

**IBM WebSphere Business Integration  
Adapters**



## **Adapter for Siebel 2000 ユーザーズ・ガイド**

*バージョン 4.1.x*



**IBM WebSphere Business Integration  
Adapters**



## **Adapter for Siebel 2000 ユーザーズ・ガイド**

*バージョン 4.1.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、81ページの『特記事項』に記載されている情報をお読みください。

本書は、IBM WebSphere Business integration Adapter for Siebel 2000 バージョン 4.1.x に適用されます。

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

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

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

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

(URL は、変更になる場合があります)

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

原 典： IBM WebSphere Business Integration Adapters  
Adapter for Siebel 2000 User Guide  
Version 4.1.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 2004.7

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

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

© Copyright International Business Machines Corporation 1997, 2004. All rights reserved.

© Copyright IBM Japan 2004

# 目次

本書について	v
対象読者	v
関連文書	v
表記上の規則	vi
<b>本リリースの新機能</b>	<b>vii</b>
リリース 4.1.x の新機能	vii
リリース 4.0.x での新機能	viii
<b>第 1 章 コネクターの概要</b>	<b>1</b>
COM サポート	1
サポートされているオペレーティング・システム	1
メタデータ主導型設計	1
アーキテクチャー	2
COM コネクタと Siebel との間の通信	3
サポートされている動詞	3
イベント通知	3
ロケール依存型データの処理	4
<b>第 2 章 コネクターのインストールと構成</b>	<b>7</b>
互換性	7
前提条件	8
アダプターと関連ファイルのインストール	8
必要な変更	9
コネクターの構成	13
コネクタ構成プロパティ	19
複数のコネクタ・インスタンスの作成	22
コネクターの始動	23
Windows サービスとしてのコネクターの実行	25
コネクターの停止	25
<b>第 3 章 ビジネス・オブジェクトの開発</b>	<b>27</b>
ビジネス・オブジェクトの構造および関係	27
ビジネス・オブジェクトの動詞処理	28
ビジネス・オブジェクトのアプリケーション固有情報	29
<b>第 4 章 トラブルシューティング</b>	<b>41</b>
Init() での失敗のエラー	41
アプリケーションへの接続不可	41
<b>付録 A. コネクターの標準構成プロパティ</b>	<b>43</b>
新規プロパティと削除されたプロパティ	43
標準コネクタ・プロパティの構成	43
標準プロパティの要約	45
標準構成プロパティ	50
<b>付録 B. Connector Configurator</b>	<b>63</b>
Connector Configurator の概要	63
Connector Configurator の始動	64
System Manager からの Configurator の実行	65

コネクタ固有のプロパティ・テンプレートの作成 . . . . .	65
新規構成ファイルの作成 . . . . .	68
既存ファイルの使用 . . . . .	69
構成ファイルの完成 . . . . .	71
構成ファイル・プロパティの設定 . . . . .	71
構成ファイルの保管 . . . . .	78
構成ファイルの変更 . . . . .	79
構成の完了 . . . . .	79
グローバル化環境における Connector Configurator の使用 . . . . .	80
<b>特記事項 . . . . .</b>	<b>81</b>
プログラミング・インターフェース情報 . . . . .	83
商標 . . . . .	83

---

## 本書について

IBM<sup>(R)</sup> WebSphere<sup>(R)</sup> Business Integration Adapter ポートフォリオは、主要な e-business テクノロジー、エンタープライズ・アプリケーション、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。この製品セットには、ビジネス・プロセス統合のコンポーネントをカスタマイズ、作成、および管理するためのツールやテンプレートが含まれています。

本書では、IBM WebSphere Business Integration Adapter for Siebel 2000 のインストール、構成、およびビジネス・オブジェクト開発について説明します。

---

## 対象読者

この資料の対象読者は、日本で Siebel 2000 を使用し、WebSphere Business Integration システムを使用するコンサルタントとお客様です。本書で説明する情報を利用する際には、次の分野について十分理解していることが前提になっています。

- コネクタ開発
- ビジネス・オブジェクト開発
- Siebel アプリケーション・アーキテクチャー
- Siebel ツール
- Visual Basic

---

## 関連文書

この製品の一連の資料では、すべての WebSphere Business Integration Adapters のインストールに共通の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。

本書では、2 つの資料「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)」と「WebSphere InterChange Server システム・インプリメンテーション・ガイド」が頻繁に参照されています。本書を印刷する場合には、これらの資料の印刷も必要となります。

以下のサイトから、関連資料をインストールすることができます。

- 一般的なアダプター情報が必要な場合、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、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/adapters/infocenter>

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

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

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

- Message 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。

- 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 (<http://www.ibm.com/software/integration/websphere/support/>) にあります。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。また、IBM Redbooks (<http://www.redbooks.ibm.com/>) にもその他の有効な情報があることがあります。

---

## 表記上の規則

本書では、以下のような規則を使用しています。

<i>courier</i> フォント	コマンド名、ファイル名、ユーザーが入力する情報、システムが画面上に出力する情報などのリテラル値を示します。
<b>太字</b>	初出語を示します。
<i>イタリック、イタリック</i>	変数名または相互参照を示します。
<i>青のアウトライン</i>	青のアウトラインは、マニュアルをオンラインで表示するときのみ見られるもので、相互参照用のハイパーリンクを示します。アウトラインの内側をクリックすると、参照先オブジェクトにジャンプします。
<i>ProductDir</i>	製品のインストール先ディレクトリーを表します。



---

## 本リリースの新機能

---

### リリース 4.1.x の新機能

2004 年 6 月更新。

アダプターは変更なしでリリースされています。

2003 年 12 月に更新されました。

今回のリリースで追加された新機能は、以下のとおりです。

- アダプターのインストール情報は、本書から移動しました。この情報の新たな入手先については、第 2 章を参照してください。
- Adapter for Siebel 2000 は Microsoft Windows NT ではサポートされなくなりました。

2003 年 7 月更新。

アダプターは、WebSphere Application Server を統合ブローカーとして使用できるようになりました。詳細については、7 ページの『互換性』を参照してください。

2003 年 3 月更新。

今後、「CrossWorlds」という名前は、システム全体を表したり、コンポーネント名やツール名を修飾する目的では使用されません。それ以外の点では、以前とほぼ同じです。例えば、「CrossWorlds System Manager」は今後「System Manager」に、「CrossWorlds InterChange Server」は「WebSphere InterChange Server」になります。

IBM WebSphere Business Integration Adapter for Siebel 2000 には、Siebel 2000 用のコネクターが組み込まれています。このアダプターは、InterChange Server (ICS) および WebSphere MQ Integrator 統合ブローカーとともに動作します。統合ブローカーは異機種アプリケーションの集合を統合し、データのルーティングなどのサービスを提供します。このアダプターには以下のものが組み込まれています。

- ビジネス・オブジェクトのサンプル
- IBM WebSphere Adapter Framework。以下のものが含まれています。
  - 開発ツール (Business Object Designer および Connector Configurator を含む)
  - API (ODK、JCDK、および CDK を含む)

本書には、このアダプターを InterChange Server (ICS) および WebSphere MQ Integrator の両方の統合ブローカーとともに使用する方法についての情報が記載されています。

---

## リリース 4.0.x での新機能

コネクタは国際化に対応しています。詳細については、4 ページの『ロケール依存型データの処理』および 43 ページの『付録 A. コネクタの標準構成プロパティ』を参照してください。

---

## 第 1 章 コネクターの概要

コネクタは、コネクタ・フレームワークとアプリケーション固有のコンポーネントという 2 つのパーツから構成されます。コネクタ・フレームワークは、すべてのコネクタに共通のコードを備え、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間の仲介役として機能します。アプリケーション固有のコンポーネントには、特定のアプリケーションに合わせて調整されたコードが含まれます。コネクタ・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間で以下のサービスを提供します。

- ビジネス・オブジェクトの受信と送信
- 始動メッセージや管理メッセージの交換の管理

ここでは、コネクタ・フレームワークとアプリケーション固有のコンポーネント (コネクタと呼びます) について説明します。以下のトピックがあります。

- 『COM サポート』
- 『サポートされているオペレーティング・システム』
- 『メタデータ主導型設計』
- 2 ページの『アーキテクチャー』
- 4 ページの『ロケール依存型データの処理』

---

### COM サポート

コネクタは、COM 通信アーキテクチャーをサポートします。COM 構成において、コネクタは、Siebel COM データ・サーバーと通信を行います。この構成により、コネクタは、Siebel ビジネス・オブジェクトと Siebel ビジネス・コンポーネントの両方に直接アクセスし、操作することができます。

---

### サポートされているオペレーティング・システム

コネクタは、Windows NT 4.0 日本語版のみに対応しています。

---

### メタデータ主導型設計

コネクタは、メタデータ主導型です。メタデータとは、IBM WebSphere Business Integration Adapter 環境では、ビジネス・オブジェクトに格納されるアプリケーション固有データのことであり、アプリケーションとコネクタの連携動作を円滑にするものです。メタデータ主導型コネクタでは、そのサポートする各ビジネス・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に基づいて処理されサポートされます。

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、属性プロパティの設定値、およびアプリケーション固有テキストの内容が組み込まれ

ています。コネクタは、メタデータ主導型であるため、そのコードを変更することなく、新規ビジネス・オブジェクトや修正後のビジネス・オブジェクトの処理が可能です。

---

## アーキテクチャー

Siebel アプリケーション・アーキテクチャーは、次の 3 つのレイヤーから構成されています。

- ユーザー・インターフェース・レイヤー
- ビジネス・オブジェクト・レイヤー
- データ・オブジェクト・レイヤー

この 3 つのレイヤーにより、アプリケーション・データとのユーザー対話操作の仕組みが実現され、データベースからアプリケーションが分離されます。アプリケーション・データは、サード・パーティーのデータベース管理システムに格納されます。

Siebel には、コネクタなどの外部プログラムを Siebel アプリケーションに統合することを可能にする Siebel オブジェクト・インターフェースが用意されています。Siebel オブジェクト・インターフェースは、外部プログラムに、そのデータと関数を公開する Siebel ビジネス・オブジェクトのメソッドから構成されています。

Siebel ビジネス・オブジェクトは、仮想テーブル、仮想テーブル間の結合、および仮想テーブル間のさまざまな関係を使用して、アプリケーション・データとビジネス・ロジックを組織化したものです。

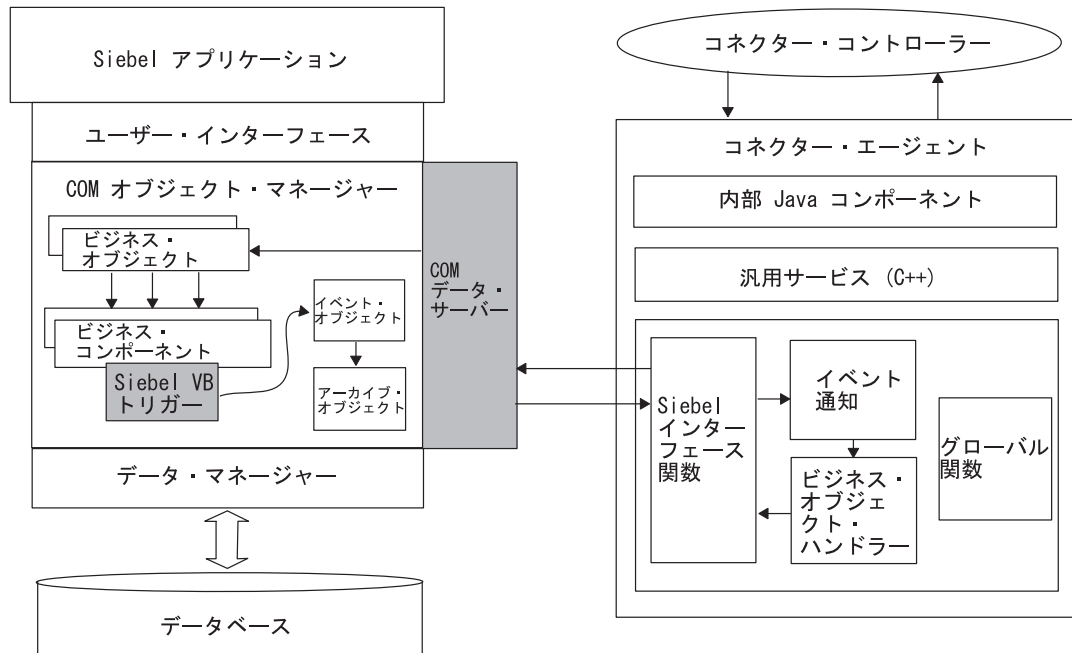
Siebel オブジェクト・インターフェースには、Siebel アプリケーション・データへのアクセスを可能にする Siebel COM データ・サーバーが組み込まれています。このインターフェースを使用すると、コネクタは、Siebel アプリケーションにおけるユーザー操作に影響することなく、ビジネス・オブジェクトおよびビジネス・コンポーネント・データにアクセスし、操作することができます。

Siebel ビジネス・オブジェクトは、Siebel アプリケーションでは、カスタマイズ可能なオブジェクト指向の子ビジネス・オブジェクトの配列となります。ビジネス・オブジェクトとは、さまざまなビジネス・コンポーネント間の関係を定義したものです。Siebel ビジネス・オブジェクトは、1 つ以上のビジネス・コンポーネントを 1 つの情報論理単位にグループ化します。ビジネス・コンポーネントとは、アカウントなどの特定エンティティの構造、振る舞い、および実際のデータを定義したものです。

Siebel ビジネス・オブジェクトは、階層構造を取ります。各ビジネス・オブジェクトには、1 つ以上の Siebel ビジネス・コンポーネントが含まれます。IBM WebSphere Business Integration Adapter システムでは、トップレベルのビジネス・オブジェクトが Siebel ビジネス・オブジェクトに対応し、子ビジネス・オブジェクトが Siebel ビジネス・コンポーネントに対応します。

## COM コネクターと Siebel との間の通信

Siebel COM コネクター・コンポーネント、および Siebel アプリケーションとの関係を図に示します。



## サポートされている動詞

コネクターでは、Create (作成)、Retrieve (検索)、Update (更新)、および Delete (削除) の各動詞がサポートされています。

## イベント通知

コネクターは、イベント・キュー・テーブルに対するデータ設定を、ビジネス・コンポーネント・イベント・ハンドラーに組み込まれている Siebel Visual Basic スクリプトに担当させています。Siebel にイベントが発生すると、Visual Basic スクリプトが、イベント・キュー・テーブルにイベントを挿入します。コネクターは、構成可能なインターバルでイベント・キュー・テーブルをポーリングし、イベントを逐次処理します。

