WebSphere Adapters

バージョン 6.0.2

Adapter for JDBC ユーザーズ・ガイド

お願い

本書をご使用になる前に、 187 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、WebSphere Adapter for JDBC バージョン 6、リリース 0、モディフィケーション 2 (製品番号 5724-L77) および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

IBM 発行のマニュアルに関する情報のページ

http://www.ibm.com/jp/manuals/

こちらから、日本語版および英語版のオンライン・ライブラリーをご利用いただけます。また、マニュアルに関するご意見やご感想を、上記ページよりお送りください。今後の参考にさせていただきます。

(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： WebSphere Adapters
Adapter for JDBC User Guide
Version 6.0.2

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2006.12

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

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

© Copyright International Business Machines Corporation 2005, 2006. All rights reserved.

© Copyright IBM Japan 2006

目次

© Copyright IBM Corp. 2005, 2006

第 1 章 本書について

本書は、WebSphere® Adapter for JDBC の実装、構成、およびデプロイを行う統合開発者を対象としています。本書を使用するには、ビジネス・インテグレーションの概念について理解し、一定のテクニカル・スキルを持っている必要があります。

統合開発者は、ビジネス・インテグレーション・ソリューションの設計、アセンブル、テスト、およびデプロイを行います。本書の情報は、エンタープライズ情報システム (EIS) と Java™ 2 Platform, Enterprise Edition (J2EE) アプリケーションの間でデータを交換する必要があるソリューションに WebSphere Adapter for JDBC をデプロイする担当者を対象としています。使用にあたっては、以下の概念、標準、およびツールに関する知識と経験が必要です。

- ビジネス・ソリューションおよび環境。
- データベース、データ・アクセスの問題、トランザクション・モデル、および異種のリレーショナル・データベース、キュー、および Web サービス間の接続。
- Service Component Architecture (SCA) プログラミング・モデルおよび Service Data Object (SDO) データ・モデルを含むビジネス・インテグレーションの仕組み。
- J2EE 標準、および J2EE アプリケーション。
- 稼働環境内で使用されるホストに応じた、WebSphere Process Server または WebSphere Enterprise Service Bus の機能および要件。ホスト・サーバーの構成方法と管理方法、および管理コンソールの使用方法について理解していることが求められます。
- WebSphere Integration Developer によって提供されるツールおよび機能。これらのツールを使用してコンポーネントを結び付け、他の統合タスクを完了する方法について理解していることが求められます。

デプロイメントを完了するには、以下の作業を行う方法を知っておく必要があります。

- テストとデプロイメントの両方に必要なスクリプト、ツール、およびテンプレートを作成する
- エンタープライズ Bean、ワークフロー、Web ページなどのエンティティー間の相互依存性を解決する
- データベース・アクセス・ロジックを効果的に使用するためのプロシージャーを作成する
- 外部データ・アクセス・ツール用のデータ・モデルを作成する
- セキュリティー対策を実施する

© Copyright IBM Corp. 2005, 2006

第 2 章 新機能

WebSphere Adapter for JDBC バージョン 6.0.2 では、アダプターのバージョン 6.0 への機能拡張が提供されています。

このリリースの新機能

WebSphere Adapter for JDBC バージョン 6.0.2 では、多くの機能強化が加えられています。例えば、クエリー・ビジネス・オブジェクト、カスタム・イベント処理、およびストアード関数をサポートします。

バージョン 6.0.2 の新機能:

- SQL データ型の CLOB および BLOB をサポートします。
- ストアード関数をサポートします。アダプターは、ReturnValue アプリケーション固有情報があるかどうかをチェックし、存在する場合はストアード関数を実行します。
- カスタム・イベント処理をサポートします。カスタム照会は、標準 SQL、ストアード・プロシージャー、またはストアード関数のいずれかを使用します。
- EDT は、「送達は 1 回のみ」に使用されなくなりました。この機能のために、アクティベーション・スペック・プロパティー AssuredOnceDelivery が新たに提供されます。
- ストアード・プロシージャーをビジネス・オブジェクトに関連付けるときに、新しいフィルターを使用して、エンタープライズ・メタデータ・ディスカバリーでストアード・プロシージャーのリストを絞り込むことができます。
- アダプターは、アクティベーション・スペック・プロパティー EventFilterType を使用して、処理するイベントをビジネス・オブジェクト・タイプによってフィルタリングすることができ、また、プロパティー FilterFutureEvents を使用して、タイム・スタンプによってイベントをフィルタリングすることができます。
- エンタープライズ・サービス・ディスカバリー中に、ユーザー指定 select ステートメントからクエリー・ビジネス・オブジェクトを生成できます。
- ストアード・プロシージャーおよびストアード関数をサポートする Execute 操作を提供します。
- DataSource インスタンスを通じたデータベース接続の確立をサポートします。DataSourceJNDIName プロパティーが、管理 (J2C) 接続ファクトリーおよびアクティベーション・スペック・プロパティーに追加されました。
- Inbound 処理の高可用性をサポートします。詳しくは、『クラスター環境での WebSphere Adapters』を参照してください。

リリース情報

WebSphere Adapter for JDBC バージョン 6.0.2 のこのリリース情報では、このリリースでの新しいフィーチャーと機能を要約し、またすべての既知の予備手段も記載しています。

© Copyright IBM Corp. 2005, 2006

このアダプターのリリース情報は、Web サイト「Adapter for JDBC release notes」にあります。

第 3 章 WebSphere Adapters の概要

IBM® WebSphere Adapters は、Java 2 Platform, Enterprise Edition (J2EE) コンポーネント (新しい e-ビジネス・アプリケーションなど) がエンタープライズ情報システム (EIS) 上のリソースと通信できるようにします。EIS は、企業の情報インフラストラクチャー (エンタープライズ・リソース・プランニング [ERP] システムなど) です。

WebSphere アダプターは、J2EE コンポーネントと EIS の間を仲介する役割を果たすため、J2EE コンポーネントが EIS のローレベル API やデータ構造を知っている必要はありません。

WebSphere Adapters は、アプリケーションとテクノロジーの 2 種類のいずれかです。

- アプリケーション・アダプターは、既存のパッケージ・アプリケーション (SAP Software、 Siebel、PeopleSoft Enterprise、JD Edwards EnterpriseOne など) に接続するため、アプリケーションに固有のデータおよびサービスを利用できます。
- テクノロジー・アダプターは、リレーショナル・データベース、フラット・ファイル、E メール・メッセージ、FTP などのテクノロジーおよびプロトコルを介してデータへの接続性を提供します。

WebSphere ファミリー製品の一部として、WebSphere Adapters は WebSphere Integration Developer、および WebSphere Process Server か WebSphere Enterprise Service Bus のいずれかと連携します。

- WebSphere Integration Developer は WebSphere アダプターのツール環境です。

  WebSphere Integration Developer は、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイするモジュールの組み立てに使用します。WebSphere Integration Developer 内から、アダプター (リソース・アダプター [RAR] ファイルとしてパッケージされています) をインポートして EIS に接続します。WebSphere Integration Developer のエンタープライズ・サービス・ディスカバリー・ウィザードが、EIS 上のデータおよびサービスを検索し、そのデータおよびサービスにアクセスするために必要なインターフェース情報を作成します。最後に、WebSphere Integration Developer が、アダプターおよびインターフェース情報を含むモジュールを生成します。
- WebSphere Process Server または WebSphere Enterprise Service Bus は WebSphere アダプターのランタイム環境です。

  WebSphere Integration Developer によって生成されるモジュールをいずれかのサーバーにデプロイします。

モジュールの生成およびデプロイメントを以下の図に示します。

© Copyright IBM Corp. 2005, 2006

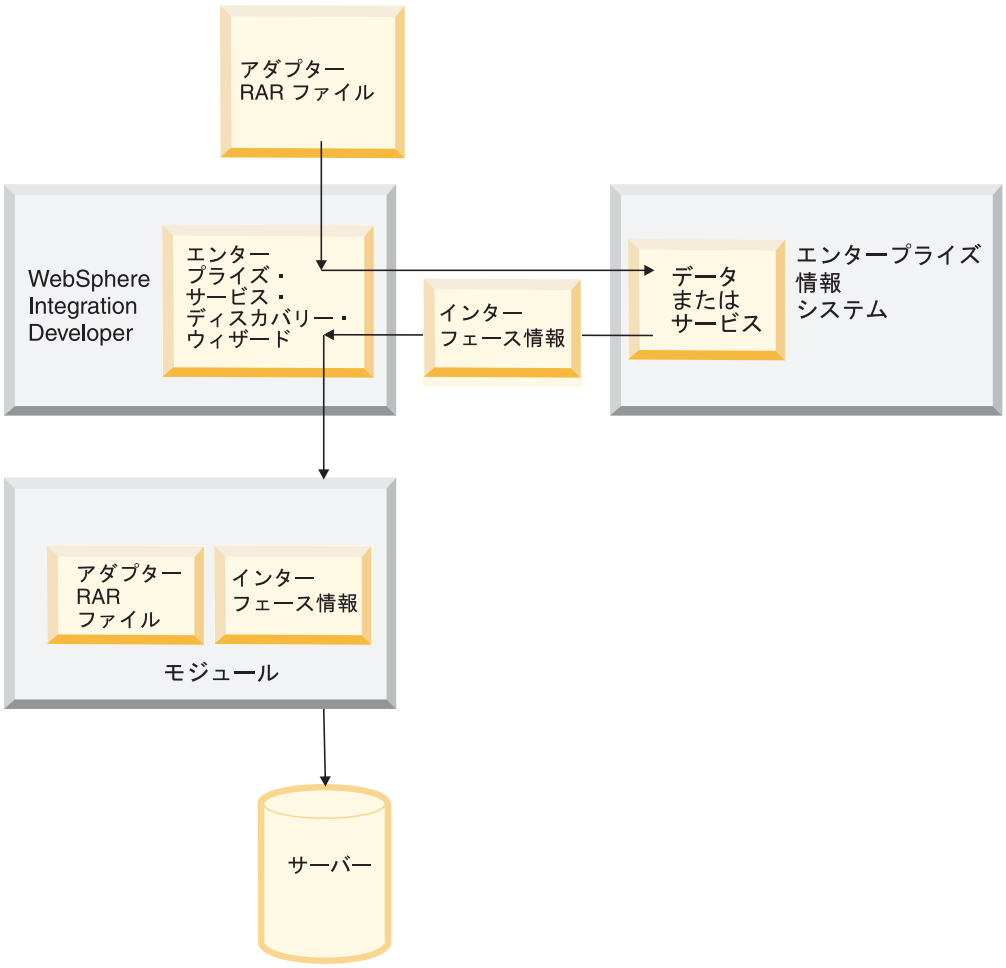


図 1. モジュールの生成とデプロイの方法

第 4 章 WebSphere Adapter for JDBC の概要

WebSphere Adapter for JDBC は、J2EE アプリケーションとエンタープライズ情報システム (EIS) の間に双方向の接続性を提供するリソース・アダプターです。このようなアプリケーションでは、ビジネス・オブジェクトという形でのデータの交換は、データベース・レベルで行われます。

Adapter for JDBC は、データベース・プロバイダーである EIS と通信します。データベースを更新した場合、別のエンタープライズ情報システムにその更新を適用する必要が生じたり、EIS 内でデータを変更した場合、その変更をデータベースに適用する必要が生じたりする場合があります。リソース・アダプターは、データベース上で作成された、JDBC 2.0 以降の仕様をサポートする JDBC ドライバーを持つアプリケーションと統合することができます。そのようなデータベースの例としては、IBM DB2®、Oracle、Microsoft® SQLServer、Sybase、Informix® などがあります。

統合をサポートするため、リソース・アダプターは EIS から受け取った要求を処理し、データベースの更新の結果として生成されたイベントを処理します。このアダプターは、これらのイベントをアプリケーション・サーバー内のさまざまな定義済みのエンドポイントに送信します。エンドポイント とは、J2EE アプリケーションまたはイベントのその他のクライアント利用者を指します。データベース内のテーブルで行われたデータ更新は、アプリケーション・サーバーに接続された、Siebel、Peoplesoft、Oracle アプリケーションなどのその他のアプリケーションに、イベント・ストアに通知されるイベント通知を通して自動的に波及させることができます。アダプターは、ビジネス・オブジェクトに指定された SQL 照会またはストアード・プロシージャーを使用してデータベース表を更新します。ビジネス・オブジェクト は、ビジネス機能またはエレメント (顧客や送り状など) を表すアプリケーション・データのコンテナーです。

ハードウェアおよびソフトウェア要件

Adapter for YOUR ADAPTER NAME をインストールする前に、ご使用の環境が必要な要件を満たしていることを確認する必要があります。これらの要件は、アダプター・インストーラーの実行用にサポートされるプラットフォーム、およびアダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件、という 2 つのカテゴリーに分類されます。

アダプター・インストーラーの実行用にサポートされているプラットフォーム

アダプター・インストーラーの実行用にサポートされているプラットフォームは、『Installing IBM WebSphere Adapters』の『Installing』セクションにあります。

© Copyright IBM Corp. 2005, 2006

アダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件

アダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件は、Web サイト『IBM WebSphere Adapters and IBM WebSphere Business Integration Adapters: software requirements』にあります。IBM WebSphere Adapters リストから「Adapter for YOUR ADAPTER NAME, Version 6.0.2」のリンクを選択します。

標準の準拠

この製品は、アクセシビリティ標準やインターネット・プロトコル標準といった、いくつかの行政標準および業界標準に準拠しています。

アクセシビリティ

IBM は、年齢や能力を問わず、すべての人が便利に使用できる製品の提供に努めています。WebSphere Adapters ソフトウェアは、完全にアクセス可能で、米国リハビリテーション法第 508 条に準拠しています。アクセシビリティ機能を使用すると、運動障害または視覚障害など身体に障害を持つユーザーがソフトウェア・プロダクトを快適に操作できるようになります。これらの機能は、WebSphere Adapters のインストールおよび管理機能に組み込まれています。

インストール

WebSphere Adapters は、グラフィカル・ユーザー・インターフェースを使用して、あるいはスクリプトを使用してサイレントに、このいずれかの方法でインストールを行うことができます。アクセシビリティを必要とするユーザーには、サイレント・インストール・メソッドをお勧めします。

管理

WebSphere Process Server または WebSphere Enterprise Service Bus の管理コンソールは、エンタープライズ・アプリケーションをデプロイし、管理するための主インターフェースです。これらのコンソールは、標準の Web ブラウザー内に表示されます。Microsoft Internet Explorer や Netscape Browser などのアクセス可能な Web ブラウザーを使用すると、次のことが可能になります。

- スクリーン・リーダー・ソフトウェアとデジタル・スピーチ・シンセサイザーを使用して、画面上に表示されている内容を聞く
- IBM ViaVoice® などの音声認識ソフトウェアを使用したデータの入力とユーザー・インターフェースへのナビゲート
- マウスの代わりにキーボードを使用して機能を操作する

提供されているグラフィカル・インターフェースの代わりに標準のテキスト・エディターやスクリプト・インターフェースまたはコマンド行インターフェースを使用すると、製品機能の構成や使用が可能になります。

場合によっては、特定の製品機能についての文書に、その機能のアクセシビリティについての追加情報が記載されています。

エンタープライズ・サービス・ディスカバリー・ウィザード

エンタープライズ・サービス・ディスカバリー・ウィザードは、アダプターを使用してエンタープライズ・アプリケーションを作成するのに使用する主コンポーネントです。このウィザードは、WebSphere Integration Developer を通じて使用可能な Eclipse プラグインとして実装されており、完全にアクセス可能です。

キーボード・ナビゲーション

この製品では、標準の Microsoft Windows® ナビゲーション・キーを使用します。

IBM とアクセシビリティ

IBM のアクセシビリティに対する取り組みについては、*IBM Accessibility Center* を参照してください。

インターネット・プロトコル・バージョン 6.0

IBM WebSphere Process Server は、インターネット・プロトコル・バージョン 6.0 との互換性を保つために WebSphere Application Server を使用しています。

IBM WebSphere Application Server Version 6.0 およびその JavaMail コンポーネントでは、デュアル・スタックのインターネット・プロトコル・バージョン 6.0 (IPv6) がサポートされています。

WebSphere Application Server でのこの互換性について詳しくは、WebSphere Application Server インフォメーション・センターの IPv6 サポートを参照してください。

IPv6 について詳しくは、 www.ipv6.org を参照してください。

Adapter for JDBC の技術的な概説

Adapter for JDBC は、Java 2 Platform, Enterprise Edition (J2EE) Connector Architecture (JCA) の下で Inbound および Outbound 処理を提供することによって、JDBC アプリケーション・プログラミング・インターフェース (API) を介してアクセス可能なデータベースの統合をサポートします。

Outbound 操作では、ビジネス・オブジェクトは、ビジネス・オブジェクトに指定された Create、Retrieve、Update、Delete、Retrieveall、Execute のいずれかの操作に従って処理される要求としてアダプターに渡されます。要求は、このアダプターの管理対象のデータベースに対して更新を適用させる必要のある別の EIS アプリケーションから受け取ります。これらの要求を処理すると、対応するデータベース表で行の作成、検索、更新または削除が行われます。

Inbound 操作では、イベント・ストアを使用する標準イベント処理、またはカスタム・イベント処理のいずれかを使用できます。

標準のイベント処理 中に、データがデータベース内のアプリケーション・テーブルで変更されると、対応するイベントがキー値などの関連情報と共にイベント・ストアと呼ばれるイベント・テーブルに挿入されます。変更されたデータを取り込むために、それぞれのテーブルにトリガーを設定でき、また、Oracle データベースに対

して提供される Oracle Change Data Capture のような他のメソッドを使用することもできます。Adapter for JDBC は、イベント・ストアをポーリングし、一まとまりのイベントを検索します。各イベントは、ビジネス・オブジェクト・タイプまたはタイム・スタンプ、あるいはその両方に基づいてフィルターに掛けることができます。これらのイベントが処理されます。それぞれのイベントは JDBC ビジネス・グラフの構成に使用されます。この後、ビジネス・グラフは、特定のビジネス・オブジェクトへのサブスクリプションを持つエンドポイントにディスパッチされます。

カスタム・イベント処理 中に、標準の SQL ステートメント、ストアード・プロシージャー、またはストアード関数として照会を入力できます。これらの操作のすべてで、以下のデータベース列のデータをこの順で持つ結果セットが返されます。

- event_id
- object_key
- object_name
- object_function

アダプターは、それぞれのイベントの JDBC ビジネス・グラフを構成して、これを特定のビジネス・オブジェクトへのサブスクリプションを持つエンドポイントに送信します。アダプターは、カスタム・イベント処理用のカスタム更新/削除照会もサポートします。カスタム更新/削除照会は、各イベントが処理された後に実行されます。これらの照会では、入力パラメーターとして event_id を取ります。更新照会では、同じレコードが以降のポーリング周期中に処理対象として取り出されないようにします。

『Adapter for JDBC 内部の処理』の図に、Inbound 操作および Outbound 操作を示します。

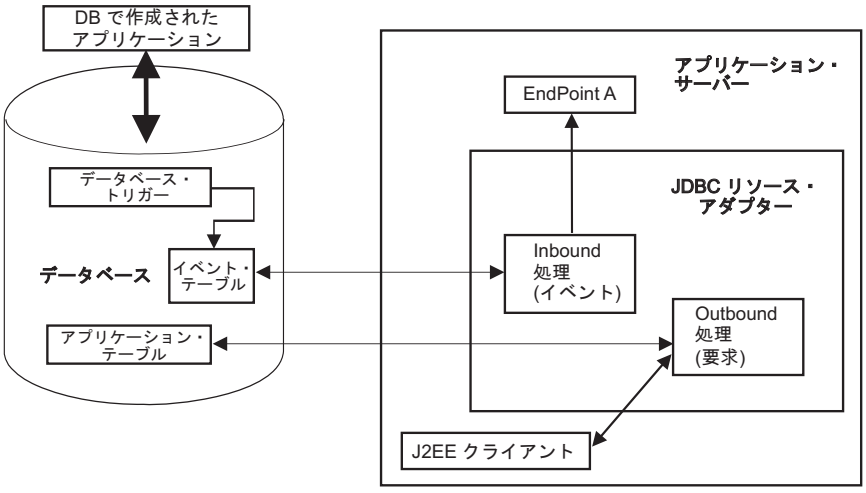


図 2. Adapter for JDBC 内部の処理

Outbound 処理

アプリケーション・コンポーネントがエンタープライズ情報システム (EIS) で操作を呼び出す (データの取得など) 必要がある場合、Adapter for JDBC は、アプリケ

ーション・コンポーネントと EIS の間のコネクターとして機能します。アダプターは特定の Outbound 操作を提供します。標準の Outbound 操作は、変更後イメージまたは差分スタイル・ビジネス・オブジェクトを処理できます。また、アダプターは、Outbound 操作のためにローカル・トランザクションと XA トランザクションの両方をサポートします。

Outbound 操作

アダプターは、ビジネス・オブジェクトによって伝えられる情報の量および目的に関連する、変更後イメージと差分という 2 つのビジネス・オブジェクト・スタイルをサポートします。*変更後イメージ* とは、ビジネス・オブジェクトに対するすべての変更が行われた後の、ビジネス・オブジェクトの状態のことです。*差分* とは、Update 操作で使用される、キー値および変更対象のデータのみを含むビジネス・オブジェクトのことです。

アダプターは、以下の操作に対して変更後イメージ・サポートを提供します。

- Create
- Update
- Delete

アダプターは、以下の操作に対して差分サポートを提供します。

- ApplyChanges

変更後イメージまたは差分サポートのない Outbound 操作:

- Retrieve
- RetrieveAll
- Execute (ストアード・プロシージャーでのみ使用可能)

動詞は、変更後イメージ・ビジネス・オブジェクトに対してのみ定義できます。*動詞* にはビジネス・オブジェクトの状態が反映され、*操作* にはアダプターによって実行される操作が反映されます。変更後イメージに対してサポートされる最上位の動詞は、次のとおりです。

- Create
- Update
- Delete
- UpdateWithDelete

サポートされる各 Outbound 操作でアダプターがビジネス・オブジェクトを処理する方法について詳しくは、『Outbound 操作』セクションを参照してください。

トランザクション管理

Adapter for JDBC は、ローカル・トランザクションと XA トランザクションの両方をサポートします。このアダプターでは、トランザクションとは、バックエンド・データベースまたはエンタープライズ情報システム (EIS) との独立した相互作用です。トランザクションは、アトミックな単位で実行する、データベースへの複数の操作から構成されます。これらの操作は、データベースの他のクライアント・アプリケーションから同時に実行される操作の影響は受けません。

Adapter for JDBC がトランザクションをサポートするのは、バックエンド・データベースがトランザクションをサポートする場合のみです。サポートされるトランザクションのタイプは、ローカル・トランザクションと XA トランザクションです。

- *ローカル・トランザクション* では、指定されたクライアント・アプリケーションが、データベースを使用したトランザクションの開始および終了を定義します。このトランザクションでは、1 フェーズ・コミット・プロトコルを使用します。
- *XA トランザクション* では、トランザクションは複数の異種データベースを使用します。このトランザクションでは、グローバル・プロトコルまたは 2 フェーズ・コミット・プロトコルを使用します。

IBM DB2 および Oracle データベースの XA トランザクション

このアダプターは、IBM DB2 および Oracle データベースの XA トランザクションのみサポートします。

XA トランザクションでは、XADataSourceName および XADatabaseName プロパティーを使用してください。これらのプロパティーについて詳しくは、『参照』セクションの『管理 (J2C) 接続ファクトリー・プロパティー』を参照してください。

XA トランザクション・プロパティーのサンプル値を以下に示します。

IBM DB2

XADataSourceName for Type 2 Driver (db2java.zip):
`COM.ibm.db2.jdbc.DB2XADataSource`

XADataSourceName for Type 4 Driver (db2jcc.jar):
`com.ibm.db2.jcc.DB2XADataSource`

XADatabaseName : *<dbname>*
ここで、*dbname* はデータベースの名前です。

Oracle

XADataSourceName: `oracle.jdbc.xa.client.OracleXADataSource`

注: DB2 データベースを使用している場合、XADatabaseName プロパティーを構成する必要があります。Oracle データベースを使用する場合、XADatabaseName プロパティーには値は必要ありません。

DataSourceJNDIName を使用したトランザクション・サポート

DataSourceJNDIName プロパティーが、ローカルまたは XA トランザクション・データ・ソースとして定義されている場合、アダプターは、データベースへのインバウンドまたはアウトバウンド接続を獲得するためにこのプロパティーを使用します。DataSourceJNDIName は、WebSphere Process Server または WebSphere Enterprise Service Bus 内部で作成された DataSource を表します。この名前は、XA または接続プール・データ・ソースを表す可能性もあります。

UserName プロパティーおよび Password プロパティーが指定されている場合、これらは DataSourceJNDIName と共に接続の獲得に使用されます。UserName および Password が指定されない場合、アダプターは、サーバー内部で DataSource が作成

されたときに設定されたユーザー名およびパスワードを使用します。
JDBCDriverClass、DatabaseURL、および XA プロパティー (XA DataSourceName および XADatabaseName) などの、その他のすべてのプロパティーは無視されます。

重要: DataSourceJNDIName プロパティーを通じて XA データ・ソースを使用する場合、XA トランザクション・サポートは、IBM DB2 データベースおよび Oracle データベースに制限されません。データベースの制限は、データベースへのデータ・ソース接続以外の接続を使用する XA トランザクション・サポートに適用されます。

Inbound 処理

Adapter for JDBC は、非同期イベント送達を行う Inbound イベント管理をサポートします。イベントは、標準イベント・テーブルとカスタム照会の 2 つのモードを使用して処理されます。

非同期イベント送達は、以下のものを使用して実行されます。

- 標準イベント・テーブル (イベント・ストアと呼びます)
- カスタム・イベント処理

標準イベント・テーブル

ユーザー・テーブルで変更を行うと、アプリケーションはエンタープライズ情報システム (EIS) 内のイベント・テーブル (イベント・ストアと呼びます) にデータを取り込みます。ユーザー・テーブルにトリガーを配置することによって更新を行います。ユーザー・テーブルは、ユーザー・テーブルへの更新に対応してイベント・テーブルにイベントを記録します。

アクティベーション・スペック・プロパティー AssuredOnceDelivery を true に設定すると、イベント・ストアの各イベントに xid (トランザクション ID) 値が設定されます。イベントが処理対象として取得されると、イベント・テーブル内でそのイベントの xid 値が更新されます。さらに、イベントが対応するエンドポイントに送達され、その後イベント・テーブルから削除されます。イベントがエンドポイントに送達される前に、データベース接続が失われたか、アプリケーションが停止した場合、イベントは完全に処理されない可能性があります。この場合、xid 列によって、イベントが再処理され、エンドポイントに送信されることが保証されます。データベース接続が再確立されるか、アダプターが再び始動されると、アダプターは、イベント・テーブルで xid 列に値を持つイベントがあるかどうかをチェックします。アダプターは、まずこれらのイベントを処理してから、ポーリング周期の間にその他のイベントをポーリングします。イベント・テーブルは次のように記述されます。

アダプターは、処理するイベントをビジネス・オブジェクト・タイプでフィルターに掛けることができます。フィルターは、アクティベーション・スペック・プロパティー EventFilterType を使用して設定します。このプロパティーは、ビジネス・オブジェクト・タイプをコンマで区切ったリストを持ちます。プロパティーで指定されたタイプのみが処理されます。プロパティーに値が指定されていない場合、フィルターは適用されず、すべてのイベントが処理されます。アクティベーション・スペック・プロパティー FilterFutureEvents が true に設定されている場合、アダプターは、タイム・スタンプに基づいてイベントをフィルターに掛けます。アダプター

は、各ポーリング周期のシステム時刻を各イベントのタイム・スタンプと比較します。イベントが将来発生するように設定されている場合は、その時刻になるまで処理されません。

カスタム・イベント処理

カスタム照会は、以下を使用して入力できます。

- 標準の SQL
- ストアード・プロシージャー
- ストアード関数

これらの項目はすべて、入力パラメーターとして PollQuantity (アダプターがランタイムに提供するアクティベーション・スペック・プロパティー) を取ります。これらの照会は結果セットを返します。この結果セットはポーリング数量に相当する数のレコードを含み、以下の列を順に持ちます。event_id、object_key、object_name、および object_function。これらの列は、以下の情報を表します。

- **event_id**: 各イベントの固有 ID。object_key の値と同一でもかまいません。
- **object_key**: イベント処理のために取り出される、テーブル内のレコードのキー値。
- **object_name**: エンタープライズ・サービス・ディスカバリーを使用して生成されたビジネス・グラフの名前。ビジネス・グラフ内のビジネス・オブジェクトは、階層ビジネス・オブジェクトであってもかまいません。各ビジネス・オブジェクトは、テーブルまたはビューを参照します。
- **object_function**: イベントに設定される Create、Update、または Delete 操作。

3 種類のカスタム照会について、以下に説明します。

標準の SQL

ユーザーは、1 つのポーリング数量の入力パラメーターを取る SQL select ステートメントを入力することができます。照会は、その他の入力パラメーターを持つことができますが、それらすべてのパラメーターの値が、SQL 照会自体で設定されている必要があります。SQL 照会は、ポーリング数量に相当する数のレコードを含む結果セットを返します。この結果セットは、event_id、object_key、object_name、および object_function の列を、この順序で含んでいなければなりません。 Adapter for JDBC は、イベント・オブジェクトを生成してイベントを処理します。

ストアード・プロシージャー

カスタム照会は、結果セットを含むストアード・プロシージャーである場合があります。ストアード・プロシージャーが呼び出されたときに、アダプターはポーリング数量の値を渡します。そのため、ユーザーが入力したプロシージャーは、ポーリング数量の入力パラメーターを受け入れる必要があります。また、ストアード・プロシージャーには、結果セット型の出力パラメーターがあります。ストアード・プロシージャーによって戻される結果セットは、event_id、object_key、object_name、および object_function の列を、この順序で含んでいなければなりません。ストアード・プロシージャーを指定する際に使用される構文を、以下に示します。

```
call <sp_name> (?, ?)
```

ここで sp_name は、実行する必要のあるストアード・プロシージャーの名前です。

注: 最初のパラメーターはポーリング数量を表し、2 番目のパラメーターは結果セットを表します。

ストアード・プロシージャーは、その他の入力パラメーターを取ることもできますが、call ステートメント自体でそれらの入力パラメーターに値を提供する必要があります。例えば、`call <sp_name> (25, ?, ?)` のようにします。

ストアード関数

カスタム照会は、結果セットを戻すストアード関数である場合もあります。ストアード関数が呼び出されたときに、アダプターはポーリング数量の値を渡します。そのため、ユーザーが入力した関数は、ポーリング数量の入力パラメーターを受け入れる必要があります。また、関数の戻り値は、結果セットでなければなりません。関数によって戻される結果セットは、event_id、object_key、object_name、および object_function の列を、この順序で含んでいなければなりません。ストアード関数を指定する際に使用される構文を、以下に示します。

? = <*sp_name*> (?)

ここで sp_name は、実行する必要のあるストアード関数の名前です。

注: 最初のパラメーターは結果セットを表し、2 番目のパラメーターはポーリング数量を表します。

ストアード関数は他の入力パラメーターを取ることができますが、call ステートメント自体でそれらの入力パラメーターに値を提供する必要があります。例えば、 ? = call <*sp_name*> (25, ?) のようにします。

アダプターは、カスタム更新照会および削除照会のサポートを提供します。ユーザーは通常、*更新照会* を使用して、同じレコードが以降のポーリング周期中に処理対象として取り出されないようにします。*削除照会* は、各イベントの処理後にレコードを削除する必要がある場合に使用します。更新照会と削除照会は、どちらもオプションです。

これらの照会は、アクティベーション・スペックで指定された場合、各イベントの処理後に実行されます。ユーザーは、これらの照会を 2 とおりの方法で入力することができます。標準 SQL ステートメントとして、またはストアード・プロシージャーとして入力します。ストアード・プロシージャー構文は、カスタム照会アクティベーション・スペック・プロパティーに指定された構文と同じです。標準 SQL とストアード・プロシージャーはどちらも、イベント ID の入力パラメーターを取ります。アダプターは、実行時にイベント ID の値を提供します。照会には、追加の入力パラメーターを含めることができますが、それらは、カスタム・イベント照会について説明したように、照会構文自体で提供される必要があります。

xid 値を格納するための標準イベント・ストアを作成した場合、カスタム照会は「送達は 1 回のみ」をサポートします。アダプターは、カスタム・イベント照会によって返されたイベントをイベント・ストアに格納し、そのイベントを xid 値で更新します。アダプターは、標準のイベント・テーブル処理と同じ方法でイベントを処理します。標準イベント・ストアは、AssuredOnceDelivery プロパティーを true に設

定したカスタム・イベント処理に使用されるため、カスタム照会は、標準イベント・テーブルで照会を行うことはできません。さらにこのシナリオでは、アダプターが、カスタム照会から取得したイベント ID 値をイベント・テーブルに取り込むため、イベント・テーブルではイベント ID 値の自動生成が行われません。カスタム照会では、イベントのフィルター処理はサポートされません。

イベント・テーブル

イベント・テーブルは以下の表に記述されています。

表 1. イベント・テーブル

名前	型	説明
xid	String	「送達は 1 回のみ」の場合の固有のトランザクション ID (xid) 値
event_id	Number	テーブル用の基本キーである固有のイベント ID
object_key	String	ビジネス・オブジェクト用のキーを含む、区切り文字で区切られたストリング (NULL 以外)
object_name	String	ビジネス・オブジェクトの名前 (NULL 以外)
object_function	String	イベントに対応した操作 (Delete、Create、Update など) (NULL 以外)
event_priority	Number	任意の正整数値 (NULL 以外)
event_time	Timestamp	イベントが生成された日時
event_status	Number	値は次のとおりです。 • 0 ポーリング開始可能 • 3 処理中 • -1 イベント処理が失敗 (NULL 以外)
event_comment	String	説明

ビジネス・オブジェクト

WebSphere Process Server または WebSphere Enterprise Service Bus の統合アプリケーションで交換されるビジネス・データは、ビジネス・オブジェクトと呼ばれる構造によって表されます。ビジネス・オブジェクトは、ビジネス・データを表すための基本的なデータ構造です。

ビジネス・オブジェクトは、オープン・スタンダードである Service Data Objects (SDO) プログラミング・モデル (共通データ・アクセスをサポートする) に基づいています。WebSphere アダプターのビジネス・オブジェクトは、SDO モデルに加えて拡張機能および機能性を提供します。Service Component Architecture (SCA) では、ビジネス・オブジェクトは統合アプリケーションのサービス・コンポーネント間を流れるデータを表します。

ビジネス・オブジェクトの命名規則

ビジネス・オブジェクト名には、ビジネス・オブジェクトが表す構造が反映されます。Customer または Address などです。通常、名前はエンタープライズ・メタデータ・ディスカバリーのメタデータ・インポート処理中に、エンタープライズ情報システム (EIS) によって指定された名前に基づいて導出されます。

エンタープライズ・メタデータ・ディスカバリーは、下線 (_) 文字を持つ名前を除き、ビジネス・オブジェクト名の特殊文字をすべて U の後に Unicode 番号を続けたもので置換します。例えば、データベース・アプリケーションの Order_Item テーブルに対するビジネス・オブジェクト名は Order_Item になり、Shipping-Address テーブルに対するビジネス・オブジェクト名は ShippingU45Address になります。

親ビジネス・オブジェクトのグラフは、そのオブジェクトに含まれるビジネス・オブジェクトの名前の後に BG を付けたものとなります。例えば、CustomerBG は Customer ビジネス・オブジェクトの親ビジネス・オブジェクトのグラフです。

ビジネス・オブジェクト名には、アダプターまたはデータベースを意味する値は含まれません。つまり、ビジネス・オブジェクト名から情報や意味が派生することはありません。名前が別の名前で置換された場合でも、アダプターの動作は同じです。

ビジネス・オブジェクトはデータベース固有のメタデータを扱います。プレフィックスに JDBC や *%AppName%* のようなストリングを使用すると、アプリケーション固有と汎用の 2 つのタイプのビジネス・オブジェクトを区別するのに役立ちます。名前の残りの部分で、ビジネス・オブジェクトが表すテーブルまたはストアード・プロシージャーを説明することができます。例えば、Human Resources (HR) のようなデータベース・アプリケーションの Employee テーブル用のビジネス・オブジェクト定義を生成する場合、相当するビジネス・オブジェクト名は HREmployee です。

メタデータ・インポート処理で導出されないビジネス・オブジェクト名の例としては、エンタープライズ・サービス・ディスカバリー・ウィザードを使用して指定したクエリー・ビジネス・オブジェクトの名前が挙げられます。異なるクエリー・ビジネス・オブジェクトに重複する名前を指定しないでください。名前が重複した場合、名前は上書きされます。

WebSphere Integration Developer のリファクタリング機能を使用して、ビジネス・オブジェクトの名前を変更することができます。詳細については、WebSphere Integration Developer の資料を参照してください。

ビジネス・オブジェクトの構造

Adapter for JDBC は、テーブルおよびビュー・ビジネス・オブジェクトをサポートします。データベースを使用するアプリケーションの場合、それぞれのビジネス・オブジェクトはデータベース表やビューに対応し、オブジェクト内の単純属性がそれぞれテーブルやビュー内の列に相当します。さらに、アダプターは、ストアード・プロシージャーおよびクエリー・ビジネス・オブジェクトもサポートします。

テーブルまたはビュー・ビジネス・オブジェクトの場合

単純属性 とは、String、Integer、または Date などの単一値を表す属性です。したがって、同じビジネス・オブジェクトに含まれる属性を、別々のデータベース表に格納することはできませんが、次の状況が考えられます。

- データベース表に、対応するビジネス・オブジェクトに含まれる単純属性の数よりも多くの列が含まれる場合があります。つまり、データベースの列の一部が、ビジネス・オブジェクト内に表されていません。ビジネス・オブジェクトの処理にとって必要な列のみを実際の設計に含めるようにします。

- ビジネス・オブジェクトに、対応するデータベース表に含まれる列の数よりも多くの単純属性が含まれる場合があります。つまり、ビジネス・オブジェクト内の属性の一部が、データベース表内に表されていません。データベースの列を表していない属性は、アプリケーション固有情報を含まずにデフォルト値が設定されているか、またはストアード・プロシージャーによって指定されています。
- ビジネス・オブジェクトは、複数のデータベース表にまたがるビューを表すことができます。アダプターでは、アプリケーション内で起動された Create、Update、Delete 操作などの各要求を処理するときに、このようなビジネス・オブジェクトを使用できます。ただし、ビジネス・オブジェクトの要求を処理する場合には、Retrieve および RetrieveAll 要求に対してのみ、このようなビジネス・オブジェクトを使用できます。

ストアード・プロシージャー・ビジネス・オブジェクトの場合

この場合、ビジネス・オブジェクトがストアード・プロシージャー用に生成されます。ストアード・プロシージャーのすべての入力および出力パラメーターには、ビジネス・オブジェクト内に対応する属性があります。入力または出力パラメーターのいずれかが、配列や構造体などの複合型である場合、対応するビジネス・オブジェクト属性は、配列または構造体の属性を含む子ビジネス・オブジェクトを持つ子ビジネス・オブジェクト型です。ストアード・プロシージャーが結果セットを戻した場合、戻された結果セットの属性を格納する子ビジネス・オブジェクトが作成されます。

ストアード・プロシージャー・ビジネス・オブジェクトの構造を示す 2 つの例を、以下に示します。ビジネス・オブジェクト ScottStrtValues および ScottStrtvaluesStrt が、1 つの入力タイプと 2 つの出力タイプを持つストアード・プロシージャーから生成されます。 出力パラメーターの 1 つは、構造体データ型です。エンタープライズ・メタデータ・ディスカバリーによって、構造体型のビジネス・オブジェクト ScottStrtValuesStrt が生成され、子オブジェクトとして親ビジネス・オブジェクト ScottStrtValues に追加されます。親ビジネス・オブジェクト内の構造体型の属性については、ChildBOType アプリケーション固有情報が Struct に設定され、型が構造体であることを示します。ChildBOTypeName アプリケーション固有情報は、データベース内のユーザー定義構造体型の値に設定されます。

ScottStrtValues ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvalues" xmlns:scottstrtvalues=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvalues" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
metadata" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt">
<import namespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/scottstrtvaluesstrt"
schemaLocation="ScottStrtvaluesStrt.xsd"/>
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
```

```
</annotation>
<complexType name="ScottStrtvalues">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
<jdbcasi:MaxNumOfRetRS>0</jdbcasi:MaxNumOfRetRS>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>IP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="strt" type="scottstrtvaluesstrt:ScottStrtvaluesStrt"
minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType>OP</jdbcasi:SPParameterType>
<jdbcasi:ChildBOType>STRUCT</jdbcasi:ChildBOType>
<jdbcasi:ChildBOTypeName>STRUCT1</jdbcasi:ChildBOTypeName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

ScottStrtValuesStrt ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:scottstrtvaluesstrt=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/
scottstrtvaluesstrt" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
```

```
</annotation>
<complexType name="ScottStrtvaluesStrt">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPName>SCOTT.STRTVALUES</jdbcasi:SPName>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="name" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="title" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="dept_num" type="int" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SPParameterType></jdbcasi:SPParameterType>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

クエリー・ビジネス・オブジェクトの場合

クエリー・ビジネス・オブジェクトの構造を示す select ステートメントの例を示します。以下の select ステートメントの場合を考えます。

```
SELECT C.custid, C.custname, A.phone FROM customer C, address A WHERE
(C.custid = A.custid) AND (C.custname LIKE ?)
```

クエリー・ビジネス・オブジェクトのフォーマットは以下のようになります。

ビジネス・オブジェクト・レベルのアプリケーション固有情報は、次のとおりです。

```
SelectStatement = SELECT C.custid, C.custname, A.phone FROM customer C,
address A WHERE (C.custid = A.custid) AND (C.custname LIKE ?)
```

属性は次のとおりです。

- custid ColumnName=CUSTID
- custname ColumnName=CUSTNAME
- phone ColumnName=PHONE
- parameter1 (1 つの ? に対応、? と同じだけの数のパラメーターが必要)
- jdbcwhereclause

サポートされる操作は RetrieveAll です。

以下の例は、クエリー・ビジネス・オブジェクト定義ファイルを示しています。

クエリー・ビジネス・オブジェクトの例

```
<?xml version="1.0" encoding="UTF-8" ?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/querybo1" xmlns:querybo1=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/querybo1" xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<import namespace="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"
schemaLocation="JDBCASI.xsd"/>

<annotation>
<appinfo source="commonj.connector.asi">
<asi:annotationSet xmlns:asi="commonj.connector.asi" asiNSURI=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata"/>
</appinfo>
</annotation>
<complexType name="QueryBO1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCBusinessObjectTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:SelectStatement>select * from customer where
pkey=?</jdbcasi:SelectStatement>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
</appinfo>
</annotation>
<sequence minOccurs="1" maxOccurs="1">
<element name="pkey" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>PKEY</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="fname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>FNAME</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lname" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>LNAME</jdbcasi:ColumnName>
```

```
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="ccode" type="string" minOccurs="0" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>CCODE</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="parameter1" type="string" minOccurs="1" maxOccurs="1">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:ColumnName>Parameter1</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="jdbcwhereclause" type="string" minOccurs="0" maxOccurs="1"
default="where pkey=?">
<annotation>
<appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
<jdbcasi:PrimaryKey>false</jdbcasi:PrimaryKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
</sequence>
</complexType>
</schema>
```

すべてのビジネス・オブジェクトの場合

ビジネス・オブジェクトは、フラットまたは階層です。フラットのビジネス・オブジェクトの属性は、すべて単純属性であり、データベース表の 1 行を表します。*階層* ビジネス・オブジェクトという用語は、あらゆるレベルの子ビジネス・オブジェクトをすべて含む、完全なビジネス・オブジェクトを指します。*個別* ビジネス・オブジェクトという用語は、そのビジネス・オブジェクトが子オブジェクトを含んでいるか、そのビジネス・オブジェクトが子として属しているかに関係なく、1 つのビジネス・オブジェクトを指します。個別ビジネス・オブジェクトは、複数のデータベース表にまたがるビューを表すことができます。*最上位* ビジネス・オブジェクトという用語は、階層の頂点にあり、それ自体は親ビジネス・オブジェクトを持たない個別ビジネス・オブジェクトを指します。

階層ビジネス・オブジェクトは、子ビジネス・オブジェクト、子ビジネス・オブジェクトの配列、またはその組み合わせを表す属性を持ちます。そして、子ビジネス・オブジェクトも、それぞれ自身の子ビジネス・オブジェクトまたはビジネス・オブジェクトの配列を持つことができます。この関係は階層の下に向かって続きます。

単一カーディナリティー関係 は、親ビジネス・オブジェクト内の属性が 1 つの子ビジネス・オブジェクトを表すときに発生します。この場合、その属性は、その子ビジネス・オブジェクトと同じタイプです。

複数カーディナリティー関係 は、親ビジネス・オブジェクト内の属性が子ビジネス・オブジェクトの配列を表すときに発生します。この場合、その属性は、その子ビジネス・オブジェクトと同じタイプです。

アダプターでは、ビジネス・オブジェクト間での以下の関係がサポートされます。

- 単一カーディナリティー関係
- 単一カーディナリティー関係および所有権のないデータ
- 複数カーディナリティー関係

どちらのカーディナリティーのタイプでも、親ビジネス・オブジェクトと子ビジネス・オブジェクトの間の関係は、その関係を保管するビジネス・オブジェクトのキー属性に含まれるアプリケーション固有情報によって記述されます。このアプリケーション固有情報の詳細については、『属性に関するアプリケーション固有情報』の表にある 『ForeignKey』を参照してください。

ビジネス・オブジェクトの単一カーディナリティー関係:

単一カーディナリティー関係では、親ビジネス・オブジェクト内の属性が 1 つの子ビジネス・オブジェクトを表します。この場合、その属性は、その子ビジネス・オブジェクトと同じタイプです。アダプターは、単一カーディナリティー関係と、所有権のない単一カーディナリティー関係およびデータをサポートします。

単一カーディナリティー関係

通常、単一カーディナリティーの子ビジネス・オブジェクトを含むビジネス・オブジェクトには、関係を表すための属性が 2 つ以上含まれます。1 つの属性のタイプは、子ビジネス・オブジェクトのタイプと同じになります。もう 1 つの属性は、子の基本キーを、外部キーとして親に格納するための単純属性です。親には、子に含まれる基本キー属性と同数の外部キー属性が含まれます。

関係を設定する外部キーが親に保管されるため、各親には特定のタイプの子ビジネス・オブジェクトを 1 つだけ格納できます。

『典型的な単一カーディナリティー関係』の図に、一般的な単一カーディナリティー関係を示します。この例では、ParentBOName ボックス内の FKey は、子の基本キーを含む単純属性であり、同様に、ParentBOName ボックス内にある Child(1) は、子ビジネス・オブジェクトを表す属性です。

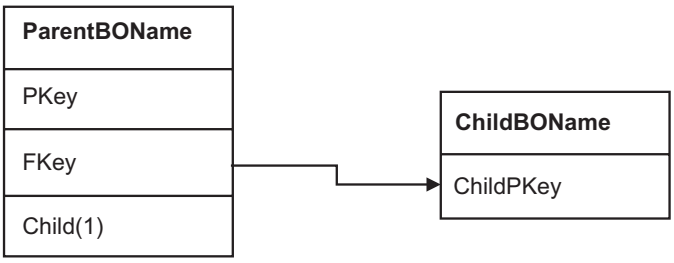


図 3. 典型的な単一カーディナリティー関係

親ビジネス・オブジェクトは、単一カーディナリティーの OWNERSHIP の子と、単一カーディナリティーの NOOWNERSHIP の子を持つことができます。参照テーブルは NOOWNERSHIP 関係に使用します。

単一カーディナリティー関係および所有権のないデータ

通常、各親ビジネス・オブジェクトは、その親ビジネス・オブジェクトに含まれる子ビジネス・オブジェクトの内部のデータを所有しています。例えば、各 Customer ビジネス・オブジェクトが Address ビジネス・オブジェクトを 1 つ含んでいる場合に、新しいカスタマーが作成されると、Customer テーブルと Address テーブルの両方に新しい行が 1 行挿入されます。挿入された新しい住所は、その新しいカスタマーに固有です。同様に、Customer テーブルからカスタマーを削除すると、Address テーブルからそのカスタマーの住所も削除されます。

ただし、複数の階層ビジネス・オブジェクトに同一のデータが含まれており、どのビジネス・オブジェクトもそのデータを所有していない場合があります。例えば、Address ビジネス・オブジェクトに *StateProvince[1]* 属性があり、これが単一カーディナリティーの *StateProvince 参照テーブル*を表しているとします。この参照テーブルは、ほとんど更新されることがないものであり、住所データからは独立して保守されています。このため、住所データの作成または変更により、この参照テーブル内のデータが影響を受けることはありません。アダプターは、既存の都道府県名を検出するか、検出に失敗するかのいずれかです。この参照テーブル内の値を追加または変更することはありません。

複数のビジネス・オブジェクトに同一の単一カーディナリティーの子ビジネス・オブジェクトが含まれている場合、各親ビジネス・オブジェクトの外部キー属性では、関係が `NOOWNERSHIP` に指定されていなければなりません。アプリケーション・サーバーからアダプターに、階層ビジネス・オブジェクトが Create、Delete、または Update 要求とともに送信された場合、アダプターは所有関係にない単一カーディナリティーの子を無視します。アダプターは、このようなビジネス・オブジェクトに対しては、Retrieve 操作のみを実行します。このような単一カーディナリティーのビジネス・オブジェクトの検索に失敗した場合、アダプターはエラーを戻して処理を停止します。

非正規化データおよび所有権のないデータ

所有関係を伴わない包含関係には、静的参照テーブルの使用を容易にするだけでなく、正規化データと非正規化データを同期化するという別の機能があります。

正規化データから非正規化データへの同期: 関係を NOOWNERSHIP に指定すると、正規化アプリケーションから非正規化アプリケーションへの同期化を行うときに、データを作成または変更することができます。例えば、正規化されたソース・アプリケーションが、A と B という 2 つのテーブルにデータを格納するものとします。また、非正規化されている宛先アプリケーションでは、1 つのテーブルにすべてのデータが格納され、エンティティー A のそれぞれにエンティティー B のデータが重複して格納されるものとします。

この例では、テーブル B のデータの変更をソース・アプリケーションから宛先アプリケーションに同期化するには、テーブル B のデータが変更されるたびにテーブル A のイベントを起動する必要があります。さらに、テーブル B のデータはテーブル A に重複して格納されているので、テーブル A の行ごとに、テーブル B で変更されたデータが含まれるビジネス・オブジェクトを送信しなければなりません。

注: 非正規化されたテーブルを更新する場合、1 行を更新した結果複数の行が変更されることがないように、レコードごとに固有キーを持たせるようにしてください。そのようなキーが存在しない場合、 Adapter for JDBC では複数レコードが更新されたことを示すエラーが発生します。

非正規化データから正規化データへの同期: 非正規化されているソース・アプリケーションから正規化されている宛先アプリケーションにデータを同期化する場合、アダプターは、正規化されているアプリケーションに含まれる所有関係にないデータに関しては、作成、削除、または更新を行いません。

正規化されているアプリケーションにデータを同期化する場合、アダプターは、所有関係にない単一カーディナリティーの子をすべて無視します。そのような子のデータを作成、除去、または変更するには、データを手動で処理する必要があります。

ビジネス・オブジェクトの複数カーディナリティー関係:

複数カーディナリティー関係では、親ビジネス・オブジェクト内の属性が子ビジネス・オブジェクトの配列を表します。属性は、子ビジネス・オブジェクトと同じ型です。アプリケーションが単一の子エンティティーを格納する場合を除き、関係を記述する外部キーは子に格納されます。親子関係は親に格納されます。

通常、子ビジネス・オブジェクトの配列を含むビジネス・オブジェクトには、関係を表す属性が 1 つだけ含まれており、通常はこの属性が基本キーになります。この属性のタイプは、子ビジネス・オブジェクトと同じタイプの配列です。親が複数の子を含むようにするため、関係を設定する外部キーは子に格納されます。

