

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。

- Connector Configurator のホーム・メニューで、**「ファイル」>「新規」>「コネクター構成」**をクリックします。**「新規コネクター」**ダイアログ・ボックスが表示されます。
- **「Integration Broker」**フィールドで、ICS 接続、WMQI 接続、WAS 接続のいずれかを選択します。

  注: WMQI を選択するには、Connector Configurator を System Manager ではなく「スタート」メニューから起動する必要があります。
- この章で後述する説明に従って**「新規コネクター」**ウィンドウの残りのフィールドに入力します。

ICS または WAS のファイルを新規に作成するには、次のいずれかの操作を行います。

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

コネクター固有のテンプレートからの構成ファイルの作成

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

1. **「ファイル」>「新規」>「コネクター構成」**をクリックします。
2. 以下のフィールドを含む**「新規コネクター」**ダイアログ・ボックス表示されます。
   - **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、「connector」という単語で終わり、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。例えば、コネクター・ファイル名が `PeopleSoft.jar` の場合は、`PeopleSoftConnector` と入力します。

     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、WMQI 接続、WAS 接続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」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。
3. 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。
4. ファイルを保管するには、**「ファイル」>「保管」>「ファイルに」**をクリックするか、**「ファイル」>「保管」>「プロジェクトに保管」**をクリックします。プロジェクトに保管するには、ブローカーとして ICS または WAS を使用しており、かつ 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。

5. この章で後述する手順に従って、「Connector Configurator」ウィンドウの各タブにあるフィールドに値を入力し、コネクタ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 つまたは複数の形式になります。

- 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、そのファイルを構成ファイル (*.cfg ファイル) として保管する必要があります。

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

1. Connector Configurator 内で、**「ファイル」>「開く」>「ファイルから」**をクリックします。
2. **「ファイル・コネクターを開く」**ダイアログ・ボックス内で、以下のいずれかのファイル・タイプを選択して、使用可能なファイルを調べます。
   - 構成 (*.cfg)
   - ICS リポジトリー (*.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、以下のステップを実行します。

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

構成ファイルの完成

構成ファイルを開くか、プロジェクトからコネクターを開くと、「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、以下のステップを実行します。

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。ここでファイルを保管するか、または 84 ページの『サポートされるビジネス・オブジェクト定義の指定』の説明に従い残りの構成フィールドに値を入力することができます。
3. 構成が完了したら、**「ファイル」>「保管」>「プロジェクトに」**を選択するか、または**「ファイル」>「保管」>「ファイルに」**を選択します。

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

   複数のコネクタ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。

   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。

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)

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のプロパティーは、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。

「標準プロパティー」と**「コネクター固有プロパティー」**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。

- **「更新メソッド」**フィールドは通知用であり、構成できません。このフィールドは、値が変更されたプロパティーをアクティブにするために必要なアクションを示します。

標準コネクター・プロパティーの設定

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

アプリケーション固有の構成プロパティーの設定

アプリケーション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」**クリックし、プロパティーに子プロパティーを追加するときは**「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。
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. 『標準コネクター・プロパティーの設定』の説明に従い、変更内容を保管するかまたは破棄するかを選択します。

各プロパティ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

「プロパティーを編集」ウィンドウの**「暗号化」**チェック・ボックスにチェックマークを付けると、アプリケーション固有のプロパティ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・ユーザーズ・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

ほとんどのコネクター・プロパティーは静的プロパティーであり、**更新メソッド**は「コンポーネント再始動」です。変更内容を有効にするには、変更したコネクター構成ファイルを保管した後でコネクターを再始動する必要があります。

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。

サポートされているビジネス・オブジェクトを指定するには、指定するビジネス・オブジェクトとそのマップがシステム内に存在している必要があります。

- 統合ブローカーが ICS または WAS の場合、ビジネス・オブジェクト定義とマップ定義が System Manager プロジェクトに保管されている必要があります。
- 統合ブローカーが WMQI の場合、ビジネス・オブジェクト定義と MQ メッセージ・セット・ファイルが存在している必要があります。

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

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

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・オブジェクトがエージェント・サポートを備えている場合、システムは、コネクター・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、大半のアプリケーション API では「緊急」レベルはサポートされないため、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 WMQI の場合

MQ メッセージ・セット・ファイル (*.set ファイル) には、メッセージ・セット ID が記述されています。この ID は、コネクターがサポートするビジネス・オブジェクトを指定するときに、Connector Configurator で必要となります。MQ メッセージ・セット・ファイルの作成の詳細については、「*WebSphere MQ Integrator Broker 用インプリメンテーション・ガイド*」を参照してください。

ビジネス・オブジェクト定義をシステムに追加するごとに、Connector Configurator を使用して、コネクターがサポートするビジネス・オブジェクトを指定する必要があります。

重要: コネクターがメタオブジェクトを必要とする場合は、ビジネス・オブジェクトと同様の方法で、各メタオブジェクトのメッセージ・セット・ファイルを作成し、それらのファイルを Connector Configurator にロードする必要があります。