コネクターでは、イベント通知に対して、Siebel\_CW\_Event テーブルと Siebel\_CW\_Archive テーブルが使用されます。この 2 つのテーブルは、コネクター・インストール作業の一環として作成し、Siebel アプリケーション内に常駐させます。テーブルはそれぞれ、Siebel\_BO\_CWEvent および Siebel\_BO\_CWArchive というビジネス・オブジェクトとして表されます。Siebel からイベントを検索する場合、コネクターは、ビジネス・オブジェクトの Siebel\_BO\_CWEvent 上でビジネス・オブジェクト・ハンドラー・メソッド doRetrieve() を呼び出します。このビジネス・オブジェクトは、Siebel CW Events オブジェクトに照会し、Status 値が 0 のイベントをすべて検索します。コネクターによるイベントの検索順序は、

Siebel\_BC\_CWEvent ビジネス・オブジェクト内の Retrieve 動詞で優先順位とソート・プロパティによって指定されます。コネクタは、次に、イベント表から検索した各イベントに対して、そのイベントを起動した実際のビジネス・オブジェクトを検索します。

コネクタは、イベントの処理を終了すると、ビジネス・オブジェクト・ハンドラーの doCreate() メソッドを呼び出して、アーカイブ表にイベントを挿入した後、doDelete() メソッドを呼び出して、イベント表からイベントを削除します。イベントの処理が成功すると、Status フィールドは Success に更新されます。イベントがエラーのため処理されない場合、Status フィールドは、Failure に更新されます。イベントが未登録の場合は、Status フィールドは Skipped に更新されます。

キュー・テーブルの作成については、9 ページの『イベント表とアーカイブ表の作成』を参照してください。

---

## ロケール依存型データの処理

コネクタは、2 バイト文字セットをサポートし、指定の言語でメッセージ・テキストを配信できるように国際化されています。コネクタが、1 文字コード・セットを使用する地域から異なるコード・セットを使用する地域にデータを転送する場合は、文字変換を実行し、データの意味を維持します。

Java 仮想マシン (JVM: Java Virtual Machine) 内の Java ランタイム環境では、データが Unicode 文字コード・セットで表現されます。Unicode には、既知の文字コード・セット (単一バイトとマルチバイトの両方) における文字のエンコードが、ほぼ一通り組み込まれています。WebSphere Business Integration システムのほとんどのコンポーネントは、Java で作成されています。したがって、ほとんどの WebSphere Business Integration システムのコンポーネント間におけるデータ転送の際には、文字変換の必要はありません。

このコネクタは、C++ で書かれているため、コネクタ・フレームワークにより、アプリケーション・データは、固有のエンコードから、Unicode コードに変換されます。コネクタのアプリケーション固有コンポーネントが、固有エンコードのデータをコネクタ・フレームワークに引き渡した後、コネクタ・フレームワークがデータを Unicode に変換し、それから ICS に送信します。

コネクタ・フレームワークは、ICS から要求ビジネス・オブジェクトを受け取ると、コネクタの Locale と CharacterEncoding の両構成プロパティから該当するロケール情報を判定します。このロケール情報を使用して、データを固有のエンコードに変換してから、コネクタの C++ アプリケーション固有コンポーネントに引き渡します。

図 1 に、文字変換の実行を示します。

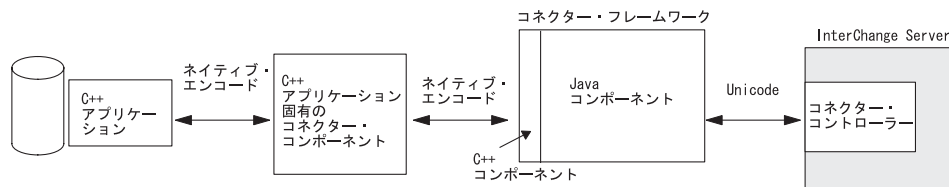


図 1. コネクターが文字のエンコードを処理する方法

ロケール・プロパティの詳細については、43 ページの『付録 A. コネクターの標準構成プロパティ』を参照してください。





---

## 第 2 章 コネクタのインストールと構成

この章では、コネクタをインストール、構成する方法、および Siebel アプリケーションとコネクタとの協調動作を構成する方法について説明します。以下のトピックがあります。

- 『互換性』
- 8 ページの『前提条件』
- 8 ページの『アダプターと関連ファイルのインストール』
- 9 ページの『必要な変更』
- 13 ページの『コネクタの構成』
- 19 ページの『コネクタ構成プロパティ』
- 22 ページの『複数のコネクタ・インスタンスの作成』
- 23 ページの『コネクタの始動』
- 25 ページの『Windows サービスとしてのコネクタの実行』
- 25 ページの『コネクタの停止』

---

### 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。アダプター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 2.1、2.2、2.3.x、および 2.4
- **統合ブローカー:**
  - WebSphere InterChange Server、バージョン 4.2.x
  - 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)

例外については、「リリース情報」を参照してください。

**注:** 統合ブローカーのインストール手順およびその前提条件については、次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。一部の資料は次の Web サイトにあります。

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

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

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

---

## 前提条件

このセクションでは、コネクターをインストールし、実行する前に、インストールしておく必要のあるソフトウェア・コンポーネントおよび実施しておく必要のある作業について説明します。

### 前提となるソフトウェア

Siebel クライアント・ソフトウェアは、コネクターが動作しているのと同じのマシンにインストールする必要があります。また、特定のセットアップ作業を完了する必要があります。

### ユーザーのセットアップ

コネクターのインストール前に、Siebel において、コネクターに対するユーザー・アカウントを作成しておく必要があります。このユーザー・アカウントには、完全なアクセス権を割り当て、ログイン名は、ApplicationUserName 構成プロパティと同一にする必要があります。ユーザー・アカウントのログイン名とパスワードのデフォルト値は、CWCONN です。

**注:** コネクターの使用前に、アプリケーションにユーザー・アカウントがあるかどうかを確認する必要があります。このユーザー・アカウントは、Siebel Tools でイベント作成用の Siebel スクリプトに指定したユーザーと同じでなければなりません。

**注:** 関連するコネクター・メッセージ・ファイルを、  
`ProductDir¥connectors¥messages¥Siebel¥COM` ディレクトリーから  
`ProductDir¥connectors¥messages` ディレクトリーにコピーする必要があります。

---

## アダプターと関連ファイルのインストール

WebSphere Business Integration Adapter 製品のインストールについては、「*WebSphere Business Integration Adapters インストール・ガイド*」を参照してください。この資料は、次の Web サイトの WebSphere Business Integration Adapters Infocenter にあります。

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

## 必要な変更

コネクタのインストールが完了したら、Siebel ファイルを新しいロケーションにコピーして、イベント表とアーカイブ表を作成し、Siebel 構成ファイル (siebeljpn.cfg) の設定値を変更する必要があります。このセクションでは、以下のトピックについて説明します。

- 『Siebel ファイルのコピー』
- 『イベント表とアーカイブ表の作成』
- 12 ページの『留意点』
- 12 ページの『構成ファイルの設定値の変更』

## Siebel ファイルのコピー

Siebel コネクタのインストール後、表 1 の情報を使用して Siebel ファイルを正しい場所にコピーしてください。

表 1. Siebel ファイルのコピー先

ファイルのインストール先ディレクトリ	ファイルのコピー先ディレクトリ
<i>ProductDir</i> ¥connectores¥Siebel¥COM¥Siebel.dll	<i>ProductDir</i> ¥connectores¥Siebel¥Siebel.dll
<i>ProductDir</i> ¥connectores¥Siebel¥COM¥repository¥CN_SiebelCOM.txt	<i>ProductDir</i> ¥connectores¥Siebel¥repository¥CN_SiebelCOM.txt
<i>ProductDir</i> ¥connectores¥Siebel¥COM¥messages¥SiebelCOMConnector.txt	<i>ProductDir</i> ¥connectores¥Siebel¥messages¥SiebelCOMConnector.txt
<i>ProductDir</i> ¥connectores¥Siebel¥COM¥repository¥Siebel_BO_CWArchive.txt	<i>ProductDir</i> ¥connectores¥Siebel¥repository¥Siebel_BO_CWArchive.txt
<i>ProductDir</i> ¥connectores¥Siebel¥COM¥repository¥Siebel_BO_CWEvent.txt	<i>ProductDir</i> ¥connectores¥Siebel¥repository¥Siebel_BO_CWEvent.txt

## イベント表とアーカイブ表の作成

この手順では、例として、Siebel Sales Enterprise アプリケーションを使用しています。Siebel Sales Enterprise の記載箇所は、実際に使用する Siebel アプリケーションの名前で置き換えます。

イベント表とアーカイブ表を作成し、ビジネス・オブジェクトを起動するには、次の手順を実行します。

1. 現在のプロジェクトがすべてチェックインされていることを確認する。
2. ローカル・データベースで、次のファイルをチェックし、ロックする。
  - New Table プロジェクト
  - Siebel Sales Enterprise プロジェクト
  - Account プロジェクトなど、変更対象のオブジェクトを含むプロジェクト
  - Dock プロジェクト

**注:** 上記のプロジェクトが、ローカル・サーバーと開発サーバーの両方でロックされていることを確認します。プロジェクトのロック方法については、「Siebel Tools Guide」の第 5 章を参照してください。

3. ローカル・データベースに対して、7 つの IBM WebSphere Business Integration Adapter パッチ・ファイルを次の順序で適用する。
  - cwttable.spf
  - cwview.spf
  - cwapplet.spf
  - cwbo.spf
  - cwbc.spf
  - cwdo.spf
  - cwscreen.spf
4. cweventmanager.spf パッチにより、イベントの再キューイングまたは削除を CW アーカイブ表に対して行う機能が追加されます。この機能が必要な場合は、パッチの適用前に、14 ページの『CW Event Manager』を参照して、詳細を理解してください。VB スクリプトを使用する場合は、cweventmanagervbscript.spf パッチを適用します。e-script を使用する場合は、cweventmanagerescript.spf パッチを適用します。
5. プロンプトで要求されたら、ローカル・データベースの CW Audit プロジェクトをロックする。プロジェクトのロック方法の詳細については、「Siebel Tools Guide」の第 5 章を参照してください。
6. 次項が作成されていることを確認する。
  - 2 つの新規表 CX\_CW\_Archive\_Q と CX\_CW\_Event\_Q
  - 1 つの新規ビジネス・オブジェクト CW Events
  - 2 つの新規ビジネス・コンポーネント CW Archive と CW Events
  - 1 つの新規ビュー CW Event List View
  - 2 つの新規アプレット CW Archive List Applet と CW Event List Applet
  - 1 つの新規画面 CW Events と 1 つの新規画面ビュー「CW Event List」ビュー
  - 2 つの新規ドック・オブジェクト CX\_CWArchive と CX\_CWEvent
7. 次のようにして、ページ・タブを作成する。
  - a. 「Application」 > 「Siebel Sales Enterprise」 > 「Page」 タブにアクセスする。
  - b. 右クリックし、メニューから「New Record」を選択する。
  - c. 画面およびテキスト名として、CW Events と入力する。
  - d. シーケンスとして、他のシーケンス番号よりも大きい番号を入力する。この選択により、アプリケーション内でのタブの表示位置が決定されます。
  - e. 非アクティブなフィールドには、チェックマークを付けないままにする。

ページ・タブの詳細については、「Siebel Tools Guide」の第 5 章を参照してください。
8. 次のようにして、画面メニュー項目を作成する。
  - a. 「Application」 > 「Siebel Sales Enterprise」 > 「Screen Menu Item」にアクセスする。
  - b. 右クリックし、「New Record」を選択する。
  - c. 画面およびテキスト名として、CW Events と入力する。

- d. シーケンスとして、他のシーケンス番号よりも大きい番号を入力する。この選択により、画面のプルダウン・メニュー内でのタブの表示位置が決定されます。
  - e. 非アクティブなフィールドには、チェックマークを付けないままにする。
9. サイトで使用する IBM WebSphere Business Integration Adapter ビジネス・オブジェクトに対応するビジネス・コンポーネントに対する Siebel VB スクリプトの追加または変更を行う。Siebel VB スクリプトは、ビジネス・オブジェクトに対して、イベント通知を起動します。
- イベントを優先順位でソートする場合は、コンパイル前にビジネス・オブジェクトの VB スクリプトで優先順位の値を編集する。詳細については、16 ページの『イベントへの優先順位の割り当て』を参照してください。
  - 複数のコネクタをインストールする場合は、VB スクリプト内で Connector Id を設定し、アクティブにする。詳細については、17 ページの『複数のコネクタのインストール』を参照してください。
  - Additional Object Key フィールドを使用する場合は、VB スクリプトで設定する必要がある。詳細については、16 ページの『イベント・ビジネス・オブジェクトにおける複数属性の設定』を参照してください。
10. ローカル・データベースに新規表の物理スキーマを適用する。これは、2 つの新規表 CX\_CW\_ARCHIVE\_Q と CX\_CW\_EVENT\_Q を照会してから、物理スキーマを作成する現在の照会を指定して行うことができます。表スペースとインデックス・スペースは、必ずブランクのままにします。ローカル・データベースに変更を適用する方法については、「Siebel Tools Guide」の第 11 章を参照してください。
11. アクティブ・ボタンを使用して、新規スキーマをアクティブにする。説明については、「Siebel Tools Guide」の第 11 章を参照してください。
12. ローカル・データベースで更新後のロックされているプロジェクトをコンパイルし、新規 Siebel リポジトリ (.srf) ファイルを作成する。プロジェクトのコンパイル方法については、「Siebel Tools Guide」の第 5 章を参照してください。
13. ローカル・データベースで Siebel Sales Enterprise を開く。次の操作を実行するには、管理特権を持っている必要があります。
- a. CW Event List View という名前の新規ビューを作成する。ヒント: ツールからビュー名をコピーし、「View Name」フィールドに貼り付けます。
  - b. CW Event List View に対して CW Responsibility という名前の新規の責任を作成する。
  - c. 新規に作成した CW Responsibility に、イベントの検討を担当する従業員またはチームを追加する。
  - d. CWCONN ユーザーを作成した後、CW Responsibility と Administrative Responsibility に追加する。
- 担当責任の詳細については、「Siebel Applications Administration Guide」の第 2 章を参照してください。
14. 実際のローカル環境で、アプリケーションをテストする。CW Event List View が表示できること、および IBM WebSphere Business Integration Adapter 対応オブジェクトの作成後、そのビューにイベントが生成されることを確認します。

例えば、Siebel に新規アカウントを作成してから、新規アカウント・イベントが、CW Event List View に表示されることをチェックします。

15. 開発サーバーに次の更新後のロックされているプロジェクトをチェックインする。
  - 新規表
  - CW 監査
  - ドック
  - Siebel Sales Enterprise
  - 使用するビジネス・オブジェクトに対するプロジェクト

