IBM WebSphere InterChange Server

テクニカル入門 (IBM WebSphere InterChange Server)

V4.3.0

IBM WebSphere InterChange Server

テクニカル入門 (IBM WebSphere InterChange Server)

V4.3.0

本書は、*IBM WebSphere InterChange Server (5724-178)* バージョン *4.3.0* および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

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

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

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

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

(URL は、変更になる場合があります)

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

原　典： IBM WebSphere InterChange Server
Technical Introduction to IBM WebSphere InterChange Server
V4.3.0

発　行： 日本アイ・ビー・エム株式会社

担　当： ナショナル・ランゲージ・サポート

第1刷 2004.10

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

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

目次

本書について

IBM(R) WebSphere(R) InterChange Server およびそれに関連する Toolset は、IBM WebSphere Business Integration Adapters とともに使用され、主要な e-business テクノロジーおよびエンタープライズ・アプリケーションとのビジネス・プロセス統合および接続性をもたらします。

本書では、InterChange Server を WebSphere Business Integration システムの統合ブローカーとして実装したシステムのインフラストラクチャーとその他の主要な要素を紹介します。

対象読者

本書は、IBM のコンサルタントおよびお客様を対象としています。

本書の前提条件

本書は入門書であり、InterChange Server 資料セット内の資料を使用することが前提条件となります。

関連資料

この製品に付属の資料の完全セットは、すべての WebSphere InterChange Server インストールに共通の機能およびコンポーネントを説明するとともに、特定のコンポーネントに関する参照資料についても記載しています。

資料は、以下のサイトからインストールすることができます。

- InterChange Server の資料:

  `http://www.ibm.com/websphere/integration/wicserver/infocenter`
- コラボレーションの資料:

  `http://www.ibm.com/websphere/integration/wbicollaborations/infocenter`
- WebSphere Business Integration Adapters の資料:

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

上記のサイトには、資料のダウンロード、インストール、および表示に関する簡単な説明があります。

表記上の規則

本書は下記の規則に従って編集されています。

`Courier フォント`	コマンド名、ファイル名、入力情報、システムが画面に出力した情報など、記述されたとおりの値を示します。
太字	初出語を示します。
イタリック、*イタリック*	変数名または相互参照を示します。

青い文字	オンラインで表示したときにのみ見られる青の部分は、相互参照用のハイパーリンクです。青い文字列をクリックすることにより、参照先オブジェクトに飛ぶことができます。
{ }	構文の記述行の場合、中括弧 {} で囲まれた部分は、選択対象のオプションです。1 つのオプションのみを選択する必要があります。
[ ]	構文の記述行の場合、この括弧 [] で囲まれた部分は、オプションのパラメーターです。
...	構文の記述行の場合、省略符号 ... は直前のパラメーターが繰り返されることを示します。例えば、`option[,...]` は、複数のオプションをコンマで区切って入力できることを示します。
< >	命名規則においては、互いの名前を区別するために、個々の名前要素を不等号括弧で囲みます (例: `<server_name><connector_name>tmp.log`)。
/, ¥	本書では、ディレクトリー・パスに円記号 (¥) を使用します。UNIX システムの場合、円記号はスラッシュ (/) に置き換えてください。本書のすべての IBM WebSphere 製品のパス名は、使用システムで IBM WebSphere 製品がインストールされたディレクトリーを基準とした相対パス名です。
%text% および *$text*	% 記号で囲まれたテキストは、Windows text システム変数またはユーザー変数の値を示します。 UNIX 環境でこれに相当する表記は *$text* となります。これは、UNIX 環境変数 *text* の値を示しています。
ProductDir	製品のインストール先ディレクトリーを表します。IBM WebSphere InterChange Server 環境では、デフォルトの製品ディレクトリーは IBM¥WebSphereICS です。IBM WebSphere Business Integration Adapters 環境では、デフォルトの製品ディレクトリーは WebSphereAdapters です。