したがって、どの子にも、親の基本キーを外部キーとして含む単純属性が 1 つ以上存在します。子には、親に含まれる基本キー属性と同数の外部キー属性が含まれます。

関係を設定する外部キーが子に保管されるので、それぞれの親は、1 つ以上の子を持つことができます (子を持たないことも可能です)。

『ビジネス・オブジェクトの複数カーディナリティーの関係』の図に、複数カーディナリティーの関係を示します。この例では、3 つの ChildBOName ボックス内の

parentId は、親の基本キーを含む単純属性であり、ParentBOName ボックス内にある Child1 は、子ビジネス・オブジェクトの配列を表す属性です。

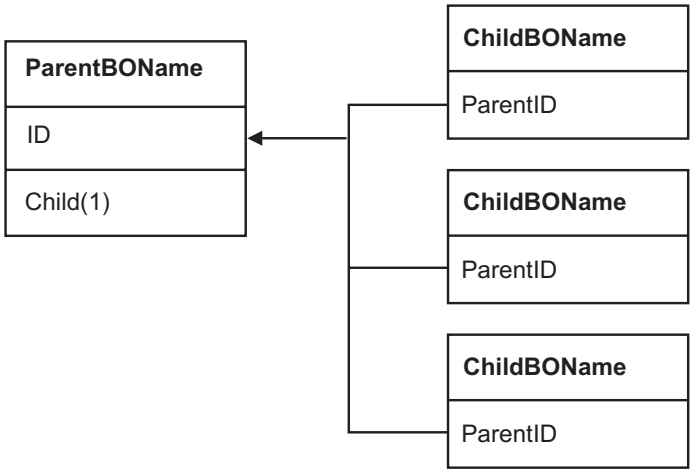


図 4. ビジネス・オブジェクトの複数カーディナリティーの関係

複数カーディナリティーの関係は、N=1 の関係である場合があります。アプリケーションによっては、親子関係を親ではなく子に格納するように、子エンティティーを 1 つ格納するものがあります。つまり、子には、親の基本キーに格納されている値と同一の値の外部キーが格納されます。

このタイプの関係がアプリケーションで使用されるのは、子のデータが親から独立して存在しておらず、親を介してのみそのデータにアクセスできる場合です。このような子のデータでは、子とその外部キー値を作成するために、親とその基本キー値があらかじめ存在していなければなりません。『N=1 の場合の複数カーディナリティーの関係』の図でこのタイプの関係を示します。

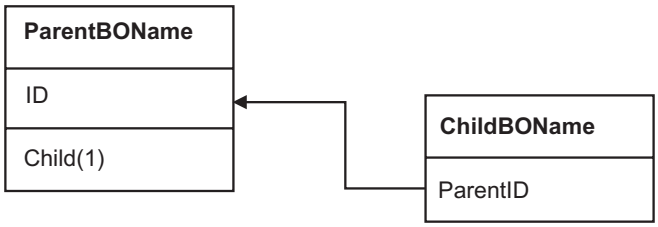


図 5. N=1 の場合の複数カーディナリティーの関係

ビジネス・オブジェクト属性のプロパティー

ビジネス・オブジェクトには、ビジネス・オブジェクトの内容を定義するために使用する属性があります。各属性は、名前、型、カーディナリティーなどのプロパティーを持ちます。ビジネス・オブジェクトは、属性で指定されるデータの単なるコンテナーです。

次の『属性プロパティー』というタイトルの表に、プロパティーとその解釈および設定値を示します。

表 2. 属性プロパティー

プロパティー	解釈と設定値
Cardinality	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各ビジネス・オブジェクト属性は、それぞれ、値 1 または複数 (n) のカーディナリティーを持ちます。
Foreign Key	カーディナリティーが *n* の子ビジネス・オブジェクトの配列が検索されると、SELECT ステートメントの WHERE 文節で外部キーが使用されます。 **注:** アダプターでは、子ビジネス・オブジェクトを表す属性を外部キーとして指定することについては、サポートしていません。 RetrieveAll 操作は、キーおよび外部キーの使用を指定変更します。
Primary Key	どのビジネス・オブジェクトでも、1 つ以上の単純属性が基本キーに指定されなければなりません。 **注:** アダプターでは、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基本キー属性として指定することについてはサポートしていません。 単純属性の基本キー・プロパティーを `true` に設定すると、アダプターは、ビジネス・オブジェクトの処理中に生成する SELECT および UPDATE の各 SQL ステートメントの WHERE 文節にその属性を追加します。 RetrieveAll 操作は、基本キーおよび外部キーの使用を指定変更します。
Name	このプロパティーは、属性が単純属性の場合は、属性の固有の名前。属性が子ビジネス・オブジェクトの場合は、ビジネス・オブジェクトの名前を表します。
Required	属性が値を含む必要があるかどうかを指定します。カーディナリティーが単一 (1) のコンテナーに対して、このプロパティーが `true` に設定されている場合、アダプターでは、その親ビジネス・オブジェクトが、この属性に対応する子ビジネス・オブジェクトを含んでいる必要があります。Create、Update、および Delete 操作でアダプターに渡されるビジネス・オブジェクトは、子ビジネス・オブジェクトも含んでいなければなりません。単純属性のカーディナリティーは単一 (1) で、コンテナー属性のカーディナリティーは複数 (n) です。ビジネス・オブジェクトが必須属性に対して有効な値またはデフォルト値を持っていないと、アダプターでは Create 操作が失敗します。このオブジェクトに対するデータベースからの検索時に使用可能なデータがない場合も、Create 操作は失敗します。
Type	単純属性の場合は属性の型 (Integer、String、 Date、Timestamp、Boolean、Double、Float など)。子ビジネス・オブジェクトの場合はビジネス・オブジェクトの型。アダプターはサポートされない型の属性を検出すると、値を引用符で囲み、その値を文字データとして処理します。

Outbound 操作

アプリケーション・コンポーネントは、エンタープライズ情報システム上のデータ検索などの操作を呼び出す必要があります。 Adapter for JDBC では、特定の

Outbound 操作が提供されます。サポートされる操作ごとにアダプターがビジネス・オブジェクトをどう処理するかについての詳細を説明します。

Create 操作

階層ビジネス・オブジェクトの場合は、Create 操作によってビジネス・オブジェクトが再帰的に全探索され、各テーブルに対応する行が作成されます。

以下で、詳しく説明します。

1. Create 操作は、所有関係を伴う単一カーディナリティーの各子ビジネス・オブジェクトを、データベース内に再帰的に挿入します。つまり、アダプターは、子ビジネス・オブジェクトおよびその子孫にあたるビジネス・オブジェクトのすべてを作成します。

   ビジネス・オブジェクト定義上、ある属性がある単一カーディナリティーの子ビジネス・オブジェクトを表すものとされている場合に、その属性が空であると、アダプターはその属性を無視します。ただし、ビジネス・オブジェクト定義により、その属性が子を表すことが必要であるにもかかわらず、子を表していない場合には、アダプターはエラーを戻して処理を停止します。

2. Create 操作は、所有関係を伴わない単一カーディナリティーの各子ビジネス・オブジェクトの有無を検索し、確認します。子がデータベース内に存在しないことを示して、Retrieve 操作が失敗した場合、アダプターはエラーを戻して処理を停止します。Retrieve 操作が成功した場合、アダプターは子ビジネス・オブジェクトを再帰的に更新します。

   注: アプリケーションのデータベースに子ビジネス・オブジェクトが存在する場合に、このアプローチが正しく機能するには、子ビジネス・オブジェクト内の基本キー属性の相互参照が、Create 操作時に正しく行われる必要があります。アプリケーション・データベースに子ビジネス・オブジェクトが存在しない場合、基本キー属性は設定してはいけません。

3. 最上位ビジネス・オブジェクトを、データベース内に次のように挿入します。
   a. 最上位ビジネス・オブジェクトの外部キー値を、対応する単一カーディナリティーの関係にある子ビジネス・オブジェクトの基本キー値に設定します。子ビジネス・オブジェクトの値は、データベース・シーケンスまたはカウンター、あるいはデータベース自体によって、子の作成時に設定される場合があります。そのため、このステップでは、アダプターが親をデータベースに挿入する前に、親の外部キー値を正しいものにします。
   b. データベースによって自動的に設定される属性のそれぞれに対して、新しい固有 ID 値を生成します。データベース・シーケンスまたはカウンターの名前は、属性のアプリケーション固有情報に格納されます。属性にデータベース・シーケンスまたはカウンターが関連付けられている場合、アダプターによって生成された値により、アプリケーション・サーバーから渡された値が上書きされます。データベース・シーケンスまたはカウンターの指定の詳細については、『単純属性のアプリケーション固有情報』の UID=AUTO を参照してください。
   c. 最上位ビジネス・オブジェクトをデータベース内に挿入します。
4. 複数カーディナリティーの子ビジネス・オブジェクトのすべてを、次のように処理します。

a. それぞれの子の外部キー値を、親に含まれる対応する基本キー属性の値を参照するように設定します。親の基本キー値は、親の作成時に生成されている可能性があります。そのため、ここでは、アダプターが子をデータベースに挿入する前に、それぞれの子の外部キー値を正しいものにします。

b. 複数カーディナリティーの子ビジネス・オブジェクトのすべてを、データベースに挿入します。

Retrieve 操作

このトピックでは、階層ビジネス・オブジェクトを検索するためにアダプターが使用する手順について説明します。

アダプターは次のように検索操作を実行します。

1. 受信した最上位ビジネス・オブジェクトから、すべての子ビジネス・オブジェクトを削除します。言い換えると、子のない最上位ビジネス・オブジェクトのコピーを作成します。
2. 最上位ビジネス・オブジェクトを、データベース内で検索します。
   - 検索の結果戻された行が 1 つの場合、アダプターは処理を継続します。
   - 検索の結果戻された行がない場合 (目的の最上位ビジネス・オブジェクトがデータベース内に存在しないことを意味します)、アダプターはエラー `RecordNotFoundException` を戻します。
   - 検索の結果戻された行が複数ある場合、アダプターはエラーを戻します。
3. 複数カーディナリティーの子ビジネス・オブジェクトのすべてを、再帰的に検索します。

   注: アダプターは、ビジネス・オブジェクトの配列を取り込むときに、一意性を保証しません。一意性の保証は、データベース側で行われなければなりません。データベースから戻された子ビジネス・オブジェクトに重複があると、アダプターは、それらの重複する子を戻します。
4. 子ビジネス・オブジェクトが所有関係にあるかどうかに関係なく、各単一カーディナリティーの子を再帰的に検索します。

   注: 単一カーディナリティーの子ビジネス・オブジェクトはすべて、ビジネス・オブジェクト内での出現順序に従って、親ビジネス・オブジェクトが処理される前に処理されます。子オブジェクトに対する所有関係の有無は、処理シーケンスを決定しませんが、処理のタイプは決定します。

RetrieveAll 操作

アダプターは、RetrieveAll 操作を使用してデータベースからビジネス・オブジェクトの配列を検索します。処理は、RetrieveAll 操作がデータベース表ビジネス・オブジェクトとユーザー指定 SQL ビジネス・オブジェクトのいずれを対象としているかに応じて異なります。

データベース表ビジネス・オブジェクトの場合

着信ビジネス・オブジェクト内に取り込まれているすべてのキー属性および非キー属性によって、選択基準が決まります。選択した属性によっては、アダプターは、データベースから最上位ビジネス・オブジェクトの複数の行を検索する場合もあり

ます。着信ビジネス・オブジェクト内に属性が取り込まれていない場合は、データベース内のそれぞれのテーブルからすべての行が検索されます。

アダプターは次のステップを実行してビジネス・オブジェクトの配列を検索します。

1. データベースから検索された行ごとに、アダプターは最上位ビジネス・グラフを構成し、検索された行をすべて使用してビジネス・グラフのコンテナーを作成します。コンテナー・ビジネス・オブジェクトの名前は *BOName* ＋ *ContainerBG* です。
2. アダプターは、Retrieve 操作を使用してコンテナー内の各ビジネス・グラフを取得します。

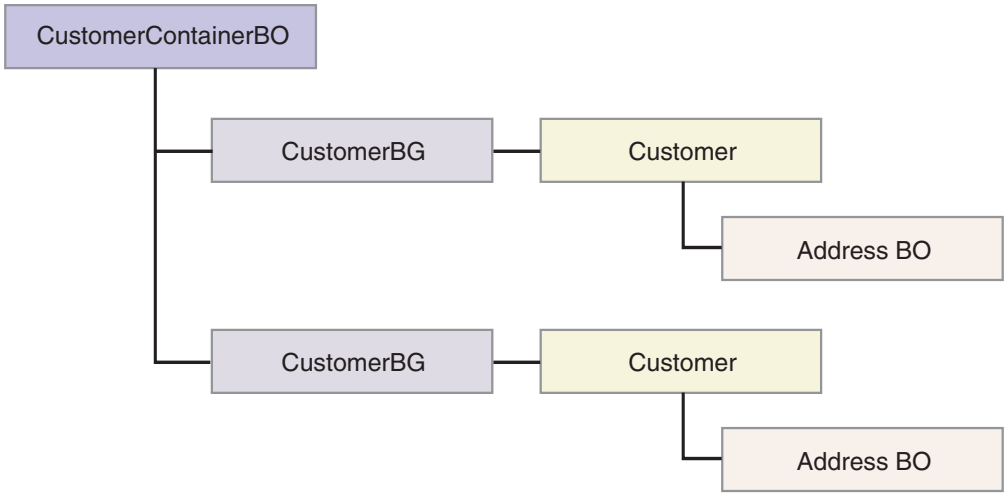


図 6. RetrieveAll 操作で戻されるビジネス・オブジェクトの構造

RetrieveAll 操作によって、次のエラーが発生する可能性があります。

- 入力オブジェクト内に取り込まれたビジネス・オブジェクトが EIS に存在しない場合、アダプターはエラー `RecordNotFoundException` を戻します。
- EIS 内のヒット数が、対話スペックで定義された ResultSetLimit の値を超えると、アダプターはエラー MatchesExceededLimitException を戻します。MatchCount プロパティーには、アダプターが EIS 内に保持するヒットの実際の数が含まれているため、制限値を高くしたり、検索を適度に詳細化することができます。

  注: ResultSetLimit を大きい数に設定すると、戻されるビジネス・オブジェクトのサイズと数によってはメモリー不足に関連する問題が発生する場合があります。
- EIS によってリカバリー不能エラーが報告された場合、アダプターはエラー EISSystemException を戻します。

ユーザー指定 SQL ビジネス・オブジェクトの場合

ユーザー指定 SQL ステートメントに対して作成されたビジネス・オブジェクトも RetrieveAll 操作をサポートします。エンタープライズ・サービス・ディスカバリー (ESD) ウィザードがユーザー指定 SQL ステートメントに対するクエリー・ビジネ

ス・オブジェクトを生成すると、アダプターは SQL SELECT ステートメントを実行し、下図に示す構造でデータベースにデータを返します。

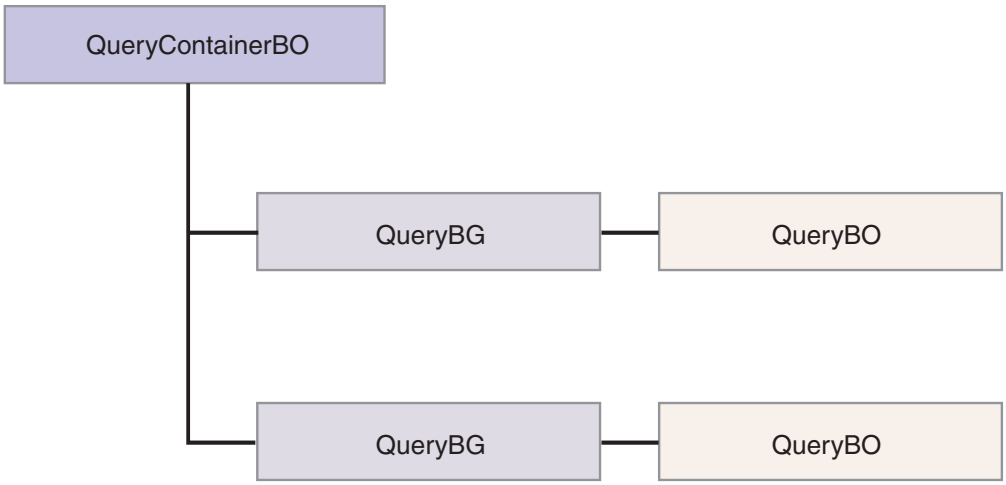


図 7. ユーザー指定 SQL ビジネス・オブジェクト

ESD によってユーザー指定 SELECT ステートメントに生成されたクエリー・ビジネス・オブジェクトを処理するために、アダプターは以下のように動作します。

1. クエリー・ビジネス・オブジェクトから SELECT SQL ステートメントを取得します。
2. クエリー・ビジネス・オブジェクトで動的 where 文節が指定されているかどうかを検査します。
   - 動的 where 文節がある場合、アダプターは、SELECT ステートメントのデフォルト where 文節をその動的 where 文節で置換します。
   - 動的 where 文節がない場合、アダプターは、SELECT ステートメントのパラメーターをクエリー・ビジネス・オブジェクトで指定された対応する値で置換します。
3. SELECT ステートメントを実行します。
4. 返された結果セットを取得し、クエリー・ビジネス・オブジェクト値にデータベースから返されたデータを設定し、前記の図に示した構造のコンテナー・ビジネス・オブジェクトを作成します。
5. クエリー・ビジネス・オブジェクトに子ビジネス・オブジェクトが定義されているかどうかに応じて、コンテナー内の各最上位クエリー・ビジネス・オブジェクトの下降検索を実行します。

   注: クエリー・ビジネス・オブジェクトを最上位ビジネス・オブジェクト以外にすることはできません。クエリー・ビジネス・オブジェクトが子クエリー・ビジネス・オブジェクトを持つことはできません。

Update 操作

Update 操作は、着信ビジネス・オブジェクトを、最上位の着信ビジネス・オブジェクトで指定された基本キーを使用してデータベースから検索されたビジネス・オブジェクトと比較することによって実行されます。

アダプターでは、階層ビジネス・オブジェクトの更新時に、以下のステップを実行します。

1. ソース・ビジネス・オブジェクトの基本キー値を使用して、データベース内の対応するエンティティーを検索します。検索されたビジネス・オブジェクトは、データベース内のデータの現在の状態を正確に表したものです。

   検索が失敗した場合 (最上位ビジネス・オブジェクトがデータベース内に存在しないことを意味します)、アダプターは `RecordNotFoundException` エラーを戻し、更新は失敗します。

   検索に成功した場合、アダプターは、検索されたビジネス・オブジェクトをソース・ビジネス・オブジェクトと比較して、どの子ビジネス・オブジェクトに関してデータベースに変更を加える必要があるかを判別します。ただし、アダプターはソース・ビジネス・オブジェクトの単純属性の値と検索されたビジネス・オブジェクトの単純属性の値を比較しません。アダプターは、非キーの単純属性すべての値を更新します。

   最上位ビジネス・オブジェクトのすべての単純属性がキーを表している場合、アダプターはその最上位ビジネス・オブジェクト用の更新照会を生成できません。この場合、アダプターは、警告を記録してからステップ 2 に進みます。
2. 最上位ビジネス・オブジェクトの子のうち、単一カーディナリティーのものすべてを再帰的に更新します。

   ビジネス・オブジェクト定義上、ある属性がある子ビジネス・オブジェクトを表すことが必須である場合には、その子ビジネス・オブジェクトがソース・ビジネス・オブジェクトと検索されたビジネス・オブジェクトの両方に存在している必要があります。存在しない場合、Update 操作は失敗し、アダプターはエラーを戻します。

   アダプターでは、所有関係にある単一カーディナリティーの子を、次のいずれかの方法で処理します。

   - ソース・ビジネス・オブジェクトおよび検索したビジネス・オブジェクトの両方に子が存在する場合、アダプターは、データベース内の既存の子を更新するのではなく、既存の子を削除して新規の子を作成します。
   - その子がソース・ビジネス・オブジェクトには存在するにもかかわらず、検索されたビジネス・オブジェクトには存在しない場合、アダプターはデータベース内にその子を再帰的に作成します。
   - その子が検索されたビジネス・オブジェクトには存在するにもかかわらず、ソース・ビジネス・オブジェクトには存在しない場合、アダプターはデータベース内のその子を再帰的に削除します。

   所有関係にない単一カーディナリティーの子に関しては、アダプターは、ソース・ビジネス・オブジェクトに存在するそのような子のすべてを、データベースから検索しようとします。アダプターは、子の検索に成功すると、その子ビジネス・オブジェクトにデータを読み込みますが、更新は行いません。これは、所有関係にない単一カーディナリティーの子はアダプターによって変更されることがないためです。

3. 検索されたビジネス・オブジェクトのすべての単純属性を更新します。ただし、ソース・ビジネス・オブジェクト内の対応する属性が指定されていない場合を除きます。

   更新されるビジネス・オブジェクトは一意である必要があるため、アダプターは、結果として 1 行のみが処理されることを確認します。複数の行が戻されている場合、アダプターはエラーを戻します。
4. 検索されたビジネス・オブジェクトの複数カーディナリティーの子のそれぞれを、次のいずれかの方法で処理します。
   - その子がソース・ビジネス・オブジェクトの配列と検索されたビジネス・オブジェクトの配列の両方に存在する場合、アダプターはデータベース内でその子を再帰的に更新します。
   - その子がソース・ビジネス・オブジェクトの配列には存在しても、検索されたビジネス・オブジェクトの配列には存在しない場合、アダプターはデータベース内でその子を再帰的に作成します。
   - その子が検索されたビジネス・オブジェクトの配列には存在しても、ソース・ビジネス・オブジェクトの配列には存在しない場合、アダプターはデータベースからその子を再帰的に削除します。ただし、親に含まれている、その子を表す属性のアプリケーション固有情報で、KeepRelationship が `true` に設定されている場合を除きます。この場合、アダプターは、データベースからその子を削除しません。

DeltaUpdate 操作

InteractionSpec 内の操作が ApplyChange で、ビジネス・グラフ内に動詞が存在しない場合、アダプターは DeltaUpdate 操作を実行します。アダプターは、ChangeSummary をインスペクションして入力階層内の各ビジネス・オブジェクトの操作を識別し、識別された操作を実行します。

次に示すように、DeltaUpdate 操作は Update 操作とは異なります。

- DeltaUpdate 操作では、更新の前に Retrieve 操作は実行されません。
- 着信ビジネス・オブジェクトとデータベース内のビジネス・オブジェクトの比較が行われません。
- 子はすべて、各子オブジェクトに設定されている動詞に基づいて処理されます。子に動詞が設定されていない場合、アダプターはエラーを戻します。

アダプターは、DeltaUpdate による階層ビジネス・オブジェクトの更新時に、以下のステップを実行します。このステップでは、ChangeSummary から、オブジェクトの変更内容のみが処理されます。

1. 親オブジェクトの子のうち、単一カーディナリティーのものすべてを再帰的に処理します。ビジネス・オブジェクト仕様で必須にマークされている子は、Inbound オブジェクトに存在していなければなりません。存在しない場合、DeltaUpdate 操作は失敗し、アダプターがエラーを戻します。
2. 親に含まれる外部キー値のうち、単一カーディナリティーの子の属性を参照するものすべてを、それぞれ対応する子の値に設定します。この処理が必要なのは、これ以前のステップで、単一カーディナリティーの子がデータベースに追加され、新しいシーケンス値が生成されている可能性があるためです。

3. 現在処理中のオブジェクトを、SQL Update ステートメントまたはストアード・プロシージャーを使用して更新します。個々のビジネス・オブジェクトのすべての単純属性が更新されます。アダプターは Update ステートメントにどの属性を追加しなければならないかを判断するのに、プロパティー・レベルの変更を使用しません。すべて更新されます。更新されているオブジェクトは固有であるため、アダプターは結果として 1 行のみが処理されていることをチェックします。複数の行が処理される場合、エラーが戻されます。
4. 現在のオブジェクトのカーディナリティー N のすべての子にある、親の属性を参照する外部キー値をすべて、対応する親の値に設定します。通常、これらの値はデータ・マッピング時に既に相互参照されています。ただし、これはカーディナリティーが N のコンテナーに含まれる新しい子には該当しない場合があります。このステップにより、カーディナリティーが N の子すべての外部キー値が正しい値になってから、それらの子の更新が行われることが徹底されます。
5. 現在のオブジェクトの、カーディナリティーが N のコンテナーをすべて更新します。

   子オブジェクトが処理されるときには、それぞれの子の動詞が取得され、適切な操作が実行されます。DeltaUpdate で子に対して許可される操作は、Create、Delete、および Update です。

   - Create 動詞が子で検出された場合、それが所有関係にある子であれば、検出された子がデータベース内に作成されます。所有関係にない子に関しては、検索により、データベースに存在するかどうかが確認されます。
   - Delete 動詞が子で検出された場合、子は削除されます。
   - Update 動詞が子で検出された場合、子はデータベース内で更新されます。

Delete 操作

Delete 操作は、データベースからの着信ビジネス・オブジェクトのプルーニングと、その後の完全なビジネス・オブジェクトの検索によって実行されます。Delete 操作は、その後階層内の各ビジネス・オブジェクトについて再帰的に適用されます。

Delete 操作では、オブジェクトのアプリケーション固有情報のステータス列名の値に応じて、物理削除と論理削除がサポートされます。ステータス列名の値が定義されている場合、アダプターは論理削除操作を実行します。ステータス列名の値が定義されていない場合、アダプターは物理削除操作を実行します。

物理削除の場合、アダプターは次の処理を行います。

- 複数カーディナリティーの子ビジネス・オブジェクトすべてを再帰的に削除します。
- 最上位ビジネス・オブジェクトを削除します。
- 所有関係にある単一カーディナリティーの子ビジネス・オブジェクトすべてを、再帰的に削除します。

論理削除の場合、アダプターは次の処理を行います。

- Update を発行して、ビジネス・オブジェクトの状況属性を、ビジネス・オブジェクトのアプリケーション固有情報によって指定されている値に設定します。アダプターでは、結果として 1 つのデータベース行だけが更新されることを確認します。それ以外の場合は、エラーを戻します。

- 所有関係にある単一カーディナリティーの子のすべて、および複数カーディナリティーの子のすべてに対し、論理削除を再帰的に実行します。アダプターは、所有関係にない単一カーディナリティーの子は削除しません。

ApplyChanges 操作

ApplyChanges 操作には、さまざまな処理が含まれます。この操作によって、Create、Update、または Delete 操作を要求するビジネス・オブジェクトに対して、アダプターがそれに応じた処理を実行することができます。

ビジネス・オブジェクトに最上位の動詞が存在する場合、そのビジネス・オブジェクトは変更後イメージとして処理されます。ビジネス・オブジェクトに最上位の動詞が存在しない場合は、ChangeSummary が処理されます。

Execute 操作

Execute 操作はストアード・プロシージャーの実行に使用します。エンタープライズ・サービス・ディスカバリー・ウィザードは、データベースのストアード・プロシージャー定義に対応する必要なストアード・プロシージャー・ビジネス・オブジェクトを生成します。アダプターは、Execute 操作を使用してストアード・プロシージャー・ビジネス・オブジェクトを処理します。

ストアード・プロシージャー、そのストアード・プロシージャーから構成されるビジネス・オブジェクト、および Execute 操作でストアード・プロシージャー・ビジネス・オブジェクトを処理するためにアダプターが使用するステップの単純な例を以下に示します。

ストアード・プロシージャーの単純な例:

```
PROCEDURE testSP(IN int x,INOUT VARCHAR(10) msgSTR, OUT int
status, OUT struct outrec, OUT array retArr)
```

(プロシージャーが 2 つの結果セットを返します)

このストアード・プロシージャーの場合に構成されるビジネス・オブジェクトの例を以下に示します。

```
BOLevel ASI
      SPName=testSP
      ResultSet=true
      MaxNumberOfResultSets=2
      ReturnValue = propName (関数の場合)。
戻り値が複合型 (array/struct/resultset) である場合は、
子ビジネス・オブジェクトに対応するプロパティー名になります。
関数の場合にのみ定義されます。
プロパティー
      x Type=IP
      msgStr Type=IO
      status Type=OP
      outrec Type OP - outrec の子 BO、ASI ChildBOType = struct
      retarr Type OP - retArr の n カーディナリティー子 BO、ASI ChildBOType = array
      childBOName1 - 最初の結果セットに対する子 BO、ASI ChildBOType = resultset
      childBOName2 - 2 番目の結果セットに対する子 BO、ASI ChildBOType = resultset
```

Execute 操作でこのストアード・プロシージャー・ビジネス・オブジェクトを処理するために、アダプターは以下の処理を行います。

1. ストアード・プロシージャー呼び出し `CALL testSP(x, msgStr, status, outrec)` を構成します。

2. 呼び出し可能ステートメントで入力パラメーターを設定します。
3. 呼び出し可能ステートメントを実行します。
4. 戻り値を取得し (関数の場合)、それがスカラー値であればその値を適切な属性に設定します。複合値 (構造体や配列など) であれば子ビジネス・オブジェクトに設定します。
5. 最初の結果セットを取得し、ResultSet1 のコンテナーを作成します。
6. 2 番目の結果セットを取得し、ResultSet2 のコンテナーを作成します。
7. 出力パラメーター `msgStr` および `status` を取得し、ビジネス・オブジェクトで対応する属性を設定します。
8. 出力パラメーター `outrec` を取得し、`outrec` で返されたデータから子ビジネス・オブジェクトを作成します。`outrec` がネストされた構造体型である場合、アダプターは階層子ビジネス・オブジェクトを再帰的に作成してデータを格納します。
9. 出力パラメーター `retArr` を取得し、`retArr` で返されたデータから n カーディナリティーの子ビジネス・オブジェクトを作成します。`retArr` がネストされた配列型である場合、アダプターは階層子ビジネス・オブジェクトを再帰的に作成してデータを格納します。

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は、アダプターに対し、ビジネス・オブジェクトの処理方法に関するアプリケーション依存の指示を与えるものです。アダプターでは、ビジネス・オブジェクトの属性または動詞、あるいはビジネス・オブジェクト自体から取得したアプリケーション固有情報を解析して、Create、Update、Retrieve、および Delete 操作のための照会を生成します。

アダプターは、ビジネス・オブジェクトのアプリケーション固有情報の一部については、キャッシュに保管し、その情報をすべての動詞の照会をビルドするために使用します。

拡張または変更されたアプリケーション固有のビジネス・オブジェクトでは、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は、アダプターの予期する構文に合致している必要があります。

ビジネス・オブジェクト・レベルのアプリケーション固有情報

ビジネス・オブジェクト・レベルのアプリケーション固有情報は、対応するデータベース表の名前を指定するとき、および物理削除または論理削除操作の実行に必要な情報を指定するときに使用されます。

ビジネス・オブジェクト・レベルでは、アプリケーション固有情報の形式は、jdbcasi.xsd スキーマ定義で定義される eXtensible Markup Language (XML) コードで構成されています。アダプターは、以下の表に示すビジネス・オブジェクト・レベルのアプリケーション固有情報をサポートします。

表 3. ビジネス・オブジェクト・レベルのアプリケーション固有情報

アプリケーション固有情報	型	説明	サポートされる双方向変換
MaxNumberOfRetRS	String	ストアード・プロシージャーが戻す結果セットの数	いいえ

表 3. ビジネス・オブジェクト・レベルのアプリケーション固有情報 (続き)

アプリケーション固有情報	型	説明	サポートされる双方向変換
ResultSet	Boolean	ストアード・プロシージャーが結果セットを戻すかどうかを示す値	いいえ
ReturnValue	String	ストアード関数ビジネス・オブジェクトの対応する属性の名前	いいえ
SelectStatement	String	select ステートメント	はい
SPName	String	ストアード・プロシージャーの名前	はい
ステータス列名	String	論理削除操作の実行に使用するデータベース列の名前	はい
ステータス値	String	ビジネス・オブジェクトが非アクティブか削除済みかを示す値	いいえ
TableName	String	ビジネス・オブジェクトに関連付けられたデータベース表の名前	はい

例えば、Customer ビジネス・オブジェクトで、そのアプリケーション固有情報に、以下の値が指定されているとします。

```
<jdbcasi:TableName>customer</jdbcasi:TableName>
<jdbcasi:StatusColumnName>status</jdbcasi:StatusColumnName>
<jdbcasi::StatusValue>deleted</jdbcasi:StatusValue>
```

アダプターが、カスタマーの削除要求を受信したとします。このような要求により、アダプターは次の SQL ステートメントを発行します。

```
UPDATE customer set status = 'deleted' where pkey = . . . .
```

アダプターは、ステータス列名が含まれていない場合や、このパラメーターに値が指定されていない場合には、データベースからビジネス・オブジェクトを物理的に削除します。つまり、ビジネス・オブジェクトでアプリケーション固有情報に StatusColumnName パラメーターが含まれる場合、アダプターは論理削除操作を実行します。ビジネス・オブジェクトでアプリケーション固有情報に StatusColumnName パラメーターが含まれない場合、アダプターは物理削除操作を実行します。

Update 操作と Delete 操作では、StatusColumnName プロパティーの値を以下のように使用できます。

- 子データを論理的に削除する場合、アダプターは StatusColumnName パラメーターの値を使用して、状況列の名前と状況値のテキストを取得します。詳細については、『Update 操作』を参照してください。
- Delete 操作を実行するとき、アダプターは、その StatusColumnName パラメーターの値を使用して、ビジネス・オブジェクト全体を物理的に削除するか、論理的に削除するかを判断します。StatusColumnName パラメーターに値が含まれている場合、アダプターは、論理削除操作を実行します。StatusColumnName パラメーターに値が含まれていない場合、アダプターは、物理削除操作を実行します。詳細については、『Delete 操作』を参照してください。

双方向言語用に使用可能に設定するアプリケーション固有情報のパラメーターは、TableName、StatusColumnName、SPName、および SelectStatement です。これらのパラメーターの形式は、BiDiEIS、BiDiMetadata、BiDiSkip、および BiDiSpecialFormat

プロパティーに設定されている属性に基づいて変換されます。これらのプロパティーについて詳しくは、 170 ページの『アダプター構成プロパティー』を参照してください。

動詞のアプリケーション固有情報

アダプターは、動詞アプリケーション固有情報を使用してデータベース内の情報の検索や更新などの操作を実行します。アダプターは、ビジネス・オブジェクトでの指定に従って、SQL 照会、ストアード・プロシージャー、またはストアード関数を使用してデータベース表を更新します。

ストアード・プロシージャーの概要:

ストアード・プロシージャーは、複数の SQL ステートメントのグループであり、1 つの論理単位を形成して特定のタスクを実行します。ストアード・プロシージャーは、アダプターがオブジェクトに対して実行する一連の操作または照会を、データベース・サーバー内にカプセル化したものです。

アダプターは、単純な SQL ステートメントを使用して、
Create、Update、Retrieve、Delete、または RetrieveAll 操作を実行できます。SQL ステートメント用の列名は、属性のアプリケーション固有の情報から取り出されます。where 文節は、ビジネス・オブジェクトで指定されたキー値を使用して構成されます。各照会は、複数のテーブルにまたがることはできません。ただし、ビューに送ることはできます。

アダプターは、次の目的でストアード・プロシージャーを呼び出します。

- ビジネス・オブジェクトを処理する前に、操作準備処理を行う。
- ビジネス・オブジェクトを処理した後で、操作後処理を行う。
- 単純な Create、Update、Delete、Retrieve または RetrieveAll ステートメントを使用せずにビジネス・オブジェクトに対して一連の操作を実行する。

アダプターでは、階層ビジネス・オブジェクトを処理するときに、ストアード・プロシージャーを使用して、最上位ビジネス・オブジェクトまたは任意の子ビジネス・オブジェクトを処理することができます。ただし、ビジネス・オブジェクト (またはビジネス・オブジェクトの配列) には、ストアード・プロシージャーが個別に用意されていなければなりません。

ストアード・プロシージャー定義:

ストアード・プロシージャーは動詞レベルで定義されます。ストアード・プロシージャー定義はそれぞれ、エレメント StoredProcedureType、StoredProcedureName、ResultSet、Parameters、および ReturnValue から構成されます。

ストアード・プロシージャー定義エレメントを以下の表に示します。

表 4. ストアード・プロシージャー定義

エレメント	型	説明	値	サポートされる双方向変換
StoredProcedure Type	String	使用するストアード・プロシージャーのタイプを定義します。これにより、ストアード・プロシージャーがいつ呼び出されるか (ビジネス・オブジェクトの処理前など) が決まります。 **注:** RetrieveAll に関連するストアード・プロシージャーのタイプは、最上位ビジネス・オブジェクトにのみ適用されます。	以下のものが可能です。 • Before*Verb*SP、 • After*Verb*SP、または • *Verb*SP ここで、verb は Create、Update、Delete、Retrieve、または RetrieveAll です。	いいえ
StoredProcedure Name	String	適切な StoredProcedureType と関連したストアード・プロシージャーの名前です。		はい
ResultSet	Boolean	この値は、ストアード・プロシージャーが結果を戻すかどうかを決定します。結果のセットが戻される場合は、結果のセットの行で戻される値を使用して、現在のビジネス・オブジェクトの N カーディナリティーの子が作成されます。	`true|false`	いいえ
Parameters	String	ストアード・プロシージャーのパラメーターのリストを定義します。 **注:** Oracle ストアード・プロシージャーの場合は、結果のセットが出力パラメーターとしてのみ戻されます。その場合、パラメーター・リスト内の値のいずれかが結果のセット (RS) になります。	以下の組み合わせ • `IP`: 入力専用 • `OP`: 出力専用 • `IO`: 入出力。	いいえ

表 4. ストアード・プロシージャー定義 (続き)

エレメント	型	説明	値	サポートされる双方向変換
ReturnValue	String	関数によって値が返されるため、プロシージャー呼び出しではなく関数呼び出しであることを示す値。戻り値が RS である場合、値は結果セットです。この結果セットは、このビジネス・オブジェクトに対応する N カーディナリティー・コンテナーの作成に使用されます。戻り値が属性である場合、値はビジネス・オブジェクトの特定の属性に割り当てられます。属性が別の子ビジネス・オブジェクトである場合、アダプターはエラーを返します。	RS またはビジネス・オブジェクト属性を指定できます。	いいえ

次に、ストアード・プロシージャー定義の例を示します。

```
<jdbcasi:JDBCBusinessObjectTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
            <jdbcasi:TableName>customer</jdbcasi:TableName><jdbcasi:Operation>
               <jdbcasi:Name>Retrieve</jdbcasi:Name>
               <jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>RetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>retrieve_cust</jdbcasi:StoredProcedureName>
<jdbcasi:ResultSet>false</jdbcasi:ResultSet>
                     <jdbcasi:Parameters>
                          <jdbcasi:Type>IP</jdbcasi:Type>
                          <jdbcasi:PropertyName>primaryKey</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                           <jdbcasi:PropertyName>custCode</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                           <jdbcasi:PropertyName>firstName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                          <jdbcasi:Type>OP</jdbcasi:Type>
                              <jdbcasi:PropertyName>lastName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    </jdbcasi:StoredProcedures>
                    <jdbcasi:StoredProcedures>
<jdbcasi:StoredProcedureType>AfterRetrieveSP</jdbcasi:StoredProcedureType>
<jdbcasi:StoredProcedureName>retrieve_cust</jdbcasi:StoredProcedureName>
                    <jdbcasi:ResultSet>false</jdbcasi:ResultSet>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>IP</jdbcasi:Type>
                           <jdbcasi:PropertyName>primaryKey</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                          <jdbcasi:PropertyName>custCode</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
```

```
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                          <jdbcasi:PropertyName>firstName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
                    <jdbcasi:Parameters>
                         <jdbcasi:Type>OP</jdbcasi:Type>
                         <jdbcasi:PropertyName>lastName</jdbcasi:PropertyName>
                    </jdbcasi:Parameters>
               </jdbcasi:StoredProcedures>
          </jdbcasi:Operation>
</jdbcasi:JDBCBusinessObjectTypeMetadata>
```

結果セットが true であるが空である場合でも、リソース・アダプター・プロパティー ReturnDummyBOForSP は出力パラメーターを戻します。RetrieveSP の場合、結果のセットが戻されます。結果のセットが空の場合は、ビジネス・オブジェクトが生成されず、プロシージャー呼び出しの戻す出力パラメーターをリトリーブする方法も存在しません。ReturnDummyBOForSP が true の場合は、対応する属性に読み込まれた出力パラメーターと入出力パラメーターの値を持つダミーのビジネス・オブジェクトが戻されます。このプロパティーのデフォルト値は false です。

ストアード関数の概要:

ストアード関数は、常に値を戻すという点を除き、ストアード・プロシージャーに類似しています。Oracle などの一部のデータベースでは、関数を明示的に定義することができます。IBM DB2 などのその他のデータベースでは、値を戻すストアード・プロシージャーが、関数のように動作します。

ストアード・プロシージャー呼び出しは、以下のように実行されます。

```
call SPName (<parameter list>)
```

これに対して、関数呼び出しは以下のように実行されます。

```
? = call FunctionName (<parameter list>)
```

ReturnValue ビジネス・オブジェクト・アプリケーション固有情報を使用して、戻り値を含む属性を指定します。ReturnValue について詳しくは、『ビジネス・オブジェクト・レベルのアプリケーション固有情報』を参照してください。

属性に関するアプリケーション固有情報

ビジネス・オブジェクト属性のアプリケーション固有情報 (ASI) は、属性が単純属性であるか、子ビジネス・オブジェクト (または子ビジネス・オブジェクトの配列) を表す属性であるかによって異なります。子を表す属性のアプリケーション固有情報は、親子関係が子に格納されるか親に格納されるかによっても異なります。

単純属性のアプリケーション固有情報

単純属性では、アプリケーション固有情報の形式は、いくつかのパラメーターとその値で構成されています。属性のアプリケーション固有情報の形式は、.xsd ファイルの以下の実例セクションに示します。

.xsd ファイルの例

```
<jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
<jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
<jdbcasi:FixedChar>true</jdbcasi:FixedChar>
```

```
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<simpleType>
   <restriction base="string">
   <maxLength value="10"/>
   </restriction>
</simpleType>
</element>
<element name="custCode" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
   <jdbcasi:ForeignKey>custinfoObj/custCode</jdbcasi:ForeignKey>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="firstName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>fname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
<element name="lastName" type="string">
<annotation>
<appinfo source="WBI">
<jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:ColumnName>lname</jdbcasi:ColumnName>
</jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
</element>
```

アダプターが処理する単純属性に必須のパラメーターは、列名のみです。例えば、以下は、列名のみを指定する形式です。ここで、ccode はカスタマー・コードを表します。

```
<jdbcasi:ColumnName>ccode</jdbcasi:ColumnName>
```

双方向言語で使用可能な属性のアプリケーション固有情報パラメーターは ColumnName です。このパラメーターの形式は、BiDiEIS、BiDi.Metadata、BiDiSkip、および BiDiSpecialFormat プロパティーに設定されている属性に基づいて変換されます。これらのプロパティーについて詳しくは、『参照』のセクションの『アダプター構成プロパティー』を参照してください。双方向プロパティーについて詳しくは、双方向言語サポートの一般技術文書、および IBM developerWorks(R) Web サイトのアダプター技術文書を参照してください。

属性のアプリケーション固有情報を設定する場合は、下記の表に示すパラメーターを指定します。

表 5. 単純属性のアプリケーション固有情報

パラメーター	型	説明	デフォルト値	サポートされる双方向変換
BLOB	Boolean	この属性に対応するデータベース列が BLOB データ型であることを指定します。BLOB データの表示中、アダプターはバイト数を 16 進値で表示するのみです。属性の型は hexBinary です。	なし	いいえ
ByteArray	Boolean	`true` の場合、アダプターはデータベースでバイナリー・データを読み取りおよび書き込みし、そのデータをストリングとしてアプリケーション・サーバーに渡します。	`false`	いいえ
ChildBOType	String	複合パラメーターの型を以下のいずれかとして指定します。 • Struct、 • Array、または • ResultSet	なし	いいえ
ChildBOTypeName	String	ChildBOType アプリケーション固有情報の値が Struct または Array の場合、このパラメーターの値はユーザー定義型の名前を表します。		いいえ
CLOB	Boolean	この属性に対応するデータベース列が CLOB データ型であることを指定します。String 属性型の場合にのみ適用可能です。 **注:** CLOB データ型は以下のように定義されます。 • CLOB 属性は String 型を持ち、この長さを使用して CLOB の長さを定義します。 • CLOB 属性は ASI CLOB=true を持ちます。	なし	いいえ
ColumnName	String	この属性に対するデータベース列の名前。	なし	はい
DateType	String	対応するエレメントを Date、Time、または TimeStamp 型として処理するように指定します。DateType パラメーターには、それに対する実際の値が保管されています。例えば以下のようになります。 • <DateType>Date</DateType> • <DateType>Time</DateType> • <DateType>TimeStamp</DateType> これらの値を設定するには、以下のフォーマットを使用します。 • Date には yyyy-mm-dd を使用します。 • Time には hh:mm:ss を使用します。 • TimeStamp には yyyy-mm-dd hh:mm:ss を使用します。	なし	いいえ

表 5. 単純属性のアプリケーション固有情報 (続き)

パラメーター	型	説明	デフォルト値	サポートされる双方向変換
FixedChar	Boolean	テーブル内の列が VARCHAR 型ではなく CHAR 型である場合に、属性を固定長とするかどうかを指定します。例えば、ある特定の属性が CHAR 型の列にリンクされている場合、アダプターはデータベースを照会するときに、属性値をその属性の最大長までブランクで埋めます。 ビジネス・オブジェクトの .xsd ファイルでは、このパラメーターは手動で更新する必要があります。ファイルはテキスト・モードで編集したり、WebSphere Integration Developer の Business Object Editor で編集したりできます。更新後は .xsd 内で検証エラーが発生していないことを確認してください。この表に続くこのパラメーターのコード例を参照してください。	`false`	いいえ
ForeignKey	String	このプロパティーの値は、親子関係が親ビジネス・オブジェクトに格納されるか、子ビジネス・オブジェクトに格納されるかによって異なります。 親に保管される場合、子ビジネス・オブジェクトのタイプと、外部キー *(Child_BO_name/Child_Property_Name)* として使用される子ビジネス・オブジェクト内の属性の名前の両方が含まれるように値を設定します。 子に保管される場合は、外部キーとして使用される親ビジネス・オブジェクト内の属性の名前のみを含むように値を設定します。 属性が外部キーではない場合は、アプリケーション固有情報にこのパラメーターを含めないでください。	なし	いいえ
OrderBy	String	このパラメーターに値が指定されている場合、このパラメーターが指定されている属性が子ビジネス・オブジェクト内に存在するものであれば、アダプターでは、検索照会の ORDER BY 文節で、その属性の値を使用します。アダプターは、子ビジネス・オブジェクトを昇順 (ASC) または降順 (DESC) で検索することができます。このパラメーターがアプリケーション固有情報に含まれていない場合、アダプターは、検索順序を指定するときに、このパラメーターが指定されている属性を使用しません。	なし	いいえ

表 5. 単純属性のアプリケーション固有情報 (続き)

パラメーター	型	説明	デフォルト値	サポートされる双方向変換
PrimaryKey	Boolean	値が true の場合、このプロパティーはこの属性に関連した列が、データベース内の対応するテーブルの基本キーであることを暗黙指定します。	なし	いいえ
SPParameterType	String	ストアード・プロシージャー・パラメーターのタイプを IP (入力専用)、OP (出力専用)、IO (入力および出力)、または RS (結果セット) から指定します。	なし	いいえ
UniqueIdentifier (UID)	String	アダプターは、ビジネス・オブジェクトの固有 ID の生成に、このパラメーターを使用します。シーケンスと ID 列 *(UID=AUTO|Sequence_Name)* の生成がサポートされます。シーケンスを定義できるのは、DB2 および Oracle データベースのみです。ID 列を定義できるのは、DB2 および Microsoft SQL Server です。属性で固有 ID が必要とされない場合は、このパラメーターをアプリケーション固有情報に含めないでください。	なし	いいえ

ビジネス・オブジェクトの .xsd ファイル内の FixedChar パラメーターの例

```
<element name="primaryKey">
<annotation>
<appinfo source="WBI">
        <jdbcasi:JDBCAttributeTypeMetadata
xmlns:jdbcasi="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
                <jdbcasi:ColumnName>pkey</jdbcasi:ColumnName>
                <jdbcasi:PrimaryKey>true</jdbcasi:PrimaryKey>
                <jdbcasi:FixedChar>true</jdbcasi:FixedChar>
        </jdbcasi:JDBCAttributeTypeMetadata>
</appinfo>
</annotation>
<simpleType>
        <restriction base="string">
                <maxLength value="10"/>
        </restriction>
</simpleType>
</element>
```

型が子ビジネス・オブジェクトの属性のアプリケーション固有情報

子ビジネス・オブジェクト (非単純属性) を参照する属性には、2 つのアプリケーション固有情報パラメーターが使用されます。このアプリケーション固有情報を設定する場合は、下記の表に示すパラメーターを指定します。

表 6. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報

パラメーター	型	説明	デフォルト値	サポートされる双方向変換
KeepRelationship	Boolean	true の場合、このパラメーターは Update 操作中に子ビジネス・オブジェクトを削除しないようにします。	なし	いいえ

表 6. 型が子ビジネス・オブジェクトの属性のアプリケーション固有情報 (続き)

パラメーター	型	説明	デフォルト値	サポートされる双方向変換
Ownership	Boolean	このパラメーターは、子ビジネス・オブジェクトが親によって所有されることを指定します。true の場合、子ビジネス・オブジェクトに対する Create、Update、および Delete 操作が許可されます。false の場合、これらの更新のうちのどれも、子ビジネス・オブジェクトには適用できません。親が作成されると、データベース内で関係の整合性が保持されるように、子が存在するかどうかが検証されます。	なし	いいえ

ビジネス・オブジェクトの .xsd ファイル内の OWNERSHIP の例

```
<element minOccurs="0" name="addressObj" type="bons0:OutboundRtasserAddress"
 maxOccurs="unbounded">
 <annotation>
   <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
  <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:Ownership>true</jdbcasi:Ownership>
  </jdbcasi:JDBCAttributeTypeMetadata>
   </appinfo>
 </annotation>
</element>


<element minOccurs="0" name="custinfoObj" type="bons1:OutboundRtasserCustinfo"
maxOccurs="1">
 <annotation>
  <appinfo source="http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
   <jdbcasi:JDBCAttributeTypeMetadata xmlns:jdbcasi=
"http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/metadata">
    <jdbcasi:Ownership>false</jdbcasi:Ownership>
   </jdbcasi:JDBCAttributeTypeMetadata>
  </appinfo>
 </annotation>
</element>
```

エンタープライズ・サービス・ディスカバリー

WebSphere Adapter for JDBC のエンタープライズ・サービス・ディスカバリー・ウィザードを使用して、データベース内のオブジェクトをディスカバーし、選択したオブジェクトからビジネス・オブジェクトを生成します。また、エンタープライズ・サービス・ディスカバリー・サポートは、アダプターが Service Component Architecture (SCA) コンポーネントとして稼働できるようにするサービス成果物も生成します。

エンタープライズ・サービス・ディスカバリー・ウィザードを使用すると、データベース内の 3 種類のオブジェクトからビジネス・オブジェクトを生成できます。最初の種類のオブジェクトには、テーブルおよびビューが含まれ、2 番目の種類に

は、ストアード・プロシージャーおよび同義語/ニックネームが含まれます。 3 番目の種類のオブジェクトは、ユーザー指定 select ステートメントから生成されたクエリー・ビジネス・オブジェクトです。

データベース内のオブジェクトの最初および 2 番目の種類の場合、エンタープライズ・サービス・ディスカバリーでデータベース内のすべてのスキーマのリストを生成することによってオブジェクトをディスカバーできます。各スキーマ内には、テーブル、ビュー、ストアード・プロシージャー、および同義語/ニックネームのリストが含まれています。それらを選択して、対応するビジネス・オブジェクトを生成するようにエンタープライズ・サービス・ディスカバリー・ウィザードに要求できます。このウィザードは、選択されたオブジェクトのメタデータを分析し、ビジネス・オブジェクト内に属性を生成します。属性は、選択されたデータベース・オブジェクトの列に基づいて生成されます。