**注:** ロックされているプロジェクトは、照会以外の方法ではチェックインしないようにします。詳細については、「Siebel Tools Guide」の第 5 章を参照してください。

16. 開発データベースに物理スキーマを適用する。これは、2 つの新規表 CX\_CW\_ARCHIVE\_Q と CX\_CW\_EVENT\_Q を照会してから、物理スキーマを作成する現在の照会を指定して行うことができます。表スペースとインデックス・スペースは、必ずブランクのままにします。
17. 開発データベース内の照会対象の表をアクティブにする。詳細については、「Siebel Tools Guide」の第 8 章を参照してください。
18. 場合に応じて適切にテスト環境と実稼働環境に移動する。データベース間で Siebel リポジトリをマイグレーションする方法については、「Siebel Tools Guide」の第 5 章を参照してください。

## 留意点

- リモート・ユーザーにディスクまたは E メールを使用して、新規リポジトリ・ファイルを配布する。このファイルを使用して、Siebel オブジェクト・ディレクトリーにある既存の .srf ファイルを置き換えます。新規リポジトリ・ファイルは、プロジェクトをコンパイルした時点でローカル・データベース上に作成されています。
- 実稼働環境の構成時に、構成ファイルのデータ・ソースをローカルからサーバーに変更する。これにより、コネクタのデータ取得相手が、各リモート・ユーザーではなく、サーバーに指定されます。コネクタ・プロパティのもとで、使用中の Siebel 構成ファイルを確認し、そのファイルで DataSource プロパティを Local から Server に変更します。
- リモート・ユーザーは、イベント表とアーカイブ表を使用して自身のシステムを更新するため、同期を取る必要がある。

## 構成ファイルの設定値の変更

ここで説明する構成ファイルの設定値は、1 つ以上変更することが必要になる場合があります。

### DataSource プロパティの設定

コネクタと同一マシン上の siebeljpn.cfg 構成ファイルで、DataSource 構成プロパティの値に Server を設定します。

## イベントの即時表示

発生とほぼ同時に、Siebel における新規到着イベントを表示するには、`siebeljpn.cfg` ファイルの `MaxCachedCursors` と `MaxCachedDataSets` の両プロパティに `0` を設定します。

## Microsoft アプリケーションとの競合

Microsoft アプリケーションと同一マシン上でコネクターを実行すると、Microsoft アプリケーションが停止する場合があります。また、Siebel のビジネス・コンポーネントに対して、既存の Visual Basic コードが存在する場合に、コネクターがイベントの処理に失敗することがあります。このような状態を防止するには、次のいずれかを実行します。

- コネクターが動作しているマシンと Siebel クライアント・アプリケーションが動作しているマシンが異なる場合には、コネクターが動作しているマシンの `siebeljpn.cfg` ファイルを開き、`EnableBasic` プロパティを `false` に変更します。
- コネクターが動作しているマシンと Siebel クライアント・アプリケーションが動作しているマシンが同一の場合には、`siebeljpn.cfg` ファイルをコピーした後、`cwsiebeljpn.cfg` に名前を変更し同一ディレクトリーに 2 番目の構成ファイルを作成してから、そのファイルの `EnableBasic` プロパティを `false` に変更します。両方の構成ファイルが、同一の `siebel.srf` リポジトリー・ファイルを必ず指すようにします。System Manager で `SiebelConfigFile` プロパティを変更して、名前変更した構成ファイルを確実に指すようにします。この操作によって、同一マシン上で、Siebel クライアント、コネクター、および Microsoft アプリケーションが共存し、正常に機能することが可能になります。

---

## コネクターの構成

以降のセクションでは、コネクターを構成する方法について説明します。

- 『共通の構成タスク』
- 15 ページの『イベント送達機構の構成』
- 17 ページの『複数のコネクターのインストール』

### 共通の構成タスク

このセクションでは、インプリメンテーションの成功を確実にするために、考慮すべき一般的な要点をいくつか取り上げて説明します。

#### -t オプションの設定

コネクターは単一スレッドであるため、その実行前に、コネクター・ショートカットの `-t` オプションを設定する必要があります。

#### Row\_Id 属性の使用

Siebel データベースにデータをコネクターが挿入するには、`Row_Id` 属性が、`CW_Siebel` ビジネス・オブジェクト内に存在する必要があります。また、更新または削除を実行するには、次のいずれかを実行する必要があります。

- `Row_Id` を相互参照する (推奨)。
- 次を実行するように、コラボレーションを変更する。

- 1 つ以上の属性に固有値を持つ「ダミー」送信オブジェクトを作成し、そのオブジェクトを送信呼び出しに引き渡す。
- 検索呼び出しから Row\_Id 値を取り出し、元の送信オブジェクトに挿入する。そのオブジェクトを送信呼び出しに引き渡す。

**注:** このオプションが機能するには、コラボレーション内にその属性名をハードコーディングする必要があります。これにより、オーバーヘッドが加わるとともに、システムの変更時に考慮すべき要素が増えます。

## 複数値リンク、複数値フィールド、複数値グループ、およびピック・リスト

コネクタを使用して、Siebel にアクセスする前に、複数値リンク、複数値フィールド、複数値グループ、およびピック・リストがすべて CW\_Siebel ビジネス・オブジェクトの定義に従って、Siebel 内に存在することを確認します。

複数値リンクは、1 対多、多対多の関係をもち、同一の画面またはアプレット内に常駐する 2 つの Siebel ビジネス・コンポーネント間に存在します。ただし、コネクタでは、同一の画面またはアプレットに常駐するかに関係なく、1 対多または多対多の関係を持つ 2 つの任意のビジネス・コンポーネント間に、複数値リンクが存在することを前提にします。

## 複数イベントの処理

コネクタに、5000 以上の複数イベントの処理が課せられている場合は、RefreshLogonCycle コネクタ・プロパティを使用します。このプロパティにより、コネクタは、指定された数のイベントを処理した後、クライアントとの接続を切断した後、再接続することが可能になります。詳細については、19 ページの『コネクタ構成プロパティ』を参照してください。

## Siebel アプリケーションへの変更

コネクタは、objects ディレクトリーにある Siebel.srf ファイルに依存します。したがって、Siebel アプリケーションに変更が行われた場合、Siebel.srf ファイルを、更新後のバージョンで置き換える必要があります。

## CW Event Manager

CW Event Manager とは、CWEvents 画面内の 1 つのビューです。このビューは、CW アーカイブ表から CW イベント表へ、コネクタによる再処理目的のもとイベントを再キューイングするために使用されます。CW Event Manager は、CW アーカイブ表から処理済みイベントを削除するために使用される場合もあります。再キューイングされたイベントには、CW アーカイブ表と CW イベント表内において、その記述に「Requeued Event」が追加されます。この追加により、CW アーカイブ表から重複したイベントを見つけ、削除することが簡単になります。

このビューには、2 つのオプションが用意されています。該当するフィールドにイベント基準を入力した後、選択したイベントの削除または再キューイングを行うことができます。開始時刻フィールドと終了時刻フィールドを使用してイベントを選択する場合は、フォーマット mm/dd/yy hh:mm:ss AM/PM を使用します。例えば、3/28/01 9:45:32 PM のように指定します。CW Event Manager が、これを 24 時間表記に変換します。



CW Event Manager は、スクリプト依存です。VB または e-script を使用しているかにより、該当する CW Event Manager パッチを適用する必要があります。パッチの適用後、次の操作を実行します。

1. Siebel Tools において、CW Event Manager ビジネス・コンポーネントを CW Events ビジネス・オブジェクトに追加する。
2. CW Event Manager 画面ビューを CW Events 画面に追加する。「Menu Text」フィールドと「View Bar Text」フィールドに CW Event Manager と入力する。
3. Siebel アプリケーションにおいて、CW Event Manager という名前の新規ビューを作成する。
4. CW Responsibilities と Admin Responsibilities に作成したビューを追加する。

## パフォーマンス

Siebel dll は、アプリケーション・イメージの検索、および送信されたビジネス・オブジェクトと検索結果との比較におけるパフォーマンス・オーバーヘッドを防止します。子の動詞は、SkipRetrieve に true がセットされている場合には、手動で設定する必要があります。そうしない場合、子は親の動詞を継承します。検索が行われないため、結果は必ずしも求めているものでない場合があります。

SkipRetrieve は ASI であり、true または false の値を取ることができます。この新規 ASI は、その親ビジネス・オブジェクトのすぐ内側にある BC Parent レベルでのみ使用することができます。この ASI を使用する場合は、ビジネス・オブジェクトの名前を nm= の後に設定し、セミコロンで SkipRetrieve ASI から区切る必要があります。

例えば、Siebel\_BCAccount では、ASI は nm=Account;SkipRetrieve=true となります。

## イベント送達機構の構成

イベント送達機構では、次の振る舞いを構成することができます。

- ポーリング単位の処理イベントの数
- イベントに割り当てる優先順位
- ビジネス・オブジェクト・トリガーによって設定される属性の数

### 1 回のポーリングによる複数イベントの処理

コネクタは、1 回のポーリングで指定された数の Siebel\_BC\_CWEvent ビジネス・オブジェクトを検索し、処理するように構成することができます。アプリケーションが多数のイベントを生成する場合、1 回のポーリングで複数のイベントを処理すると、パフォーマンスが改善されます。

ポーリング単位のイベント数は、Siebel\_BC\_CWEvent ビジネス・オブジェクトの Retrieve 動詞でアプリケーション固有情報を使用して指定します。このビジネス・オブジェクトのアプリケーション固有情報は、他のアプリケーション固有ビジネス・オブジェクトと同様、System Manager を使用して変更することができます。

ポーリング単位の処理イベントの数を指定するには、属性値ペア `max=n` を使用します。例えば、1 回のポーリングで 25 個のイベント・ビジネス・オブジェクトをコネクタが検索するように指定するには、次のアプリケーション固有情報を `Retrieve` 動詞で使用します。

```
[Verb]
Name = Retrieve
AppSpecificInfo =max=25
[End]
```

**注:** ポーリング単位の最大イベント数を設定しない場合、コネクタでは、すべてのイベントが検索されます。これが、システム・パフォーマンス問題の原因になる場合があります。この値は、システム・パフォーマンスの低下を伴わないで、コネクタが検索可能なイベントの最大数に基づいて設定します。デフォルト値は、1 回のポーリングあたり 25 イベントです。

## イベントへの優先順位の割り当て

コネクタによる処理イベントの選択順序を指定するため、イベントに対して、優先順位の値を割り当てることができます。イベント優先順位を使用すると、イベント・ストア内のイベント数が、1 回のポーリングでコネクタの検索する最大イベント数を超える事態に、コネクタ・ポーリング方法によって対応することが可能になります。

イベント優先順位は、各ビジネス・コンポーネントの Siebel VB スクリプトで設定されます。デフォルトでは、すべてのビジネス・オブジェクトの優先順位には 1 が設定され、イベントは作成日でソートされます。

優先順位ソートを使用可能にするには、まず、VB スクリプトを編集して、優先順位を 0 (最高) と 5 (最低) の間で変更します。その後、名前値ペア `sort=Priority, EventTs(ASCENDING)` を、Siebel\_BC\_CWEvent オブジェクトの `Retrieve` 動詞のアプリケーション固有情報に追加します。この名前値ペアにより、イベントが、まず、優先順位によって、次に、作成日で昇順にソートされることが指定されます。

ポーリング単位の検索イベントの最大数は、常時指定します。優先順位ソートも指定する場合は、最大イベント属性値ペアを優先順位属性値ペアの前に置きます。例えば、優先順位ソートを指定するには、Siebel\_BC\_CWEvent オブジェクトの `Retrieve` 動詞で次のアプリケーション固有情報を使用します。

```
[Verb]
Name = Retrieve
AppSpecificInfo =max=25;sort=Priority, EventTs(ASCENDING)
[End]
```

`sort=Priority` の指定がない場合は、ソートはデフォルトの作成日ソートになり、優先順位はチェックされません。

## イベント・ビジネス・オブジェクトにおける複数属性の設定

加入配信ビジネス・オブジェクトでは、Siebel VB スクリプトの Object Key フィールドに指定することによって属性を 1 つ以上を設定することができます。属性の全長が、Object Key フィールドの長さ (80 文字) を超える場合は、最大 250 文字まで対応する Additional Object Key フィールドを使用します。Object Key は、

必須値です。したがって、Additional Object Key フィールドを使用する場合は、Object Key フィールドにダミー値を設定する必要があります。

Additional Object Key フィールドに設定するには、次のようにします。

1. Siebel\_BC\_CWEvent と Siebel\_BC\_CWArchive の両オブジェクトで、Object Key のアプリケーション固有情報に Additional Object Key を設定する。
2. Siebel\_BC\_CWEvent において、AppSpecificInfo プロパティに Object Key を設定して、ダミーのキー属性を作成する。
3. ダミー・キーと同一内容の非キー属性を Siebel\_BC\_CWArchive に追加する。Object Key とダミー・キー属性の順序を、必ず両方のオブジェクトで同一にします。
4. すべてのオブジェクトの Siebel VB スクリプトに Additional Object Key を設定する。
5. すべてのオブジェクトの Siebel VB スクリプトに、ダミー値を使用して、Object Key をハードコーディングする。この作業は、Siebel では、Object Key が必須値であるため必要となります。

スクリプトでは、名前値ペアを使用するか、スラッシュ (/) が先頭に来る完全修飾パスを使用するかして、属性および値を指定することができます。複合キーの構文は、次のとおりです。

```
/attribute[/attribute]=value[;attribute[/attribute]=value]
```

属性値のみを指定することも可能です。その場合は、最初の子ビジネス・オブジェクト属性の第 1 キーが設定されます。

例えば、Siebel\_BCContact ビジネス・オブジェクトの ContactId などのように、単一キーのみを設定する VB スクリプトには、次のような行を記述します。

```
RowId=GetFieldValue("Id")
Set EventBC = EventBO.GetBusComp("CW Events")
EventBC.SetFieldValue "Object Key" , RowId
```

Siebel\_BCContact ビジネス・オブジェクトにおいて、ContactId と AccountId の両属性に対してキーを設定するスクリプトには、次のような行を記述します。

```
RowId = GetFieldValue("Id")
Account = GetFieldValue ("Account Id")
Key = "/Siebel_BCContact/ContactId=" + RowId +
";/Siebel_BCContact/AccountId=" + Account
EventBC = EventBO.GetBusComp("CW Events")
EventBC.SetFieldValue "Object Key" , Key
```

注: このコード例は、標準的な Siebel\_BOContact ビジネス・オブジェクトの構造に基づいています。

## 複数のコネクターのインストール

特定タイプの Siebel イベントの処理を合理化するため、Siebel クライアントに複数のコネクターをインストール、構成し、特定のビジネス・オブジェクトを処理するように各コネクターを構成することができます。複数のコネクターが同一の Siebel アプリケーションにアクセス可能となるため、各コネクターは、指定のタイプのイ

