IBM WebSphere Business Integration Server Express and Express Plus

Adapter for iSeries ユーザーズ・ガイド

Server Express バージョン 2.0.x

IBM WebSphere Business Integration Server Express and Express Plus

Adapter for iSeries ユーザーズ・ガイド

Server Express バージョン 2.0.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、 83 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、コネクター・バージョン 2.0.x、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 Server Express and Express Plus Adapter for iSeries User Guide Server Express Version 2.0.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 2005.8

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

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

目次

本書について

製品 IBM(R) WebSphere(R) Business Integration Server Express および IBM(R) WebSphere(R) Business Integration Server Express Plus は、InterChange Server Express、関連する Toolset Express、CollaborationFoundation、およびソフトウェア統合アダプターのセットで構成されています。Toolset Express に含まれるツールは、ビジネス・オブジェクトの作成、変更、および管理に役立ちます。プリパッケージされている各種アダプターは、お客様の複数アプリケーションにまたがるビジネス・プロセスに応じて、いずれかを選べるようになっています。標準的な処理のテンプレートである CollaborationFoundation は、カスタマイズされたプロセスを簡単に作成できるようにするためのものです。

アダプター・ポートフォリオは、主要な e-business テクノロジー、エンタープライズ・アプリケーション、レガシー、およびメインフレーム・システムに統合コネクティビティーを提供します。本製品には、コンポーネントをカスタマイズ、作成、および管理するためのツールとテンプレートが含まれており、これにより、ビジネス・プロセスの統合を実現します。

本書では、Adapters for iSeries(TM) 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 システムの一部としてコネクターを実装する WebSphere コンサルタントおよびお客様を対象読者としています。本書の情報を利用するには、以下の分野に関する十分な知識が必要です。

- コネクター開発
- ビジネス・オブジェクト開発
- i5/OS アプリケーションのアーキテクチャー
- iSeries 統合ファイル・システム

関連文書

本書の対象製品の一連の関連文書には、WebSphere Business Integration Server Express のどのインストールにも共通する機能とコンポーネントの解説のほか、特定のコンポーネントに関する参考資料が含まれています。

文書は次のサイトで、ダウンロード、インストール、および表示することができます。

http://www.ibm.com/websphere/wbiserverexpress/infocenter

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

http://www.ibm.com/software/integration/websphere/support/

関心のあるコンポーネント・エリアを選択し、「Technotes」セクションと「Flashes」セクションを参照してください。

表記上の規則

本書では、以下のような規則を使用しています。

`Courier フォント`	コマンド名、ファイル名、入力情報、システムが画面に出力した情報など、記述されたとおりの値を示します。
太字	初出語を示します。
イタリック、 ***イタリック***	変数名または相互参照を示します。
青のアウトライン	オンラインで表示したときにのみ見られる青のアウトラインは、相互参照用のハイパーリンクです。アウトラインの内側をクリックすると、参照先オブジェクトにジャンプします。
{ }	構文の記述行の場合、中括弧 {} で囲まれた部分は、選択対象のオプションです。1 つのオプションのみを選択する必要があります。
[ ]	構文の記述行の場合、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指定できることを意味します。
< >	命名規則では、不等号括弧は名前の個々の要素を囲み、各要素を区別します。(例: `<server_name><`*`connector_name`*`>tmp.log`)
/, ¥	本書では、規則としてディレクトリー・パスに円記号 (¥) を使用します。Linux システムの場合には、円記号をスラッシュ (/) に置き換えてください。すべての WebSphere Business Integration システム製品のパス名は、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 `$`*`text`*	% 記号で囲まれたテキストは、Windows の `text` システム変数またはユーザー変数の値を示します。Linux 環境での同等の表記は `$`*`text`* で、これは、Linux 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

バージョン 2.0

- iSeries 上でのデータ・キュー・アクセスのサポート
- ポーリングのサポートによるデータ・キューのモニター。アダプターが両方向になりました。これによってアダプターは、イベント処理および要求処理のいずれもサポートします。
- ODA 機能による iSeries マシン上での RPGLE ソース・プログラムおよびデータ・キューからのビジネス・オブジェクト仕様の生成
- 単一の要求ビジネス・オブジェクトを使用した、特定 RPG プログラムの複数回呼び出し機能。ビジネス・オブジェクトは、同じ RPG プログラムを複数回呼び出した場合に返されるそれぞれの値を持つことができるようになりました。
- RPG でプログラムが 1 つの呼び出して 2 回呼び出された場合の問題も、このバージョンで修正されました。

第 1 章 概要

本章では、Adapter for iSeries について説明します。アダプターには、IBM Toolbox for Java (Java(TM) クラスのセット) を使用して、iSeries または i5/OS システム上で RPG プログラムを実行およびデータ・キューにアクセスする機能が用意されています。IBM Toolbox for Java には、データ・キューへのアクセスおよびプログラムを実行するためのクラスのセットが用意されています。アダプターは、これらのクラスと、着信ビジネス・オブジェクトからの情報を使用して、プログラムのパラメーター・リストを作成し、プログラムの実行およびデータ・キューの読み取り/書き込みを行います。iSeries アダプターは現在、要求処理およびイベント処理のいずれもサポートしています。

アダプターは、アプリケーション固有のコンポーネントとコネクター・フレームワークで構成されています。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応じて調整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との仲介役として機能し、そのコードはいずれのアダプターにも共通です。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との間で以下のようなサービスを提供します。

- ビジネス・オブジェクトの送信
- 始動メッセージと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、この 2 つのコンポーネントをまとめてアダプターと呼びます。

統合ブローカー InterChange Server Express とアダプターの関係の詳細については、「*システム管理ガイド*」または「*システム・インプリメンテーション・ガイド*」を参照してください。

本章の内容は、次のとおりです。

iSeries i5/OS システムの概要

IBM iSeries (i5/OS と呼称) は、高度に統合された信頼性の高いサーバー・プラットフォームで、複数のオペレーティング環境を同時に実行して業務に活用できます。特性として保全性とセキュリティーに優れているので、多くの重要なアプリケーションで使用することができます。

RPG は、単純な報告書作成プログラム (Report Program Generator、このプログラムの名前の出所) から、iSeries マシン上の強力なアプリケーション開発プロシージャー型言語に発展してきました。現在 RPG は、iSeries 上で ILE (統合化言語環境) によってサポートされています。

ホスト・サーバーは、図 1 に示すように、アプリケーションを実行している装置 (クライアント PC など) からの要求を処理して、文書の印刷などのタスクを実行できるようにします。iSeries および i5/OS コンピューターは、ファイル、データベース、アプリケーション、メール、プリント、マルチメディア、FAX、およびワイヤレス通信などの多くのタスクを一度に実行できる全機能サーバーです。タスク・サーバーは、システム上でそれぞれ個別のジョブとして実行されます。また、サーバー・ジョブは、それぞれソケット接続を介してデータ・ストリームを送受信します。

このようなホスト・サーバーの 1 つに、リモート・コマンドおよび分散プログラム呼び出しサーバーがあります。このサーバーは、iSeries システムまたは i5/OS システムでのプログラムを実行します。

IBM Toolbox for JAVA には多数のパッケージが含まれており、これらのパッケージはさまざまな機能に対応しています。例えば、アクセス・クラスは、サインオン情報の管理、ソケット接続の作成と維持、およびデータの送受信を行います。一方、コマンド呼び出しクラスは、iSeries と i5/OS のバッチ・コマンドを実行します。

IBM の Adapter for iSeries は、アクセス・クラスとプログラム呼び出しクラスを使用して RPG プログラムを呼び出します。iSeries または i5/OS 形式と Java 形式の間で数値データと文字データを変換する機能は、データ変換クラスによって提供されます。

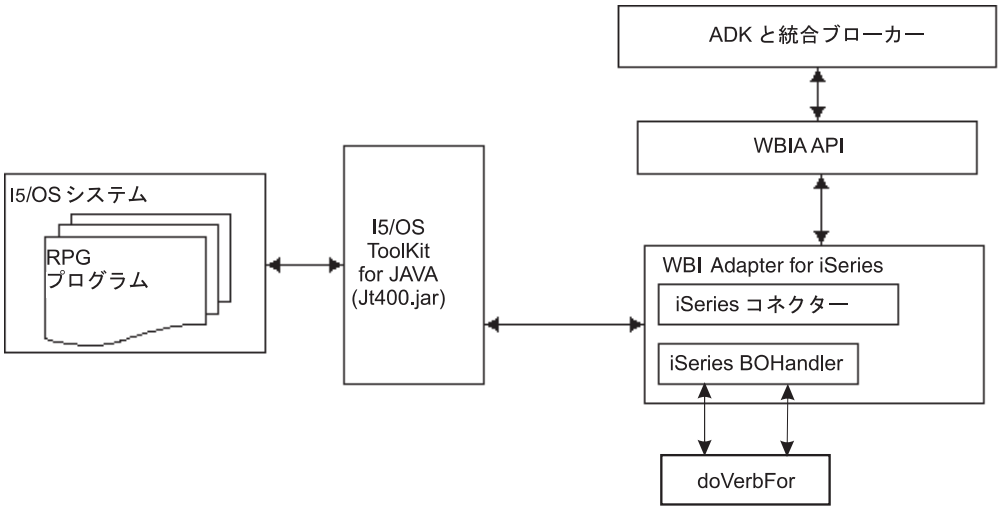


図 1. i5/OS クライアント - サーバー・アーキテクチャーの概要

i5/OS 上で稼働する i5/OS はさまざまなタイプのタスクを処理することができますが、Adapter for iSeries から使用されるのはリモート・コマンドおよび分散プログラム呼び出しサーバーのみです。このサーバーは、i5/OS システムでのプログラムを実行します。

Adapter for iSeries を使用した場合のクライアントからサーバーへの接続のしくみを、3 ページの図 2 に示します。

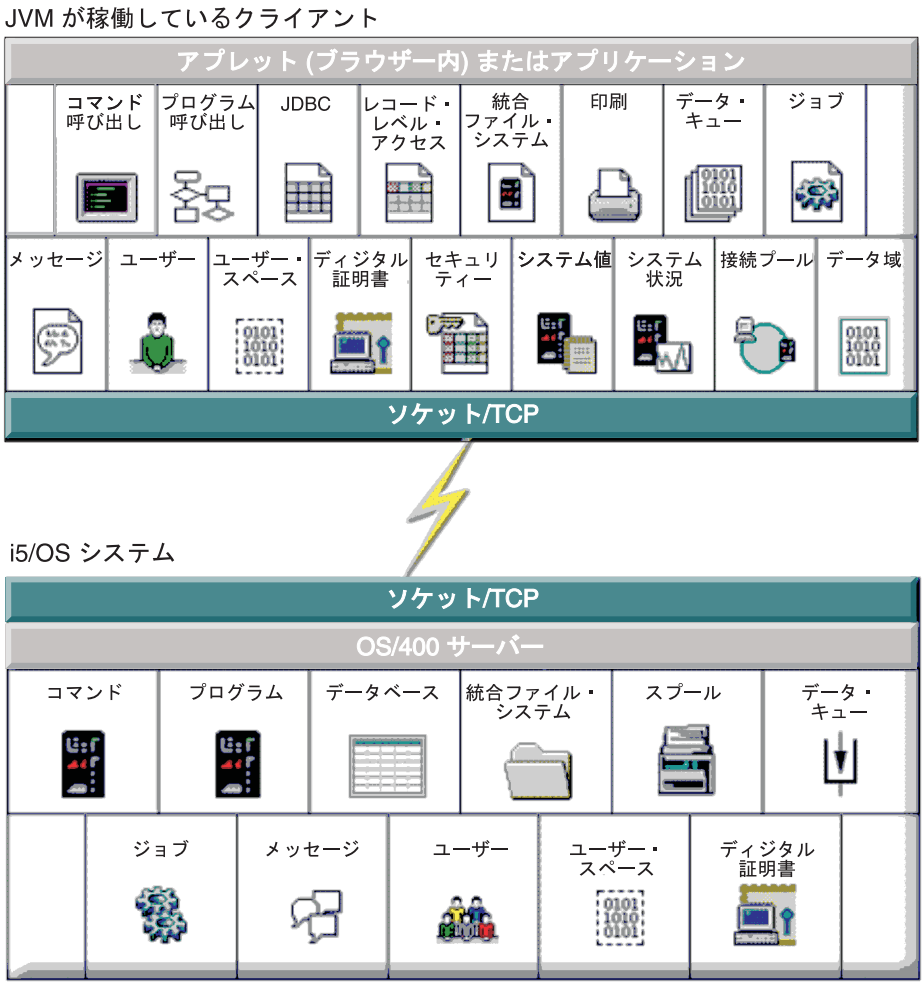


図 2. Adapter for iSeries を使用した場合の接続のしくみ

データ・キュー

iSeries 上のデータ・キューによって、ジョブ間の高速なコミュニケーションが可能になります。従ってデータ・キューは、ジョブ間でデータを同期したり受け渡すための優れた方法になります。iSeries 上のデータ・キューによって次の処理が可能になります。

- 多くのジョブが同時にデータ・キューにアクセスできます。
- データ・キュー上のメッセージはフリー・フォーマットです。
- データ・キューは、同期処理にも非同期処理にも使用できます。
- データ・キュー上のメッセージは、次のいずれかの方法で順序を指定できます。
  - 後入れ先出し法 (LIFO)
  - 先入れ先出し法 (FIFO)
  - キー付き

キー付きデータ・キュー上の各メッセージは、対応するキーを持ちます。メッセージは、対応するキーを指定することによってのみ、キューから取り出すことができます。

アダプターの動作方法

ここでは、アダプターでビジネス・オブジェクトがどのように処理されるかについて説明します。

ビジネス・オブジェクトの処理

アダプターは、統合ブローカーからビジネス・オブジェクト要求を受信すると、RPG プログラムのパラメーター・リストを作成し、iSeries システムまたは i5/OS システムとの接続を確立して、RPG プログラムを実行します。

着信したビジネス・オブジェクトには、接続関連の子属性が含まれています。この属性の情報は、iSeries システムまたは i5/OS システムへの接続に使用されます。

コネクターの動作

コネクターは、統合ブローカーとビジネス・オブジェクト・ハンドラーの間で、ビジネス・オブジェクトを受け渡します。ビジネス・オブジェクトは、以下の手順で処理されます。

1. コネクターがフレームワークに BOHandler を登録します。
2. フレームワークから BOHandler に BO 要求が送信されます。
3. BOHandler で、着信ビジネス・オブジェクトの属性の情報に基づいて RPG プログラムのパラメーター・リストが作成されます。
4. BOHandler から、iSeries システムまたは i5/OS システムで実行される RPG プログラムが呼び出されます。

   注: これは、基本的には iSeries システム上または i5/OS システム上の RPG プログラムを実行するための呼び出しです。これらのシステムは、この呼び出しの後、成功または失敗を通知するメッセージを戻します。
5. BOHandler が RPG プログラムの実行結果をアダプター・フレームワークに戻します。また、戻されたパラメーターをビジネス・オブジェクトに取り込みます。

アダプターは、Java で作成されており、次の 2 つのコンポーネントで構成されています。

- コネクター
- ビジネス・オブジェクト・ハンドラー

第 2 章 iSeries アダプターのインストール

この章では、コネクターのインストールと構成のプロセスについて説明します。本章の内容は、次のとおりです。

iSeries 環境のアダプター

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。

- ブローカーの互換性
- ソフトウェア前提条件
- アダプターのプラットフォーム

ブローカーの互換性

アダプターが使用するアダプター・フレームワークは、アダプターと通信する統合ブローカーのバージョンとの互換性を備えている必要があります。Adapter for iSeries バージョン 2.0 は、以下のバージョンのアダプター・フレームワークおよび統合ブローカー InterChange Server Express バージョン 4.2.x 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、2.5、および 2.6。

注: 統合ブローカーおよびその前提条件のインストールに関する説明については、「システム・インプリメンテーション・ガイド」を参照してください。

アダプターのプラットフォーム

アダプターは次のプラットフォームで実行されます。

- **Linux:**
  Red Hat 2.1 Enterprise Linux WS/AS/ES 3.0 (Update 2)、Intel (IA32)
  SuSE Linux ES 8.1、Intel (IA32)
  SuSE Linux ES 9.0、Intel (IA32)

  注: WebSphere Business Integration Adapter Framework V2.6 の TMTP (Tivoli Monitoring for Transaction Performance) コンポーネントは、Linux Red Hat ではサポートされません。

- **Windows:**
  Windows 2003 (Standard Edition または Enterprise Edition)

- OS/400 V5R2 および i5/OS V5R3。i5/OS は、特に明記されていない限り、OS/400 および i5/OS を指します。

前提条件

コネクターを使用するには、ご使用の環境に以下のものが用意されている必要があります。

1. 以下の Java 環境と JAR ファイルが必要です。
   - JDK 1.4 以降
   - JSSE (Java Secure Socket Extension) 1.0
   - Jt400.jar ファイル

     注: アダプターが Windows で実行されている場合は、次の Toolbox Web サイトから最新の jt400.jar ファイルをダウンロードできます。http://www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm。jt400.jar は、%Product_dir%¥connectors¥iSeries ディレクトリーにコピーする必要があります。

