

IBM WebSphere Business Integration Adapters



Adapter for iSeries ユーザーズ・ガイド

バージョン 2.1.x

IBM WebSphere Business Integration Adapters



Adapter for iSeries ユーザーズ・ガイド

バージョン 2.1.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、91 ページの『特記事項』に記載されている情報をお読みください。

本書は、コネクター・バージョン 2.1.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 Adapters
Adapter for iSeries User Guide
Version 2.1.x

発 行： 日本アイ・ピー・エム株式会社

担 当： ナショナル・ランゲージ・サポート

第1刷 2005.12

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

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

© Copyright International Business Machines Corporation 2003, 2005. All rights reserved.

© Copyright IBM Japan 2005

目次

本書について	v
対象読者	v
関連文書	v
表記上の規則	vi
本リリースの新機能	ix
バージョン 2.1.x	ix
バージョン 2.0.x	ix
以前のバージョン	ix
第 1 章 概要	1
iSeries および AS/400 システムの概要	1
アダプターの動作方法	4
第 2 章 iSeries アダプターのインストール	7
iSeries 環境のアダプター	7
iSeries アダプターと関連ファイルのインストール	9
インストール済みファイルの構造	9
インストール後の作業	10
第 3 章 iSeries アダプターの構成	11
コネクターの構成	11
第 4 章 コネクター用のビジネス・オブジェクトについて	35
コネクター・メタデータの定義	35
ビジネス・オブジェクト構造の概要	36
RPG、COBOL、および Java プログラムのビジネス・オブジェクト構造	36
iSeries データ・キューのビジネス・オブジェクト構造	40
ポーリングのためにメタオブジェクトを構成	41
ビジネス・オブジェクト属性のプロパティの指定	43
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	44
第 5 章 ビジネス・オブジェクトの作成および変更	47
ODA for iSeries の概要	47
ビジネス・オブジェクト定義の生成	47
ビジネス・オブジェクト情報の指定	53
ビジネス・オブジェクトのアップロード	55
第 6 章 トラブルシューティングおよびエラー処理	57
エラー処理	57
ロギング	57
トレース・メッセージ	57
付録. コネクターの標準構成プロパティ	61
新規プロパティ	61
標準コネクター・プロパティの概要	61
標準プロパティのクイック・リファレンス	63
標準プロパティ	70
索引	89

特記事項	91
プログラミング・インターフェース情報	93
商標	93

本書について

IBM^(R) WebSphere^(R) Business Integration Adapters 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、IBM WebSphere Business Integration Adapter 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
- OS/400 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

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

関連文書については、以下のサイトからインストールできます。

アダプターの一般情報が必要な場合、アダプターを WebSphere Message Broker (WebSphere MQ Integrator, WebSphere MQ Integrator Broker, WebSphere Business Integration Message Broker) とともに使用する場合は、およびアダプターを WebSphere Application Server とともに使用する場合は、以下のサイトを参照してください。

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

アダプターを InterChange Server とともに使用する場合は、以下のサイトを参照してください。

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

Message Broker (WebSphere MQ Integrator Broker, WebSphere MQ Integrator, および WebSphere Business Integration Message Broker) の詳細については、以下のサイトを参照してください。

<http://www.ibm.com/software/integration/mqfamily/library/manualsa/>

WebSphere Application Server の詳細については、以下をご覧ください。

<http://www.ibm.com/software/webservers/appserv/library.html>

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

JT400 および iSeries ACCESS の詳細については、以下をご覧ください。

<http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html>

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 青のアウトライン	変数名または相互参照を示します。 オンラインで表示したときのみ見られる青のアウトラインは、相互参照用のハイパーリンクです。アウトラインの内側をクリックすると、参照先オブジェクトにジャンプします。
{ }	構文の記述行の場合、中括弧 {} で囲まれた部分は、選択対象のオプションです。1 つのオプションのみを選択する必要があります。
[]	構文の記述行の場合、大括弧 [] で囲まれた部分は、オプション・パラメーターです。
...	構文の記述行の場合、省略符号 ... は直前のパラメーターが繰り返されることを示します。例えば、option[,...] は、複数のオプションをコンマで区切って指定できることを意味します。
< >	命名規則では、不等号括弧は名前の個々の要素を囲み、各要素を区別します。(例: <code><server_name><connector_name>tmp.log</code>)
/, ¥	本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムの場合には、円記号をスラッシュ (/) に置き換えてください。すべての 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。

<code>%text%</code> および <code>\$text</code>	% 記号で囲まれたテキストは、Windows の <code>text</code> システム変数またはユーザー変数の値を示します。UNIX 環境での同等の表記は <code>\$text</code> です。これは、UNIX 環境変数 <code>text</code> の値を示します。
<code>ProductDir</code>	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 2.1.x

- iSeries ODA は RPG および RPGLE ソース・プログラムをサポートします。
- AIX 5.3、Linux RedHat 3.0 Update 1、Linux SuSe 8.1 SP3、および Java Toolbox V5.3.0.6 をサポートします。ハードウェアおよびソフトウェアのサポートの詳細については、7 ページの『iSeries 環境のアダプター』を参照してください。
- IBM Tivoli License Manager (ITLM) をサポートします。
- アダプターは、COBOL および JAVA で記述された .PGM を呼び出すことができるようになりました。この機能をサポートするため、新しい動詞「CALLPGM」が追加されました。
- JVM 1.4.2 をサポートします。
- iSeries アダプターおよび ODA に対するグローバリゼーション・サポートを行います。アダプターおよび ODA は G1 言語をサポートしますが、ASI はサポートしません。

バージョン 2.0.x

- 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 RPG 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 つの呼び出しで 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。

以前のバージョン

以前のバージョンでの変更を、以下のセクションに記述します。

バージョン 1.1.x

RPG (Report Program Generator) 3 のサポートが追加されました。

バージョン 1.1 以降、Adapter for iSeries は Microsoft Windows NT ではサポートされなくなりました。

アダプターのインストールに関する情報が本書から移動しました。この情報の新しい掲載先については、第 2 章 (9 ページの『iSeries アダプターと関連ファイルのインストール』) を参照してください。

バージョン 1.0.x

バージョン 1.0.x は、IBM WebSphere Business Integration Adapter for iSeries の最初のリリースです。このアダプターは、iSeries システムまたは AS/400 システムで RPG (Report Program Generator) 4 プログラムを実行する機能を提供します。

第 1 章 概要

この章では、IBM WebSphere Business Integration Adapter for iSeries について説明します。このアダプターは、IBM Toolbox for Java (JavaTM クラスのセット) を使用して、iSeries システム上または AS/400 システム上にある、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提供されます。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設定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。

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

本章の内容は、次のとおりです。

- 『iSeries および AS/400 システムの概要』
- 4 ページの『アダプターの動作方法』

iSeries および AS/400 システムの概要

IBM iSeries (AS/400 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。

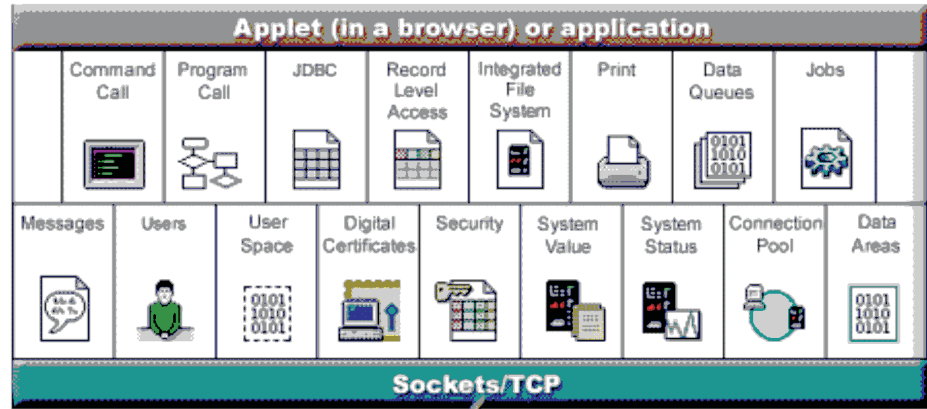
ホスト・サーバーは、3 ページの図 1 に示すように、アプリケーションを実行している装置 (クライアント PC など) からの要求を処理して、文書の印刷などのタスクを実行できるようにします。iSeries コンピューターと AS/400 コンピューターは機能の充実したサーバーであり、ファイル操作、データベース操作、アプリケーション実行、メール処理、印刷処理、マルチメディア処理、FAX 処理、無線通信などの多数のタスクを同時に実行することができます。タスク・サーバーは、システム上でそれぞれ個別のジョブとして実行されます。また、サーバー・ジョブは、それぞれソケット接続を介してデータ・ストリームを送受信します。

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 AS/400 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 AS/400 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 AS/400) 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。

JVM が稼働しているクライアント



AS/400 システム

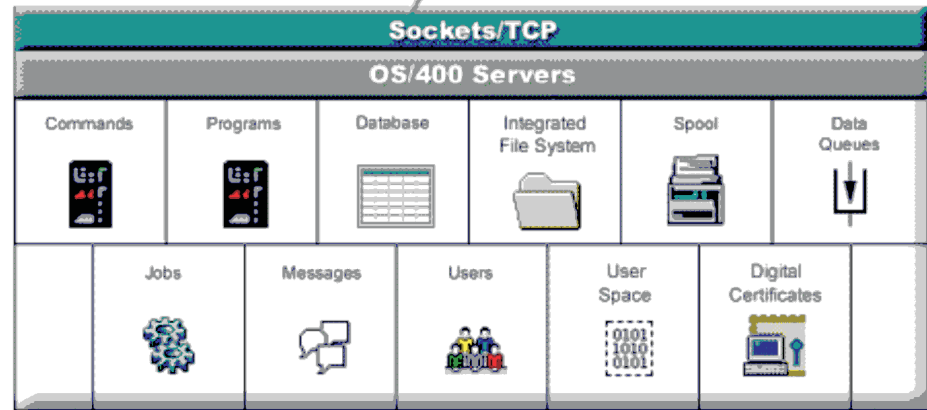


図1. AS/400 クライアント - サーバー・アーキテクチャーの概要

AS/400 上で稼働する OS/400 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、AS/400 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図を、4 ページの図2 に示します。