イベントを処理し、統合ブローカーに引き渡すことができます。これにより、Siebel アプリケーションとの送受信データの転送スピードとスループットが改善できます。

複数のコネクタをインストールおよび設定するには、次の手順に従います。

1. 実行する追加の各コネクタに対して、`%ProductDir%\connectors` にある既存の Siebel フォルダをコピーして貼り付け、新規フォルダを作成する。必要に応じて、適切にフォルダの名前を変更します。例えば、Siebel1、Siebel2 などのようにします。
2. 同一規則を使用して、各新規フォルダ内のコネクタ DLL (`siebel2000.dll`) の名前を変更する。例えば、Siebel2 フォルダでは、コネクタ DLL に `siebel2.dll` という名前を付けます。
3. それぞれの新規コネクタごとに、`%ProductDir%\repository\Siebel2000` ディレクトリにあるコネクタ定義ファイル `CN_Siebel2000.txt` のコピーを作成し、新規コネクタ名を使用して名前を付ける。例えば、`CN_Siebel1.txt` のようにします。
4. Business Object Designer を使用して、ビジネス・オブジェクト定義をリポジトリにコピーする。

**注:** InterChange Server (ICS) が統合ブローカーである場合は、この方法の代わりに `repos_copy` コマンドを使用して定義をリポジトリにコピーできます。WebSphere MQ Integrator Broker が統合ブローカーである場合は、システム・コマンドを使用してファイルをリポジトリにコピーできます。

5. コピーと貼り付けを使用して、コネクタの新規コピーを System Manager で作成する。フォルダと `.dll` ファイルに使用した同一の命名規則を適用します。例えば、`Siebel1Connector` のようにします。
6. System Manager で、各新規コネクタの `ApplicationName` プロパティにそれぞれ新しい名前を付ける。例えば、`Siebel2Connector` のようにします。
7. 各コネクタの `ConnectorId` プロパティに、そのコネクタの番号を設定する。例えば、`Siebel1Connector` の `ConnectorId` プロパティには、1 を設定します。`ConnectorId` の値は、整数値にする必要があります。詳細については、19 ページの『コネクタ構成プロパティ』を参照してください。
8. すべてのコネクタが、イベントに対して、同一の Siebel アプリケーションをポーリングするように、`SiebelConfigFile` プロパティを設定する。
9. System Manager で、各コネクタに対して、サポートされるビジネス・オブジェクトを指定する。
10. 各コネクタにコラボレーションを割り当てる。コラボレーションが設定済みである場合は、その再作成が必要になることがあります。
11. 各新規コネクタ用にコネクタ・ショートカットをコピーした後、ショートカットの名前を対応するコネクタの名前に変更する。

各コネクタは、`ConnectorId` プロパティ値に一致するイベントを取り込みます。`ConnectorId` プロパティに対して値が未割り当ての場合、コネクタは、すべてのイベントを取り込みます。

**注:** 複数 InterChange Server の実行ライセンスを持っているサイトの場合は、異なる InterChange Server の動作するコネクタに対して、別々の ConnectorId 構成プロパティを指定することができます。

12. 新規コネクタによってサポートされるビジネス・オブジェクトに対して、Siebel Visual Basic スクリプトを変更する。各スクリプトの ConnectorId フィールドを編集して、該当するコネクタ ID を指すようにする。例えば、Siebel2 コネクタが Account ビジネス・オブジェクトをポーリングしている場合は、次のコードを Visual Basic スクリプトに追加して、Account の ConnectorId フィールドに 2 を設定します。

```
.ActivateField "Connector Id"  
.SetFieldValue "Connector Id", "2"
```

---

## コネクタ構成プロパティ

コネクタは、実行する前に、コネクタ標準構成プロパティとコネクタ特定構成プロパティの両方を構成しておく必要があります。コネクタ・プロパティを構成するには、Connector Configurator を使用します。Connector Configurator には、System Manager からアクセスします。構成値は入力するたびに、リポジトリに保管されます。

### 標準コネクタ・プロパティ

標準構成プロパティにより、すべてのコネクタが使用する情報が提供されます。これらのプロパティの資料については、43 ページの『付録 A. コネクタの標準構成プロパティ』を参照してください。

表 2 には、付録の構成プロパティの中で、このコネクタに特定の情報を示します。

表 2. このコネクタに特定のプロパティ情報

プロパティ	注
AgentConnections	このコネクタは単一スレッドであるため、このプロパティのデフォルト値は、変更しないでください。
CharacterEncoding	コネクタは、このプロパティを使用して固有エンコードを指定します。
Locale	このコネクタは国際化されているため、このプロパティの値は変更可能です。現在サポートされているロケールを確認するには、コネクタのリリース情報を参照してください。

**重要:** WebSphere MQ Integrator Broker は、複数のロケールをサポートしません。ご使用のシステムのすべてのコンポーネント (例えば、すべてのアダプター、アプリケーション、および統合ブローカー自体) が同じロケールに設定されていることを確認してください。

**注:** このコネクタはすべての統合ブローカーをサポートするため、すべてのブローカーの構成プロパティはコネクタに関係しています。

## 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。コネクタ固有のプロパティを使用すると、エージェントの再コード化と再構築をしなくても、コネクタ内で静的な情報または論理を変更することもできます。

表 3 には、コネクタに対するコネクタ固有の構成プロパティのリストを示します。プロパティの説明については、以下の各セクションを参照してください。

表 3. Siebel 固有のコネクタのプロパティ早見表

名前	指定可能な値	デフォルト
『ApplicationPassword』		CWCONN.
『ApplicationUserName』		CWCONN.
『ConnectorId』		
『SiebelConfigFile』		siebeljpn.cfg.
『SiebelDirectory』		¥sea¥¥client¥¥bin
21 ページの 『Transactional』	true または false	true
21 ページの 『UseDefaults』	true または false	false.
21 ページの 『ViewMode』	1 (営業担当員) 2 (マネージャー) 3 (全員) 4 (個人)	3 (全員)
21 ページの 『ConnectErrors』		「An Oracle database error has occurred. Possibly the database name is invalid.」 これは、接続が Siebel 99.5 で終了する場合に表示されるメッセージです。
22 ページの 『RefreshLogonCycle』		
22 ページの 『RestartConnectorCycle』		

### ApplicationPassword

Siebel アプリケーションにおけるユーザー・アカウント用のパスワードです。

### ApplicationUserName

Siebel におけるユーザー・アカウントの名前です。

### ConnectorId

複数コネクタが構成されている場合、特定の Siebel イベントを処理する上で使用されるコネクタを表します。値は、整数値にする必要があります。

### SiebelConfigFile

Siebel 構成ファイルの名前です。

### SiebelDirectory

Siebel COM サポートの場合、Siebel 構成ファイルのディレクトリーとファイル名は、既知のものである必要があります。通常は、クライアント・インストール・ディレクトリーにします。

## Transactional

Transactional に true が設定されている場合は、nomvl=1 のビジネス・オブジェクトは、Request、Update、Create、または Delete の各操作で処理されません。Transactional に false が設定されている場合は、すべてのイベントが処理されますが、CWSiebelConn において nomvl=1 のオブジェクトに対するトランザクション・サポートはありません。

Transactional プロパティに false が設定されている場合は、Siebel において、Cascade Delete プロパティの設定が必要になることがあります。Cascade Delete は、Siebel では、オプションの値です。Cascade Delete プロパティは、マスター・ビジネス・コンポーネントが削除された場合に、リンクのビジネス・コンポーネント詳細レコードに対して実行されるアクションを示します。詳細については、「*Siebel Tools Guide*」を参照してください。

## UseDefaults

Create 操作において、UseDefaults に true が設定または省略されている場合、コネクタは、各必須のビジネス・オブジェクト属性に対して、デフォルト値または有効値が指定されているかどうかをチェックします。デフォルト値または有効値が指定されている場合は、Create 操作は成功します。そうでない場合は失敗します。

このプロパティに false が設定されている場合は、コネクタは、有効値のみをチェックします。有効値が指定されていない場合は、Create 操作は失敗します。

## ViewMode

システムにおける、可視性レベルを設定します。例えば、0=SalesRep, 1=Manager のように設定します。完全な値セットについては、Siebel 文書の「オブジェクト・インターフェース解説書」を参照してください。

## ConnectErrors

この値がコネクタ・プロパティに設定されている場合は、コネクタは、Siebel ログに、指定されたエラー・メッセージが表示されると必ず終了します。このプロパティは、エラー・メッセージに基づいて、コネクタを停止する場合に使用されます。このフィールドには、複数のメッセージを指定することができます。区切り文字として、コロン「:」を使用して、各メッセージを分離します。

テキストには、ストリング「Text:」の後にエラー・メッセージのセクションを含める必要があります。

**重要:** マッチング・ストリングが、Siebel コネクタ・ログに検出されると、コネクタが終了することになるため、値の設定には注意してください。

デフォルト・メッセージ値は、「An Oracle database error has occurred. Possibly the database name is invalid.」です。これは、接続が Siebel 99.5 で終了する場合に表示されるメッセージです。

**注:** コネクタは、MaxEvents 構成プロパティのサポートをすでに停止しています。ポーリング対象イベントの最大数の設定は、コネクタ・レベルではなく、ビジネス・オブジェクト・レベルでサポートされています。ポーリング対

象のイベント数を設定する方法については、15 ページの『1 回のポーリングによる複数イベントの処理』を参照してください。

## RefreshLogonCycle

このプロパティは、その数以上になるとコネクタ・エージェントが切断、リスタートされるコネクタの処理する要求数を示します。

## RestartConnectorCycle

この値が設定されていると、コネクタは、指定されている数のオブジェクトの処理が終了した後、最初のポーリング呼び出し内で完全に終了します。このプロパティと RefreshLogonCycle との違いは、RefreshLogonCycle は、Siebel API からコネクタ・エージェントを切断する以外は何もしないことです。

---

## 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・インスタンス用に新規ディレクトリを作成します。
- 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。

## 新規ディレクトリの作成

それぞれのコネクタ・インスタンスごとにコネクタ・ディレクトリを作成する必要があります。このコネクタ・ディレクトリには、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・インスタンスを一意的に示します。

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

```
ProductDir¥repository¥connectorInstance
```

## ビジネス・オブジェクト定義の作成

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

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



`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。

## コネクタ定義の作成

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

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

## 始動スクリプトの作成

始動スクリプトは以下のように作成します。

1. 初期コネクタの始動スクリプトをコピーし、コネクタ・ディレクトリ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、22 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。

---

## コネクタの始動

コネクタは、**コネクタ始動スクリプト**を使用して明示的に始動する必要があります。始動スクリプトは、次に示すようなコネクタのランタイム・ディレクトリ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。始動スクリプトの名前は、表 4 に示すように、オペレーティング・システム・プラットフォームによって異なります。

表 4. コネクタの始動スクリプト

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

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

- Windows システムで「スタート」メニューから。

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

- コマンド行から。

– Windows システム:

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

– UNIX ベースのシステム:

```
connector_manager_connName -start
```

ここで、*connName* はコネクタの名前であり、*brokerName* は以下の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文字列を指定します。

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

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

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

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

このツールを使用して、コネクタのロード、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。

---

## Windows サービスとしてのコネクターの実行

コネクターのインストールを終了したら、IBM WebSphere Business Integration Adapter プログラム・グループ内で、`-t` オプションをコネクターのショートカットに追加します。コネクターは単一スレッドであるため、コネクター・ショートカットには、`-t` オプションを追加する必要があります。

コネクター・ショートカットに `-t` オプションを設定するには、次のようにします。

1. NT Services セットアップ・プログラムを実行する。
2. 「NT サービスとしてコネクターを構成する (Configure Connectors as NT Services)」ウィンドウで、コネクターを選択する。
3. 「AppEndWrapper」フィールドに、オプションとして `-t` を追加する。

`-t` オプションは、Windows サービスとしてコネクターを実行する場合にも必要になります。詳細については、「*システム・インストール・ガイド (Windows 版)*」を参照してください。

---

## コネクターの停止

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

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

```
connector_manager_connName -stop
```

ここで、`connName` はコネクターの名前です。

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

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

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

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

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

---

## 第 3 章 ビジネス・オブジェクトの開発

この章では、コネクタによるビジネス・オブジェクトの処理方法、およびビジネス・オブジェクトを変更する場合に参考になる事項について説明します。

Siebel のビジネス・オブジェクトを適切に作成または変更するには、Siebel アーキテクチャーにおけるオブジェクトの関係を理解する必要があります。以下のトピックがあります。

- 『ビジネス・オブジェクトの構造および関係』
- 29 ページの『ビジネス・オブジェクトのアプリケーション固有情報』

---

### ビジネス・オブジェクトの構造および関係

コネクタでは、Siebel アプリケーション固有ビジネス・オブジェクトに対する Create (作成)、Retrieve (検索)、Update (更新)、および Delete (削除) の各動詞がサポートされています。アプリケーション固有ビジネス・オブジェクトのセマンティクスは、そのビジネス・オブジェクト定義にカプセル化されています。コネクタ内のハードコーディングされた命令に従って、Siebel アプリケーション固有ビジネス・オブジェクトを処理するコネクタ・ロジックは存在しません。例えば、コネクタは、特定の型と数のエンティティから構成される特定のビジネス・オブジェクトを前提としません。コネクタの前提とは、任意のオブジェクトは、1 つ以上のエンティティから構成される可能性があるということです。

Siebel ビジネス・コンポーネントは、2 通りの方法で関連付けできます。同ビジネス・コンポーネントは、一価リンクにより、1 対 1 関係で関連付けることも、1 対多関係を表す複数値 (MVL: Multi-Value Link) フィールドを使用することもできます。複数値フィールドの例として、Account ビジネス・コンポーネントの Street Address フィールドがあります。ビジネス・コンポーネントは、ピック・リストによって、多対 1 関係で関連付けられます。ビジネス・コンポーネント・メソッドでは、ピック・リスト・ビジネス・コンポーネントの特定の値を検索して、その値をフィールドに設定する操作のサポートが実現されています。最終的に、ビジネス・コンポーネントは、交点テーブルを使用して、多対多関係で関連付けることが可能です。

さまざまなビジネス・コンポーネントをカプセル化するビジネス・オブジェクト・コンテキストという Siebel 概念をサポートするため、トップレベルのビジネス・オブジェクトは、該当する Siebel ビジネス・オブジェクトに対応させる必要があります。トップレベルのビジネス・オブジェクト・アプリケーション固有情報には、対応する Siebel ビジネス・オブジェクトの名前が含まれる必要があります。その場合、各トップレベル属性が、1 つの Siebel ビジネス・コンポーネントに対応することになります。

ビジネス・コンポーネントに対応するビジネス・オブジェクト定義内で、各属性を通して、単純フィールドまたは複数値グループ (MVG: Multi-Value Group) フィールド

ドが指定されます。単純属性の属性データは、単純データ型となります。MVG フィールドに対応する属性は、子 (コンテナ) ビジネス・オブジェクトとして取り扱われることになります。

このビジネス・オブジェクトの構造はメタデータの一部であり、このため、コネクタは、すべてのビジネス・オブジェクトの処理を同一方式で行うことが可能になります。コネクタでは、Siebel オブジェクトの追加は、ビジネス・オブジェクト定義が、オブジェクトに対して指定されている場合にサポートすることができます。