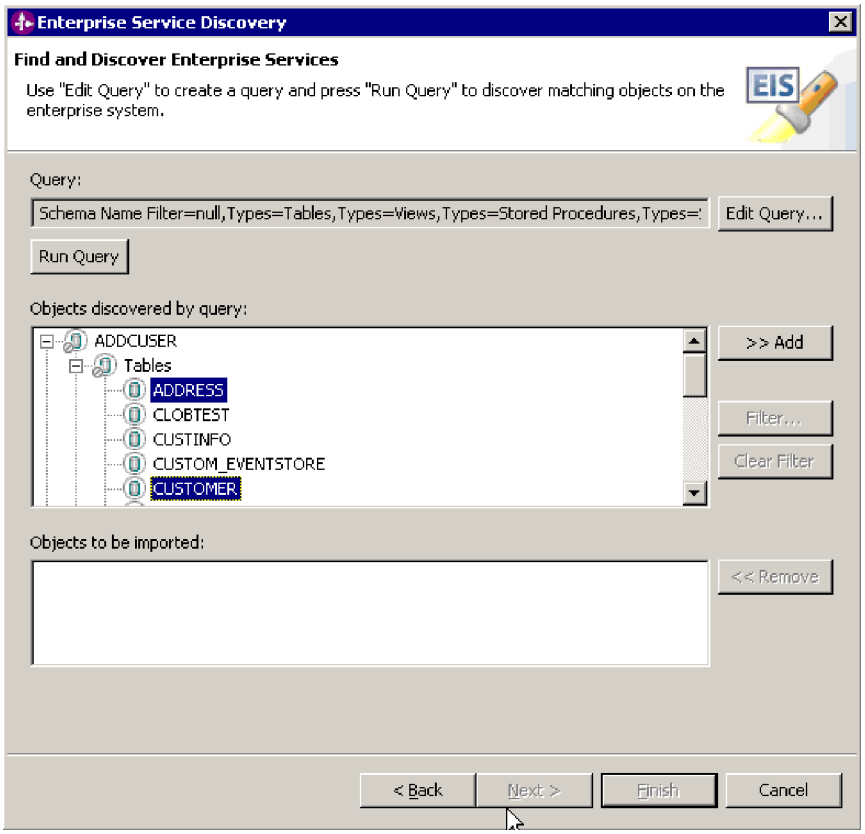


図 8. インポートするビジネス・オブジェクトの選択

オブジェクトの 3 番目の種類であるクエリー・ビジネス・オブジェクトの場合、select ステートメントを 1 つ指定します。ビジネス・オブジェクト名と、where 文節のすべてのパラメーターの型およびダミー値を指定する必要があります。エンタープライズ・サービス・ディスカバリーは、このユーザー指定 select ステートメントを実行して返された結果セットのメタデータを分析し、列とその型のリストを取得します。エンタープライズ・サービス・ディスカバリーは、この情報を取得した後にこのユーザー指定 select ステートメントに対するクエリー・ビジネス・オブジェクトを 1 つ生成します。

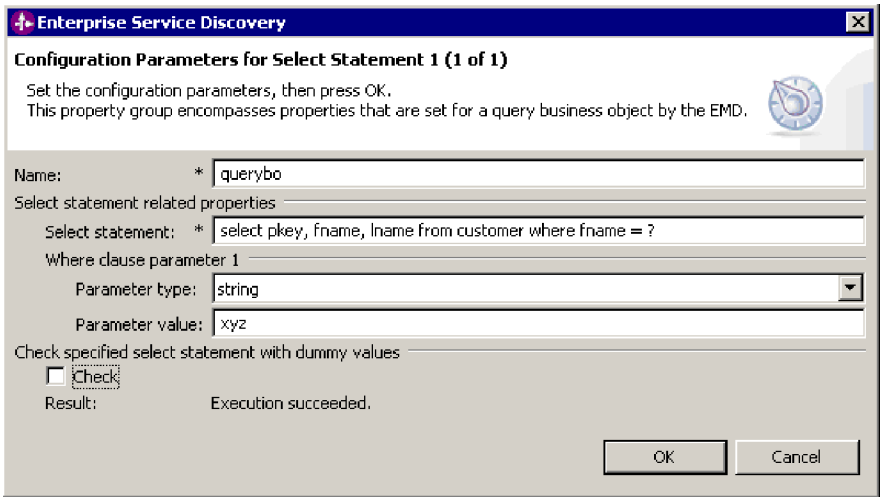


図 9. 「select ステートメントの構成パラメーター (Configuration Parameters for Select Statement)」ウィンドウ

その機能をまとめるために、エンタープライズ・サービス・ディスカバリー・ウィザードでは以下の処理を行います。

- データベース・オブジェクトに対応するビジネス・オブジェクトを生成する。
- データベース・オブジェクト内の列に対応するビジネス・オブジェクトの属性を生成する。
- ビジネス・オブジェクトに関するアプリケーション固有情報を設定する。
- インポート、エクスポート、および Web サービス記述言語 (WSDL) ファイルの生成に使用するサービス記述 (Inbound および Outbound) を提供する。

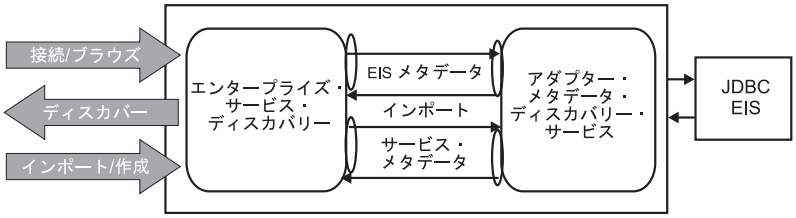


図 10. エンタープライズ・サービス・ディスカバリーの機能の概要

システム機能のディスカバリー

テーブル、ビュー、ストアード・プロシージャー、および同義語/ニックネームに基づくビジネス・オブジェクト生成の場合、エンタープライズ・サービス・ディスカバリーはデータベースを分析してスキーマを識別します。ウィザードは、次にデータベースからすべてのオブジェクトのリストを作成し、それをツリー構造として表示します。ディスカバーされたデータベース・オブジェクトを選択すると、ビジネス・オブジェクトを生成できます。ユーザー指定 select ステートメントからクエリー・ビジネス・オブジェクトを生成することもできます。

スキーマは、ツリー内の最上位ノードとして表示されます。スキーマ・ノードは、生成用には選択できません。各スキーマの下には、Tables、Views、Stored Procedures、および Synonyms/Nicknames とラベル付けされたノードが存在します。これらのノードは生成用に選択できません。これらのノードの下にリストされるオブジェクトは、特定のスキーマに対するテーブル、ビュー、ストアード・プロシージャー、および同義語/ニックネームの名前です。これらのノードは生成用に選択可能とマークされます。特定のスキーマにテーブル、ビュー、ストアード・プロシージャー、または同義語がない場合は何もリストされません。

スキーマ、ノード・タイプ、およびデータベース・オブジェクトのフィルター操作

ツリーが生成および表示される前に、ツリーに表示させるスキーマまたは各スキーマの下に表示されるノード・タイプのリストを絞り込む場合、フィルター・プロパティーを指定できます。指定しないと、すべてのスキーマおよびノード・タイプが表示されます。 SchemaNameFilter プロパティーはスキーマのフィルター操作に使用し、Types プロパティーはノード・タイプのフィルター操作に使用します。さらに、オブジェクトを選択するときにプロパティー DefineASI、QueryBO、および QueryBOCount を使用できます。これらのプロパティーについては、『参照』セクションの『フィルターおよびノードのプロパティー』で説明しています。

テーブル、ビュー、ストアード・プロシージャー、および同義語/ニックネームのいずれかのノードを選択して「**フィルター (Filter)**」をクリックすると、エンタープライズ・サービス・ディスカバリー・ウィザードが ObjectNameFilter を照会します。ObjectNameFilter プロパティーを使用して、表示するデータベース・オブジェクトのリストをフィルタリングできます。『参照』セクションの『フィルターおよびノードのプロパティー』で、ObjectNameFilter プロパティーについて説明しています。

ユーザー指定 select ステートメントからのクエリー・ビジネス・オブジェクトの生成

ユーザー指定 select ステートメントからクエリー・ビジネス・オブジェクトを生成するには、QueryBO プロパティーを選択し、QueryBOCount を同時に生成可能なクエリー・ビジネス・オブジェクトの最大数に設定します。その後、照会を実行してツリーを表示させると、いくつかのスキーマ最上位ノードとともに新しい最上位「照会ステートメント (Query Statements)」ノードが表示されます。照会ステートメントは生成するために選択できません。このノードを展開すると、QueryBOCount に相当する数までのノードについて「Select ステートメント 1 (Select Statement 1)」および「Select ステートメント 2 (Select Statement 2)」などのラベルが付いたノードが表示されます。これらのノードは、クエリー・ビジネス・オブジェクトを生成するために選択できます。

オブジェクトの選択および生成

ビジネス・オブジェクトを生成するには、データベース・オブジェクト・ノードを選択します。そうすると、エンタープライズ・サービス・ディスカバリー・ウィザードで、選択したノードのオブジェクトのビジネス・オブジェクトが生成されます。

複数のデータベース・オブジェクト・ノードを選択できます。Define ASI プロパティーを true に設定したときに選択したテーブルまたはビュー・タイプのノードごと

に、エンタープライズ・サービス・ディスカバリー・ウィザードは、StatusColumnName および StatusValue のアプリケーション固有情報に対するオブジェクト・レベル・パラメーターと、動詞アプリケーション固有情報の関連を要求します。

StatusColumnName の場合、ビジネス・オブジェクト属性のリストが表示され、そのリストから 1 つを選択できます。StatusValue の値は入力する必要があります。これらの値は、ビジネス・オブジェクト・レベルのアプリケーション固有情報に設定されます。

ビジネス・オブジェクトへのストアード・プロシージャーの関連付け

さらに、ストアード・プロシージャーのビジネス・オブジェクトへの関連付けを選択することもできます。サポートされるすべての StoredProcedureType 値のリストが表示され、構成対象を選択できます。詳細については、『ストアード・プロシージャー定義』の StoredProcedureType を参照してください。値をいくつか選択すると、対応する動詞アプリケーション固有情報プロパティー・グループがエンタープライズ・サービス・ディスカバリー・ウィザードに表示され、構成できるようになります。

例えば、BeforeCreateSP タイプおよび AfterUpdateSP タイプのみを構成する場合は、StoredProcedureType に対するこれらの動詞アプリケーション固有情報の値を選択します。すると、それぞれの値に対して 1 つずつ、合計 2 つの動詞アプリケーション固有情報プロパティー・グループが表示されます。

各プロパティー・グループには、適切なスキーマ名およびストアード・プロシージャー名の選択に使用するプロパティー Schema および StoredProcedure があります。*Schema プロパティー* には、データベースで使用可能なスキーマがリストされます。SPSchemaNameFilter プロパティーを使用すると、スキーマをフィルターに掛けることができます。スキーマのリストから少なくとも 1 つのスキーマを選択する必要があります。選択すると、そのスキーマで使用可能なストアード・プロシージャーのリストが *StoredProcedure プロパティー* に表示されます。

ストアード・プロシージャーの選択を容易にするために、各 StoredProcedure プロパティーに SPNameFilter プロパティーが提供されており、ストアード・プロシージャーのリストを絞り込むことができます。フィルター機構には「先頭文字 (Start With)」と「ワイルドカード (Wildcard)」の 2 種類が用意されています。サポートされるワイルドカードは * および ? です。SPNameFilter プロパティーを設定しない場合は、スキーマの下のすべてのストアード・プロシージャーが表示されます。

表 7. ストアード・プロシージャーのフィルター・プロパティー

プロパティー	型	説明
SPSchemaNameFilter	String	スキーマをフィルターに掛けるために使用するテキスト。Schema プロパティーを設定する必要があります。SPSchemaNameFilter プロパティーを設定しない場合は、データベースのすべてのスキーマが表示されます。

表 7. ストアード・プロシージャーのフィルター・プロパティー (続き)

プロパティー	型	説明
SPNameFilter	String	スキーマのストアード・プロシージャーをフィルターに掛けるために使用するテキスト。 StoredProcedure プロパティーを設定する必要があります。フィルター機構としては「先頭文字 (Start With)」と「ワイルドカード (Wildcard)」の 2 種類がサポートされます。SPNameFilter プロパティーを設定しない場合は、スキーマの下のすべてのストアード・プロシージャーが表示されます。

表 8. ワイルドカード・フィルター機構の例

フィルター基準	A*B?d
一致するストリング	**A**dsf**B**n**D**
一致するストリング	**A**sdfsdfsdf**B**w**d**
一致するストリング	**A**234dsf**B**6**d**

割り当てられた各ストアード・プロシージャーに対して、ストアード・プロシージャーの入出力パラメーターのリストが示されます。各パラメーターは、ビジネス・オブジェクトの属性のリストを持っています。パラメーターのタイプは、プロパティーの記述にリストされます (IP、OP、IO など)。ストアード・プロシージャー・パラメーターごとに、ビジネス・オブジェクト属性を 1 つ選択できます。ビジネス・オブジェクトの動詞アプリケーション固有情報 (CreateSP や UpdateSP など) に、ストアード・プロシージャーが 1 つ割り当てられているすべてのストアード・プロシージャー・タイプが追加されます。詳細については、『動詞のアプリケーション固有情報』を参照してください。

選択した任意のアプリケーション固有情報を StoredProcedureType プロパティーから除去できます。対応する動詞アプリケーション固有情報プロパティー・グループもすべて削除されます。

テーブルまたはビューから生成されたビジネス・オブジェクトにストアード・プロシージャーを関連付けるときに、ストアード・プロシージャーが関数である場合は、そのストアード・プロシージャーから値が返されます。 1 つの ReturnValue アプリケーション固有情報の値が動詞アプリケーション固有情報に追加されます。このアプリケーション固有情報が存在する場合は、関数によって値が返されるため、関数呼び出しであってプロシージャー呼び出しではないことが示されます。

このアプリケーション固有情報の値が RS である場合は、戻り値が結果セットであり、その結果セットがこのビジネス・オブジェクトに対応する N カーディナリティー・コンテナーの作成に使用されることが示されます。ただし、このアプリケーション固有情報の値がビジネス・オブジェクト属性名である場合は、戻り値がビジネス・オブジェクトの特定の属性に割り当てられます。

このアプリケーション固有情報の値が別の子ビジネス・オブジェクトである場合、アダプター・ランタイムはエラーを返します。要約すると、戻り値が単純データ型である場合はエンタープライズ・サービス・ディスカバリーによって 1 つのビジネス・オブジェクト属性を戻り値にバインドでき、このアプリケーション固有情報の値がそのビジネス・オブジェクト属性の名前に設定されます。しかし、戻り値が結

果セットである場合、エンタープライズ・サービス・ディスカバリーはこのアプリケーション固有情報の値を RS に設定します。

注: Oracle データベースの場合、結果セットは戻り値としてではなく、出力パラメーターとして返さなければなりません。出力パラメーターのタイプは RS に設定され、このパラメーターが結果セットを返すために使用されることを示します。

複合データ型

ストアード・プロシージャーからビジネス・オブジェクト (ストアード・プロシージャー・ビジネス・オブジェクトと呼びます) を生成するには、ストアード・プロシージャーの入力パラメーターにダミー値を設定する必要があります。それにより、エンタープライズ・サービス・ディスカバリーがこのストアード・プロシージャーを実行して返される結果セットの最大数を取得し、これらの結果セットのメタデータを取得して子ビジネス・オブジェクトを適宜生成できます。

選択されたストアード・プロシージャーが入出力パラメーターまたは戻り値として複合データ型 (Struct、Array、ResultSet など) を取る場合、必要な情報がウィザードから要求されます。パラメーターが複合型の場合は、プロパティー SPComplexParameterType から ResultSet/Struct/Array を選択してこのパラメーターのデータ型を指定する必要があります。Struct または Array の場合は、対応するユーザー定義型名の名前をプロパティー SPComplexParameterTypeName に指定する必要があります。

例えば、Struct_TEMP という名前の 1 つの Struct オブジェクトをデータベースに作成し、1 つの入力パラメーターとして型を設定する場合は、このプロパティーの値を Struct_TEMP に設定する必要があります。エンタープライズ・サービス・ディスカバリーは、この型名の取得によってのみメタデータを取得し、対応する子ビジネス・オブジェクトを生成できます。ストアード・プロシージャーが ResultSet を返す場合は、このストアード・プロシージャーから返される結果セットの数をプロパティー MaxNumberOfResultSets に設定する必要があります。この値は、ランタイムによって処理される、返される結果セットの最大数を表します。

ストアード・プロシージャー・ビジネス・オブジェクトの場合、エンタープライズ・サービス・ディスカバリーは、ネストされた構造体および配列をサポートし、任意の数の層にわたるネストされた階層をサポートできます。エンタープライズ・サービス・ディスカバリーは、これらのネストされたすべての構造体および配列に対して対応する子ビジネス・オブジェクトを生成できます。Adapter for JDBC ランタイムも、ネストされた構造体および配列をサポートできます。

表 9. ストアード・プロシージャー・ビジネス・オブジェクトの複合データ型プロパティー

プロパティー	型	説明
SPComplexParameter Type	String	可能な値は Struct、Array、または ResultSet です。
SPComplexParameterTypeName	String	ユーザー定義型の名前。このプロパティーは、SPComplexParameterType の値が Struct または Array の場合に必要です。

表 9. ストアード・プロシージャー・ビジネス・オブジェクトの複合データ型プロパティー (続き)

プロパティー	型	説明
MaxNumberOfResultSets	Integer	Adapter for JDBC ランタイムによって処理される、返される結果セットの最大数。エンタープライズ・サービス・ディスカバリーも、この数のビジネス・オブジェクトを作成します。

QueryBO プロパティーおよびユーザー指定 select ステートメント

QueryBO プロパティーが選択されている場合、エンタープライズ・サービス・ディスカバリー・ウィザードはいくつかの情報を要求します。QueryBOName は、このクエリー・ビジネス・オブジェクトの名前を指定するために使用します。SelectStatement プロパティーは、1 つの select ステートメントを入力するために使用します。select ステートメントを入力すると、エンタープライズ・サービス・ディスカバリーがその where 文節にあるパラメーターの数を自動的に検出します。パラメーターが 1 つ検出されると、対応するプロパティー・グループが 1 つ表示され、そのパラメーターのデータ型およびダミー値を指定できます。 select ステートメントの where 文節からパラメーターを 1 つ除去すると、エンタープライズ・サービス・ディスカバリーがこの除去を検出してすべてのプロパティー・グループを除去し、残りのパラメーターに必要なプロパティー・グループを再作成します。

このようなプロパティー・グループは、いずれも 2 つのプロパティー WhereParameterType および WhereParameterValue を含みます。 WhereParameterType は、サポートされるすべてのデータ型のリストを持ちます。1 つのデータ型を選択して、データベース内のこのパラメーターの実際のデータ型を指定できます。このパラメーターのデータ型を選択した後、WhereParameterValue プロパティーにダミー値を 1 つ入力できます。この値は、select ステートメントの実行に使用されます。

例えば、以下の select ステートメントの場合、

```
select * from customer where id=? and age=?
```

エンタープライズ・サービス・ディスカバリーは、その where 文節にあるパラメーターの数が 2 であることを認識します。次に、ウィザードはプロパティー・グループを 2 つ表示し、それら 2 つのパラメーターに対するデータ型およびダミー値をユーザーが設定できるようにします。第 1 パラメーターの場合は WhereParameterType プロパティーを `string` に設定し、WhereParameterValue を `Mike` に設定します。第 2 パラメーターの場合は WhereParameterType プロパティーを `int` に設定し、WhereParameterValue を `27` に設定します。エンタープライズ・サービス・ディスカバリーは、この select ステートメントを実行して返された結果セットを取得します。

結果セットを取得した後、エンタープライズ・サービス・ディスカバリーは、メタデータを分析してすべての列の列名および列の型を取得します。返された結果セットの列ごとに、ウィザードは対応する属性を 1 つ QueryBO に生成します。また、where 文節のパラメーターごとに、エンタープライズ・サービス・ディスカバリーは対応する属性を 1 つ QueryBO に生成します。QueryBO は、*jdbcwhereclause* という名前の属性を 1 つ生成します。エンタープライズ・サービス・ディスカバリー

は、この属性のデフォルト値として where 文節を設定します。この属性は、デフォルトの where 文節を置換する 1 つの動的 where 文節をランタイムに設定するために使用されます。

表 10. クエリー・ビジネス・オブジェクトのプロパティー

プロパティー	型	説明
QueryBOName	String	クエリー・ビジネス・オブジェクトの名前。
SelectStatement	String	ユーザー指定 select ステートメント。
WhereParameterType	String	1 つのパラメーターのデータ型。サポートされるデータ型のリストから選択されます。
WhereParameterValue	String	1 つのパラメーターのダミー値。エンタープライズ・サービス・ディスカバリーは、この値を使用して select ステートメントを実行します。

ビジネス・オブジェクト内部のすべてのコンテンツ・データで、双方向変換がサポートされます。ただし、QueryBO の jdbcwhereclause 属性を除きます。 where 文節内のストリング定数のみに双方向言語サポートが提供されます。

重要: jdbcwhereclause 属性の双方向変換が正しく行われるようにするためには、BiDiContext を属性レベルのアプリケーション固有情報として追加し、その特殊フォーマットを SAP WHERE CLAUSE に設定します。

双方向言語サポートは、SelectStatement プロパティー内のストリング定数のみで提供されます。 SQL ステートメントは複雑である場合があり、双方向言語サポートを提供すると、破損する可能性があるためです。

データベース・オブジェクトを選択した後、メタデータ選択プロパティーの値を設定する必要があります。エンタープライズ・サービス・ディスカバリー・ウィザードは、これらのプロパティーを照会します。これらのプロパティーについて詳しくは、『参照』セクションの『メタデータ選択プロパティー』を参照してください。

データ記述

データ記述は、アダプターが Service Component Architecture (SCA) コンポーネントとして稼働できるようにするためにエンタープライズ・サービス・ディスカバリー処理によって生成されるサービス構成の一部です。

データ記述には、クライアント・アプリケーションとアダプターの間で実行時に引き渡しされる、アダプターのビジネス・オブジェクトの構造と内容の定義が含まれます。データ記述を使用すると、クライアント・アプリケーションが要求に対応する適切なデータ・オブジェクトを作成し、応答として戻されたデータ・オブジェクトを解釈することができます。データベース・コンポーネントから生成されたデータ記述は、XML スキーマとして表されます。

- ビジネス・オブジェクトは、複合タイプの定義にマップします。
- ビジネス・オブジェクトの属性は、エレメント・タイプの定義にマップします。
- ビジネス・オブジェクトのアプリケーション固有の情報は、複合タイプでの注釈に含まれます。

- ビジネス・オブジェクト内の各プロパティーのアプリケーション固有情報は、エレメント・タイプに関する注釈に含まれます。

注: ビジネス・オブジェクトおよび属性レベルのアプリケーション固有のプロパティーのテンプレートが、JDBC アダプターのメタデータ・スキーマに定義されています。このスキーマ・ファイルの名前は JDBCASI.xsd です。生成されたスキーマ・ファイルの注釈には、このテンプレートへの参照が含まれます。

ビジネス・オブジェクトのスキーマおよびアプリケーション固有情報

ビジネス・オブジェクトのスキーマは、選択したデータベース・コンポーネントから作成されます。各コンポーネントは最上位ビジネス・オブジェクトになります。

テーブル、ビュー、ストアード・プロシージャー、および同義語/ニックネームに基づくビジネス・オブジェクトの場合は、エンタープライズ・サービス・ディスカバリー・ウィザードで *PrefixSchemaNameObjectName* の形のビジネス・オブジェクト名が生成されます。ここで、

- *Prefix* はプレフィックスという名前の接続プロパティーで指定された値です。Prefix は必須ではありません。指定しないと、ビジネス・オブジェクト名にプレフィックスが追加されません。
- *SchemaName* は、オブジェクトが属するスキーマの名前です。
- *ObjectName* はテーブル、ビュー、ストアード・プロシージャー、または同義語/ニックネームの名前です。

ビジネス・オブジェクト名ではグローバル化文字がサポートされます。

クエリー・ビジネス・オブジェクトの場合は、エンタープライズ・サービス・ディスカバリー・ウィザードで *PrefixQueryBOName* の形のビジネス・オブジェクト名が生成されます。ここで、

- *Prefix* はプレフィックスという名前の接続プロパティーで指定された値です。Prefix は必須ではありません。指定しないと、ビジネス・オブジェクト名にプレフィックスが追加されません。
- *QueryBOName* は、QueryBOName という名前の構成プロパティーで指定された値です。

ビジネス・オブジェクト名ではグローバル化文字がサポートされます。

テーブルまたはビューに基づくビジネス・オブジェクトの場合は、エンタープライズ・サービス・ディスカバリー・ウィザードでアプリケーション固有情報属性の TableName が *schemaname.tablename* の形の値に設定されます。また、ビジネス・オブジェクト・レベルのアプリケーション固有情報が、『テーブルまたはビューに基づくビジネス・オブジェクトの場合のビジネス・オブジェクトのアプリケーション固有情報 (ASI)』の表にリストされているように設定されます。ビジネス・オブジェクトに選択した操作が設定されます。ビジネス・オブジェクトの属性は列に基づいて作成されます。

表 11. テーブルまたはビューに基づくビジネス・オブジェクトの場合のビジネス・オブジェクトのアプリケーション固有情報 (ASI)

ビジネス・オブジェクト ASI	エンタープライズ・サービス・ディスカバリー・ウィザードによって設定	追加情報
TableName	はい	オブジェクトの実際の名前に設定します。
ステータス列名	はい	オブジェクトの選択時に指定します。
ステータス値	はい	オブジェクトの選択時に指定します。

ストアード・プロシージャーに基づくビジネス・オブジェクトの場合、エンタープライズ・サービス・ディスカバリー・ウィザードでビジネス・オブジェクト・レベルのアプリケーション固有情報 SPName が *schemaname.spname* の形の値に設定されます。また、ビジネス・オブジェクト・レベルのアプリケーション固有情報が、『ストアード・プロシージャーに基づくビジネス・オブジェクトの場合のビジネス・オブジェクトのアプリケーション固有情報 (ASI)』の表にリストされているように設定されます。ビジネス・オブジェクトの属性は、ストアード・プロシージャーの入出力パラメーターに基づいて作成されます。ストアード・プロシージャーが 1 つの戻り値を持つ場合は、対応するビジネス・オブジェクト属性が作成されます。戻り値または入出力パラメーターに複合データ型がある場合、ウィザードによってそれらの子ビジネス・オブジェクトが作成されます。

エンタープライズ・サービス・ディスカバリーは、ネストされた構造体および配列をサポートできます。返される ResultSet からこれらの子ビジネス・オブジェクトが生成される場合、それらの子ビジネス・オブジェクトの名前の形式は *PrefixSchemaNameSPNameRetRS#* になります。例えば、あるストアード・プロシージャーが 2 つの結果セットを返す場合、エンタープライズ・サービス・ディスカバリーはそれらに対応する 2 つの子ビジネス・オブジェクトを作成します。名前は *PrefixSchemaNameSPNameRetRS1* および *PrefixSchemaNameSPNameRetRS2* です。

子ビジネス・オブジェクトが、複合データ型の ResultSet、Struct、または Array を持つ入出力パラメーターから生成される場合、これらの子ビジネス・オブジェクト名の形式は *PrefixSchemaNameSPNameParameterName* になります。ネストされた構造体および配列に対応する子ビジネス・オブジェクトの場合、ビジネス・オブジェクト名の形式は *PrefixSchemaNameSPNameParameterNameColumnName* になります。

表 12. ストアード・プロシージャーに基づくビジネス・オブジェクトの場合のビジネス・オブジェクトのアプリケーション固有情報 (ASI)

ビジネス・オブジェクト ASI	エンタープライズ・サービス・ディスカバリー・ウィザードによって設定	追加情報
SPName	はい	ストアード・プロシージャーの実際の名前に設定します。
ResultSet	はい	ストアード・プロシージャーが 1 つ以上の ResultSet を返す場合は `true` に設定し、そうでなければ `false` に設定します。

表 12. ストアード・プロシージャーに基づくビジネス・オブジェクトの場合のビジネス・オブジェクトのアプリケーション固有情報 (ASI) (続き)

ビジネス・オブジェクト ASI	エンタープライズ・サービス・ディスカバリー・ウィザードによって設定	追加情報
MaxNumofRetRS	はい	アダプター・ランタイムによって処理される返される結果セットの最大数。
ReturnValue	はい	ストアード・プロシージャーが戻り値を持つ場合は、ビジネス・オブジェクト属性名に設定します。戻り値が単純データ型の場合は、属性も単純データ型です。戻り値が ResultSet の場合、この属性は子ビジネス・オブジェクトを指します。

クエリー・ビジネス・オブジェクトの場合、SelectStatement という名前の 1 つのビジネス・オブジェクト・レベルのアプリケーション固有情報が追加されます。値は完全な select ステートメントです。

表 13. クエリー・ビジネス・オブジェクトのビジネス・オブジェクト・アプリケーション固有情報 (ASI)

ビジネス・オブジェクト ASI	エンタープライズ・サービス・ディスカバリー・ウィザードによって設定	追加情報
SelectStatement	はい	オブジェクトの選択時に指定します。

また、エンタープライズ・サービス・ディスカバリー・ウィザードでは、すべてのビジネス・オブジェクトが最上位であるため、すべてのビジネス・オブジェクトのビジネス・グラフも生成されます。ビジネス・グラフの名前は、ビジネス・オブジェクト名に「BG」が付いたものです。例えば、JDBCSchema1Customer という名前のビジネス・オブジェクトのビジネス・グラフの名前は JDBCSchema1CustomerBG となります。ビジネス・オブジェクトに設定された操作はビジネス・グラフにも設定されます。

エンタープライズ・サービス・ディスカバリーは、ストアード・プロシージャー・ビジネス・オブジェクトを生成する場合に、必要であれば ResultSet、Struct、Array などの子ビジネス・オブジェクトを作成します。テーブル・ビジネス・オブジェクト間の親子関係は、Business Object Editor を使用して手動で作成します。

エンタープライズ・サービス・ディスカバリーは、同義語がストアード・プロシージャーのものであっても、同義語/ニックネームに基づくビジネス・オブジェクトもテーブルおよびビューに基づくオブジェクトのように処理します。

ビジネス・オブジェクトの属性およびアプリケーション固有情報

ビジネス・オブジェクトの属性は、データベース・オブジェクトの列リストから作成されます。エンタープライズ・サービス・ディスカバリー・ウィザードで、列名に属性名が設定されます。属性名ではグローバル化文字がサポートされます。アダプターは、属性の名前、タイプ、およびアプリケーション固有情報を追加します。

JDBC メタデータによって戻されるタイプは、『JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ』の表にリストされているとおり、ビジネス・オブジェクト属性タイプにマップされます。リストされている JDBC タイプのみがアダプターでサポートされます。リストされていないタイプの列は、ビジネス・オブジェクトに追加されません。例えば、「テーブル *yyyy* の列名 *xxxx* のタイプはサポートされていません。ビジネス・オブジェクトには追加されません」という情報メッセージが作成されます。

表 14. JDBC メタデータ列タイプとビジネス・オブジェクト属性タイプ

JDBC メタデータの列タイプ	**ビジネス・オブジェクト属性のタイプ**
BIT	BOOLEAN
CHAR LONGVARCHAR VARCHAR	STRING
INTEGER NUMERIC SMALLINT TINYINT BIGINT	INT
TIME TIMESTAMP DATE	STRING
DECIMAL	STRING
DOUBLE FLOAT	DOUBLE
REAL	FLOAT
BLOB	HEXBINARY
CLOB	STRING
BINARY VARBINARY LONGBINARY	HEXBINARY

『属性情報』の表に、エンタープライズ・サービス・ディスカバリー・ウィザードによって設定される属性情報をリストします。

表 15. 属性情報

属性情報	エンタープライズ・サービス・ディスカバリーによって設定	追加情報
Cardinality	いいえ	単一カーディナリティー関係および複数カーディナリティー関係の両方で、親ビジネス・オブジェクトと子ビジネス・オブジェクトの間の関係は、その関係を保管するビジネス・オブジェクトのキー属性に含まれるアプリケーション固有情報によって記述されます。
MinOccurs/MaxOccurs	はい	列が基本キーではなくかつ NULL が不可能な場合、この属性は必須で、値は 1 以上に設定されます。
Name	はい	属性の名前。これは双方向言語の場合に使用可能化されます。
Type	はい	『JDBC メタデータ列とビジネス・オブジェクト属性タイプ』の表のとおりに設定します。

エンタープライズ・サービス・ディスカバリー・ウィザードは、ビジネス・オブジェクトの属性に関するアプリケーション固有情報 (ASI) を『属性に関するアプリケーション固有情報』の表で示したとおりに設定します。詳細については、『属性に関するアプリケーション固有情報』を参照してください。

表 16. 属性に関するアプリケーション固有情報

属性 ASI	エンタープライズ・サービス・ディスカバリーによって設定	追加情報	サポートされる双方向変換
BLOB	はい	列のデータ型が BLOB の場合は `true` に設定します。	いいえ
ByteArray	いいえ	バイナリー・データ型の列の場合は `true` に設定します。true の場合、アダプターはデータベースでバイナリー・データを読み書きします。アダプターは、ビジネス・オブジェクトにバイナリー・データを設定します。属性の型は hexBinary です。	いいえ
ChildBOType	はい	属性が複合データ型の場合は、型を Struct、Array、または ResultSet として指定します。	いいえ

表 16. 属性に関するアプリケーション固有情報 (続き)

属性 ASI	エンタープライズ・サービス・ディスカバリーによって設定	追加情報	サポートされる双方向変換
ChildBOTypeName	はい	ChildBOType の値が Struct または Array の場合、この値はユーザー定義型の名前を表します。	いいえ
CLOB	はい	列のデータ型が CLOB の場合は `true` に設定します。	いいえ
ColumnName	はい	実際の列名に設定します。	はい
DateType	はい	可能な値は Date、Time、または TimeStamp のみです。	いいえ
FixedChar	いいえ	ビジネス・オブジェクト .xsd ファイルでは手動で更新する必要があります。テキスト・モード、または WebSphere Integration Developer の Business Object Editor のいずれかを使用してファイルを編集します。ファイルを更新したら、検証エラーがないことを確認してください。『単純属性のアプリケーション固有情報』セクション内の .xsd ファイルの FixedChar の例を参照してください。	いいえ
ForeignKey	いいえ	外部キー属性では、親子関係が親に保管される場合、このパラメーターの値には、子ビジネス・オブジェクトのタイプと、外部キーとして使用される子ビジネス・オブジェクト内の属性の名前の両方が含まれます。関係が子に保管される場合は、値には、外部キーとして使用される親ビジネス・オブジェクト内の属性の名前のみが含まれます。	いいえ
OrderBy	いいえ	値が指定されている場合、属性が子ビジネス・オブジェクト内に存在するのであれば、アダプターは、検索項目の ORDER BY 文節でその属性の値を使用します。	いいえ
PrimaryKey	はい	列が基本キーの場合、PrimaryKey は `true` に設定されます。	いいえ
SPParameterType	はい	属性がストアード・プロシージャー・パラメーターに関連付けられている場合は、この値を実パラメーター・タイプ (IP、OP、または IO) に設定します。	いいえ
UID	いいえ	属性がビジネス・オブジェクトの固有 ID を必要とする場合に使用されます。シーケンスと ID 列の生成がサポートされます。	いいえ

表 17. 子ビジネス・オブジェクト型の属性の属性アプリケーション固有情報

属性 ASI	エンタープライズ・サービス・ディスカバリーによって設定	追加情報	サポートされる双方向変換
KeepRelationship	いいえ	パラメーターが true に設定されている場合、Update 操作中に子ビジネス・オブジェクトを削除しないようにします。	いいえ
Ownership	いいえ	パラメーターが true に設定されている場合、子ビジネス・オブジェクトに対する Create、Update、および Delete 操作が許可されます。	いいえ

ビジネス・オブジェクトにストアード・プロシージャーを追加する場合、動詞アプリケーション固有情報 (ASI) は『動詞のアプリケーション固有情報』の表に指定されているとおりに設定されます。有効なストアード・プロシージャー・タイプについては、『動詞のアプリケーション固有情報』のセクションを参照してください。

表 18. 動詞のアプリケーション固有情報

動詞 ASI またはストアード・プロシージャーのパラメーター・エレメント	エンタープライズ・サービス・ディスカバリーによって設定	追加情報	サポートされる双方向変換
Parameters	はい	ストアード・プロシージャーのパラメーターをリストします。	はい
PropertyName	はい	選択したビジネス・オブジェクト属性の名前に設定します。詳細については、『オブジェクトの選択および生成』を参照してください。	はい
ResultSet	いいえ	ストアード・プロシージャーから ResultSet が返される場合は、ビジネス・オブジェクト定義でこのパラメーターを true に設定します。	いいえ
ReturnValue	はい	ストアード・プロシージャーが戻り値を持つ場合は、このパラメーターを RS またはビジネス・オブジェクト属性の名前に設定します。詳細については、『オブジェクトの選択および生成』を参照してください。	いいえ
StoredProcedure	はい	ストアード・プロシージャー名に設定します。	はい
StoredProcedure Type	はい	タイプのリストから選択します。	いいえ
Type	はい	ストアード・プロシージャー・パラメーターのタイプ (IP、OP、IO、または RS) に設定します。	いいえ

階層ビジネス・オブジェクトの作成

エンタープライズ・サービス・ディスカバリー・ウィザードでは、フラット・テーブルを生成し、ビジネス・オブジェクトを表示します。このウィザードでは、データベースに定義された異なるテーブル間の外部キー制限は使用せずに、関係を自動的に作成します。手動でリンクさせるにはこれらが必要です。ビジネス・オブジェクト定義は、テキスト・モードで更新したり、Business Object Editor を使用して更新したりできます。エンタープライズ・サービス・ディスカバリー・ウィザードでは、ストアード・プロシージャー・ビジネス・オブジェクトの階層を自動的に生成します。

単一または複数カーディナリティー子ビジネス・オブジェクトの .xsd 定義ファイルの例をここに示します。エレメント custInfoObj は単一カーディナリティー子ビジネス・オブジェクトで、addressObj は複数カーディナリティー子ビジネス・オブジェクトです。

単一または複数カーディナリティー子ビジネス・オブジェクトの .xsd ファイルの例

```
<element name="addressObj" minOccurs="1" type="Address:Address"
maxOccurs="unbounded">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>true</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
                        <element name="custInfoObj" minOccurs="0" type=
"CustInfo:CustInfo" maxOccurs="1">
                        <annotation>
                        <appinfo source="WBI">
                        <pasi:JDBCAttributeTypeMetadata xmlns:pasi=
"urn:app:jdbc:asi">
                                <pasi:Ownership>false</pasi:Ownership>
                        </pasi:JDBCAttributeTypeMetadata>
                        </appinfo>
                        </annotation>
                        </element>
```

グローバリゼーションおよび双方向変換

アダプターは、1 バイト文字セットとマルチバイト文字セットをサポートし、メッセージ・テキストを指定された言語で配信できるようにグローバル化されています。アダプターは双方向変換も実行します。双方向変換とは、1 つのファイルに左から右 (ヘブライ語やアラビア語など) と右から左 (URL やファイル・パスなど) の両方の意味内容を含むデータを処理するタスクを指します。

グローバリゼーション

Java 仮想マシン (JVM) 内の Java ランタイム環境は、 Unicode 文字コード・セットでデータを表現します。Unicode には、最もよく知られた文字コード・セット (単一バイトとマルチバイトの両方) の文字エンコードが含まれています。WebSphere Business Integration システムのコンポーネントは Java で記述されています。そのため、WebSphere Business Integration のシステム・コンポーネント間でデータを転送するときに文字を変換する必要はありません。

エラー・メッセージや通知メッセージを適切な言語や個々の国や地域に合った形でログに記録するために、アダプターは稼働先システムのロケールを使用します。

双方向変換

アラビア語やヘブライ語などの言語は右から左に書きますが、テキストには左から右に書かれる部分も埋め込まれるため、双方向スクリプトになります。ソフトウェア・アプリケーションで双方向スクリプトを扱う場合は、その表示と処理のためにさまざまな規格を使用します。WebSphere Process Server および WebSphere Enterprise Service Bus は Windows 標準形式を使用しますが、WebSphere Process Server または WebSphere Enterprise Service Bus とデータを交換するエンタープライズ情報システムは別の形式を使用できます。WebSphere Adapters は、2 つのシステム間でやり取りされる双方向スクリプト・データの変換を行うことによって、トランザクションの両側でデータが正確に処理および表示されるようにします。

双方向形式

WebSphere Process Server および WebSphere Enterprise Service Bus は、ILYNN (暗黙、左から右、オン、オフ、公称) の双方向形式を使用します。これは Windows によって使用される形式です。エンタープライズ情報システムが別の形式を使用する場合、アダプターは、データを WebSphere Process Server または WebSphere Enterprise Service Bus に導入する前に形式を変換します。

双方向形式は 5 つの属性から構成されます。双方向プロパティーを設定する場合は、これらの各属性に値を割り当ててください。属性および設定値を以下の表に示します。

表 19. 双方向形式の属性

文字の位置	目的	値	説明	デフォルト設定
1	スキーマの配列	I または V	暗黙 (Implicit) (論理 (Logical))、または表示 (Visual)	I
2	方向	L R C D	左から右 右から左 コンテキスト LTR コンテキスト RTL	L
3	対称スワッピング	Y または N	対称スワッピングがオンまたはオフ	Y
4	シェーピング	S N I M F B	テキストの形状を指定する (Text is shaped) テキストの形状を指定しない (Text is not shaped) 語頭形 (Initial shaping) 語中形 (Middle shaping) 語尾形 (Final shaping) 独立形 (Isolated shaping)	N
5	数字シェーピング	H C N	ヒンディ語 コンテキスト 公称	N

アダプターは、データを左から右の論理形式に変換してから WebSphere Process Server または WebSphere Enterprise Service Bus に送信します。

双方向プロパティーの使用

複数の双方向プロパティーを使用して、コンテンツ・データとメタデータの両方の変換を制御できます。特別な双方向プロパティーを設定してコンテンツ・データまたはメタデータのいずれかを双方向変換から除外するか、変換中に特別な処理が必要なデータを識別できます。

以下の表で、4 種類の双方向プロパティーについて説明します。

表 20. 双方向プロパティーのタイプ

プロパティー・タイプ	データ変換
EIS	コンテンツ・データ (エンタープライズ情報システムによって送信されるデータ) の形式を制御します。
メタデータ	メタデータ (コンテンツ・データに関する情報を提供するデータ) の形式を制御します。
スキップ	変換から除外するコンテンツまたはメタデータを識別します。
特殊形式	変換処理中に異なる処理を必要とするファイル・パスまたは URL などの特定のテキストを識別する。コンテンツ・データまたはメタデータのいずれかに設定できる。

3 つの領域で双方向変換を制御するプロパティーを設定できます。

- **リソース・アダプター・プロパティー:** これらのプロパティーは、TurnBiDiOff プロパティー (アダプター・インスタンスが双方向変換を実行するかどうかを制御します) などのデフォルト構成設定を格納します。これらのプロパティーを構成するには、サーバーの管理コンソールを使用します。
- **管理 (J2C) 接続ファクトリー・プロパティー:** これらのプロパティーは、エンタープライズ情報システムとの Outbound 接続インスタンスを作成するためにランタイムに使用されます。管理接続ファクトリー・プロパティーは作成後、デプロイメント記述子に格納されます。
- **アクティベーション・スペック・プロパティー:** これらのプロパティーは、メッセージ・エンドポイントに対する Inbound イベント処理構成情報を保持します。エンタープライズ・サービス・ディスカバリーを実行するときに設定するか、サーバーの管理コンソールを使用します。

ビジネス・オブジェクトの注釈

一部のアダプターでは、ビジネス・オブジェクト内部の双方向プロパティーに注釈を付けることができます。特にビジネス・オブジェクトまたはビジネス・オブジェクトの一部の変換を制御する情報を追加するには、これを行います。以下のレベルで注釈を追加するには、Business Object Editor (WebSphere Integration Developer 内のツール) を使用します。

- ビジネス・オブジェクト
- ビジネス・オブジェクトのアプリケーション固有の属性
- ビジネス・オブジェクト属性
- ビジネス・オブジェクト属性のアプリケーション固有の属性

プロパティーのスコープと検索機構

アダプター用の双方向プロパティーに値を設定して、該当する場合にビジネス・オブジェクトに注釈を付けると、アダプターは双方向変換を実行します。実行時には、プロパティー設定の階層の継承と、検索機構に依存するロジックを使用します。

リソース・アダプター内で定義されたプロパティーは階層のトップにありますが、他の領域で定義されたプロパティーまたはビジネス・オブジェクト内で注釈が付けられたプロパティーは下位の階層にあります。このため、例えば、リソース・アダプターの EIS タイプの双方向プロパティーのみに値を設定すると、Inbound (アクティベーション・スペック) トランザクションと Outbound (管理接続ファクトリー) トランザクションのいずれで発生するかにかかわらず、定義された EIS タイプの双方向プロパティーを必要とする変換によってそれらの値が継承および使用されます。

しかし、リソース・アダプターとアクティベーション・スペックの両方の EIS タイプの双方向プロパティーに値を設定した場合、Inbound トランザクションから発生した変換は、アクティベーション・スペックに設定された値を使用します。

処理ロジックでは、変換時に使用する双方向プロパティーの値を、検索機構を使用して検索します。検索機構は、変換が生じるレベルから検索を開始し、適切なプロパティー・タイプを持つ定義済みの値を対象に、階層の上位に向かって検索を進めます。検出された最初の有効値が使用されます。階層の検索は、子から親の方向にのみ進行します。兄弟は検索の対象になりません。

第 5 章 アダプター実装の計画

アダプターをインストールする前に、クラスター化されたサーバー環境を使用することによって、アダプターのパフォーマンスと可用性を向上させる方法を検討します。また、アダプターのインストール、構成、およびデプロイのためのロードマップを見直して、これらのタスクを達成するために実行すべきことの概要を習得することもできます。

クラスター環境での WebSphere Adapters

WebSphere アダプター・エンタープライズ・アーカイブ (EAR) モジュールをクラスター化されたサーバー環境にデプロイすることで、アダプターのパフォーマンスおよび可用性を向上させることができます。EAR モジュール内のアダプター・インスタンスは、統合されているすべてのサーバーで複製されます。

WebSphere Process Server および WebSphere Application Server Network Deployment は、クラスター化された環境をサポートしています。クラスターとは、ワークロードの平衡を取り、高可用性とスケーラビリティーを提供するために、一緒に管理されるサーバー・グループのことです。サーバー・クラスターをセットアップするときには、デプロイメント・マネージャー・プロファイルを作成してください。デプロイメント・マネージャーのサブコンポーネントである HAManager により、アダプター・インスタンスを活動状態にするよう JCA コンテナーに通知されます。JCA コンテナーにより、アダプター・インスタンスのランタイム環境が提供されます。クラスター環境の詳細については、http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/ index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm_cluster_v61.html をご覧ください。

クラスター化された環境では、アダプター・インスタンスにて、Inbound 操作および Outbound 操作の両方を処理することができます。

Inbound 操作の高可用性

Inbound 操作は、エンタープライズ情報システム (EIS) アプリケーションのデータを更新した結果、起動するイベントに基づいています。アダプターは、イベント・リスナーを介して、またはイベント・テーブルをポーリングすることで更新を検出するよう構成されます。その後、アダプターはイベントをそのエンドポイントにパブリッシュします。

クラスター化された環境では、複数のアダプター・インスタンスが同じイベントを検出する場合があります。このシナリオでは、イベント処理の重複またはデータ不信の可能性が高まります。例えば、2 つのアダプター・インスタンスが同一のイベント・タイプ・フィルターを使用して同一のイベント・テーブルを同時にポーリングした場合、片方のアダプター・インスタンスにより、もう片方のアダプター・インスタンスが依存しているデータが変更されたり、アダプター・インスタンスが失敗することがあります。クラスター化された環境では、イベント・リスニング・アダプター・アーキテクチャーのリスクが並行して存在します。

© Copyright IBM Corp. 2005, 2006

この状態を避けるため、Inbound アダプター・インスタンスの HAManager により、singleton の振る舞いが強制されます。すべてのアダプター・インスタンスが開始していたとしても、それらのインスタンスのいずれかにより、イベントが検出され、EIS アプリケーションのタイプごとにエンドポイントにパブリッシュされます。

アダプター・モジュールをクラスターにデプロイすると、JCA コンテナーにより ResourceAdapter Bean の enableHASupport プロパティーが検査されます。enableHASupport プロパティーの値が真である場合、JCA コンテナーにより、すべてのアダプター・インスタンスがポリシー 1 of N を持つ HAManager に登録されます。このポリシーとは、クラスター化されたサーバーのいずれかのみが、このアダプター・インスタンスに対しイベントのポーリング (またはリスニング) を開始することを意味します。クラスター内のその他のアダプター・インスタンスが開始していても、それらのインスタンスは、アクティブなアダプター・インスタンスがイベントの処理を完了するまで、アクティブ・イベントに関して休止のままとなります。ポーリング・スレッドが開始しているサーバーが何らかの理由でシャットダウンした場合は、バックアップ・サーバーのいずれかで稼働しているアダプター・インスタンスが活動状態になります。

Outbound 操作の高可用性

クラスター化された環境では、Outbound 要求の実行に、複数のアダプター・インスタンスが使用可能です。そのため、ご使用の環境に Outbound 要求のために同一の WebSphere Adapter と対話するアプリケーションが複数ある場合、アダプター・モジュールをクラスター化された環境にデプロイすることで、パフォーマンスが向上することがあります。

WebSphere Application Server Network Deployment には、Outbound 処理をアダプター・インスタンス間に分散するワークロード管理機能があります。そのため、クラスター化された環境での Outbound 操作は、単一サーバー環境での Outbound 操作と類似します。つまり、1 つのアダプター・インスタンスにより、一度に 1 つの Outbound 要求のみが処理されます。ワークロード管理について詳しくは、http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/ index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/trun_wlm.htmlを参照してください。

注: アダプター・インスタンスは、クラスター化されたサーバー環境に複製されます。 enableHASupport プロパティーが true に設定されている場合は (デフォルト設定)、複製されたアダプター・インスタンスの一つのみがイベントをアクティブにポーリングし、その他のインスタンスは待機モードをとります。enableHASupport プロパティーが false に設定されている場合は、クラスター・メンバー上の複製されたすべてのアダプター・インスタンスがイベントをアクティブにポーリングします。これにより、イベントが重複する場合があります。単一サーバー環境では、enableHASupport の値を false に変更しないでください。このプロパティーの値の変更については、本書のリソース・アダプター・プロパティーに関するセクションを参照してください。クラスター環境でアダプターの複製がサポートされるかどうかを確認するには、本書のソフトウェア/ハードウェア要件に関するセクションを参照してください。

メインフレーム・データ・アクセス

Adapter for JDBC は、IBM WebSphere Information Integrator Classic Federation for z/OS® を使用するメインフレーム・データ・アクセスをサポートします。この製品は、メインフレーム・データベースへの読み取り/書き込み接続性を備えた Web および分散アプリケーションを提供します。

この製品は、高性能な SQL 駆動アクセスと、メインフレーム・データ・ソースの統合を実現します。好みのインターネット・ツールとアプリケーションを搭載したデスクトップを使用して、メインフレームの主幹業務情報に透過的にアクセスすることができます。

IBM z/OS プラットフォームで Adapter for JDBC with IBM WebSphere Information Integrator Classic Federation を使用する際に問題が発生した場合は、IBM 製品のサポート Web ページ http://www-306.ibm.com/software/integration/wbiadapters/support で、構成要件について説明している技術情報を参照してください。

アダプターをインストール、構成、デプロイするためのロードマップ

ランタイム環境でアダプターを使用できるようにするには、まずアダプターをインストール、構成、およびデプロイする必要があります。これらのタスクの概要を理解すれば、各タスクを達成するのに必要な手順を実行できるようになります。

WebSphere Adapter を正常にインストールしたら、WebSphere Integration Developer を使用して WebSphere Adapter を構成します。その後、エンタープライズ・アーカイブ (EAR) ファイルとして WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイします。次の図にこのタスクのフローを示します。また、図の後ろに示す手順では、各タスクの概要を説明しています。インストールについての詳細な説明については、『*IBM WebSphere Adapters のインストール (Installing IBM WebSphere Adapters)*』を参照してください。アダプターの構成とデプロイについては、アダプターの文書を参照してください。