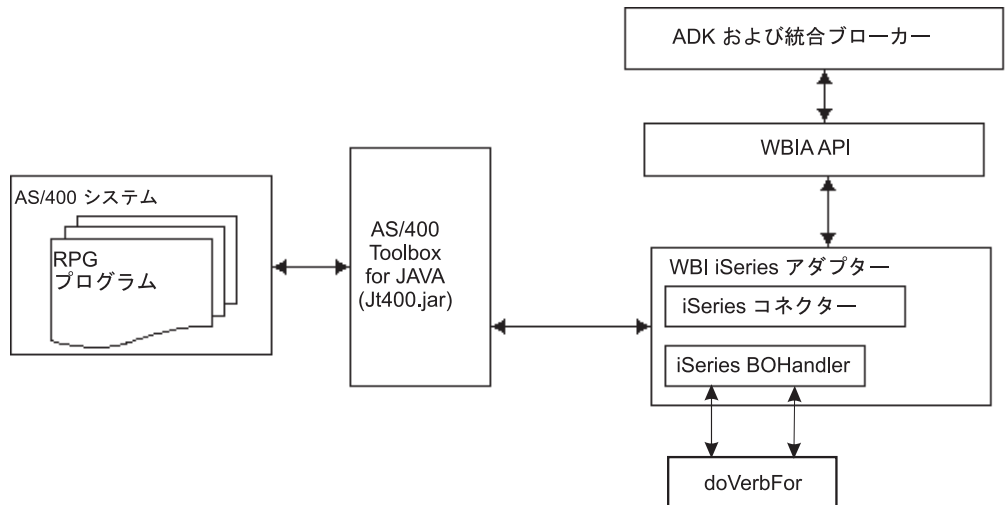


図2. Adapter for iSeries を使用した場合の接続図

データ・キュー

iSeries 上のデータ・キューによって、ジョブ間的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 AS/400 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 AS/400 システムへの接続に使用されます。

コネクタの動作

コネクタは、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 AS/400 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 AS/400 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
- 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』
- 9 ページの『iSeries アダプターと関連ファイルのインストール』
- 9 ページの『インストール済みファイルの構造』
- 10 ページの『インストール後の作業』

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。以下の各セクションのほか、次の技術文書でもハードウェアおよびソフトウェア要件を参照できます。

<http://www.ibm.com/support/docview.wss?uid=swg27006249>

- 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 2.6 および 2.7。

統合ブローカー:

- WebSphere InterChange Server、バージョン 4.2.2、4.3
- WebSphere MQ Integrator、バージョン 2.1.0
- WebSphere MQ Integrator Broker、バージョン 2.1.0
- WebSphere Business Integration Message Broker、バージョン 5.0
- WebSphere Application Server Enterprise、バージョン 5.0.2 (WebSphere Studio Application Developer Integration Edition、バージョン 5.0.1 と併用)
- WebSphere Business Integration Server Foundation 5.1、5.1.1

例外については、「リリース情報」を参照してください。

注: 統合ブローカーおよびその前提条件のインストールに関する説明については、以下の資料を参照してください。WebSphere InterChange Server (ICS) について

は、「システム・インストール・ガイド (UNIX 版)」または「システム・インストール・ガイド (Windows 版)」を参照してください。

Message Brokers (WebSphere MQ Integrator Broker、WebSphere MQ Integrator、および WebSphere Business Integration Message Broker) の場合は、「*WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド*」およびそれぞれの Message Brokers のインストールに関する資料を参照してください。

この資料の一部は、

<http://www.ibm.com/software/integration/mqfamily/library/manualsa/> の Web サイトで参照できます。

WebSphere Application Server については、「*アダプター実装ガイド (WebSphere Application Server)*」および次の資料を参照してください。

<http://www.ibm.com/software/webservers/appserv/library.html>

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。

- Windows 2000、2003
- Sun Solaris 8、9
- AIX 5.2、5.3
- HP-UX 11i
- Linux RedHat AS/ ES/WS 3.0 with Update 1
- Linux SuSe 8.1 with SP3

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 1.3 以降
 - JSSE (Java Secure Socket Extension) 1.0
 - Jt400.jar ファイル

注: IBM Toolbox for Java (ライセンス製品 5722-JC1) V5R2 のファイルは、<http://www-1.ibm.com/servers/eserver/series/toolbox/downloads.htm> の

Toolbox Web サイトからダウンロードできます。

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。

- WBIA.jar ファイル
 - CrossWorlds.jar ファイル
 - BIA_iSeries.jar ファイル
2. Adapter for iSeries は、以下のいずれかのバージョンの OS/400 を介して AS/400 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
 - バージョン 4 リリース 1 から 3

- OS/400 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: OS/400 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。以下のリンクを参照して、適切な PTF を適用してください。

<http://www-1.ibm.com/servers/eserver/series/toolbox/hostservicepackdetail.htm>

- 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 アダプター製品のインストールについては、次のサイトで WebSphere Business Integration Adapters インフォメーション・センターにある「WebSphere Business Integration Adapters のインストール」のガイドを参照してください。

<http://www.ibm.com/websphere/integration/wbiadapters/library/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

<code>ProductDir</code> のサブディレクトリー	説明
<code>\connectors\iSeries\BIA_iSeries.jar</code>	iSeries コネクターのみが使用するクラスが含まれている
<code>\connectors\iSeries\start_iSeries.bat</code>	iSeries コネクターの始動スクリプト (Windows)
<code>\connectors\iSeries\start_iSeries.sh</code>	iSeries コネクターの始動スクリプト (Unix)
<code>\connectors\iSeries\ext\</code>	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 (<code>start_iSeries.bat</code> または <code>start_iSeries.sh</code>) でディレクトリーを指定します。
<code>\connectors\messages\BIA_iSeriesAdapter.txt</code>	コネクターのメッセージ・ファイル
<code>\ODA\iSeries\BIA_iSeriesODA.jar</code>	iSeries ODA
<code>\ODA\iSeries\start_iSeriesODA.bat</code>	ODA 始動ファイル (Windows)
<code>\ODA\iSeries\start_iSeriesODA.sh</code>	ODA 始動ファイル (UNIX)

表 1. コネクタのファイル構造 (続き)

<i>ProductDir</i> のサブディレクトリ	説明
Data¥App¥BIA_iSeriesAdapterTemplate	コネクタのリポジトリ定義。

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、11 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。

- 『コネクタの構成』

コネクタの構成

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・プロパティを使用します。また、その次のセクションで説明するコネクタ固有のプロパティも使用します。

このセクションには、以下のトピックがあります。

- 『Connector Configurator の概要』
- 12 ページの『Connector Configurator の始動』
- 13 ページの『System Manager からのコンフィギュレーターの実行』
- 13 ページの『コネクタ固有のプロパティ・テンプレートの作成』
- 16 ページの『新規構成ファイルの作成』
- 19 ページの『構成ファイル・プロパティの設定』
- 27 ページの『構成ファイルの保管』
- 28 ページの『構成ファイルの変更』
- 28 ページの『構成の完了』
- 29 ページの『グローバル化環境における Connector Configurator の使用』
- 29 ページの『コネクタの始動』
- 31 ページの『コネクタの停止』
- 31 ページの『複数のコネクタ・インスタンスの作成』

Connector Configurator の概要

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

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (これらをまとめて WebSphere Message Brokers (WMQI) と呼びます。)
- WebSphere Application Server (WAS)

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

- コネクタを構成するための**コネクタ固有のプロパティ・テンプレート**を作成します。
- **コネクタ構成ファイル**を作成します。インストールするコネクタごとに構成ファイルを 1 つ作成する必要があります。

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

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

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

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

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

ただし**コネクタ固有プロパティ**の場合は、最初にプロパティを定義し、その値を設定する必要があります。このため、特定のアダプターのコネクタ固有プロパティのテンプレートを作成します。システム内で既にテンプレートが作成されている場合には、作成されているテンプレートを使用します。システム内でまだテンプレートが作成されていない場合には、14 ページの『新規テンプレートの作成』のステップに従い、テンプレートを新規に作成します。

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

Connector Configurator の始動

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

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

スタンドアロン・モードでのコンフィギュレーターの実行

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

これを行うには、以下のステップを実行します。

- 「スタート」>「プログラム」から、「IBM WebSphere InterChange Server」>「IBM WebSphere Business Integration Tools」>「Connector Configurator」をクリックします。
- 「ファイル」>「新規」>「コネクタ構成」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します。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、WAS のいずれかを選択します。

Connector Configurator を個別に実行して構成ファイルを生成してから、System Manager に接続してこの構成ファイルを System Manager プロジェクトに保存することもできます (19 ページの『構成ファイルの完成』を参照)。

System Manager からのコンフィギュレーターの実行

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

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

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

既存の構成ファイルを編集するには、以下のステップを実行します。

- 「System Manager」ウィンドウの「コネクタ」フォルダーでいずれかの構成ファイルを選択し、右クリックします。Connector Configurator が開き、この構成ファイルの統合ブローカー・タイプおよびファイル名が上部に表示されます。
- Connector Configurator で「ファイル」>「開く」を選択します。プロジェクトまたはプロジェクトが保管されているディレクトリーからコネクタ構成ファイルを選択します。
- 「標準のプロパティ」タブをクリックし、この構成ファイルに含まれているプロパティを確認します。

コネクタ固有のプロパティ・テンプレートの作成

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

コネクタ固有プロパティのテンプレートを新規に作成するか、または既存のコネクタ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、14 ページの『新規テンプレートの作成』を参照してください。

- 既存のファイルを使用する場合には、既存のテンプレートを変更し、新しい名前
でこのテンプレートを保管します。既存のテンプレートは
%WebSphereAdapters%bin%Data%App ディレクトリーにあります。

新規テンプレートの作成

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

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

1. 「ファイル」>「新規」>「コネクタ固有プロパティー・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。ご使用のコネクタで使用するコネクタ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

プロパティの一般特性の選択を終えたら、「値」タブをクリックします。

値の指定: 「値」タブを使用すると、プロパティの最大長、最大複数値、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. 「プロパティを編集」表示でプロパティの名前を選択します。
3. 「最大長」および「最大複数値」のフィールドに値を入力します。

新規プロパティ値を作成するには、以下のステップを実行します。

1. 「プロパティを編集」リストでプロパティを選択し、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、「プロパティ値」ダイアログ・ボックスで入力した値と、以前に作成した値が表示されます。

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

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

値が作成されて、グリッドに表示されると、そのテーブルの表示内から編集できるようになります。

テーブルにある既存の値の変更を行うには、その行の行番号をクリックして行全体を選択します。次に「値」フィールドを右マウス・ボタンでクリックし、「値の編集 (Edit Value)」をクリックします。

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

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

1. 「使用可能なプロパティ」表示で、依存プロパティとして指定するプロパティを選択します。
2. 「プロパティを選択」フィールドで、ドロップダウン・メニューを使用して、条件値を持たせるプロパティを選択します。

3. 「条件演算子」フィールドで以下のいずれかを選択します。

== (等しい)

!= (等しくない)

> (より大)

< (より小)

>= (より大か等しい)

<= (より小か等しい)