## キー属性の指定

Siebel ビジネス・オブジェクトの開発時、キー属性は、常時、オブジェクトの先頭に置きます。これにより、コネクタが、オブジェクトを処理する場合に、最初にキー値を取得することが保証されます。オブジェクトの他の位置にキー属性を置くと、処理エラーにつながる場合があります。オブジェクトのキー属性とは、Siebel におけるオブジェクトの ID のことです。

**注:** Siebel 対応コネクタでは、トップレベルのビジネス・オブジェクトの子 (Siebel BO) を除くと、子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列を表す属性をキー属性として指定することはサポートされていません。

**注:** Siebel 対応コネクタ用のビジネス・オブジェクト開発時には、ビジネス・オブジェクトとビジネス・コンポーネントの間に 1 対 1 関係が必ず成立する必要があります。

---

## ビジネス・オブジェクトの動詞処理

コネクタでは、Create (作成)、Retrieve (検索)、Update (更新)、および Delete (削除) の各動詞がサポートされています。ここでは、Siebel ビジネス・オブジェクト上に動詞を設定する上でのいくつかの考慮事項について説明します。

### Create

Create 動詞が、親オブジェクトのみに設定されている場合でも、すべての子オブジェクトは、Create 動詞を自動的に継承します。ただし、Create 動詞が明示的に 1 つの子オブジェクトで設定されている場合は、他のすべての子オブジェクトにおいても、明示的に設定される必要があります。

### Update

Update 動詞は、親オブジェクトに限り設定されます。

### Delete

Delete 動詞は、子オブジェクト上の動詞を考慮しません。

Siebel アプリケーションが、オブジェクトに対して削除を実行すると、オブジェクト・データがそのデータベースから物理的に削除されます。コネクタでは、事前削除トリガーを使用して、Delete イベント・オブジェクトのデータを取り込みます。Delete イベント・ビジネス・オブジェクトに必要なすべての属性が事前削除操

作で取り込まれることを確認する必要があります。詳細については、16 ページの『イベント・ビジネス・オブジェクトにおける複数属性の設定』を参照してください。

## Retrieve

ある子オブジェクトの動詞に、Retrieve がセットされている場合は、その親オブジェクトの動詞にも、Retrieve がセットされている必要があります。Retrieve 動詞は、親オブジェクト内の最初の子オブジェクトまたはコンテナ・オブジェクトのみを検索します。

コネクタでは、コラボレーションが、アプリケーション・レコードを一意的に示すキー属性を持たない場合でも、ビジネス・オブジェクト要求における Retrieve がサポートされます。この種類の Retrieve は、「非キー値検索」または「コンテンツ検索」と呼ばれます。コネクタでは、基数および子オブジェクトに対するキー属性の設定の有無に応じて、異なった Retrieve 操作が実行されます。

- 子オブジェクトが単一基数であり、1 つのキー属性が設定されている場合、コネクタは一致したレコードを返す。
- 子オブジェクトが単一基数であり、キー属性が設定されていない場合、コネクタは最初に一致したレコードを返す。
- 子オブジェクトが複数基数で、ビジネス・オブジェクト・コンテナには 1 つのオブジェクトしか含まれていなく、キーも設定されていない場合は、一致したレコードのすべてが返される。例外は、Retrieve 動詞アプリケーション固有情報により、マッチング・イベントの合計数未満の検索対象イベント最大数が定義されている場合です。
- 子オブジェクトが複数カーディナリティーで、ビジネス・オブジェクト・コンテナには複数のオブジェクトが含まれている場合は、各下位の (含まれている) オブジェクトは、単一カーディナリティー・オブジェクトとして処理される。すなわち、設定されているキーが 1 つの場合には、コネクタは、1 つの一致したレコードを返しますが、設定されているキーがない場合には、最初に一致したレコードのみを返します。

---

## ビジネス・オブジェクトのアプリケーション固有情報

ビジネス・オブジェクト定義内のアプリケーション固有情報により、コネクタは、ビジネス・オブジェクトの処理方法に関して、アプリケーション依存命令を取得します。メタデータ主導型コネクタでは、その対応しているビジネス・オブジェクトの設計方法について前提事項が規定されているため、ビジネス・オブジェクトに対する変更は、コネクタの規則に従う必要があります。そうでない場合、コネクタは、変更されたビジネス・オブジェクトを正しく処理できません。したがって、Siebel アプリケーション固有ビジネス・オブジェクトの変更または作成を行う場合は、ビジネス・オブジェクト定義内のアプリケーション固有情報が、コネクタの前提とする構文に一致することを確認する必要があります。

このセクションでは、Siebel ビジネス・オブジェクト、属性、および動詞に対するアプリケーション固有情報について説明します。

## ビジネス・オブジェクトのアプリケーション固有情報

ビジネス・オブジェクトのトップレベルにあるアプリケーション固有情報には、Siebel ビジネス・オブジェクトの名前が指定されています。例えば、親ビジネス・オブジェクト Siebel\_BO\_Account のビジネス・オブジェクト・アプリケーション固有情報には、次に示すように、Siebel Account オブジェクトが指定されています。

```
[BusinessObjectDefinition]
Name = Siebel_BO_Account
Version = 1.0.0
AppSpecificInfo = Account
```

コンテナ属性では、オブジェクト・レベルのアプリケーション固有情報は使用されません。

## 属性のアプリケーション固有情報形式

コネクタでは、単純属性とコンテナ属性に対するアプリケーション固有情報が使用されます。アプリケーション固有プロパティ・フィールドは、ObjectEventId 属性に対して空である必要があります。

### 単純属性のアプリケーション固有情報

単純属性の場合、アプリケーション固有情報は、次のテーブルのリストに示されている名前値ペアから構成されます。名前値ペアは、順序に左右されず、セミコロンで区切られます。

パラメーター	説明
nm=	Siebel ビジネス・コンポーネント内の対応するフィールドの名前。
pick=	多対 1 関係を指定する (ピック・リスト)。デフォルト値は false です。
set=	ピック・リストとのみ使用される。ピック・リストの参照する Siebel ビジネス・コンポーネント内のキーの属性名を指定する。

上記のパラメーターの使用例は、以降のセクションに記載されます。

### 単純属性のフィールド名

単純ビジネス・オブジェクト属性に対するアプリケーション固有情報には、Siebel ビジネス・コンポーネント内の対応するフィールドの名前を指定する必要があります。これに対するアプリケーション固有情報は、次のとおりです。

```
nm=fieldname
```

例えば、Siebel\_BCAccount ビジネス・オブジェクト内の Main Phone 属性に対するアプリケーション固有情報では、Main Phone Number が Siebel Account ビジネス・コンポーネント内の対応するフィールドであることが指定されます。このビジネス・オブジェクト属性内のアプリケーション固有情報を次に示します。

```
Name = Main Phone
Type = String
IsKey = false
AppSpecificInfo = nm=Main Phone Number
```



後方互換性を考慮して、Siebel フィールド名に対する属性のアプリケーション固有情報に定数 `nm` が指定されず、アプリケーション固有情報に他のパラメーターが存在しない場合は、定数 `nm` がデフォルトとなります。アプリケーション固有情報に追加パラメーターが存在する場合は、完全な形の名前値ペアが必要になります。すなわち、この属性に対して、`nm=parameter` がただ一つのパラメーターである場合には、次の 2 通りのアプリケーション固有情報形式は、同一の意味になります。

```
AppSpecificInfo =Main Phone Number
AppSpecificInfo= nm=Main Phone Number
```

## ピック・リストを使用した外部キー関係

Siebel では、2 つのビジネス・コンポーネント間の外部キー関係は、ピック・リストによって定義されます。関連するピック・リストが存在するフィールドの場合は、その `PickList` プロパティと `PickList` マップにより、2 つのビジネス・コンポーネント間の関係が定義されます。`PickList` マップ内の属性の 1 つは、通常、`Account Id` または `Product Id` などの ID となります。

ビジネス・オブジェクト内の単純属性に対して、Siebel ビジネス・コンポーネント・フィールドに関連するピック・リストが存在する場合、ビジネス・オブジェクト内の単純属性アプリケーション固有情報は、コネクタが、その情報を取得し、その単純属性を外部キーとして使用できるようにコーディングする必要があります。

属性に対してピック・リストを指定するには、ビジネス・オブジェクト内に 2 つの属性を含める必要があります。最初の属性は、関連するビジネス・コンポーネントの外部キー・フィールドを示すもので、2 番目の属性は、フィールド・プロパティとして、ピック・リストを持つビジネス・コンポーネント内のフィールドに対応するものです。2 つの属性が必要となるのは、ピック・リスト関係が、オブジェクト ID ではなく、オブジェクト名に基づくことによります。

`Pick List` 属性に対するアプリケーション固有情報において、記述 `pick=1` を使用して、この属性が、ピック・リストであることを指定します。次に、ピック・リスト内で選択するレコードを指定するには、記述 `set=fieldname` を使用します。`set` パラメーターにより、選択するレコードを発見するため、関連付けられているオブジェクトのどの属性を検索指定で使用するかを指定します。

例えば、`Siebel_BCAsset` ビジネス・オブジェクトを作成し、そのビジネス・オブジェクト内のある属性を外部キーとして、`Siebel_BCInternalProduct` ビジネス・オブジェクトに追加することを考えているとします。`Siebel Asset Mgmt` ビジネス・コンポーネントにおける `Product Name` フィールドは、`Internal Product` ビジネス・コンポーネントに対するピック・リストであるため、外部キーに対する属性およびピック・リストに対するもう一つの属性を追加することになります。これらの属性を、次に示すように、このビジネス・オブジェクト内で定義します。

```
[Key Attribute]
Name = ProductId
Type = String
Cardinality = 1
IsForeignKey = true
AppSpecificInfo = Product Id
```

```
[Pick List Attribute]
Name = ProductName
```

```
Type = string
Cardinality = n
AppSpecificInfo = nm=ProductName;pick=1;set=Id
```

場合によって、ピック・リスト値は、複数の属性に基づいて選択されます。例えば、同名の Account が複数ある場合、Contact 検索では、Account 名が、ただ一つのピック・リスト値として設定されているときには、その名前の最初の Account が取得されます。正しいデータが確実に検索されるようにするため、ピック・リストを複数のフィールドで限定することができます。次の例では、Contact ビジネス・オブジェクトは、Account、Site、および City によって限定されます。

```
[Key Attribute]
Name=ContactId
Type=String
Cardinality=1
AppSpecificInfo=Id
```

```
Name=Last_Name
Type=String
Cardinality=1
AppSpecificInfo=Last Name
```

```
Name=First_Name
Type=String
Cardinality=1
AppSpecificInfo=First Name
```

```
Name=Site
Type=String
Cardinality=1
AppSpecificInfo=N/A
```

```
Name=City
Type=String
Cardinality=1
AppSpecificInfo=N/A
```

```
Name=Account
Type=String
Cardinality=1
AppSpecificInfo=nm=Account;pick=1;set=Name;restrict=Location:Site,City:City
```

ピック・リスト・フィールドを限定する AppSpecificInfo は、次の構文に従います。

```
restrict=<column name>:<Siebel GUI name>,<column name>:<Siebel GUI name>
```

限定フィールドの個数には制限はありません。set パラメーターの後の属性間にスペースは指定しないでください。すべての限定フィールドは、ビジネス・オブジェクトに属性として追加する必要があり、また、AppSpecificInfo を持たせないようにします。AppSpecificInfo 属性は、限定フィールドではプレースホルダーとして機能します。

Retrieve では、アプリケーション固有情報 pick=1 により、ProductName 属性が、ピック・リスト・ビジネス・コンポーネントに対応することが指定され、set パラメーターにより、Id の値が、コネクタの選択するレコードを表すものであることが指定されます。

Create では、Asset ビジネス・コンポーネントとの包含関係を持つピック・リストに対して、新規の Product ビジネス・コンポーネントを、次のように作成する必要があります。

```
[Key Attribute]
Name = ProductId
Type = String
Cardinality = 1
IsForeignKey = true
AppSpecificInfo = Product Id
```

```
[Pick List Attribute]
Name = Siebel_BCInternalProduct
Type = Siebel_BCInternalProduct
Relationship = containment
AppSpecificInfo = nm=Siebel_BCInternalProduct;pick=1;set=Id
```

Pick List 属性に対する次のマッピングのヒントに注意してください。

- Retrieve 操作により、Pick List 属性の値を Pick List ビジネス・コンポーネントの名前に対応させ、キー属性の値をキーに対応させる。
- Create または Update 操作により、Pick List 属性をキーに対応させ、キー属性をヌル値に対応させる。Pick List リンクは、フィールドの名前により定義されるため、コネクタでは、キー属性値に任意の値を設定することが可能であり、Siebel は、その値を検証しません。Pick List 属性にキー値が含まれていて、ピック操作が、Pick List コンポーネントを使用して実行される場合は、キーの検証が行われます。ピック操作がフィールドを検出すると、新規オブジェクトに対応するピック内のすべての属性を追加し、新規オブジェクトが作成されます。
- ピック・リストからリンクを削除するには、Pick List 属性の値をヌルに、キー属性の値をブランクに、それぞれ対応付ける。

## コンテナ属性のアプリケーション固有テキスト

コンテナ属性の場合、アプリケーション固有情報には、次のテーブルのリストにまとめられている名前値ペアが記述されます。名前値ペアは順序に左右されず、セミコロンで区切ります。

パラメーター	説明
nm= mvg=1 pick=1 assc=1	Siebel ビジネス・コンポーネント内の複数値フィールドの名前。 Siebel ビジネス・コンポーネント間の関係型を指定する。mvg=1 は 1 対多関係を指定する。pick=1 は多対 1 関係、assc=1 は多対多関係を指定する。関係型の指定されていない場合、デフォルト値は mvg=1 である。
to=;from=	コネクタが、to 属性に from 属性の値を設定するためのプリプロセス命令。from 属性には、このテキスト・パラメーターを含む属性の処理前に値が設定される必要がある。to 属性は、ヌルの場合に限り設定される。このテキストは、属性を含むオブジェクトが、1 対 1 関係を持っている場合に限り使用することができる。
keep=1 delOnly=1	Update 操作に対する制限を指定する。

## コンテナ属性のフィールド名

子ビジネス・オブジェクトを参照するコンテナ属性に対するアプリケーション固有情報には、その親ビジネス・コンポーネントに関連する複数値フィールドの名前を指定する必要があります。複数値フィールドは、Siebel におけるビジネス・コンポーネントの親子関係を定義する複数値リンクを表します。これに対するアプリケーション固有情報は、次のとおりです。

nm=multiValueFieldName

例えば、ビジネス・オブジェクト Siebel\_BCContact が、子ビジネス・オブジェクト Siebel\_BCBusinessAddress に対するコンテナ属性を持つものとして、このコンテナ属性に対するアプリケーション固有情報には、Siebel Business Address ビジネス・コンポーネントに対するリンクを含む多値フィールドとして、Street Address が指定されます。

```
[Example of Container Attribute]
Name = PrimaryAddress
Type = Siebel_BC_BusinessAddress
ContainedObjectVersion = 1.0.0
Relationship = Containment
Cardinality = n
IsKey = false
IsForeignKey = false
AppSpecificInfo = nm=Street Address
```