     アダプターが iSeries で実行されている場合は、ロケーション /QIBM/ProdData/HTTP/Public/jt400/lib/jt400.jar から jt400.jar ファイルを参照できます。
   - WBIA.jar ファイル
   - CrossWorlds.jar ファイル
   - BIA_iSeries.jar ファイル
2. iSeries アダプターは、以下のいずれかのバージョンの i5/OS を介して i5/OS に接続するように設計されています。
   - バージョン 5 リリース 1 から 3
   - バージョン 4 リリース 1 から 3
3. i5/OS のホスト・サーバー・オプションがインストールされ、実行されている必要があります。

   注: i5/OS データ・キュー・サーバーで peek 関数が正常に実行されるようにするには、PTF を適用する必要があります。次のリンクから適切な PTF を入手する必要があります。

   ```
   http://www-1.ibm.com/servers/eserver/iseries/toolbox/
   hostservicepackdetail.htm
   ```
4. RPG III または IV が実行されている必要があります。

iSeries アダプターと関連ファイルのインストール

WebSphere Business Integration Server Express アダプター製品のインストールの詳細については、「*WebSphere Business Integration Server Express インストール・ガイド*」(Windows 版、Linux 版、OS/400 および i5/OS 版) を参照してください。このガイドは、次のサイトの WebSphere Business Integration Server Express Adapters Infocenter にあります: http://www.ibm.com/websphere/wbiserverexpress/infocenter。

インストール済みファイルの構造

アダプターのインストールでは、コネクターに関連付けられた標準ファイルがシステムにコピーされます。ユーティリティーにより、コネクターが *ProductDir*¥connectors¥iSeries ディレクトリーにインストールされ、コネクターへのショートカットが「スタート」メニューに追加されます。

注: *ProductDir* は、製品のインストール先ディレクトリーを表します。

表 1 に、コネクターが使用するファイル構造と、インストーラーを使用してコネクターをインストールする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。

表 1. コネクターのファイル構造

***ProductDir* のサブディレクトリー**	**説明**
¥connectors¥iSeries¥BIA_iSeries.jar	iSeries コネクターのみが使用するクラスが含まれている
¥connectors¥iSeries¥start_iSeries.bat	iSeries コネクターの始動スクリプト (Windows)
/connectors/iSeries/start_iSeries.sh	iSeries コネクターの始動スクリプト (Linux)
/connectors/iSeries/start_iSeries.sh	iSeries コネクターの始動スクリプト (i5/OS)
/connectors/iSeries/BIA_iSeries.jar	iSeries コネクターの始動スクリプト (i5/OS)
¥connectors¥iSeries¥ext¥	ODA 生成の .jar ファイルを保管できるディレクトリー。このディレクトリーに保管する場合は、始動スクリプト (start_iSeries.bat または start_iSeries.sh) でディレクトリーを指定します。
¥connectors¥messages¥iSeriesAdapter.txt	コネクターのメッセージ・ファイル
¥ODA¥iSeries¥BIA_iSeriesODA.jar	iSeries ODA
¥ODA¥iSeries¥start_iSeriesODA.bat	ODA 始動ファイル (Windows)
/ODA/iSeries/start_iSeriesODA.sh	ODA 始動ファイル (Linux)
/ODA/iSeries/start_iSeriesODA.sh	ODA 始動ファイル (i5/OS)
repository¥iSeries¥CN_iSeries.txt	コネクターのリポジトリー定義
¥connectors¥iSeries¥ start_iSeries_service.bat	始動スクリプト (Windows)

インストール後の作業

インストールが終わったら、始動する前に、アダプターを構成する必要があります。詳細については、9 ページの『第 3 章 iSeries アダプターの構成』を参照してください。

第 3 章 iSeries アダプターの構成

この章では、コネクター構成のプロセスについて説明します。本章の内容は、次のとおりです。

- 『コネクターの構成』

コネクターの構成

Adapter for iSeries の構成には、次のセクションで説明する標準コネクター・プロパティーを使用します。また、その次のセクションで説明するコネクター固有のプロパティーも使用します。

このセクションには、以下のトピックがあります。

- 『Connector Configurator Express の概要』
- 10 ページの『Connector Configurator Express の始動』
- 11 ページの『System Manager からの Connector Configurator Express の実行』
- 11 ページの『コネクター固有のプロパティー・テンプレートの作成』
- 14 ページの『新規構成ファイルの作成』
- 17 ページの『構成ファイル・プロパティーの設定』
- 24 ページの『構成ファイルの保管』
- 24 ページの『構成の完了』
- 24 ページの『グローバル化環境における Connector Configurator Express の使用』
- 25 ページの『コネクターの始動』
- 27 ページの『コネクターの停止』
- 28 ページの『複数のコネクター・インスタンスの作成』

Connector Configurator Express の概要

Connector Configurator Express では、統合ブローカー、つまり InterChange Server Express 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。

Connector Configurator Express 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**を作成します。
- **コネクター構成ファイル**を作成します。インストールするコネクターごとに構成ファイルを 1 つ作成する必要があります。
- 構成ファイル内のプロパティーを設定します。
  場合によっては、コネクター・テンプレートでプロパティーに対して設定されているデフォルト値を変更する必要があります。また、サポートされるビジネス・オブジェクト定義と、InterChange Server Express の場合はコラボレーションとともに使用するマップを指定し、必要に応じてメッセージング、ロギング、トレース、およびデータ・ハンドラー・パラメーターを指定する必要があります。

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で使用されるので、新規に定義する必要はありません。構成ファイルを作成すると、Connector Configurator Express によって標準プロパティーがそのファイルに挿入されます。ただし、Connector Configurator Express 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 Express 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、11 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 Express は、Windows 環境でのみ実行できます。Linux 環境でコネクターを実行する場合は、Windows で Connector Configurator Express を使用して構成ファイルを変更し、このファイルを Linux 環境へコピーします。

Connector Configurator Express の始動

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

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

スタンドアロン・モードでの Connector Configurator Express の実行

Connector Configurator Express を別個に実行して、コネクター構成ファイルを編集することができます。

これを行うには、以下のステップを実行します。

- 「**スタート**」>「**プログラム**」から、「**IBM WebSphere Business Integration Express**」>「**Toolset Express**」>「**開発**」>「**Connector Configurator Express**」をクリックします。
- 「**ファイル**」>「**新規**」>「**コネクター構成**」を選択します。
- 「**システム接続統合ブローカー (System Connectivity Integration Broker)**」の隣のプルダウン・メニューをクリックすると、InterChange Server Express が表示されます。

Connector Configurator Express 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 (16 ページの『構成ファイルの完成』を参照)。

System Manager からの Connector Configurator Express の実行

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

Connector Configurator Express 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 Express**」をクリックします。「Connector Configurator Express」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。

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。

- 「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、右マウス・ボタンでクリックします。Connector Configurator Express が開き、この構成ファイルの統合ブローカー・タイプおよびファイル名が上部に表示されます。
- Connector Configurator Express で「**ファイル**」>「**開く**」を選択します。プロジェクトまたはプロジェクトが保管されているディレクトリーからコネクター構成ファイルを選択します。
- 「標準のプロパティー」タブをクリックし、この構成ファイルに含まれ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、または既存のコネクター定義をテンプレートとして使用します。

- テンプレートの新規作成については、『新規テンプレートの作成』を参照してください。
- 既存のファイルを使用する場合には、既存のテンプレートを変更し、新しい名前でこのテンプレートを保管します。既存のテンプレートは ¥WebSphereAdapters¥bin¥Data¥App ディレクトリーにあります。

新規テンプレートの作成

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

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

1. 「**ファイル**」>「**新規**」>「**コネクター固有プロパティー・テンプレート**」をクリックします。
2. 「**コネクター固有プロパティー・テンプレート**」ダイアログ・ボックスが表示されます。
   - 「**新規テンプレート名を入力してください**」の下の「**名前**」フィールドに、新規テンプレートの名前を入力します。テンプレートから新規構成ファイルを作成するためのダイアログ・ボックスを開くと、この名前が再度表示されます。
   - テンプレートに含まれているコネクター固有のプロパティー定義を調べるには、「**テンプレート名**」表示でそのテンプレートの名前を選択します。そのテンプレートに含まれているプロパティー定義のリストが「**テンプレートのプレビュー**」表示に表示されます。
3. テンプレートを作成するときには、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。ご使用のコネクターで使用するコネクター固有のプロパティーが表示されるテンプレートが見つからない場合は、自分で作成する必要があります。
   - 既存のテンプレートを変更する場合には、「**変更する既存のテンプレートを選択してください: 検索テンプレート**」の下の「**テンプレート名**」テーブルのリストから、テンプレート名を選択します。
   - このテーブルには、現在使用可能なすべてのテンプレートの名前が表示されます。テンプレートを検索することもできます。

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

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

プロパティーの一般特性の選択を終えたら、「**値**」タブをクリックします。

値の指定: 「**値**」タブを使用すると、プロパティーの最大長、最大複数値、デフォルト値、または値の範囲を設定できます。編集可能な値も許可されます。これを行うには、以下の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**」のフィールドに値を入力します。

新規プロパティー値を作成するには、以下のステップを実行します。

1. 「**プロパティーを編集**」リストでプロパティーを選択し、右マウス・ボタンでクリックします。
2. ダイアログ・ボックスから「**追加**」を選択します。
3. 新規プロパティー値の名前を入力し、「OK」をクリックします。右側の「**値**」パネルに値が表示されます。

「**値**」パネルには、3 つの列からなるテーブルが表示されます。

「**値**」の列には、「**プロパティー値**」ダイアログ・ボックスで入力した値と、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: コネクター固有プロパティー・テンプレート**」ダイアログ・ボックスが表示されます。

依存プロパティーは、別のプロパティーの値が特定の条件に合致する場合にのみ、テンプレートに組み込まれて、構成ファイルで使用されるプロパティーです。例えば、テンプレートに `PollQuantity` が表示されるのは、トランスポート機構が JMS であり、`DuplicateEventElimination` が `True` に設定されている場合のみです。プロパティーを依存プロパティーとして指定し、依存する条件を設定するには、以下の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。入力した情報が、Connector Configurator Express によって、Connector Configurator Express 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 XML 文書として保管されます。

新規構成ファイルの作成

構成ファイルを新規に作成する場合は、構成ファイルの名前を指定し、統合ブローカー InterChange Server Express を選択する必要があります。

- 「System Manager」ウィンドウで「**コネクター**」フォルダーを右マウス・ボタンでクリックし、「**新規コネクターの作成**」を選択します。Connector Configurator Express 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。
- スタンドアロン・モードの場合は、Connector Configurator Express で「**ファイル**」>「**新規**」>「**コネクター構成**」を選択します。「新規コネクター」ウィンドウで、新規コネクターの名前を入力します。

統合ブローカーを選択して、構成ファイルに表示されるプロパティーを決定する必要もあります。ブローカーを選択するには、以下のステップを実行します。

- 「**Integration Broker**」フィールドで、InterChange Server Express を選択します。
- 本章で後述するように、「**新規コネクター**」ウィンドウの残りのフィールドの値をドロップダウン・メニューを使用して選択します。

コネクター固有のテンプレートからの構成ファイルの作成

コネクター固有のテンプレートを作成すると、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、システムにインストールされているコネクターのファイル名に対応した一意の名前でなければなりません。

     重要: Connector Configurator Express 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**

     「InterChange Server Express」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**」をクリックします。プロジェクトに保管するには、System Manager が実行中でなければなりません。
ファイルとして保管する場合は、「**ファイル・コネクターを保管**」ダイアログ・ボックスが表示されます。`*.cfg` をファイル・タイプとして選択し、「**ファイル名**」フィールド内に名前が正しいスペル (大文字と小文字の区別を含む) で表示されていることを確認してから、ファイルを保管するディレクトリーにナビゲートし、「**保管**」をクリックします。Connector Configurator Express 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 Express」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、iSeries コネクターの場合は CN_iSeries.txt です)。
- InterChange Server Express リポジトリー・ファイル。
  以前にコネクターの InterChange Server Express インプリメンテーションの際に使用された定義が、そのコネクターの構成に使用されたリポジトリー・ファイルに残されていることがあります。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 Express でそのファイルを開き、構成を修正してから、再度保管する必要があります。

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

1. Connector Configurator Express, 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) (この場合、ICS は InterChange Server Express の省略形です)

     InterChange Server Express 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 Express 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 Express」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。InterChange Server Express がブローカーであることを確認します。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、「InterChange Server Express」を選択します。
2. 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 20 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 Express 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 Express 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、または既存のコネクター構成ファイルを開くと、Connector Configurator Express に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 Express には、InterChange Server Express を実行するコネクターの以下のカテゴリーについてプロパティーの値が必要です。

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

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

- 関連付けられたマップ

重要: Connector Configurator Express 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。
- プロパティーごとに「**更新メソッド**」フィールドが表示されます。これは、変更された値をアクティブにするためにコンポーネントまたはエージェントの再始動が必要かどうかを示します。この設定を構成することはできません。

標準コネクター・プロパティーの設定

標準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、57 ページの『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。

コネクターを実行する前に、少なくとも以下の標準コネクター構成プロパティーを設定しておく必要があります。

- AgentTraceLevel
- ApplicationName
- ControllerStoreAndForwardMode
- ControllerTraceLevel
- DeliveryTransport

標準のプロパティーの値を変更するには、以下のステップを実行します。

1. 値を設定するフィールド内でクリックします。
2. 値を入力するか、ドロップダウン・メニューが表示された場合にはメニューから値を選択します。
3. 標準のプロパティ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 Express 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 Express 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。

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れば、コネクター・エージェント内の静的な情報やロジックを、エージェントの再コーディングや再ビルドを行わずに変更することができます。

表 2 に、コネクターのコネクター固有の構成プロパティーをリストします。各プロパティーの説明については、以降のセクションを参照してください。

表 2. コネクター固有のプロパティー

名前	指定可能な値	デフォルト値	必須
ApplicationName	`iSeriesAdapter`	なし	はい
UseDefaults	*デフォルト値*	なし	はい
MessageFileName	`BIA_iSeriesAdapter.txt`	`BIA_iSeriesAdapter.txt`	いいえ
PollQuantity	1 より大きな整数	1	いいえ

ApplicationName: 各コネクターに指定する必要がある、固有の名前です。

UseDefaults: 例えば、プログラムの入力パラメーターの一部は、値が一定です。これらを表す属性は、デフォルト値を持つように設計することができます。UseDefaults プロパティーが `true` に設定されている場合、デフォルト値が指定されていなければ、アダプターはエラーにより停止し、エラー・メッセージ VerbProcessingFailedException をスローします。UseDefaults が設定されていない場合や `false` に設定されている場合は、デフォルト値が指定されていなければ、MaxLength に指定されている長さになるようにスペースを埋め込んだストリングが属性の値としてアダプターで作成されます。

MessageFileName: エラー・メッセージ・ファイルの名前とパスです。エラー・メッセージ・ファイルがメッセージの標準位置である %CROSSWORLDS%¥connectors¥messages にない場合に指定します。メッセージ・ファイル名が完全修飾パスでない場合、メッセージ・ファイルは、HOME 環境変数、または起動パラメーター user.home で指定されたディレクトリー内にあるとみなされます。コネクター・メッセージ・ファイルが存在しない場合は、ファイル BIA_iSeriesAdapter.txt がメッセージ・ファイルとして使用されます。

PollQuantity: PollQuantity は 1 より大きな整数値で、データ・キューからポーリングする項目の数を指定します。PollQuantity の値として `n` を指定すると、メタオブジェクトを使用して構成された各キューは `n` 回ポーリングされることに注意してください。デフォルト値は 1 になります。

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**」をクリックします。子プロパティーを追加するには、親の行番号を右マウス・ボタンでクリックし、「**子を追加**」をクリックします。
2. プロパティーまたは子プロパティーの値を入力します。
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. 18 ページの『標準コネクター・プロパティーの設定』で説明したように、変更内容を保管するかまたは破棄するかを選択します。

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

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

コネクター・プロパティーの暗号化: 「コネクター固有プロパティー」ウィンドウの「**暗号化**」チェック・ボックスにチェックマークを付けると、アプリケーション固有のプロパティーを暗号化することができます。値の暗号化を解除するには、「**暗号化**」チェック・ボックスをクリックしてチェックマークを外し、「**検証**」ダイアログ・ボックスに正しい値を入力し、「**OK**」をクリックします。入力された値が正しい場合は、暗号化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: 付録 A『コネクターの*標準構成プロパティー*』の 58 ページの『構成プロパティー値の概要』にある更新メソッドの説明を参照してください。

サポートされるビジネス・オブジェクト定義の指定

コネクタ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 Express の「**サポートされているビジネス・オブジェクト**」タブを使用します。汎用ビジネス・オブジェクトと、アプリケーション固有のビジネス・オブジェクトの両方を指定する必要があり、またそれらのビジネス・オブジェクト間のマップの関連を指定することが必要です。