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

新規構成ファイルの作成

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

- 「System Manager」ウィンドウで「コネクタ」フォルダーを右クリックし、「新規コネクタの作成」を選択します。Connector Configurator が開き、「新規コネクタ」ダイアログ・ボックスが表示されます。
- スタンドアロン・モードの場合は、Connector Configurator で「ファイル」>「新規」>「コネクタ構成」を選択します。「新規コネクタ」ウィンドウで、新規コネクタの名前を入力します。

また、統合ブローカーも選択する必要があります。選択したブローカーによって、構成ファイルに記述されるプロパティが決まります。ブローカーを選択するには、以下のステップを実行します。

- 「Integration Broker」フィールドで、ICS 接続、WebSphere Message Brokers 接続、WAS 接続のいずれかを選択します。
- 本章の後半で説明するように、「新規コネクタ」ウィンドウの他のフィールドをドロップダウンします。

コネクタ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」ダイアログ・ボックス表示されません。

- **名前**

コネクタ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のファイル名に対応した一意の名前でなければなりません。

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

- **システム接続**

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

- 「コネクタ固有プロパティ・テンプレート」を選択します。

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

使用するテンプレートを選択し、「**OK**」をクリックします。

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

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

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

既存ファイルの使用

使用可能な既存ファイルは、以下の 1 つまたは複数の形式になります。

- **コネクタ定義ファイル。**

コネクタ定義ファイルは、特定のコネクタのプロパティと、適用可能なデフォルト値がリストされたテキスト・ファイルです。コネクタの配布パッケージ

ジの `¥repository` ディレクトリー内には、このようなファイルが格納されていることがあります (通常、このファイルの拡張子は `.txt` です。例えば、XML コネクタの場合は `CN_XML.txt` です)。

- ICS リポジトリー・ファイル。
コネクタの以前の ICS インプリメンテーションで使用した定義は、そのコネクタの構成で使用されたリポジトリー・ファイルで使用可能になります。そのようなファイルの拡張子は、通常 `.in` または `.out` です。
- コネクタの以前の構成ファイル。
これらのファイルの拡張子は、通常 `*.cfg` です。

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

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

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

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

ICS 環境でのコネクタ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せん。

- すべてのファイル (`*.*`)

コネクタのアダプター・パッケージに `*.txt` ファイルが付属していた場合、または別の拡張子で定義ファイルが使用可能である場合は、このオプションを選択します。

3. ディレクトリー表示内で、適切なコネクタ定義ファイルへ移動し、ファイルを選択し、「開く」をクリックします。

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

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

構成ファイルの完成

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

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

1. 「標準のプロパティ」タブで、BrokerType プロパティの値フィールドを選択します。ドロップダウン・メニューで、値 ICS、WMQI、または WAS を選択します。
2. 選択したブローカーに関連付けられているプロパティが「標準のプロパティ」タブに表示されます。ここでファイルを保管するか、または 23 ページの『サポートされるビジネス・オブジェクト定義の指定』の説明に従い、残りの構成フィールドに値を入力することができます。
3. 構成が完了したら、「ファイル」>「保管」>「プロジェクトに」を選択するか、または「ファイル」>「保管」>「ファイルに」を選択します。

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

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

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

構成ファイル・プロパティの設定

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

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

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

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

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

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

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

標準プロパティとコネクタ固有プロパティの違いは、以下のとおりです。

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

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

- 背景がグレーのフィールドは、標準のプロパティを表します。値を変更することはできますが、名前の変更およびプロパティの除去はできません。
- 背景が白のフィールドは、アプリケーション固有のプロパティを表します。これらのプロパティは、アプリケーションまたはコネクタの特定のニーズによって異なります。値の変更も、これらのプロパティの除去も可能です。
- 「値」フィールドは構成できます。
- プロパティごとに「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・プロパティの設定

標準構成プロパティにより、すべてのコネクタ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の資料については、61 ページの『コネクタの標準構成プロパティ』を参照してください。

重要: 本書で解説するコネクタはすべての統合ブローカーをサポートしているため、関係する構成プロパティは、すべてのブローカーに適合する構成プロパティのみになります。

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

- AgentTraceLevel
- ApplicationName
- ControllerStoreAndForwardMode
- ControllerTraceLevel
- DeliveryTransport

標準のプロパティの値を変更するには、以下のステップを実行します。

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

コネクタ固有のプロパティ: コネクタ固有の構成プロパティは、コネクタが実行時に必要とする情報を提供します。また、コネクタ固有の構成プロパティを使用すれば、コネクタ・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、コネクタのコネクタ固有の構成プロパティをリストします。各プロパティの説明については、以降のセクションを参照してください。

表 2. コネクタ固有の構成プロパティ

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い場合は、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い場合は、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. 20 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・ユーザズ・ガイドにあります。

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

更新メソッド: 付録 A『コネクターの標準構成プロパティ』の 62 ページの『構成プロパティ値の概要』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 の「サポートされているビジネス・オブジェクト」タブを使用します。汎用ビジネス・オブジェクトと、アプリケーション固有のビジネス・オブジェクトの両方を指定する必要があり、またそれらのビジネス・オブジェクト間のマップの関連を指定することが必要です。

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

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

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

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

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

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

一般に、コネクタ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・エージェントによってサポートされるよう指定するには、「エージェント・サポート」ボックスにチェックマークを付けます。「Connector Configurator」ウィンドウでは「エージェント・サポート」の選択の妥当性は検査されません。

最大トランザクション・レベル: コネクタの最大トランザクション・レベルは、そのコネクタ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の場合、選択可能な項目は「最大限の努力」のみです。

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

ご使用のブローカーが WebSphere Message Broker の場合: スタンドアロン・モードで作業している (System Manager に接続していない) 場合、手動でビジネス・オブジェクト名を入力する必要があります。

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

WebSphere Business Integration Message Broker 5.0 を使用している場合、「メッセージ・セット ID」はオプションのフィールドです。値を指定するときは、固有の値を指定しなくてもかまいません。ただし、WebSphere MQ Integrator 2.1 または WebSphere MQ Integrator Broker 2.1 を使用している場合は、固有の ID を指定する必要があります。

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

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

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

関連付けられたマップ (ICS のみ)

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

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

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

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

「関連付けられたマップ」タブには以下のフィールドが表示されます。

• ビジネス・オブジェクト名

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

• 関連付けられたマップ

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

• 明示的

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

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

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

以下のステップを実行して、マップを明示的にバインドします。

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

リソース (ICS)

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

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

メッセージング (ICS)

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

トレース/ログ・ファイル値の設定

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

ログとトレースの値を変更するには、以下のステップを実行します。

1. 「トレース/ログ・ファイル」タブをクリックします。
2. ログとトレースのどちらでも、以下のいずれかまたは両方へのメッセージの書き込みを選択できます。

- コンソールに (STDOUT):
ログ・メッセージまたはトレース・メッセージを STDOUT ディスプレイに書き込みます。

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

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

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

データ・ハンドラー

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

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

構成ファイルの保管

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

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

- System Manager から 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。
- WebSphere Application Server プロジェクトをセットアップしている場合には、このファイルを WebSphere Application Server プロジェクトに保管することもできます。

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

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

構成ファイルの変更

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

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

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

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

構成の完了

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

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

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

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

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

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

CharacterEncoding および Locale 標準構成プロパティのドロップ・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 プロパティの値のリストにロケール `en_GB` を追加するには、`stdConnProps.xml` ファイルを開き、以下に太字で示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>
  </ValidValues>
  <DefaultValue>en_US</DefaultValue>
</Property>
```

コネクタの始動

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

```
ProductDir¥connectors¥connName
```

ここで、`connName` はコネクタを示します。

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

始動スクリプトの名前は、表 3 に示すように、オペレーティング・システム・プラットフォームによって異なります。

表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
UNIX ベースのシステム	connector_manager
Windows	start_connName.bat

始動スクリプトは、実行時に、デフォルトで *Productdir* (以下のコマンドを参照) に構成ファイルがあることを予期します。これは、構成ファイルを置く場所です。

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

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

- Windows システムで「スタート」メニューから。

「プログラム」>「IBM WebSphere Business Integration Adapters」>「アダプター」>「コネクター」を選択します。デフォルトでは、プログラム名は「IBM WebSphere Business Integration Adapters」となっています。ただし、これはカスタマイズすることができます。あるいは、ご使用のコネクターへのデスクトップ・ショートカットを作成することもできます。

- コマンド行から。

– Windows システム:

```
start_connName connName brokerName [-cconfigFile ]
```

– UNIX ベースのシステム:

```
connector_manager -start connName brokerName [-cconfigFile ]
```

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

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

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

- WebSphere Application Server または InterChange Server ブローカーとともに実行する System Manager の開始時に起動するアダプター・モニターから:

このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- System Manager (すべてのブローカーで使用可能) から:

このツールを使用して、コネクターのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- Windows システムでは、Windows サービスとして始動するようにコネクタを構成することができます。この場合、Windows システムがブートしたとき (自動サービスの場合)、または Windows サービス・ウィンドウを通じてサービスを始動したとき (手動サービスの場合) に、コネクタが始動します。

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

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

コネクタの停止

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

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