後方互換性を考慮して、Siebel フィールド名に対する属性のアプリケーション固有情報に定数 nm が指定されず、アプリケーション固有情報に他のパラメーターが存在しない場合は、定数 nm がデフォルトとなります。アプリケーション固有情報に追加パラメーターが存在する場合は、完全な形の名前値ペアが必要になります。

## ビジネス・オブジェクト間の親子関係

フィールド名に加えて、コンテナ属性に対するアプリケーション固有情報には、Siebel における親子ビジネス・コンポーネントの関係型を定義するパラメーターを含めることができます。関係型を指定するには、次の値を使用します。

パラメーター	説明
mvg=1	関係が、1 対多関係または Siebel 複数値リンクであることを指定する。
pick=1	関係がピック・リストであることを指定する。親子のビジネス・オブジェクト間でピック・リスト関係を指定する方法の詳細については、37 ページの『ピック・リスト関係の指定』を参照してください。
assc=1	交点テーブルにより、関係が多対多関係であることを指定する。この場合、Create 操作により、コネクタは、コンテナ内のビジネス・オブジェクトの値の設定されているフィールドを使用して、ビジネス・コンポーネントを検索します。コネクタは、一致するオブジェクトを検出すると、それを親ビジネス・コンポーネントに関連付けます。オブジェクトが検出されない場合は、エラーがログに記録され、ビジネス・オブジェクト要求は失敗します。
nomv=1	親子のオブジェクト間にリンクまたは複数値リンク関係があるものの、親ビジネス・オブジェクトには複数値フィールドが存在しないことを指定する。この場合、Transactional プロパティを設定する必要があります。Transactional プロパティの設定方法の詳細については、20 ページの『コネクタ固有のプロパティ』を参照してください。

関係パラメーターが指定されていない場合は、mvg=1 が使用されます。関係型パラメーターに 0 を設定することは有効ではありません。関係型に false を設定した場合は、そのパラメーターをインクルードしないでください。

例として、前のセクションで示された、子ビジネス・オブジェクト Siebel\_BCBusinessAddress に対するコンテナ属性は、パラメーター `mvg=1` を含むと、Street Address が、Siebel Contact ビジネス・コンポーネントを Siebel Business Address ビジネス・コンポーネントにリンクする複数値フィールドであることを示します。

```
[Example of Container Attribute]
Name = PrimaryAddress
Type = Siebel_BC_BusinessAddress
ContainedObjectVersion = 1.0.0
Relationship = Containment
Cardinality = n
IsKey = false
IsForeignKey = false
AppSpecificInfo = nm=Street Address;mvg=1
```

## ある属性の値を別の属性に割り当てる操作

属性アプリケーション固有情報は、コネクタがある属性の値を取得した後、その値を別の属性に割り当ててから、後者の属性を処理するようにコーディングすることができます。この機能は、Retrieve 操作で使用され、検索する子レコードを指定するため、コンテナ属性で主に使用されます。

この機能を使用するには、ビジネス・オブジェクト定義内の属性アプリケーション固有情報を編集して、次の記述を含めます。

```
from=attribute path;to=attribute path;
```

属性パス値は、次のいずれかを取ることができます。

- 現在のビジネス・オブジェクト内の属性名。
- スラッシュ (/) が先頭に来る完全修飾パス。パスは、その先頭がトップレベルのビジネス・オブジェクトの属性である必要があり、また、Siebel またはビジネス・オブジェクト名ではなく、ビジネス・オブジェクト属性名から構成されることに注意してください。図については、図 2 を参照してください。
- `business component name/attribute name` 形式の簡略パス。図については、図 3 を参照してください。

絶対パスは、トップレベル・ビジネス・オブジェクトから階層を下って進みます。`from=;to=;` 組パラメーターは、単純属性またはコンテナ属性のすべてで使用でき、属性の処理前に動作します。次の規則に注意してください。

- `from` 属性は、命令中の `to` 属性の処理前に値が設定される必要がある。
- `to` 属性は、ヌル値の場合に限り設定される。
- パスが、`from` パラメーターにとって無効である場合は、`to` パラメーターにはヌルが設定される。パスが `to` パラメーターにとって無効であっても、エラー・フラグは立ちません。
- `from/to` 疑似命令は、子ビジネス・オブジェクトの属性のアプリケーション固有情報においてのみ指定できる。すなわち、トップレベル・ビジネス・オブジェクトでは指定できません。

例えば、Siebel\_BCAccount ビジネス・オブジェクトに、子ビジネス・オブジェクト Siebel\_BCBusinessAddress が含まれている場合は、Siebel\_BCAccount オブジェクト内の属性により、ピック・リストを検索するアドレスを指定することができます。この例では、PrimaryAddressID がキー属性であり、Siebel\_BCBusinessAddress

がピックアップ対象のオブジェクトです。コネクターは、PrimaryAddressID 属性の値を取得し、その値を使用して、特定のアドレスを検索します。子属性は、親ビジネス・オブジェクトの属性の後に処理されます。次の図では、親から子へのビジネス・オブジェクトの属性の処理フローを示します。

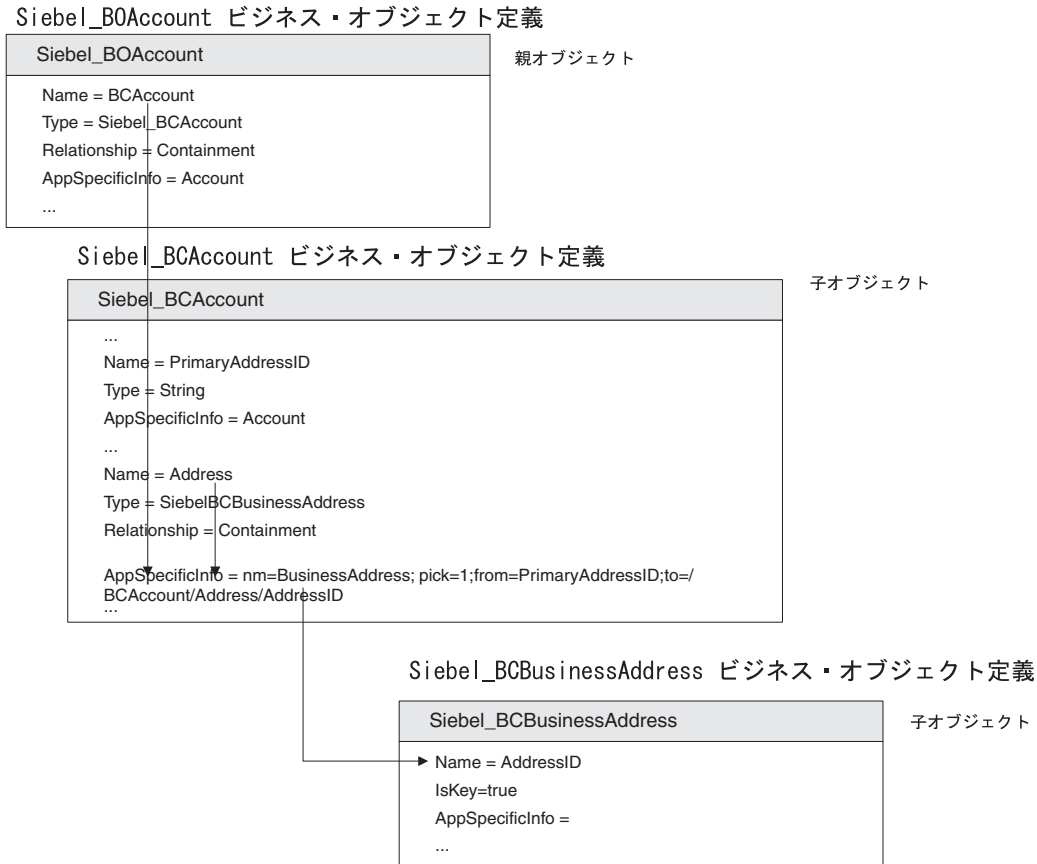


図2. 絶対パスを使用して、ある属性の値を別の属性に割り当てる

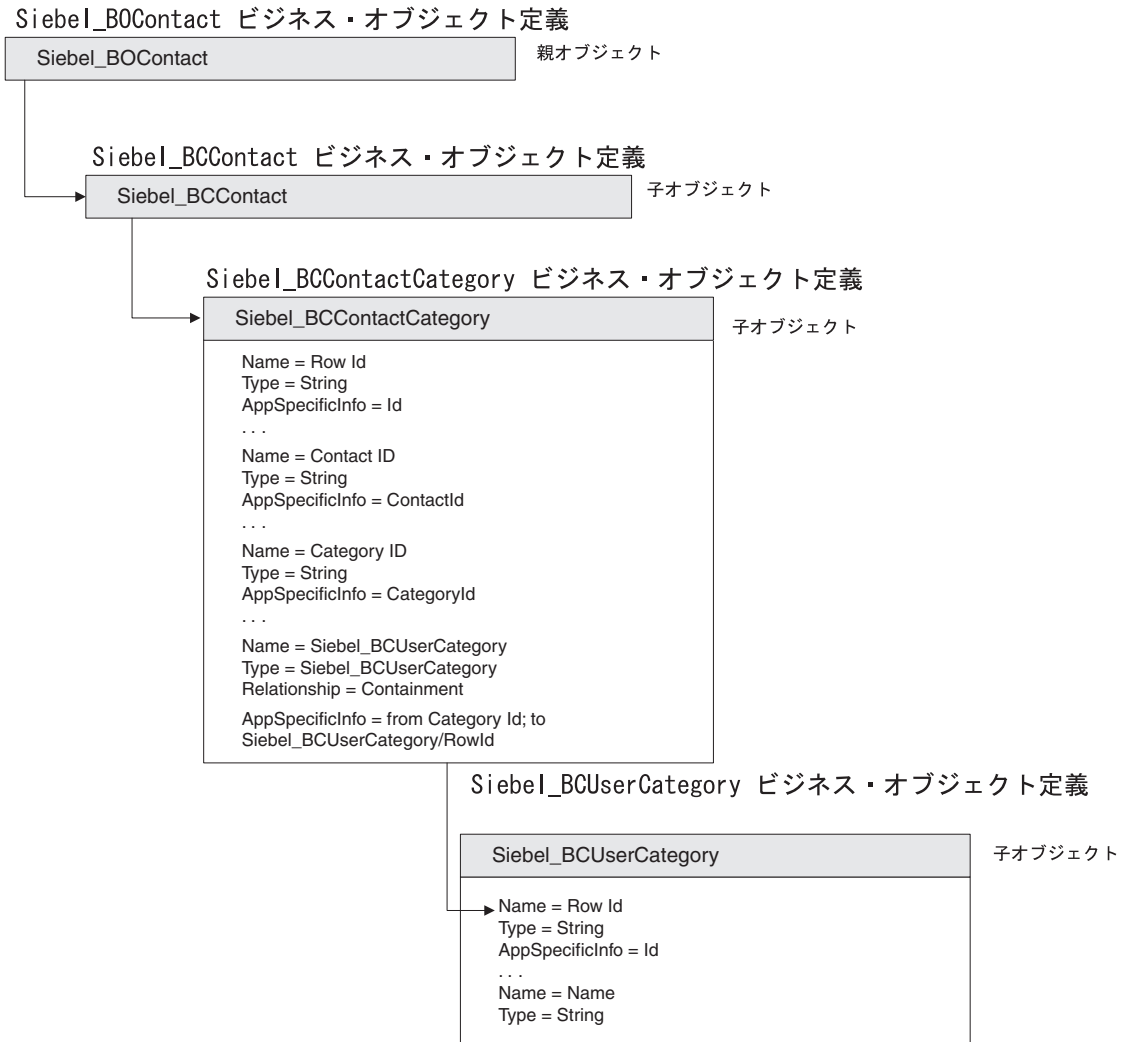


図 3. 簡略パスを使用して、ある属性の値を別の属性に割り当てる

### ピック・リスト関係の指定

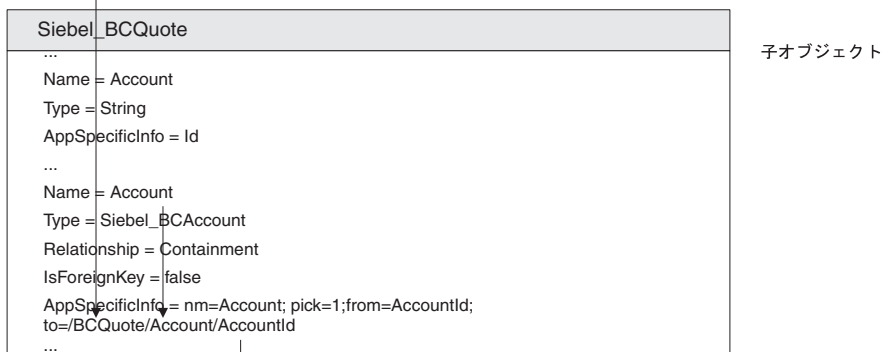
ピック・リスト関係の中には、同一のトランザクション内でピック対象の子オブジェクトを作成することを必要とするものがあります。IBM WebSphere Business Integration Adapter ビジネス・オブジェクトでは、親子のビジネス・オブジェクト間のピック・リスト関係は、キー属性とピック対象のオブジェクトに対する単一カーディナリティーのコンテナ属性という 2 つの属性によって表されます。この属性セットは、PickList マップに含まれていないピック・リスト・ビジネス・コンポーネントの属性の全部または一部を検索するために使用できます。

例えば、Siebel\_BCQuote ビジネス・オブジェクトは、Quote と Account 間のピック・リスト関係を指定するために 2 つの属性を含むように設計されています。下図に示すように、AccountId がキー属性で、Account がピック・リスト・オブジェクトです。

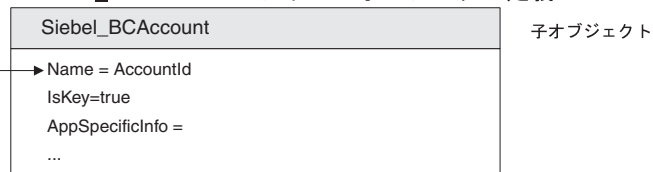
### Siebel\_BOQuote ビジネス・オブジェクト定義



### Siebel\_BCQuote ビジネス・オブジェクト定義



### Siebel\_BCAccount ビジネス・オブジェクト定義



アプリケーション固有情報において、pick=1 は、コンテナ属性がピック・リストを表し、from= パラメーターがキー属性に対するポインターであり、to= パラメーターが Siebel\_BCAccount ビジネス・オブジェクトのキー属性を指していることを示します。