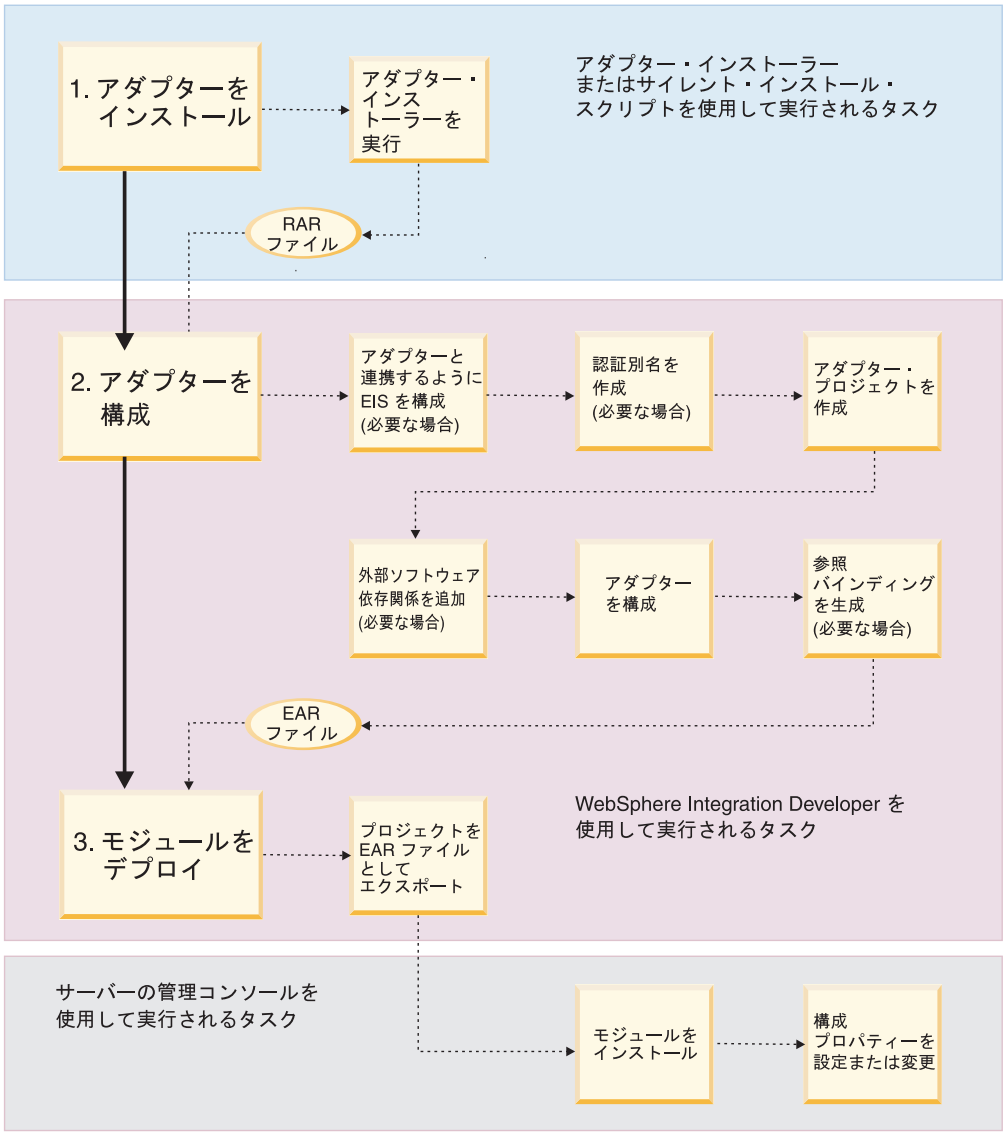


図 11. アダプターをインストール、構成、デプロイするためのロードマップ

1. **アダプターのインストール**
   a. インストーラー (グラフィカル・ユーザー・インターフェース)、またはサイレント・インストールを実行するスクリプトを使用します。どちらのメソッドでも、リソース・アダプター・アーカイブ (RAR) ファイルがワークステーションにインストールされます。この RAR ファイルを使用して、アダプターを構成してください。
2. **アダプターの構成**
   a. (必要であれば) アダプターと連携するようエンタープライズ情報システム (EIS) を構成します。この手順は EIS アプリケーション内から実行してください。
   b. (必要であれば) アプリケーションにアクセスするための認証別名を作成します。

c. アダプターの RAR ファイルをインポートして、WebSphere Integration Developer (J2EE パースペクティブ) でアダプター・プロジェクトを作成します。

d. (必要であれば) WebSphere Integration Developer を使用して、アダプターで必要な任意の外部依存関係をアダプター・プロジェクトに追加します。これらの依存関係は、アダプターをデプロイするときにエクスポートされるバンドル済みの EAR ファイルの一部としても必要です。

e. アダプターを構成するには、WebSphere Integration Developer の Business Integration パースペクティブから、エンタープライズ・サービス・ディスカバリー・ウィザードを実行します。エンタープライズ・サービス・ディスカバリー・ウィザードでは、ビジネス・インテグレーション・コンポーネントが生成され、アダプターを最初に構成するときに必要なすべての情報を入力できます。エンタープライズ・サービス・ディスカバリー・ツールからの出力は、1 つ以上のビジネス・オブジェクトとインポートまたはエクスポート・ファイルを含む、ビジネス・インテグレーション・モジュール・プロジェクトに保管されます。

f. (必要な場合) WebSphere Integration Developer を使用して、エンタープライズ・サービス・ディスカバリー・ウィザードで作成したコンポーネントの参照バインディングを生成します。

3. **モジュールのデプロイ**

a. WebSphere Integration Developer の J2EE パースペクティブから、ビジネス・インテグレーション・モジュール・プロジェクトを EAR ファイルとしてエクスポートします。

b. WebSphere Process Server または WebSphere Enterprise Service Bus にモジュールをインストールします。

c. (必要な場合) サーバーの管理コンソールで、次のプロパティーを設定 (または変更) します。

- リソース・アダプター・プロパティー
- 管理 (J2C) 接続ファクトリー・プロパティー
- EIS のアクティブ化仕様プロパティー

第 6 章 アダプターのインストール

アダプターをインストールするには、システム前提条件を確認し、マイグレーション手順を実行し、またすべてのアダプターに共通のインストール手順を実行する必要があります。最後に、WebSphere Adapter for JDBC に固有な追加のインストール手順を実行します。

インストールの前提条件

Adapter for JDBC をインストールする前に、ご使用の環境が必要なハードウェア要件とソフトウェア要件のすべてを満たしていることを確認する必要があります。これらの要件は、アダプター・インストーラーの実行用にサポートされるプラットフォーム、およびアダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件、という 2 つのカテゴリーに分類されます。

アダプター・インストーラーの実行用にサポートされているプラットフォーム

アダプター・インストーラーの実行用にサポートされているプラットフォームは、『Installing IBM WebSphere Adapters』の『Installing』セクションにあります。

アダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件

アダプターの構成、デプロイ、実行のためのハードウェア要件とソフトウェア要件は、Web サイト『IBM WebSphere Adapters and IBM WebSphere Business Integration Adapters: software requirements』にあります。IBM WebSphere Adapters のリストから「Adapter for JDBC, Version 6.0.2」のリンクを選択します。

追加の JAR ファイル

WebSphere Integration Developer バージョン 6.0.1.1 以前のバージョンを使用している場合は、コネクター・プロジェクトのクラスパスに 3 つの追加 JAR ファイルを手動で追加する必要があります。その実行方法について詳しくは、『参照』セクションの『WebSphere Integration Developer バージョン 6.0.1.1 以前への jar ファイルの追加』を参照してください。

バージョン 6.0.2 へのマイグレーション

WebSphere Adapter for JDBC バージョン 6.0.0.x からマイグレーションするには、まず、後方互換性に関する情報を検討し、RAR ファイルをバージョン 6.0.2 の RAR ファイルと置き換える手順を実行します。次に、いくつかのアクティベーション・スペック・プロパティーを編集し、イベント・テーブルに xid 列を追加します。

© Copyright IBM Corp. 2005, 2006

後方互換性

このバージョンの Adapter for JDBC ランタイムは、バージョン 6.0.0.x との後方互換性があります。そのため、6.0.0.x バージョンのアダプターで生成されたエンタープライズ・アーカイブ (EAR) ファイルは、Adapter for JDBC バージョン 6.0.2 ランタイムでもそのまま機能します。

マイグレーションに関する考慮事項

以前のバージョンの Adapter for JDBC リソース・アダプター・アーカイブ (RAR) ファイルを、Adapter for JDBC バージョン 6.0.2 RAR ファイルに置き換える必要があります。

RAR ファイルを置き換えるには、まず、デプロイ済みのエンタープライズ・アーカイブ (EAR) ファイル用のプロジェクト交換 (PI) ファイルを WebSphere Integration Developer にインポートします。

次に、古い RAR ファイルを除去して、最新のアダプター RAR ファイルをインポートします。

マイグレーションの実行

Adapter for JDBC バージョン 6.0.2 へマイグレーションするには、アクティベーション・スペック・プロパティーの一部を更新して、インバウンド・サービス記述を編集します。また、イベントのフィルター操作を実行したい場合は、プロパティーを設定する必要もあります。イベント・テーブルを変更して、トランザクション ID (xid) 列を追加する必要があります。

始める前に

以前のバージョンのリソース・アダプター・アーカイブ (RAR) ファイルを最新のアダプター RAR ファイルに置き換えます。

このタスクの実行方法

1. プロパティーを更新してインバウンド・サービス記述を編集する

   バージョン 6.0.0.x の Adapter for JDBC を使用して生成されたプロジェクト交換 (PI) ファイルをインポートした後に、アセンブリー・エディターを使用して、次の表に示すアクティベーション・スペック・プロパティーのみを更新することによって、インバウンド・サービス記述を編集します。

表 21. マイグレーション中に更新するアクティベーション・スペック・プロパティー

プロパティー	ERROR! SEGMENT DATA CORRUPTED, SEGDATA=型	説明	デフォルト値	グローバル化
AssuredOnceDelivery	Boolean	プロパティーを true に設定すると、イベント・ストアの各イベントに xid 値が設定されます。さらに、各イベントが対応するエンドポイントに送達され、その後イベント・テーブルから削除されます。	true	いいえ
CustomDeleteQuery	String	各イベントが処理された後に実行されるカスタム削除照会。これは双方向言語で使用される場合に使用可能化されます。	なし	はい
CustomEventQuery	String	カスタム・イベント処理用の SQL 照会、ストアード・プロシージャー、またはストアード関数。この照会は、EventQueryType が Dynamic に設定されている場合に、ポーリング周期中に毎回実行されます。これは双方向言語で使用される場合に使用可能化されます。	なし	はい
CustomUpdateQuery	String	同じイベントが次回のイベント周期で処理対象として取り出されないように、各イベントが処理された後に実行されるカスタム更新照会。これは双方向言語で使用される場合に使用可能化されます。	なし	はい
DataSource JNDIName	String	データベースへの接続を確立するためにアダプターによって使用される名前。ユーザー名およびパスワードも設定されている場合は、それらも接続の確立時に使用されます。ユーザー名もパスワードも設定されていない場合は、DataSourceJNDIName プロパティーのみが使用されます。	なし	はい
EventFilterType	String	アダプターは、処理するイベントをビジネス・オブジェクト・タイプでフィルターに掛けることができます。 EventFilterType には、コンマで区切られたビジネス・オブジェクト・タイプのリストが含まれ、このプロパティーで指定されたタイプのみが処理対象として取り出されます。プロパティーに値が指定されていない場合、フィルターは適用されず、すべてのイベントが処理対象として取り出されます。	NULL	はい
EventQueryType	String	標準イベント・ストアまたはカスタム照会のいずれを使用するかを決定します。有効な値は、標準イベント・ストア用の Standard とカスタム・イベント処理用の Dynamic です。	なし	いいえ

表 21. マイグレーション中に更新するアクティベーション・スペック・プロパティー (続き)

プロパティー	ERROR! SEGMENT DATA COR-RUPTED, SEGDATA=型	説明	デフォルト値	グローバル化
FilterFutureEvents	Boolean	このプロパティーが true に設定されている場合、アダプターは、タイム・スタンプに基づいてイベントをフィルターに掛けます。アダプターは、各ポーリング周期のシステム時刻を各イベントのタイム・スタンプと比較します。イベントが将来発生するように設定されている場合は、その時刻になるまで処理対象として取り出されません。	false	いいえ
SPAfterPoll	String	ポーリング周期の終了ごとに、実行する任意のストアード・プロシージャー。これは、1 つの入力パラメーター (ポーリング数量) を取ります。これは双方向言語で使用される場合に使用可能化されます。	なし	はい
SPBeforePoll	String	実際のポーリング照会の呼び出しの前に実行する任意のストアード・プロシージャー。これは、1 つの入力パラメーター (ポーリング数量) を取ります。これは双方向言語で使用される場合に使用可能化されます。	なし	はい

2. **オプション:** イベントをフィルターに掛ける場合はプロパティーを設定する

   FilterFutureEvents (タイム・スタンプによるイベント・フィルター操作) および EventFilterType (ビジネス・オブジェクト・タイプによるイベント・フィルター操作) の 2 つのアクティベーション・スペック・プロパティーを使用して、イベント・フィルター操作を使用可能にすることができます。

   デフォルトでは、FilterFutureEvents は false に設定されています。タイム・スタンプによってイベントをフィルタリングする場合は、設定を変更します。

   ビジネス・オブジェクト・タイプによってフィルタリングするには、EventFilterType プロパティーに、コンマで区切ったビジネス・オブジェクト・タイプのリストを設定します。そのようなフィルター操作を行わない場合は、ブランクにします。

3. イベント・テーブルに xid 列を追加する

   注: イベント・テーブルを変更する前に、イベント配布テーブル (EDT) にイベントがないことを確認してください。Adapter for JDBC バージョン 6.0.2 は EDT を使用しませんが、異なるメカニズムを使用して、イベントの「送達は 1 回のみ」を提供します。また、マイグレーションの進行中にイベント・テーブルにイベントが追加されないようにしてください。

   以下のサンプル SQL ステートメントを使用して、イベント・テーブルを変更します。

```
ALTER TABLE <tablename> ADD COLUMN xid VARCHAR(255);
```

インストールの実行

アダプターをインストールする基本的な手順は、すべての WebSphere Adapters で同じです。グラフィカル・ユーザー・インターフェースを使用するか、サイレント・インストールを実行することで、アダプターをインストールできます。共通のインストール手順を実行した後、WebSphere Adapter for JDBC 固有のインストール手順も実行する必要があります。

始める前に

インストールの前提条件を検討します。

このタスクの実行方法

1. すべてのアダプターに共通する基本的なインストール手順を使用して、アダプターをインストールします。 この手順は、『Installing IBM WebSphere Adapters』の『Installing』で説明されています。
2. 以下の、WebSphere Adapter for JDBC 固有の手順を実行します。
   a. JDBC ドライバー JAR ファイルがクラス・パス内に含まれるように JDBC ドライバーを設定します。

      JDBC エンタープライズ・サービス・ディスカバリーは、エンタープライズ・サービス・ディスカバリー・ウィザード内のプロジェクト内に存在します。JDBC エンタープライズ・サービス・ディスカバリー処理を実行するには、JDBC ドライバー JAR ファイルがこのツールの JDBC プロジェクトのクラス・パスに含まれている必要があります。

結果

リソース・アダプター・アーカイブ (RAR) ファイルが、アダプターがインストールされているワークステーションにコピーされます。デフォルトのインストール・ロケーションを受け入れた場合、RAR ファイルは、C:¥Program Files¥IBM¥ResourceAdapters¥JDBC¥adapter¥jdbc¥deploy¥CWYBC_JDBC.rar ディレクトリーに配置されます。

次の作業

アダプターを構成します。

インストール済みファイルの構造

アダプターをインストールしたら、インストールされたファイルおよびディレクトリーを見ることができます。それらのすべてのファイルやディレクトリーでは、インストール・ディレクトリーがルートとなります。

例えば、アダプターのインストール・ディレクトリーが c:¥WebSphereBI の場合、CWYBC_JDBC.rar ファイルの絶対パスは c:¥WebSphereBI¥adapter¥jdbc¥deploy¥CWYBC_JDBC.rar になります。

リソース・アダプター・アーカイブ (RAR) ファイルには、アダプターのファイルとエンタープライズ・サービス・ディスカバリー・ウィザードのファイルの両方が含まれています。

UNIX® および Windows のプラットフォームでは、インストール済みディレクトリーおよびファイルの同じ構造が共用されます。唯一の相違はディレクトリー・パスの指定です (UNIX の場合はスラッシュ、Windows では円記号を使用)。

以下の表に、WebSphere Adapter for JDBC の UNIX/Linux ディレクトリーおよびファイルをリストします。ディレクトリーおよびファイルはカテゴリーごとにグループ化されています。

表 22. UNIX/Linux のディレクトリーおよびファイルの構造

ファイルおよびディレクトリーのカテゴリー	ディレクトリーおよびファイル
Adapter for JDBC RAR ファイル	adapter/jdbc/deploy/CWYBC_JDBC.rar
IBM Support Assistant プラグイン ZIP ファイル	adapter/jdbc/ISAPlugin/ com.ibm.esupport.client.SS6FE6_RAJDBC.zip
IBM Tivoli® License Manager (ITLM) ファイル	adapter/jdbc/5724L77E060002.sys
メッセージ・ファイル	adapter/jdbc/messages/CWYBC_JDBC_messages.tar
ファウンデーション・クラスのメッセージ・ファイル	adapter/jdbc/messages/ CWYBS_AdapterFoundation_messages.tar
サンプル・チュートリアル Project Interchange ファイル	adapter/jdbc/samples/referencefiles/Tutorial1.zip
IBM DB2 用サンプル・スクリプト	adapter/jdbc/samples/scripts/scripts_db2.sql
Oracle 用サンプル・スクリプト Microsoft SQL 用サンプル・スクリプト	adapter/jdbc/samples/scripts_oracle.sql adapter/jdbc/samples/scripts_mssql.sql

以下の表に、WebSphere Adapter for JDBC の Windows ディレクトリーおよびファイルをリストします。ディレクトリーおよびファイルはカテゴリーごとにグループ化されています。

表 23. Windows のディレクトリーおよびファイルの構造

ファイルおよびディレクトリーのカテゴリー	ディレクトリーおよびファイル
Adapter for JDBC RAR ファイル	adapter¥jdbc¥deploy¥CWYBC_JDBC.rar
IBM Support Assistant プラグイン ZIP ファイル	adapter¥jdbc¥ISAPlugin¥ com.ibm.esupport.client.SS6FE6_RAJDBC.zip
IBM Tivoli License Manager (ITLM) ファイル	adapter¥jdbc¥5724L77E060002.sys
メッセージ・ファイル	adapter¥jdbc¥messages¥CWYBC_JDBC_messages.zip
ファウンデーション・クラスのメッセージ・ファイル	adapter¥¥jdbc¥messages¥ CWYBS_AdapterFoundation_messages.zip
サンプル・チュートリアル Project Interchange ファイル	adapter¥jdbc¥samples¥referencefiles¥Tutorial1.zip

表 23. Windows のディレクトリーおよびファイルの構造 (続き)

ファイルおよびディレクトリーのカテゴリー	ディレクトリーおよびファイル
IBM DB2 用サンプル・スクリプト	adapter¥jdbc¥samples¥scripts¥scripts_db2.sql
Oracle 用サンプル・スクリプト	adapter¥jdbc¥samples¥scripts_oracle.sql
Microsoft SQL 用サンプル・スクリプト	adapter¥jdbc¥samples¥scripts_mssql.sql

アダプターのアンインストール

アダプターをアンインストールする手順は、すべての WebSphere Adapters で同じです。グラフィカル・ユーザー・インターフェースを使用するか、サイレント・アンインストールを実行することで、アダプターをアンインストールできます。

このタスクの概説

アダプターのアンインストールは、インストールの問題をトラブルシューティングするために必要なタスクとなる場合があります。アダプターのアンインストール手順については、『WebSphere Adapters のインストール (Installing WebSphere Adapters)』の『アンインストール (Uninstalling)』セクションを参照してください。

注: 既にデプロイされているアダプターをアンインストールする必要がある場合は、 183 ページの『関連製品情報』の『必要な場合のあるその他のアダプター関連情報』セクションを参照してください。

第 7 章 デプロイメントのためのアダプターの構成

WebSphere Adapter for YOUR ADAPTER NAME を WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイできるように構成するには、WebSphere Integration Developer を使用して、アダプター・プロジェクトを作成し、必要なファイルをプロジェクトに追加し、ディスカバーしたいビジネス・オブジェクトやそれらをディスカバーするシステムを指定します。

アダプター操作のための EIS の構成

Inbound イベント処理を実行する前に、データベース内にイベント・ストアをセットアップする必要があります。必要に応じて、ユーザー・テーブル上にトリガーを設定できます。

このタスクの実行方法

1. イベント・ストアをセットアップします。 Inbound 処理を実行する前に、データベース内にイベント・ストアをセットアップする必要があります。 IBM DB2、Oracle、または Microsoft SQLServer データベース用イベント・ストアをセットアップするために、以下のようなサンプル・スクリプトが用意されています。
   - `WBIA_JDBC_EventStore_DB2.sql`
   - `WBIA_JDBC_EventStore_Oracle.sql`
   - `WBIA_JDBC_EventStore_MSSQL.sql`
2. **オプション:** トリガーをユーザー・テーブル上にセットアップします。 ユーザー・テーブルへの変更によって、イベント・ストアに保管されるイベントが自動的に生成されるように、必要に応じてトリガーをユーザー・テーブル上にセットアップします。

認証別名の作成

サーバー上の管理コンソールを使用して、認証別名を作成します。管理コンソールで、グローバル・セキュリティーを構成し、Outbound 要求の処理に使用される認証別名のパスワードを設定します。

このタスクの概説

アダプターを構成する前に、認証別名をサーバー上に作成する必要があります。認証別名を作成するには、以下の手順を使用してください。

このタスクの実行方法

1. 管理コンソールの「ようこそ」ページで、**「セキュリティー」** → **「グローバル・セキュリティー」**をクリックします。
2. 「認証」見出しの下で、**「JAAS 構成」** → **「J2C 認証データ」**をクリックします。
3. **「新規作成」**をクリックします。

© Copyright IBM Corp. 2005, 2006

4. 「別名」、「ユーザー ID」、「パスワード」、および「説明」の各フィールドに必要な情報を入力します。

   注: 入力したユーザー ID とパスワードは、Outbound 処理のためデータベースとの接続を確立するために使用されます。
5. 「**OK**」をクリックします。
6. 「**保管**」をクリックした後、再度「**保管**」をクリックします。

WebSphere Integration Developer でのアダプター・プロジェクトの作成

モジュールの作成とデプロイのプロセスを開始するには、まずアダプター・プロジェクトを作成します。このアダプター・プロジェクトには、アダプター自体と他の関連する成果物が含まれています。 RAR ファイル (インストール時に、ご使用のローカル・ファイル・システムにコピーされます) を WebSphere Integration Developer にインポートすることによって、プロジェクトを作成します。

始める前に

Adapter for JDBC がインストール済みで、認証別名を作成していることを確認してください。

このタスクの概説

アダプター・プロジェクト (WebSphere Integration Developer では、*コネクター・プロジェクト* と呼びます) を作成して、アダプター (アダプターのインストール・ディレクトリーからインポートします) をそれに関連する成果物とともに含めます。すべてのプロジェクトが必要なものを完備しています。つまり、各プロジェクトはプロジェクト外のオブジェクトを参照しません。

アダプター・プロジェクトを作成するには、以下の手順を使用してください。

このタスクの実行方法

1. WebSphere Integration Developer が現在実行されていない場合は、開始します。
   a. 「**スタート**」 → 「**プログラム**」 → 「**IBM WebSphere**」 → 「**Integration Developer V6.0.2**」 → 「**WebSphere Integration Developer V6.0.2**」をクリックします。
   b. ワークスペースを指定するようにプロンプトが表示された場合は、デフォルト値を受け入れます。

      ワークスペースとは、WebSphere Integration Developer がプロジェクトを保管するディレクトリーのことです。
   c. 「WebSphere Integration Developer」ウィンドウが表示されたら、「ようこそ」ページを閉じます。
2. 以下のようにして、J2EE パースペクティブに切り替えます。
   a. 「**ウィンドウ**」 → 「**パースペクティブを開く**」 → 「**その他**」をクリックします。
   b. 「**J2EE**」をクリックします。

「**J2EE**」が「パースペクティブの選択」ウィンドウに表示されない場合は、「**すべて表示**」チェック・ボックスを選択して「**J2EE**」をクリックし、「**OK**」をクリックします。

c. 「使用可能化の確認」ウィンドウが表示された場合は、「**常に機能を使用可能にし、今後このメッセージを表示しない**」を選択してください。

d. 「**OK**」をクリックします。

3. 「**コネクター・プロジェクト**」を右クリックして RAR ファイルをインポートし、「**インポート**」 → 「**RAR ファイル**」をクリックします。

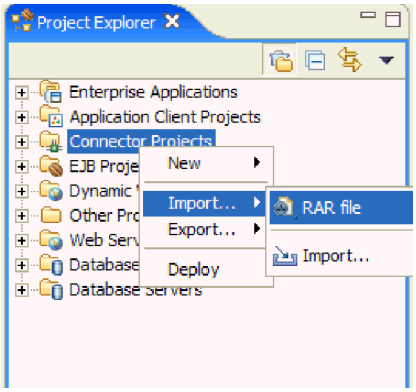


図 12. RAR ファイルのインポート

4. 「コネクター・インポート」ウィンドウで、「**参照**」をクリックして、Adapter for JDBC がインストールされたディレクトリーへナビゲートします。
5. 「**CWYBC_JDBC.rar**」をクリックします。

   コネクター・プロジェクトは、RAR ファイルと同じ名前を持ちます。

   CWYBC_JDBC.rar という名前のプロジェクトが既にワークスペース内に存在する場合、「**コネクター・プロジェクト**」フィールド内の名前には番号が付加されています (例えば CWYBC_JDBC1)。
6. **オプション:** 「**コネクター・プロジェクト**」フィールドにプロジェクト用の別の名前を入力するか、またはデフォルト値を受け入れます。
7. **オプション:** 「**ターゲット・サーバー**」フィールドでアダプターのデプロイ先のサーバーを選択するか、またはデフォルト値を受け入れます。
8. 「**モジュールを EAR プロジェクトに追加**」チェック・ボックスをクリアします。

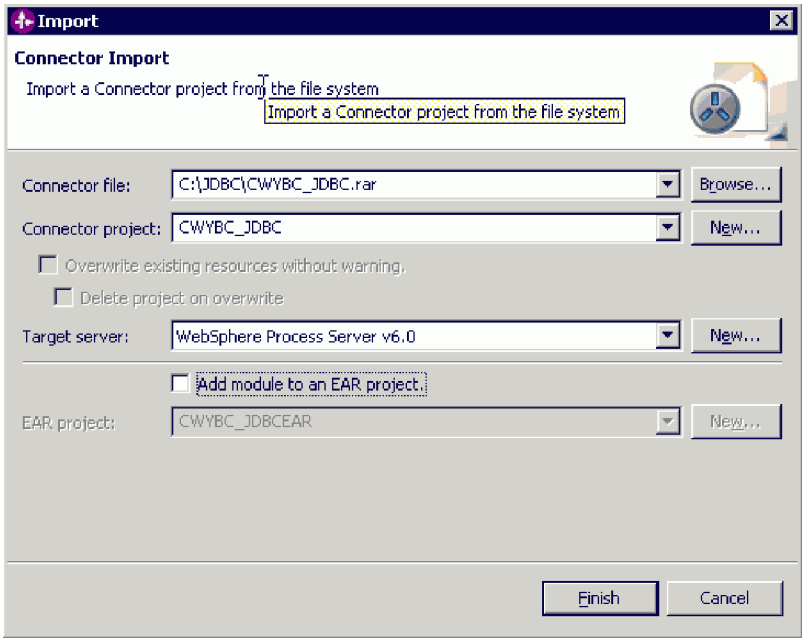


図 13. 「モジュールを EAR プロジェクトに追加」チェック・ボックスのクリア

チェックマークを外すと、「EAR プロジェクト」フィールドは無効になります。

9. **「終了」**をクリックします。
10. JDBC ドライバーをコネクター・プロジェクトに追加します。

    コネクター・プロジェクトがアプリケーション・サーバーにデプロイする EAR ファイルの一部となるように、JDBC ドライバーをコネクター・プロジェクトに追加する必要があります。これは、RAR ファイルのインポート後、またはアプリケーション・サーバーへの EAR のインストール後に実行できます。

オプション	説明
RAR ファイルをインポートした後で JDBC ドライバーを追加するには、次のようにします。	JAR ファイルをワークスペース内の適切なフォルダーに追加します。 JAR ファイルは、例えば、次のロケーションに追加できます。 `c:¥workspace¥CWYBC_JDBC¥connectorModule`
EAR をアプリケーション・サーバーにインストールした後で JDBC ドライバーを追加するには、次のようにします。	アプリケーションをインストールした後、JAR ファイルを WebSphere Process Server または WebSphere Enterprise Service Bus の installedApps ディレクトリーの RAR サブディレクトリーに追加します。アプリケーションのアプリケーション・サーバーへのインストールの詳細については、『モジュールのデプロイ』を参照してください。

結果

新規の J2EE コネクター・プロジェクトが作成されました。内容を確認するには、「プロジェクト・エクスプローラー」内のプロジェクトを展開します。例えば、コネクター・プロジェクトが CWYBC_JDBC という名前の場合は、**CWYBC_JDBC** を展開します。

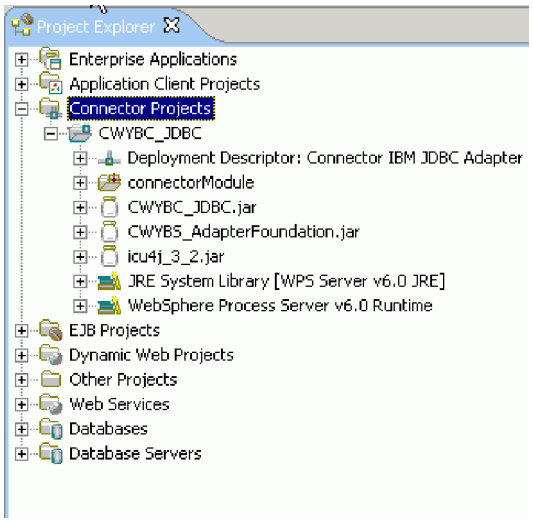


図 14. 「プロジェクト・エクスプローラー」ウィンドウ内のコネクター・プロジェクト

次の作業

必要な外部依存関係をプロジェクトに追加します。

外部ソフトウェア依存関係の追加

WebSphere Integration Developer でプロジェクトを作成したら、データベース・ドライバーをアダプター・プロジェクトに追加する必要があります。これを行うには、JDBC ドライバーを Java ビルド・パスに追加します。これには、ご使用のデータベースに提供されている外部 JAR ファイルを使用してください。

このタスクの実行方法

1. WebSphere Integration Developer 内の「J2EE」パースペクティブ内で、「**コネクター・プロジェクト**」を展開します。 アダプター・プロジェクト **CWYBC_JDBC** を右クリックして、「**プロパティー**」を選択します。

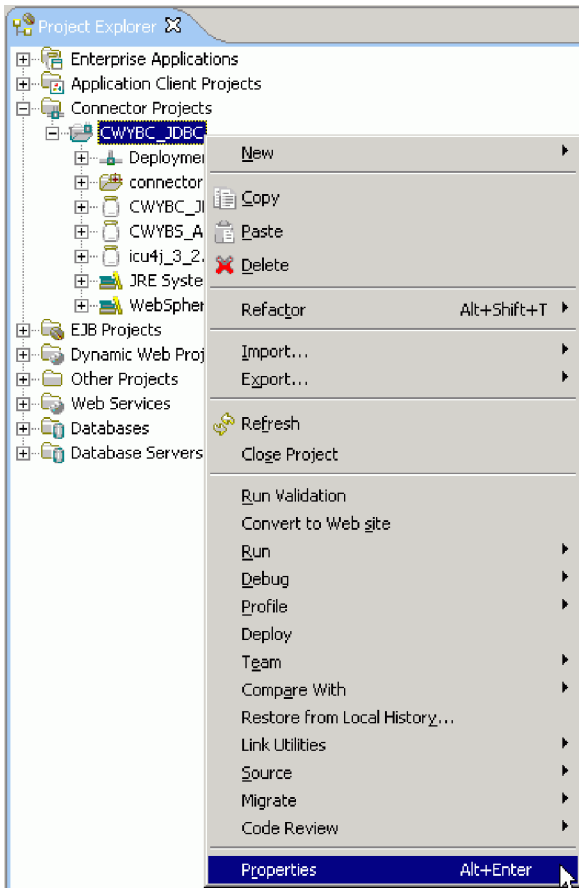


図 15. プロジェクトの「プロパティー」オプションの選択

2. ナビゲーション・サイドバーで、「**Java のビルド・パス**」をクリックします。「**ライブラリー**」タブを選択し、「**外部 JAR の追加**」をクリックします。
3. 「JAR の選択」ウィンドウで、JDBC ドライバー JAR ファイルを含むローカル・ファイル・システム上のディレクトリーにナビゲートします。「**開く**」をクリックします。

   JDBC ドライバー JAR ファイルが Java ビルド・パスに追加されます。
4. 「**OK**」をクリックします。

結果

アダプター・プロジェクトには JDBC ドライバーへの参照が含まれています。これは、WebSphere Integration Developer の「プロジェクト・エクスプローラー」ウィンドウで確認できます。

Outbound 処理を行うためのアダプターの構成

Outbound 処理を行えるよう WebSphere Adapter for YOUR ADAPTER NAME を構成するには、WebSphere Integration Developer でエンタープライズ・サービス・ディスカバリー・ウィザードを使用して、エンタープライズ・サービス・ディスカバリーの接続プロパティーを設定し、エンタープライズ情報システムにあるビジネス・

オブジェクトまたはサービスを選択し、Outbound 処理するためのビジネス・オブジェクトの定義や関連する成果物を生成します。

エンタープライズ・サービス・ディスカバリーを使用したビジネス・オブジェクトの生成

ビジネス・オブジェクトを生成するには、まず接続プロパティーを設定します。その後データベース・オブジェクトを照会するクエリーを実行して、サービス記述に必要なオブジェクトを選択できます。この選択したオブジェクトを構成して、成果物およびプロパティー値を新規ビジネス・インテグレーション・モジュールに保管します。

エンタープライズ・サービス・ディスカバリー接続プロパティーの設定

アダプター・プロジェクトを作成し、データベース・ドライバーを追加した後、Adapter for JDBC 用にエンタープライズ・サービス・ディスカバリー・ウィザードを始動する必要があります。ウィザードを使用して、ご使用のデータベース・インスタンス用に接続プロパティーの値を設定します。

このタスクの概説

エンタープライズ・サービス・ディスカバリー処理には、ディスカバリーのためのデータベース接続やサービス記述の作成にこれらのプロパティーが必要となります。

このタスクの実行方法

1. エンタープライズ・サービス・ディスカバリーを開始します。
   a. WebSphere Integration Developer で、「**ウィンドウ**」 → 「**パースペクティブを開く**」 → 「**その他**」を選択して、「**ビジネス・インテグレーション**」パースペクティブに移動します。「**ビジネス・インテグレーション (デフォルト) (Business Integration (default))**」をクリックして、「**OK**」をクリックします。
   b. 「**ファイル**」をクリックして、「**新規**」>「**エンタープライズ・サービス・ディスカバリー**」を右クリックします。

      「**エンタープライズ・サービス・ディスカバリー**」が表示されない場合は、「**新規**」 → 「**その他**」をクリックし、「**ビジネス・インテグレーション**」を展開して、「**エンタープライズ・サービス・ディスカバリー**」をクリックします。次に、「**次へ**」をクリックします。
   c. 「エンタープライズ・サービス・ディスカバリー」ウィンドウで、ご使用のアダプターのオプションを選択して、「**次へ**」をクリックします。

      まだ RAR ファイルをインポートしていない場合は、このウィンドウの「**リソース・アダプターのインポート (Import Resource Adapter)**」をクリックしてインポートします。
2. 接続プロパティーの値を設定する。

   ディスカバリーおよびサービス記述作成のため、ターゲット EIS インスタンスへの接続に使用するエンタープライズ・サービス・ディスカバリー接続プロパテ

ィーの値を設定する必要があります。双方向スクリプトデータ処理を使用可能にするには、双方向変換を活動状態にして双方向プロパティーの値を設定する必要があります。

a. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウで、「**接続構成 (Connection Configuration)**」プロパティーの値を入力します。

   必要な値は、アスタリスクでマークされています。ユーザー名、パスワード、データベース URL、および JDBC ドライバー・クラスを入力します。データベース URL および JDBC ドライバー・クラスの適切な値については、ご使用のドライバーの資料を確認してください。

b. 双方向機能を活動状態にするには、「**BiDi 変換**」の横にあるチェック・ボックスを選択します。次に、双方向プロパティーの値を設定します。

これらのプロパティーの詳細については、『参照』セクションの『エンタープライズ・サービス・ディスカバリー接続プロパティー』を参照してください。

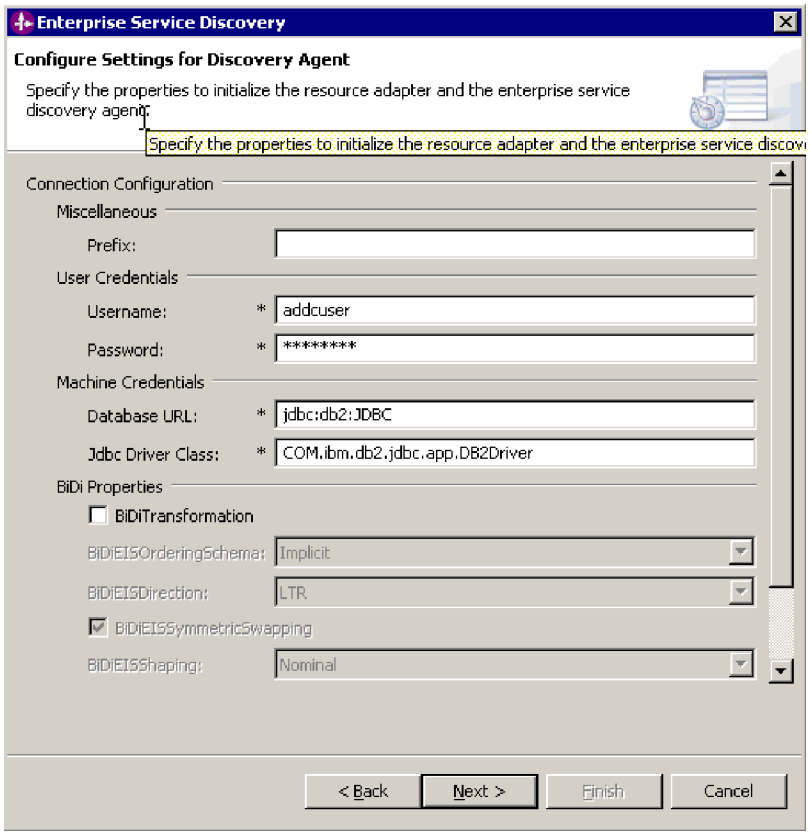


図 16. ディスカバリー・エージェント・ウィンドウの設定を構成する

3. ロギング・オプションを選択する。

   「**ロギング・オプション (Logging options)**」は、エンタープライズ・サービス・ディスカバリー処理のロギングとトレースのセットアップにのみ使用されます。ただし、ロギング・レベルとトレース・レベルは、アダプターのものと同じで

す。アダプターのロギング・レベルとトレース・レベルの詳細については、『トラブルシューティング・ツールの構成』を参照してください。

a. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウの下部で、**「拡張を表示」**をクリックします。ボタンが**「拡張を非表示」**に変わります。

b. **「ロギング・オプション (Logging options)」**で、ログ・ファイルの出力ロケーションを入力するか、参照します。エンタープライズ・サービス・ディスカバリーのロギング・レベルとトレース・レベルを選択します。

c. **「次へ」**をクリックします。

結果

ディスカバリー・サービスは、接続プロパティーを使用して、オブジェクトの選択およびナビゲーション用に表示されるメタデータ・ツリーを作成します。

ビジネス・オブジェクトおよびサービスの選択

接続プロパティーの構成後に、データベース・オブジェクトに対する照会を実行します。エンタープライズ情報システム (EIS) 内のオブジェクトの構造を理解するために、メタデータ・ツリー構造を参照して、サービス記述に必要なオブジェクトを選択します。

始める前に

照会を実行する前に、フィルター・プロパティーを指定して、ツリー構造内に表示されるスキーマ、ノード、またはオブジェクトのリストを絞り込むことができます。

このタスクの実行方法

1. フィルター・プロパティーを指定する。

   a. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、**「照会の編集 (Edit Query)」**をクリックします。「照会フィルター・プロパティー」ウィンドウで、**「スキーマ名フィルター」**プロパティー・フィールドにテキストを入力します。指定したストリングで始まるスキーマが表示されます。使用するスキーマを選択します。

   b. **「タイプ」**プロパティー・フィールドに、テーブル、ビュー、ストアード・プロシージャー、同義語/ニックネームなどの項目がリストされます。そのリストからノードを追加または除去することができます。

   c. 「照会フィルター・プロパティー」ウィンドウで、**「ビジネス・オブジェクト ASI の追加」**にチェック・マークを付け、**「OK」**をクリックします。

      そうすると、ステップ 2 でメタデータ照会を実行したときにオブジェクトを追加するたびに、アプリケーション固有情報を入力するための「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」ウィンドウが表示されます。

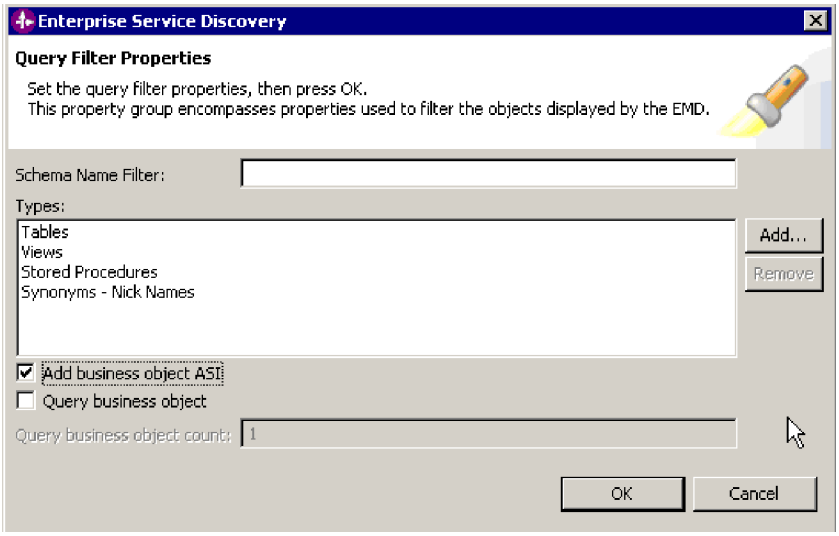


図 17. 「照会フィルター・プロパティー」ウィンドウ

2. メタデータ照会を実行する。「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**照会の実行 (Run Query)**」をクリックします。

   照会によりディスカバーされたオブジェクトが、上部ペインに表示されます。テーブル、ビュー、ストアード・プロシージャー、同義語/ニックネームは、スキーマ名によってソートされます。
3. オブジェクトをフィルタリングする。

   スキーマのフィルタリングと同様の方法で、オブジェクトをフィルタリングすることができます。

   a. テーブル、ビュー、ストアード・プロシージャー、または同義語/ニックネーム・ノードを選択します。「**フィルター (Filter)**」をクリックします。 エンタープライズ・サービス・ディスカバリー・ウィザードによって、そのノードに対して表示するデータベース・オブジェクトのリストをフィルタリングするための「**オブジェクト名フィルター**」が照会されます。

   b. テキストで入力すると、指定されたストリングで始まるデータベース・オブジェクトが表示されます。
4. オブジェクトを強調表示して、「**追加**」をクリックすることによって、インポート用のオブジェクトを選択する。 選択したオブジェクトが下部ペインに表示されます。

   注: 選択したオブジェクトを除去する必要がある場合は、それを強調表示して「**除去 (Remove)**」をクリックします。
5. ビジネス・オブジェクト ASI を追加する。

   「照会フィルター・プロパティー」ウィンドウで「**ビジネス・オブジェクト ASI の追加**」を選択すると、オブジェクトを追加するたびに、アプリケーション固有の情報を入力するための「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」というウィンドウが表示されま

す。このウィンドウは、ここで、**ADDRESS** オブジェクトの動詞に関するアプリケーション固有の情報パラメーターとともに示されています。

「**ステータス列名**」フィールドは、論理削除に使用され、「**ステータス値**」は、「ステータス列名」用に選択された属性に対応する列で更新する値を指定します。

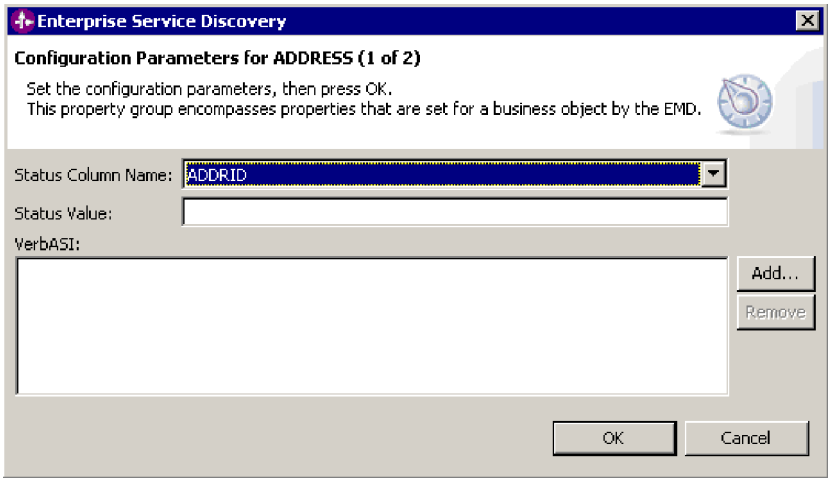


図 18. 「ADDRESS の構成パラメーター (Configuration Parameters for ADDRESS)」ウィンドウ

a. 「**追加**」をクリックします。

b. ストアード・プロシージャーのリストが表示されます。ストアード・プロシージャーを選択して、ストアード・プロシージャーのアプリケーション固有情報を ADDRESS オブジェクトに関連付けます。「**OK**」をクリックします。

   例えば、ストアード・プロシージャーでテーブルにレコードを作成する場合は、「**CreateSP**」構成パラメーターの詳細を入力します。

c. 「**スキーマ**」名および「**ストアード・プロシージャー**」名の値を入力します。ストアード・プロシージャー名を入力すると、「**ストアード・プロシージャー・パラメーター**」フィールドが表示されます。 パラメーターごとに、パラメーターに関連付ける値をビジネス・オブジェクト属性のリストから選択します。

d. ASI の入力を完了したら、「**OK**」をクリックします。

e. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**インポートするオブジェクト (Objects to be imported)**」ペイン内に選択されたオブジェクトが表示されます。「**次へ**」をクリックします。

複数の追加オブジェクトを選択した場合は、最初のオブジェクトの「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」ウィンドウが表示されます。ASI を入力して「**OK**」をクリックする

と、次のオブジェクトのウィンドウが表示されます。オブジェクト・レベル、動詞、および属性に関するアプリケーション固有情報については、『アプリケーション固有の情報』を参照してください。

結果

選択されたオブジェクトが、アダプターで使用できるようにインポートされました。

選択済みオブジェクトの構成

データベース・オブジェクトを選択した後、インポート・ファイルのメタデータ選択プロパティーの値を指定する必要があります。

始める前に

詳しくは、『参照』セクションの『メタデータ選択プロパティー』を参照してください。

このタスクの実行方法

1. ネーム・スペースを指定します。

   デフォルト値が、「オブジェクトの構成 (Configure objects)」ウィンドウに表示されます。このデフォルト値は、メタデータ・スキーマ JDBCASI.xsd のネーム・スペースです。ネーム・スペースは、ビジネス・オブジェクト・スキーマを論理的に分離するため、ビジネス・オブジェクト名の前に付加されます。
2. 「**サービス・タイプ**」として「**Outbound**」を選択します。

   クエリー・ビジネス・オブジェクトとストアード・プロシージャー・ビジネス・オブジェクトは、サービス・タイプ「Outbound」のみをサポートします。
3. 操作を選択します。
   a. 「オブジェクトの構成 (Configure objects)」ウィンドウの「**操作**」フィールドに、選択したサービス・タイプに対してアダプターがサポートしている操作がリストされます。この操作のリストに追加するには、「**追加**」ボタンをクリックします。

      例えば、操作を削除し、後でそれを組み込む場合は次のようにします。
   b. 「追加」ウィンドウで、操作のリストから選択し、「**OK**」をクリックします。

      指定された操作は、生成されるすべてのビジネス・オブジェクトに対して設定されます。
4. RetrieveAll 操作で検索するレコードの最大数を入力して、レコード最大数を設定します。 デフォルト値は 100 です。
5. 生成された .xsd ファイルが保管されるロケーションへのパスを入力して、BO ロケーションを指定します。
6. 完了したら、「**次へ**」をクリックします。

成果物の生成

メタデータ選択プロパティーを指定したら、特定のデータベースへの通信チャネルをセットアップするためにアダプターが使用するプロパティーを構成します。これらのプロパティーには、リソース・アダプター・プロパティー、管理 (J2C) 接続ファクトリー・プロパティーが含まれています。また、すべての成果物およびプロパティー値を保管できる新規ビジネス・インテグレーション・モジュールを作成する必要があります。

このタスクの実行方法

1. 新規モジュール名を指定する。
   - 下に示されている「成果物の生成 (Generate artifacts)」ウィンドウで、新規モジュール名が「**モジュール (Module)**」フィールドに表示されている場合は、アクションを実行する必要はありません。
   - 新規モジュール名が表示されていない場合は、「**新規 (New)**」をクリックする必要があります。「新規統合プロジェクト (New Integration Project)」ウィンドウが表示された場合は、「**モジュール・プロジェクトの作成 (Create a module project)**」を選択して、「**次へ**」をクリックします。「新規モジュール」ウィンドウで、モジュール名を入力して「**終了**」をクリックします。
2. **オプション:** 「成果物の生成 (Generate Artifacts)」ウィンドウで、サービス記述用のフォルダーを指定することができます。これは、新規モジュール内でサービス記述を保管するフォルダーを入力または参照して行います。フォルダー名を指定しない場合、成果物 (インポート・ファイルおよび WSDL ファイル) は、モジュールのルート・フォルダー (すなわちモジュール名のフォルダー) に保管されます。
3. インポート・ファイル名は、「**名前**」フィールドに表示されます。「**記述**」フィールドにコメントを追加することもできます。
4. 「**操作名の編集 (Edit Operation names)**」をクリックして、作成しているビジネス・オブジェクトの全操作のデフォルト名を確認します。

   デフォルト名は、オペレーション名とビジネス・オブジェクト名の組み合わせからなります。
5. 「**コネクターをモジュールとともに配置 (Deploy connector with module)**」チェック・ボックスを選択します。

   注: 「**コネクターをモジュールとともに配置 (Deploy connector with module)**」にチェック・マークを付けて、アダプター・プロジェクトの RAR ファイルが、アプリケーション・サーバーにデプロイする EAR ファイルに組み込まれるようにします。
6. ランタイムに EIS への接続に使用される構成プロパティー値を指定します。
   - 「**ディスカバーされた接続プロパティーを使用 (Use discovered connection properties)**」を選択して、プロパティー値を設定します。
   - プロジェクトのデプロイ後、新規ビジネス・オブジェクトの作成が必要になったときに、「**サーバーで指定された接続プロパティーを使用 (Use connection properties specified on server)**」を選択することができます。この選択は、アプリケーション・サーバー上に既にあるプロパティーを使用することを示します。