注: コネクターによっては、特定のビジネス・オブジェクトを、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) 追加の構成の実行がサポートされるとして指定する必要があります。

統合ブローカーとしての InterChange Server Express: 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、以下のステップを実行します。

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

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

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

サポートされるリストからビジネス・オブジェクトを削除すると、コネクター定義が変更され、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 Express」ウィンドウでは、「エージェント・サポート」を選択しても問題ないかどうかの検証は行われません。

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

ほとんどのコネクターの場合、選択可能な項目は「最大限の努力」のみです。

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

関連付けられたマップ

各コネクターは、ビジネス・オブジェクト定義とそれらに関連付けられたマップのうち現在 InterChange Server Express でアクティブであるものを示す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 Express」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、関連付けられたマップを明示的にバインドすることが必要になります。

  明示的バインディングが必要なのは、特定のサポートされるビジネス・オブジェクトに複数のマップが存在する場合のみです。InterChange Server Express は、ブ

ート時、各コネクターのサポートされるビジネス・オブジェクトのそれぞれ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。

以下のをステップを実行して、マップを明示的にバインドします。

1. 「**明示的 (Explicit)**」列で、バインドするマップのチェック・ボックスにチェックマークを付けます。
2. ビジネス・オブジェクトに関連付けるマップを選択します。
3. 「Connector Configurator Express」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 InterChange Server Express 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。

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 Express 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。これらの値は、Connector Configurator Express 内で変更できます。

ログとトレースの値を変更するには、以下のステップを実行します。

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

     注: STDOUT オプションは、Windows プラットフォームで実行しているコネクターの「**トレース/ログ・ファイル**」タブでのみ使用できます。
   - ファイルに:
     ログ・メッセージまたはトレース・メッセージを指定されたファイルに書き込みます。ファイルを指定するには、ディレクトリー・ボタン (省略符号) をクリックし、指定する格納場所に移動し、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『コネクターの標準構成プロパティー』にある ContainerManagedEvents の下の説明を参照してください。

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 Express では、構成中に選択したブローカー・モードでファイルを保管します。Connector Configurator Express のタイトル・バーには現在のブローカー・モードが常に表示されます。

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

- System Manager から Integration Component Library に *.con 拡張子付きファイルとして保管します。
- 指定したディレクトリーに保管します。
- スタンドアロン・モードで、ディレクトリー・フォルダーに *.cfg 拡張子付きファイルとして保管します。デフォルトでは、このファイルは ¥WebSphereAdapters¥bin¥Data¥App に保管されます。

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、「システム・インプリメンテーション・ガイド」を参照してください。

構成の完了

コネクターの構成ファイルを作成した後で、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 Express の使用

Connector Configurator Express 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 Express 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 Express 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>
```

コネクターの始動

コネクターは、**コネクター始動スクリプト**を使用して明示的に開始する必要があります。Windows システムでは、始動スクリプトは次のコネクターのランタイム・ディレクトリーに存在していなければなりません。*ProductDir*¥connectors¥*iSeries*。

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

i5/OS システムでは、始動スクリプトはコネクターの実行に使用する /QIBM/UserData/WBIServer44/<instance>/connectors/<ConnInstance/ に存在していなければなりません。

InterChange Server Express の場合:
/QIBM/UserData/WBIServer44/WebSphereICSName/connectors/iSeries/ start_iSeries.sh iSeries WebsphereICSName

JMS の場合: /QIBM/UserData/WBIServer44/WebSphereAdapters/connectors/iSeries/ start_iSeries.sh iSeries
dummy -c/QIBM/UserData/WBIServer44/WbSphereAdapters/connectors /iSeries/iSeriesConnector.cfg

表 3 が示すとおり、始動スクリプトの名前はオペレーティング・システム・プラットフォームにより異なります。

表 3. コネクターの始動スクリプト

オペレーティング・システム	始動スクリプト
Linux	connector_manager

表 3. コネクターの始動スクリプト (続き)

オペレーティング・システム	始動スクリプト
i5/OS	start_iSeries.sh
Windows	start_*iSeries*.bat

始動スクリプトが実行されると、デフォルトで構成ファイルが *Productdir* にあることが要求されます (下記のコマンドを参照)。この場所に構成ファイルを配置します。

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

- **Windows システムでのコネクターの開始**
  - 「**スタート**」メニューから、「**プログラム**」>「**IBM WebSphere Business Integration Express**」>「**アダプター**」>「**コネクター**」を選択します。デフォルトでは、プログラム名は「IBM WebSphere Business Integration Server Express」ですが、カスタマイズできます。あるいは、ご使用のコネクターへのデスクトップ・ショートカットを作成することもできます。
  - Windows コマンド行から、次を入力します: start_connName connName brokerName {-cconfigFile}
  - コネクターは、Windows システムの Windows サービスとして始動するように構成することができます。この場合、Windows システムがブートしたとき (自動サービスの場合)、または Windows サービス・ウィンドウを通じてサービスを始動したとき (手動サービスの場合) に、コネクターが始動します。
- **Linux システムでのコネクターの開始**:
  - コマンド行から、以下を入力します。

    connector_manager -start *connName brokerName* [-c*configFile*]

    ここで、*connName* はコネクターの名前であり、*brokerName* は統合ブローカーを表します。
  - InterChange Server Express の場合は、*brokerName* に InterChange Server Express インスタンスの名前を指定します。
- **i5/OS システムでのコネクターの開始**
  - WebSphere Business Integrations Server Express Console がインストールされている Windows システムから、「**IBM WebSphere Business Integration Server Express**」>「**Toolset Express**」>
    「**管理**」>「**コンソール**」を選択します。次に、i5/OS システム名または IP アドレスと、*JOBCTL 特殊権限を持つユーザーのプロファイルおよびパスワードを指定します。コネクターのリストからコネクターを選択して、「開始」をクリックします。
  - コンソールを使用してアダプターを自動的に開始するには、submit_adapter.sh スクリプトを使用します。アダプター・エージェントは、自動始動するようにコンソールから設定することもできます。このオプションを設定するには、アダプター・エージェントを右マウス・ボタンでクリックし、「**自動始動**」チェック・ボックスをクリックして使用可能にします。

– バッチ・モードでは、i5/OS コマンド行から CL コマンド QSH を実行し、QSHELL 環境から /QIBM/ProdData/WBIServer44/bin/submit_adapter.sh *connName WebSphereICSName pathToConnNameStartScript jobDescriptionName* を実行する必要があります。ここで、*connName* はコネクター名、*WebSphereICSName* は Interchange Server Express サーバー名 (デフォルトは QWIBDFT44)、*pathToConnNameStartScript* はコネクター始動スクリプトの絶対パス、*jobDescriptionName* は QWIBSVR44 ライブラリーで使用するジョブ記述の名前です。

– 対話モードでは、CL コマンド QSH を実行し、QSHELL 環境から */QIBM/UserData/WBIServer44/WebSphereICSName/connectors/iSeries/start_iSeries.sh iSeries WebSphereICSName* [-cConfigFile] を実行する必要があります。ここで、*connName* はコネクターの名前であり、*WebSphereICSName* は InterChange Server Express インスタンスの名前です。

コマンド行の始動オプションなどのコネクターの始動方法の詳細については、「システム*管理ガイド*」を参照してください。

コネクターの停止

コネクターを停止する方法は、コネクターが始動された方法によって異なります。

- **Windows**:
  – コネクター用の別個の「コンソール」ウィンドウを作成する始動スクリプトを起動できます。このウィンドウで、「q」と入力して Enter キーを押すと、コネクターが停止します。
  – コネクターを Windows のサービスとして始動するように構成できます。この場合、Windows システムのシャットダウン時に、コネクターは停止します。
- **i5/OS**:
  – コンソールを使用して、または QSHELL で「submit_adapter.sh」スクリプトを使用してコネクターを始動した場合は、次の 2 つの方法のうちの 1 つを使用してコネクターを停止できます。
  – WebSphere Business Integration Server Express Console がインストールされている Windows システムから、「IBM **WebSphere Business Integration Express**」>「**Toolset Express**」>「**管理**」>「**コンソール**」を選択します。次に、i5/OS システム名または IP アドレスと、*JOBCTL 特殊権限を持つユーザーのプロファイルおよびパスワードを指定します。リストから iSeries アダプターを選択して、「停止」ボタンを選択します。 CL コマンド WRKACTJOB SBS (QWIBSVR44) を使用して Server Express 製品に対するジョブを表示します。リストをスクロールして、コネクターのジョブ記述に一致するジョブ名のジョブを探します。CL コマンド WRKSMBJOB を使用してジョブを表示します。ジョブ名は QWBIISRSC です。このジョブに対してオプション 4 を選択し、F4 を押して ENDJOB コマンドのプロンプトを取得します。次に、オプション・パラメーターとして *IMMED を指定し、Enter を押します。

    注: QWBISVR44 サブシステムが終了すると、コネクターは終了します。

  – QSHELL から start_connName.sh スクリプトを使用してアダプターを始動した場合は、F3 を押してコネクターを終了します。/QIBM/ProdData/WBIServer44/bin ディレクトリーにあるスクリプト *stop_adapter.sh* を使用して、エージェントを停止することもできます。

- **Linux**:

  コネクターはバックグラウンドで実行されるので、個別のウィンドウはありません。代わりに、以下のコマンドを実行してコネクターを停止します。

  ```
  connector_manager -stop connName
  ```

  ここで、*connName* はコネクターの名前です。

複数のコネクター・インスタンスの作成

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

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

新規ディレクトリーの作成

- **Windows プラットフォームの場合**:

  ProductDir¥connectors¥connectorInstance

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

  ProductDir¥repository¥connectorInstance

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

  InterChange Server Express サーバー名を startup.bat のパラメーターとして指定できます。例: start_iseries.bat connName serverName

- **i5/OS プラットフォームの場合:**