属性の順序が、Retrieve 操作で重要なのは、AccountId の値をまず検索しないと、子オブジェクトの外部キーとしてその値を定義できないためです。Create 操作または Update 操作では、AccountId の値は外部キーであり、オブジェクトの作成後に検索されます。

ピック・リスト・コンテナとして完全なビジネス・オブジェクトを使用する必要はありません。必須のキー・セットのみを持つオブジェクトで十分です。コネクタは、ピック・リスト・コンテナの処理に次の規則を使用します。

- ピック・リスト・ビジネス・オブジェクトのキー属性が一切設定されていない場合は、新規オブジェクトが、Siebel において作成され、選択される。
- ピック・リスト・ビジネス・オブジェクトのキー属性のいずれかが設定されている場合は、コネクタは、そのオブジェクトを検索し、選択する。そのビジネス・オブジェクトのピック・リスト・オブジェクトが検出されない場合は、コネクタは、エラーをログに記録します。エラーは、オブジェクト・キーが有効でない場合に発生することがあります。



次に、コンテナ・ビジネス・オブジェクト上の Pick List 属性のマップに対するガイドラインを示します。

- コラボレーションからコネクタへのビジネス・オブジェクト要求である場合、キー属性の対応は、前述の単純属性の場合と同一のガイドラインに従う。
- コンテナ属性の対応は、既知のキーの場合に限りキーとする。
- ピック・リスト・オブジェクトを作成する場合は、そのピック・リスト・オブジェクトに対する指定に従って、必要な属性をすべて対応付ける。
- Delete 操作では、キー属性には 1 つのスペース文字を設定し、ピック・リスト・コンテナ属性には null を設定する。

## 複数カーディナリティー・オブジェクトの更新操作に対する制約の指定

階層ビジネス・オブジェクトに対する Update 操作により、コネクタは、まず、アプリケーションからデータを検索し、エンティティーの現在値を含む 1 つの階層ビジネス・オブジェクトを作成します。次に、ビジネス・オブジェクト要求内のビジネス・オブジェクトをアプリケーション・ビジネス・オブジェクトと比較して、そのアプリケーション内に存在しない子ビジネス・オブジェクトに対するレコードを作成し、同アプリケーション内に存在するレコードを更新し、同アプリケーション内に存在するが、コラボレーション・ビジネス・オブジェクトに存在しないレコードを削除します。

このデフォルトの振る舞いに追加して、keep パラメーターと delOnly パラメーターを使用して、子オブジェクトを更新するようにコネクタの振る舞いを設定することができます。

- 子 (下位) オブジェクトを参照する属性の場合、属性アプリケーション固有情報は、コネクタが既存の関係を維持しながら、Update 操作の間に追加される新規子ビジネス・オブジェクトに対して新しい関係を作成するように指定することができます。デフォルト値は、ビジネス・オブジェクト要求内で、既存の関係が表現されていない場合に、その関係を表す行を削除することです。

既存の関係を維持するには、アプリケーション固有情報を編集して、ストリング keep=1; を記述します。この変更によって、コネクタは、子オブジェクトの Delete 動詞を無視するように指示されます。例えば、既存の連絡先が、1 つのアドレスに関連付けられ、コネクタが、2 つの新規アドレスを表す子ビジネス・オブジェクトを含むが、既存のアドレスに対する子ビジネス・オブジェクトを含まない Update ビジネス・オブジェクトを受け取ったものとします。コンテナ属性アプリケーション固有情報に、値 keep=1; が指定されている場合、連絡先は、既存アドレスと新規アドレスの両方を持つように更新されます。このアプリケーション固有情報が空である場合は、既存のアドレスを含む行は削除され、新規のアドレスが追加されます。

属性アプリケーション固有情報には、コネクタが、Delete 操作のみを実行するように指定できます。このオプションは、ラッパー・コラボレーションの場合に使用されます。既存の関係を削除するには、アプリケーション固有情報を編集して、ストリング delOnly=1; を記述するようにします。

- keep=1 あるいは delOnly=1 のいずれも指定されていない場合、コネクタは、すべての子ビジネス・オブジェクトを、子オブジェクトの動詞に基づいて処理します。

## 動詞アプリケーション固有情報の形式

ビジネス・オブジェクト Retrieve 動詞に対するアプリケーション固有情報には、コネクタが、検索のたびに限定された数のオブジェクトのみを検索するように指定できます。オブジェクトの一部を検索するアプリケーション固有情報は max=n です。5 つのオブジェクトのみを検索するように指定された Retrieve 動詞の例は、次のとおりです。

```
[Verb]
Name = Retrieve
AppSpecificInfo = max=5
```

他の動詞の場合、アプリケーション固有プロパティは使用されないため、ビジネス・オブジェクト定義の作成時、ブランクのままにするか省略するかします。

## Create 動詞と Update 動詞のキー属性

Create または Update 要求では、Object Key 値が RowId と異なる場合、Siebel アプリケーションは、Object Key 属性を消去してそのレコード用の独自の RowId を作成します。

**注:** Create 要求と Update 要求において、キー属性として、RowId を使用することをお勧めします。

---

## 第 4 章 トラブルシューティング

この章では、Siebel 用コネクタの実行時に表示される可能性のあるエラーおよび可能なエラー解決策について説明します。

---

### Init() での失敗のエラー

コネクタの実行時に、次のエラー・メッセージが表示される場合があります。

メッセージ: Siebel 警告: Init() に対する init() が失敗しました。

このエラー・メッセージは、次のことが原因で発生する可能性があります。

- Siebel アプリケーションへのログイン時に表示されるポップアップ・ウィンドウまたはメッセージ・ボックスがある。コネクタは、アプリケーションへの接続中にポップアップ・ウィンドウまたはメッセージ・ボックスを処理できないため、失敗します。
- トリガーまたは Siebel カスタム・トリガーにポップアップ・ウィンドウまたはメッセージ・ボックスがある。コネクタは、これらのトリガーの実行中にポップアップ・ウィンドウまたはメッセージ・ボックスを処理できないため、失敗します。
- コネクタが実行されているマシンの siebeljpn.cfg ファイルが Siebel アプリケーションの cfg ファイルと異なるスクリプト言語 (VBScript または Escript) に設定されている。

---

### アプリケーションへの接続不可

Siebel アプリケーション接続が失われると、Siebel がエラー 4096 を返します。このエラー番号は、Siebel のバージョンに従って、それぞれ別のメッセージ文に関連付けられます。Siebel では、エラー・メッセージが文書化されていないため、ConnectError プロパティが追加され、Siebel の特定バージョン用のメッセージを設定できるようになっています。指定されたメッセージを受信すると、コネクタは停止します。このプロパティのデフォルト値は、Siebel 99.5 の終了時点で受信されたエラー・メッセージ文となります。詳細については、20 ページの表 3 を参照してください。



---

## 付録 A. コネクターの標準構成プロパティ

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

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

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

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

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

---

### 新規プロパティと削除されたプロパティ

以下の標準プロパティは、本リリースで追加されました。

#### 新規プロパティ

- XMLNamespaceFormat

#### 削除されたプロパティ

- RestartCount

---

### 標準コネクター・プロパティの構成

アダプター・コネクターには 2 つのタイプの構成プロパティがあります。

- 標準構成プロパティ
- コネクター固有の構成プロパティ

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

## Connector Configurator の使用

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

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

## プロパティ値の設定と更新

プロパティ・フィールドのデフォルトの長さは 255 文字です。

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

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

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

### • 動的

変更を System Manager に保管すると、変更が即時に有効になります。例えば WebSphere Message Brokers で稼働している場合など、コネクタがスタンドアロン・モードで (System Manager から独立して) 稼働している場合は、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)

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

### • コンポーネント再始動

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

### • サーバー再始動

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

特定のプロパティの更新方法を確認するには、「Connector Configurator」ウィンドウ内の「更新メソッド」列を参照するか、次に示す 45 ページの表 5 の「更新メソッド」列を参照してください。

## 標準プロパティの要約

表 5 は、標準コネクタ構成プロパティの早見表です。標準プロパティの依存関係は RepositoryDirectory に基づいているため、コネクタによっては使用されないプロパティがあり、使用する統合ブローカーによってプロパティの設定が異なる可能性があります。

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

**注:** 表 5 の「注」列にある「Repository Directory は REMOTE」という句は、ブローカーが InterChange Server であることを示します。ブローカーが WMQI または WAS の場合には、リポジトリ・ディレクトリは LOCAL に設定されます。

表 5. 標準構成プロパティの要約

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

表 5. 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
ControllerStoreAndForwardMode	true または false	true	動的	Repository Directory は <REMOTE> (ブローカーは ICS)
ControllerTraceLevel	0 から 5	0	動的	Repository Directory は <REMOTE> (ブローカーは ICS)
DeliveryQueue		CONNECTORNAME/DELIVERYQUEUE	コンポーネント再始動	JMS トランスポートのみ
DeliveryTransport	MQ、IDL、または JMS	JMS	コンポーネント再始動	Repository Directory がローカルの場合、値は JMS のみ
DuplicateEventElimination	true または false	false	コンポーネント再始動	JMS トランスポートのみ: Container Managed Events は <NONE> でなければならない
FaultQueue		CONNECTORNAME/FAULTQUEUE	コンポーネント再始動	JMS トランスポートのみ
jms.FactoryClassName	CxCommon.Messaging.jms.IBMMQSeriesFactory または CxCommon.Messaging.jms.SonicMQFactory または任意の Java クラス名	CxCommon.Messaging.jms.IBMMQSeriesFactory	コンポーネント再始動	JMS トランスポートのみ
jms.MessageBrokerName	FactoryClassName が IBM の場合は crossworlds.queue.manager を使用。 FactoryClassName が Sonic の場合 localhost:2506 を使用。	crossworlds.queue.manager	コンポーネント再始動	JMS トランスポートのみ
jms.NumConcurrentRequests	正整数	10	コンポーネント再始動	JMS トランスポートのみ
jms.Password	任意の有効なパスワード		コンポーネント再始動	JMS トランスポートのみ
jms.UserName	任意の有効な名前		コンポーネント再始動	JMS トランスポートのみ



表 5. 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	Repository Directory は <REMOTE> (ブローカーは ICS)
JvmMaxNativeStackSize	スタックのサイズ (キロバイト単位)	128k	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)
JvmMinHeapSize	ヒープ・サイズ (メガバイト単位)	1m	コンポーネント再始動	Repository Directory は <REMOTE> (ブローカーは ICS)
ListenerConcurrency	1 から 100	1	コンポーネント再始動	Delivery Transport は MQ でなければならぬ
Locale	en_US、ja_JP、ko_KR、zh_CN、zh_TW、fr_FR、de_DE、it_IT、es_ES、pt_BR 注: これは、サポートされるロケールの一部です。	en_US	コンポーネント再始動	
LogAtInterchangeEnd	true または false	false	コンポーネント再始動	Repository Directory は <REMOTE> でなければならぬ (ブローカーは ICS)
MaxEventCapacity	1 から 2147483647	2147483647	動的	Repository Directory は <REMOTE> でなければならぬ (ブローカーは ICS)
MessageFileName	パスまたはファイル名	CONNECTORNAMEConnector.txt	コンポーネント再始動	
MonitorQueue	任意の有効なキュー名	CONNECTORNAME/MONITORQUEUE	コンポーネント再始動	JMS トランスポートのみ: DuplicateEvent Elimination は true でなければならぬ

表 5. 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	Repository Directory は <REMOTE> でなければなりません (ブローカーは ICS)
OADMaxNumRetry	正数	1000	動的	Repository Directory は <REMOTE> でなければなりません (ブローカーは ICS)
OADRetryTimeInterval	正数 (単位: 分)	10	動的	Repository Directory は <REMOTE> でなければなりません (ブローカーは ICS)
PollEndTime	HH:MM	HH:MM	コンポーネント再始動	
PollFrequency	正整数 (単位: ミリ秒)  no (ポーリングを使用不可にする)  key (コネクタのコマンド・プロンプト・ウィンドウで文字 p が入力された場合にのみポーリングする)	10000	動的	
PollQuantity	1 から 500	1	エージェント再始動	JMS トランスポートのみ: Container Managed Events を指定
PollStartTime	HH:MM (HH は 0 から 23、MM は 0 から 59)	HH:MM	コンポーネント再始動	
RepositoryDirectory	メタデータ・リポジトリの場所		エージェント再始動	ICS の場合は <REMOTE> に設定する。 WebSphere MQ Message Brokers および WAS の場合: C:\crossworlds¥repository に設定する

表 5. 標準構成プロパティの要約 (続き)

プロパティ名	指定可能な値	デフォルト値	更新メソッド	注
RequestQueue	有効な JMS キュー名	CONNECTORNAME/REQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
ResponseQueue	有効な JMS キュー名	CONNECTORNAME/RESPONSEQUEUE	コンポーネント再始動	Delivery Transport が JMS の場合: Repository Directory が <REMOTE> の場合のみ必要
RestartRetryCount	0 から 99	3	動的	
RestartRetryInterval	適切な正数 (単位: 分): 1 から 2147483547	1	動的	
RHF2MessageDomain	mrm, xml	mrm	コンポーネント再始動	Delivery Transport が JMS であり、かつ WireFormat が CwXML である
SourceQueue	有効な WebSphere MQ 名	CONNECTORNAME/SOURCEQUEUE	エージェント再始動	Delivery Transport が JMS であり、かつ Container Managed Events が指定されている場合のみ
SynchronousRequestQueue		CONNECTORNAME/ SYNCHRONOUSREQUESTQUEUE	コンポーネント再始動	Delivery Transport は JMS
SynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	Delivery Transport は JMS
SynchronousResponseQueue		CONNECTORNAME/ SYNCHRONOUSRESPONSEQUEUE	コンポーネント再始動	Delivery Transport は JMS
WireFormat	CwXML、CwBO	CwXML	エージェント再始動	Repository Directory が <REMOTE> でない場合は CwXML。Repository Directory が <REMOTE> であれば CwBO
WsifSynchronousRequestTimeout	0 以上の任意の数値 (ミリ秒)	0	コンポーネント再始動	WAS のみ
XMLNamespaceFormat	short、long	short	エージェント再始動	WebSphere MQ Message Brokers および WAS のみ

---

## 標準構成プロパティ

このセクションでは、各標準コネクタ構成プロパティの定義を示します。

### AdminInQueue

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

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

### AdminOutQueue

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

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

### AgentConnections

`RepositoryDirectory` が `<REMOTE>` の場合のみ適用可能です。

`AgentConnections` プロパティは、`orb.init[]` により開かれる ORB (オブジェクト・リクエスト・ブローカー) 接続の数を制御します。