7. プロパティー値を設定します。必須のプロパティー・フィールドにはアスタリスクが付いています。

   Outbound 処理の場合は、管理 (J2C) 接続ファクトリー・プロパティーおよびリソース・アダプター・プロパティー用のプロパティー・フィールドが表示されます。

   双方向変換を活動化している場合は、Outbound 処理用の双方向プロパティー・フィールドが表示されます。

   これらの構成プロパティーの詳細については、『参照』セクションの『アダプター構成プロパティー』を参照してください。

8. 「**J2C 認証データ・エントリー (J2C Authentication Data Entry)**」フィールドで、アプリケーション・サーバー上にセットアップする認証別名を指定します。

   名前は大文字小文字が区別されます。詳しくは、『認証別名の作成』を参照してください。

9. 接続プロパティーである「**ユーザー名**」、「**パスワード**」、「**データベース URL**」、および「**JDBC ドライバー・クラス**」を入力します。これらは、エンタープライズ・サービス・ディスカバリー・ウィザードの始動後にセットアップします。

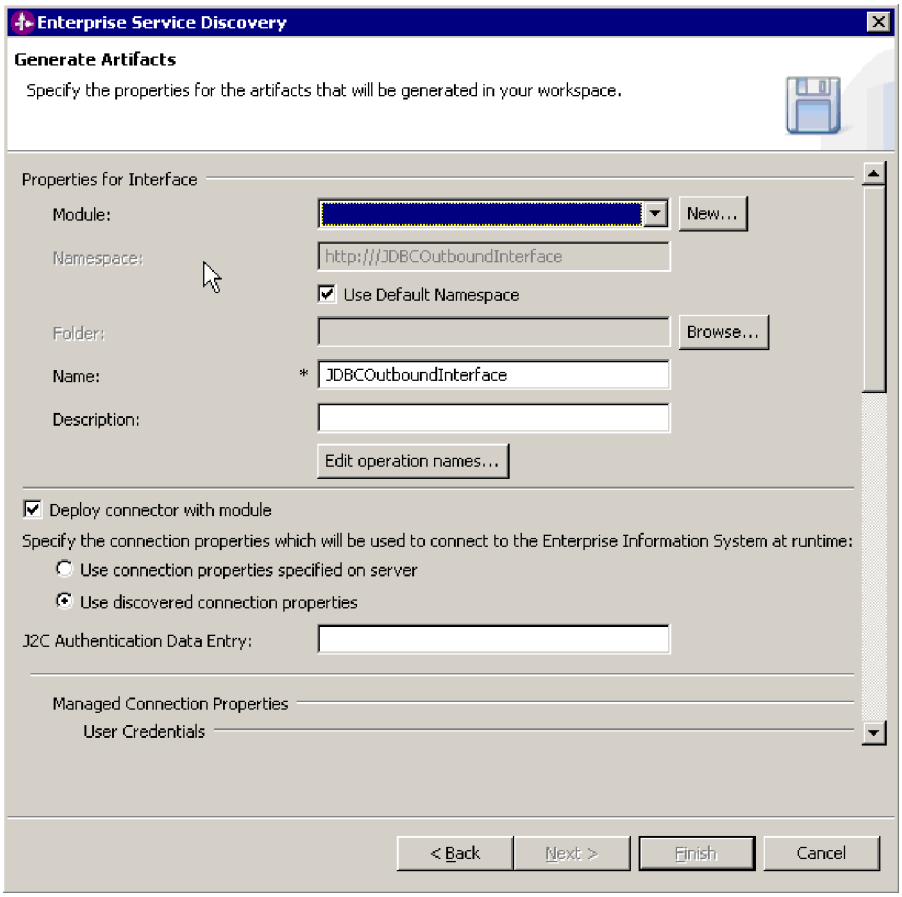


図 19. 「成果物の生成 (Generate Artifacts)」ウィンドウ

10. 必要なプロパティーの設定を完了したら、「**終了**」をクリックします。

参照バインディングの生成

参照バインディングは、その他の WebSphere Business Integration Service Component Architecture (SCA) コンポーネントがアダプターにアクセスするために使用します。その他のサーバー・プロセスへアダプターをリンクしてプロジェクト・モジュールからアダプターへの参照を作成します。

このタスクの概説

参照バインディング生成は、テスト環境でのみ必須です。実稼働環境でアダプターを配置する際には必要ありません。

このタスクの実行方法

1. SCA コンポーネントを作成する
   a. WebSphere Integration Developer の「**ビジネス・インテグレーション (Business Integration)**」パースペクティブに移動します。「**ビジネス・インテグレーション**」タブで、「**JDBCEMD**」モジュールを右クリックし、「**アプリケーションから開く**」>「**アセンブリー・エディター**」を選択します。
   b. 「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウが表示され、モジュールのインポート・コンポーネントがビューに示されます。新規

コンポーネントを作成するには、ウィンドウの左側 (縦) のフレームで 2 番目のアイコン (インポート) をクリックします。

c. アイコンの新規メニューで下のアイコンをクリックします (これには、「**スタンドアロン参照 (Standalone References)**」という吹き出しヘルプがあります)。カーソルが配置アイコンに変わります。

d. パレットをクリックして、新規コンポーネントを「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウに追加します。

2. スタンドアロン参照を作成する

a. モジュールのインポート・コンポーネントをクリックして新規コンポーネントにドラッグします。これにより、インポート・コンポーネントから新規コンポーネントへ線が引かれます。

b. 「線の追加 (Add Wire)」ウィンドウで、「**OK**」をクリックします。別の「線の追加 (Add Wire)」ウィンドウが表示され、WSDL インターフェースを Java に変換するかどうかが尋ねられます。「**いいえ**」をクリックします。

c. 「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウに、新規「**スタンドアロン参照 (Standalone Reference)**」コンポーネントが、モジュールのインポート・コンポーネントに接続する「線」とともに表示されます。

「**ファイル**」>「**保管**」をクリックしてアセンブリー・ダイアグラムを保管します。

詳細については、http://www.ibm.com/software/integration/wid にある WebSphere Integration Developer のユーザー・ガイドを参照してください。

Inbound 処理を行うためのアダプターの構成

Inbound 処理を行えるよう WebSphere Adapter for YOUR ADAPTER NAME を構成するには、WebSphere Integration Developer でエンタープライズ・サービス・ディスカバリー・ウィザードを使用して、アダプターの接続プロパティーを設定し、エンタープライズ情報システムにあるビジネス・オブジェクトまたはサービスを選択し、Inbound 処理するためのビジネス・オブジェクトの定義や関連する成果物を生成します。

エンタープライズ・サービス・ディスカバリーを使用したビジネス・オブジェクトの生成

ビジネス・オブジェクトを生成するには、まず接続プロパティーを設定します。その後データベース・オブジェクトを照会するクエリーを実行して、サービス記述に必要なオブジェクトを選択できます。この選択したオブジェクトを構成して、成果物およびプロパティー値を新規ビジネス・インテグレーション・モジュールに保管します。

エンタープライズ・サービス・ディスカバリー接続プロパティーの設定

アダプター・プロジェクトを作成し、データベース・ドライバーを追加した後、Adapter for JDBC 用にエンタープライズ・サービス・ディスカバリー・ウィザードを始動する必要があります。ウィザードを使用して、ご使用のデータベース・インスタンス用に接続プロパティーの値を設定します。

このタスクの概説

エンタープライズ・サービス・ディスカバリー処理には、ディスカバリーのためのデータベース接続やサービス記述の作成にこれらのプロパティーが必要となります。

このタスクの実行方法

1. エンタープライズ・サービス・ディスカバリーを開始します。
   a. WebSphere Integration Developer で、「**ウィンドウ**」 → 「**パースペクティブを開く**」 → 「**その他**」を選択して、「**ビジネス・インテグレーション**」パースペクティブに移動します。「**ビジネス・インテグレーション (デフォルト) (Business Integration (default))**」をクリックして、「**OK**」をクリックします。
   b. 「**ファイル**」をクリックして、「**新規**」>「**エンタープライズ・サービス・ディスカバリー**」を右クリックします。

      「**エンタープライズ・サービス・ディスカバリー**」が表示されない場合は、「**新規**」 → 「**その他**」をクリックし、「**ビジネス・インテグレーション**」を展開して、「**エンタープライズ・サービス・ディスカバリー**」をクリックします。次に、「**次へ**」をクリックします。
   c. 「エンタープライズ・サービス・ディスカバリー」ウィンドウで、ご使用のアダプターのオプションを選択して、「**次へ**」をクリックします。

      まだ RAR ファイルをインポートしていない場合は、このウィンドウの「**リソース・アダプターのインポート (Import Resource Adapter)**」をクリックしてインポートします。
2. 接続プロパティーの値を設定する。

   ディスカバリーおよびサービス記述作成のため、ターゲット EIS インスタンスへの接続に使用するエンタープライズ・サービス・ディスカバリー接続プロパティーの値を設定する必要があります。双方向スクリプトデータ処理を使用可能にするには、双方向変換を活動状態にして双方向プロパティーの値を設定する必要があります。
   a. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウで、「**接続構成 (Connection Configuration)**」プロパティーの値を入力します。

      必要な値は、アスタリスクでマークされています。ユーザー名、パスワード、データベース URL、および JDBC ドライバー・クラスを入力します。データベース URL および JDBC ドライバー・クラスの適切な値については、ご使用のドライバーの資料を確認してください。
   b. 双方向機能を活動状態にするには、「**BiDi 変換**」の横にあるチェック・ボックスを選択します。次に、双方向プロパティーの値を設定します。

   これらのプロパティーの詳細については、『参照』セクションの『エンタープライズ・サービス・ディスカバリー接続プロパティー』を参照してください。

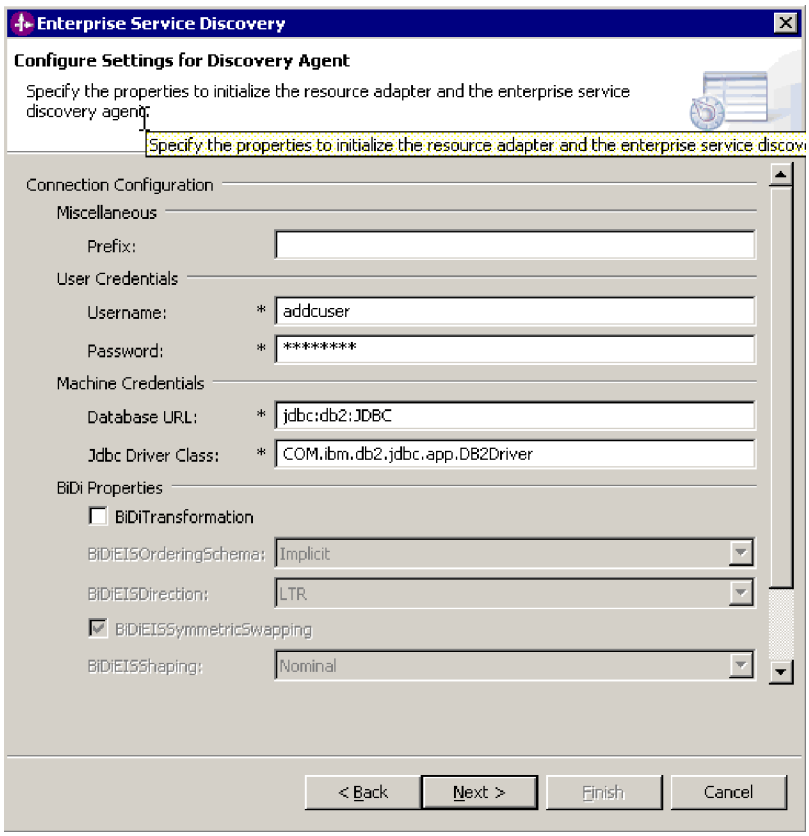


図 20. ディスカバリー・エージェント・ウィンドウの設定を構成する

3. ロギング・オプションを選択する。

   「**ロギング・オプション (Logging options)**」は、エンタープライズ・サービス・ディスカバリー処理のロギングとトレースのセットアップにのみ使用されます。ただし、ロギング・レベルとトレース・レベルは、アダプターのものと同じです。アダプターのロギング・レベルとトレース・レベルの詳細については、『トラブルシューティング・ツールの構成』を参照してください。

   a. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウの下部で、「**拡張を表示**」をクリックします。ボタンが「**拡張を非表示**」に変わります。

   b. 「**ロギング・オプション (Logging options)**」で、ログ・ファイルの出力ロケーションを入力するか、参照します。エンタープライズ・サービス・ディスカバリーのロギング・レベルとトレース・レベルを選択します。

   c. 「**次へ**」をクリックします。

結果

ディスカバリー・サービスは、接続プロパティーを使用して、オブジェクトの選択およびナビゲーション用に表示されるメタデータ・ツリーを作成します。

ビジネス・オブジェクトおよびサービスの選択

接続プロパティーの構成後に、データベース・オブジェクトに対する照会を実行します。エンタープライズ情報システム (EIS) 内のオブジェクトの構造を理解するために、メタデータ・ツリー構造を参照して、サービス記述に必要なオブジェクトを選択します。

始める前に

照会を実行する前に、フィルター・プロパティーを指定して、ツリー構造内に表示されるスキーマ、ノード、またはオブジェクトのリストを絞り込むことができます。

このタスクの実行方法

1. フィルター・プロパティーを指定する。
   a. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**照会の編集 (Edit Query)**」をクリックします。「照会フィルター・プロパティー」ウィンドウで、「**スキーマ名フィルター**」プロパティー・フィールドにテキストを入力します。指定したストリングで始まるスキーマが表示されます。使用するスキーマを選択します。
   b. 「**タイプ**」プロパティー・フィールドに、テーブル、ビュー、ストアード・プロシージャー、同義語/ニックネームなどの項目がリストされます。そのリストからノードを追加または除去することができます。
   c. 「照会フィルター・プロパティー」ウィンドウで、「**ビジネス・オブジェクト ASI の追加**」にチェック・マークを付け、「**OK**」をクリックします。

      そうすると、ステップ 2 でメタデータ照会を実行したときにオブジェクトを追加するたびに、アプリケーション固有情報を入力するための「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」ウィンドウが表示されます。

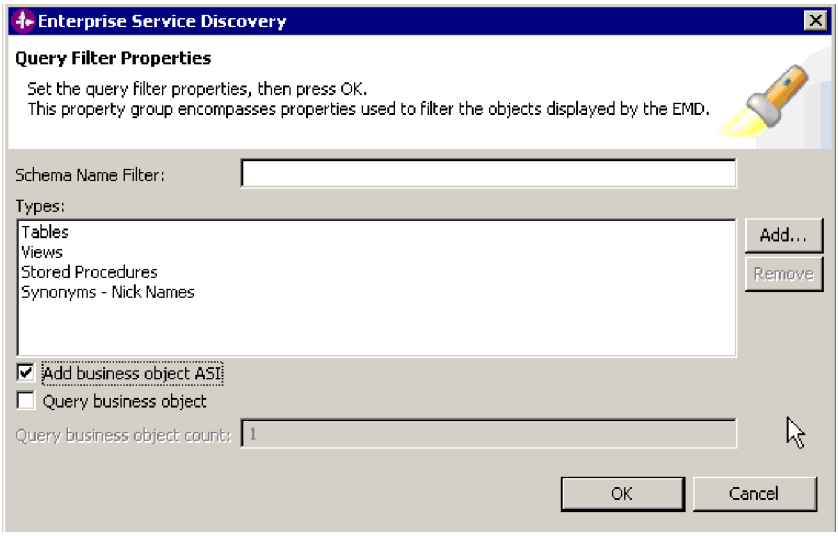


図 21. 「照会フィルター・プロパティー」ウィンドウ

2. メタデータ照会を実行する。「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**照会の実行 (Run Query)**」をクリックします。

   照会によりディスカバーされたオブジェクトが、上部ペインに表示されます。テーブル、ビュー、ストアード・プロシージャー、同義語/ニックネームは、スキーマ名によってソートされます。
3. オブジェクトをフィルタリングする。

   スキーマのフィルタリングと同様の方法で、オブジェクトをフィルタリングすることができます。

   a. テーブル、ビュー、ストアード・プロシージャー、または同義語/ニックネーム・ノードを選択します。「**フィルター (Filter)**」をクリックします。 エンタープライズ・サービス・ディスカバリー・ウィザードによって、そのノードに対して表示するデータベース・オブジェクトのリストをフィルタリングするための「**オブジェクト名フィルター**」が照会されます。

   b. テキストで入力すると、指定されたストリングで始まるデータベース・オブジェクトが表示されます。
4. オブジェクトを強調表示して、「**追加**」をクリックすることによって、オブジェクトを選択する。 選択したオブジェクトが下部ペインに表示されます。

   注: 選択したオブジェクトを除去する必要がある場合は、それを強調表示して「**除去 (Remove)**」をクリックします。
5. ビジネス・オブジェクト ASI を追加する。

   「照会フィルター・プロパティー」ウィンドウで「**ビジネス・オブジェクト ASI の追加**」を選択すると、オブジェクトを追加するたびに、アプリケーション固有の情報を入力するための「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」というウィンドウが表示されます。このウィンドウは、ここで、**ADDRESS** オブジェクトの動詞に関するアプリケーション固有の情報パラメーターとともに示されています。

「**ステータス列名**」フィールドは、論理削除に使用され、「**ステータス値**」は、「ステータス列名」用に選択された属性に対応する列で更新する値を指定します。

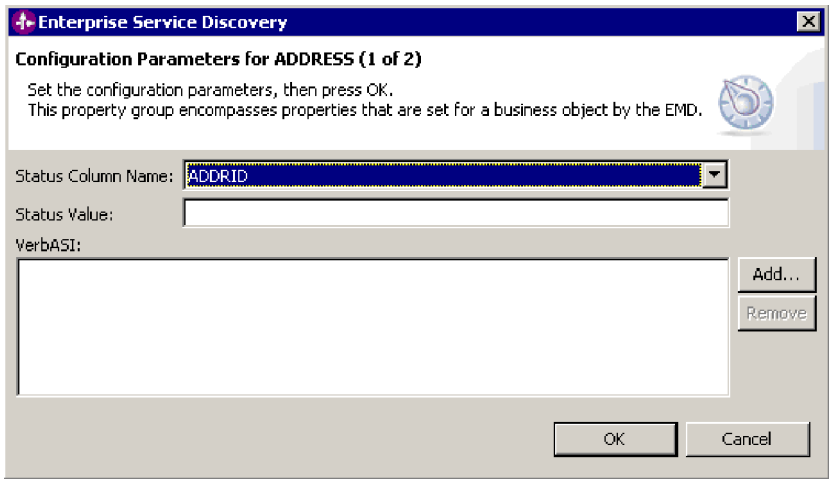


図 22. 「ADDRESS の構成パラメーター (Configuration Parameters for ADDRESS)」ウィンドウ

a. 「**追加**」をクリックします。
b. ストアード・プロシージャーのリストが表示されます。ストアード・プロシージャーを選択して、ストアード・プロシージャーのアプリケーション固有情報を ADDRESS オブジェクトに関連付けます。「**OK**」をクリックします。

   例えば、ストアード・プロシージャーでテーブルにレコードを作成する場合は、「**CreateSP**」構成パラメーターの詳細を入力します。
c. 「**スキーマ**」名、「**ストアード・プロシージャー**」名の値を入力します。ストアード・プロシージャー名を入力すると、「**StoredProcedureParameters**」フィールドが表示されます。 パラメーターごとに、パラメーターに関連付ける値をビジネス・オブジェクト属性のリストから選択します。
d. ASI の入力を完了したら、「**OK**」をクリックします。
e. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**インポートするオブジェクト (Objects to be imported)**」ペイン内に選択されたオブジェクトが表示されます。「**次へ**」をクリックします。

複数の追加オブジェクトを選択した場合は、最初のオブジェクトの「*(オブジェクトの名前)* の構成パラメーター (Configuration Parameters for *(name of object)*)」ウィンドウが表示されます。ASI を入力して「**OK**」をクリックすると、次のオブジェクトのウィンドウが表示され、これが続きます。オブジェクト・レベル、動詞、および属性に関するアプリケーション固有情報については、『アプリケーション固有の情報』を参照してください。

選択済みオブジェクトの構成

データベース・オブジェクトを選択した後、エクスポート・ファイルのメタデータ選択プロパティーの値を指定する必要があります。

始める前に

メタデータ選択プロパティーの詳細については、『参照』セクションを参照してください。

このタスクの実行方法

1. ネーム・スペースを指定します。

   デフォルト値が、「オブジェクトの構成 (Configure objects)」ウィンドウに表示されます。このデフォルト値は、メタデータ・スキーマ JDBCASI.xsd のネーム・スペースです。ネーム・スペースは、ビジネス・オブジェクト・スキーマを論理的に分離するため、ビジネス・オブジェクト名の前に付加されます。
2. 「**サービス・タイプ**」として「**Inbound**」を選択します。

   クエリー・ビジネス・オブジェクトとストアード・プロシージャー・ビジネス・オブジェクトは、サービス・タイプ「Outbound」のみをサポートします。
3. 操作を選択します。
   a. 「オブジェクトの構成 (Configure objects)」ウィンドウの「**操作**」フィールドに、選択したサービス・タイプに対してアダプターがサポートしている操作がリストされます。この操作のリストに追加するには、「**追加**」ボタンをクリックします。

      例えば、操作を削除し、後でそれを組み込む場合は次のようにします。
   b. 「追加」ウィンドウで、操作のリストから選択し、「**OK**」をクリックします。

      指定された操作は、生成されるすべてのビジネス・オブジェクトに対して設定されます。
4. レコード最大数の値を設定しないでください。Inbound 処理には適用されません。
5. 生成された .xsd ファイルが保管されるロケーションへのパスを入力して、BO ロケーションを指定します。
6. 完了したら、「**次へ**」をクリックします。

成果物の生成

メタデータ選択プロパティーを指定したら、特定のデータベースへの通信チャネルをセットアップするためにアダプターが使用するプロパティーを構成します。これらのプロパティーには、リソース・アダプター・プロパティーおよびアクティベーション・スペック・プロパティーが含まれています。また、すべての成果物およびプロパティー値を保管できる新規ビジネス・インテグレーション・モジュールを作成する必要があります。

このタスクの実行方法

1. 新規モジュール名を指定する。

- 下に示されている「成果物の生成 (Generate artifacts)」ウィンドウで、新規モジュール名が「**モジュール (Module)**」フィールドに表示されている場合は、アクションを実行する必要はありません。
- 新規モジュール名が表示されていない場合は、「**新規 (New)**」をクリックする必要があります。「新規統合プロジェクト (New Integration Project)」ウィンドウが表示された場合は、「**モジュール・プロジェクトの作成 (Create a module project)**」を選択して、「**次へ**」をクリックします。「新規モジュール」ウィンドウで、モジュール名を入力して「**終了**」をクリックします。

2. **オプション:** 「成果物の生成 (Generate Artifacts)」ウィンドウで、サービス記述用のフォルダーを指定します。これは、新規モジュール内でサービス記述を保管するフォルダーを入力または参照して行います。フォルダー名を指定しない場合、成果物 (エクスポート・ファイルおよび WSDL ファイル) は、モジュールのルート・フォルダー (すなわちモジュール名のフォルダー) に保管されます。
3. エクスポート・ファイル名は、「**名前**」フィールドに表示されます。「**記述**」フィールドにコメントを追加することもできます。
4. 「**操作名の編集 (Edit Operation names)**」をクリックして、作成しているビジネス・オブジェクトの全操作のデフォルト名を確認します。

   デフォルト名は、オペレーション名とビジネス・オブジェクト名の組み合わせから成ります。
5. 「**コネクターをモジュールとともに配置 (Deploy connector with module)**」チェック・ボックスを選択します。

   注: 「**コネクターをモジュールとともに配置 (Deploy connector with module)**」にチェック・マークを付けて、アダプター・プロジェクトの RAR ファイルが、アプリケーション・サーバーにデプロイする EAR ファイルに組み込まれるようにします。
6. ランタイムに EIS への接続に使用される構成プロパティー値を指定します。
   - 「**ディスカバーされた接続プロパティーを使用 (Use discovered connection properties)**」を選択して、プロパティー値を設定します。
   - プロジェクトのデプロイ後、新規ビジネス・オブジェクトの作成が必要になったときに、「**サーバーで指定された接続プロパティーを使用 (Use connection properties specified on server)**」を選択することができます。この選択は、アプリケーション・サーバー上に既にあるプロパティーを使用することを示します。
7. プロパティー値を設定します。必須のプロパティー・フィールドにはアスタリスクが付いています。

   Inbound 処理の場合は、アクティベーション・スペック・プロパティーおよびリソース・アダプター・プロパティー用のプロパティー・フィールドが表示されます。

   双方向変換を活動化している場合は、Inbound 処理用の双方向プロパティー・フィールドが表示されます。

   これらの構成プロパティーの詳細については、『参照』セクションの『アダプター構成プロパティー』を参照してください。

8. 「**J2C 認証データ・エントリー (J2C Authentication Data Entry)**」フィールドで、アプリケーション・サーバー上にセットアップする認証別名を指定します。

   詳しくは、『認証別名の作成』を参照してください。
9. 「**JNDI ルックアップ名 (JNDI Lookup Name)**」フィールドで、アプリケーション・サーバー上で使用する接続ファクトリーの名前を指定します。

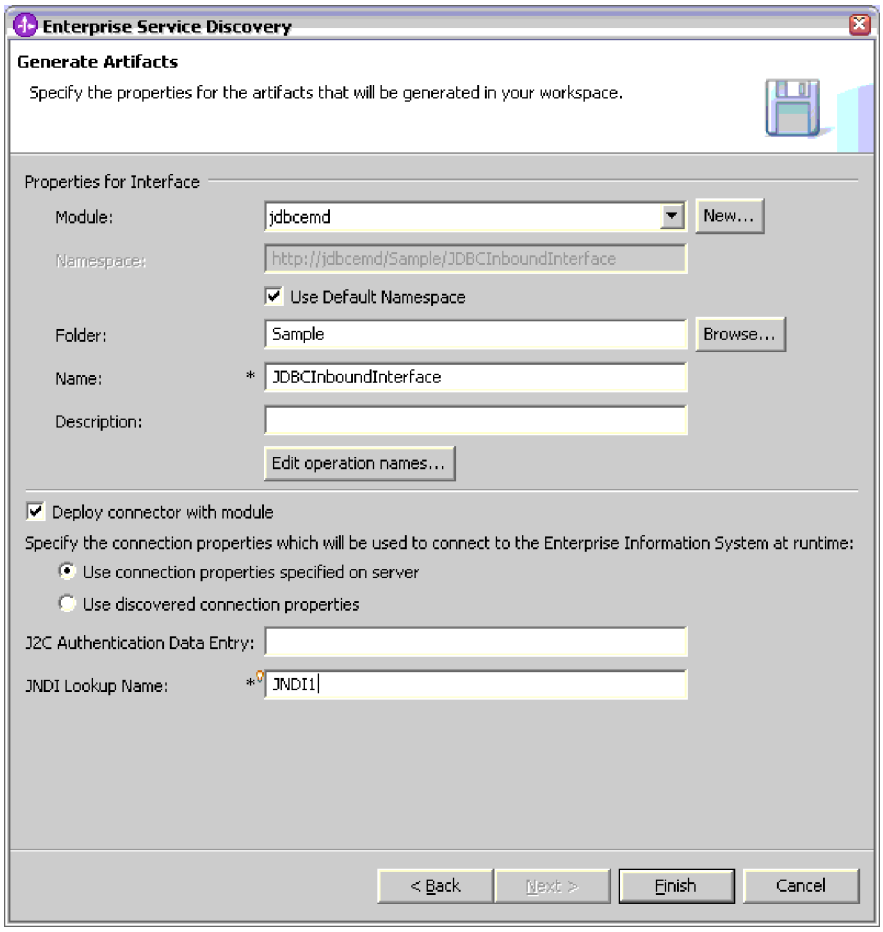


図 23. 「成果物の生成 (Generate Artifacts)」ウィンドウ

参照バインディングの生成

参照バインディングは、その他の WebSphere Business Integration Service Component Architecture (SCA) コンポーネントがアダプターにアクセスするために使用します。その他のサーバー・プロセスへアダプターをリンクしてプロジェクト・モジュールからアダプターへの参照を作成します。

このタスクの概説

参照バインディング生成は、テスト環境でのみ必須です。実稼働環境でアダプターを配置する際には必要ありません。

このタスクの実行方法

1. SCA コンポーネントを作成する

a. WebSphere Integration Developer の「**ビジネス・インテグレーション (Business Integration)**」パースペクティブに移動します。「**ビジネス・インテグレーション**」タブで、「**JDBCEMD**」モジュールを右クリックし、「**アプリケーションから開く**」>「**アセンブリー・エディター**」を選択します。

b. 「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウが表示され、モジュールのエクスポート・コンポーネントがビューに示されます。新規コンポーネントを作成するには、ウィンドウの左側 (縦) のフレームで最初のアイコン (コンポーネント) をクリックします。

c. アイコンの新規メニューの最初のアイコンをクリックします (これには、「**実装タイプなしのコンポーネント (Component with no implementation type)**」という吹き出しヘルプがあります)。カーソルが配置アイコンに変わります。

d. パレットをクリックして、新規コンポーネントを「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウに追加します。

2. コンポーネントを実装タイプなしで作成します。

a. モジュールのエクスポート・コンポーネントをクリックして新規コンポーネントにドラッグします。これにより、エクスポート・コンポーネントから新規コンポーネントへ線が引かれます。

b. 「線の追加 (Add Wire)」ウィンドウで、「**OK**」をクリックします。

c. 「アセンブリー・ダイアグラム (Assembly Diagram)」ウィンドウに、新規コンポーネントが、モジュールのエクスポート・コンポーネントに接続する「線」とともに表示されます。

「**ファイル**」>「**保管**」をクリックしてアセンブリー・ダイアグラムを保管します。

3. 新規コンポーネントの実装を生成します。

a. 「**実装タイプなしのコンポーネント (Component with no implementation type)**」を右クリックし、「**実装の生成 . . . (Generate implementation . . .)**」 → 「**Java**」を選択します。

b. 「**デフォルト・パッケージ**」を選択して、「**OK**」をクリックします。 これにより、インバウンド・モジュールのエンドポイントが作成されます。

c. **オプション:** エンドポイント・メソッドのそれぞれのエンドポイントで受信されたデータ・オブジェクトを出力するため、print ステートメントを追加します。

d. 「**ファイル**」 → 「**保管**」をクリックしてアセンブリー・ダイアグラムを保管します。

詳細については、http://www.ibm.com/software/integration/wid にある WebSphere Integration Developer のユーザー・ガイドを参照してください。

第 8 章 モジュールのデプロイ

モジュールをアプリケーション・サーバーにデプロイするには、アダプター・プロジェクトをエンタープライズ・アーカイブ (EAR) ファイルとしてエクスポートし、モジュールをインストールし、その後、エンタープライズ・サービス・ディスカバリー・ウィザードで設定されていなかった任意の構成プロパティーを追加します。

プロジェクトを EAR ファイルとしてエクスポート

エンタープライズ・サービス・ディスカバリー・ウィザードを使用して、EAR ファイルとして作成したアダプター・プロジェクトをエクスポートします。 EAR ファイルを作成することによって、アダプター・プロジェクトのすべてのコンテンツをアプリケーション・サーバーに容易にデプロイできる形式で取り込みます。

始める前に

プロジェクトを EAR ファイルとしてエクスポートするには、その前にビジネス・オブジェクトを選択して構成し、成果物を生成しておく必要があります。

このタスクの概説

プロジェクト・ファイルは、WebSphere Integration Developer のワークスペース内の J2EE コネクター・プロジェクトです。プロジェクトを EAR ファイルとしてローカル・ファイル・システムにエクスポートするには、以下の手順を実行してください。

このタスクの実行方法

1. WebSphere Integration Developer で、「選択」ウィンドウの「エクスポート」パネルで、リストから「**EAR ファイル**」を選択し、「**次へ**」をクリックします。
2. 「EAR エクスポート」ウィンドウの「**EAR プロジェクト**」フィールドで、ビジネス・インテグレーション・モジュールを選択します。 これにより、モジュール名 **JDBCEMD** にサフィックス **App** が付けられます。
3. EAR ファイルを作成するロケーションを入力するか、参照します。すべてのチェック・ボックスをクリックし、EAR ファイルで作成したものをすべて組み込みます。「**終了**」をクリックします。

結果

アダプター・プロジェクトが、EAR ファイルにエクスポートされます。

次の作業

サーバーの管理コンソールを使用して、モジュールをインストールします。これにより、モジュールがアプリケーション・サーバーにデプロイされます。

© Copyright IBM Corp. 2005, 2006

モジュールのインストール

アダプター・プロジェクトのインストールは、デプロイメント・プロセスの最終手順です。アダプター・プロジェクトをサーバーにインストールして実行すると、プロジェクト・モジュールの一部として組み込まれているアダプターが、インストール済みアプリケーションの一部として稼働します。

始める前に

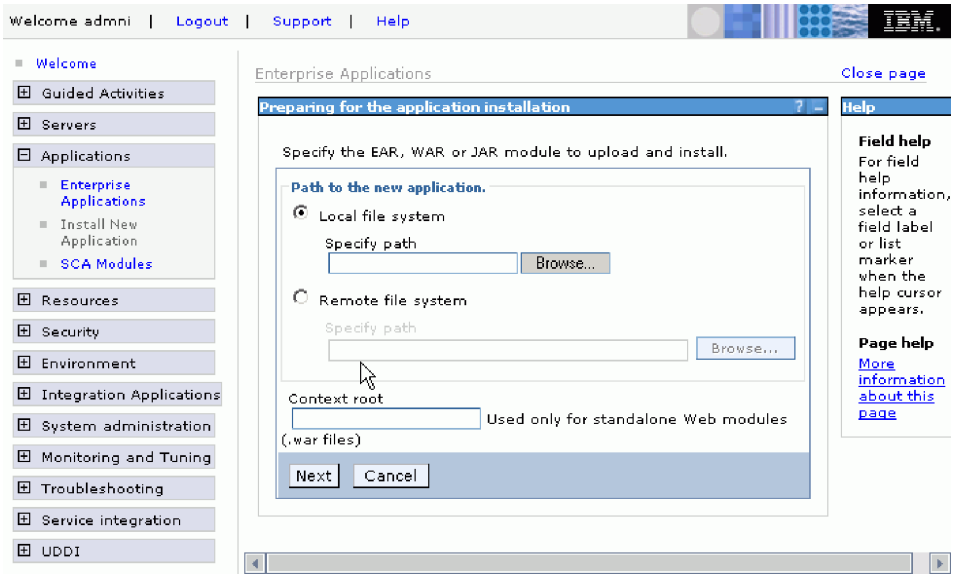
アダプター・プロジェクトをインストールする前に、プロジェクト・モジュールを EAR ファイルとしてエクスポートしておく必要があります。

このタスクの概説

アダプター・モジュールをインストールするには、次の手順を実行します。アダプター・プロジェクト・アプリケーションのクラスター化については、http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jspを参照してください。

このタスクの実行方法

1. サーバー・インスタンスを右クリックし、「**管理コンソールの実行**」を選択して、WebSphere Process Server 管理コンソールを開きます。
2. 管理コンソール・ウィンドウで、「**アプリケーション**」 → 「**新規アプリケーションのインストール**」をクリックします。

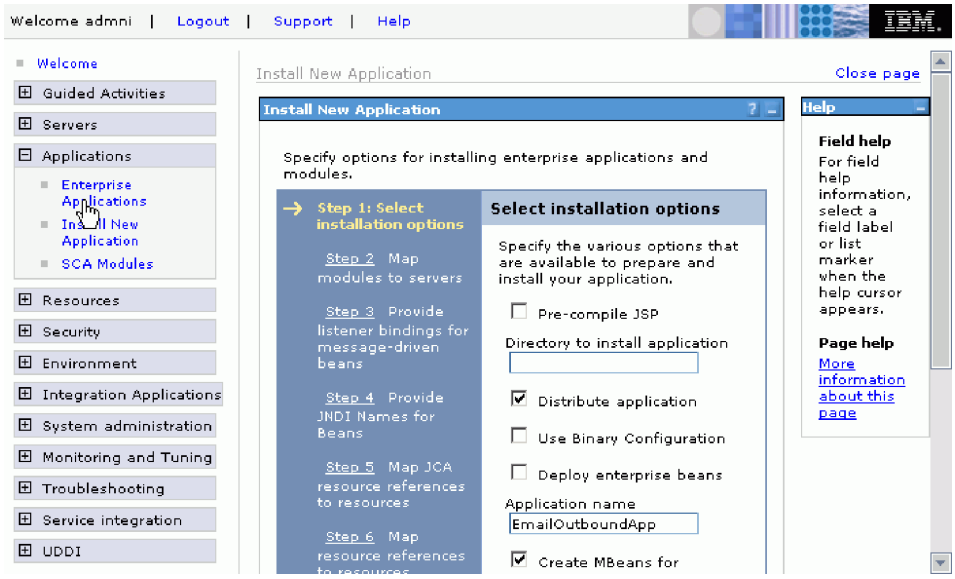


「アプリケーション・インストールの準備 (Preparing for the application installation)」ウィンドウ

3. 「**参照**」をクリックして、EAR ファイルを位置指定し、「**次へ**」をクリックします。
4. **オプション:** クラスター環境にデプロイする場合は、「ステップ 2: サーバーへのモジュールのマッピング (Step 2: Mapping modules to servers)」が表示されるまで、「**次へ**」を繰り返しクリックします。次に、「**モジュール**」とサーバー・クラスター名を選択し、「**適用**」をクリックします。注: アダプター・インスタンスは enableHASupport が真に設定されている場合に、クラスター化され

た環境に複製されます。単一サーバー環境の enableHASupport の値を変更しないでください。 **注:** アダプター・インスタンスは、**enableHASupport** が真に設定されている場合に、クラスター化された環境に複製されます。単一サーバー環境の **enableHASupport** の値を変更しないでください。

5. 「ステップ 6: リソース参照をリソースにマップ (Step 6: Map resource reference to resources)」が表示されるまで、「**次へ**」をクリックします。



「新規アプリケーションのインストール (Install New Application)」ウィンドウ

6. 選択認証データ・エントリー・リストから「**SCA 認証別名 (SCA Auth Alias)**」を選択します。
7. モジュールのチェック・ボックスを選択し、「**適用 (Apply)**」をクリックします。
8. 「**次へ**」をクリックします。 すべてのインストール・オプションの要約が表示されます。
9. すべてのオプションが正しいことを確認し、「**終了**」をクリックします。
10. アプリケーションが正常にインストールされたことを確認します。
11. インストール・メッセージのリストの最後にある「**マスター構成に保管**」リンクをクリックします。
12. 「**保管**」をクリックします。

結果

プロジェクトがデプロイされ、デプロイされたアプリケーションの「エンタープライズ・アプリケーション」ウィンドウが表示されます。

次の作業

リソース・アダプター、管理接続ファクトリー、アクティベーション・スペック、またはデータ形式変更の各プロパティーを設定またはリセットする場合、もしくはアダプター・プロジェクト・アプリケーションをクラスター化する場合は、トラブルシューティング・ツールを構成する前に、WebSphere Process Server 管理コンソールを使用して、これらの作業を行う必要があります。

構成設定の管理コンソールからの設定または変更

アダプター・プロジェクトがサーバー上にデプロイされたら、必要に応じて、特定のデータベース・アプリケーションへの通信チャネルをセットアップするためにアダプターが使用するプロパティーを再構成することができます。次に、トラブルシューティング・ツールを構成して、アダプターを開始することができます。

このタスクの概説

通常、構成プロパティーは、アダプター・プロジェクトの作成時にエンタープライズ・サービス・ディスカバリー・ウィザードを使用して設定されます。サーバー上の管理コンソールを使用してプロパティーを再設定するか、エンタープライズ・サービス・ディスカバリー・ウィザードで設定した値に従って各プロパティーが取り込まれているかどうかを確認します。

リソース・アダプター・プロパティーの設定

リソース・アダプター・プロパティーは、ロギングおよびトレース、双方向言語サポート、さらにはアダプターのデフォルト構成プロパティーなどのアダプター固有のアクティビティーから構成されます。管理コンソールを使用してプロパティーを再設定するか、あるいは単に、値がエンタープライズ・サービス・ディスカバリー・ウィザードを使用して設定したものと一致しているかどうかを確認します。

このタスクの実行方法

1. アダプター・プロジェクトを開く
   a. 管理コンソールで、「構成」ペインにアダプター・プロジェクトの名前が表示されます。「**関連項目**」見出しの下で「**コネクター・モジュール**」を選択します。
   b. プロジェクト RAR ファイル名が表示されます。ファイル名の横にあるチェック・ボックスをクリックします。
   c. 「**追加プロパティー**」の下で「**リソース・アダプター**」を選択します。
2. リソース・アダプター・プロパティー値を編集する
   a. アダプター・プロジェクトが「**一般プロパティー**」 → 「**名前**」の下に表示されます。「**追加プロパティー**」の下で、「**カスタム・プロパティー**」をクリックします。
   b. リソース・アダプター・プロパティーとその値のリストが表示されます。必要に応じてプロパティーを編集します。 『参照』セクションの『リソース・アダプター・プロパティー』でプロパティーとその説明を参照してください。

      注: 「**カスタム・プロパティー**」リストのアダプター・レベルで 「**データベース・ベンダー**」プロパティーを探します。 このプロパティーは、アプリケーションを開始するために必要です。このプロパティーがブランクの場合は値を入力して保管します。

管理 (J2C) 接続ファクトリー・プロパティーの設定

管理 (J2C) 接続ファクトリー・プロパティーは、エンタープライズ情報システムとの Outbound 接続インスタンスを作成するためにランタイムに使用されます。管理

コンソールを使用してプロパティーを再設定するか、あるいは単に、値がエンタープライズ・サービス・ディスカバリー・ウィザードを使用して設定したものと一致しているかどうかを確認します。

このタスクの実行方法

1. アダプター・プロジェクトを開く。
   a. 管理コンソールで、「構成」ペインにアダプター・プロジェクトの名前が表示されます。「**関連項目**」見出しの下で「**コネクター・モジュール**」を選択します。
   b. プロジェクト RAR ファイル名が表示されます。ファイル名の横にあるチェック・ボックスをクリックします。
   c. 「**追加プロパティー**」の下で「**リソース・アダプター**」を選択します。
2. 管理 (J2C) 接続ファクトリー・プロパティーの値を編集します。
   a. アダプター・プロジェクトが「**一般プロパティー**」>「**名前**」の下に表示される。「**追加プロパティー**」で、「**J2C 接続ファクトリー**」をクリックします。
   b. アダプター・プロジェクト名が、EJB プロジェクトで指定した JNDI 名とともに表示されます。アダプターのファクトリー接続の横にあるチェック・ボックスをクリックします。
   c. J2C 接続ファクトリーのプロパティーと値が表示されます。必要に応じてプロパティーの値を編集します。 これらのプロパティーの詳細については、『参照』セクションの『管理 (J2C) 接続ファクトリー・プロパティー』を参照してください。

EIS のアクティベーション・スペック・プロパティーの設定

アクティベーション・スペック・プロパティーは、Inbound 処理に必要です。管理コンソールを使用してプロパティーを再設定するか、あるいは単に、値がエンタープライズ・サービス・ディスカバリー・ウィザードを使用して設定したものと一致しているかどうかを確認します。

このタスクの実行方法

1. アダプター・プロジェクトを開く
   a. 管理コンソールで、「構成」ペインにアダプター・プロジェクトの名前が表示されます。「**関連項目**」見出しの下で「**コネクター・モジュール**」を選択します。
   b. プロジェクト RAR ファイル名が表示されます。ファイル名の横にあるチェック・ボックスをクリックします。
   c. 「**追加プロパティー**」の下で「**リソース・アダプター**」を選択します。
2. アダプター・プロジェクトが「**一般プロパティー**」>「**名前**」の下に表示される。「**追加プロパティー**」で、「**J2C 活動化仕様**」をクリックします。

   これらのプロパティーについて詳しくは、『参照』セクションの『アクティベーション・スペック・プロパティー』を参照してください。

第 9 章 トラブルシューティング・ツールの構成

要件に合うようトラブルシューティング・ツールを構成します。アダプターのロギングを使用可能にし、イベント処理の状況を制御します。 Common Event Infrastructure を使用可能にし、アダプターに関する診断情報を収集します。トレース・レベルを設定し、アダプターのログ・ファイルおよびトレース・ファイルに収集される情報のレベルを決定します。サポートに関する情報に素早くアクセスしたり、IBM ソフトウェア製品の問題判別を行うための保守サービス・ツールを得られるように、IBM Support Assistant をインストールします。

Common Event Infrastructure (CEI) を使用したトレースの使用可能化

Common Event Infrastructure (CEI) を構成して、トレースを使用可能にし、アダプターの詳細レベルを制御します。

始める前に

CEI を使用してトレースを使用可能にする前に、次のタスクを実行してください。

- 診断トレース・サービスを使用可能にする。
- IBM WebSphere Adapters イベント定義ファイルを CEI カタログにパブリッシュして、これらのイベント定義を設定できるようにする。

これらのタスクの実行方法の説明については、ご使用のサーバーの Web サイトにある CEI の文書を参照してください。

- WebSphere Process Server の場合: http://www.ibm.com/software/integration/wps
- WebSphere Enterprise Service Bus の場合: http://www.ibm.com/software/integration/wsesb

トレースを使用可能にし、トレースの詳細レベルを制御するには、以下の手順を実行します。

このタスクの実行方法

1. 管理コンソールにて、「**トラブルシューティング**」をクリックします。
2. 「**ログおよびトレース**」を選択します。
3. サーバーのリストにて、ご使用のサーバーの名前をクリックします。
4. 「一般プロパティー」領域で「**ログ詳細レベルの変更**」をクリックし、アダプター・コンポーネント用の **com.ibm.j2ca.*** を選択します。 次の表に記載しているように、アダプター・タイプごとにサブコンポーネントがあります。

アダプター	パッケージ名
WebSphere Adapter for Email	com.ibm.j2ca.email.*
WebSphere Adapter for Flat Files	com.ibm.j2ca.flatfile.*
WebSphere Adapter for FTP	com.ibm.j2ca.ftp.*
WebSphere Adapter for JDBC	com.ibm.j2ca.jdbc.*
WebSphere Adapter for JD Edwards EnterpriseOne	com.ibm.j2ca.jde.*

© Copyright IBM Corp. 2005, 2006

アダプター	パッケージ名
WebSphere Adapter for SAP Software	com.ibm.j2ca.sap.*
WebSphere Adapter for Siebel Business Applications	com.ibm.j2ca.siebel.*

5. ご使用のアダプターと一致するコンポーネントを選択します。各アダプター・コンポーネントには、ロギング用と CEI 用の 2 つのサブコンポーネントがあります。次のとおりです。
   - *subcomponent_name*.log.*adapter_ID*
   - *subcomponent_name*.cei.*adapter_ID*

   例えば、com.ibm.j2ca.siebel.cei.*adapter_ID1*。デプロイされたアダプターのインスタンスごとに、システムは別々の ID を表示します。
6. 使用可能にする CEI アダプター ID を選択します。
7. サービス・コンポーネント・イベントで収集するビジネス・オブジェクトの詳細レベルをリストから選択します。
   - **オフ**。CEI をオフにします。
   - **詳細**。 CEI をオンにしますが、ビジネス・オブジェクト・ペイロードはパブリッシュしません。これは、WebSphere Integration Developer のイベント制御の詳細レベル、「空」に相当します。
   - **より詳細**。 CEI をオンにし、ビジネス・オブジェクトのペイロードの説明のみをパブリッシュします。これは、WebSphere Integration Developer のイベント制御の詳細レベル、「ダイジェスト」に相当します。
   - **極めて詳細**。 CEI をオンにし、ビジネス・オブジェクト・ペイロードをすべてパブリッシュします。これは、WebSphere Integration Developer のイベント制御の詳細レベル、「フル」に相当します。
   - **すべて**。「**高 (finest)**」と同じです。

   各イベント・コンテンツ・レベルの意味 (空、ダイジェスト、フル)、および共通ベース・イベント・モデルと Common Event Infrastructure の使用法の詳細については、ご使用のプロセス・サーバーの資料を参照してください。

ロギング・プロパティーの構成

ログを使用可能にし、ログの出力プロパティー (ログのロケーション、詳細レベル、出力形式など) を設定するには、管理コンソールを使用します。

このタスクの概説

アダプターでモニター対象イベントをログに記録できるようにするには、モニターしたいサービス・コンポーネントのイベント・ポイント、イベントごとに必要となる詳細レベル、およびイベントをログにパブリッシュするのに使用する出力のフォーマットを指定する必要があります。管理コンソールを使用して、次のタスクを実行します。

- 特定のイベント・ログを使用可能または使用不可に設定する
- ログの詳細レベルを指定する
- ログ・ファイルの保管場所と保持数を指定する。
- ログの出力形式を指定する。

ログ・アナライザーの出力形式を設定した場合は、ログ・アナライザー・ツール (プロセス・サーバーに同梱されるアプリケーション) を使用して、トレース出力を開くことができます。これは、2 つの異なるサーバー・プロセスからのトレースを相関しようとする場合に便利です。なぜなら、これにより、ログ・アナライザーのマージ機能が使用できるからです。

プロセス・サーバー (サービス・コンポーネントとイベント・ポイントを含む) のモニターの詳細については、ご使用のプロセス・サーバーの資料を参照してください。

ログ構成は、静的または動的に変更できます。アプリケーション・サーバーを開始または再始動すると、静的構成が有効になります。動的構成 (実行時構成) の変更は、直ちに適用されます。

ログが作成されると、そのログの詳細レベルが構成データから設定されます。特定のログ名に対して、構成データが使用可能でない場合、そのログのレベルは、ログの親から取得されます。親ログに構成データが存在しない場合、そのログの親が確認される、という具合に、ヌル以外のレベル値があるログが見つかるまでツリーを上昇します。ログのレベルを変更すると、その変更はログの子に伝搬されます。また、必要に応じて、ログの子からその子へと変更が再帰的に伝搬されます。

ロギングを使用可能にし、ログの出力プロパティーを設定するには、以下の手順を実行します。

このタスクの実行方法

1. 管理コンソールのナビゲーション・ペインで、「**サーバー**」 → 「**アプリケーション・サーバー**」をクリックします。
2. 作業したいサーバーの名前をクリックします。
3. 「**トラブルシューティング**」で「**ログおよびトレース**」をクリックします。
4. 「**ログ詳細レベルの変更**」をクリックします。
5. 変更を有効にするには、以下を行います。
   - 構成を静的に変更する場合は、「**構成**」タブをクリックします。
   - 構成を動的に変更する場合は、「**ランタイム**」タブをクリックします。
6. 変更したいロギング・レベルのパッケージを選択します。 WebSphere Adapters 用のパッケージ名は、**com.ibm.j2ca** で始まります。
   - アダプターの基本コンポーネントの場合は、**com.ibm.j2ca.base** を選択します。
   - アダプターの基本コンポーネントとすべてのデプロイ済みアダプターの場合は、 **com.ibm.j2ca.base.*** を選択します。
   - 特定のアダプターの場合は、そのパッケージ名を選択します。

アダプター	パッケージ名
WebSphere Adapter for Email	com.ibm.j2ca.email
WebSphere Adapter for Flat Files	com.ibm.j2ca.flatfile
WebSphere Adapter for FTP	com.ibm.j2ca.ftp
WebSphere Adapter for JDBC	com.ibm.j2ca.jdbc

アダプター	パッケージ名
WebSphere Adapter for JD Edwards EnterpriseOne	com.ibm.j2ca.jde
WebSphere Adapter for SAP Software	com.ibm.j2ca.sap
WebSphere Adapter for Siebel Business Applications	com.ibm.j2ca.siebel

7. パッケージ名をクリックし、ロギング・レベルを選択します。

ロギング・レベル	説明
致命的	タスクを続行できない。または、コンポーネントが機能しない。
重大	タスクを続行できないが、コンポーネントは機能する。このロギング・レベルには、差し迫った致命的エラーを示す (すなわち、リソースが枯渇寸前であることを強く示唆する) 状況も含まれる。
警告	潜在的なエラーが発生したか、重大エラーが差し迫っている。このロギング・レベルには、例えばリソース・リークの可能性など、進行性の障害を示す状況も含まれる。
監査	サーバーの状態やリソースに影響を与える重大なイベントが発生した。
情報	タスクが稼働中である。このロギング・レベルには、タスクの全体的な進行を概説する一般情報が含まれる。
構成	構成の状況が報告されるか、構成変更が発生した。
詳細	サブタスクが稼働中である。このロギング・レベルには、サブタスクの進行を詳細に説明した一般情報が含まれる。