  ```
  /QIBM/UserData/WBIServer44/WebShereICSName/connectors/connectorInstance
  ```

  ここで、connectorInstance はコネクター・インスタンスを固有に識別し、WebSphereICSName はコネクターの実行に使用する Interchange Server Express インスタンスの名前です。

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

  /QIBM/UserData/WBIServer44/WebSphereICSName/repository/connectorInstance。
  ここで、WebSphereICSName はコネクターの実行に使用する Interchange Server Express インスタンスの名前です。

- **Linux プラットフォームの場合**:
  ProductDir/connectors/connectorInstance。ここで、connectorInstance はコネクター・インスタンスを一意的に示します。コネクターにコネクター固有のメタオブジェクトがある場合、コネクター・インスタンス用のメタオブジェクトを作成する必要があります。メタオブジェクトをファイルとして保管する場合は、ProductDir/repository/connectorInstance ディレクトリーを作成して、ここにファイルを保管します。InterChange Server Express サーバー名を connector_manager のパラメーターとして指定できます。例: connector_manager -start connName WebSphereICSName [-cConfigFile]

ビジネス・オブジェクト定義の作成

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

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

   ```
   ProductDir¥repository¥initialConnectorInstance
   ```

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

コネクター定義の作成

Connector Configurator Express で、コネクター・インスタンスの構成ファイル (コネクター定義) を作成するには、次の手順を使用します。

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

始動スクリプトの作成

始動スクリプトを作成するには、以下のステップを実行します。

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

   dirname
2. この始動スクリプトを、『ビジネス・オブジェクト定義の作成』で作成したコネクター・ディレクトリーに格納します。
3. (Windows の場合のみ) 始動スクリプトのショートカットを作成します。
4. (Windows の場合のみ) 初期コネクターのショートカット・テキストをコピーし、新規コネクター・インスタンスの名前に一致するように (コマンド行で) 初期コネクターの名前を変更します。

5. (i5/OS の場合のみ) 次の情報を使用して、コネクターのジョブ記述を作成します。
   CRTDUPOBJ(QWBIISRSC) FROMLIB(QWBISVR44)OBJTYPE(*JOBD)TOLIB (QWBISVR44) NEWOBJ(newemailname)。ここで newemailname は、新規コネクターのジョブ記述に使用する 10 文字の名前です。
6. (i5/OS の場合のみ) 新規コネクターを WebSphere Business Integration Server Express Console に追加します。WebSphere Business Integration Server Express Console の詳細については、コンソールに付属のオンライン・ヘルプを参照してください。

第 4 章 コネクター用のビジネス・オブジェクトについて

この章では、iSeries ビジネス・オブジェクトの構造、コネクターがビジネス・オブジェクトを処理する方法、およびコネクターにおけるビジネス・オブジェクトの前提事項について説明します。この情報は、既存の iSeries 用ビジネス・オブジェクトを変更するためのガイドとして、または新規ビジネス・オブジェクトをインプリメントするための提案として使用してください。

本章の内容は、次のとおりです。

- 『コネクター・メタデータの定義』
- 32 ページの『RPG プログラムのビジネス・オブジェクト構造』
- 36 ページの『iSeries データ・キューのビジネス・オブジェクト構造』
- 37 ページの『ポーリングのためにメタオブジェクトを構成』
- 39 ページの『ビジネス・オブジェクト属性のプロパティーの指定』
- 40 ページの『ビジネス・オブジェクトの属性レベルのアプリケーション・テキストの指定』

Adapter for iSeries のビジネス・オブジェクトの作成を自動化する Object Discovery Agent (ODA) ユーティリティーについては、43 ページの『第 5 章 ビジネス・オブジェクトの作成および変更』を参照してください。

コネクター・メタデータの定義

iSeries コネクターはメタデータ主導型です。WebSphere Business Integration システムでは、メタデータはビジネス・オブジェクトに保管されたアプリケーション固有の情報であり、コネクターとアプリケーションの対話を支援します。メタデータ主導型のコネクターによって処理される各ビジネス・オブジェクトは、コネクター内にハードコーディングされた命令ベースではなく、ビジネス・オブジェクト定義内にエンコードされたメタデータをベースにしてサポートされています。ビジネス・オブジェクト・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ています。コネクターはメタデータ主導型であるため、新規の、または変更されたビジネス・オブジェクトを、コネクター・コードの変更を必要とせずに処理することができます。

コネクターには、サポートされるビジネス・オブジェクトの構造とアプリケーション固有情報のフォーマットに関する前提事項があります。

したがって、ビジネス・オブジェクトを作成または変更するときには、その変更内容が、コネクターが従うように設計された規則に準拠する必要があります。規則に従っていない場合は、コネクターが新規または変更済みビジネス・オブジェクトを正しく処理できません。

ビジネス・オブジェクト構造の概要

WebSphere Business Integration システムでは、ビジネス・オブジェクト定義はタイプ名、サポートされる動詞、および属性からなっています。アプリケーション・ビジネス・オブジェクトは、ビジネス・オブジェクト定義のインスタンスです。特定のアプリケーションのデータ構造および属性プロパティーが反映されます。

一部の属性は、そのものにデータが含まれているのではなく、データが含まれている子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を指しています。

WebSphere Business Integration Adapter ビジネス・オブジェクトは、フラットなものと階層のものがあります。フラット・ビジネス・オブジェクトには、単純な属性、つまり単一値 (例えば、ストリングなど) を表す属性のみが含まれています。階層ビジネス・オブジェクトには、単純属性と、値が含まれている子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列の両方が含まれています。

親ビジネス・オブジェクトの属性に単一の子ビジネス・オブジェクトが含まれているときには、カーディナリティー 1 のコンテナー・オブジェクト、つまり単一カーディナリティーの関係が生じます。この場合は、子ビジネス・オブジェクトが、レコードを 1 つしか含むことのできないコレクションを表します。属性のタイプは、子ビジネス・オブジェクトのタイプと同じです。

親ビジネス・オブジェクトの属性に子ビジネス・オブジェクトの配列が含まれているときには、カーディナリティー n のコンテナー・オブジェクト、つまり複数カーディナリティーの関係が生じます。この場合は、子ビジネス・オブジェクトが、複数のレコードを含むことのできるコレクションを表します。属性のタイプは、子ビジネス・オブジェクトの配列のタイプと同じです。

階層ビジネス・オブジェクトの属性は、単純属性である場合と、単一カーディナリティーの子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す属性である場合があります。それらのビジネス・オブジェクトにはそれぞれ、単一カーディナリティーの子ビジネス・オブジェクトやビジネス・オブジェクトの配列などが含まれます。

RPG プログラムのビジネス・オブジェクト構造

Adapter for iSeries のビジネス・オブジェクトは、フラット・ビジネス・オブジェクトです。各属性は、入力、出力、または入出力パラメーターになります。属性の 1 つは、Business Object Designer Express 用のキーになっている必要があります。

図 3. iSeries の親ビジネス・オブジェクト

また、タイプが Connection の子属性もあります。これには、i5/OS マシンへの接続に関する情報である HostName、UserName、および Password が含まれています。これらの属性はすべて必須であるため、すべての属性について `is Required` が true に設定されます。この接続ビジネス・オブジェクトは、すべての iSeries ビジネス・オブジェクトの子属性です。

図 4. iSeries の子ビジネス・オブジェクト

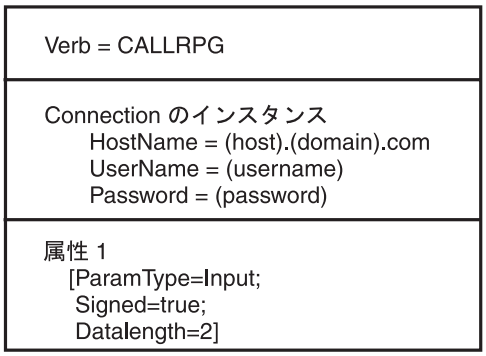


図 5. RPG ビジネス・オブジェクトの例

注: 上記の図の大括弧で囲まれている情報は、アプリケーション固有の情報を表しています。

RPG ビジネス・オブジェクトは動詞 CALLRPG で構成され、プログラム名 (IFSFile の絶対パス) は動詞の ASI として設定されます。接続関連の子属性とは別に、RPG ビジネス・オブジェクトには RPG プログラム・パラメーターに対応する

属性があります。属性の名前は、ソース・プログラムに指定された対応するパラメーターの名前と同じです。属性の MaxLength プロパティー (RPG パラメーターを表す) は、ソース・プログラムの PARM 仕様に指定された対応するパラメーター長から導出されます。パラメーターが数値である場合は、対応する小数部の長さも、ASI で `DecimalPositions=n` および `packedDec=true` のように使用されます。

アダプターを使用すると、複数インスタンスを持つ単一要求ビジネス・オブジェクトによって、PGM を複数回呼び出すことができます。図 6 に例を示します。

General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

図 6. 複数インスタンスを持つ単一要求ビジネス・オブジェクト

図 7 に接続情報を持つ親ビジネス・オブジェクトおよび PGM のパラメーター情報を持つ子ビジネス・オブジェクトを示します。接続情報は、接続ビジネス・オブジェクトに含まれ、Verb ASI は呼び出されるプログラムのパスと同じです。

General | Attributes

Business Object Level Application-specific information:

Supported Verbs:

	Name	Application-specific information
1	CALLRPG	/QSYS.LIB/PNPLIB.LIB/PRPG4.PGM
2		

図 7. 接続情報を持つ親ビジネス・オブジェクト

図 8 に、子ビジネス・オブジェクトとそのプログラムのパラメーター情報に対応する属性を示します。

General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

図 8. 子ビジネス・オブジェクトとその対応するパラメーター情報

図 9 の例のビジネス・オブジェクトは、子が複数のビジネス・オブジェクトの 2 つのインスタンスを持つことから、プログラムは 2 回実行されます。複数レコードの検索には、該当する XSD ファイルを使用してください。

⊟ Connection	Connection	
UserName	String	Rajesh
Password	String	password
HostName	String	abc.in.ibm.com
ObjectEventId	String	
⊟ MultiRecord[n]	multi_child	
⊟ [0]	multi_child	
NAME1	String	RAJ
SERIAL1	String	10
DEPT1	String	A
ObjectEventId	String	
⊟ [1]	multi_child	
NAME1	String	PRABHU
SERIAL1	String	11
DEPT1	String	B
ObjectEventId	String	
ObjectEventId	String	

図 9. 子が複数のビジネス・オブジェクトの 2 つのインスタンスを持つビジネス・オブジェクト

コネクター・ビジネス・オブジェクトの処理

コネクターは、統合ブローカー、つまり InterChange Server Express と i5/OS システムの間で、ビジネス・オブジェクトを受け渡します。

コネクターは、統合ブローカーからビジネス・オブジェクトを渡されると、以下の処理を実行します。

1. 接続関連の子属性の情報を使用して、i5/OS システムに接続します。

2. ビジネス・オブジェクトの属性に基づいて、RPG プログラムのパラメーター・リストを作成します。
3. ビジネス・オブジェクトに対応する RPG プログラムを実行します。
4. プログラムの実行結果 (成功または失敗) を戻します。

ビジネス・オブジェクトを作成するには、Business Object Designer Express ODA を使用します。ビジネス・オブジェクト定義を作成してから、必須の属性を追加します。その後で、ビジネス・オブジェクトがサポートされるようにコネクターを構成します。 Business Object Designer Express ODA の詳細については 43 ページの『第 5 章 ビジネス・オブジェクトの作成および変更』 を参照してください。

iSeries データ・キューのビジネス・オブジェクト構造

データ・キュー・ビジネス・オブジェクトの場合は、属性がデータ・キュー・フィールドを表します。これとは別に、接続関連の子属性があります。データ・キュー・フィールドの一部に AS400Structure があれば、親子などの関係が存在する場合があります。サポートされる有効な動詞は GetQueue および PutQueue です。キューに関するアプリケーション固有情報はビジネス・オブジェクト・レベルになります。値はデータ・キューの絶対 IFSFile パスです。属性の全長は、キュー内のエレメントの最大長と等しくなければなりません。この値は、iSeries マシン上にキューを作成したとき定義されます。

パラメーターのタイプは、Input、Output または InOut です。接続オブジェクトおよびそのすべての属性のいずれも必須に設定されます。iSeries ODA は、ASI を持つすべての属性をデフォルトで `ParamType=InOut` として生成します。ただし、プログラム・ロジックに対して Input または Output に変更することが妥当と確認されれば、変更できます。

データ・キュー・ビジネス・オブジェクトの例については、37 ページの図 10 を参照してください。

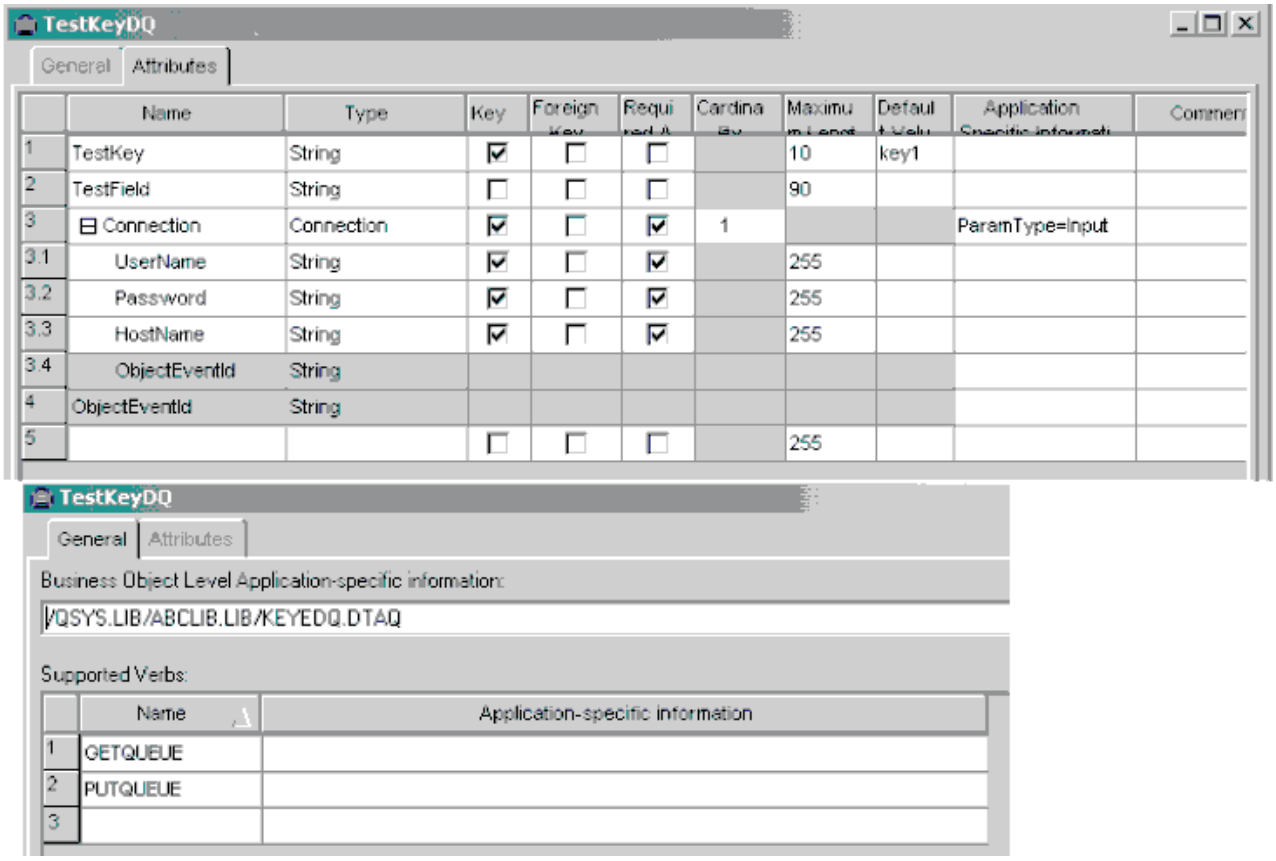


図 10. データ・キュー・ビジネス・オブジェクト

ポーリングのためにメタオブジェクトを構成

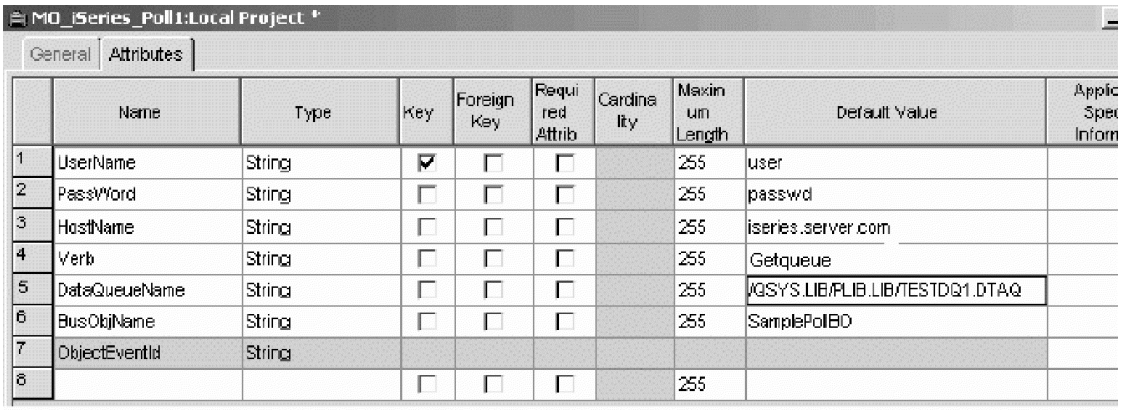
iSeries アダプターは、データベースまたはデータ・ファイルへの変更に関する完全なデータ・キュー・メッセージを受け取るすべてのデータ・キューに、メタオブジェクトを使用します。完全なデータ・キュー・メッセージを受け取るすべてのデータ・キューについて、メタオブジェクトを構成する必要があります。

メタオブジェクト名は必ず `MO_iSeries` から始まります。各メタオブジェクトには、データ・キューに関する情報が保持されます。すべてのメタオブジェクトにダミー動詞を追加する必要があります。

メタオブジェクト内の属性 (ホスト名、ユーザー名、およびパスワード) の値は、静的なデフォルト値です。これらのデフォルト属性は、コネクターを始動するとコネクター・エージェントに値がキャッシュされるので、動的に変更することはできません。別のマシンで同じデータ・キューにアクセスするには、デフォルト値を変更して iSeries アダプターのインスタンスを再始動するか、または新しいマシン情報に合わせて別のメタオブジェクトを構成する必要があります。

ビジネス・オブジェクトの動詞は該当するストリングに設定され、DataQueueName 属性はデータ・キューの IFS File パスに設定されます。BusObjName 属性には、対応するビジネス・オブジェクト (メタオブジェクトに記述されている動詞を含む) の

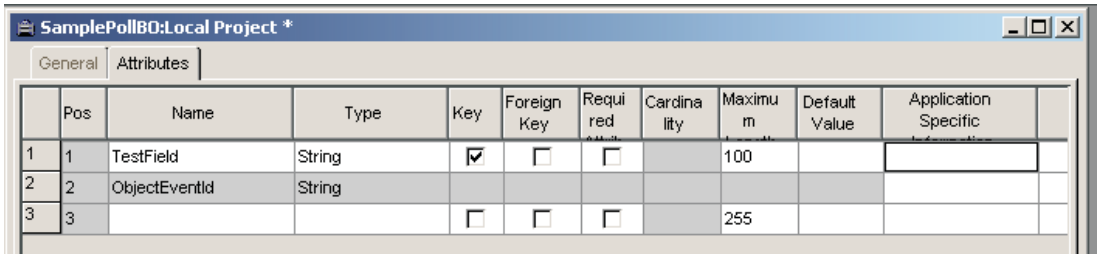
名前が含まれています。キューから読み込まれる詳細がこのビジネス・オブジェクトに書き込まれます。メタオブジェクトの属性を、下の例に示します (図 11)。対応する SamplePollBO を図 12 に示します。


MO_iSeries_Poll1:Local Project *

General | Attributes

	Name	Type	Key	Foreign Key	Required Attrib	Cardinality	Maximum Length	Default Value	Application Specific Information
1	UserName	String	☑	☐	☐		255	user	
2	PassWord	String	☐	☐	☐		255	passwd	
3	HostName	String	☐	☐	☐		255	iseries.server.com	
4	Verb	String	☐	☐	☐		255	Getqueue	
5	DataQueueName	String	☐	☐	☐		255	/QSYS.LIB/PLIB.LIB/TESTDQ1.DTAQ	
6	BusObjName	String	☐	☐	☐		255	SamplePollBO	
7	ObjectEventId	String							
8			☐	☐	☐		255		

図 11. 順次データ・キュー用のポーリング・ビジネス・オブジェクトの例

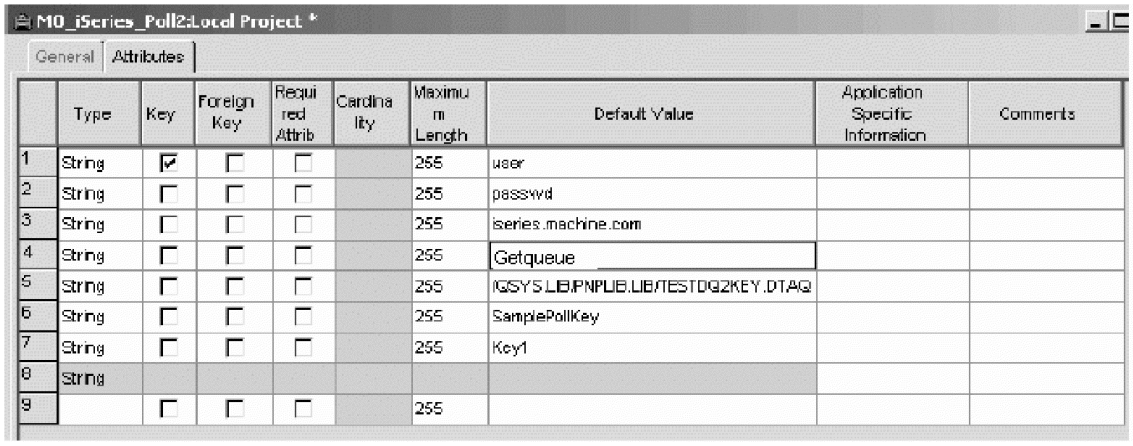

SamplePollBO:Local Project *

General | Attributes

	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum	Default Value	Application Specific
1	1	TestField	String	☑	☐	☐		100		
2	2	ObjectEventId	String							
3	3			☐	☐	☐		255		

図 12. 対応する SamplePollBO

キー付きデータ・キューの場合は、図 13 に示すようにメタオブジェクトに「key」というキー属性が追加されます。ポーリング呼び出しでは、キー情報を使用して、キー付きデータ・キューから対応するメッセージを取得します。


MO_iSeries_Poll2:Local Project *

General | Attributes

	Type	Key	Foreign Key	Required Attrib	Cardinality	Maximum Length	Default Value	Application Specific Information	Comments
1	String	☑	☐	☐		255	user		
2	String	☐	☐	☐		255	passwd		
3	String	☐	☐	☐		255	iseries.machine.com		
4	String	☐	☐	☐		255	Getqueue		
5	String	☐	☐	☐		255	/QSYS.LIB/PNPLIB.LIB/TESTDQ2KEY.DTAQ		
6	String	☐	☐	☐		255	SamplePollKey		
7	String	☐	☐	☐		255	Key1		
8	String								
9		☐	☐	☐		255			