このプロパティのデフォルト値は 1 に設定されます。必要に応じてこの値を変更できます。

### AgentTraceLevel

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

### ApplicationName

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

### BrokerType

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

### CharacterEncoding

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

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

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

## ConcurrentEventTriggeredFlows

`RepositoryDirectory` が `<REMOTE>` の場合のみ適用可能です。

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

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

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

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

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

## ContainerManagedEvents

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

デフォルト値はありません。

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

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

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

これらのプロパティーはアダプター固有ですが、例の値は次のようになります。

- 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 を JMS に設定した場合にのみ表示されます。

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

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

## ControllerStoreAndForwardMode

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

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

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

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

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

デフォルト値は true です。

## ControllerTraceLevel

RepositoryDirectory が <REMOTE> の場合のみ適用可能です。

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

## DeliveryQueue

DeliveryTransport が JMS の場合のみ適用されます。

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

デフォルト値は CONNECTORNAME/DELIVERYQUEUE です。

## DeliveryTransport

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

- RepositoryDirectory がリモートの場合は、DeliveryTransport プロパティの指定可能な値は MQ、IDL、または JMS であり、デフォルトは IDL になります。
- RepositoryDirectory がローカル・ディレクトリーの場合は、指定可能な値は JMS のみです。

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

## WebSphere MQ および IDL

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

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

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

## JMS

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

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

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

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

この環境では、WebSphere MQ クライアント内でメモリーが使用されるため、(サーバー側の) コネクター・コントローラーと (クライアント側の) コネクターの両方を始動するのは困難な場合があります。ご使用のシステムのプロセス・ヒープ・サイズが 768M 未満である場合には、次のように設定することをお勧めします。

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

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

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

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

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

## DuplicateEventElimination

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

このプロパティーは、`false` に設定することもできます。

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



## FaultQueue

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

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

## JvmMaxHeapSize

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

デフォルト値は `128M` です。

## JvmMaxNativeStackSize

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

デフォルト値は `128K` です。

## JvmMinHeapSize

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

デフォルト値は `1M` です。

## jms.FactoryClassName

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

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

## jms.MessageBrokerName

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

デフォルト値は `crossworlds.queue.manager` です。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。

リモート・メッセージ・ブローカーに接続すると、このプロパティは次の (必須) 値をとります。

`QueueMgrName:<Channel>:<HostName>:<PortNumber>`

各変数の意味は以下のとおりです。

`QueueMgrName`: キュー・マネージャー名です。

`Channel`: クライアントが使用するチャンネルです。

`HostName`: キュー・マネージャーの配置先のマシン名です。

`PortNumber`: 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、次のようになります。

```
jms.MessageBrokerName = WBIMB.Queue.Manager:CHANNEL1:RemoteMachine:1456
```

## jms.NumConcurrentRequests

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

デフォルトはありません。

## jms.UserName

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

デフォルトはありません。

## ListenerConcurrency

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

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

## Locale

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

ロケール名は、次の書式で指定します。

```
ll_YY.codeset
```

ここで、以下のように説明されます。

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

デフォルトでは、ドロップダウン・リストには、サポートされるロケールの一部のみが表示されます。ドロップダウン・リストに、サポートされる他の値を追加する

には、製品ディレクトリーにある `¥Data¥Std¥stdConnProps.xml` ファイルを手動で変更する必要があります。詳細については、本書の Connector Configurator に関する付録を参照してください。

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

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

## LogAtInterchangeEnd

RepositoryDirectory が `<REMOTE>` の場合のみ適用可能です。

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

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

## MaxEventCapacity

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

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

## MessageFileName

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

## MonitorQueue

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

デフォルト値は CONNECTORNAME/MONITORQUEUE です。

## OADAutoRestartAgent

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

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

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

デフォルト値は false です。

## OADMaxNumRetry

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

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

デフォルト値は 1000 です。

## OADRetryTimeInterval

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

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

デフォルト値は 10 です。

## PollEndTime

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

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

## PollFrequency

これは、前回のポーリングの終了から次のポーリングの開始までの間の間隔です。PollFrequency 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 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。一部のアダプターでは、これは個別のスレッドで部分的に実行されます。これにより、次のポーリング・アクションまで処理が非同期に実行されます。
- PollFrequency 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。

PollFrequency は以下の値のいずれかに設定します。

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

デフォルト値は 10000 です。

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

## PollQuantity

コネクターがアプリケーションからポーリングする項目の数を指定します。アダプターにコネクター固有のポーリング数設定プロパティがある場合、標準プロパティの値は、このコネクター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。

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

コネクターは、1 回目のポーリングを受けると、メッセージの本文を選出します。これは、本文が添付とも見なされるからです。本文の MIME タイプにはデータ・ハンドラーが指定されていないので、コネクターは本文を無視します。

コネクターは PO の最初の添付を処理します。この添付の MIME タイプには対応する DH があるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。

2 回目のポーリングを受けると、コネクターは PO の 2 番目の添付を処理します。この添付の MIME タイプには対応する DH があるので、コネクターはビジネス・オブジェクトを Visual Test Connector に送信します。

これが受け入れられると、PO の 3 番目の添付が届きます。

## PollStartTime

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

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

## RequestQueue

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

デフォルト値は CONNECTOR/REQUESTQUEUE です。

## RepositoryDirectory

コネクターが XML スキーマ文書を読み取るリポジトリの場所です。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 の場合は、この値を <local directory> に設定する必要があります。

## ResponseQueue

DeliveryTransport が JMS の場合のみ適用可能で、RepositoryDirectory が <REMOTE> の場合のみ必須です。

JMS 応答キューを指定します。JMS 応答キューは、応答メッセージをコネクター・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。

## RestartRetryCount

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

デフォルト値は 3 です。

## RestartRetryInterval

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

デフォルト値は 1 です。

## RHF2MessageDomain

WebSphere Message Brokers および WAS でのみ使用されます。

このプロパティにより、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 WMQI 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。この構成可能なドメイン名により、ユーザーは WMQI ブローカーによるメッセージ・データの処理方法を追跡できます。

サンプル・ヘッダーを以下に示します。

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

デフォルト値は `mrm` ですが、このプロパティには `xml` も設定できます。このプロパティは、`DeliveryTransport` が JMS に設定されており、かつ `WireFormat` が `CwXML` に設定されている場合にだけ表示されます。

## SourceQueue

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

JMS イベント・ストアを使用する JMS 対応コネクタでの保証付きイベント・デリバリーをサポートするコネクタ・フレームワークに、JMS ソース・キューを指定します。詳細については、51 ページの『`ContainerManagedEvents`』を参照してください。

デフォルト値は `CONNECTOR/SOURCEQUEUE` です。

## SynchronousRequestQueue

`DeliveryTransport` が JMS の場合のみ適用されます。

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

デフォルトは `CONNECTORNAME/SYNCHRONOUSREQUESTQUEUE` です。

## SynchronousResponseQueue

`DeliveryTransport` が JMS の場合のみ適用されます。

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

デフォルトは `CONNECTORNAME/SYNCHRONOUSRESPONSEQUEUE` です。

## SynchronousRequestTimeout

`DeliveryTransport` が JMS の場合のみ適用されます。

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

デフォルト値は 0 です。

## WireFormat

トランスポートのメッセージ・フォーマットです。

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

## WsifSynchronousRequestTimeout

WAS 統合ブローカーでのみ使用されます。

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

デフォルト値は 0 です。

## XMLNamespaceFormat

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

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

デフォルト値は short です。



---

## 付録 B. Connector Configurator

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

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

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

### 注:

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

この付録では、次のトピックについて説明します。

- 『Connector Configurator の概要』
- 64 ページの『Connector Configurator の始動』
- 65 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』
- 68 ページの『新規構成ファイルの作成』
- 71 ページの『構成ファイル・プロパティの設定』
- 80 ページの『グローバル化環境における Connector Configurator の使用』

---

## Connector Configurator の概要

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

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

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

- コネクタを構成するためのコネクタ固有のプロパティ・テンプレートを作成する。
- **コネクタ構成ファイル**を作成する。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する。  
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 Configurator の実行』を参照)。

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

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

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

ただし**コネクタ固有プロパティ**の場合は、最初にプロパティを定義し、その値を設定する必要があります。このため、特定のアダプターのコネクタ固有プロパティのテンプレートを作成します。システム内で既にテンプレートが作成されている場合には、作成されているテンプレートを使用します。システム内でまだテンプレートが作成されていない場合には、66 ページの『新規テンプレートの作成』のステップに従い、テンプレートを新規に作成します。

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

---

## Connector Configurator の始動

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

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

### スタンドアロン・モードでの Configurator の実行

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

これを行うには、以下のステップを実行します。

- 「スタート」>「プログラム」から、「IBM WebSphere InterChange Server」>「IBM WebSphere Business Integration 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: Integration Broker」の隣のプルダウン・メニューをクリックします。使用しているブローカーに応じて、ICS、WebSphere Message Brokers、または WAS を選択します。

Connector Configurator を個別に実行して構成ファイルを生成してから、System Manager に接続してこの構成ファイルを System Manager プロジェクトに保存することもできます (71 ページの『構成ファイルの完成』を参照)。

---

## System Manager からの Configurator の実行

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

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

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

既存の構成ファイルを編集するには、以下のステップを実行します。

- 「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、66 ページの『新規テンプレートの作成』を参照してください。
- 既存のファイルを使用する場合には、既存のテンプレートを変更し、新しい名前でのこのテンプレートを保管します。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を作成し、プロパティの一般特性および値を定義し、プロパティ間の依存関係を指定する方法について説明します。次にそのテンプレートを保管し、新規コネクタ構成ファイルを作成するためのベースとして使用します。

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・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・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定義を調べるには、「テンプレート名」表示でそのテンプレートの名前を選択します。そのテンプレートに含まれているプロパティ定義のリストが「テンプレートのプレビュー」表示に表示されます。
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定義に類似したプロパティ定義が含まれている既存のテンプレートを使用できます。ご使用のコネクタで使用するコネクタ固有のプロパティが表示されるテンプレートが見つからない場合は、自分で作成する必要があります。
  - 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: コネクタ固有プロパティ・テンプレート」ダイアログ・ボックスが表示されます。このダイアログ・ボックスには、定義済みプロパティの「一般」特性のタブと「値」の制限のタブがあります。「一般」表示には以下のフィールドがあります。

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

プロパティの一般特性の選択を終えたら、「値」タブをクリックします。

## 値の指定

「値」タブを使用すると、プロパティの最大長、最大複数値、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: コネクター固有プロパティ・テンプレート」ダイアログ・ボックスが表示されます。

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

1. 「使用可能なプロパティ」表示で、依存プロパティとして指定するプロパティを選択します。

2. 「プロパティを選択」フィールドで、ドロップダウン・メニューを使用して、条件値を持たせるプロパティを選択します。
3. 「条件演算子」フィールドで以下のいずれかを選択します。

== (等しい)

!= (等しくない)

> (より大)

< (より小)

>= (より大か等しい)

<= (より小か等しい)

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

---

## 新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。
- この章で後述する説明に従って「新規コネクタ」ウィンドウの残りのフィールドに入力します。

## 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す。

- すべてのファイル (`*.*`)

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

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

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

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



---

## 構成ファイルの完成

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

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

1. 「標準のプロパティ」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 74 ページの『サポートされるビジネス・オブジェクト定義の指定』の説明に従い残りの構成フィールドに値を入力することができます。
3. 構成が完了したら、「ファイル」>「保管」>「プロジェクトに」を選択するか、または「ファイル」>「保管」>「ファイルに」を選択します。

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

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

Connector Configurator では、ファイルを保管する前に、必須の標準プロパティ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

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

Connector Configurator では、すべてのブローカーで実行されているコネクターで、以下のカテゴリーのプロパティ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
- コネクター固有のプロパティ
- 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ーの場合に該当する)

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

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

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

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

標準プロパティとコネクタ固有プロパティの違いは、以下のとおりです。

- コネクタ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

標準のプロパティの値を変更するには、以下の手順を実行します。

1. 値を設定するフィールド内でクリックします。

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

## アプリケーション固有の構成プロパティの設定

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

1. グリッドの左上端の部分で右マウス・ボタンをクリックします。ポップアップ・メニュー・バーが表示されます。プロパティを追加するときは「追加」をクリックします。子プロパティを追加するには、親の行番号で右マウス・ボタンをクリックし、「子を追加」をクリックします。
2. プロパティまたは子プロパティの値を入力します。
3. プロパティを暗号化するには、「暗号化」ボックスを選択します。
4. 72 ページの『標準コネクタ・プロパティの設定』の説明に従い、変更内容を保管するかまたは破棄するかを選択します。

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

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

### コネクタ・プロパティの暗号化

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

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

プロパティ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 44 ページの『プロパティ値の設定と更新』にある更新メソッドの説明を参照してください。

## サポートされるビジネス・オブジェクト定義の指定

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

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

## ご使用のブローカーが ICS の場合

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

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

1. 「ビジネス・オブジェクト名」リスト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、以下の手順を実行します。

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

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

## ご使用のブローカーが WAS の場合

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

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

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

## 関連付けられているマップ (ICS のみ)

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

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

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

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

「関連付けられたマップ」タブには以下のフィールドが表示されます。

- **ビジネス・オブジェクト名**

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

- **関連付けられたマップ**

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

- **明示的**

場合によっては、関連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。

以下の手順を実行して、マップを明示的にバインドします。

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

## リソース (ICS)

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

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

## メッセージング (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、以下の手順を実行します。

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

- コンソールに (STDOUT):

ログ・メッセージまたはトレース・メッセージを STDOUT ディスプレイに書き込みます。

**注:** STDOUT オプションは、Windows プラットフォームで実行しているコネクタの「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・モードでファイルを保管します。Connector Configurator のタイトル・バーには現在のブローカー・モード (ICS、WMQI、または WAS) が常に表示されます。

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



- System Manager から、統合コンポーネント・ライブラリーに \*.con 拡張子付きファイルとして保管します。
- 指定したディレクトリーに保管します。
- スタンドアロン・モードで、ディレクトリー・フォルダーに \*.cfg 拡張子付きファイルとして保管します。デフォルトでは、このファイルは %WebSphereAdapters%bin%Data%App に保管されます。
- WebSphere Application Server プロジェクトをセットアップしている場合には、このファイルを WebSphere Application Server プロジェクトに保管することもできます。

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

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

---

## 構成ファイルの変更

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

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

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

- Connector Configurator で既存の構成ファイルを開きます。
- 「標準のプロパティー」タブを選択します。
- 「標準のプロパティー」タブの「**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、以下に太文字で示した行を追加してください。

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

---

## 特記事項

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

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

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

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、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 Burlingame Laboratory Director  
IBM Burlingame Laboratory  
577 Airport Blvd., Suite 800  
Burlingame, CA 94010  
U.S.A

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

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

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

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

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

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

#### 著作権使用許諾

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

---

## プログラミング・インターフェース情報

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

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

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

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

---

## 商標

以下は、IBM Corporation の商標です。

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

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

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

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

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。



WebSphere Business Integration Adapter Framework V2.4.0







Printed in Japan