8. 「**適用 (Apply)**」をクリックします。
9. 「**OK**」をクリックします。
10. 静的な構成変更を有効にするには、プロセス・サーバーを停止し、再始動します。

ログ・ファイル名およびトレース・ファイル名の変更

デフォルトでは、プロセス・サーバー上にあるすべてのプロセスおよびアプリケーションのログ情報およびトレース情報は、それぞれ SystemOut.log ファイルおよび trace.log ファイルに書き込まれます。アダプター・ログおよびトレース情報を他のプロセスとは分離して保持するには、管理コンソールを使用してファイル名を変更します。

このタスクの概説

アダプター・モジュールをアプリケーション・サーバーにデプロイした後は、ログ・ファイル名およびトレース・ファイル名はいつでも変更できます。

ログ構成は、静的または動的に変更できます。アプリケーション・サーバーを開始または再始動する際、静的構成変更がアプリケーションに反映されます。動的構成変更またはランタイム構成変更は、即座に適用されます。

ログ・ファイルおよびトレース・ファイルは、*install_root*/profiles/*profile_name*/logs/*server_name* フォルダーにあります。

ログ・ファイル名およびトレース・ファイル名を設定または変更するには、次の手順を実行します。

このタスクの実行方法

1. ナビゲーション・ペインで、「**エンタープライズ・アプリケーション**」をクリックします。
2. アダプター・アプリケーションの名前をクリックします。これは、アダプターの EAR ファイル名から .ear ファイル拡張子を除いたものです。 例えば、EAR ファイルの名前が Accounting_OutboundApp.ear である場合は、**Accounting_OutboundApp** をクリックします。
3. 「**コネクター・モジュール**」をクリックします。
4. アダプターの RAR ファイルの名前をクリックし、アダプターを選択します。RAR ファイルを以下の表に示します。

アダプター	RAR ファイル名
WebSphere Adapter for Email	CWYEM_Email.rar
WebSphere Adapter for Flat Files	WYFF_FlatFile.rar
WebSphere Adapter for FTP	CWYFT_FTPFile.rar
WebSphere Adapter for JDBC	CWYBC_JDBC.rar
WebSphere Adapter for JD Edwards EnterpriseOne	CWYED_JDE.rar
WebSphere Adapter for SAP Applications	CWYAP_SAPAdapter.rar CWYAP_SAPAdapterTX.rar
WebSphere Adapter for Siebel Business Applications	CWYEM_Siebel.rar

5. リソース・アダプターの名前をクリックします。
6. 「カスタム・プロパティー」領域で、次のようにファイル名を指定します。
   - ログ・ファイル名を変更するには、「**ログ・ファイル名**」の「**値**」フィールドに名前を入力します。デフォルトでは、このログは SystemOut.log ファイルにあります。
   - トレース・ファイル名を変更するには、「**トレース・ファイル名**」の「**値**」フィールドに名前を入力します。デフォルトでは、このログは trace.log ファイルにあります。
7. 静的な構成変更を有効にするには、プロセス・サーバーを停止し、再始動します。

IBM Support Assistant のインストールまたはアップグレード

IBM Support Assistant (ISA) は、お客様が IBM ソフトウェア製品に関する質問や問題を解決できるよう支援する、無料でローカルなソフトウェア保守ワークベンチです。インストールした製品用のプラグインをインストールしてください。ここでは、サポートに関する情報に素早くアクセスしたり、問題判別を行うための保守サービス・ツールを得られます。IBM Support Assistant のインストールとアップグレードは、簡単に行うことができます。

このタスクの概説

IBM Support Assistant では、次のサービスを提供しています。

- 症状に基づいたデータ収集
- 統合検索インターフェースを使用して、IBM サポート情報、IBM ニュースグループといったリソース (一度の検索で複数のリソース) にアクセス
- IBM 研修資料への簡単なアクセス
- 便利なリンクを使用した、IBM 製品ホーム・ページ、製品サポート・ページ、および製品フォーラムやニュースグループへの簡単なアクセス
- ISA プラグインおよびツールを容易に更新およびインストールするためのツール・フレームワークおよび更新マネージャー
- 重要なシステム・データを IBM に電子的に送信することで、問題管理レコードを素早く解決

IBM Support Assistant のバージョン 2 とバージョン 3 の両方を単一のコンピューターにインストールして実行することにより、さまざまな IBM ソリューションのサポートを利用できます。

IBM Support Assistant をインストールおよびアップグレードするには、以下の手順を実行します。

このタスクの実行方法

1. 以下の IBM Support Assistant Web ページにアクセスします。

   http://www.ibm.com/software/support/isa/
2. Web ページの説明に従って ISA バージョン 3.0 をダウンロードした後、ツールを抽出、インストール、および使用します。
3. ISA を始動します。
4. **更新プログラム**・コンポーネントを開きます。
5. 「**アップグレード**」タブで、ISA をバージョン 3.0.1 以上にアップグレードします。
6. 「**新しい製品およびツール (New Products and Tools)**」タブで、アダプターのプラグインをインストールします。 WebSphere 製品のリストから、アダプターのプラグインを選択します。アダプターごとに、オプションの言語パック・プラグインがあります。これを使用すると、英語以外の言語でアプリケーション固有の情報を表示できます。

第 10 章 アダプターの管理

アダプターを開始、停止、およびトラブルシューティングするには、サーバーの管理コンソールを使用します。

アダプターの開始

「停止中 (Stopped)」の状況にあるアダプターを開始するには、管理コンソールを使用します。デフォルトでは、アダプターは、サーバーの開始時に自動的に開始します。

始める前に

このタスクを完了させるために、サーバーの管理コンソールを実行しておく必要があります。

アダプターを開始するには、次の手順を実行します。

このタスクの実行方法

1. 「エンタープライズ・アプリケーション」ページで、「**アプリケーション**」 → 「**エンタープライズ・アプリケーション**」をクリックします。
2. 開始したいアダプターのチェック・ボックスを選択します。
3. 「**開始**」をクリックします。

結果

アダプターの状況が「始動済み」に変化し、アダプターが開始したことを示すメッセージがページの上部に表示されます。

サーバーの管理コンソールを使用して、アダプターを停止します。

アダプターの停止

アダプターを停止するには、サーバーの管理コンソールを使用します。

始める前に

このタスクを完了させるために、サーバーの管理コンソールを実行しておく必要があります。

アダプターを停止するには、次の手順を実行します。

このタスクの実行方法

1. 「エンタープライズ・アプリケーション」ページで、「**アプリケーション**」 → 「**エンタープライズ・アプリケーション**」をクリックします。
2. 停止したいアダプターのチェック・ボックスをクリアします。
3. 「**停止 (Stop)**」をクリックします。

© Copyright IBM Corp. 2005, 2006

結果

アダプターの状況が「停止中 (Stopped)」に変化し、アダプターが停止したことを示すメッセージがページの上部に表示されます。

サーバーの管理コンソールを使用して、アダプターをトラブルシューティングします。

トラブルシューティングとサポート

共通のトラブルシューティング手法とセルフ・ヘルプ情報は、問題を迅速に識別して解決するのに役立ちます。必要な場合は、IBM ソフトウェア・サポートへの連絡手順に従ってください。

例外: XAResourceNotAvailableException

com.ibm.ws.Transaction.XAResourceNotAvailableException 例外の報告がプロセス・サーバーのログに繰り返し含まれているときは、トランザクション・ログを除去し、問題を訂正してください。

症状:

アダプターが始動すると、プロセス・サーバーのログ・ファイルに以下の例外が繰り返し記録されます。

com.ibm.ws.Transaction.XAResourceNotAvailableException

問題:

プロセス・サーバーがリソースのトランザクションをコミットまたはロールバックしている間に、そのリソースが除去されました。アダプターは、始動するとトランザクションのリカバリーを試みますが、リソースが除去されているため、それができません。

解決策:

この問題を訂正するには、以下の手順を実行します。

1. プロセス・サーバーを停止します。
2. そのトランザクションを含むトランザクション・ログ・ファイルを削除します。例外トレース内の情報を使用して、トランザクションを識別します。これにより、サーバーは、それらのトランザクションのリカバリーを試みないようになります。

   注: テスト環境または開発環境では、通常はトランザクション・ログをすべて削除できます。 WebSphere Integration Developer では、トランザクション・ログ・ディレクトリー *server_install_directory*¥profiles¥*profile_name*¥tranlog に含まれるファイルとサブディレクトリーを削除します。

   実稼働環境では、処理する必要のないイベントを表すトランザクションのみを削除します。これを行う方法の一つは、アダプターを再インストールし、使用した元のイベント・データベースをそのアダプターに参照させ、不要な

トランザクションのみを削除することです。もう一つの方法は、以下のディレクトリー内の log1 ファイルまたは log2 ファイルからトランザクションを削除することです。

server_install_directory¥profiles¥*profile_name*¥tranlog¥*node_name*¥wps¥*server_name*¥transaction¥tranlog

3. プロセス・サーバーを始動します。

セルフ・ヘルプ・リソース

IBM ソフトウェア・サポートのセルフ・ヘルプ・リソースを使用すると、最新のサポート情報や技術文書を入手したり、サポート・ツールやフィックスをダウンロードしたり、 WebSphere Adapter for YOUR ADAPTER NAME に関する問題を防止したりできます。セルフ・ヘルプ・リソースは、アダプターの問題を診断し、IBM ソフトウェア・サポートに連絡する際にも役立ちます。

WebSphere Adapters のソフトウェア・サポート Web サイト (http://www.ibm.com/software/integration/wbiadapters/supp) では、以下のリソースを入手できます。

- フラッシュ (テクニカル・サポートからのアラート)
- テクニカル・ノート

  WebSphere Adapters のテクニカル・ノートのリストは、http://www.ibm.com/support/search.wss?rs=695&tc=SSMKUK で入手できます。
- プログラム診断依頼書 (APAR)
- 製品のインフォメーション・センター、マニュアル、IBM Redbooks™、およびホワイト・ペーパーを含む技術情報。
- 教育関連のオファリング
- *IBM ソフトウェア・サポート・ハンドブック*

このサイトに登録すると、My Support を使用して、ご自分の用途に合わせてサポート・ページをカスタマイズできます。

IBM ソフトウェア・サポートへの連絡

IBM ソフトウェア・サポートでは、オンラインまたは電話にて、WebSphere Adapters のサポートが提供されています。 IBM ソフトウェア・サポートに連絡する前に、問題に関する情報を収集しておくと、サポートの対応速度が大幅に向上します。

始める前に

問題が不良に関係するものだと思われる場合は、IBM ソフトウェア・サポートが支援いたします。IBM ソフトウェア・サポートに問い合わせる前に、お客様の会社が有効な IBM ソフトウェア保守契約を締結し、お客様が IBM に問題を送信する許可ユーザーである必要があります。必要なソフトウェア保守契約のタイプは、所有している製品のタイプによって異なります。

- IBM が提供するソフトウェア製品 (Tivoli、 Lotus®、 Rational® 製品、Windows、 Linux®、または UNIX オペレーティング・システム上で稼働する

DB2 および WebSphere 製品を含むが、これらに限定されない) については、以下のいずれかの方法でパスポート・アドバンテージに登録してください。次のいずれかの方法で登録できます。

オンライン

米国のパスポート・アドバンテージの Web ページ (http://www-306.ibm.com/software/support/pa.html) にアクセスし、「**How to Enroll**」をクリックします。

電話 お住まいの国で使用できる電話番号を確認するには、Web の IBM Software Support Handbook のお問い合わせページ (http://techsupport.services.ibm.com/guides/contacts.html) にアクセスし、地域名をクリックします。

- IBM eServer™ ソフトウェア製品 (zSeries®、pSeries®、および iSeries™ 環境で稼働する DB2 および WebSphere 製品を含むが、これらに限定されない) については、IBM 営業担当員または IBM ビジネス・パートナーを通してソフトウェア保守契約を購入できます。eServer ソフトウェア製品のサポートについて詳しくは、IBM Technical Support Advantage の Web ページ (http://www-03.ibm.com/servers/eserver/techsupport.html) にアクセスしてください。

必要なソフトウェア保守契約のタイプが不明な場合は、アメリカ合衆国の 1-800-IBMSERV (1-800-426-7378) までお問い合わせください。その他の国のお客様は、IBM Software Support Handbook の Web 窓口ページ (http://techsupport.services.ibm.com/guides/contacts.html) にアクセスし、お客様がお住まいの地域名をクリックして担当のサポート窓口の電話番号をご確認ください。

このタスクの概説

IBM Software Support Handbook には、お使いの IBM 製品のサービスとサポートについての詳細情報が記載されています。ハンドブック (http://techsupport.services.ibm.com/guides/handbook.html) を参照してください。

IBM ソフトウェア・サポートに問い合わせるには、以下の手順に従います。

このタスクの実行方法

1. 問題の説明と背景情報の収集。 サポート・スペシャリストに問題を説明するときは、できるだけ具体的にお願いします。スペシャリストがお客様の問題解決を効率的にお手伝いできるように、関連する背景情報をすべてお知らせください。時間を節約するため、以下の質問に対する回答を用意しておいてください。
   - 問題が発生したときに実行していたソフトウェアのバージョンは? オペレーティング・システムと関連製品のバージョンを説明してください。
   - 問題は以前に発生しましたか? それとも、単独の問題ですか?
   - どのような手順が障害につながりますか?
   - 問題は繰り返し発生しますか? その場合、どのような手順が障害につながりますか?
   - ハードウェア、オペレーティング・システム、ネットワーキング・ソフトウェアなどといったシステムに変更を加えましたか?
   - この問題に対する解決方法を現在実行していますか。その場合は、問題を報告するときに説明を行う準備をしておいてください。

- 問題の症状に関連するログ、トレース、メッセージはありますか? IBM ソフトウェア・サポートからこれらの情報をお願いすることがあります。

2. 問題のビジネス・インパクトの判別。 問題を報告する際には、重大度レベルについてお尋ねします。そのため、報告する問題のビジネス・インパクトを理解し、評価する必要があります。次の表に記載された基準を使用してください。

表 24. 問題報告のための重大度基準

重大度	説明
1	**重大なビジネス・インパクト**: プログラムを使用できないため、業務に重大な影響を及ぼす。この状態は、即時のソリューションを必要とする。
2	**大きなビジネス・インパクト**: プログラムは使用可能だが、機能が著しく限定されている。
3	**ある程度のビジネス・インパクト**: プログラムは使用可能だが、比較的重要性の低い (業務上重大ではない) 機能が使用不能となっている。
4	**最小のビジネス・インパクト**: 問題はほとんど業務に影響を及ぼさない、またはその問題に対する合理的な回避策が講じられている。

3. 問題を IBM ソフトウェア・サポートに提出します。 問題を提出するには次の方法があります。
   - **オンライン**。IBM Software Support サイトで「Submit and track problems」ページ (http://www.ibm.com/software/support/probsub.html) に移動して、適切な問題提出ツールに情報を入力します。
   - **電話**。お住まいの国で使用できる電話番号を確認するには、Web の IBM Software Support Handbook のお問い合わせページ (http://techsupport.services.ibm.com/guides/contacts.html) にアクセスし、地域名をクリックします。

結果

お客様の提出される問題が、報告されていないソフトウェアの問題点、または資料の不備や不正確さに起因する場合、IBM ソフトウェア・サポートはプログラム診断依頼書 (APAR) を作成します。APAR では問題を詳細に記述し、その解決を追跡します。

次の作業

IBM ソフトウェア・サポートでは、APAR が解決されフィックスが配布されるまでの間、インプリメントできる予備手段を可能であれば常に提供します。 IBM では、解決済みの APAR を製品サポートの Web ページで毎日公開しています。これにより、同じ問題を抱える別のユーザーも問題を解決することが可能となります。

いくつかの一般的な問題の解決策

データベースを使用して Adapter for JDBC を実行中に発生する可能性のある問題の一部を、その解決策および予備手段と共に説明します。これらの問題および解決策は、ソフトウェア・サポート Web サイトの技術情報としても文書化されています。

ビジネス・オブジェクトで NCHAR 型の Oracle 列の属性が作成されない

問題

Oracle データベース・オブジェクトからビジネス・オブジェクトが生成されたとき、ビジネス・オブジェクトで NCHAR 型の列の属性が作成されません。

原因

JDBC ドライバーが、列の型を `Other` として戻しました。これはエンタープライズ・サービス・ディスカバリーではサポートされていません。

解決策

Business Object Designer ツールを使用して、サポートされる型の属性を手動でビジネス・オブジェクトに追加します。

JDBC エンタープライズ・サービス・ディスカバリーの始動時にクラス・ローダー違反が発生

問題

データ・パースペクティブでデータベースへの接続を使用した後に、JDBC エンタープライズ・サービス・ディスカバリーを使用することができません。エンタープライズ・サービス・ディスカバリー・ウィザードの 2 番目のパネルの最後に、例外 com.ibm.adapter.framework.api.ImportException が生成されます。
理由: pc:0 でクラス・ロードの制約に違反がありました
(クラス: oracle/jdbc/driver/OracleConnection
メソッド: getWrapper()Loracle/jdbc/OracleConnection;)。

このエラーは、以下の状態のどちらでも発生します。

- エンタープライズ・サービス・ディスカバリーを通じてデータベースへの接続を確立すると、データ・パースペクティブからデータベースに接続しようとしたときにエラーが発生します。
- データ・パースペクティブを通じてデータベースへの接続を確立すると、エンタープライズ・サービス・ディスカバリーからデータベースに接続しようとしたときにエラーが発生します。

原因

データ・パースペクティブおよびエンタープライズ・サービス・ディスカバリーが独自のクラス・ローダーを使用しているために、エラーが発生します。DLL (JDBC ドライバーが使用するネイティブ・ライブラリー) が、いったんデータ・パースペクティブにロードされると、エンタープライズ・サービス・ディスカバリーに再度ロードすることができません。JVM には、どの時点でも 1 つのクラス・ローダーのみがネイティブ・ライブラリーをロード可能であるという、固有の制約事項があります。そのため、クラス・ローダー A が DLL B をロードした場合、クラス・ローダー A が解放され、ガーベッジ・コレクションが実行されるまで、その他のクラス・ローダーは DLL B をロードできません。実際にはユーザーはガーベッジ・コレクションを制御できないため、通常、別のクラス・ローダーが DLL B をロー

ドするようにしたい場合は、JVM を再始動する必要があるということになります。この制限は、既知のものであり、WebSphere® Application Server では文書化されています。

解決策

このエラーが発生したときは、WebSphere Integration Developer を再始動することが唯一の解決策です。

Oracle 10g と共に XA を使用すると接続のクローズのエラーが発生する

問題

Oracle 10g を使用して XA トランザクションを実行するために Adapter for JDBC が使用されると、Adapter は Closed Connection (接続のクローズ) 例外 javax.resource.ResourceException: Closed Connection を生成します。

原因

これは、Oracle 10g データベース・ドライバーの既知の問題です。この問題については、Oracle で「3488761 Connection closed error from OracleConnection.getConnection() - 10G drivers」というバグが記録されています。

解決策

このバグは、Oracle 10g リリース 2 ドライバーで修正されています。予備手段としては、Oracle 9i JDBC Thin ドライバーを使用して、XA トランザクションのためにデータベースに接続します。

Oracle でトランザクションを開始中にエラーが発生する

問題

Oracle データベースを使用して XA トランザクションを実行するために Adapter for JDBC が使用されると、次のエラーが生成されます。WTRN0078E: トランザクション・マネージャーがトランザクションのリソースで start を呼び出そうとして、エラーが発生しました。エラー・コードは XAER_RMERR でした。 (WTRN0078E: An attempt by the transaction manager to call start on a transactional resource has resulted in an error. The error code was XAER_RMERR.)

原因

Oracle データベース・サーバーで XA トランザクションをサポートするには、いくつかのコマンドを実行する必要があります。

解決策

Oracle ディレクトリーに含まれる 2 つのスクリプトを実行する必要があります。これらのスクリプトを実行するために必要な許可を持つためには、SYSOPER または SYSDBA として Oracle にログインしていなければならないため、おそらくこのアクティビティーは、Oracle データベース管理者が実行する必要があります。スクリプトは次のとおりです。

```
<ORACLE_HOME>javavm¥install
file: initxa.sql
file: initjvm.sql
```

initxa.sql スクリプトは、データベースを XA 用に構成します。正常に実行されると、データベースは XA 向けに構成されます。このスクリプトは、一度目の試行で正常に実行される場合もあります。ただし、データベースのメモリー・スペースの一部が小さすぎるために、正常に実行されない可能性が高くなっています。

これを修正するには、initjvm.sql スクリプトを実行します。このスクリプトもおそらく失敗しますが、失敗したときに、どのパラメーターを調整する必要があるかが示されます。パラメーターは、次のファイルに格納されます。

```
<ORACLE_HOME>¥database
file: init<DATABASE_SID>.ora
```

以下の表に、通常増加させる必要のある 2 つのパラメーターを示します。ご使用の特定のデータベース構成では、異なるパラメーターの調整が必要となる可能性があります。

表 25.

パラメーター名	最小値
java_pool_size	12000000
shared_pool_size	24000000

アダプターを使用して JDBC (タイプ 2 またはタイプ 4) ユニバーサル・ドライバーにより IBM z/OS 上の DB2 に接続する

問題と原因

z/OS 上の DB2 では、デフォルトで位置インデックスを使用し、列名を使用しないストアード・プロシージャー・メタデータの照会をサポートします。Adapter for JDBC は列名を使用します。解決策では、z/OS プラットフォームで DB2 と共に WebSphere Adapter for JDBC を使用する手順を示しています。

解決策

Adapter for JDBC を使用するか、または JDBC API で DB2 Connect™ を使用して、z/OS® 上の DB2® に接続するには、以下の接続要件が満たされていることを確認してください。

- ユニバーサル JDBC ドライバーの物理表現は、db2jcc.jar ファイルです。このファイルへのパスが、クラス・パスに設定されていることを確認してください。
- データベース URL: タイプ 2 またはタイプ 4 のどちらのドライバーを使用するかを決定するには、接続の形式を検討します。

  タイプ 2: jdbc:db2:database
  (例: jdbc:db2:MyDB。MyDB はデータベース名。)

  タイプ 4: jdbc:db2://server:port/database
  (例: jdbc:db2://9.182.15.129:50000/MyDB。MyDB はデータベース名。)
- ドライバー・クラス: com.ibm.db2.jcc.DB2Driver。

タイプ 2 ドライバーとタイプ 4 ドライバーの両方で、ドライバー・クラスは同一です。

- クラス・パスに db2jcc_license_cisuz.jar ファイルのパスを設定します。

  タイプ 2 ドライバーとタイプ 4 ドライバーの両方で、ライセンス JAR ファイルは同一です。DB2 for z/OS サーバーおよび DB2 for i5/OS® サーバーにアクセスするには、有効な DB2 Connect™ のライセンスが必要です。DB2 クライアントは、DB2 Connect ライセンスがなければ、zSeries サーバーおよび iSeries サーバーへの接続を提供しません。

  DB2 Connect のライセンス交付および使用法については、以下のページを参照してください。
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0303zikopoulos1/0303zikopoulos1.html
  http://www-128.ibm.com/developerworks/db2/library/techarticle/0301zikopoulos/0301zikopoulos.html

エンタープライズ・サービス・ディスカバリーを使用してストアード・プロシージャーのメタデータをインポートする際に問題が発生する可能性があります。Adapter for JDBC を使用して、ストアード・プロシージャーを使用し、DB2 からメタデータをインポートするには、DB2 を以下のステップで説明するように再構成する必要があります。前述のステップに加えて、以下のステップを実行してください。

- DB2 に APAR の PQ62695、PQ55393、PQ56616、PQ54605、PQ46183、および PQ62139 を適用します。
- アダプターでストアード・プロシージャーを使用したい場合は、下記のステップを実行します。これは、PQ62695 のフィックスの一部です。このフィックスは、JDBC および ODBC 仕様で文書化されているスキーマ・メタデータ API に対応する結果セットを生成することが可能なストアード・プロシージャーを導入しています。

  これらのプロシージャーは、DB2 Universal Driver で提供される JDBC および ODBC ドライバーが使用します。以下のステップを実行して、ストアード・プロシージャーのサポートを使用可能にします。

  1. APAR を適用します。
  2. ZPARM アセンブリー・ジョブ DSNTIJUZ 内の DESCSTAT 変数の値を確認します。 DESCSTAT 変数の値が NO である場合は、YES に変更します。

     注: DESCSTAT のデフォルトは、V7 では NO ですが、V8 では YES に変更されました。
  3. ZPARM モジュールを再アセンブルし、再初期化します。
  4. DSNTIJMS という名前の JCL ジョブを実行します。このメンバーは、db2prefix.SDSNSAMP データ・セット内にあります。
  5. DB2 を再始動します。

ラッパー・ビジネス・オブジェクトの代わりにトランザクションを使用

これは、英語バージョンのみに適用されます。

問題

WebSphere Business Integration Adapter for JDBC は、トランザクションのサポートを統合ブローカー (WebSphere InterChange Server) に公開していませんでした。そのため、ユーザーが 2 つの無関係なオブジェクト (例えば、カスタマーおよびオーダー) を同時にエンタープライズ情報システム (EIS) で作成されるようにしたい場合に、問題が発生していました。

これを実行するには、2 つの別々の要求を送信する必要があり、要求と要求の間にアダプターが停止すると、一方のオブジェクトのみが作成されました。 WebSphere InterChange Server アダプター・インターフェースの制限のため、ラッパー・オブジェクトを通じて単一のバッチで複数の要求をアダプターへ送信できるようにすることが解決策でした。このラッパー・オブジェクトは、1 から n 個の子ビジネス・オブジェクトを保持するダミー・コンテナーであり、子ビジネス・オブジェクトのそれぞれが正常に処理されてからでなければ、バックエンド・データベースへのコミットが発行されませんでした。

解決策

新しい WebSphere Adapter for JDBC (JCA) は、ローカル・トランザクションとグローバル・トランザクションをサポートします。例えばカスタマーおよびオーダーを同時に作成する要求を送信したい場合、アダプター・クライアント (メディエーションまたはビジネス・プロセスなど) からトランザクションを開始し、適切と考える要求を送信することができるようになりました。送信したいものをすべて送信したら、トランザクションをコミットします。ラッパー・ビジネス・オブジェクトは必要ありません。

リモート DB2 データベースを用いたアウトバウンド・サポートのための XA トランザクションの使用

これにより、リモート DB2 データベースと共に Adapter for JDBC を使用する XA トランザクション・サポートのための手順、データベース・バージョン、および構成要件が提供されます。

リモート DB2 データベースでの XA トランザクションの使用

リモート DB2 データベースの追加

1. DB2 サーバー・マシンで db2admin (<DB2_Installpath>¥SQLLIB¥BIN) コマンドを実行します。
2. DB2 構成アシスタントを開きます。
3. 「**表示 (View)**」 → 「**拡張表示 (Advanced View)**」に移動します。

リモート・システムの追加、インスタンス・ノードの追加、データベースの追加、データベース接続のテストという 4 つのタスクを、この順序で実行します。

リモート・システムの追加

1. 「**システム**」タブを選択します。
2. メニューから、「**選択済み**」 → 「**システムの追加 (Add System)**」を選択します。
3. 「**システム名 (System name)**」フィールドで、ターゲット・データベースが配置されている物理マシン、サーバーシステム、またはワークステーションを指定します。サーバー・システム上のシステム名は、DB2SYSTEM DAS 構成パラメーターによって定義されます。ユーザーはこの値を使用する必要があります。
4. 「**ホスト名**」フィールドに、ホスト名か、またはターゲット・データベースが存在するインターネット・プロトコル (IP) アドレスを入力します。
5. 「**ノード名 (Node name)**」フィールドで、データベースが配置されているリモート・ノードのローカル・ニックネームを指定します。選択するノード名は、ノード・ディレクトリーまたは管理ノード・ディレクトリーに存在していないものにする必要があります。
6. オペレーティング・システムを選択して、「**OK**」をクリックします。

インスタンス・ノードの追加

1. 「**インスタンス・ノード (Instance Nodes)**」タブを選択します。
2. メニューから、「**選択済み**」 → 「**インスタンス・ノードの追加 (Add Instance Node)**」を選択します。
3. 「**システム名 (System name)**」フィールドで、ターゲット・データベースが配置されている物理マシン、サーバーシステム、またはワークステーションを指定します。リモート・システムの追加タスクで追加したシステムを選択します。
4. 「**インスタンス名**」フィールドに、ターゲット・データベースが配置されているインスタンスの名前 (DB2 など) を入力します。
5. 「**インスタンス・ノード名 (Instance node name)**」フィールドで、データベースが配置されているカタログされたシステム (ノード) の固有のニックネームを指定します。選択するノード名は、ノード・ディレクトリーまたは管理ノード・ディレクトリーに存在していないものにする必要があります。
6. オペレーティング・システムを選択して、ホスト名を入力します。リモート・システムの追加タスクのステップ 4 と同じホスト名を使用します。
7. リモート DB2 インスタンスが実行されているポート番号を入力します。
8. 「**OK**」をクリックします。

データベースの追加

1. 「**データベース**」タブを選択します。
2. メニューから、「**選択済み**」 → 「**データベースの追加**」を選択します。
3. 「**インスタンス・ノード (Instance node)**」フィールドで、インスタンス・ノードの追加タスクで作成したインスタンスを選択します。「**データベース名 (Database name)**」フィールドで追加するデータベースの名前を指定します。
4. 「**別名 (Alias)**」フィールドで、ワークステーション上で実行中のアプリケーションが使用できるローカル・ニックネームを指定します。何も入力されない場合、別名はデータベース名と同じになります。別名は、固有でなければなりません。

注: この別名値を、アダプターの XADatabaseName プロパティーに入力する必要があります。

データベース接続のテスト

1. 「**データベース**」タブを選択します。
2. データベースの追加タスクで追加されたデータベースを選択します。
3. メニューから、「**選択済み**」 → 「**接続のテスト**」を選択します。
4. 「**CLI**」チェック・ボックスを選択し、ユーザー ID およびパスワードを入力し、「**接続のテスト**」をクリックします。これにより、接続の成功が戻されます。

Universal Driver を使用した Adapter for JDBC での XA トランザクションの使用

Adapter for JDBC (JCA) およびリモート DB2 データベースと共に XA トランザクションを使用するには、以下のバージョンのソフトウェアおよび構成プロパティーが必要です。

- DB2 バージョン: 8.2
- JDBC ドライバー: UDB ドライバー (db2jcc.jar) タイプ 4
- XA データ・ソース名: com.ibm.db2.jcc.DB2XADataSource
- XA データベース名: これは、ローカル DB2 クライアントで構成されたリモート・データベース別名です。
- データベース URL: jdbc:db2://hostname:port/databasename
- JDBC ドライバー・クラス: com.ibm.db2.jcc.DB2Driver

WebSphere Adapters で Inbound 操作用と Outbound 操作用に別個のログが用意されない

問題

インポートおよびエクスポートを作成して、それらを同一のアダプター・インスタンスにバインドしたとき (同じ AdapterID プロパティーを使用)、例えばインポートのログには「a.log」、エクスポートのログには「b.log」などのように、2 つのバインディングのログに異なる名前を付けると、プロジェクトを WebSphere Process Server にデプロイした後には、リソース・アダプターには「b.log」という名前の 1 つのログしかないことが分かります。

原因

WebSphere® Adapters は、Inbound 操作および Outbound 操作からのメッセージングを区別しないため、アダプター・ログ・ファイルを 1 つしか作成しません。

解決策

インポートおよびエクスポートが同一のアダプター・インスタンスにバインドしている場合、Inbound 操作および Outbound 操作を同一のログ名を用いて構成します。インバウンドおよびアウトバウンドが別々のアダプターに属し、それぞれのアダプター ID が異なる場合は、この問題は発生しません。

第 11 章 クイック・スタート・チュートリアル

アダプターのセットアップおよびデプロイについての実践的知識を得るために、チュートリアルを実行してください。チュートリアルで完了させる必要がある事項はすべて、そのチュートリアル内に記載されています。前提条件タスク (アダプターのインストールなど) を既に完了している場合、チュートリアルは 1 時間以内に完了させることができます。

概要

チュートリアルでは、J2EE コンポーネントがアダプターを使用してデータベースに要求を送信できるようにアダプターを構成するための、すべての手順の解説を提供します。チュートリアルは、親子関係を持つレコードをデータベース表で作成する例を示します。このシナリオでは、ビジネス・オブジェクトに添付されたストアード・プロシージャーの使用方法も示します。

学習目標

チュートリアルを完了すると、以下の操作を実行できるようになります。

- WebSphere Integration Developer でのアダプター・プロジェクトの作成
- データベースからサービスおよび関連ビジネス・オブジェクトをディスカバーし、それらをアダプター・プロジェクトの一部にする操作
- WebSphere Process Server テスト環境にインストールするデプロイ可能なモジュールの作成
- モジュールが正しく作動することを確認し、モジュール実行の結果を確認するためのモジュールのテスト

所要時間

このチュートリアルが完了するまでに 1 時間はかかりません。

対象読者

チュートリアルは、WebSphere Process Server または WebSphere Enterprise Service Bus にデプロイするように Adapter for JDBC を構成する、統合開発者に向けたものです。

前提条件

チュートリアルを始める前に、以下の作業が完了していることを確認してください。

- 前提ソフトウェアをすべてインストールする
- Adapter for JDBC をインストールする
- データベースにアクセスするためのすべての情報 (ユーザー ID およびパスワードなど) を入手していることを確認する
- 以下の手順を使用して、テーブルおよびストアード・プロシージャーを作成する

© Copyright IBM Corp. 2005, 2006

テーブルおよびストアード・プロシージャーの作成

チュートリアルを実行するには、以下のテーブルおよびストアード・プロシージャーをデータベース内で作成する必要があります。

- Customer テーブルおよび Address テーブルを作成するためのスクリプト:

```
CREATE TABLE CUSTOMER (
          "PKEY" VARCHAR(10) NOT NULL PRIMARY KEY,
          "FNAME" VARCHAR(20),
          "LNAME VARCHAR(20),
          "CCODE" VARCHAR(10);

CREATE TABLE ADDRESS (
          "ADDRID" VARCHAR(10) NOT NULL PRIMARY KEY,
          "CUSTID" VARCHAR(10),
          "CITY" VARCHAR(20),
          "ZIPCODE" VARCHAR(10);
```

- ストアード・プロシージャーを作成します。

  IBM DB2 Development Center を使用して、ストアード・プロシージャーを作成することができます。Development Center で、ウィザードを使用して新規 Java ストアード・プロシージャーを作成します。Address テーブルで挿入を実行するストアード・プロシージャーを作成します。

チュートリアル: データベース表でのレコードの作成

このシナリオでは、親ビジネス・オブジェクトおよび子ビジネス・オブジェクトを使用してレコードを作成し、ストアード・プロシージャーを子ビジネス・オブジェクトに関連付けます。最初に、アダプター・プロジェクトを作成してから、エンタープライズ・サービス・ディスカバリー・ウィザードを使用して、Customer ビジネス・オブジェクトおよび Address ビジネス・オブジェクトを生成します。アダプターと新規に生成されたビジネス・オブジェクトを含むモジュールを作成し、そのモジュールをテスト環境にデプロイします。

認証別名の作成

サーバーの管理コンソールを使用して、認証別名をサーバー上に作成します。管理コンソールから、Outbound 要求の処理のためにデータベースにアクセスできるようにするユーザー ID およびパスワードを使用して、認証別名を構成します。

このタスクの概説

アダプターを構成する前に、認証別名をサーバー上に作成する必要があります。認証別名を作成するには、以下の手順を使用してください。

このタスクの実行方法

1. **「スタート」** → **「プログラム」** → **「IBM WebSphere」** → **「Integration Developer V6.0.2」** → **「WebSphere Integration Developer V6.0.2」**をクリックして、WebSphere Integration Developer を開始します。
2. ワークスペースを指定するようにプロンプトが表示された場合は、デフォルト値を受け入れます。 ワークスペースとは、WebSphere Integration Developer がプロジェクトを保管するディレクトリーのことです。

3. 「WebSphere Integration Developer」ウィンドウが表示されたら、「ようこそ」ページを閉じます。
4. **「ウィンドウ」 → 「パースペクティブを開く」**をクリックして、「ビジネス・インテグレーション」パースペクティブに移動します。次に、**「ビジネス・インテグレーション (デフォルト) (Business Integration (default))」**をクリックし、**「OK」**をクリックします。
5. 管理コンソールを表示します。
   a. **「サーバー」**タブをクリックします。
   b. **「WebSphere Process Server v6.0.2」**で**「始動済み」**という状況が表示されない場合は、**「WebSphere Process Server v6.0.2」**を右クリックして、**「開始」**をクリックします。
   c. **「WebSphere Process Server v6.0.2」**を右クリックし、**「管理コンソールの実行」**をクリックします。

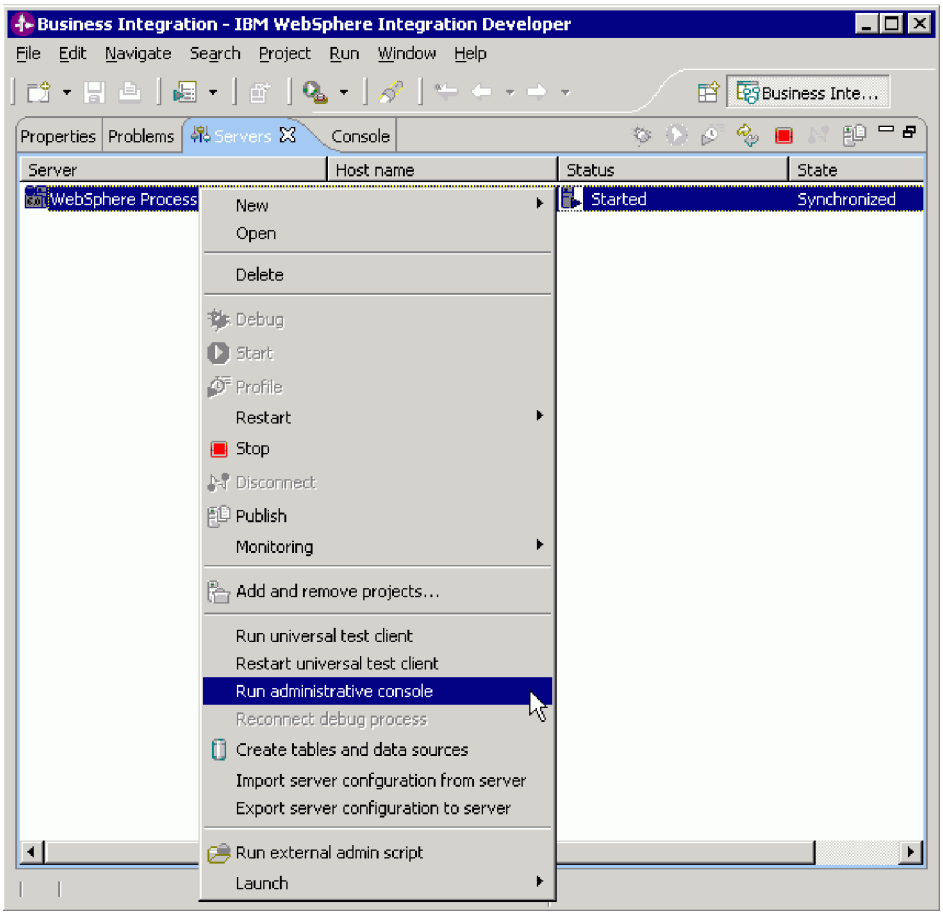


図 24. サーバー上の管理コンソールの実行

   d. **admin** と入力して、**「ログイン」**をクリックすることにより、管理コンソールにログインします。
6. 管理コンソールで、**「セキュリティー」 → >「グローバル・セキュリティー」**をクリックします。

図 25. グローバル・セキュリティーの選択

7. 「認証」見出しの下で、「**JAAS 構成**」 → 「**J2C 認証データ**」をクリックします。
8. 認証別名「**SCA_Auth_Alias**」を選択します。

図 26. 認証別名の選択

- 「**SCA_Auth_Alias**」 項目がまだ存在しない場合は、「**新規**」をクリックして、「一般プロパティー」ウィンドウの「**別名**」フィールドに入力します。
- それ以外の場合は、「**SCA_Auth_Alias**」をクリックします。

認証別名は大文字小文字が区別されます。

9. データベースへの接続に必要なユーザー ID およびパスワードを入力します。

ご使用のデータベースが要求するフォーマット (すべて大文字など) で、パスワードを入力します。

図 27. 認証データの入力

10. 「**OK**」をクリックします。

    「別名」リストに名前が表示されたら、メモを取っておきます。この名前を以降の構成ウィンドウで使用します。

11. 「**保管**」をクリックした後、再度「**保管**」をクリックします。 サーバーを再始動すると、変更が有効になります。

結果

「成果物の生成」でアダプター・プロパティーを構成する際に使用する認証別名が作成されました。

WebSphere Integration Developer でのアダプター・プロジェクトの作成

データベースと通信するためのモジュールの作成のプロセスを開始するには、まずアダプター・プロジェクトを作成します。このアダプター・プロジェクトには、アダプター自体と他の関連する成果物が含まれています。リソース・アダプター・アーカイブ (RAR) ファイル (インストール時にローカル・ファイル・システムにコピーされます) を WebSphere Integration Developer 内の「コネクター・プロジェクト」フォルダーにインポートすることによって、プロジェクトを作成します。

1. WebSphere Integration Developer で、「J2EE」パースペクティブに移動します。
   a. 「**ウィンドウ**」 → 「**パースペクティブを開く**」 → 「**その他**」をクリックします。
   b. 「**J2EE**」をクリックします。

「**J2EE**」が表示されない場合は、「**すべて表示**」チェック・ボックスを選択して「**J2EE**」をクリックし、「**OK**」をクリックします。

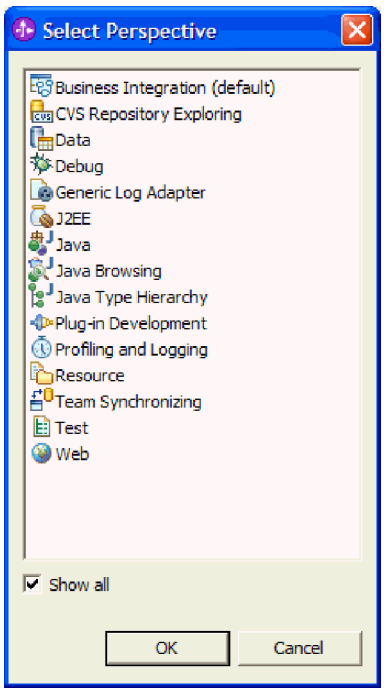


図 28. 「パースペクティブの選択」リストからの「J2EE」の選択

c. 「使用可能化の確認」ウィンドウが表示された場合は、「**常に機能を使用可能にし、今後このメッセージを表示しない**」を選択してください。

d. 「**OK**」をクリックします。

2. 「**コネクター・プロジェクト**」を右クリックして RAR ファイルをインポートし、「**インポート**」 → 「**RAR ファイル**」をクリックします。

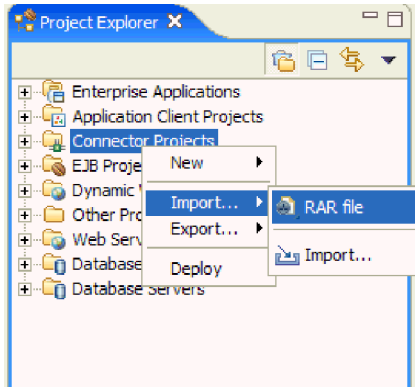


図 29. RAR ファイルのインポート

3. 「**参照**」をクリックして、Adapter for JDBC がインストールされたディレクトリーへナビゲートすることによって、ローカル・ファイル・システム上で RAR ファイルを検出します。

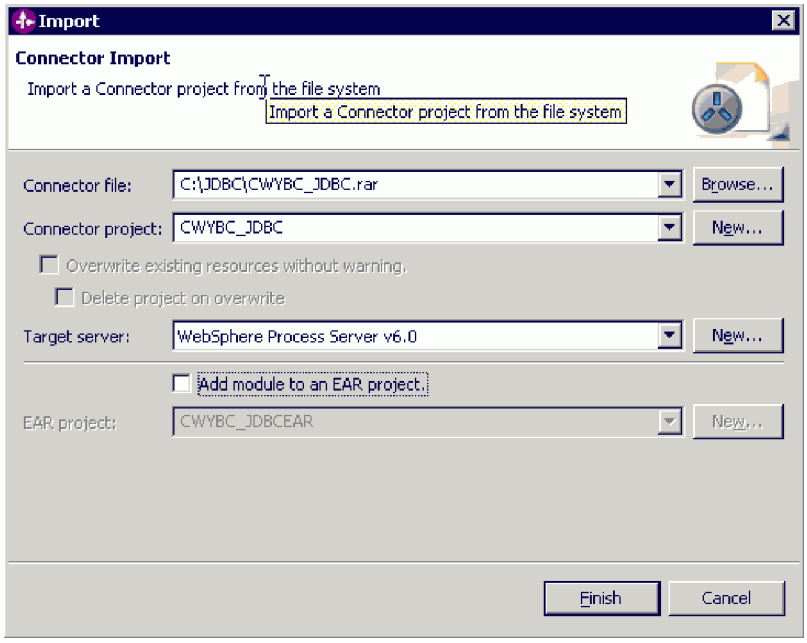


図 30. インストール・ディレクトリーからの RAR ファイルの選択

4. 「**コネクター・プロジェクト**」のデフォルト設定 (**CWYBC_JDBC**) を受け入れます。

   アダプター・プロジェクトは、RAR ファイルと同じ名前を持ちます。

   CWYBC_JDBC.rar という名前のプロジェクトが既にワークスペース内に存在する場合、「**コネクター・プロジェクト**」フィールド内の名前には番号が付加されています (例えば CWYBC_JDBC1)。
5. 「**ターゲット・サーバー**」フィールドのデフォルト値を受け入れます。

   デフォルト値は、WebSphere Integration Developer の一部としてインストールされた WebSphere Process Server のテスト環境です。
6. 「**モジュールを EAR プロジェクトに追加**」チェック・ボックスをクリアします。

   チェックマークを外すと、「EAR プロジェクト」フィールドは無効になります。
7. 「**終了**」をクリックします。

結果

CWYBC_JDBC という名前の新規 J2EE アダプター・プロジェクトが作成されました。内容を確認するには、**CWYBC_JDBC** を展開します。

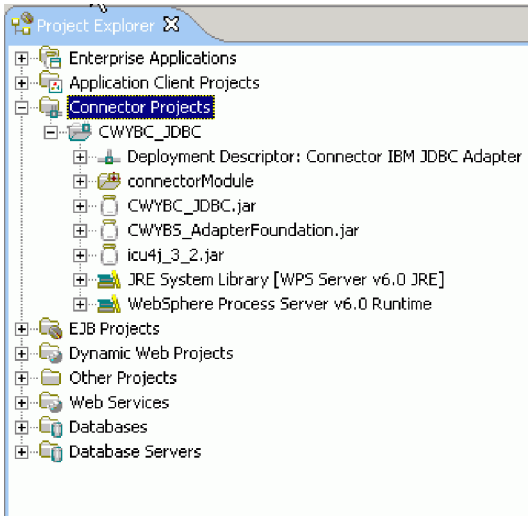


図 31. 「プロジェクト・エクスプローラー」ウィンドウ内の CWYBC_JDBC アダプター・プロジェクト

外部依存関係の追加

WebSphere Integration Developer でプロジェクトを作成したら、データベース・ドライバーをアダプター・プロジェクトに追加する必要があります。これを行うには、JDBC ドライバーを Java ビルド・パスに追加します。これには、ご使用のデータベースに提供されている外部 JAR ファイルを使用してください。

1. WebSphere Integration Developer の「J2EE」パースペクティブ内で、「**コネクター・プロジェクト**」を展開します。 アダプター・プロジェクト **CWYBC_JDBC** を右クリックして、「**プロパティー**」を選択します。

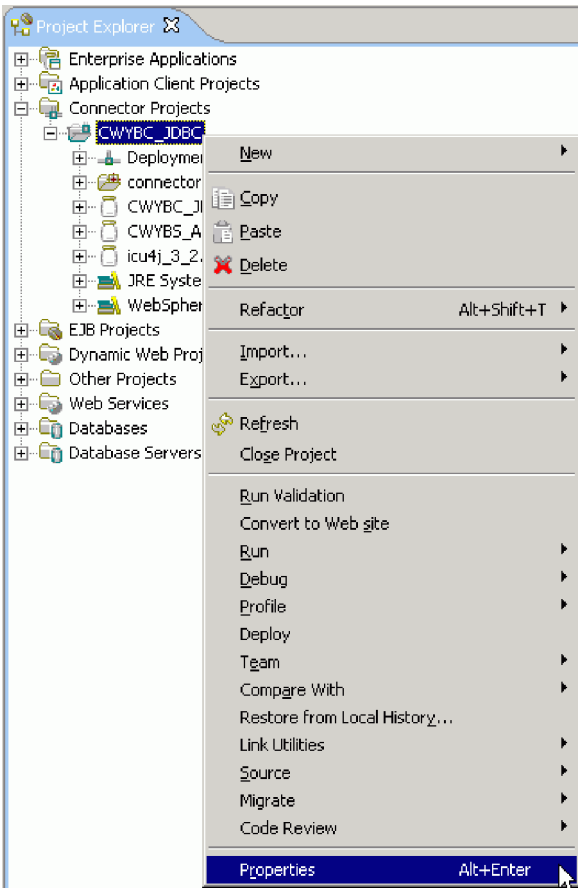


図 32. プロジェクトの「プロパティー」オプションの選択

2. ナビゲーション・サイドバーで、「**Java のビルド・パス**」をクリックします。「**ライブラリー**」タブを選択し、「**外部 JAR の追加**」をクリックします。

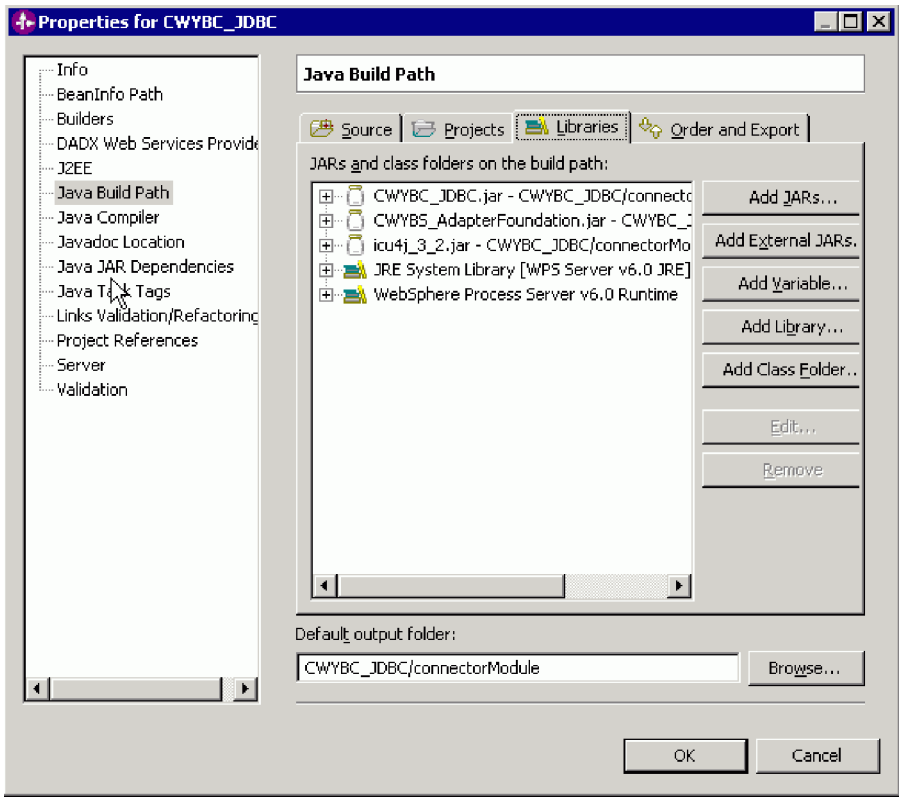


図 33. 外部 JAR ファイルの追加

3. 「JAR の選択」ウィンドウで、JDBC ドライバー JAR ファイルを含むローカル・ファイル・システム上のディレクトリーにナビゲートします。「**開く**」をクリックします。