図 13. キー付きデータ・キュー用のポーリング・ビジネス・オブジェクトの例

図 14 に、対応するキー付きデータ・キュー用のポーリング・ビジネス・オブジェクトの例を示します。

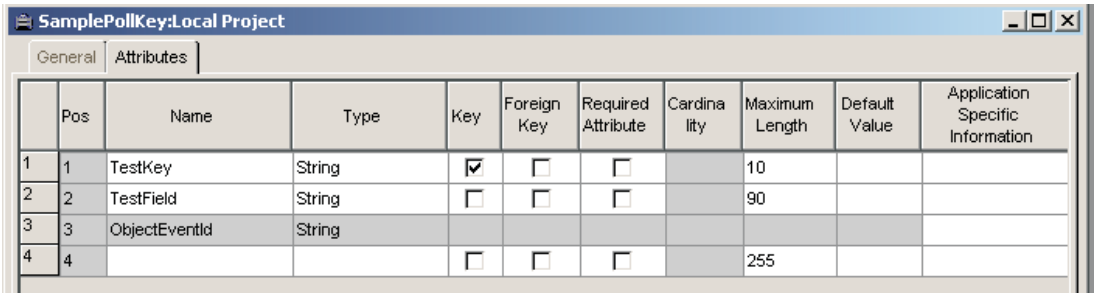


図 14. キー付きデータ・キュー用のポーリング・ビジネス・オブジェクトの例

ビジネス・オブジェクト属性のプロパティーの指定

iSeries コネクターには、ビジネス・オブジェクト属性に設定できるさまざまなプロパティーがあります。このセクションでは、コネクターがこれらのプロパティーのいくつかを解釈する方法と、ビジネス・オブジェクトを変更する際にそのプロパティーをセットする方法を解説します。

次の表に、単純属性のプロパティーを示します。

表 4. ビジネス・オブジェクト属性のプロパティー

属性	**説明**
Name	属性の固有の名前
Type	単純な属性はすべてストリング・タイプである必要があります。
MaxLength	属性値の長さが、属性に指定された MaxLength より長く、属性が入力パラメーターを表す場合は、値が Maxlength 値までトリムされます。値の長さが MaxLength 未満である場合は、スペースが埋め込まれます。
IsKey	未使用
IsForeignKey	未使用
Isrequired	すべての入力パラメーターについてこの属性を `true` に設定する必要があります。
AppSpecInfo	`ParamType=<value>:Offset=<value>: Signed=<True/False>:DataLength= <value>:PackedDec=<True/False>: ZonedDec=<True/False>: DecimalPositions=<value>`
DefaultValue	属性に設定された場合、入力パラメーターに設定されていないときには、コネクターがこの値を使用します。

ビジネス・オブジェクトの属性レベルのアプリケーション・テキストの指定

以下に示す情報は、ビジネス・オブジェクトの属性レベルのアプリケーション・テキストの一部です。

表 5. ビジネス・オブジェクト属性

プロパティー	値	説明
ParamType	Input/Output/Inout	属性が表すパラメーターのタイプを示します。
Offset	任意の整数値	バイト配列内でパラメーター値の開始位置からのオフセットを示します。
Signed	true/false	データ型が integer/short/long の場合に、符号付きかどうかを示します。このプロパティーが設定されていない場合、値は符号なしと見なされます。
DataLength	任意の整数値	データ型が integer、short、または long の場合に適用されます。データ型が符号の有無を指定できるものである場合に、データ長の識別に使用されます。このプロパティーが設定されていない場合、デフォルト値の 4 が使用されます。
DecimalPositions	任意の整数値	データ型がゾーン 10 進数またはパック 10 進数である場合に適用されます。このプロパティーの値は、小数部の桁数を表します。
PackedDec	true/false	true に設定されている場合、属性はパック 10 進数を表します。
ZonedDec	true/false	true に設定されている場合、属性はゾーン 10 進数を表します。

iSeries 用または i5/OS 用の Toolbox からのデータ変換

iSeries 用または AS/400 用の Toolbox には、データ変換クラスが含まれています。次の表に、iSeries と AS/400 のデータ型とそれに対応する IBM WebSphere Business Integration のデータ型の組み合わせを、使用するデータ変換クラスと共に示します。

表 6. 変換されるデータ型とデータ変換クラス

iSeries/AS400 のデータ型	IBM WBI のデータ型	データ変換クラス
i5/OS 形式の符号付き 2 バイト数値	Integer、アプリケーション固有の情報 - Signed=true; DataLength=2	AS400Bin2

表 6. 変換されるデータ型とデータ変換クラス (続き)

iSeries/AS400 のデータ型	**IBM WBI のデータ型**	**データ変換クラス**
i5/OS 形式の符号付き 4 バイト数値	Integer、アプリケーション固有の情報 - Signed=true; DataLength=4	AS400Bin4
i5/OS 形式の符号付き 2 バイト浮動小数点数	Float	AS400Float4
i5/OS 形式の符号付き 4 バイト浮動小数点数	Double	AS400Float8
i5/OS 形式の符号なし 2 バイト数値	Integer、アプリケーション固有の情報 - Signed=false; DataLength=2	AS400UnsignedBin2
i5/OS 形式の符号なし 4 バイト数値	Integer、アプリケーション固有の情報 - Signed=false; DataLength=4	AS400UnsignedBin4
i5/OS 形式のパック 10 進数	String (MaxLength 属性プロパティーに桁数を指定する必要があります)、アプリケーション固有の情報 - DecimalPositions=<小数部の桁数>; PackedDec=true	AS400PackedDecimal
i5/OS 形式のゾーン 10 進数	String (MaxLength 属性プロパティーに桁数を指定する必要があります)、アプリケーション固有の情報 - DecimalPositions=<小数部の桁数>; ZonedDec=true	AS400ZonedDecimal
文字データ	String (MaxLength に文字データの最大長を指定します)	AS400Text
日付データ	String (MaxLength に日付データの最大長を指定します)	AS400Text

第 5 章 ビジネス・オブジェクトの作成および変更

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

本章の内容は、次のとおりです。

ODA for iSeries の概要

Object Discovery Agent (ODA) を使用すると、ビジネス・オブジェクト定義を生成することができます。ビジネス・オブジェクト定義は、ビジネス・オブジェクトのテンプレートです。ODA は、指定されたアプリケーション・オブジェクトを検査し、ビジネス・オブジェクト属性に対応するオブジェクトのエレメントを「発見」し、ビジネス・オブジェクト定義を生成して情報を表します。Business Object Designer Express は、Object Discovery Agent へアクセスして対話式で操作するためのグラフィカル・インターフェースを提供します。

ODA for iSeries は、iSeries システム上の RPGLE プログラムおよびデータ・キュー・オブジェクトへアクセスするためのビジネス・オブジェクト定義を生成します。Business Object Designer Express のウィザードによって、これらの定義の作成プロセスが自動化されます。ODA を使用して、ビジネス・オブジェクトおよび Connector Configurator Express を作成し、それらをサポートするようにコネクターを構成します。

ビジネス・オブジェクト定義の生成

このセクションでは、Business Object Designer Express で iSeries ODA を使用して、ビジネス・オブジェクト定義を生成する方法について説明します。Business Object Designer Express の起動および使用法については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。

iSeries ODA の始動

次のいずれかのスクリプトを使用して、iSeries ODA を始動できます。

- Windows - `start_iSeriesODA.bat`

  注: また、インストーラーが Windows 環境用に自動的に作成するショートカットを使用して、iSeries ODA を始動することもできます。

- Linux - `start_iSeriesODA.sh`
- i5/OS - `start_iSeriesODA.sh`

Business Object Designer Express を使用して、iSeries ODA を選択、構成、および実行します。Business Object Designer Express は、個々のスクリプト・ファイルまたはバッチ・ファイルの AGENTNAME 変数に指定されている名前に基づいて、各 ODA を探し出します。

Business Object Designer Express の実行

Business Object Designer Express は、ODA を使用してビジネス・オブジェクト定義を生成するためのステップを順に進めていくことができるウィザードを提供します。

エージェントの選択

最初に、ODA エージェントを選択する必要があります。

1. Business Object Designer Express を開きます。
2. 「**ファイル**」 > 「**ODA を使用して新規作成**」をクリックします。「ビジネス・オブジェクト・ウィザード - ステップ 1/6 - エージェントの選択」ウィンドウが開きます。
3. 「検索されたエージェント」リストから「ODA/AGENTNAME」(start_iSeriesODA スクリプト内) を選択し、「次へ」をクリックします。(希望のエージェントがリストされていない場合は、「エージェントの検索」をクリックしてください。)

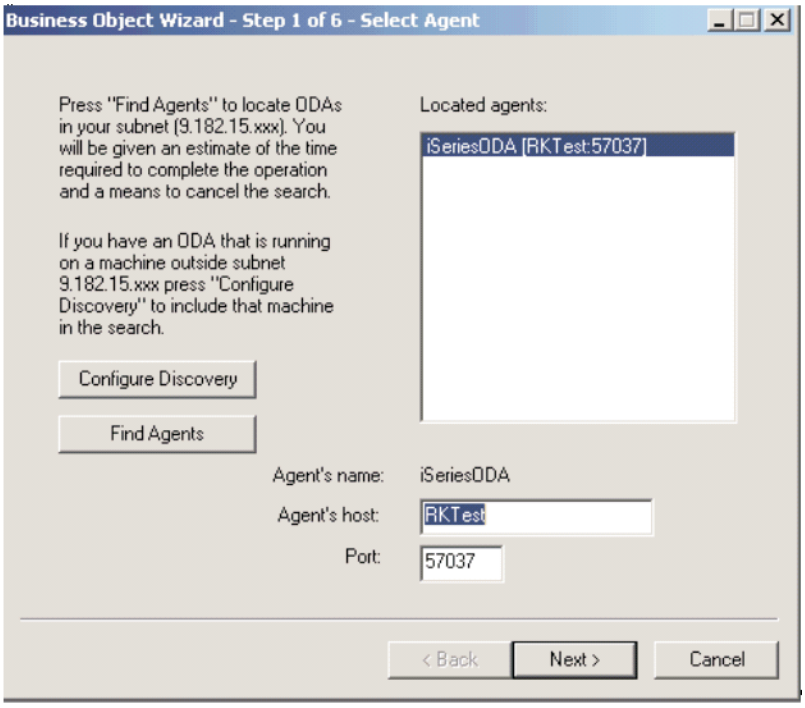


図 15. 「エージェントの選択 (Select Agent)」ウィンドウ

エージェントの構成

「エージェントの選択」ウィンドウで「次へ」をクリックすると、「ビジネス・オブジェクト・ウィザード - ステップ 2/6 - エージェントの構成」ウィンドウが開きます。

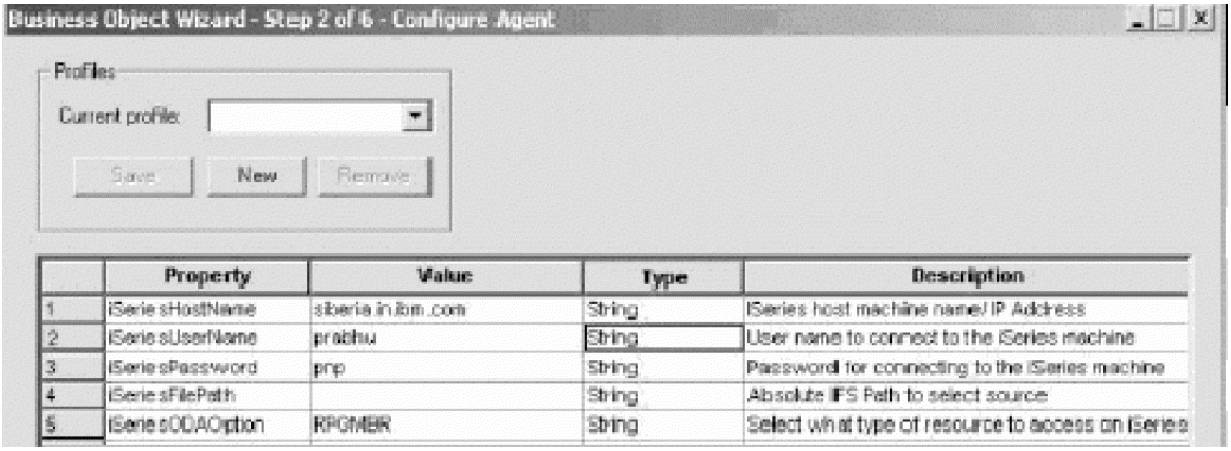


図 16. 「エージェントの構成 (Configure Agent)」ウィンドウ

この画面で設定するプロパティーについては、表 5 で説明しています。この画面で入力するすべての値をプロファイルに保管できます。次に ODA を実行する際は、プロパティー・データを再入力するのではなく、ドロップダウン・メニューからプロファイルを選択して、保管した値を再利用します。指定した値のセットごとに、複数のプロファイルを保管できます。

表 7. 「エージェントの構成」プロパティー

プロパティー名	デフォルト値	タイプ	説明
iSeriesHostName		String	(必須) iSeries ホスト・マシン名
iSeriesUserName		String	(必須) iSeries マシンへの接続に使用されるユーザー名
iSeriesPassword		String	(必須) iSeries マシンへの接続に使用されるパスワード

表 7. 「エージェントの構成」プロパティー (続き)

プロパティー名	デフォルト値	タイプ	説明
iSeriesFilePath	/QSYS.LIB/	String	選択ソースへの絶対 IFS パス。ただし、正しいパスを入力してオブジェクトを検索する必要があります。例えば、RPG ソース・メンバーの場合は、「/QSYS.LIB/PNPLIB LIB/QRPGSRC.FILE」と、またはデータ・キューの場合は「/QSYS.LIB/ PNPLIB/」と入力します。
iSeriesODAOption		String	iSeries 上でアクセスするリソースのタイプ。現在、RPGMBR と DTAQ の 2 つのオプションがあります。この選択に基づいて、対応するビジネス・オブジェクト定義を作成するためにソース・ファイルにアクセスします。
TraceFileName	なし	String	トレース・メッセージ・ファイルの名前
TraceLevel	5	Integer	(必須) エージェントのトレース・レベル (0 から 5)。
MessageFile	なし	String	(必須) ODA が表示するすべてのメッセージが含まれているメッセージ・ファイルの名前。メッセージ・ファイルの名前を正しく指定しないと、ODA はメッセージなしで実行されます。

ODA で新規プロファイルを作成するときには、「プロファイル」グループ・ボックスの「新規」ボタンと「保管」ボタンを使用します。ODA をもう一度使用するときは、既存プロファイルを選択できます。 45 ページの表 7 に定義されているように、各プロパティーの値を入力します。

必要フィールドがブランクのまま残されていたり、エラーがあると (ユーザー名が無効など)、対応するエラー・メッセージがポップアップ表示されます。

注: プロファイルを使用すると、プロパティー値が自動的に入力されます。ただし、これらの値は必要に応じて変更できます。また、新しい値を保管することもできます。

ビジネス・オブジェクトの選択

図 17 に示されている「ビジネス・オブジェクト・ウィザード - ステップ 3/6 - ソースの選択」ウィンドウが開きます。

この画面には、RPGLE ソース・ファイルの *.MBR またはデータ・キューの *.DTAQ ファイルがリストされ、ユーザーはファイルの名前を選択できます。ファイル・タイプは iSeriesODAOption エージェントのプロパティーによって決まります。IFS ディレクトリーは展開可能なツリーのノードとして表され、ソース名 (MBR および DTAQ) はリーフ・ノードとして表されます。ユーザーは、同じ IFS ディレクトリーまたは別の IFS ディレクトリーから複数のソース (リーフ・ノードのみ) を選択できます。この画面を使用して、ビジネス・オブジェクト定義を生成する任意の数のソース・ファイルを選択します。

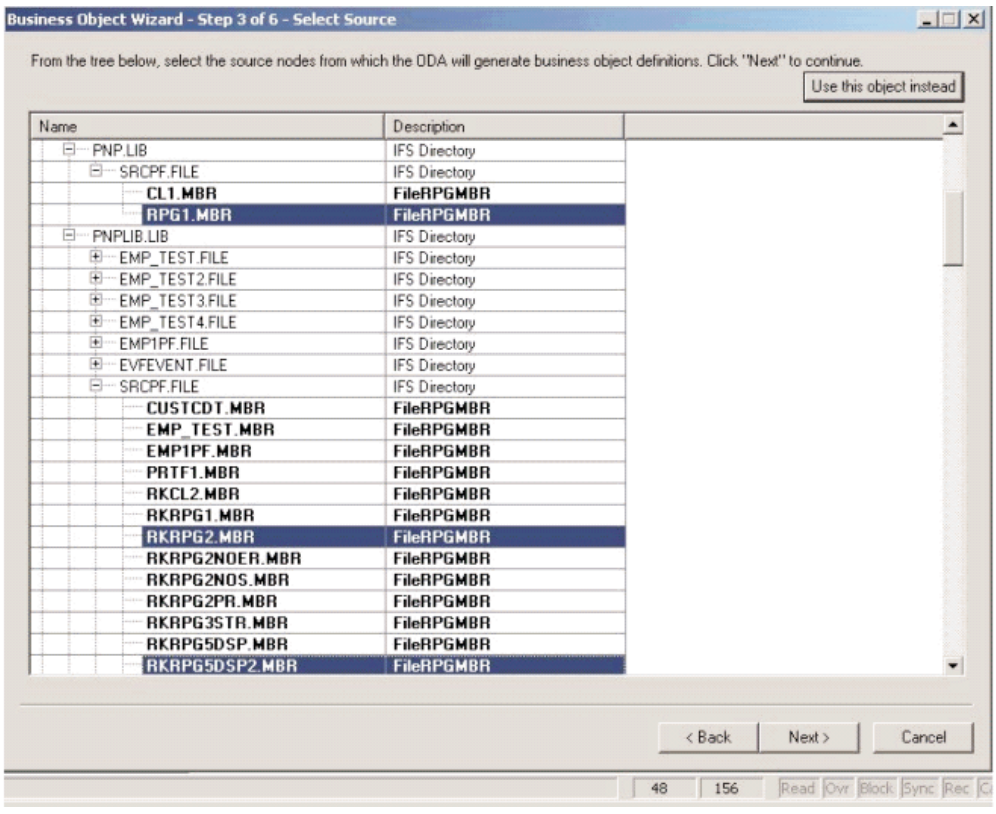


図 17. 「ソースの選択 (Select Source)」ウィンドウ

1. 必要に応じて、ノードを展開して、サブノードのリストを表示します。
2. 使用するソース・ファイルを選択します。「次へ」をクリックします。
3. 複数のノードを選択するには、ツリー構造に関する詳細情報が記載されている「*ビジネス・オブジェクト開発ガイド*」を参照してください。

オブジェクト選択の確認

「ビジネス・オブジェクト・ウィザード - ステップ 4/6 - ビジネス・オブジェクト定義のソース・ノードの確認 」ウィンドウが開きます。この画面に、選択したオブジェクトが表示されます。

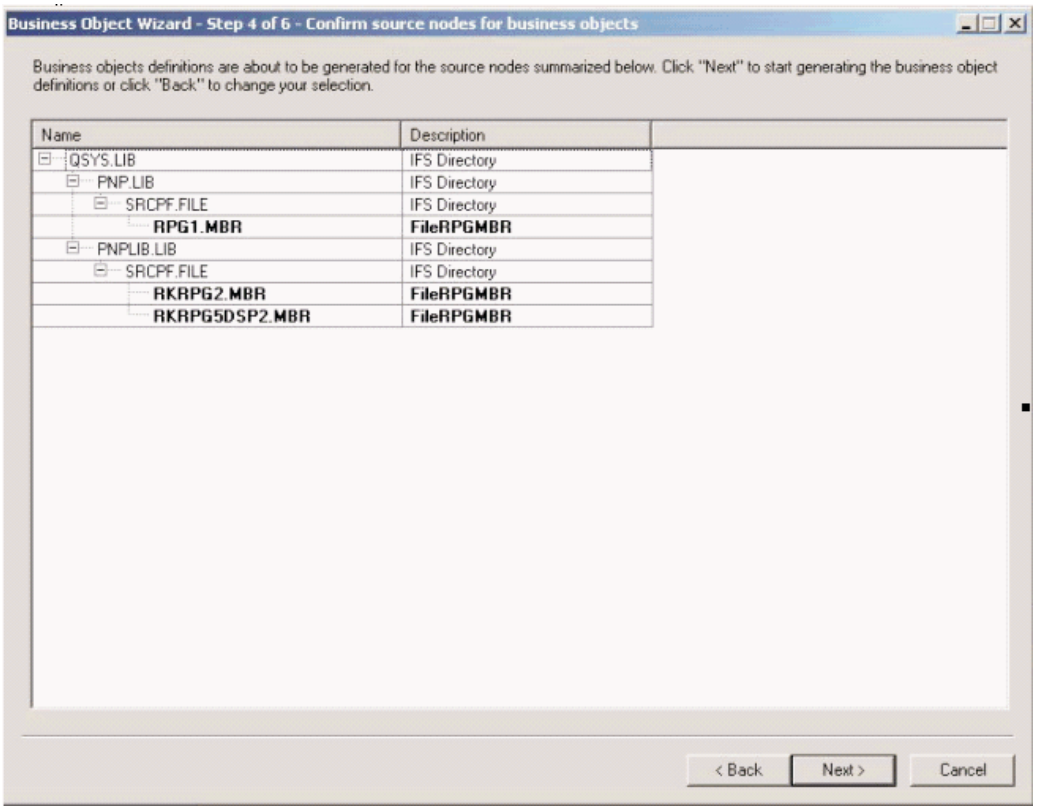


図 18. 「ビジネス・オブジェクトのソース・ノードの確認 (Confirm source nodes for business objects)」ウィンドウ

「戻る」をクリックして変更するか、または「次へ」をクリックしてリストが正しいことを確認します。「ビジネス・オブジェクト・ウィザード - ステップ 5/6 - ビジネス・オブジェクトの生成中...」ウィンドウが開きます。ウィンドウには、ビジネス・オブジェクトが生成中であることを示すメッセージが表示されます。

ビジネス・オブジェクトの生成

ノード・ソースを確認したら、iSeries ODA によってビジネス・オブジェクトが生成されます。「ビジネス・オブジェクト・ウィザード - ステップ 6/6 - ビジネス・オブジェクト定義の保管...」ウィンドウが開きます。

1. ビジネス・オブジェクト定義のコピーを別のファイルに「保管」するウィンドウにチェックマークをつけるか、別のウィンドウに新しいビジネス・オブジェクト定義を「開く」にチェックマークを付けます。後者を選択した場合は、Business Object Designer Express が起動し、そのアプリケーションでビジネス・オブジェクトが開きます。
2. 操作を完了して ODA を閉じるには、「ODA をシャットダウン」にチェックマークを付けて、「完了」をクリックします。

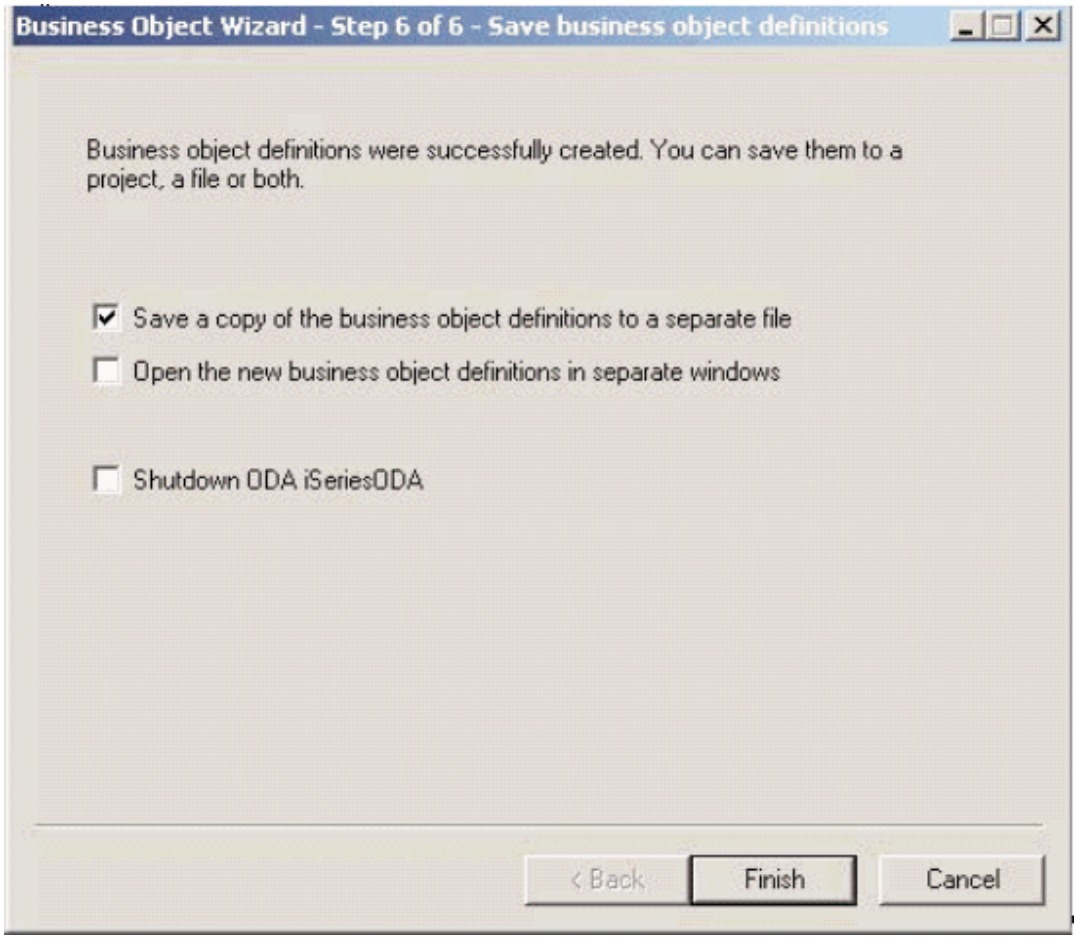


図 19. 「ビジネス・オブジェクト定義の保管 (Save business object definitions)」ウィンドウ

ビジネス・オブジェクト情報の指定

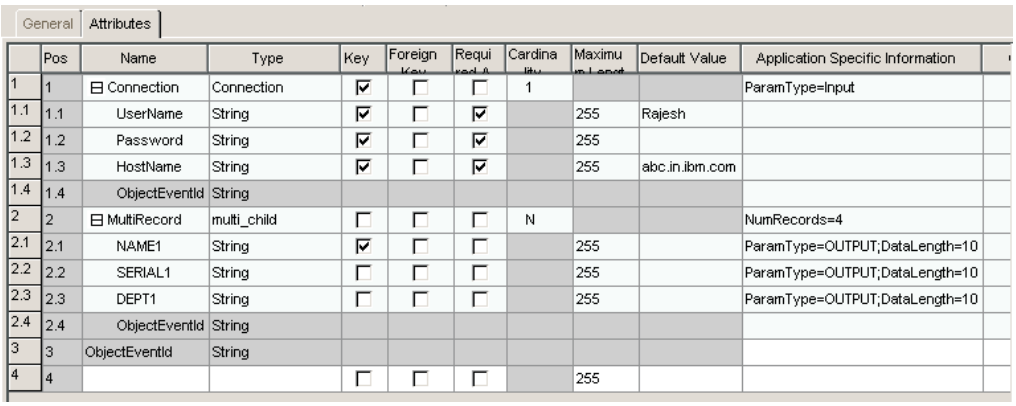
ビジネス・オブジェクトを作成したら、ビジネス・オブジェクト・レベルの ASI および属性レベルの ASI を指定できます。

このセクションでは、Business Object Designer Express で ODA を使用して、この情報を指定する方法について説明します。情報のカテゴリーの詳細、および iSeries コネクターでのビジネス・オブジェクト構造にとっての役割の詳細については、31 ページの『第 4 章 コネクター用のビジネス・オブジェクトについて』を参照してください。

属性レベル ASI の指定

Business Object Designer Express は、ビジネス・オブジェクトの属性を表示します。iSeries コネクターでの属性レベル ASI の詳細については、40 ページの『ビジネス・オブジェクトの属性レベルのアプリケーション・テキストの指定』を参照してください。

属性は、「位置」列の数値で定義されているように、ビジネス・オブジェクト構造での出現順に「属性」タブにリストされます。



	Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	⊟ Connection	Connection	☑	☐	☐	1			ParamType=Input
1.1	1.1	UserName	String	☑	☐	☑		255	Rajesh	
1.2	1.2	Password	String	☑	☐	☑		255		
1.3	1.3	HostName	String	☑	☐	☑		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	⊟ MultiRecord	multi_child	☐	☐	☐	N			NumRecords=4
2.1	2.1	NAME1	String	☑	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	☐	☐	☐		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			☐	☐	☐		255		

図 20. 属性 ASI の設定

ウィンドウには、各属性の名前、タイプ、および ASI 情報が示されます。このウィンドウでは、ODA がまだキーを指定していないビジネス・オブジェクトごとに、キーを指定する必要があります (キーは、Business Object Designer Express がビジネス・オブジェクトを妥当性検査および保管するために必要です)。

ビジネス・オブジェクトでは、セキュリティー上、パスワードはデフォルト値として設定されず、トレースもされません。順次データ・キュー用に生成されたビジネス・オブジェクトは、そのデータ長に対応する 1 つの属性を持っています。キー付きデータ・キューの場合、属性が 2 つあり、最初の属性がキーに対応し、もう 1 つの属性は残りの長さ、(つまりデータ長 - キー長) になります。属性が ASI DataLength = X および DecimalPositions=Y のビジネス・オブジェクトでは、X より大きな桁数 (小数部の桁数が Y より大きいため) の属性値でビジネス・オブジェクトを送信すると、コネクターがこの値を小数部の長さが Y になるように切り捨てることによって、データ長を X に維持し、ビジネス・オブジェクトを正常に処理します。整数部の長さが X-Y を超えると、コネクターはエラーをスローします。例えば、属性の ASI が PackedDecimal=True;Datalength=10;DecimalPositions=2 なら、値 112345678、12345678.1、および 12345678.12 は受け入れられますが、12345678.123 では小数部の上限 2 を超える余分の小数部の桁が切り捨てられ、値は 12345678.12 とされます。これに対してトレース・メッセージ「切り捨てられたストリング値 :<12345678.12> (DecimalPositions=2)(Truncated String Value:<12345678.12> for DecimalPositions=2)」が発行されます。値 123456789.12 は、エラー「長さが無効です (Length is not valid)」をログに記録します。

また、このウィンドウでは、必要に応じて子オブジェクト・キーを設定でき、次の情報を指定できます。

- コネクターがビジネス・オブジェクトを処理するために属性が必要かどうか。必要な場合は、「必要」チェック・ボックスをクリックします。
- 属性の最大長を、「最大長」列に表示される値と異なる値にするかどうか。
- 属性にデフォルト値を設定するかどうか。設定する場合は、「デフォルト」列に値を入力します。

ビジネス・オブジェクト・レベル ASI の指定