サポートされるビジネス・オブジェクトを指定するには、以下の手順を実行します。

1. **「サポートされるビジネス・オブジェクト」**タブを選択し、**「ロード」**をクリックします。**「メッセージ・セット ID ファイルを開く」**ダイアログ・ボックスが表示されます。
2. コネクターのメッセージ・セット・ファイルを格納したディレクトリーに移動し、1 つ以上の適切なメッセージ・セット・ファイル (*.set) を選択します。
3. **「開く」**をクリックします。**「ビジネス・オブジェクト名」**フィールドには、*.set ファイル内に格納されているビジネス・オブジェクト名が表示されます。また、各ビジネス・オブジェクトの数値メッセージ・セット ID が、対応する「メッセージ・セット ID」フィールドにリストされます。
   メッセージ・セット ID を変更しないでください。これらの名前と数値 ID は、構成ファイルを保管するときに保管されます。
4. ビジネス・オブジェクトを構成に追加するときには、ビジネス・オブジェクトのメッセージ・セット・ファイルをロードする必要があります。構成内にすでに存在するビジネス・オブジェクト名を含むメッセージ・セットをロードしようとした場合、または重複するビジネス・オブジェクト名を含むメッセージ・セット・ファイルをロードしようとした場合、Connector Configurator は重複を検出し、**「結果のロード」**ダイアログ・ボックスを表示します。
   このダイアログ・ボックスは、重複が存在する 1 つ以上のビジネス・オブジェクト名を表示します。表示された各重複名の**「メッセージ・セット ID」**フィールド内でクリックし、使用するメッセージ・セット ID を選択します。

ご使用のブローカーが WAS の場合

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

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

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

関連付けられたマップ (ICS のみ)

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付けられたマップのリストをサポートします。このリストは、**「関連付けられたマップ」**タブを選択すると表示されます。

ビジネス・オブジェクトのリストには、エージェントでサポートされるアプリケーション固有のビジネス・オブジェクトと、コントローラーがサブスクライブ・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」タブでビジネス・オブジェクトを追加指定した場合、その内容は、「Connector Configurator」ウィンドウの**「ファイル」**メニューから**「プロジェクトに保管」**を選択して、変更を保管した後に、このリストに反映されます。

- **関連付けられたマップ**

  この表示には、コネクターの、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**

  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、各コネクターでサポートされるそれぞれのビジネス・オブジェクトにマップを自動的にバインドしようとします。複数のマップでその入力データとして同一のビジネス・オブ

ジェクトが使用されている場合、サーバーは、他のマップのスーパーセットである 1 つのマップを見付けて、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. **「明示的 (Explicit)」**列で、バインドするマップのチェック・ボックスにチェックマークを付けます。
2. ビジネス・オブジェクトに関連付けるマップを選択します。
3. 「Connector Configurator」ウィンドウの**「ファイル」**メニューで、**「プロジェクトに保管」**をクリックします。
4. プロジェクトを ICS に配置します。
5. 変更を有効にするため、サーバーをリブートします。

リソース (ICS)

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

メッセージングの構成 (ICS)

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。

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データ・ハンドラーを使用できるわけではありません。

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

構成ファイルの保管

コネクターの構成が完了したら、コネクター構成ファイルを保管します。Connector Configurator では、構成中に選択したブローカー・モードで構成ファイルが保管されます。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。

ICS の場合:

- System Manager から、`*.con` 拡張子付きファイルとして ICS ユーザー・プロジェクトに保管するか、
- 指定したディレクトリーに保管します。
- ファイルをローカル構成ファイルとして使用する場合には、スタンドアロン・モードで、`*.cfg` 拡張子が付いたファイルとしてディレクトリー・フォルダーに保管します。

WMQI の場合:

- 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。

WAS の場合:

- System Manager から、*.con 拡張子付きファイルとして WAS ユーザー・プロジェクトに保管するか、
- 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。

構成ファイルを作成し、そのプロパティーを設定した後に、ご使用のコネクターに対応した適切な場所にこの構成ファイルを配置する必要があります。

- 統合ブローカーとして ICS を使用している場合は、構成を System Manager プロジェクトに保管し、System Manager を使用してそのファイルを ICS にロードします。
- 統合ブローカーとして WMQI を使用している場合は、構成ファイルを適切な場所にコピーします。これは、ご使用のコネクターの始動ファイルに指定されている構成ファイルの場所と完全に同じ場所でなければなりません。
- 統合ブローカーとして WAS を使用している場合には、ファイルを WAS ユーザー・プロジェクトに保管します。**「ファイル」>「エクスポート」**を選択し、.wsdl ファイルを作成します。作成された .wsdl ファイルは WSAD-IE へインポートできます。
  また、構成ファイルを .jar ファイルとして指定のディレクトリーへエクスポートできます。

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 インプリメンテーション・ガイド*」
- WMQI: 「*WebSphere MQ Integrator Broker 用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**「BrokerType」**フィールドで、ご使用のブローカーに合った値を選択します。
  現行値を変更すると、プロパティー画面の利用可能なタブおよびフィールド選択がただちに変更され、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、以下に太文字で示した行を追加してください。

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

                    <DefaultValue>en_US</DefaultValue>
            </ValidValues>
    </Property>
```

特記事項

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

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

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

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、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 Burlingame Laboratory Director
IBM Burlingame Laboratory
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。

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

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

プログラミング・インターフェース情報

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

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

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

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

商標

以下は、IBM Corporation の商標です。

IBM
IBM ロゴ
AIX
CrossWorlds
DB2
DB2 Universal Database
Domino
Lotus
Lotus Notes
MQIntegrator
MQSeries
Tivoli
WebSphere

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

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

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

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

WebSphere Business Integration Adapter Framework V2.4.0