   JDBC ドライバー JAR ファイルが Java ビルド・パスに追加されます。
4. 「**OK**」をクリックします。

結果

アダプター・プロジェクトには JDBC ドライバーへの参照が含まれています。これは、WebSphere Integration Developer の「プロジェクト・エクスプローラー」ウィンドウで確認できます。ドライバーが追加されて、アダプターがデータベースに接続できるようになり、テーブル、ストアード・プロシージャーなどのビジネス・オブジェクトを生成できるようになりました。

Outbound 処理を行うためのアダプターの構成

WebSphere Adapter for JDBC を Outbound 処理用に構成するには、WebSphere Integration Developer でエンタープライズ・サービス・ディスカバリー・ウィザードを使用します。構成する際には、エンタープライズ・サービス・ディスカバリーの接続プロパティーを設定しエンタープライズ情報システムでビジネス・オブジェクトを選択し、Outbound 処理用のビジネス・オブジェクト定義を生成します。

エンタープライズ・サービス・ディスカバリー接続プロパティーの設定

エンタープライズ・サービス・ディスカバリー・ウィザードを始動して、ご使用のデータベース・インスタンス用に接続プロパティーの値を設定する必要があります。エンタープライズ・サービス・ディスカバリー処理には、ディスカバリーのためのデータベース接続やサービス記述の作成にこれらのプロパティーが必要となります。

1. エンタープライズ・サービス・ディスカバリーを開始します。
   a. WebSphere Integration Developer で、**「ウィンドウ」** → **「パースペクティブを開く」** → **「その他」**を選択して、**「ビジネス・インテグレーション」**パースペクティブに移動します。**「ビジネス・インテグレーション (デフォルト) (Business Integration (default))」**をクリックして、**「OK」**をクリックします。
   b. **「ファイル」**をクリックして、**「新規」** → **「エンタープライズ・サービス・ディスカバリー」**を右クリックします。

      「エンタープライズ・サービス・ディスカバリー」が表示されない場合は、**「新規」** → **「その他」**をクリックし、**「ビジネス・インテグレーション」**を展開して、**「エンタープライズ・サービス・ディスカバリー」**をクリックします。次に、**「次へ」**をクリックします。
2. 「エンタープライズ・サービス・ディスカバリー」ウィンドウで、**「'CWYBC_JDBC' コネクター・プロジェクトからの JDBC EMD アダプター (バージョン 6.0.2) (JDBC EMD Adapter (version 6.0.2) from the 'CWYBC_JDBC' Connector Project)」**を選択して**「次へ」**をクリックします。
3. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウで、データベースへの接続に必要な**「接続構成」**プロパティーを設定します。

   「ユーザー名」、**「パスワード」**、**「データベース URL」**、および**「JDBC ドライバー・クラス」**を入力し、**「次へ」**をクリックします。データベース URL および JDBC ドライバー・クラスの適切な値については、ご使用のドライバーの資料を確認してください。

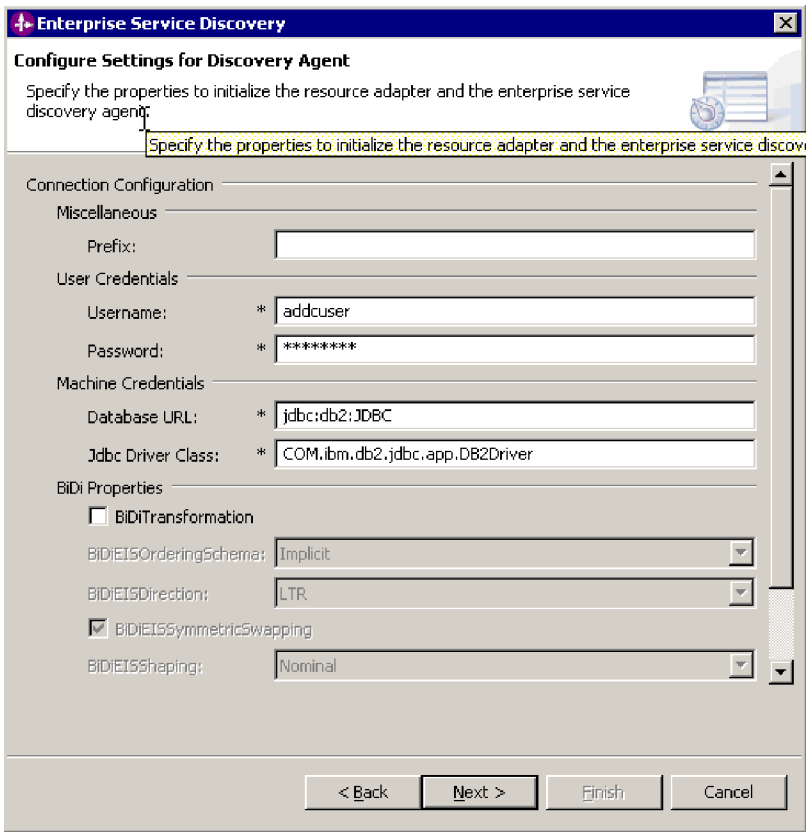


図 34. ディスカバリー・エージェント・ウィンドウの設定を構成する

4. 構成中に発生する可能性のあるエラーを確認できるように、ロギング・レベルを設定します。

   JDBCMetadataDiscovery.log というログ・ファイルのデフォルト・ロケーションは、ワークスペース内の .metadata フォルダーです。

   a. 「ディスカバリー・エージェントの設定の構成 (Configure Settings for Discovery Agent)」ウィンドウの下部で、「**拡張を表示**」をクリックします。ボタンが「**拡張を非表示**」に変わります。
   b. 「**ロギング・レベル**」で、「**FINEST**」 を選択します。
   c. 「**次へ**」をクリックします。

ビジネス・オブジェクトおよびサービスの選択

照会を実行して、Address データベース・オブジェクトおよび Customer データベース・オブジェクトを検出します。ストアード・プロシージャーを Address ビジネス・オブジェクトに関連付けます。最後に、これらのオブジェクトをインポート対象として選択します。

1. 「エンタープライズ・サービスの検索とディスカバリー (Find and Discovery Enterprise Services)」ウィンドウで、「**照会の編集(Edit Query)**」をクリックします。

   「照会の編集 (Edit Query)」を使用して、ビジネス・オブジェクトのアプリケーション固有情報を追加することができます。また、「照会の編集 (Edit Query)」を使用して、ツリー構造で表示されるスキーマ、ノード、またはオブ

ジェクトのリストを絞り込むことができます。このシナリオでは、フィルター操作を使用する必要はありません。

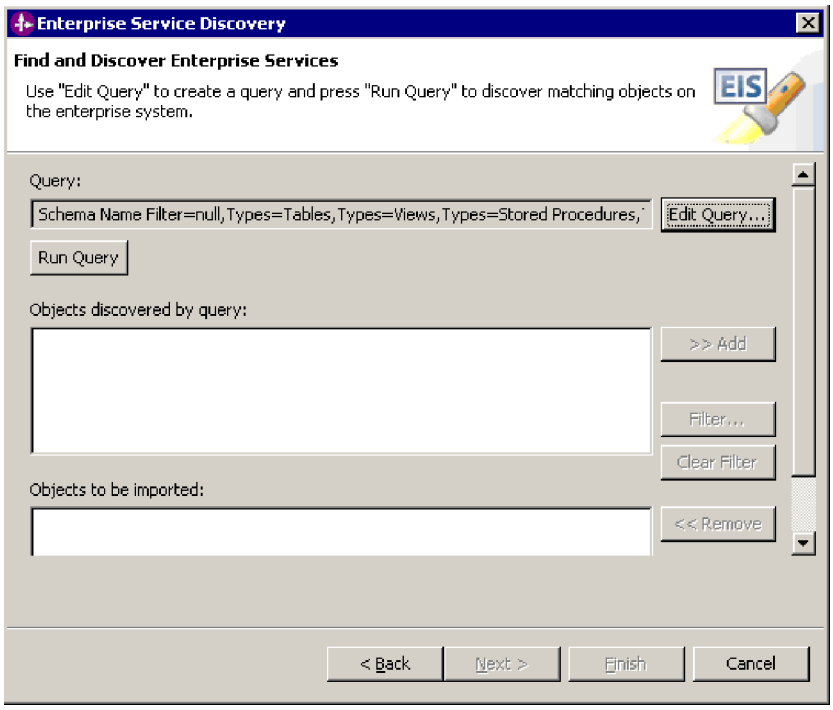


図 35. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウ

2. 「照会フィルター・プロパティー」ウィンドウで、「**ビジネス・オブジェクト ASI の追加**」チェック・ボックスにチェック・マークを付け、「**OK**」をクリックします。

   すると、メタデータ照会を実行したときにオブジェクトを追加するたびに、アプリケーション固有情報 (ASI) を入力するためのウィンドウが表示されます。

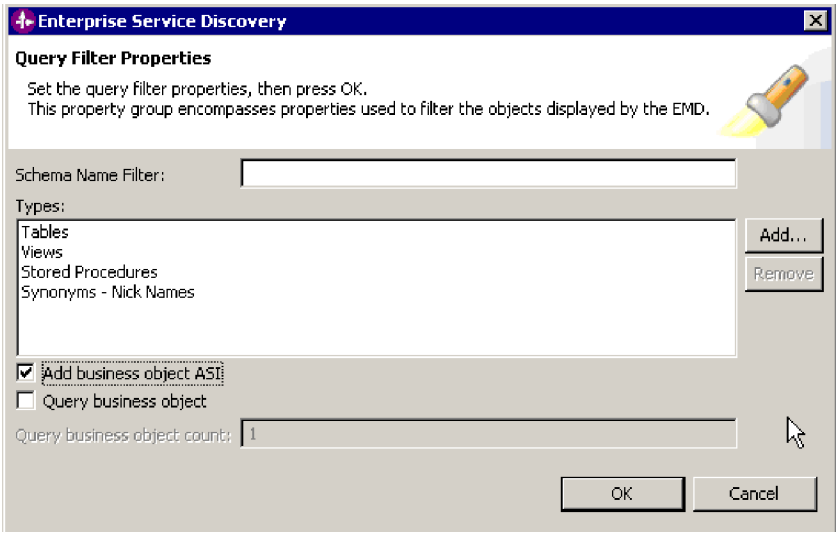


図 36. 「照会フィルター・プロパティー」ウィンドウ

3. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**照会の実行 (Run Query)**」をクリックして、メタデータ照会を実行します。

   照会によりディスカバーされたオブジェクトが、上部ペインに表示されます。テーブル、ビュー、ストアード・プロシージャー、同義語/ニックネームは、スキーマ名によってソートされます。

4. **ADDCUSER** スキーマ名を展開し、「**テーブル**」タイプを展開することによって、リスト内でオブジェクトを検索します。
5. **Address** テーブルおよび **Customer** テーブルをクリックしてから、「**>>追加**」をクリックして、これらのオブジェクトをインポート対象として選択します。

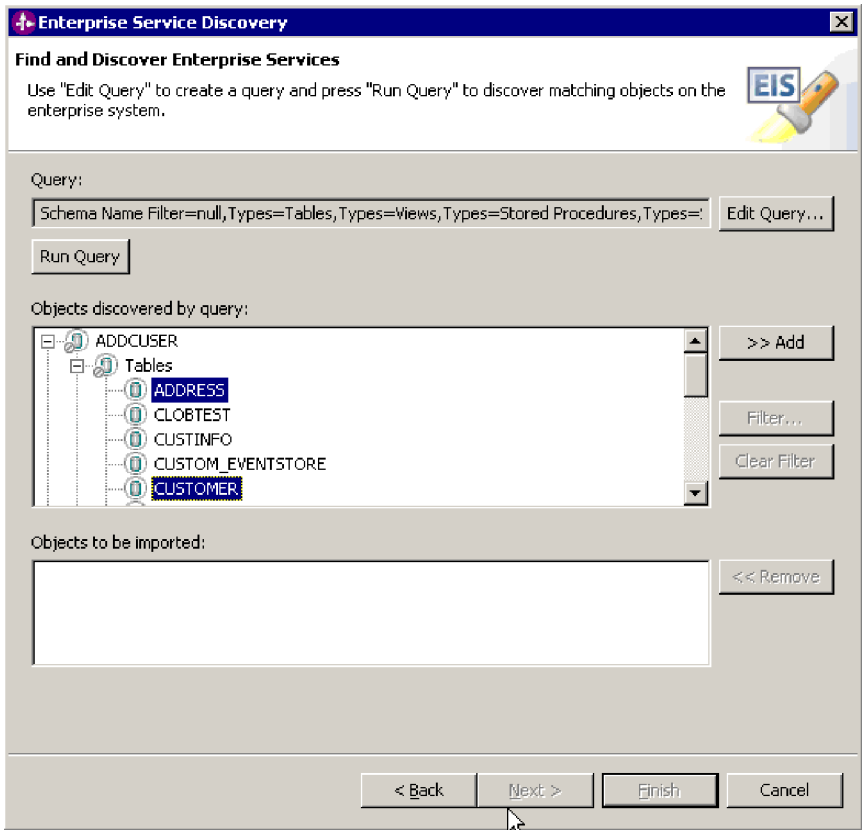


図 37. 照会でディスカバーされたオブジェクトの選択

ステップ 2 でビジネス・オブジェクト ASI を追加することを選択したため、次のウィンドウが表示されます。

6. 「ADDRESS の構成パラメーター (Configuration Parameters for ADDRESS)」ウィンドウで、「**追加**」をクリックします。

「**ステータス列名**」は、論理削除のみに使用され、「**ステータス値**」は、「ステータス列名」用に選択された属性に対応する列で更新する値を指定します。このシナリオでは論理削除を実行しないため、これらの 2 つのパラメーターはそのままにしておきます。

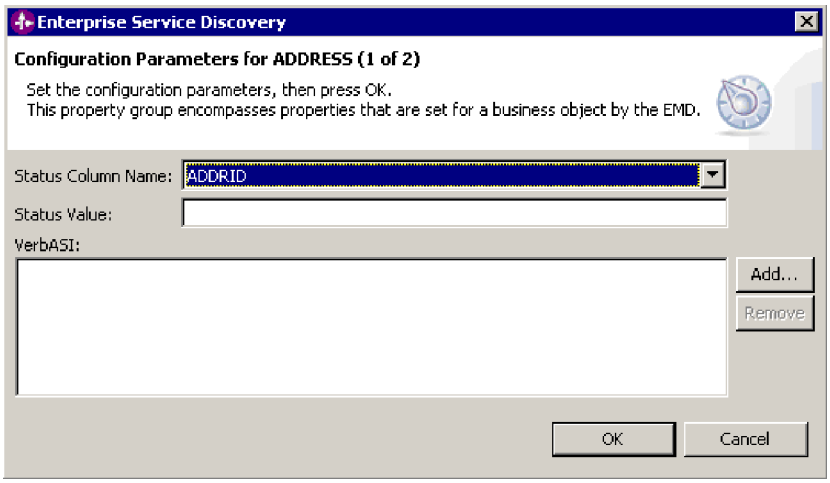


図 38. 「ADDRESS の構成パラメーター (Configuration Parameters for ADDRESS)」ウィンドウ

ストアード・プロシージャーのリストが表示されます。

7. 「**CreateSP**」を選択して、「**OK**」をクリックします。 これにより、CreateSP ストアード・プロシージャーのアプリケーション固有情報が ADDRESS テーブルに関連付けられます。ビジネス・オブジェクトで Create 操作が呼び出されると、CreateSP で指定されたストアード・プロシージャーが呼び出され、挿入操作を実行します。

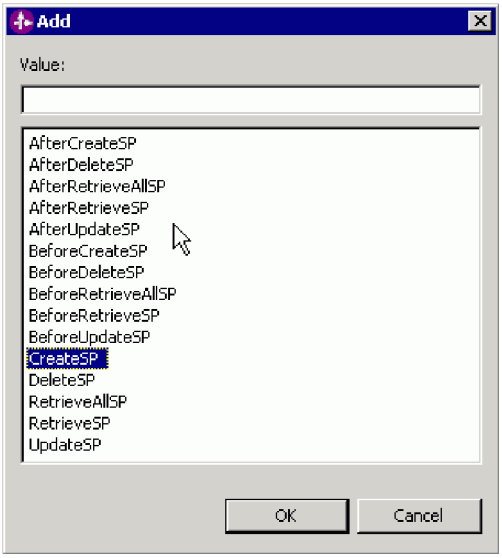


図 39. 動詞 ASI に追加するストアード・プロシージャーのリスト

8. 「ADDRESS の構成パラメーター (Configuration Parameters for ADDRESS)」ウィンドウの「**CreateSP**」の下で、「**スキーマ**」フィールドおよび「**ストアード・プロシージャー**」フィールドに値を入力します。

   次の値を使用してください。
   - スキーマ: ADDCUSER
   - ストアード・プロシージャー: CREATEADDRESS

9. 「**ストアード・プロシージャー**」名を入力すると、「**ストアード・プロシージャー・パラメーター**」フィールドが表示されます。 「**OK**」をクリックします。

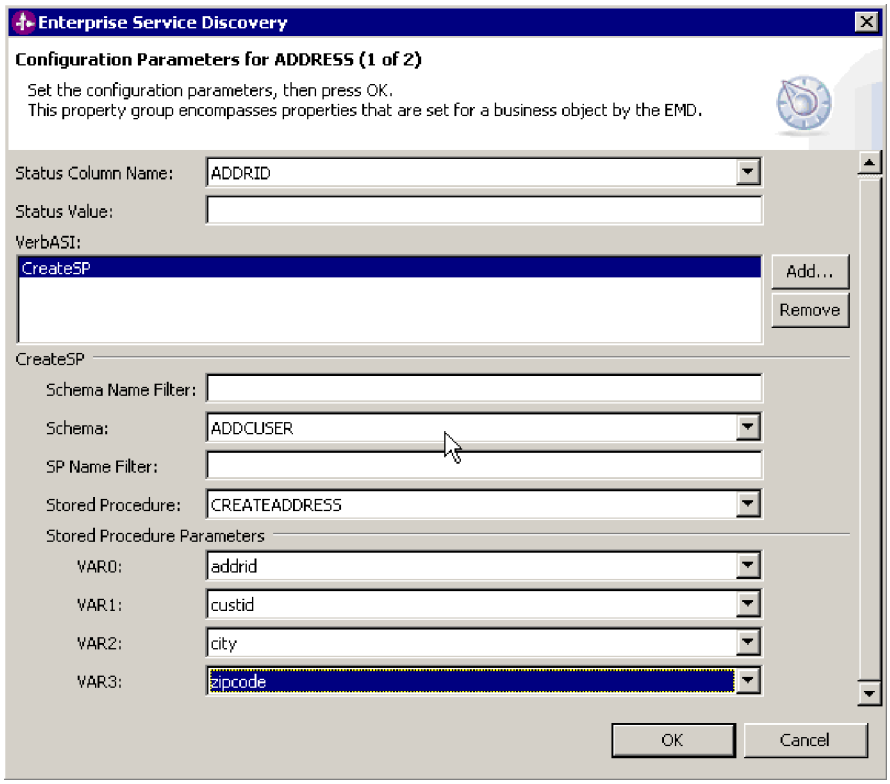


図 40. CreateSP の値

「CUSTOMER の構成パラメーター (Configuration Parameters for CUSTOMER)」ウィンドウが表示されます。

複数の追加ビジネス・オブジェクトを選択した場合は、最初のオブジェクトのビジネス・オブジェクト ASI を追加するウィンドウが表示されます。アプリケーション固有情報を入力すると、2 番目のビジネス・オブジェクトについて同じウィンドウが表示されます。

10. Customer ビジネス・オブジェクトにストアード・プロシージャーは関連付けられないため、動詞アプリケーション固有情報を追加する必要はありません。「**OK**」をクリックします。
11. 「エンタープライズ・サービスの検索とディスカバー (Find and Discover Enterprise Services)」ウィンドウで、「**インポートするオブジェクト (Objects to be imported)**」ペイン内に **ADDRESS** および **CUSTOMER** が表示されます。「**次へ**」をクリックします。

結果

選択されたオブジェクトが、アダプターで使用できるようにインポートされました。

選択済みオブジェクトの構成

データベース・オブジェクトを選択した後、インポート・ファイルのメタデータ選択プロパティーの値を指定する必要があります。

このタスクの実行方法

1. 「オブジェクトの構成 (Configure Objects)」ウィンドウで、「**サービス・タイプ**」として 「**Outbound**」 を選択します。

   「**操作**」フィールドに、アウトバウンド・サービス・タイプに対してアダプターがサポートしているオブジェクトがリストされます。
2. 「**ネーム・スペース**」フィールドおよび「**最大レコード数**」フィールドのデフォルト値を変更しないでください。「**BO ロケーション**」はブランクのままにします。

   ネーム・スペースは、ビジネス・オブジェクト・スキーマを論理的に分離するため、ビジネス・オブジェクト名の前に付加されます。このデフォルト値は、メタデータ・スキーマ JDBCASI.xsd のネーム・スペースです。

   「最大レコード数」は、RetrieveAll 操作で検索するレコードの最大数を表します。

   「BO ロケーション」は、生成された .xsd ファイルが保管されるロケーションへのパスを表します。
3. 「**次へ**」をクリックします。

成果物の生成

アダプターが使用するプロパティーを再構成して、特定のデータベースへの通信チャネルをセットアップします。また、すべての成果物およびプロパティー値を保管できる新規ビジネス・インテグレーション・モジュールを作成する必要があります。

1. 「成果物の生成 (Generate Artifacts)」ウィンドウで、「**新規**」をクリックして新規モジュールを作成します。 「新規統合プロジェクト (New Integration Project)」ウィンドウが表示された場合は、「**モジュール・プロジェクトの作成 (Create a module project)**」を選択して、「**次へ**」をクリックします。

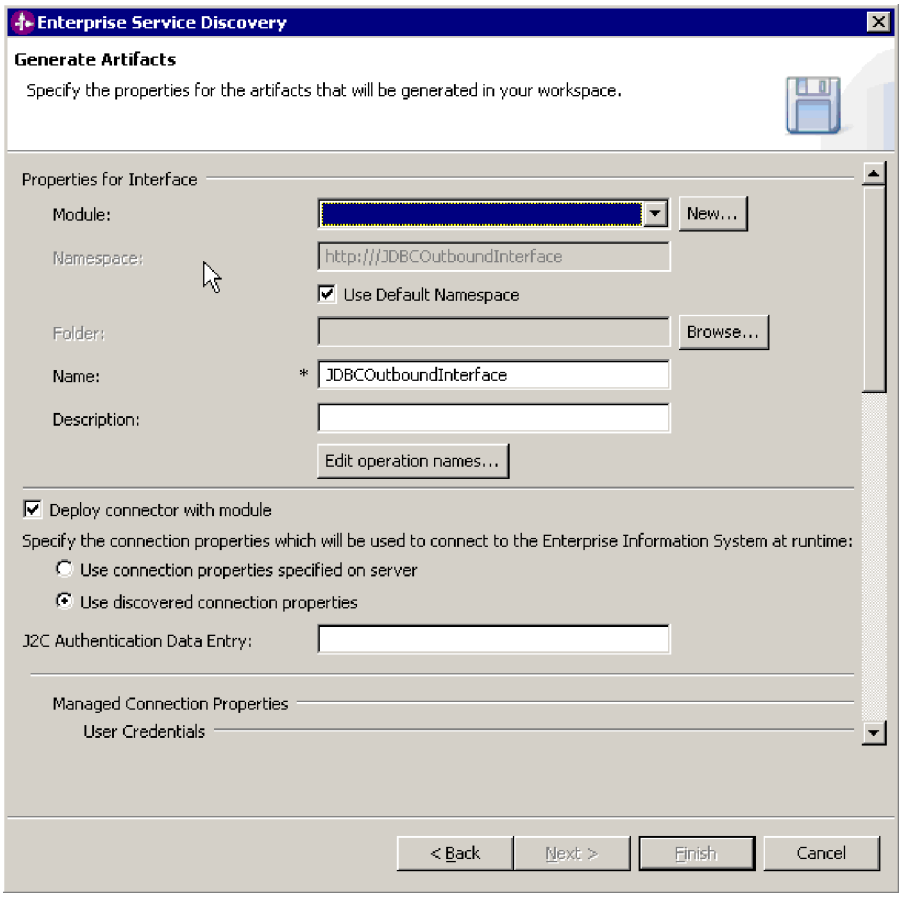


図 41. 「成果物の生成 (Generate Artifacts)」ウィンドウ

2. 「新規モジュール」ウィンドウで、「**モジュール名 (Module Name)**」フィールドに **JDBCOutbound** と入力し、「**終了**」をクリックします。

   デフォルトの「**モジュール・ロケーション (Module Location)**」を変更しないでください。

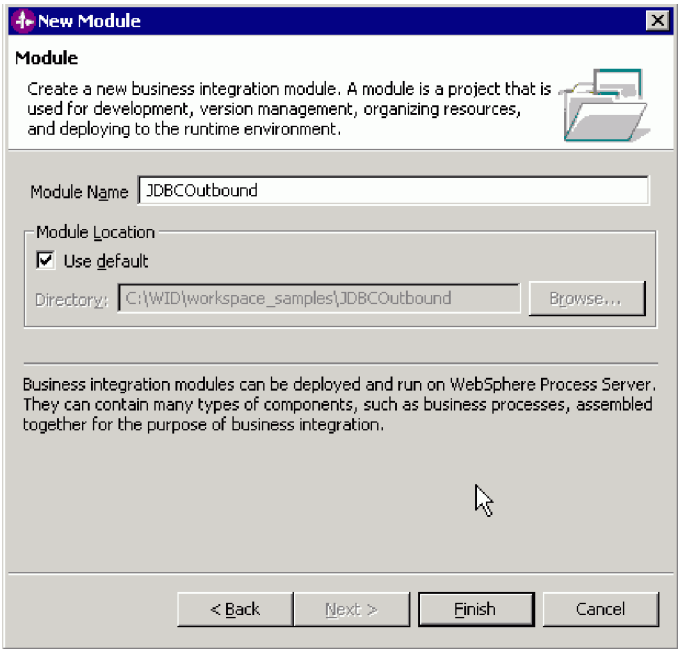


図 42. 「新規モジュール」ウィンドウ

3. 「成果物の生成 (Generate Artifacts)」ウィンドウで、**「ディスカバーされた接続プロパティーを使用 (Use discovered connection properties)」**を選択します。

   これにより、実行時にエンタープライズ情報システムへの接続に使用される構成プロパティー値を設定できます。Outbound 処理の場合は、管理 (J2C) 接続ファクトリー・プロパティーおよびリソース・アダプター・プロパティー用のプロパティーが表示されます。必須のプロパティー・フィールドにはアスタリスクが付いています。

   アダプター構成プロパティーの詳細については、『参照』セクションを参照してください。

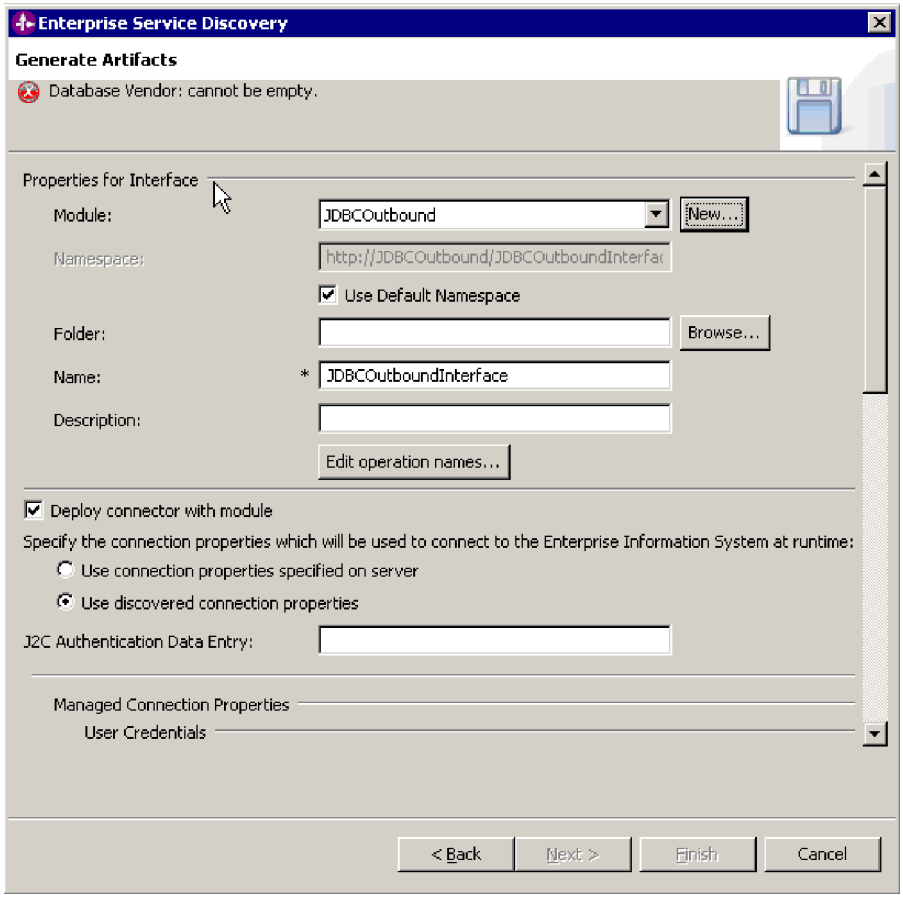


図 43. ディスカバーされた接続プロパティーの使用 (Use discovered connection properties)

4. 「**JDBC 認証データ入力項目 (JDBC Authentication Data Entry)**」に、**SCA_Auth_Alias** と入力します。これは、前に作成した認証別名です。

   名前は大文字小文字が区別されます。

5. 接続プロパティーである「**ユーザー名**」、「**パスワード**」、「**データベース URL**」、および「**JDBC ドライバー・クラス**」を入力します。これらは、エンタープライズ・サービス・ディスカバリー・ウィザードの始動後に設定します。

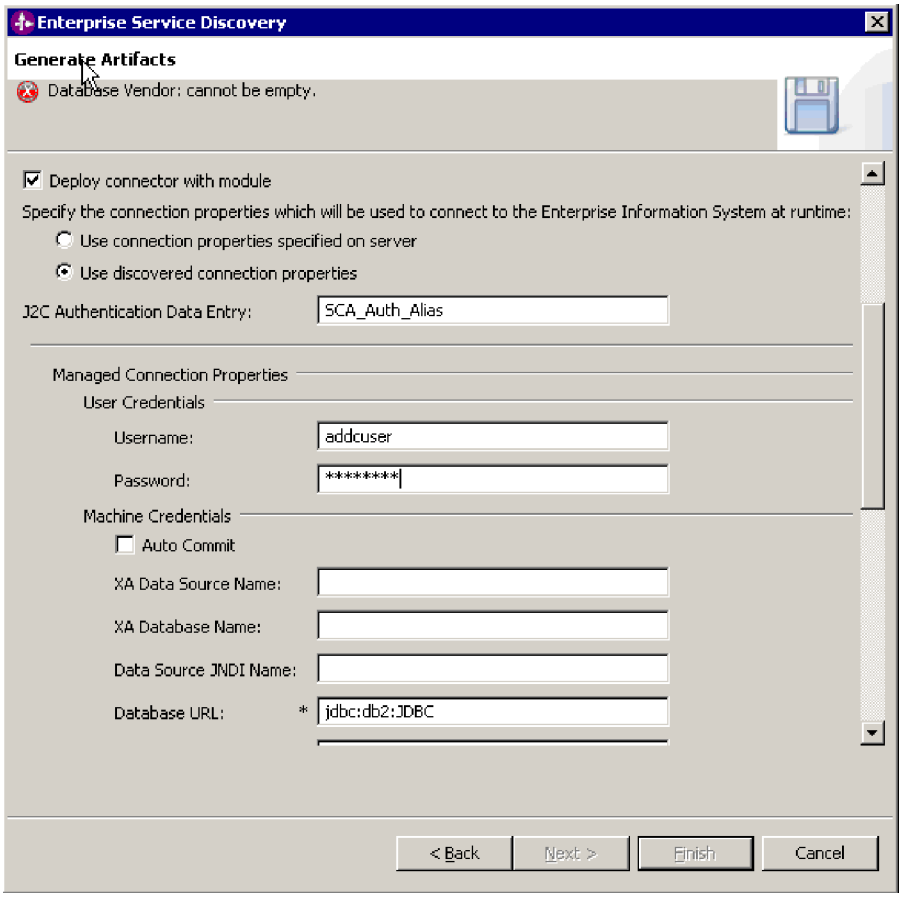


図 44. 別名および管理接続プロパティーの設定

6. アスタリスクでマークされた各プロパティー・フィールドのみに値を指定します。

   アダプター・ログ・ファイルおよびトレース・ファイルが生成されるようにしたい場合は、ロギング・プロパティーおよびトレース・プロパティーも設定できます。次の図で表示されるプロパティー値を使用することができます。
7. 必要なプロパティーの設定を完了したら、「**終了**」をクリックします。

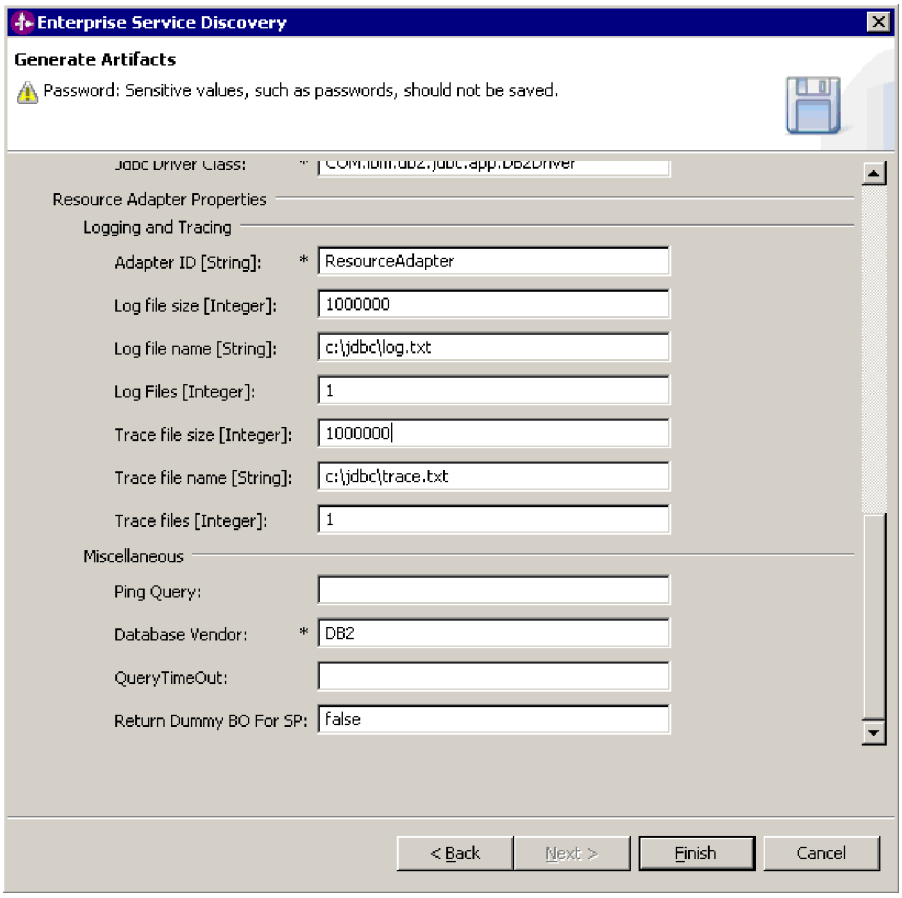


図 45. リソース・アダプター・プロパティーの設定

出力をチュートリアル・サンプル・ファイルと比較することによって、結果を検証します。「ビジネス・インテグレーション」パースペクティブの「ビジネス・インテグレーション」ペインで、**JDBCOutbound** モジュールを展開します。「**データ型**」フォルダーを展開して、すべてのビジネス・オブジェクトのリストを表示します。

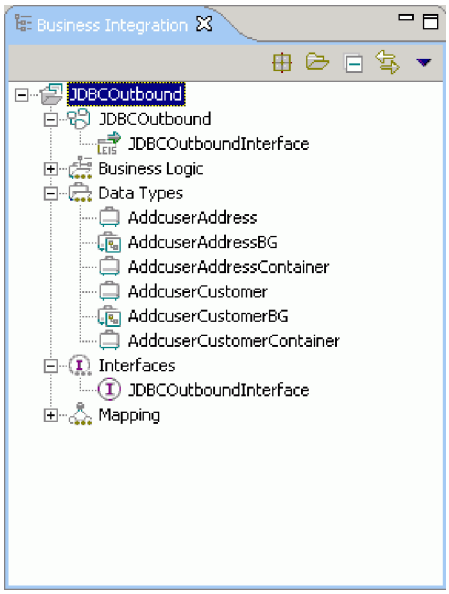


図 46. 成果物の生成の結果の検証

親子関係でのビジネス・オブジェクトのリンク

成果物の生成後、Customer ビジネス・オブジェクトおよび Address ビジネス・オブジェクトを親子関係でリンクする必要があります。追加のアプリケーション固有情報を提供して、オブジェクトのリンクをサポートします。

このタスクの実行方法

1. 「**データ型**」の下で、「**AddcuserCustomer**」をダブルクリックして、ビジネス・オブジェクト・デザイナーでビジネス・オブジェクトを開きます。
2. 「**ビジネス・オブジェクトへの属性の追加 (Add an attribute to a business object)**」のアイコンをクリックします。

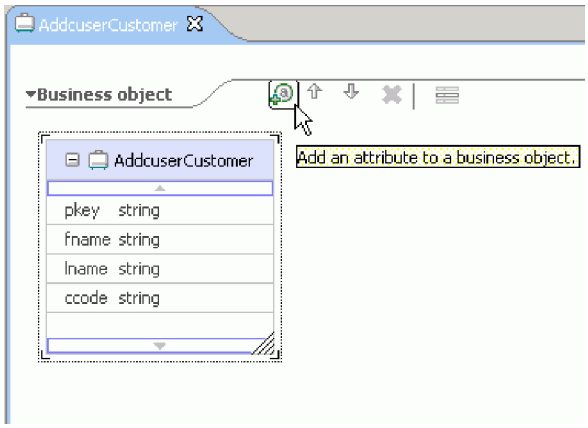


図 47. ビジネス・オブジェクトへの属性の追加

これにより、**attribute1** という新規の属性が作成されます。この属性のタイプは AddcuserAddress です。

3. **addrObj** と入力して、この新規の属性の名前を変更します。

4. **string** タイプをクリックして、属性のタイプを変更します。 この属性に関連付けることのできるタイプのリストが表示されます。 **AddcuserAddress** タイプを選択します。

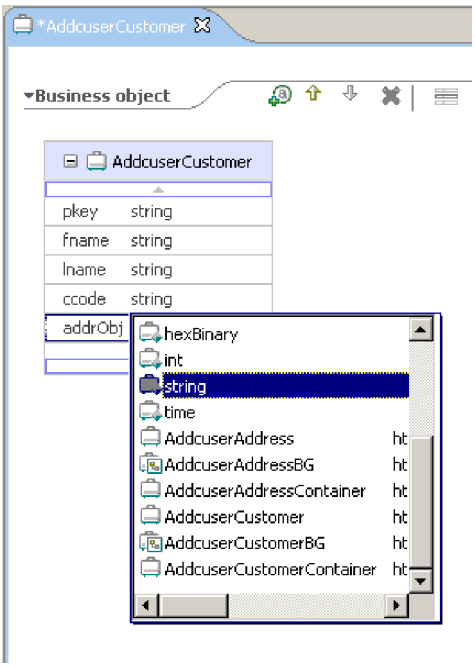


図 48. 属性のタイプの変更

5. この Address 子オブジェクトが複数カーディナリティー・タイプであることを指示します。すなわち、1 つの Customer オブジェクトが複数の Address オブジェクトを持つことができます。
   a. 新規属性 **addrObj AddcuserAddress** を選択します。
   b. 「**プロパティー**」タブで、「**記述**」をクリックします。
   c. 「**Array**」チェック・ボックスを選択します。

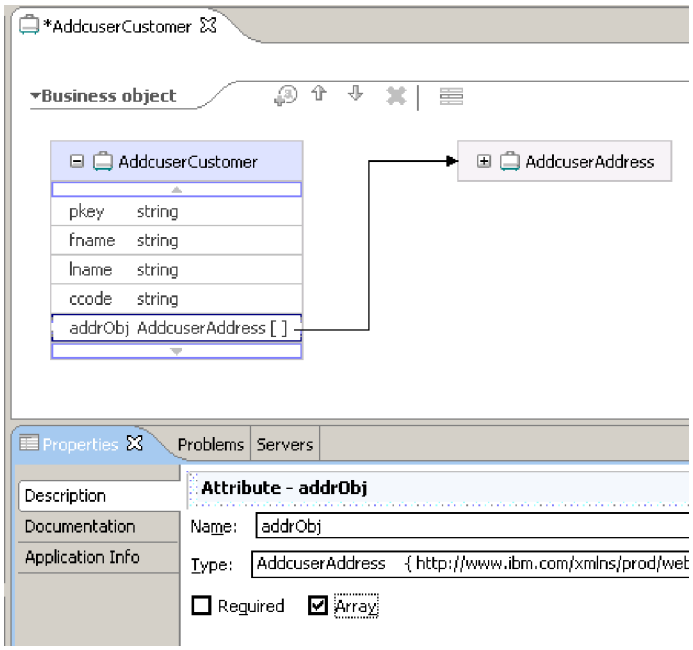


図 49. 複数カーディナリティー用に Array にチェック・マークを付加

6. addrObj AddcuserAddress 属性の属性アプリケーション固有情報を追加します。
   a. 「**プロパティー**」タブで、「**アプリケーション情報**」をクリックします。
   b. **JDBCAttributeTypeMetadata** のチェック・ボックスを選択して、新たに追加された属性に属性アプリケーション固有情報を追加します。

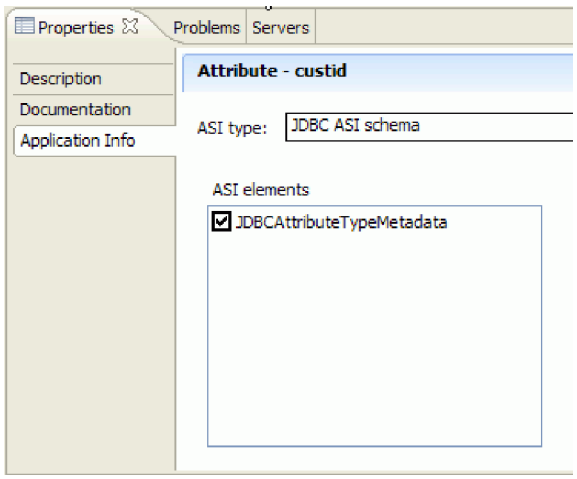


図 50. 属性アプリケーション固有情報の追加

   c. 「**ASI エレメント・プロパティー (ASI element properties)**」の下で、「**jdbcasi: JDBCAttributeTypeMetadata**」 を右クリックします。「**子の追加 (Add Child)**」 → 「**jdbcasi:Ownership**」を選択します。

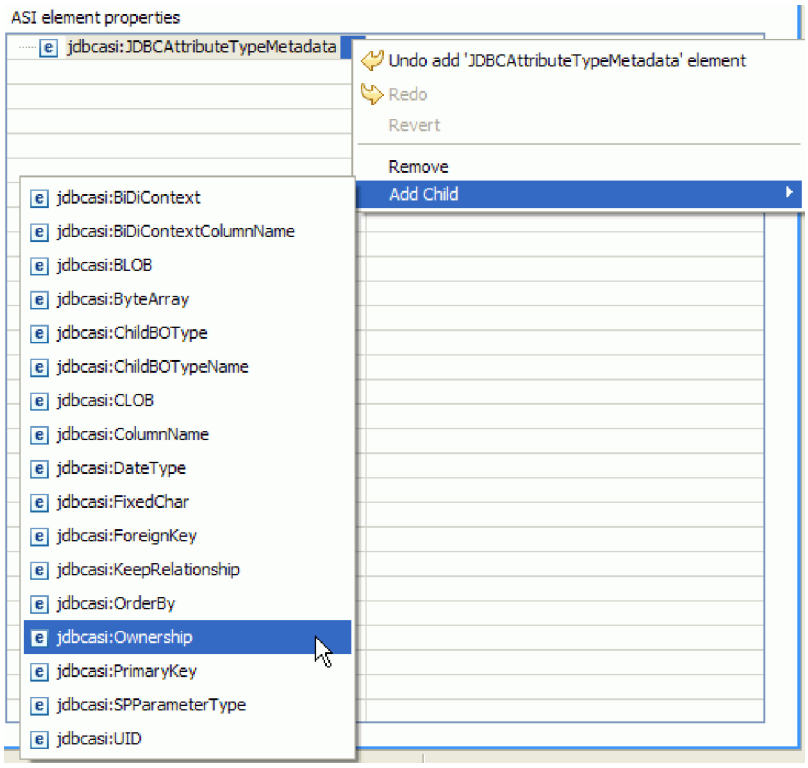


図 51. ASI エレメント・プロパティー

d. **jdbcasi:Ownership** のデフォルト値の **true** を使用します。 所有権値を true に設定することによって、親が子を所有することを示します。すなわち、親がなければ子は存在できません。

e. 「**保管**」をクリックして Customer ビジネス・オブジェクトを保管します。

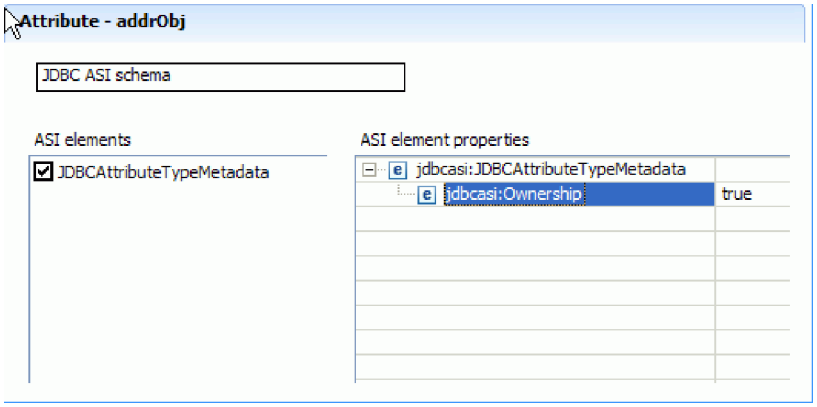


図 52. 所有権の設定

7. AddcuserAddress ビジネス・オブジェクトで外部キーを設定します。

   a. 「**データ型**」の下で、「**AddcuserAddress**」をダブルクリックして、ビジネス・オブジェクト・デザイナーでこのビジネス・オブジェクトを開きます。

   b. **custid** 属性をクリックします。

   c. 「**プロパティー**」タブで、「**アプリケーション情報**」をクリックします。

d. **jdbcasi:JDBCAttributeTypeMetadata** を右クリックします。「**子の追加 (Add Child)**」 → 「**jdbcasi: ForeignKey**」を選択します。値 **pkey** を入力します。

pkey 値は、Customer テーブル内の基本キーを参照する Customer オブジェクト内の属性を表します。

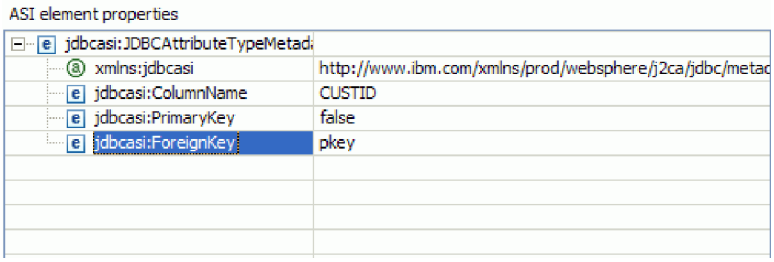


図 53. 外部キーの設定

テスト用のモジュールのデプロイ

エンタープライズ・サービス・ディスカバリー・プロセスにより、エンタープライズ情報システムのインポート・ファイルを含む Service Component Architecture (SCA) モジュールが生成されます。WebSphere Integration Developer を使用して、サーバーにアクセスし、SCA モジュールをサーバーのテスト環境にデプロイします。

1. WebSphere Integration Developer で、「**ウィンドウ (Windows)**」 → 「**ビューの表示 (Show View)**」 → 「**サーバー**」をクリックして、「サーバー」ビューに移動します。

   注: サーバーを始動する前に、データベース・ドライバー JAR ファイルが WebSphere Process Server のランタイムの lib フォルダーに存在することを確認してください。
2. 「**サーバー**」タブをクリックし、「**WebSphere Process Server v6.0.2**」を右クリックして、「**開始**」を選択します。
3. サーバー状況が「**始動済み**」である場合は、サーバーを右クリックし、「**プロジェクトの追加および除去**」を選択します。

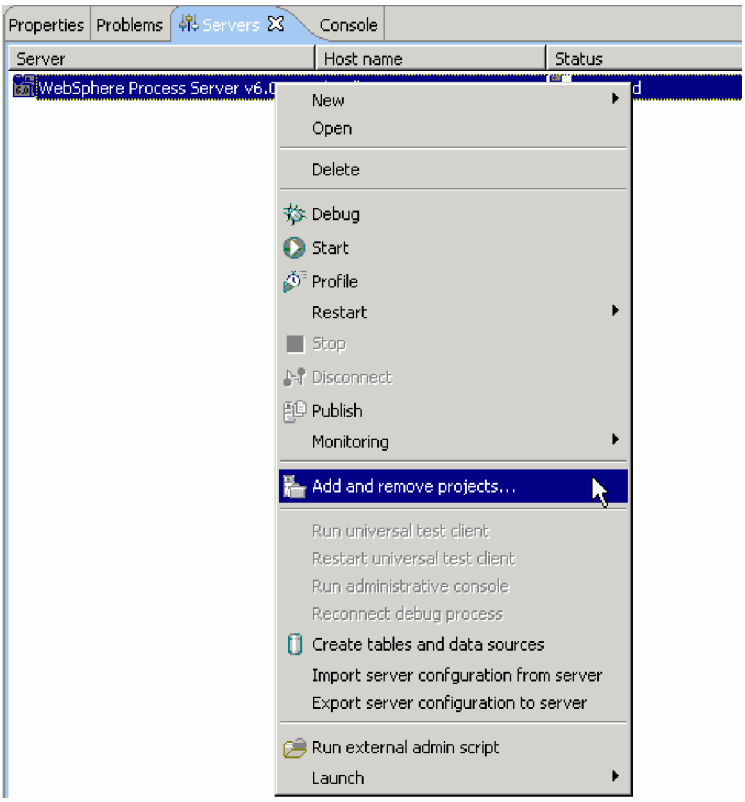


図 54. プロジェクト・オプションの追加および除去

4. 「**JDBCOutboundApp**」を選択して、「**追加**」をクリックします。

   モジュールは、「**使用可能プロジェクト**」リストから「**構成プロジェクト**」リストに移動します。

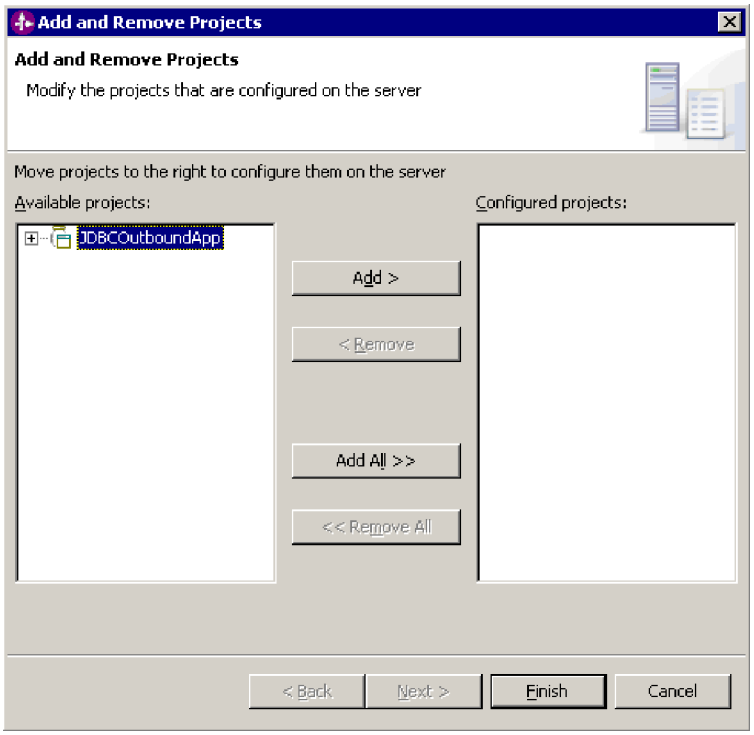


図 55. サーバーにデプロイするモジュールの選択

5. 「**終了**」をクリックします。

   これにより、モジュール (プロジェクトとも呼ばれる) がサーバーにデプロイされます。

   モジュールがサーバーに追加されている間に、右下のペインの「**コンソール**」タブに、状況メッセージ・ログが表示されます。モジュールを追加中に問題が発生した場合は、『チュートリアルのトラブルシューティング』を参照してください。