属性レベルの ASI を指定したら、ビジネス・オブジェクト・レベルの ASI を表示および変更できます。ビジネス・オブジェクト・レベル ASI の詳細については、40 ページの『ビジネス・オブジェクトの属性レベルのアプリケーション・テキストの指定』を参照してください。

ビジネス・オブジェクト・レベルの ASI は、「一般」タブにリストされます。「ビジネス・オブジェクト・レベル・アプリケーション固有の情報」フィールドに表示される ASI 値には、このビジネス・オブジェクトを表すプロキシー・クラスの名前が含まれます。コネクターはこの情報を使用して、プロキシー・クラスをビジネス・オブジェクトにマップします。また、サーバー・サイドのビジネス・オブジェクトの場合は (コネクターがサーバーとしても実行される場合)、コネクターがこの情報を使用して、インプリメンテーション・クラスをビジネス・オブジェクトにマップします。

この画面には、ビジネス・オブジェクトがサポートするすべての動詞がリストされ、各動詞ごとに ASI が提供されます。この画面では、ビジネス・オブジェクトの ASI およびサポートする動詞を変更できます。

注: 動詞レベル ASI で述べられる PGM のパスを確認して変更する必要があります。ODA は、選択された .MBR IFS ファイルを有効な RPGLE ソース・ファイルと想定します。RPGLE ソース・ファイルではない場合、ODA は入力パラメーターなしでビジネス・オブジェクトを生成します。

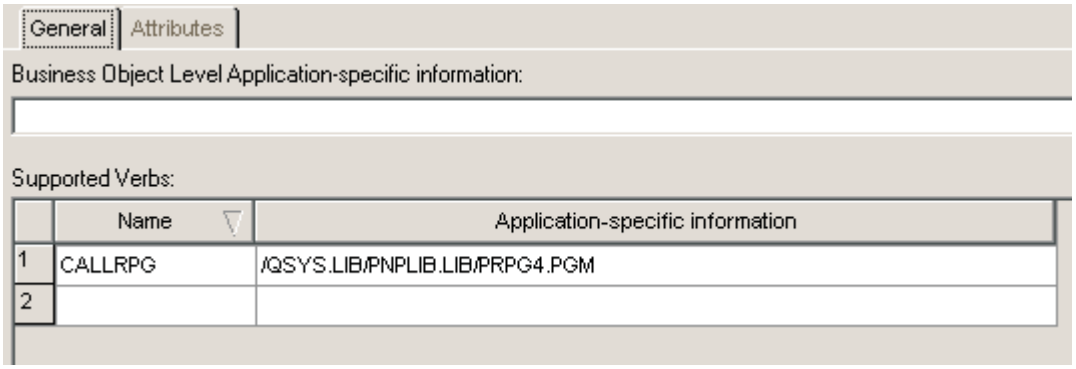


図 21. ビジネス・オブジェクト・レベルの ASI を設定

ビジネス・オブジェクトのアップロード

新規作成したビジネス・オブジェクト定義ファイルは、作成が終わったら、統合ブローカー InterChange Server Express にアップロードする必要があります。

ビジネス・オブジェクト定義ファイルをローカル・マシンに保管して、それらをサーバー上のリポジトリーにアップロードする必要がある場合には、「システム・インプリメンテーション・ガイド」を参照してください。

第 6 章 トラブルシューティングおよびエラー処理

この章では、Adapter for iSeries がエラーを処理する方法について説明します。アダプターは、ロギング・メッセージとトレース・メッセージを生成します。この章には次のセクションがあります。

本章の内容は、次のとおりです。

- 『エラー処理』
- 『ロギング』
- 『トレース・メッセージ』

エラー処理

コネクターによって生成されたすべてのエラー・メッセージは、BIA_ISERIESAdapter.txt という名前のメッセージ・ファイルに格納されます。(このファイルの名前は、コネクター構成標準プロパティー LogFileName によって決定されます。)

すべてのエラーは VerbProcessingFailedException に変換されます。

ロギング

アダプターは、トレース・レベルに関係なく、処理中に異常条件が発生すると必ず、エラー・メッセージをログに記録します。また、そのようなエラーが発生した場合、コネクターは、処理に失敗したビジネス・オブジェクトが、受信時点でどのような状態であったかを示すテキスト表現も出力します。テキストを iSeries アダプター・ログ・ファイルに書き込みます。このファイルの名前は、コネクター・プロパティー LogFileName に対応します。メッセージには、条件の詳細記述と結果が記載され、デバッグ（ビジネス・オブジェクト・ダンプ、スタック・トレースなど）に役立つ追加情報が含まれていることもあります (例外の場合)。

iSeries アダプターのメッセージ ID の範囲は 93000 から 94000 です。

トレース・メッセージ

トレースは、コネクターの動作を細かく追跡するためにオンにすることができるオプションのデバッグ・フィーチャーです。トレース・メッセージは構成可能で、動的に変更できます。必要な詳細に応じて、さまざまなレベルを設定できます。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。

また、ファイルへ書き込むようにトレースを構成することもできます。次の表では、iSeries コネクターが各トレース・レベルで出力するトレース・メッセージのタイプについて説明します。すべてのトレース・メッセージが、コネクター・プロパティー TraceFileName で指定されたファイルに書き込まれます。これらのメッセージは、IBM WebSphere Business Integration Adapter アーキテクチャーによって出力されたトレース・メッセージに追加されます。トレース・メッセージの構成の詳細

については、『コネクターの構成』にあるコネクター構成プロパティーを参照してください。トレースの詳細 (使用可能にして設定する方法を含む) については、「コネクター開発ガイド」を参照してください。

表 6 に、コネクターのトレース・メッセージ・レベルの推奨される内容をリストします。

表 8. トレース・メッセージの内容レベルの説明

トレース・レベル	トレース・メッセージ
レベル 0	コネクターのバージョンを識別するトレース・メッセージが必要な場合は、このトレース・レベルを使用します。このレベルでは、その他のトレースは実行されません。
レベル 1	なし
レベル 2	次のことを行うトレース・メッセージが必要な場合は、このトレース・レベルを使用します。 • コネクターが処理する各オブジェクトに使用される BO ハンドラーを識別する • ビジネス・オブジェクトが統合ブローカーにポストされるたびにログに記録する • 要求ビジネス・オブジェクトが受信されるたびにそれを示すトレース・メッセージ
レベル 3	なし
レベル 4	次のことを行うトレース・メッセージが必要な場合は、このトレース・レベルを使用します。 • アプリケーション固有の情報を示す。この例として、ビジネス・オブジェクト内のアプリケーション固有の情報フィールドを処理するメソッドによって戻される値があります。 • コネクターが関数を実行または終了する時を示す。これらのメッセージは、コネクターの処理フローをトレースするのに役立ちます。 • スレッド固有処理を記録する。例えば、コネクターが複数のスレッドを作成する場合は、メッセージによって各新規スレッドの作成がログに記録されます。

表 8. トレース・メッセージの内容レベルの説明 (続き)

トレース・レベル	トレース・メッセージ
レベル 5	次のことを行うトレース・メッセージが必要な場合は、このトレース・レベルを使用します。 • コネクターの初期設定を示す。このタイプのメッセージには、例えば、ブローカーから検索された各コネクター・コンフィギュレーター・プロパティーの値などが含まれます。 • コネクターが実行中に作成する各スレッドの状況を詳述する。 • アプリケーションで実行されたステートメントを表す。コネクター・ログ・ファイルには、ターゲット・アプリケーションで実行されたすべてのステートメントと、置換された変数の値 (ある場合のみ) が記録されます。 • ビジネス・オブジェクト・ダンプを記録する。コネクターは、処理開始前およびオブジェクト処理の終了後に、ビジネス・オブジェクトのテキスト表現を出力します。処理前には、コネクターがコラボレーションから受け取るオブジェクトを表示し、処理後には、コネクターがコラボレーションに戻すオブジェクトを表示します。

付録. 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 Server Express アダプターのコネクター・コンポーネントの標準構成プロパティーについて説明します。この情報は、InterChange Server Express を対象とします。

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

新規プロパティー

本リリースには、次の標準プロパティーが追加されました。

- AdapterHelpName
- ControllerEventSequencing
- jms.ListenerConcurrency
- jms.TransportOptimized
- TivoliTransactionMonitorPerformance

標準コネクター・プロパティーの概要

コネクターには 2 つのタイプの構成プロパティーがあります。

- 標準構成プロパティー (フレームワークで使用される)
- アプリケーション、またはコネクター固有の構成プロパティー (エージェントで使用される)

これらのプロパティーで、アダプター・フレームワークとエージェントの実行時の動作が決まります。

このセクションでは、Connector Configurator Express の始動方法と、すべてのプロパティーに共通の特性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 Express の始動

コネクター・プロパティーの構成は Connector Configurator Express から行います。Connector Configurator Express には、System Manager からアクセスします。Connector Configurator Express の使用法の詳細については、本書の Connector Configurator Express に関するセクションを参照してください。

Connector Configurator Express と System Manager は、Windows システム上でのみ動作します。コネクターを Linux 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。

Linux 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、Linux の統合ブローカーに接続してから、コネクター用の Connector Configurator Express を開く必要があります。

構成プロパティー値の概要

コネクターは、以下の順序に従ってプロパティーの値を決定します。

1. デフォルト
2. InterChange Server Express 統合ブローカーのリポジトリー
3. ローカル構成ファイル
4. コマンド行

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。STRING プロパティー・タイプの長さの制限はありません。INTEGER タイプの長さは、アダプターが実行しているサーバーによって決まります。

コネクターは、始動時に構成値を取得します。実行時セッション中に 1 つ以上のコネクター・プロパティーの値を変更する場合は、プロパティーの更新メソッドによって、変更を有効にする方法が決定されます。

プロパティーの更新特性、つまり、コネクター・プロパティーの変更を有効にする方法および条件は、プロパティーの性質により異なります。

標準コネクター・プロパティーには、以下の 4 種類の更新メソッドがあります。

- **動的**
  新規の値は、変更内容が System Manager に保管されると、即時に有効になります。ただし、コネクターがスタンドアロン・モードの場合です (System Manager とは独立)。
- **エージェント再始動 (InterChange Server Express のみ)**
  新規の値は、コネクター・エージェントを停止して再始動した後にのみ、有効になります。
- **コンポーネント再始動**
  新規の値は、System Manager でコネクターを停止して再始動した後でのみ、有効になります。エージェントまたはサーバー・プロセスは停止して再始動する必要はありません。
- **システム再始動**
  新規の値は、コネクター・エージェントおよびサーバーを停止して再始動した後にのみ、有効になります。

特定のプロパティーの更新方法を確認するには、「Connector Configurator Express」ウィンドウ内の「**更新メソッド**」列を参照するか、59 ページの表 9 の「更新メソッド」列を参照してください。

標準プロパティーを常駐できる場所は、次の 3 箇所あります。一部のプロパティーは、複数の場所に常駐できます。

- **ReposController**
  プロパティーはコネクター・コントローラーに常駐し、そこでのみ有効です。エージェント・サイドで値を変更した場合、コントローラーには影響しません。
- **ReposAgent**
  プロパティーはエージェントに常駐し、そこでのみ有効です。プロパティーによっては、ローカル構成でこの値をオーバーライドできます。

- **LocalConfig**
  プロパティーはコネクターの構成ファイルに常駐し、構成ファイルを介してのみ動作できます。コントローラーはプロパティーの値を変更できず、システムが再配置されてコントローラーが明示的に更新されない限り、構成ファイルに行なわれた変更を認識しません。

標準プロパティーの早見表

表 9 は、標準コネクター構成プロパティーの早見表です。すべてのコネクターがこれらのプロパティーをすべて必要であるとは限りません。プロパティー設定は異なることがあります。

各プロパティーの説明については、表に続くセクションを参照してください。

注: 表 9 の『注』列の、句「RepositoryDirectory は <REMOTE> に設定されます」は、ブローカーが InterChange Server Express であることを示します。

表 9. 標準構成プロパティーの要約

プロパティー名	指定可能な値	デフォルト値	更新メソッド	注
AdapterHelpName	有効な <Regional Setting> ディレクトリーのある <ProductDir>¥bin¥Data¥App¥Help の有効なサブディレクトリーの 1 つ	テンプレート名 (有効な場合)、またはブランク・フィールド	コンポーネント再始動	サポートされる地域の設定。chs_chn、cht_twn、deu_deu、esn_esp、fra_fra、ita_ita、jpn_jpn、kor_kor、ptb_bra、および enu_usa (デフォルト) を含む。
AdminInQueue	有効な JMS キュー名	*<CONNECTORNAME>*/ADMININQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
AdminOutQueue	有効な JMS キュー名	*<CONNECTORNAME>*/ADMINOUTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
AgentConnections	1 から 4	1	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が MQ または IDL で、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
AgentTraceLevel	0 から 5	0	ICS の場合は動的、そうでない場合はコンポーネント再始動	
ApplicationName	アプリケーション名	コネクター・アプリケーション名に指定された値	コンポーネント再始動	

表 9. 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
BrokerType	ICS	ICS	コンポーネント再始動	
CharacterEncoding	サポートされる任意のコード。リストには、次のサブセットが表示されます。ascii7、ascii8、SJIS、Cp949、GBK、Big5、Cp297、Cp273、Cp280、Cp284、Cp037、Cp437。	ascii7	コンポーネント再始動	このプロパティーは、C++ コネクターの場合にのみ有効です。
CommonEventInfrastructure	true または false	false	コンポーネント再始動	
CommonEventInfrastructureURL	URL ストリング (例えば、corbaloc:iiop:host:2809)。	デフォルト値はありません。	コンポーネント再始動	このプロパティーは、CommonEvent Infrastructure の値が true の場合にのみ有効です。
ConcurrentEventTriggeredFlows	1 から 32767	1	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
ContainerManagedEvents	ブランクまたは JMS	ブランク	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
ControllerEventSequencing	true または false	true	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
ControllerStoreAndForwardMode	true または false	true	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
ControllerTraceLevel	0 から 5	0	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。

表 9. 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
DeliveryQueue	任意の有効な JMS キュー名	*<CONNECTORNAME>* /DELIVERYQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
DeliveryTransport	IDL または JMS	RepositoryDirectory の値が <REMOTE> の場合は IDL、それ以外は JMS。	コンポーネント再始動	RepositoryDirectory の値が <REMOTE> でない場合、このプロパティーに有効な値は JMS のみです。
DuplicateEventElimination	true または false	false	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
EnableOidForFlowMonitoring	true または false	false	コンポーネント再始動	このプロパティーは、BrokerType の値が ICS の場合にのみ有効です。
FaultQueue	任意の有効なキュー名	*<CONNECTORNAME>* /FAULTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory, CxCommon.Messaging.jms.SonicMQFactory, または 任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
jms.ListenerConcurrency	1 から 32767	1	コンポーネント再始動	このプロパティーは、jms.TransportOptimized の値が true の場合にのみ有効です。
jms.MessageBrokerName	jms.FactoryClassName の値が IBM の場合は、crossworlds.queue.manager を使用します。	crossworlds.queue.manager	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
jms.Password	任意の有効なパスワード		コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
jms.TransportOptimized	true または false	false	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS で、BrokerType の値が ICS の場合にのみ有効です。
jms.UserName	任意の有効な名前		コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。

表 9. 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
JvmMaxHeapSize	ヒープ・サイズ (メガバイト単位)	128m	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
JvmMaxNativeStackSize	スタックのサイズ (キロバイト単位)	128k	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
JvmMinHeapSize	ヒープ・サイズ (メガバイト単位)	1m	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
ListenerConcurrency	1 から 100	1	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が MQ の場合にのみ有効です。
ロケール	これは、サポートされるロケールの一部です。en_US、ja_JP、ko_KR、zh_CN、zh_TW、fr_FR、de_DE、it_IT、es_ES、pt_BR	en_US	コンポーネント再始動	