```
connector_manager_connName -stop
```

ここで、*connName* はコネクタの名前です。

- Adapter Monitor から (WebSphere Business Integration Adapters 製品のみ)。Adapter Monitor は System Manager 始動時に起動されます。

このツールを使用して、コネクタのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- System Monitor から (WebSphere InterChange Server 製品のみ)。

このツールを使用して、コネクタのロード、アクティブ化、非アクティブ化、休止、シャットダウン、または削除を行うことができます。

- Windows システムでは、Windows サービスとして始動するようにコネクタを構成することができます。この場合、Windows システムのシャットダウン時に、コネクタは停止します。

複数のコネクタ・インスタンスの作成

コネクタ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、コネクタの複数のインスタンスを作成して実行するように、ご使用のシステムを設定することができます。次のようにする必要があります。

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

新規ディレクトリーの作成

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

`ProductDir¥connectors¥connectorInstance`

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

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

`ProductDir¥repository¥connectorInstance`

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

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

`ProductDir¥repository¥initialConnectorInstance`

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

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

1. 初期コネクターの構成ファイル (コネクター定義) をコピーし、名前変更します。
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: 始動スクリプトは以下のように作成します。

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

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

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

第 4 章 コネクタ用のビジネス・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・メタデータの定義』
- 36 ページの『RPG、COBOL、および Java プログラムのビジネス・オブジェクト構造』
- 40 ページの『iSeries データ・キューのビジネス・オブジェクト構造』
- 41 ページの『ポーリングのためにメタオブジェクトを構成』
- 43 ページの『ビジネス・オブジェクト属性のプロパティの指定』
- 44 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』

IBM WebSphere Business Integration 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) ユーティリティについては、47 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』の第 5 章を参照してください。

コネクタ・メタデータの定義

iSeries コネクタ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・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、フラットなものと同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・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・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、COBOL、および Java 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 用のキーになっている必要があります。

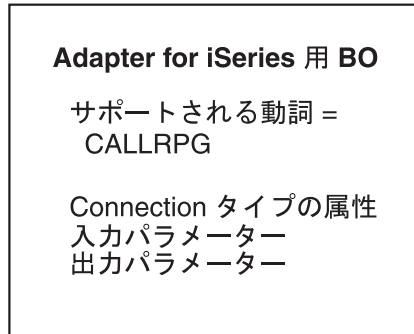


図3. *iSeries* の親ビジネス・オブジェクト

また、タイプが **Connection** の子属性もあります。これには、AS/400 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。

注: **iSeries** ビジネス・オブジェクトでは、接続ビジネス・オブジェクトのアプリケーション固有の情報をブランクにすることができます。これは、ビジネス・オブジェクト **ASI** がアダプターによって処理されないためです。

図5の大括弧内の情報は、ビジネス・オブジェクトのアプリケーション固有の情報を表しています。

注: 動詞名には、大/小文字の区別はありません。例えば、動詞 **getqueue** は、**GETQUEUE** や **GetQueue** と入力しても構いません。



図4. *iSeries* の子ビジネス・オブジェクト

Verb = CALLRPG
Connection のインスタンス HostName = (host).(domain).com UserName = (username) Password = (password)
属性 1 [ParamType=Input; Signed=true; Datalength=2]

図 5. RPG ビジネス・オブジェクトの例

RPG ビジネス・オブジェクトは動詞 CALLRPG または CALLPGM のいずれかで構成され、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。CALLRPG は RPG プログラムを呼び出すために使用され、CALLPGM は PGM (RPG、COBOL、および Java プログラムを含む) を呼び出すために使用されます。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 (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	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1			ParamType=Input	
1.1	1.1	UserName	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255	Rajesh		
1.2	1.2	Password	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255			
1.3	1.3	HostName	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255	abc.in.ibm.com		
1.4	1.4	ObjectEventId	String								
2	2	☐ MultiRecord	multi_child	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	N			NumRecords=4	
2.1	2.1	NAME1	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10	
2.2	2.2	SERIAL1	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10	
2.3	2.3	DEPT1	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10	
2.4	2.4	ObjectEventId	String								
3	3	ObjectEventId	String								
4	4			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		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	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	1			ParamType=Input
1.1	1.1	UserName	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255	Rajesh	
1.2	1.2	Password	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255		
1.3	1.3	HostName	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		255	abc.in.ibm.com	
1.4	1.4	ObjectEventId	String							
2	2	☐ MultiRecord	multi_child	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	N			NumRecords=4
2.1	2.1	NAME1	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10
2.2	2.2	SERIAL1	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10
2.3	2.3	DEPT1	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		ParamType=OUTPUT;DataLength=10
2.4	2.4	ObjectEventId	String							
3	3	ObjectEventId	String							
4	4			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		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・ビジネス・オブジェクトの処理

コネクタは、統合ブローカーと AS/400 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、AS/400 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 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を構成します。Business Object Designer ODA の詳細については 47 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。

注: どの動詞名にも大/小文字の区別はありません。例えば、動詞 GetQueue は、GETQUEUE や GetQueue と入力しても構いません。

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:

```
Verb=GETQUEUE
BO LEVEL ASI
QSYS.LIB/MYLIB.LIB/MYQUEUE.DTAQ

Connection Instance
  HostName=ibm.siberia.in.com
  UserName=Prapulla
  Password=Prapulla