結果

モジュールをサーバーに追加した後は、データベースでレコードが正しく作成されたことをテストすることができます。

モジュールのテスト

WebSphere Process Server テスト環境を使用して、モジュールをテストします。

1. **JDBCOutboundApp** モジュールを右クリックして、「**テスト**」 → 「**モジュールのテスト (Test Module)**」を選択します。

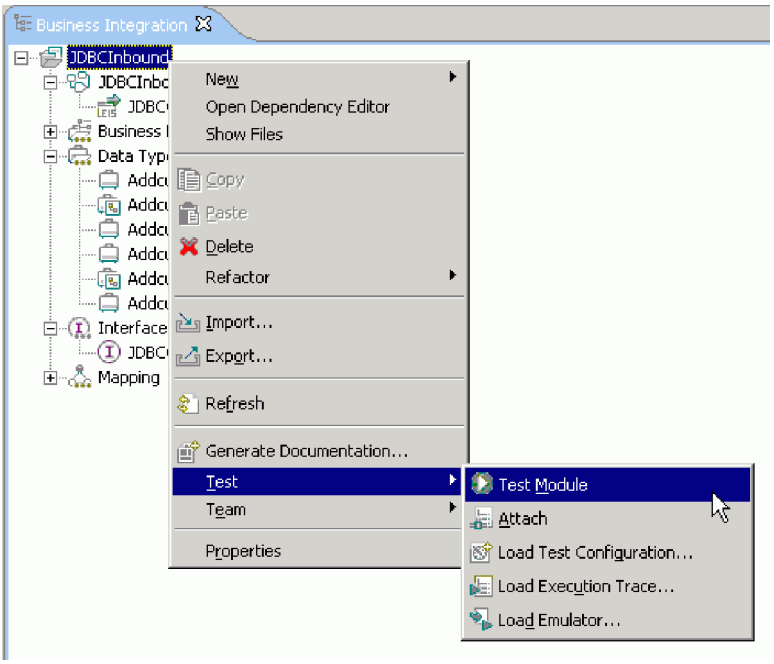


図 56. モジュールのテスト

2. このシナリオでは、データベース内にレコードを作成しました。レコードが正しく作成されたことをテストするには、「**操作**」フィールドで **createAddcuserCustomer** を選択します。

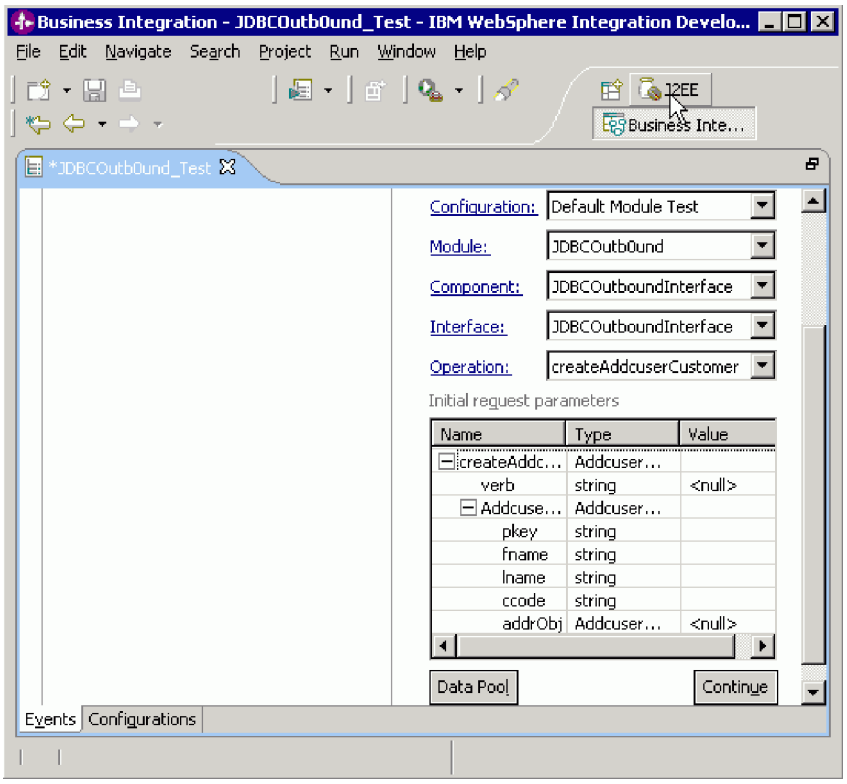


図 57. 作成したレコードの選択

3. 「**動詞 (verb)**」の値として **Create** を選択します。親ビジネス・オブジェクト **AddcuserCustomer** の値を選択します。
4. **addrObj** 属性を右クリックし、「**要素の追加**」を選択して子オブジェクトを作成します。
5. 子オブジェクト **AddcuserAddress** の値を選択します。

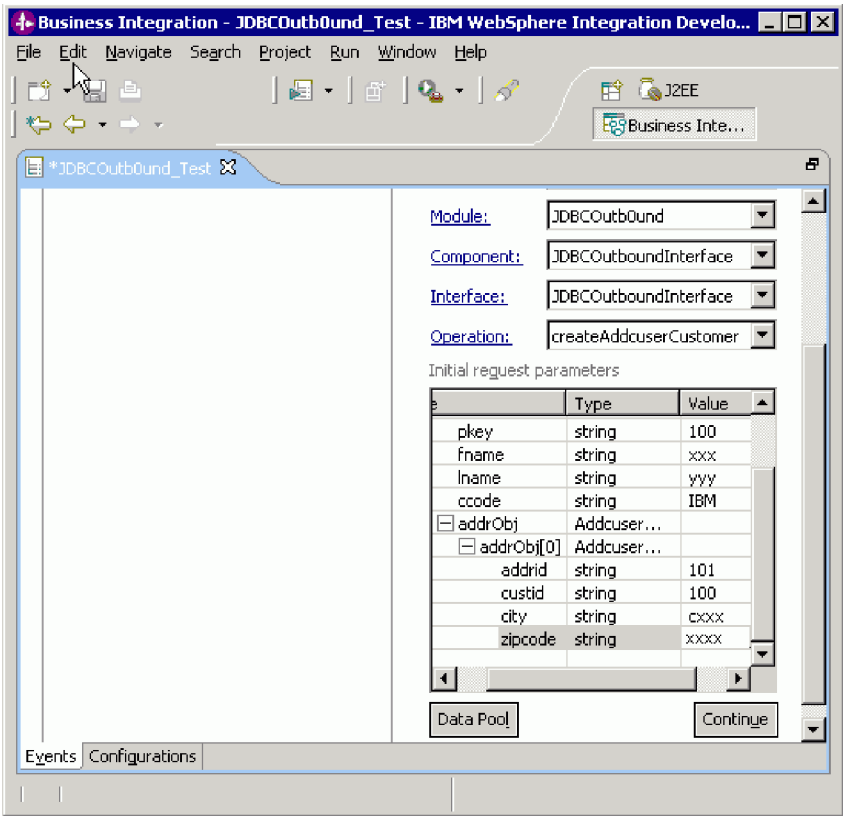


図 58. 子オブジェクトの値の選択

6. 「**続行 (Continue)**」をクリックして、アウトバウンド・サービスを実行します。
7. 「デプロイメント・ロケーションの選択 (Select Deployment Location)」ウィンドウで、「**WebSphere Process Server v6.0.2**」 をクリックします。
8. 「**終了**」をクリックします。
9. アウトバウンド・サービスの出力をチェックして、データベース EIS 内のデータが予期される値と一致すること、すなわち新規レコードがデータベースに挿入されたことを確認します。

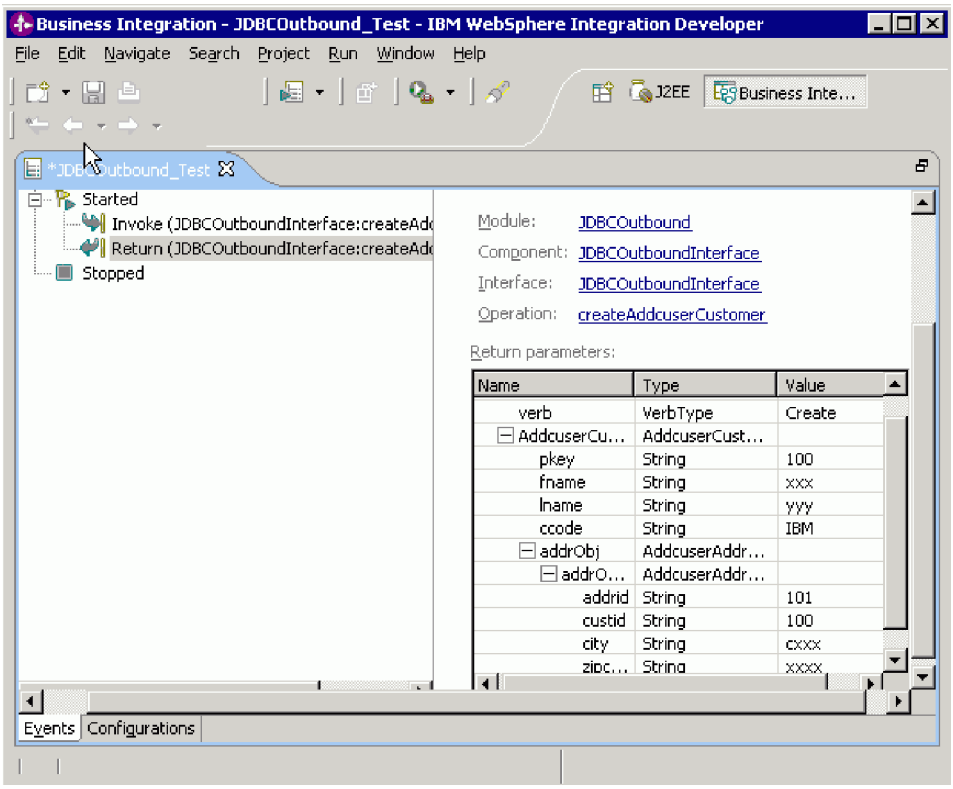


図 59. アウトバウンド・サービスの結果のチェック

チュートリアルのトラブルシューティング

チュートリアルを使用する際には、アダプターが始動に失敗した場合、またはアダプターが処理中に失敗した場合に、対策を取る必要があります。

1. アダプターが始動に失敗する可能性があります。この問題が発生した場合は、アダプター・メッセージ状況ログ・ファイルを調べて、失敗の原因を見つけます。

   アダプターが始動に失敗する原因として、一般的なものを以下に示します。

表 26. アダプター始動失敗の一般的な原因

エラー	原因
ドライバー・クラスが存在しません。 (Driver class does not exist.)	データベース・ドライバー JAR ファイルが、ランタイムの lib フォルダーに存在しません。
ログオン・エラー。ユーザー名/パスワードが無効です。 (Logon error; invalid username/password.)	認証別名に、データベースに接続するための正しいユーザー名またはパスワードが含まれていません。

2. アダプターは処理中に失敗する可能性があります。この問題が発生した場合は、アダプター・メッセージ状況ログ・ファイルを調べて、失敗の原因を見つけます。

アダプターが処理中に失敗する原因として、一般的なものを以下に示します。

表 27. 処理中のアダプターの失敗の一般的な原因

エラー	原因
基本キーが存在しません。 (Primary key does not exist.)	テーブルに、基本キーが定義されていません。したがって、ビジネス・オブジェクトの PrimaryKey アプリケーション固有情報は `true` に設定されていません。
基本キーを持つレコードが既に存在します。 (A record with the primary key already exists.)	レコードは既にデータベースに存在します。データベースに存在しない基本キーを持つレコードを挿入してください。

第 12 章 サンプル・アダプター成果物の表示

各チュートリアルのサンプル成果物を表示するには、アダプターに同梱されるクイック・スタート参照ファイルを IBM WebSphere Integration Developer にインポートします。これらの成果物は、参照専用であることに注意してください。通常は、エンタープライズ情報システム環境では実行されません。チュートリアルを一通り完了していない場合でも、自分で成果物を作成する前に、参照ファイルを使用して、正しく生成された成果物の例を表示できます。

始める前に

クイック・スタート参照ファイルを samples ディレクトリーの referencefiles サブディレクトリーに配置します。各クイック・スタート・チュートリアルには、プロジェクト交換 zip ファイルがあります。例えば、クイック・スタート・チュートリアル 1 には、Tutorial1.zip があります。

重要: クイック・スタート参照ファイルで提供される成果物は、変更または使用しないでください。これらは、表示専用で提供されています。

参照ファイルには、サード・パーティーのライブラリーは含まれていません。 参照ファイルが IBM WebSphere Integration Developer にインポートされると、従属するライブラリーがないという理由でコンパイル・エラー・メッセージが生成されることがあります。参照ファイル内の成果物が、ご使用のエンタープライズ情報システム (EIS) と互換性がない場合があります。 これらは、EIS のバージョンと構成によって異なります。

このタスクの概説

クイック・スタート参照ファイルを WebSphere Integration Developer にインポートし、各クイック・スタート・チュートリアルに関連付けられているサンプル成果物を表示します。

このタスクの実行方法

1. WebSphere Integration Developer の Business Integration パースペクティブで、「**ファイル**」 → 「**インポート**」をクリックします。
2. 「インポート」ウィンドウで「**プロジェクト交換 (Project Interchange)**」を選択し、「**次へ**」をクリックします。
3. 表示したいチュートリアル成果物を含むプロジェクト交換ファイルを選択します。
4. 「**すべてを選択 (Select All)**」をクリックし、プロジェクト交換ファイルにすべてのプロジェクトをインポートします。
5. 「**終了**」をクリックします。

結果

ビジネス・インテグレーション・モジュールが、以下の成果物とともに作成されます。

© Copyright IBM Corp. 2005, 2006

- サービスのインポートおよびエクスポート定義
- ビジネス・オブジェクト (サービス・データ・オブジェクト)
- インターフェース。

第 13 章 参照

アダプター・プロパティー (エンタープライズ・サービス・ディスカバリー接続プロパティー、オブジェクト選択プロパティー、メタデータ選択プロパティー、およびアダプター構成プロパティー)、メッセージ、および関連製品情報に関する詳細情報を参照用に提供します。また、サンプルのインポート・ファイル、エクスポート・ファイル、および WSDL ファイルも提供します。JAR ファイルを WebSphere Integration Developer v6.0.1.1 (およびそれ以前のバージョン) に追加する方法を説明しています。

エンタープライズ・サービス・ディスカバリー接続プロパティー

エンタープライズ・サービス・ディスカバリー処理には、ディスカバリーのエンタープライズ情報システム (EIS) への接続やサービス記述の作成にこれらのプロパティーが必要となります。

これらのプロパティーを使用して、ディスカバリー・サービスは、オブジェクトの選択およびナビゲーションで表示されるメタデータ・ツリーを作成します。

エンタープライズ・サービス・ディスカバリー・ウィザードは、双方向接続プロパティーを使用して、エンタープライズ情報システムに渡すデータに適切な双方向変換を適用します。

WebSphere Integration Developer のエンタープライズ・サービス・ディスカバリー・ウィザードを実行する場合は、以下にリストした接続プロパティーを指定します。

Adapter for JDBC 用のエンタープライズ・サービス・ディスカバリー接続プロパティー

プロパティー	必須であるかどうか	グローバル化	説明
データベース URL	はい	はい	接続先のデータベースの URL。URL は、ドライバーによって異なる場合があります。このプロパティーはドライバーの説明に従って指定してください。このプロパティーは双方向言語の場合に使用可能。
JDBC ドライバー・クラス	はい	いいえ	データベースへの接続に使用されるデータベース・ドライバーの名前
パスワード	はい	はい	対応するユーザー名に対するパスワード。このプロパティーは双方向言語の場合に使用可能。

© Copyright IBM Corp. 2005, 2006

プロパティー	必須であるかどうか	グローバル化	説明
プレフィックス	いいえ	はい	ビジネス・オブジェクトの名前に追加されるプレフィックス。このプロパティーは双方向言語の場合に使用可能。
ユーザー名	はい	はい	データベースへのログイン用のユーザー名。このプロパティーは双方向言語の場合に使用可能。

双方向接続プロパティー

プロパティー	値	文字の位置	説明
BiDi 順序付けスキーマ	Implicit Visual	1	論理 (暗黙)。Implicit がデフォルト。 Visual
BiDi 方向	LTR RTL contextual_LTR contextual_RTL	2	左から右。LTR (L) がデフォルト。 右から左 コンテキスト LTR コンテキスト RTL
BiDi 対称スワッピング	はい いいえ	3	対称スワッピングはオン。はい (Y) がデフォルト。 対称スワッピングはオフ。
BiDi シェーピング	nominal shaped initial middle final isolated	4	公称 (N)。これはデフォルトです。 位置に従って形状を指定された文字。 文字は語頭形。 文字は語中形。 文字は語尾形。 文字は独立形。
BiDi 数字シェーピング	nominal national contextual	5	公称 (N)。これはデフォルトです。 ヒンディ語 (公用語) コンテキスト

オブジェクト選択プロパティー

エンタープライズ・サービス・ディスカバリー・ウィザードは、ビジネス・オブジェクトの選択処理中に、フィルター・プロパティーおよびノード・プロパティーを使用します。これらのプロパティーを設定すると、ツリー構造で表示されるスキーマ、ノード、またはビジネス・オブジェクトのリストが絞り込まれます。オブジェクトの選択後に、これらのプロパティーを設定してビジネス・オブジェクトを構成します。

フィルターおよびノードのプロパティー

ツリーに表示されるスキーマ、ノード・タイプ、またはデータベース・オブジェクトのリストを絞り込む場合は、ビジネス・オブジェクトのディスカバリー中にフィルター・プロパティーとノード・プロパティーを指定できます。これらを指定しないと、すべてのスキーマ、ノード・タイプ、またはデータベース・オブジェクトが表示されます。また、これらのプロパティーを指定して、指定した SELECT ステートメントからビジネス・オブジェクトを生成することもできます。

スキーマまたはノード・タイプをフィルターに掛けるか、ビジネス・オブジェクトのアプリケーション固有情報を設定するか、または指定した SELECT ステートメントからビジネス・オブジェクトを生成するためには、以下の表にリストされたプロパティーを使用します。

表 28. フィルター・プロパティー

フィルター・プロパティー	型	説明	デフォルト値
DefineASIs	Boolean	true の場合、生成するオブジェクトを選択すると、ビジネス・オブジェクトのアプリケーション固有情報 (ASI) 値を設定するよう求めるプロンプトが出されます。チェック・ボックスを選択して、このプロパティーを true に設定します。	false
QueryBO	Boolean	true の場合は、クエリー・ビジネス・オブジェクトを生成するために SELECT ステートメントを指定できます。チェック・ボックスを選択して、このプロパティーを true に設定します。	false
QueryBOCount	Integer	同時に生成できるクエリー・ビジネス・オブジェクトの最大数。このプロパティーは、QueryBO プロパティーが true に設定されている場合にのみ使用可能になります。	1
SchemaNameFilter	String	スキーマをフィルターに掛けるために使用するストリング。指定したストリングで始まるスキーマのみが表示されます。このプロパティーを設定しないと、すべてのスキーマが表示されます。	なし
Types	String: 複数の値	リストされたすべてのスキーマの下に表示するデータベース・オブジェクト・タイプのノード (テーブル、ビュー、ストアード・プロシージャー、または同義語/ニックネーム) を決定します。値はエンタープライズ・サービス・ディスカバリーによって設定されます。複数のタイプを選択できます。	なし

データベース・オブジェクトをフィルターに掛けるには、以下の表にリストされたプロパティーを使用します。

表 29. ノード・プロパティー

ノード・プロパティー	型	説明	デフォルト値
ObjectNameFilter	String	データベース・オブジェクトをフィルターに掛けるために使用するストリング。指定したテキストで始まるデータベース・オブジェクトのみが表示されます。ObjectNameFilter を指定しないと、すべてのオブジェクトが表示されます。	なし

メタデータ選択プロパティー

データベース・オブジェクトを選択した後、メタデータ選択プロパティーの値を設定する必要があります。エンタープライズ・サービス・ディスカバリー・ウィザードは、ビジネス・オブジェクトを生成するための元となる選択済みデータベース・オブジェクトのリストを確認した後、メタデータ選択プロパティーを照会します。

メタデータ値を設定するには、以下の表にリストされたプロパティーを使用します。

表 30. メタデータ選択プロパティー

プロパティー	型	説明
BO ロケーション	String	生成された .xsd ファイルが保管されるロケーションへのパス。 DataDescription で RelativePath として設定されます。
レコード最大数	Integer	RetrieveALL 操作で検索するレコードの最大数。この値は、OutboundServiceDescription の InteractionSpec のインスタンスに取り込まれます。デフォルト値は 100 です。
ネーム・スペース	String	この値は、ビジネス・オブジェクト名を論理的に分離するため、ビジネス・オブジェクト名の前に付加されます。例: `http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc/Schema1Customer` このプロパティーは、最初はすべてのビジネス・オブジェクトでデフォルトのネーム・スペース `http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc` に設定されています。
操作	String	デフォルトでは、選択されたサービス・タイプに対してアダプターがサポートする操作のリストです。これらの操作から選択します。指定された操作は、生成されるすべてのビジネス・オブジェクトに対して設定されます。クエリー・ビジネス・オブジェクトは、RetrieveAll のみをサポートします。
サービス・タイプ	String	Inbound または Outbound が選択されます。この値は、エンタープライズ・サービス・ディスカバリーによって設定されます。クエリー・ビジネス・オブジェクトとストアード・プロシージャー・ビジネス・オブジェクトは、Outbound サービス・タイプのみをサポートします。

アダプター構成プロパティー

このプロパティー・グループには、特定のデータベース・アプリケーションへの通信チャネルをセットアップするためにアダプターが使用する属性が含まれています。これらのプロパティーには、リソース・アダプター・プロパティー、管理 (J2C) 接続ファクトリー・プロパティー、およびアクティベーション・スペック・プロパティーが含まれています。

リソース・アダプター・プロパティー

リソース・アダプター・プロパティーは、ロギングおよびトレース、双方向言語サポート、さらにはアダプターのデフォルト構成プロパティーなどのアダプター固有のアクティビティーから構成されます。エンタープライズ・サービス・ディスカバリー・ウィザードを使用するか、サーバーの管理コンソールを使用して、これらのプロパティーを構成します。

アダプターを構成する際は、以下の表にリストされている各リソース・アダプター・プロパティーを指定します。双方向テキスト変換が活動化されているときに構成する必要がある双方向リソース・アダプターのプロパティーについては、このセクションの別の表で説明しています。

表 31. Adapter for JDBC 用のリソース・アダプター・プロパティー

プロパティー	型	説明	グローバル化	デフォルト値
BO ネーム・スペース	String	このアダプターで使用されるビジネス・オブジェクト定義のネーム・スペースです。この値は、メタデータ・ディスカバリー処理で指定された値から取得されます。このプロパティーは必須です。	いいえ	なし
データベース・ベンダー	String	特殊な処理の際に、アダプターが使用する RDBMS を指定します。IBM DB2、Oracle、または Microsoft SQL Server データベースを使用する場合は、値を `DB2`、`Oracle`、または `MSSQLServer` に設定してください。database. 異なるデータベースを使用する場合、値は該当するデータベースの名前に設定してください。デフォルト以外のデータベースを使用している場合は、適切なドライバーがロードされていることを確認してください。このプロパティーが `「Others」` に設定された場合、アダプターは、ドライバーを探して、使用するデータベースを決定します。アダプターで処理が正常に行われるようにするには、なんらかの値が必要です。	いいえ	なし
enableHASupport	String	enableHASupport プロパティーが `true` に設定されている場合、複製されたアダプター・インスタンスのうちの 1 つのみがイベントをアクティブにポーリングし、その他のインスタンスは待機モードになります。enableHASupport プロパティーが `false` に設定されている場合は、クラスター・メンバー上の複製されたすべてのアダプター・インスタンスがイベントをアクティブにポーリングします。これにより、イベントが重複する場合があります。単一サーバー環境では、enableHASupport の値を false に変更しないでください。		true
ログ・ファイル名	String	ログ・ファイルの絶対パス。このプロパティーは必須です。	いいえ	なし
ログ・ファイル数	Integer	使用するログ・ファイルの数。ログ・ファイルが最大サイズに達すると、別のログ・ファイルが開始されます。	いいえ	1
ログ・ファイル・サイズ (LogMaxFileSize)	Integer	ログ・ファイルのサイズ (キロバイト単位)。値を指定しないと、ファイルに最大サイズが設定されません。	いいえ	なし
Ping クエリー	String	データベースへの接続の有効性をテストするのに使用する SQL 照会。	いいえ	なし

表 31. Adapter for JDBC 用のリソース・アダプター・プロパティー (続き)

プロパティー	型	説明	グローバル化	デフォルト値
クエリー・タイムアウト	Integer	これにより、すべての SQL ステートメントで照会にかかる最大時間を秒数で指定します。値を指定しない場合は、照会にタイムアウトを設定しません。照会の処理に、指定された秒数より長い時間が必要な場合は、データベースにより、キャプチャーされる SQL 例外が生成されます。関連メッセージが、ログ・ファイルに記録されます。	はい	なし
ReturnDummyBO ForSP	Boolean	このプロパティーは、結果のセットが空の場合でも出力パラメーターを戻すために使用されます。 RetrieveSP の場合、結果のセットが戻されます。結果のセットが空の場合は、ビジネス・オブジェクトが生成されず、プロシージャー呼び出しの戻す出力パラメーターもリトリーブできません。このプロパティー値が `true` の場合、出力の値を持つダミーのビジネス・オブジェクトと、対応する属性にデータが取り込まれた入出力パラメーターが戻されます。	いいえ	false
トレース・ファイル名	String	トレース・ファイルの絶対パス。このプロパティーは必須です。	いいえ	なし
トレース・ファイル数	Integer	使用するトレース・ファイルの数。トレース・ファイルが最大サイズに達すると、別のトレース・ファイルが開始されます。	いいえ	1
トレース・ファイルのサイズ (TraceFileSizeMax)	Integer	トレース・ファイルのサイズ (キロバイト単位)。値を指定しないと、ファイルに最大サイズが設定されません。	いいえ	なし

以下の表で、双方向テキスト変換が活動化されているときにのみ構成する必要があるリソース・アダプター・プロパティーについて説明します。

Adapter for JDBC 用の双方向リソース・アダプター・プロパティー

プロパティー	型	説明
BiDi コンテキスト EIS	String	ビジネス・オブジェクトのランタイム・インスタンスすべてにあるコンテンツ・データの双方向形式を定義します。
BiDiContextMetadata	String	デプロイメント記述子に保管されたメタデータまたは構成プロパティーの双方向形式を定義します。
BiDiContextSkip	Boolean	リソース・アダプター・レベルでの双方向言語サポートの呼び出しを制御するフラグ。`true` と等しければ、サポートは呼び出されません。`false` と等しければ、サポートは呼び出されます。

プロパティー	型	説明
BiDiContextSpecialFormat	String	コネクター・プロパティーのすべてにおいて、特殊な処理を受けるプロパティーのカテゴリーを指定します。
BiDiContextTurnBiDiOff	Boolean	JDBC アダプターの双方向変換の呼び出しを制御するフラグ。このパラメーターは、すべてのレベルの他の双方向パラメーターのどれよりも高い優先順位を持っています。言い換えると、この値が `true` に設定されている場合には双方向言語サポートが呼び出されず、チェックや検索も実行されません。 `false` に設定されている場合は、双方向言語サポートが呼び出されます。

管理 (J2C) 接続ファクトリー・プロパティー

管理接続ファクトリー (MCF) 構成プロパティーは、エンタープライズ情報システムとの Outbound 接続インスタンスを作成するためにランタイムに使用されます。MCF プロパティーは作成後、デプロイメント記述子に格納されます。

J2C 接続ファクトリーは、接続プールを管理します。これは、アダプターによるアプリケーションから単一の JDBC アプリケーション・インスタンスへの Outbound 接続に関する構成情報を提供します。

アダプターを構成する際は、以下の表にリストされている各プロパティーを指定します。このセクションにある 2 番目と 3 番目の表では、双方向テキスト変換が活動化されているときにのみ構成する必要がある双方向管理 (J2C) 接続ファクトリー・プロパティーを定義しています。

注: エンタープライズ・サービス・ディスカバリー・ウィザードではこれらのプロパティーを管理接続プロパティーと呼び、WebSphere Process Server では (J2C) 接続ファクトリー・プロパティーと呼びます。

表 32. Adapter for JDBC 用の管理 (J2C) 接続ファクトリー・プロパティー

プロパティー	型	説明	グローバル化	サポートされる双方向変換
自動コミット	Boolean	接続で設定される自動コミットの値。	いいえ	いいえ
データベース URL	String	データベースへの接続に使用されるデータベース URL。	はい	はい

表 32. *Adapter for JDBC 用の管理 (J2C) 接続ファクトリー・プロパティー (続き)*

プロパティー	型	説明	グローバル化	サポートされる双方向変換
DataSourceJNDIName		データベースへの接続の確立に使用される JNDI データ・ソース名。ユーザー名プロパティーおよびパスワード・プロパティーも設定されている場合は、それらも接続の確立時に使用されます。ユーザー名プロパティーもパスワード・プロパティーも設定されていない場合は、このプロパティーのみが使用されます。	はい	いいえ
JDBC ドライバー・クラス	String	データベースへの接続に使用されるドライバーの JDBC ドライバー・クラス。	いいえ	いいえ
パスワード	String	対応するユーザー名に対するパスワード。	はい	はい
ユーザー名	String	エンタープライズ情報システムにログインするためのユーザー名。	はい	はい
XADataSourceName	String	データベースへの XA 接続を確立するための XA データ・ソース名。	いいえ	いいえ
XADatabaseName	String	XA 接続で使用するデータベース名。IBM DB2 データベースを使用している場合は、このプロパティーを設定する必要があります。	はい	はい

Adapter for JDBC 用の双方向管理 J2C 接続ファクトリー・プロパティー

プロパティー	型	説明
BiDi コンテキスト EIS	String	EIS への固有接続用アダプターによってサポートされるビジネス・オブジェクト・ランタイム・インスタンスすべてにあるコンテンツ・データの双方向形式を定義します。
BiDiContextMetadata	String	EIS への固有接続のメタデータまたは構成プロパティーの双方向形式を定義します。
BiDiContextSkip	Boolean	リソース・アダプター・レベルでの双方向言語サポートの呼び出しを制御するフラグ。`true` と等しければ、サポートは呼び出されません。`false` と等しければ、サポートは呼び出されます。
BiDiContextSpecialFormat	String	EIS への固有接続のコネクター・プロパティーすべての場合に、特殊な処理を受けるプロパティーのカテゴリーを指定します。

プロパティー	型	説明
データベース URL BiDi プロパティー		*DatabaseURLEIS、DatabaseURLSpecialFormat (デフォルト値は JDBC_URL_SQL)、DatabaseURLSkip
パスワード BiDi プロパティー		*PasswordEIS、PasswordSkip
ユーザー名 BiDi プロパティー		*UserNameEIS、UserNameSkip
XADatabaseName BiDi プロパティー		*XADatabaseNameEIS、XADatabaseNameSkip

*BiDi サポートの各プロパティーに関連付けられた 3 つの双方向 (BiDi) プロパティー。これらのプロパティーは以下の表に記述されています。

BiDi プロパティー	型	説明
BiDiContextCP<property_Name>EIS	String	EIS への固有接続の固有コネクター構成プロパティーの双方向形式を定義します。
BiDiContextCP<property_Name>SpecialFormat	String	コネクター構成プロパティー・レベルで双方向言語サポートの呼び出しを制御するフラグ。true と等しければ、サポートは呼び出されません。false と等しければ、サポートは呼び出されます。
BiDiContextCP<property_Name>Skip	Boolean	EIS への固有接続における固有コネクター・プロパティーの特殊形式を定義します。

アクティベーション・スペック・プロパティー

アクティベーション・スペック・プロパティーは、メッセージ・エンドポイント用の Inbound イベント処理の構成情報を保持します。アクティベーション・スペック・プロパティーは、エンタープライズ・サービス・ディスカバリー・ウィザード、WebSphere Application Server 管理コンソール、または WebSphere Enterprise Service Bus 管理コンソールのいずれかを介して設定できます。

アダプターを構成する際は、以下の表にリストされている各アクティベーション・スペック・プロパティーを指定します。 2 番目の表では、双方向テキスト変換が使用されている場合にのみ構成する必要がある双方向アクティベーション・スペック・プロパティーについて説明しています。

表 33. Adapter for JDBC 用のアクティベーション・スペック・プロパティー

プロパティー	型	説明	グローバル化	デフォルト値
AssuredOnceDelivery	Boolean	プロパティーを true に設定すると、イベント・ストアの各イベントに xid 値が設定されます。さらに、各イベントが対応するエンドポイントに送達され、その後イベント・テーブルから削除されます。	いいえ	true

表 33. *Adapter for JDBC* 用のアクティベーション・スペック・プロパティー *(続き)*

プロパティー	型	説明	グローバル化	デフォルト値
CustomDeleteQuery	String	各イベントが処理された後に実行されるカスタム削除照会。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
CustomEventQuery	String	カスタム・イベント処理用の SQL 照会、ストアード・プロシージャー、またはストアード関数。この照会は、EventQueryType が Dynamic に設定されている場合に、ポーリング周期中に毎回実行されます。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
CustomUpdateQuery	String	同じイベントが次回のイベント周期で処理対象として取り出されないように、各イベントが処理された後に実行されるカスタム更新照会。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
データベース URL	String	エンタープライズ情報システム (EIS) への接続を作成するためのドライバー固有の URL。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
DataSource JNDIName	String	データベースへの接続を確立するためにアダプターによって使用される名前。ユーザー名およびパスワードも設定されている場合は、それらも接続の確立時に使用されます。ユーザー名もパスワードも設定されていない場合は、DataSourceJNDIName プロパティーのみが使用されます。	はい	なし
送達タイプ	String	このプロパティーにより、イベントをパブリッシュする順序が決定されます。値は、`ORDERED` または `UNORDERED` のいずれかです。*Ordered* は、イベントが一度に 1 つずつパブリッシュされることを、*unordered* は、イベントが一度にすべてパブリッシュされることを意味します。	はい	ORDERED
EventFilterType	String	アダプターは、処理するイベントをビジネス・オブジェクト・タイプでフィルターに掛けることができます。 EventFilterType には、コンマで区切られたビジネス・オブジェクト・タイプのリストが含まれ、このプロパティーで指定されたタイプのみが処理対象として取り出されます。プロパティーに値が指定されていない場合、フィルターは適用されず、すべてのイベントが処理対象として取り出されます。	はい	NULL

表 33. Adapter for JDBC 用のアクティベーション・スペック・プロパティー (続き)

プロパティー	型	説明	グローバル化	デフォルト値
イベント順序	String	イベントが取得および処理される順序。期待される値はイベント・テーブルのコンマで区切られた列名です。このプロパティーは双方向言語の場合に使用可能。	いいえ	なし
EventQueryType	String	標準イベント・ストアまたはカスタム照会のいずれを使用するかを決定します。有効な値は、標準イベント・ストア用の `Standard` とカスタム・イベント処理用の `Dynamic` です。	いいえ	なし
イベント・テーブル名	String	EIS が Inbound 処理のために生成するイベントを格納する EIS 内のテーブル。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
FilterFutureEvents	Boolean	このプロパティーが `true` に設定されている場合、アダプターは、タイム・スタンプに基づいてイベントをフィルターに掛けます。アダプターは、各ポーリング周期のシステム時刻を各イベントのタイム・スタンプと比較します。イベントが将来発生するように設定されている場合は、その時刻になるまで処理対象として取り出されません。	いいえ	false
JDBC ドライバー・クラス	String	EIS へ接続するために使用される JDBC ドライバー・クラス。	いいえ	なし
パスワード	String	EIS からイベントを検索する際にユーザーを認可するためのパスワード。このプロパティーは双方向言語の場合に使用可能。	はい	なし
ポーリング間隔	0 以上の整数	新規 Inbound イベントの EIS イベント・ストアをポーリングする間隔 (ミリ秒単位)。`0` を指定すると、アダプターは周期の間待機することはありません。ポーリング周期は固定の間隔で設定されます。つまり、ポーリング周期の実行が遅延すると (例えば、前のポーリング周期の完了が予想よりも遅れた場合など)、遅れを取り戻すために次の周期がすぐに開始されます。このプロパティーは必須です。	はい	500
ポーリング数量	0 より大きい整数。	このプロパティーは、ポーリング周期ごとに各エンドポイントに配信するイベントの数を決定するのに使用されます。このプロパティーは必須です。	はい	なし
RetryLimit	Integer	アダプターによる EIS への接続の再試行の回数。値 `0` (ゼロ) を指定すると、アダプターは無限に試行を繰り返します。	いいえ	0

表 33. Adapter for JDBC 用のアクティベーション・スペック・プロパティー (続き)

プロパティー	型	説明	グローバル化	デフォルト値
RetryInterval	Integer	この値が非負数のとき、アダプターで EIS へのインバウンド接続に関連するエラーが発生した場合、アダプターは、接続が再度確立されるまで、指定された間隔後に接続の再試行を繰り返します。値はミリ秒単位です。	いいえ	60000 (1 分)
SPAfterPoll	String	ポーリング周期の終了ごとに、実行する任意のストアード・プロシージャー。これは、1 つの入力パラメーター (ポーリング数量) を取ります。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
SPBeforePoll	String	実際のポーリング照会の呼び出しの前に実行する任意のストアード・プロシージャー。これは、1 つの入力パラメーター (ポーリング数量) を取ります。これは双方向言語で使用される場合に使用可能化されます。	はい	なし
StopPollingOnError	Boolean	`true` の場合、ポーリング中にエラーが発生すると、アダプターはポーリングを停止します。`false` の場合、エラーが発生すると、アダプターは例外をログに記録し、ポーリングを続行します。	いいえ	false
ユーザー名	String	Inbound イベントのため EIS にログインする際のユーザー名。これは双方向言語で使用される場合に使用可能化されます。	はい	なし

Adapter for JDBC 用の双方向アクティベーション・スペック・プロパティー

プロパティー	型	説明
BiDi コンテキスト EIS	String	EIS への固有接続用アダプターによってサポートされるビジネス・オブジェクト・ランタイム・インスタンスすべてにあるコンテンツ・データの双方向形式を定義します。
BiDiContextMetadata	String	EIS への固有接続のメタデータまたは構成プロパティーの双方向形式を定義します。
BiDiContextSkip	Boolean	リソース・アダプター・レベルでの双方向言語サポートの呼び出しを制御するフラグ。`true` と等しければ、サポートは呼び出されません。`false` と等しければ、サポートは呼び出されます。

プロパティー	型	説明
BiDiContextSpecialFormat	String	EIS への固有接続のコネクター・プロパティーすべての場合に、特殊な処理を受けるプロパティーのカテゴリーを指定します。
CustomDeleteQuery BiDi プロパティー		*CustomDeleteQueryEIS CustomDeleteQuerySkip
CustomEventQuery BiDi プロパティー		*CustomEventQueryEIS CustomEventQuerySkip
CustomUpdateQuery BiDi プロパティー		*CustomUpdateQueryEIS CustomUpdateQuerySkip
データベース URL BiDi プロパティー		*DatabaseURLEIS、DatabaseURLSpecialFormat (デフォルト値は JDBC_URL_SQL)、DatabaseURLSkip
イベント・テーブル名 BiDi プロパティー		*EventTableNameEIS、EventTableNameSkip
EventOrderBy BiDi プロパティー		*EventOrderByEIS、EventOrderBySkip
パスワード BiDi プロパティー		*PasswordEIS、PasswordSkip
SPAfterPoll BiDi プロパティー		*SPAfterPollEIS、SPAfterPollSkip
SPBeforePoll BiDi プロパティー		*SPBeforePollEIS、SPBeforePollSkip
ユーザー名 BiDi プロパティー		*UserNameEIS、UserNameSkip

*BiDi サポートの各プロパティーに関連付けられた 3 つの双方向 (BiDi) プロパティー。これらのプロパティーは以下の表に記述されています。

BiDi プロパティー	型	説明
BiDiContextCP<property_Name>EIS	String	EIS への固有接続の固有コネクター構成プロパティーの双方向形式を定義します。
BiDiContectCP<property_Name>SpecialFormat	String	コネクター構成プロパティー・レベルで双方向言語サポートの呼び出しを制御するフラグ。`true` と等しければ、サポートは呼び出されません。`false` と等しければ、サポートは呼び出されます。
BiDiContextCP<property_Name>Skip	Boolean	EIS への固有接続の固有コネクター・プロパティーの特殊形式を定義します。

重要: CustomEventQuery、CustomUpdateQuery、および CustomDeleteQuery プロパティーのカスタム照会は、それらがストアード・プロシージャーまたはストアード関数の形式であれば、双方向変換用にサポートされます。しかし、標準 SQL の場合、SQL ステートメント内のストリング定数のみがサポートされます。

対話スペック・プロパティー

アダプターは、クライアント・インターフェースとして対話スペックを使用します。対話スペックは、オペレーション名を指定します。アダプターは、名前を抽出し、対応するオペレーションを入力ビジネス・オブジェクト・インスタンスに対して実行します。

表 34. Adapter for JDBC 用の対話スペック・プロパティー

プロパティー	型	説明	グローバル化	デフォルト値
ResultSetLimit	Integer	RetrieveAll 操作時に返す結果のセットの最大数。データベース内のヒット数が ResultSetLimit の値を超える場合、アダプターはエラー MatchesExceededLimitException を返します。アダプターは、メモリー不足の問題を回避するために、このプロパティーを使用します。	いいえ	100

インポート、エクスポートおよび WSDL ファイルの例

Service Component Architecture (SCA) のインポート・ファイルとエクスポート・ファイル、および Web サービス記述言語 (WSDL) ファイルは、エンタープライズ・サービス・ディスカバリーの処理中に生成される成果物です。

エクスポート・ファイルとインポート・ファイル、および対応する WSDL ファイルの例を以下に示します。

エクスポート (Inbound) ファイルの例

```
<?xml version="1.0" encoding="UTF-8"?>
<scdl:Export xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:_="http://JDBCEMD/inbound"
xmlns:eis="http://www.ibm.com/xmlns/prod/websphere/scdl/eis/6.0.0"
xmlns:scdl="http://www.ibm.com/xmlns/prod/websphere/scdl/6.0.0"
xmlns:wsdl="http://www.ibm.com/xmlns/prod/websphere/scdl/wsdl/6.0.0"
name="inbound/JDBCInboundInterface">
  <interfaces>
    <interface xsi:type="wsdl:WSDLPortType" portType="_:JDBCInboundInterface"/>
  </interfaces>
  <esbBinding xsi:type="eis:EISExportBinding"
dataBindingType="com.ibm.j2ca.extension.emd.runtime.WBIDataBindingImpl">
    <resourceAdapter name="JDBCEMDApp.IBM JDBC Adapter"
type="com.ibm.j2ca.jdbc.JDBCResourceAdapter">
      <properties/>
    </resourceAdapter>
    <connection type="com.ibm.j2ca.jdbc.inbound.JDBCActivationSpec"
selectorType="com.ibm.j2ca.extension.emd.runtime.WBIFunctionSelectorImpl">
      <properties>
        <BONamespace>http://www.ibm.com/xmlns/prod/websphere/j2ca/jdbc</BONamespace>
        <jdbcDriverClass>COM.ibm.db2.jdbc.app.DB2Driver</jdbcDriverClass>
        <databaseURL>jdbc:db2:somedb<databaseURL>
        <password>abcdefg</password>
        <userName>db2admin</userName>
      </properties>
    </connection>
    <methodBinding method="createDb2adminCustomer"
nativeMethod="emitCreateAfterImageDb2adminCustomer"/>
    <methodBinding method="updateDb2adminCustomer"
nativeMethod="emitUpdateAfterImageDb2adminCustomer"/>
    <methodBinding method="deleteDb2adminCustomer"
nativeMethod="emitDeleteAfterImageDb2adminCustomer"/>
  </esbBinding>
</scdl:Export>
```

インポート (Outbound) サービス記述の例

```
 <?xml version="1.0" encoding="UTF-8"?>
<scdl:Import xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:_="http://JDBCEMD/outbound"
xmlns:eis="http://www.ibm.com/xmlns/prod/websphere/scdl/eis/6.0.0"
xmlns:scdl="http://www.ibm.com/xmlns/prod/websphere/scdl/6.0.0"
xmlns:wsdl="http://www.ibm.com/xmlns/prod/websphere/scdl/wsdl/6.0.0"
name="outbound/JDBCOutboundInterface">
  <interfaces>
    <interface xsi:type="wsdl:WSDLPortType" portType="_:JDBCOutboundInterface"/>
  </interfaces>
  <esbBinding xsi:type="eis:EISImportBinding"
dataBindingType="com.ibm.j2ca.extension.emd.runtime.WBIDataBindingImpl">
    <resourceAdapter name="JDBCEMDApp.IBM JDBC Adapter"
type="com.ibm.j2ca.jdbc.JDBCResourceAdapter">
      <properties/>
    </resourceAdapter>
    <connection type="com.ibm.j2ca.jdbc.JDBCManagedConnectionFactory"
interactionType="com.ibm.j2ca.jdbc.JDBCInteractionSpec">
      <properties>
        <databaseURL>jdbc:db2:somedb</databaseURL>
        <jdbcDriverClass>COM.ibm.db2.jdbc.app.DB2Driver</jdbcDriverClass>
        <password>abcdefg</password>
        <userName>db2admin</userName>
      </properties>
    </connection>
    <methodBinding method="createDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Create</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="updateDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Update</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="deleteDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Delete</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="retrieveDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>Retrieve</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="retrieveallDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>RetrieveAll</functionName>
        </properties>
      </interaction>
    </methodBinding>
    <methodBinding method="applychangesDb2adminCustomer">
      <interaction>
        <properties>
          <functionName>ApplyChanges</functionName>
        </properties>
```

```
      </interaction>
    </methodBinding>
  </esbBinding>
</scdl:Import>
```

WSDL ファイルの例

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="CUSTOMER"
        targetNamespace="http://test/j2c/jdbc/customer"
        xmlns:tns="http://test/j2c/jdbc/customer"
        xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:datans="http://test/j2c/jdbc/customer">
        <wsdl:types>
                <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
                        <xsd:import namespace="http://test/j2c/jdbc/customer"
                                schemaLocation="CUSTOMER.xsd">
                        </xsd:import>
                </xsd:schema>
        </wsdl:types>
        <wsdl:message name="CUSTOMERRequest">
                <wsdl:part name="request" element="datans:CUSTOMER"></wsdl:part>
        </wsdl:message>
        <wsdl:portType name="Customer">
                <wsdl:operation name="updateCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
                <wsdl:operation name="createCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
                <wsdl:operation name="retrieveCustomer">
                        <wsdl:input message="tns:CustomerRequest"></wsdl:input>
                        <wsdl:output message="tns:CustomerRequest"></wsdl:output>
                </wsdl:operation>
        </wsdl:portType>
</wsdl:definitions>
```

WebSphere Integration Developer バージョン 6.0.1.1 以前への jar ファイルの追加

WebSphere Integration Developer バージョン 6.0.1.1 以前を使用している場合は、コネクター・プロジェクトのクラスパスに 3 つの jar ファイルを手動で追加する必要があります。

WebSphere Integration Developer のコネクター・プロジェクトに jar ファイルを追加するには、アダプターとそのすべての前提条件を事前にインストールしておく必要があります。

1. WebSphere Integration Developer を開きます。
2. J2EE パースペクティブで、コネクター・プロジェクトを右クリックし、「**プロパティー**」を選択します。
3. 「**Java ビルド・パス (Java Build Path)**」を選択し、「**外部 jar の追加 (Add External Jars)**」をクリックします。

4. WebSphere Process Server または Enterprise Server Bus の Install/lib フォルダーを選択し、 ffdcSupport.jar、aspectjrt.jar、および icu4j_3_2.jar を選択します。
5. 「**開く (Open)**」をクリックし、「**OK**」をクリックします。

メッセージ

IBM WebSphere Adapters が発行するメッセージは、 WebSphere Adapter バージョン 6.0.2 のインフォメーション・センターに文書化されています。

アダプター・メッセージを表示するには、『WebSphere Adapter メッセージ (WebSphere Adapters messages)』リンクにアクセスしてください。

関連製品情報

以下のリンク、インフォメーション・センター、Redbooks、および Web ページには、 IBM WebSphere Adapter for YOUR ADAPTER NAME に関する情報が含まれています。

必要な場合のあるその他の情報

表 35. 必要な場合のある WebSphere Adapters の情報

情報	その検索方法
ビジネス・オブジェクト・エディターを使用してビジネス・オブジェクトを編集する方法	WebSphere Integration Developer に関する文書を含む IBM WebSphere Business Process Management インフォメーション・センターで、トピック「ビジネス・オブジェクトの編集 (Editing Business Objects)」を検索します。
デプロイしたアダプターをアンインストールする方法	WebSphere Application Server のライブラリー・ページで、ご使用のバージョンの WebSphere Application Server のインフォメーション・センターを開き、トピック「アプリケーションのアンインストール (Uninstalling applications)」を検索します。

関連製品の情報

- WebSphere Adapters バージョン 6.0
- WebSphere Business Integration Adapters
- WebSphere Integration Developer
- WebSphere Process Server
- WebSphere Enterprise Service Bus
- WebSphere Application Server

Redbooks

- WebSphere Adapter Development Redbook
- WebSphere Redbooks ドメイン (WebSphere Redbooks domain)

developerWorks® リソース

- WebSphere Adapter Toolkit
- WebSphere Business Integration ゾーン (WebSphere business integration zone)

サポートおよび支援

- WebSphere Adapters 製品サポート (WebSphere Adapters product support)
- WebSphere Adapters テクニカル・ノート (WebSphere Adapters technotes) - 「**追加の検索用語 (Additional search terms)**」フィールドにアダプター名を指定し、「**Go (Go)**」をクリックします。

第 14 章 用語集

IBM WebSphere Adapters の用語集は、 WebSphere Adapter バージョン 6.0.2 のインフォメーション・センターに含まれています。

これを表示するには、『WebSphere Adapter の用語集 (WebSphere Adapters glossary)』リンクにアクセスしてください。

© Copyright IBM Corp. 2005, 2006

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

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

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

〒106-8711
東京都港区六本木 3-2-12
IBM World Trade Asia Corporation
Intellectual Property Law & Licensing

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

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

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

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

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

© Copyright IBM Corp. 2005, 2006

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

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

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

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

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

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

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

著作権使用許諾:

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

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _年を入れる_. All rights reserved.

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

プログラミング・インターフェース情報

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

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

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

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

商標

IBM および関連の商標については、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

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

索引

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

[ア行]

[カ行]

[サ行]

© Copyright IBM Corp. 2005, 2006

[タ行]

[ナ行]

[ハ行]

［マ行］

［ヤ行］

［ラ行］

A

C

D

E

F

I

J

O

P

Q

R

S

U

W

X

Z

Printed in Japan

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