LogAtInterchangeEnd	true または false	false	コンポーネント再始動	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
MaxEventCapacity	1 から 2147483647	2147483647	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
MessageFileName	有効なファイル名	InterchangeSystem.txt	コンポーネント再始動	

表 9. 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
MonitorQueue	任意の有効なキュー名	*<CONNECTORNAME>*/MONITORQUEUE	コンポーネント再始動	このプロパティーは、DuplicateEventElimination の値が true で、ContainerManagedEvents に値がない場合にのみ有効です。
OADAutoRestartAgent	true または false	false	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
OADMaxNumRetry	正整数	1000	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
OADRetryTimeInterval	正整数 (単位: 分)	10	動的	このプロパティーは、RepositoryDirectory の値が <REMOTE> に設定されて、BrokerType の値が ICS の場合にのみ有効です。
PollEndTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒)	10000	ブローカーが ICS の場合は動的。そうでない場合は、コンポーネント再始動。	
PollQuantity	1 から 500	1	エージェント再始動	このプロパティーは、ContainerManagedEvents の値が JMS の場合にのみ有効です。
PollStartTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	
RepositoryDirectory	ブローカーが ICS の場合は <REMOTE>、そうでない場合は有効なローカル・ディレクトリー。	ICS の場合、値を <REMOTE> に設定します。	エージェント再始動	

表 9. 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
RequestQueue	有効な JMS キュー名	*<CONNECTORNAME>* /REQUESTQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
ResponseQueue	有効な JMS キュー名	*<CONNECTORNAME>* /RESPONSEQUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
RestartRetryCount	0 から 99	3	ICS の場合は動的。そうでない場合はコンポーネント再始動	
RestartRetryInterval	1 から 2147483647 までの値 (分単位)。	1	ICS の場合は動的。そうでない場合はコンポーネント再始動	
RHF2MessageDomain	mrm または xml	mrm	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS で、WireFormat の値が CwXML の場合にのみ有効です。
SourceQueue	任意の有効な WebSphere MQ キュー名	*<CONNECTORNAME>* /SOURCEQUEUE	エージェント再始動	このプロパティーは、ContainerManagedEvents の値が JMS の場合にのみ有効です。
SynchronousRequest Queue	任意の有効なキュー名	*<CONNECTORNAME>* /SYNCHRONOUSREQUEST QUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
SynchronousRequest Timeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
SynchronousResponse Queue	任意の有効なキュー名	*<CONNECTORNAME>* /SYNCHRONOUSRESPONSE QUEUE	コンポーネント再始動	このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。
TivoliMonitorTransaction Performance	true または false	false	コンポーネント再始動	

表 9. 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
WireFormat	CwXML または CwBO	CwXML	エージェント再始動	このプロパティーの値は、RepositoryDirectory の値が <REMOTE> に設定されていない場合は、CwXML でなければなりません。RepositoryDirectory の値が <REMOTE> に設定されている場合、値は CwBO でなければなりません。
WsifSynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	このプロパティーは、BrokerType の値が ICS の場合は無効です。
XMLNameSpaceFormat	short または long	short	エージェント再始動	このプロパティーは、BrokerType の値が ICS の場合は無効です。

標準のプロパティー

このセクションでは、標準コネクター構成プロパティーについて説明します。

AdapterHelpName

AdapterHelpName プロパティーは、コネクター固有の全般ヘルプ・ファイルが配置されているディレクトリーの名前です。ディレクトリーは <*ProductDir*>¥bin¥Data¥App¥Help に存在して、少なくとも言語ディレクトリー enu_usa が格納されている必要があります。ロケールに従って、他のディレクトリーが格納されている場合があります。

デフォルト値はテンプレート名 (有効な場合) か、ブランクです。

AdminInQueue

AdminInQueue プロパティーは、統合ブローカーが管理メッセージをコネクターに送信するときに使用するキューを指定します。

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

AdminOutQueue

AdminOutQueue プロパティーは、コネクターによる統合ブローカーへの管理メッセージの送信に使用されるキューを指定します。

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

AgentConnections

AgentConnections プロパティーは、ORB の初期化時にオープンされる ORB (オブジェクト・リクエスト・ブローカー) 接続数を制御します。

このプロパティーのデフォルト値は 1 です。

AgentTraceLevel

AgentTraceLevel プロパティーはアプリケーション固有のコンポーネントのトレース・メッセージのレベルを設定します。コネクターは、設定されたトレース・レベル以下の該当するトレース・メッセージをすべてデリバリーします。

デフォルト値は 0 です。

ApplicationName

ApplicationName プロパティーは、コネクター・アプリケーションの名前を一意に識別します。この名前は、システム管理者が統合環境をモニターするために使用されます。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。

デフォルトはコネクターの名前です。

BrokerType

BrokerType プロパティーは、使用する統合ブローカー・タイプを指定します。値は ICS (InterChange Server Express) です。

CharacterEncoding

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

注: Java ベースのコネクターでは、このプロパティーは使用しません。C++ ベースのコネクターは、このプロパティーに ascii7 という値を使用します。

デフォルトでは、サポートされる文字エンコードの一部のみが表示されます。サポートされる他の値をリストに追加するには、製品ディレクトリー (*<ProductDir>*) 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、本書の Connector Configurator Express に関する付録を参照してください。

ConcurrentEventTriggeredFlows

ConcurrentEventTriggeredFlows プロパティーは、コネクターがイベントのデリバリー時に並行処理できるビジネス・オブジェクトの数を決定します。この属性の値を、並行してマップおよびデリバリーするビジネス・オブジェクトの数に設定します。例えば、このプロパティーの値を 5 に設定すると、5 個のビジネス・オブジェクトが並行して処理されます。

このプロパティーを 1 よりも大きい値に設定すると、ソース・アプリケーションのコネクターが、複数のイベント・ビジネス・オブジェクトを同時にマップして、複数のコラボレーション・インスタンスにそれらのビジネス・オブジェクトを同時にデリバリーすることができます。これにより、統合ブローカーへのビジネス・オブジェクトのデリバリーにかかる時間、特にビジネス・オブジェクトが複雑なマップ

を使用している場合のデリバリー時間が短縮されます。ビジネス・オブジェクトのコラボレーションに到達する速度を増大させると、システム全体のパフォーマンスを向上させることができます。

ソース・アプリケーションから宛先アプリケーションまでのフロー全体に並行処理を実装するには、次のプロパティーを構成する必要があります。

- Maximum number of concurrent events プロパティーの値を増加して、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する必要があります。
- 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、要求を並行して処理できるように構成されている必要があります。

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

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

デフォルト値は 1 です。

ContainerManagedEvents

ContainerManagedEvents プロパティーにより、JMS イベント・ストアを使用する JMS 対応コネクターが、保証付きイベント・デリバリーを提供できるようになります。保証付きイベント・デリバリーでは、イベントはソース・キューから除去され、一つの JMS トランザクションとして宛先キューに配置されます。

このプロパティーを 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、およびDHClass (データ・ハンドラー・クラス)プロパティーを設定したデータ・ハンドラーも構成する必要があります。DataHandlerConfigMOName (オプションのメタオブジェクト名) を追加することもできます。これらのプロパティーの値を設定するには、Connector Configurator Express の「**データ・ハンドラー**」タブを使用します。

これらのプロパティーはアダプター固有ですが、以下にサンプル値をいくつか示します。

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

「**データ・ハンドラー**」タブのこれらの値のフィールドは、ContainerManagedEvents プロパティーの値を 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。

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

デフォルト値はありません。

ControllerEventSequencing

ControllerEventSequencing プロパティーにより、コネクター・コントローラーでイベント順序付けが使用可能になります。

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

デフォルト値は true です。

ControllerStoreAndForwardMode

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

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

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

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

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

デフォルト値は true です。

ControllerTraceLevel

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

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

デフォルト値は 0 です。

DeliveryQueue

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

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

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

DeliveryTransport

DeliveryTransport プロパティーはイベントのデリバリーのためのトランスポート機構を指定します。Java Messaging Service の場合、値は JMS です。

- RepositoryDirectory プロパティーの値に <REMOTE> が設定され、DeliveryTransport プロパティーの値に IDL または JMS を設定できる場合、デフォルトは IDL です。
- RepositoryDirectory プロパティーの値がローカル・ディレクトリーの場合は、指定可能な値は JMS のみです。

RepositoryDirectory プロパティーの値が 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。

デフォルト値は JMS です。

JMS

JMS トランスポート機構は、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すると、jms.MessageBrokerName、jms.FactoryClassName、jms.Password、jms.UserName などの追加の JMS プロパティーが Connector Configurator Express にリストされます。jms.MessageBrokerName プロパティーと jms.FactoryClassName プロパティーは、このトランスポートの必須プロパティーです。

InterChange Server Express (ICS) が統合ブローカーの場合に、以下の環境でコネクターに 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があります。

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。ご使用のインストール済み環境のプロセス・ヒープ・サイズが 768MB 未満である場合には、次の変数およびプロパティーを設定してください。

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

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

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

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

- IPCCBaseAddress プロパティーの値を 11 または 12 に設定します。このプロパティーの詳細については、「*WebSphere Business Integration Server Express インストール・ガイド*」(Windows 版、Linux 版、または OS/400 および i5/OS 版) を参照してください。

DuplicateEventElimination

このプロパティーの値が `true` の場合、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、コネクター開発時に、コネクターに対し、アプリケーション固有のコードでビジネス・オブジェクト ObjectEventId 属性として、一意のイベント ID が設定されている必要があります。

注: このプロパティーの値が `true` の場合、保証されたイベント・デリバリーを提供するために MonitorQueue プロパティーを使用可能にする必要があります。

デフォルト値は `false` です。

EnableOidForFlowMonitoring

このプロパティーの値が `true` の場合、アダプター・ランタイムで、着信 ObjectEventID がフロー・モニターのための外部キーとしてマークされます。

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

デフォルト値は `false` です。

FaultQueue

コネクターでメッセージを処理中にエラーが発生すると、コネクターは、そのメッセージ (と状況標識および問題記述) を FaultQueue プロパティーで指定されたキューに移動します。

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

jms.FactoryClassName

jms.FactoryClassName プロパティーは、JMS プロバイダーのためにインスタンスを生成するクラス名を指定します。このプロパティーは、DeliveryTransport プロパティーの値が `JMS` の場合は、設定する必要があります。

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

jms.ListenerConcurrency

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

このプロパティーは、jms.OptimizedTransport プロパティーの値が true の場合にのみ有効です。

デフォルト値は 1 です。

jms.MessageBrokerName

jms.MessageBrokerName プロパティーは、JMS プロバイダーのために使用するブローカー名を指定します。DeliveryTransport プロパティーでデリバリー・トランスポート機構として JMS を指定する場合は、このコネクター・プロパティーを設定する必要があります。

リモート・メッセージ・ブローカーに接続すると、このプロパティーは次の値を要求します。
QueueMgrName:*Channel*:*HostName*:*PortNumber*
ここで、各値は以下のとおりです。
QueueMgrName はキュー・マネージャー名です。
Channel はクライアントが使用するチャネルです。
HostName はキュー・マネージャーの配置先のマシン名です。
PortNumber はキュー・マネージャーが listen に使用するポートの番号です。

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

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

jms.NumConcurrentRequests

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

デフォルト値は 10 です。

jms.Password