Attribute1
```

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、コネクタを始動するとコネクタ・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	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	user	
2	PassWord	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	passwd	
3	HostName	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	iseries.server.com	
4	Verb	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	Create	
5	DataQueueName	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	IQSYS.LIB/PLIB.LIB/TESTDQ1.DTAQ	
6	BusObjName	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	SamplePollBO	
7	ObjectEventId	String							
8			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		

図 10. 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 Attrib	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	TestField	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		100		
2	2	ObjectEventId	String							
3	3			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255		

図 11. 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	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	user		
2	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	passwd		
3	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	iseries.machine.com		
4	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	Create		
5	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	IQSYS.LIB/PNPLIB.LIB/TESTDQ2KEY.DTAQ		
6	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	SamplePollKey		
7	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255	Key1		
8	String								
9		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			

図 12. 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。

	Pos	Name	Type	Key	Foreign Key	Required Attribute	Cardinality	Maximum Length	Default Value	Application Specific Information
1	1	TestKey	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		10		
2	2	TestField	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		90		
3	3	ObjectEventId	String							
4	4			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		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

ビジネス・オブジェクト属性のプロパティの指定

iSeries コネクタには、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があります。このセクションでは、コネクタがこれらのプロパティ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

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、入力パラメータに設定されていないときには、コネクタ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	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 AS/400 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
AS/400 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
AS/400 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
AS/400 形式の符号付き 2 バイト浮動小数点数	Float	AS400Float4

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
AS/400 形式の符号付き 4 バイト浮動小数点数	Double	AS400Float8
AS/400 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
AS/400 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
AS/400 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
AS/400 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』
- 『ビジネス・オブジェクト定義の生成』
- 53 ページの『ビジネス・オブジェクト情報の指定』
- 55 ページの『ビジネス・オブジェクトのアップロード』

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要素を「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 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 RPG および 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 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 を作成し、それらをサポートするようにコネクタを構成します。

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 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 の起動および使用方法については、「*IBM WebSphere Business Integration Adapters* ビジネス・オブジェクト開発ガイド」を参照してください。

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。

- UNIX - start_iSeriesODA.sh

Business Object Designer を使用して、iSeries ODA を選択、構成、および実行します。Business Object Designer 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 の実行

Business Object Designer 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 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。)

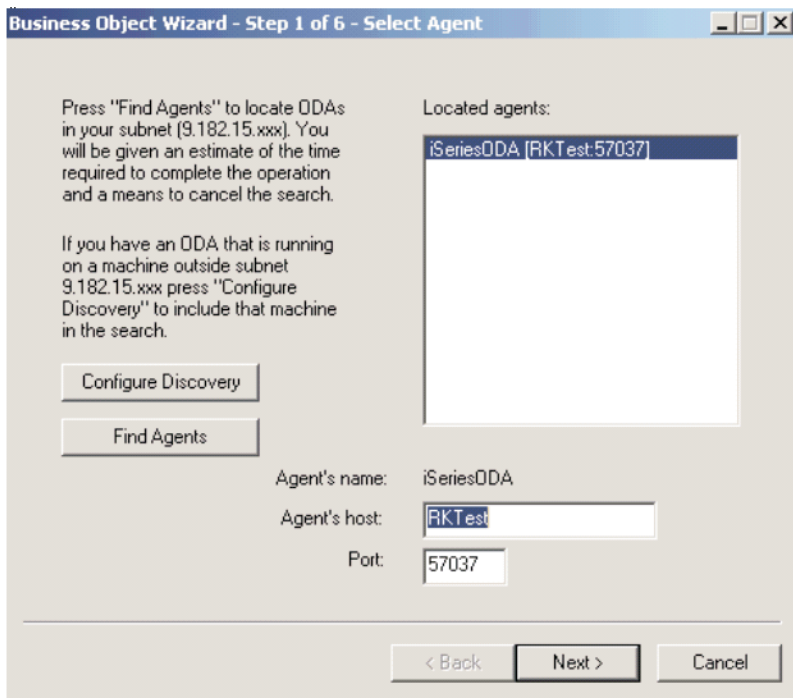


図 14. 「エージェントの選択」ウィンドウ

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。

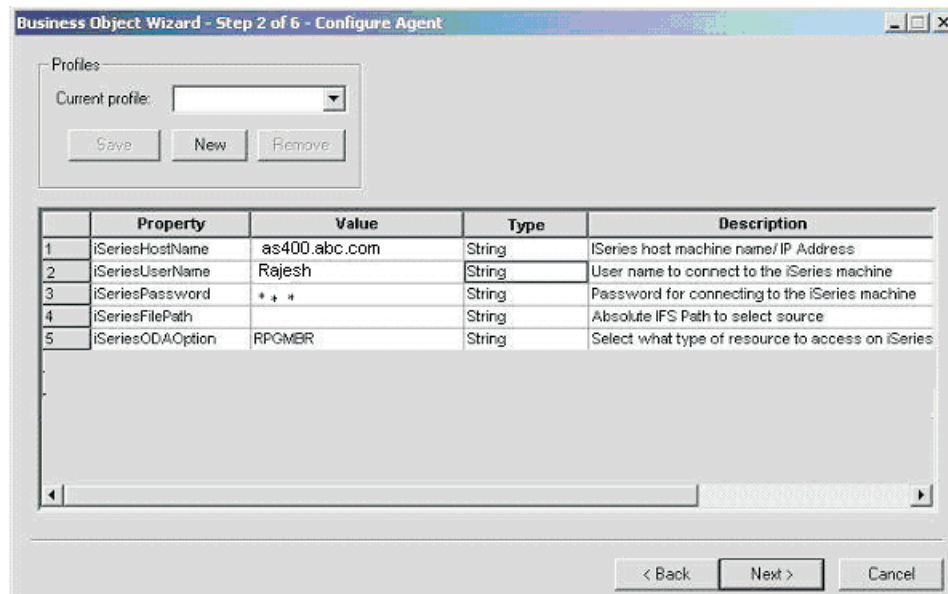


図 15. 「エージェントの構成」ウィンドウ

この画面で設定するプロパティについては、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・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

プロパティ名	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
iSeriesFilePath	/QSYS.LIB/	String	選択ソースへの絶対 IFS パス

表7. 「エージェントの構成」プロパティ (続き)

プロパティ名	デフォルト値	タイプ	説明
iSeriesODAOption		String	<p>iSeries 上でアクセスするリソースのタイプ。現在は、RPGMBR、RPGLEMBR、およびDTAQ の 3 つのオプションがあります。RPGMBR が選択された場合、入力された IFSFiles は RPG 形式のソース・プログラムと見なされます。RPGLE が入力された場合、選択された IFSFiles は RPGLE 形式のソース・プログラムと見なされます。</p> <p>DTAQ は、選択された IFS ファイルが DataQueue であること、つまり順次またはキー付きデータ・キューであることを示します。</p> <p>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。</p>

注: 異なるレコード長の RPG および RPGLE ソース・プログラムが言語ごとのフォーマット仕様としてサポートされています。

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には、既存プロファイルを選択できます。49 ページの表7 に定義されているように、各プロパティの値を入力します。

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、RPG または 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 iSeriesODAOOption エージェントのプロパティ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。

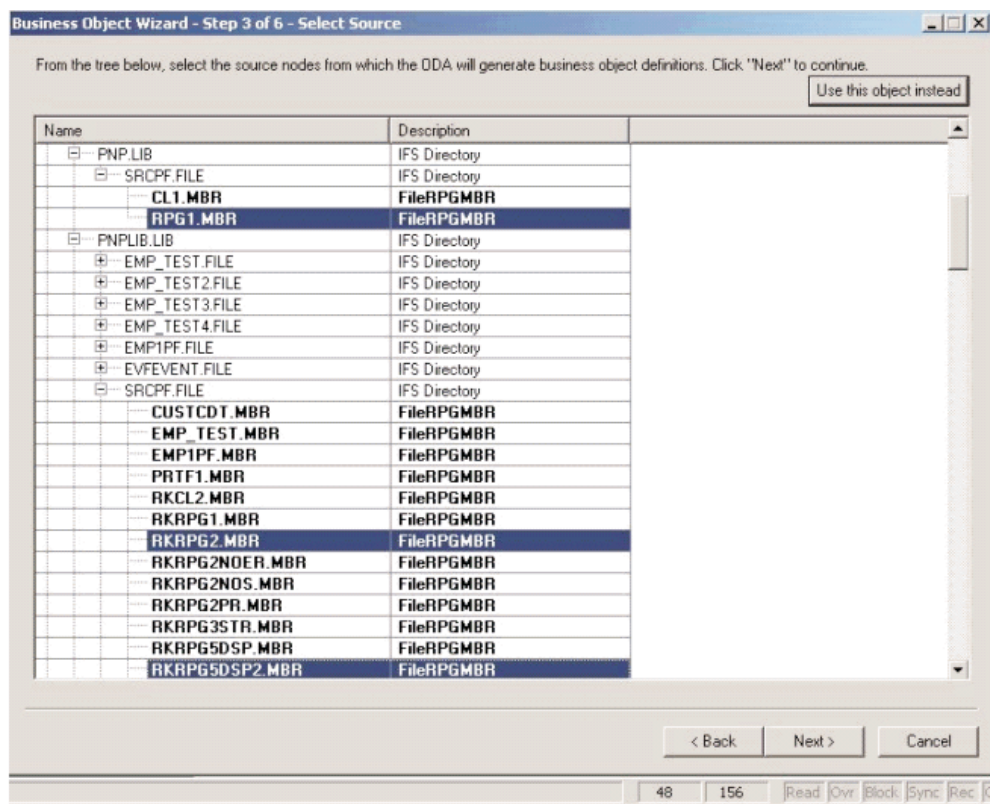


図 16. 「ソースの選択」ウィンドウ

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。

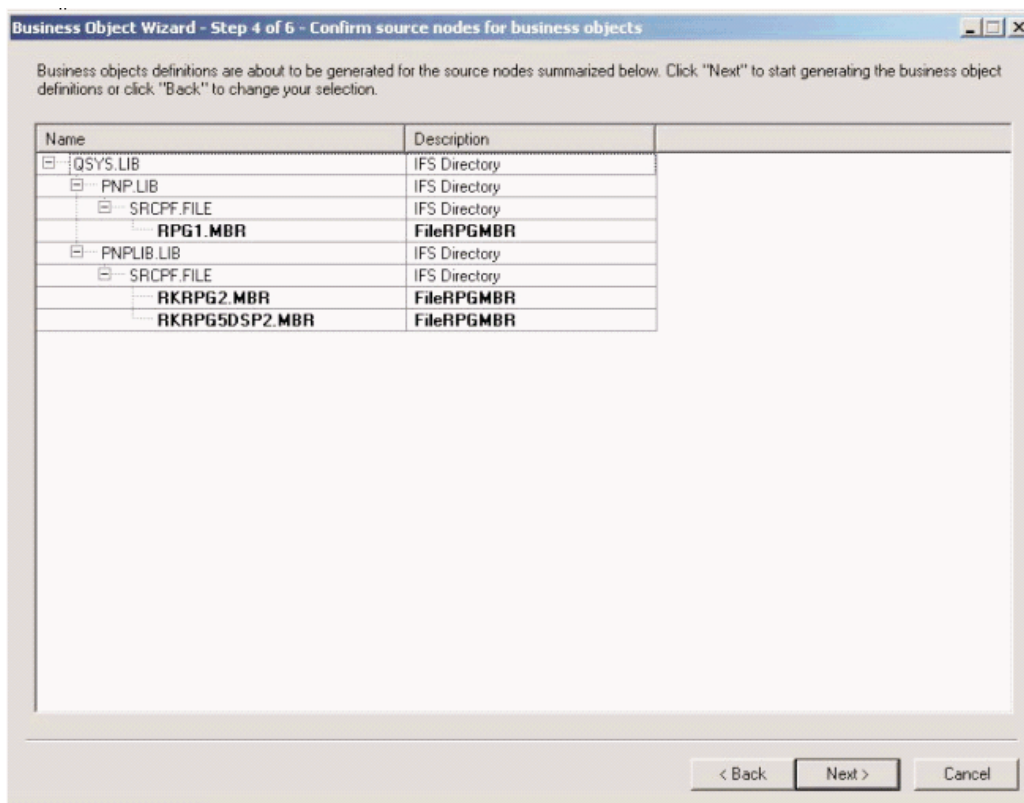


図 17. 「ソース・ノードの確認」 ウィンドウ

「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 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。

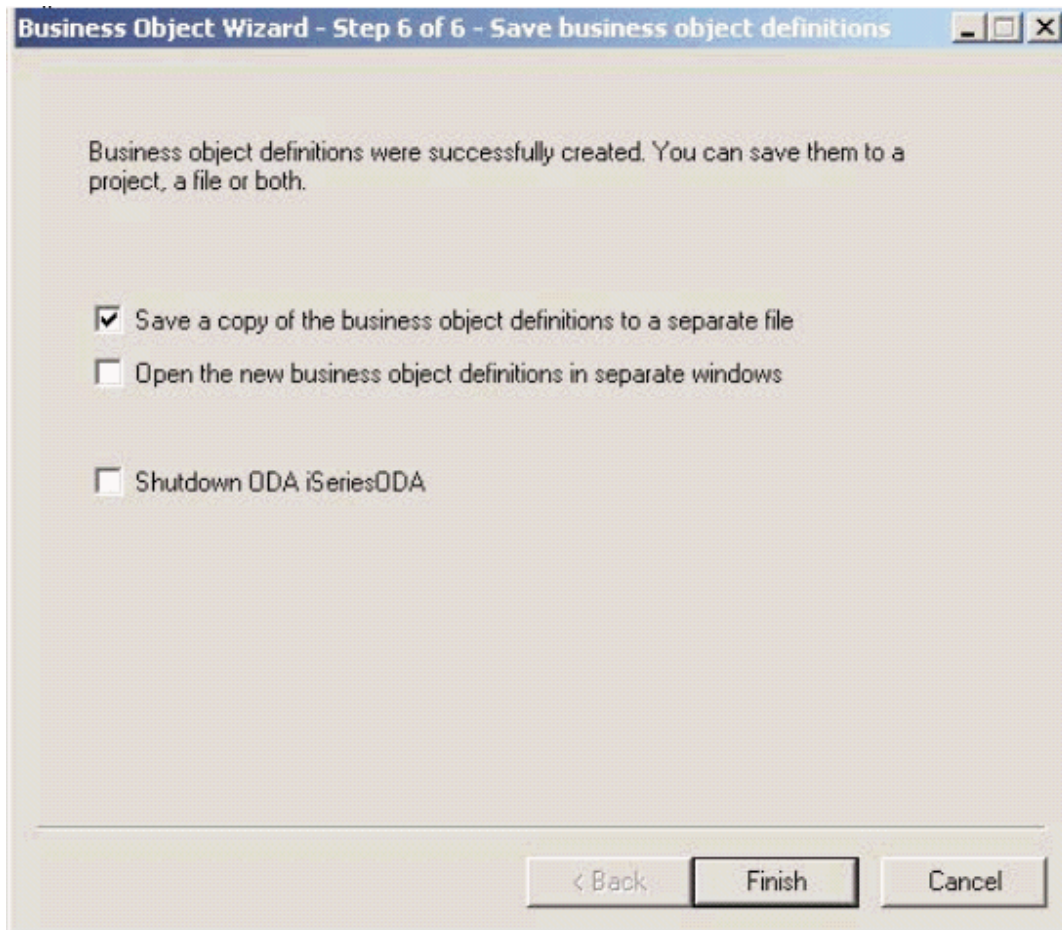


図 18. 「ビジネス・オブジェクト定義の保管」ウィンドウ

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 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、35 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 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、44 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。

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

図 19. 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 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) 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。

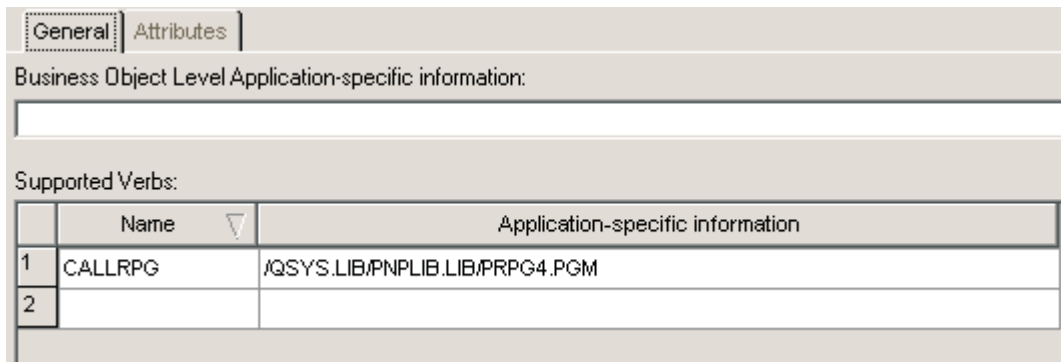
- コネクタがビジネス・オブジェクトを処理するために属性が必要かどうか。必要な場合は、「必要」チェック・ボックスをクリックします。
- 属性の最大長を、「最大長」列に表示される値と異なる値にするかどうか。
- 属性にデフォルト値を設定するかどうか。設定する場合は、「デフォルト」列に値を入力します。

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、44 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・クラスの名前が含まれます。コネクタはこの情報を使用して、プロキシ・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 (コネクタがサーバーとしても実行される場合)、コネクタ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。



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

図 20. 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、作成が終わったら必ず、統合ブローカーにアップロードする必要があります。アップロードの処理は、WebSphere InterChange Server、WebSphere MQ Integrator Broker、または WebSphere Application Server のいずれが実行されているかによって異なります。

- WebSphere InterChange Server: ビジネス・オブジェクト定義ファイルをローカル・マシンに保管した場合、それらをサーバー上のリポジトリにアップロードする必要があるときには、「*WebSphere InterChange Server* 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。
- WebSphere MQ Integrator Broker: ビジネス・オブジェクト定義を Business Object Designer の外部の統合ブローカーにエクスポートする必要があります。詳細については「*WebSphere MQ Integrator Broker* 用インプリメンテーション・ガイド」を参照してください。

- WebSphere Application Server: 詳細については、「アダプター実装ガイド (WebSphere Application Server)」を参照してください。

第 6 章 トラブルシューティングおよびエラー処理

この章では、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、コネクター標準構成プロパティ `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 `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 `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。 <ul style="list-style-type: none"> 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。 <ul style="list-style-type: none"> 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	<p>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。</p> <ul style="list-style-type: none"> • コネクタの初期設定を示す。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・コンフィギュレーター・プロパティの値などが含まれます。 • コネクタ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・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

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

- WebSphere InterChange Server (ICS)
- WebSphere MQ Integrator、WebSphere MQ Integrator Broker、および WebSphere Business Integration Message Broker (これらをまとめて WebSphere Message Brokers (Connector Configurator では WMQI と表記されます) と呼びます)。
- Information Integrator (II)
- WebSphere Application Server (WAS)

アダプターが DB2 Information Integrator をサポートしている場合は、WMQI オプションと DB2 II 標準プロパティを使用します (63 ページの表 9 の「注」列を参照してください)。

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

このコネクタ固有のプロパティの詳細については、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

このリリースでは、次の標準プロパティが追加されました。

- BOTrace

標準コネクタ・プロパティの概要

コネクタには 2 つのタイプの構成プロパティがあります。

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

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

このセクションでは、Connector Configurator の始動方法とすべてのプロパティに共通する特性を説明します。コネクタ固有の構成プロパティについては、該当するアダプターのユーザーズ・ガイドを参照してください。

Connector Configurator の始動