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

デフォルト値はありません。

jms.TransportOptimized

jms.TransportOptimized プロパティーは、WIP (処理中の作業) が最適化されているかどうかを判別します。WIP の最適化は、WebSphere MQ プロバイダーが行う必要があります。最適化された WIP を操作する場合、メッセージング・プロバイダーで以下の処理が可能である必要があります。

1. キューから取り出さないで、メッセージを読み取る。
2. メッセージ全体を受信側のメモリー・スペースに転送しないで、特定の ID のメッセージを削除する。

3. 特定の ID (リカバリーのために必要) を使用してメッセージを読み取る。
4. 読み取られなかったイベントが発生する時点を追跡する。

JMS API は、上記の 2 および 4 項を満足しないので、最適化された WIP には使用できませせんが、MQ Java API は 4 つの条件をすべて満足するので、最適化された WIP には必須です。

このプロパティーは、DeliveryTransport の値が JMS で、BrokerType の値が ICS の場合にのみ有効です。

デフォルト値は false です。

jms.UserName

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

デフォルト値はありません。

JvmMaxHeapSize

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

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

デフォルト値は 128M です。

JvmMaxNativeStackSize

JvmMaxNativeStackSize プロパティーは、エージェントの最大ネイティブ・スタック・サイズ (キロバイト単位)を指定します。

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

デフォルト値は 128K です。

JvmMinHeapSize

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

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

デフォルト値は 1M です。

ListenerConcurrency

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

ティーにより、データベースへの複数イベントの書き込み操作をバッチ処理できるので、システム・パフォーマンスが向上します。

このプロパティーは、MQ トランスポートを使用するコネクターでのみ有効です。DeliveryTransport プロパティーの値は MQ でなければなりません。

デフォルト値は 1 です。

Locale

Locale プロパティーは、言語コード、国または地域、および、必要な場合には、関連文字コード・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 は 2 文字の言語コードです (小文字)。
TT は 2 文字の国または地域コード (大文字) です。
codeset は関連文字コード・セットの名前です (通常はオプション)。

デフォルトでは、サポートされるロケールの一部のみがリストされます。サポートされる他の値をリストに追加するには、<*ProductDir*>¥bin ディレクトリーにある ¥Data¥Std¥stdConnProps.xml ファイルを変更します。詳細については、本書の Connector Configurator Express に関する付録を参照してください。

コネクターが国際化に対応していない場合、このプロパティーの有効な値は en_US のみです。特定のコネクターがグローバル化に対応しているかどうかを判別するには、そのコネクターのユーザーズ・ガイドを参照してください。

デフォルト値は en_US です。

LogAtInterchangeEnd

LogAtInterchangeEnd プロパティーは、統合ブローカーのログ宛先にエラーを記録するかどうかを指定します。

ログ宛先にログを記録すると、電子メール通知もオンになります。これにより、エラーまたは致命的エラーが発生した場合には InterchangeSystem.cfg ファイルの MESSAGE_RECIPIENT に指定された値を宛先とする電子メール・メッセージが生成されます。例えば、LogAtInterChangeEnd の値が true の場合にコネクターからアプリケーションへの接続が失われると、指定されたメッセージ宛先に電子メール・メッセージが送信されます。

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

デフォルト値は false です。

MaxEventCapacity

MaxEventCapacity プロパティーは、コントローラー・バッファー内のイベントの最大数を指定します。このプロパティーは、フロー制御機能で使用されます。

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

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

MessageFileName プロパティーは、コネクター・メッセージ・ファイルの名前を指定します。メッセージ・ファイルの標準位置は、製品ディレクトリーの ¥connectors¥messages です。メッセージ・ファイルが標準位置に格納されていない場合は、メッセージ・ファイル名を絶対パスで指定します。

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

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

デフォルト値は `InterchangeSystem.txt` です。

MonitorQueue

MonitorQueue プロパティーは、コネクターが重複イベントをモニターするために使用する論理キューを指定します。

このプロパティーは、DeliveryTransport プロパティーの値が `JMS` で、DuplicateEventElimination の値が `true` の場合にのみ有効です。

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

OADAutoRestartAgent

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

自動再始動機能およびリモート再始動機能を使用可能にするには、このプロパティーを `true` に設定する必要があります。WebSphere MQ により起動される OAD 機能の構成方法については、「*WebSphere Business Integration Server Express インストール・ガイド (Windows 版)*」、「*WebSphere Business Integration Server Express インストール・ガイド (Linux 版)*」または「*WebSphere Business Integration Server Express インストール・ガイド (OS/400 および i5/OS 版)*」を参照してください。

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

デフォルト値は `false` です。

OADMaxNumRetry

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

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

デフォルト値は `1000` です。

OADRetryTimeInterval

OADRetryTimeInterval プロパティーは、WebSphere MQ により起動される Object Activation Daemon (OAD) の再試行時間間隔の分数を指定します。コネクター・エージェントがこの再試行時間間隔内に再始動しない場合は、コネクター・コントローラーはコネクター・エージェントを再び再始動するように OAD に要求します。OAD はこの再試行プロセスを OADMaxNumRetry プロパティーで指定された回数だけ繰り返します。このプロパティーを有効にするには、OADAutoRestartAgent プロパティーを `true` に設定する必要があります。

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

デフォルト値は `10` です。

PollEndTime

PollEndTime プロパティーは、イベント・キューのポーリングを停止する時刻を指定します。形式は *HH:MM* です。ここで、*HH* は 0 から 23 時、*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、値のない `HH:MM` ですが、この値は変更する必要があります。

アダプター・ランタイムが以下のことを検出した場合、

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

PollFrequency プロパティーに構成された値を使用してポーリングします。

PollFrequency

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

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

次の値は、このプロパティーに有効です。

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

デフォルト値は `10000` です。

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

PollQuantity

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

このプロパティーは、`DeliveryTransport` プロパティーの値が `JMS` で、`ContainerManagedEvents` プロパティーに値が設定されている場合にのみ有効です。

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

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

PollStartTime

PollStartTime プロパティーは、イベント・キューのポーリングを開始する時刻を指定します。形式は *HH:MM* です。ここで、*HH* は 0 から 23 時、*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、値のない HH:MM ですが、この値は変更する必要があります。

アダプター・ランタイムが以下のことを検出した場合、

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

PollFrequency プロパティーに構成された値を使用してポーリングします。

RepositoryDirectory

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

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

統合ブローカーが WebSphere Message Broker または WAS の場合、この値はデフォルトで <*ProductDir*>¥repository に設定されます。ただし、この値は任意の有効なディレクトリー名に設定できます。

RequestQueue

RequestQueue プロパティーは、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するキューを指定します。

このプロパティーは、DeliveryTransport プロパティーの値が JMS の場合にのみ有効です。

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

ResponseQueue

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

このプロパティーは、DeliveryTransport プロパティーの値が JMS の場合にのみ有効です。

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

RestartRetryCount

RestartRetryCount プロパティーは、コネクターによるコネクター自体の再始動の試行回数を指定します。このプロパティーを並列接続の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クライアント側のアプリケーション固有のコンポーネントの再始動を試行する回数が指定されます。

デフォルト値は 3 です。

RestartRetryInterval

RestartRetryInterval プロパティーは、コネクターによるコネクター自体の再始動の試行間隔を分単位で指定します。このプロパティーを並列リンクの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クライアント側のアプリケーション固有のコンポーネントの再始動を試行する間隔が指定されます。

このプロパティーに可能な値は、1 から 2147483647 までです。

デフォルト値は 1 です。

RHF2MessageDomain

RHF2MessageDomain プロパティーにより、JMS ヘッダーのドメイン名フィールドの値を構成できます。JMS トランスポートを介してデータを WebSphere メッセージ・ブローカーに送信するときに、アダプター・フレームワークにより JMS ヘッダー情報、ドメイン名、および固定値 mrm が書き込まれます。構成可能なドメイン名により、WebSphere Message Broker がメッセージ・データを処理する仕組みを追跡できます。

これはヘッダーの例です。

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

このプロパティーは、BrokerType の値が ICS の場合は無効です。 また、DeliveryTransport プロパティーの値が JMSで、WireFormat プロパティーの値が CwXML の場合にのみ有効です。

指定可能な値は、mrm および xml です。デフォルト値は mrm です。

SourceQueue

SourceQueue プロパティーは、JMS イベント・ストアを使用する JMS 対応コネクターでの保証付きイベント・デリバリーをサポートするコネクター・フレームワークに、JMS ソース・キューを指定します。詳細については、67 ページの『ContainerManagedEvents』を参照してください。

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

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

SynchronousRequestQueue

SynchronousRequestQueue プロパティーは、同期応答を要求する要求メッセージをコネクター・フレームワークからブローカーに配信します。このキューは、コネクターが同期実行を使用する場合にのみ必要です。同期実行の場合、コネクター・フレームワークは、同期要求キューにメッセージを送信し、同期応答キューでブローカーからの応答を待機します。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。

このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。

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

SynchronousRequestTimeout

SynchronousRequestTimeout プロパティーは、コネクターが同期要求への応答を待機する時間をミリ秒単位で指定します。指定された時間内に応答を受信できなかった場合、コネクターは元の同期要求メッセージ (およびエラー・メッセージ) を障害キューに移動します。

このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。

デフォルト値は 0 です。

SynchronousResponseQueue

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

このプロパティーは、DeliveryTransport の値が JMS の場合にのみ有効です。

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

TivoliMonitorTransactionPerformance

TivoliMonitorTransactionPerformance プロパティーで、IBM Tivoli Monitoring for Transaction Performance (ITMTP) が実行時に起動されるかどうかを指定します。

デフォルト値は false です。

WireFormat

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

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

索引

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

[ア行]

[カ行]

[サ行]

[タ行]

[ハ行]

[マ行]

[ラ行]

B

C

I

O

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものであり、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。製造元は、お客様が提供するいかなる情報も、お客様になんら義務も負わせない適切な方法で、使用もしくは配布することがあります。本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム（本プログラムを含む）との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
577 Airport Blvd., Suite 800
Burlingame, CA 94010
U.S.A

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。著作権使用許諾: 本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

プログラミング・インターフェース情報は、プログラムを使用してアプリケーション・ソフトウェアを作成する際に役立ちます。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
CICS
CrossWorlds
DB2
DB2 Universal Database
IMS
Informix
i5/OS
iSeries
Lotus
Lotus Domino
Lotus Notes
MQIntegrator
MQSeries
MVS
OS/400
Passport Advantage
SupportPac
WebSphere
z/OS
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 および Pentium 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。
Linux は、Linus Torvalds の米国およびその他の国における商標です。他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

WebSphere Business Integration Server Express and Express Plus には、Eclipse Project (http://www.eclipse.org/) により開発されたソフトウェアが含まれています。

WebSphere Business Integration Server Express バージョン 4.4、および WebSphere Business Integration Server Express Plus 4.4

Printed in Japan

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