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

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

UNIX 上で動作するコネクターのコネクター・プロパティを設定する場合は、Windows マシン上で System Manager を起動し、UNIX の統合ブローカーに接続してから、コネクター用の Connector Configurator を開く必要があります。

構成プロパティ値の概要

コネクターは、以下の順序に従ってプロパティの値を決定します。

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

プロパティ・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・タイプの長さには制限がありません。INTEGER タイプの長さは、アダプターが稼働しているサーバーによって決まります。

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

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

標準コネクター・プロパティには、以下の 4 種類の更新メソッドがあります。

• 動的

変更を System Manager に保管すると、新規値が即時に有効になります。ただし、例えば WebSphere Message Broker のいずれかで稼働している場合など、コネクターがスタンドアロン・モードである (System Manager から独立している) 場合は、構成ファイルでのみプロパティを変更できます。この場合、動的更新は実行できません。

• エージェント再始動 (ICS のみ)

コネクター・エージェントを停止して再始動しなければ、新規値が有効になりません。

• コンポーネント再始動

System Manager でコネクターを停止してから再始動しなければ、新規値が有効になりません。エージェント・プロセスまたはサーバー・プロセスを停止、再始動する必要はありません。

- **システム再始動**

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

特定のプロパティの更新方法を確認するには、「Connector Configurator」ウィンドウ内の「更新メソッド」列を参照するか、63 ページの表9 の「更新メソッド」列を参照してください。

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

- **ReposController**

プロパティはコネクタ・コントローラーに常駐し、そこでのみ有効になります。エージェント側で値を変更した場合、コントローラーには影響しません。

- **ReposAgent**

プロパティはエージェントに常駐し、そこでのみ有効になります。プロパティによっては、ローカル構成がこの値よりも優先されます。

- **LocalConfig**

プロパティはコネクタの構成ファイルに常駐し、構成ファイルを通じてのみ動作します。コントローラーは、プロパティの値を変更することができず、システムを再配置してコントローラーを明示的に更新するまで、構成ファイルに加えられた変更を認識しません。

標準プロパティのクイック・リファレンス

表9 は、標準コネクタ構成プロパティの早見表です。すべてのコネクタで、これらすべてのプロパティが必要なわけではありません。統合ブローカーによってプロパティの設定は異なります。

各プロパティの説明は、表の後のセクションを参照してください。

注: 表9の「注」列にある「RepositoryDirectory が <REMOTE> に設定されている」という句は、ブローカーが InterChange Server であることを示しています。ブローカーが WMQI または WAS の場合には、リポジトリ・ディレクトリーは <ProductDir>%repository に設定されます。

表9. 標準構成プロパティの要約

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

表9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
AdminOutQueue	有効な JMS キュー名	<CONNECTORNAME> /ADMINOUTQUEUE	コンポー ネント 再始動	このプロパティは、 DeliveryTransport の値が JMS である場合のみ 有効です。
AgentConnections	1 から 4	1	コンポー ネント 再始動	このプロパティは、 DeliveryTransport の値が MQ または IDL で、 Repository Directory の値が <REMOTE> に設定され、 BrokerType の値が ICS である場合のみ有効です。
AgentTraceLevel	0 から 5	0	ブローカーが ICS である 場合は動的で、 それ以外の 場合は コンポー ネント 再始動	
ApplicationName	アプリケーション名	コネクター・ アプリケーション名 として指定された値	コンポー ネント 再始動	
BiDi.Application	次の双方向属性の 有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポー ネント 再始動	このプロパティは、 BiDi.Transformation の値が true である場合のみ 有効です。
BiDi.Broker	次の双方向属性の 有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポー ネント 再始動	このプロパティは、 BiDi.Transformation の値が true である場合のみ 有効です。 BrokerType の値が ICS である場合、 プロパティは 読み取り専用です。
BiDi.Metadata	次の双方向属性の 有効な組み合わせ 最初の文字: I、V 2 番目の文字: L、R 3 番目の文字: Y、N 4 番目の文字: S、N 5 番目の文字: H、C、N	ILYNN (5 文字)	コンポー ネント 再始動	このプロパティは、 BiDi.Transformation の値が true である場合 のみ有効です。
BiDi.Transformation	true または false	false	コンポー ネント 再始動	このプロパティは、 BrokerType の値が WAS ではない場合のみ 有効です。

表9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
BOTrace	none または keys または full	none	エージェント再始動	このプロパティは、AgentTraceLevel の値が 5 より低い場合のみ有効です。
BrokerType	ICS、WMQI、WAS	ICS	コンポーネント再始動	
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	このプロパティは、C++ コネクタに対してのみ有効です。
CommonEventInfrastructure	true または false	false	コンポーネント再始動	
CommonEventInfrastructureURL	例えば corbaloc:iiop:host:2809 などの URL スtring	デフォルト値なし。	コンポーネント再始動	このプロパティは、CommonEventInfrastructure の値が true である場合のみ有効です。
ConcurrentEventTriggeredFlows	1 から 32,767	1	コンポーネント再始動	このプロパティは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ContainerManagedEvents	ブランクまたは JMS	ブランク	コンポーネント再始動	このプロパティは、Delivery Transport の値が JMS である場合のみ有効です。
ControllerEventSequencing	true または false	true	動的	このプロパティは、Repository Directory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ControllerStoreAndForwardMode	true または false	true	動的	このプロパティは、Repository Directory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
ControllerTraceLevel	0 から 5	0	動的	このプロパティは、RepositoryDirectory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。

表 9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
DeliveryQueue	任意の有効な JMS キュー名	<CONNECTORNAME>/DELIVERYQUEUE	コンポーネント再始動	このプロパティは、Delivery Transport の値が JMS である場合のみ有効です。
DeliveryTransport	MQ、IDL、または JMS	RepositoryDirectory の値が <REMOTE> である場合は IDL で、それ以外の場合は JMS	コンポーネント再始動	RepositoryDirectory の値が <REMOTE> ではない場合、このプロパティで有効な値は JMS だけです。
DuplicateEventElimination	true または false	false	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
EnableOidForFlowMonitoring	true または false	false	コンポーネント再始動	このプロパティは、BrokerType の値が ICS である場合のみ有効です。
FaultQueue	任意の有効なキュー名。	<CONNECTORNAME>/FAULTQUEUE	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory、CxCommon.Messaging.jms.SonicMQFactory、または任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.ListenerConcurrency	1 から 32767	1	コンポーネント再始動	このプロパティは、jms.TransportOptimized の値が true である場合のみ有効です。
jms.MessageBrokerName	jms.FactoryClassName の値が IBM である場合は、crossworlds.queue.manager を使用します。	crossworlds.queue.manager	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.Password	任意の有効なパスワード		コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
jms.TransportOptimized	true または false	false	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS で BrokerType の値が ICS である場合のみ有効です。
jms.UserName	任意の有効な名前		コンポーネント再始動	このプロパティは、Delivery Transport の値が JMS である場合のみ有効です。

表9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
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	このプロパティは、Repository Directoryの値が <REMOTE>に設定され、BrokerTypeの値が ICS である場合のみ有効です。
JvmMaxNativeStackSize	スタックのサイズ (キロバイト単位)	128k	コンポーネント再始動	このプロパティは、Repository Directoryの値が <REMOTE>に設定され、BrokerTypeの値が ICS である場合のみ有効です。
JvmMinHeapSize	ヒープ・サイズ (メガバイト単位)	1m	コンポーネント再始動	このプロパティは、Repository Directoryの値が <REMOTE>に設定され、BrokerTypeの値が ICS である場合のみ有効です。
ListenerConcurrency	1 から 100	1	コンポーネント再始動	このプロパティは、DeliveryTransportの値が MQ である場合のみ有効です。
Locale	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	このプロパティは、Repository Directoryの値が <REMOTE>に設定され、BrokerTypeの値が ICS である場合のみ有効です。
MaxEventCapacity	1 から 2147483647	2147483647	動的	このプロパティは、Repository Directoryの値が <REMOTE>に設定され、BrokerTypeの値が ICS である場合のみ有効です。
MessageFileName	有効なファイル名	InterchangeSystem.txt	コンポーネント再始動	
MonitorQueue	任意の有効なキュー名	<CONNECTORNAME>/MONITORQUEUE	コンポーネント再始動	このプロパティは、DuplicateEventEliminationの値が true で ContainerManagedEventsに値がない場合のみ有効です。

表9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
OADAutoRestartAgent	true または false	false	動的	このプロパティは、Repository Directory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
OADMaxNumRetry	正整数	1000	動的	このプロパティは、Repository Directory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
OADRetryTimeInterval	正整数 (単位: 分)	10	動的	このプロパティは、Repository Directory の値が <REMOTE> に設定され、BrokerType の値が ICS である場合のみ有効です。
PollEndTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒)	10000	ブローカーが ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
PollQuantity	1 から 500	1	エージェント再始動	このプロパティは、ContainerManagedEvents の値が JMS である場合のみ有効です。
PollStartTime	HH = 0 から 23 MM = 0 から 59	HH:MM	コンポーネント再始動	
RepositoryDirectory	ブローカーが ICS である場合は <REMOTE> で、それ以外の場合は有効な任意のローカル・ディレクトリー	ICS の場合、値は <REMOTE> に設定されます。 WMQI および WAS の場合、値は <ProductDir %repository> です。	エージェント再始動	
RequestQueue	有効な JMS キュー名	<CONNECTORNAME>/REQUESTQUEUE	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
ResponseQueue	有効な JMS キュー名	<CONNECTORNAME>/RESPONSEQUEUE	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。

表9. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
RestartRetryCount	0 から 99	7	ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
RestartRetryInterval	1 から 2147483647 までの分単位の値。	1	ICS である場合は動的で、それ以外の場合はコンポーネント再始動	
ResultsSetEnabled	true または false	false	コンポーネント再始動	DB2II をサポートするコネクタートよってのみ使用されます。 このプロパティは、DeliveryTransport の値が JMS で BrokerType の値が WMQI である場合のみ有効です。
ResultsSetSize	正整数	0 (結果セットのサイズが無制限であることを意味します)	コンポーネント再始動	DB2II をサポートするコネクタートよってのみ使用されます。 このプロパティは、ResultsSetEnabled の値が true である場合のみ有効です。
RHF2MessageDomain	mrml または xml	mrml	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS で WireFormat の値が CwXML である場合のみ有効です。
SourceQueue	有効な WebSphere MQ キュー名	<CONNECTORNAME>/SOURCEQUEUE	エージェント再始動	このプロパティは、ContainerManagedEvents の値が JMS である場合のみ有効です。
SynchronousRequest Queue	任意の有効なキュー名。	<CONNECTORNAME>/SYNCHRONOUSREQUEST QUEUE	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
SynchronousRequest Timeout	0 から任意の数まで (ミリ秒)	0	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。
SynchronousResponse Queue	任意の有効なキュー名	<CONNECTORNAME>/SYNCHRONOUSRESPONSE QUEUE	コンポーネント再始動	このプロパティは、DeliveryTransport の値が JMS である場合のみ有効です。

表 9. 標準構成プロパティの要約 (続き)

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

標準プロパティ

このセクションでは、標準コネクタ構成プロパティについて説明します。

AdapterHelpName

AdapterHelpName プロパティは、コネクタ固有の全般ヘルプ・ファイルがあるディレクトリの名前です。ディレクトリは <ProductDir>%bin%Data%App%Help にある必要があります。また、少なくとも言語ディレクトリ enu_usa を含んでいる必要があります。ロケールによっては、その他のディレクトリが含まれることもあります。

有効またはブランクである場合、デフォルト値はテンプレート名です。

AdminInQueue

AdminInQueue プロパティは、統合ブローカーからコネクタへ管理メッセージが送信されるときに使用されるキューを指定します。

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

AdminOutQueue

AdminOutQueue プロパティは、コネクタから統合ブローカーへ管理メッセージが送信されるときに使用されるキューを指定します。

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

AgentConnections

AgentConnections プロパティは、ORB が初期設定するときオープンされる ORB (Object Request Broker (オブジェクト・リクエスト・ブローカー)) 接続の数を制御します。

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

このプロパティのデフォルト値は 1 です。

AgentTraceLevel

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

デフォルトはコネクタの名前です。

BiDi.Application

BiDi.Application プロパティは、このアダプターによってサポートされるビジネス・オブジェクト形式で外部アプリケーションからアダプターに送信されるデータに対して、双方向フォーマットを指定します。このプロパティは、アプリケーション・データの双方向属性を定義します。属性は次のとおりです。

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

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

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Broker

BiDi.Broker プロパティは、サポートされるビジネス・オブジェクトの形式でアダプターから統合ブローカーへ送信されるデータに対して双方向スクリプト・フォーマットを指定します。これは、前述の BiDi.Application にリストされているようなデータの双方向属性を定義します。

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

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Metadata

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

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

デフォルト値は ILYNN (暗黙、左から右、オン、オフ、公称) です。

BiDi.Transformation

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

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

デフォルト値は false です。

BOTrace

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

注: これは、AgentTraceLevel プロパティが 5 より小さく設定されている場合のみ適用されます。

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

- ビジネス・オブジェクトの属性をすべてダンプする場合は、-xBOTrace=Full と入力します。
- ビジネス・オブジェクトのキーだけをダンプする場合は、-xBOTrace=Keys と入力します。
- ビジネス・オブジェクト属性のダンプを使用不可にする場合は、-xBOTrace=None と入力します。

デフォルト値は false です。

BrokerType

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

CharacterEncoding

CharacterEncoding プロパティは、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、このプロパティは使用しません。 C++ コネクターは、このプロパティに対して値 `ascii7` を使用します。

デフォルトでは、サポートされる文字エンコードのサブセットだけが表示されます。リストに、サポートされる他の値を追加するには、製品ディレクトリーにある `¥Data¥Std¥stdConnProps.xml` ファイルを手動で変更する必要があります (<ProductDir>)。詳細については、本書の Connector Configurator の付録を参照してください。

CommonEventInfrastructure

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

デフォルト値は `false` です。

CommonEventInfrastructureContextURL

CommonEventInfrastructureContextURL は、 Common Event Infrastructure (CEI) サーバー・アプリケーションを実行する WAS サーバーにアクセスするために使用します。このプロパティは、使用する URL を指定します。

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

デフォルト値はブランク・フィールドです。

ConcurrentEventTriggeredFlows

ConcurrentEventTriggeredFlows プロパティは、コネクターがイベントのデリバリー時に並行処理できるビジネス・オブジェクトの数を決定します。この属性の値を、並行してマップおよびデリバリーされるビジネス・オブジェクトの数に設定します。例えば、このプロパティの値を 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、次のプロパティを構成する必要があります。

- **Maximum number of concurrent events** プロパティの値を増やして、コラボレーションが複数のスレッドを使用できるように構成します。
- 宛先アプリケーションのアプリケーション固有コンポーネントが複数の要求を並行して実行できるように構成する必要があります。つまり、このコンポーネントは、マルチスレッド化するか、またはコネクタ・エージェント並列処理が使用可能で複数プロセスに対応するように構成する必要があります。 **Parallel Process Degree** 構成プロパティは、1 より大きい値に設定する必要があります。

ConcurrentEventTriggeredFlows プロパティは、単一スレッドで順次実行されるコネクタ・ポーリングには影響しません。

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

デフォルト値は 1 です。

ContainerManagedEvents

ContainerManagedEvents プロパティにより、JMS イベント・ストアを使用する JMS 対応コネクタが、保証付きイベント・デリバリーを提供できるようになります。保証付きイベント・デリバリーでは、イベントはソース・キューから除去され、1 つの JMS トランザクションとして宛先キューに配置されます。

このプロパティが JMS に設定されている場合は、保証付きイベント・デリバリーを使用できるように次のプロパティ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の値を設定するには、**Connector Configurator** の「データ・ハンドラー」タブを使用します。

これらのプロパティはアダプター固有ですが、いくつかの値の例を次に示します。

- **MimeType** = text/xml
- **DHClass** = com.crossworlds.DataHandlers.text.xml
- **DataHandlerConfigMOName** = M0_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** プロパティが値 JMS に設定されている場合のみ表示されます。

注: ContainerManagedEvents を JMS に設定した場合、コネクターはその pollForEvents() メソッドを呼び出さないため、このメソッドの機能は使用不可になります。

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

デフォルト値はありません。

ControllerEventSequencing

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

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

デフォルト値は true です。

ControllerStoreAndForwardMode

ControllerStoreAndForwardMode プロパティは、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。

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

デフォルト値は true です。

ControllerTraceLevel

ControllerTraceLevel プロパティは、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。

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

デフォルト値は 0 です。

DeliveryQueue

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

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

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

DeliveryTransport

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

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

RepositoryDirectory プロパティの値が MQ または IDL である場合、コネクタは、CORBA IIOP を使用してサービス呼び出し要求と管理メッセージを送信します。

DeliveryTransport プロパティの値が MQ である場合、アダプター開始スクリプトにコマンド行パラメーター WhenServerAbsent を設定して、InterChange Server のシャットダウン時にアダプターを一時停止するかシャットダウンするかを示すことができます。

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

WebSphere MQ および IDL

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

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

- 非同期 (ASYNC) 通信:
WebSphere MQ を使用すると、アプリケーション固有のコンポーネントは、サーバーが利用不能である場合でも、イベントをポーリングして永続的に格納することができます。
- サーバー・サイド・パフォーマンス:
WebSphere MQ を使用すると、サーバー・サイドのパフォーマンスが向上します。最適化モードでは、WebSphere MQ はイベントへのポインターのみをリポジトリ・データベースに格納するので、実際のイベントは WebSphere MQ キュー

一内に残ります。これにより、サイズが大きい可能性のあるイベントをリポジトリ・データベースに書き込むことがなくなります。

- エージェント・サイド・パフォーマンス:
WebSphere MQ を使用すると、アプリケーション固有のコンポーネント側のパフォーマンスが向上します。WebSphere MQ を使用すると、コネクタ・ポーリング・スレッドは、イベントを取得した後、コネクタ・キューにそのイベントを入れ、次のイベントを取得します。この方法は IDL よりも高速です。IDL ではコネクタ・ポーリング・スレッドが必要で、コネクタ・ポーリング・スレッドは、イベントを抽出してネットワーク経由でサーバー・プロセスに到達し、イベントをリポジトリ・データベースに永続的に保管して、次のイベントを抽出します。

JMS

JMS トランスポート・メカニズムは、Java Messaging Service (JMS) による、コネクタとクライアント・コネクタ・フレームワークの間の通信を可能にします。

JMS をデリバリー・トランスポートとして選択した場合は、

`jms.MessageBrokerName`、`jms.FactoryClassName`、`jms.Password`、および `jms.UserName` などの追加の JMS プロパティが Connector Configurator 内にリストされます。プロパティ `jms.MessageBrokerName` および `jms.FactoryClassName` は、このトランスポートで必須です。

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

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

この環境では、WebSphere MQ クライアント内でメモリーが使用されるため、(サーバー側の) コネクタ・コントローラーと (クライアント側の) コネクタの両方を始動するのは困難な場合があります。ご使用のシステムのプロセス・ヒープ・サイズが 768MB 未満である場合は、次の変数とプロパティを設定します。

- `CWSharedEnv.sh` スクリプト内で `LDR_CNTRL` 環境変数を設定します。

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

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

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

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

DuplicateEventElimination

このプロパティーの値が `true` である場合は、JMS 対応コネクタによって、重複イベントがデリバリー・キューに配信されないようにすることができます。コネクタの開発時にこの機能を使用するには、コネクタに対し、アプリケーション固有のコード内でビジネス・オブジェクトの `ObjectEventId` 属性として固有のイベント ID が設定されている必要があります。

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

デフォルト値は `false` です。

EnableOidForFlowMonitoring

このプロパティーの値が `true` である場合は、アダプター・ランタイムは、着信 `ObjectEventID` をフロー・モニター用外部キーとしてマーク付けします。

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

デフォルト値は `false` です。

FaultQueue

コネクタでメッセージ処理中にエラーが発生すると、コネクタは、そのメッセージ (および状況標識と問題記述) を `FaultQueue` プロパティーで指定されているキューに移動させます。

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

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

jms.ListenerConcurrency

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

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

デフォルト値は `1` です。

jms.MessageBrokerName

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

リモート・メッセージ・ブローカーに接続する場合、このプロパティに次の値を指定する必要があります。

`QueueMgrName:Channel:HostName:PortNumber`

ここで、以下のように説明されます。

`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ンネルです。

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

デフォルト値は 10 です。

jms.Password

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

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

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

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

デフォルト値は false です。

jms.UserName

jms.UserName プロパティは、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。

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

デフォルト値は 128M です。

JvmMaxNativeStackSize

JvmMaxNativeStackSize プロパティは、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。

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

デフォルト値は 128K です。

JvmMinHeapSize

JvmMinHeapSize プロパティは、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。

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

デフォルト値は 1M です。

ListenerConcurrency

ListenerConcurrency プロパティは、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。

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

デフォルト値は 1 です。

Locale

Locale プロパティは、言語コード、国または地域、および、希望する場合には、関連した文字コード・セットを指定します。このプロパティの値は、データの照合やソート順、日時フォーマット、通貨規格で使用されている記号などの国/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 バイト文字の言語コード (小文字) です。

`TT` は、2 バイト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 (オプション)。

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

コネクタが国際化に対応していない場合、このプロパティの有効な値は `en_US` のみです。特定のコネクタがグローバル化に対応しているかどうかを判別するには、そのアダプターのユーザズ・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 プロパティは、統合ブローカーのログ宛先にエラーを記録するかどうかを指定します。

ログ宛先にログを記録すると、電子メール通知もオンになります。これにより、エラーまたは致命的エラーが発生すると、InterchangeSystem.cfg ファイルの MESSAGE_RECIPIENT の値で指定された宛先に対して電子メール・メッセージが生成されます。例えば、LogAtInterChangeEnd の値が true である場合にコネクタ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。

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

デフォルト値は false です。

MaxEventCapacity

MaxEventCapacity プロパティは、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。このプロパティは、フロー制御フィーチャーによって使用されます。

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

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

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

デフォルト値は `InterchangeSystem.txt` です。

MonitorQueue

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

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

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

OADAutoRestartAgent

OADAutoRestartAgent プロパティは、コネクタが自動再始動機能およびリモート再始動機能を使用するかどうかを指定します。この機能では、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を再始動したり、System Monitor からリモート・コネクタを始動したりします。

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

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

デフォルト値は false です。

OADMaxNumRetry

OADMaxNumRetry プロパティは、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) がコネクタの再始動を自動的に試行する回数の最大数を指定します。このプロパティを有効にするには、OADAutoRestartAgent プロパティを true に設定する必要があります。

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

デフォルト値は 1000 です。

OADRetryTimeInterval

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

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

デフォルト値は 10 です。

PollEndTime

PollEndTime プロパティは、イベント・キューのポーリングを停止する時刻を指定します。形式は HH:MM です。ここで、HH は 0 から 23 時を表し、MM は 0 から 59 分を表します。

このプロパティには必ず有効な値を指定してください。デフォルト値は HH:MM で、値が指定されていないので、これは変更する必要があります。

アダプター・ランタイムが次のことを検出した場合の処理は以下のとおりです。

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

PollFrequency プロパティで構成された値を使用してポーリングが実行されます。

PollFrequency

PollFrequency プロパティは、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 プロパティで指定された間隔にわたって遅延します。
- このサイクルを繰り返します。

このプロパティで有効な値は次のとおりです。

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

デフォルト値は 10000 です。

重要: 一部のコネクタでは、このプロパティの使用が制限されています。このようなコネクタ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 プロパティは、コネクタがアプリケーションからポーリングする項目の数を指定します。アダプタにコネクタ固有のポーリング数設定プロパティがある場合、標準プロパティの値は、このコネクタ固有のプロパティの設定値によりオーバーライドされます。

このプロパティは、DeliveryTransport プロパティの値が JMS で ContainerManagedEvents プロパティに値がある場合のみ有効です。

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

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

PollStartTime

PollStartTime プロパティは、イベント・キューのポーリングを開始する時刻を指定します。形式は *HH:MM* です。ここで、*HH* は 0 から 23 時を表し、*MM* は 0 から 59 分を表します。

このプロパティには必ず有効な値を指定してください。デフォルト値は *HH:MM* で、値が指定されていないので、これは変更する必要があります。

アダプター・ランタイムが次のことを検出した場合の処理は以下のとおりです。

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

PollFrequency プロパティで構成された値を使用してポーリングが実行されます。

RepositoryDirectory

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

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

統合ブローカーが WebSphere Message Broker または WAS である場合、この値はデフォルトで **<ProductDir>%repository** に設定されます。ただし、任意の有効なディレクトリ名に設定することもできます。

RequestQueue

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

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

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

ResponseQueue

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

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

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

RestartRetryCount

RestartRetryCount プロパティは、コネクタによるコネクタ自体の再始動の試行回数を指定します。このプロパティを並列に接続されたコネクタに対して使用すると、マスター・コネクタのアプリケーション固有コンポーネントがクライアント・コネクタのアプリケーション固有コンポーネントの再始動を試行する回数が指定されます。

デフォルト値は 7 です。

RestartRetryInterval

RestartRetryInterval プロパティは、コネクタによるコネクタ自体の再始動の試行間隔を分単位で指定します。このプロパティを並列にリンクされたコネクタに対して使用すると、マスター・コネクタのアプリケーション固有コンポーネントがクライアント・コネクタのアプリケーション固有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。

このプロパティで指定できる値は、1 から 2147483647 です。

デフォルト値は 1 です。

ResultsSetEnabled

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

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

デフォルト値は **false** です。

ResultsSetSize

ResultsSetSize プロパティは、**Information Integrator** へ戻ることができるビジネス・オブジェクトの最大数を定義します。このプロパティを使用できるのは、アダプターが **DB2 Information Integrator** をサポートしている場合のみです。

このプロパティは、**ResultsSetEnabled** プロパティの値が **true** である場合のみ有効です。

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

RHF2MessageDomain

RHF2MessageDomain プロパティにより、**JMS** ヘッダーのドメイン名フィールドの値を構成できます。**JMS** トランスポートを介してデータを **WebSphere** メッセージ・ブローカーに送信するときに、アダプター・フレームワークにより **JMS** ヘッ

ダー情報、ドメイン名、および固定値 `mrm` が書き込まれます。構成可能ドメイン名を使用すると、WebSphere メッセージ・ブローカーがメッセージ・データをどのように処理するかを追跡できます。

次に、ヘッダーの例を示します。

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

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

指定できる値は `mrm` と `xml` です。デフォルト値は `mrm` です。

SourceQueue

`SourceQueue` プロパティは、`JMS` イベント・ストアを使用する `JMS` 対応コネクタでの保証付きイベント・デリバリーをサポートするコネクタ・フレームワークに、`JMS` ソース・キューを指定します。詳細については、74 ページの『`ContainerManagedEvents`』を参照してください。

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

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

SynchronousRequestQueue

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

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

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

SynchronousRequestTimeout

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

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

デフォルト値は `0` です。

SynchronousResponseQueue

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

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

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

TivoliMonitorTransactionPerformance

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

デフォルト値は false です。

WireFormat

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

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

WsifSynchronousRequestTimeout

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

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

デフォルト値は 0 です。

XMLNamespaceFormat

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

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

デフォルト値は short です。

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]

アダプターの動作方法 4
アプリケーション固有の構成プロパティ
設定 22
インストール後
作業 10
インストール済みファイルの構造 9
エラー処理 57

[カ行]

構成
完了 28
構成ファイル
完成 19
既存ファイルの使用 17
コネクタ固有のテンプレートを使用
して作成 16
新規作成 16
プロパティの設定 19
変更 28
保管 27
コネクタ固有のプロパティ 21
コネクタ固有のプロパティ・テンプレ
ート
作成 13
コネクタの標準構成プロパティ 61
コネクタ・インスタンス
複数作成 31

[サ行]

属性レベルの ASI
指定 53

[タ行]

データ変換
iSeries 用 または AS/400 用の
ToolBox 44
データ・キュー
概要 4
データ・ハンドラー 27

トレース/ログ・ファイルの値
設定 27
トレース・メッセージ 57

[ハ行]

ビジネス・オブジェクト
アップロード 55
概要 35
構造の概要 36
作成および変更 47
サポートされる定義の指定 23
情報の指定 53
処理 4
生成 52
属性プロパティの指定 43
属性レベルのアプリケーション・テキ
ストの指定 44
データ・キュー構造 40
定義の生成 47
RPG プログラム構造 36
ビジネス・オブジェクト・レベル ASI
指定 55
標準コネクタ・プロパティ
設定 20

[マ行]

マップ 25
メタオブジェクト
ポーリングのために構成 41
メタデータ
定義 35
メッセージング 26

[ラ行]

リソース 26
ロギング 57

B

Business Object Designer
実行 48

C

Connector Configurator
概要 11
グローバル化環境における使用 29

Connector Configurator (続き)
始動 12
スタンドアロン・モードで実行 12
System Manager から実行 13

I

iSeries アダプター
インストール 7
インストールおよび関連ファイル 9
概要 1
環境 7
構成 11
コネクタの始動 29
コネクタの停止 31
前提条件 8
動作 5
トラブルシューティング 57
プラットフォーム 8
ブローカーの互換性 7
iSeries および AS/400 システム
概要 1

O

Object Discovery Agent (ODA)
概要 47
始動 47

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

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

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

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

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

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

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

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

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

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

IBM 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。その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

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

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

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

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

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

Intel、Intel ロゴ、Intel Inside、Intel Inside ロゴ、Intel Centrino、Intel Centrino ロゴ、Celeron、Intel Xeon、Intel SpeedStep、Itanium、および Pentium は、Intel Corporation の米国およびその他の国における商標です。

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

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

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

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



WebSphere Business Integration Adapters バージョン 6.0



WebSphere Business Integration Adapter Framework V2.4.0



Printed in Japan