リリース新機能

この章では、本書で取り上げる IBM WebSphere Business Integration システムの新機能について説明します。

WebSphere InterChange Server バージョン 4.3 の新機能

IBM WebSphere InterChange Server リリース 4.3 には、以下の機能があります。

- エンドツーエンド・プライバシー。これは、発信元のノードから送信されたメッセージが中間ノードを経由して宛先ノードに到達するまでの間、メッセージを保護します。エンドツーエンド・プライバシーは、InterChange Server に浸透性および非対称の動的セキュリティーを提供します。
- 役割ベースのアクセス制御。これは、InterChange Server へのアクセスを求めるユーザーを認証した上で、ユーザーに割り当てられた役割に基づいて、個々のコンポーネントへのユーザーのアクセスを制限します。
- 動的サービス呼び出し。これは、コラボレーション開発時に事前定義されなかった、明示的にバインドされていない宛先に対するサービス呼び出しをコラボレーションが行えるようにします。
- IBM Tivoli License Manager (ITLM) のサポート
- 言語固有の振る舞いのサポート

WebSphere InterChange Server バージョン 4.2.2 の新機能

IBM WebSphere InterChange Server リリース 4.2.2 には、以下の機能があります。

- InterChange Server (ICS) およびその他の ICS コンポーネントは、現在では、サード・パーティーの VisiBroker ORB ではなく、IBM Java Object Request Broker (ORB) を使用します。

WebSphere InterChange Server バージョン 4.2.1 の新機能

IBM WebSphere InterChange Server リリース 4.2.1 では、本書は新規 System Manager 機能の概要について説明しています。

第 1 章 IBM WebSphere InterChange Server の概要

本書では、IBM WebSphere InterChange Server を IBM WebSphere Business Integration システムの統合ブローカーとして使用するための概要について説明します。WebSphere Business Integration システムに実装された InterChange Server は、アプリケーション間の問題を解決するための分散インフラストラクチャーを提供します。これには、次の機能が含まれます。

- さまざまなソース間でビジネス情報をやり取りして、インターネットを介してビジネス取引を実行できます。
- エンタープライズ環境において、さまざまなアプリケーションを使用してビジネス情報を処理したり発送したりできます。

この章では、InterChange Server を統合ブローカーとして使用する統合システムのアーキテクチャー、コンポーネント、および処理フローの概要について説明します。この章は次のセクションから構成されます。

注: 本書で示される図は、単なる例であり、構造や概念を表すために使用されています。図は、必ずしも実際のコンポーネントを表すものではありません。

InterChange Server および WebSphere Business Integration システム

IBM WebSphere InterChange Server およびそれに関連する Toolset は、**WebSphere Business Integration システム**で IBM(R) WebSphere(R) Business Integration Adapters とともに使用されます。 WebSphere Business Integration システムは、大きく分けて統合ブローカーとアダプター・セットで構成されています。これらを使用することにより、さまざまなビジネス・アプリケーションが、情報の整合された転送を介してビジネス・オブジェクトの形式でデータ交換を行えるようになります。

InterChange Server 以外にも、WebSphere Business Integration システムの中枢として使用可能な統合ブローカーがあります。他には、WebSphere MQ Integrator

Broker、WebSphere Business Integration Message Broker、および WebSphere Application Server があります。これらの統合ブローカーはいずれも WebSphere Business Integration Adapters を使用でき、このアダプターは、アプリケーション接続のためにコネクターを活用し、アプリケーション間で交換されるビジネス・データのコンテナーとしてビジネス・オブジェクトを使用します。ただし、これら 2 つのブローカーを使用する統合システム間でいくつか相違点があります。したがってニーズに合ったブローカーを選択することがこの時点では重要です。

本書では、InterChange Server を統合ブローカーとして実装した統合システムの概念について重点的に説明します。WebSphere Message Brokers (WebSphere MQ Integrator、WebSphere MQ Integrator Broker、または WebSphere Business Integration Message Broker) のいずれかを統合ブローカーとして統合システムに実装する方法については、「*WebSphere Message Brokers 使用アダプター・インプリメンテーション・ガイド*」を参照してください。WebSphere Application Server を統合ブローカーとして統合システムに実装する方法については、「*アダプター実装ガイド (WebSphere Application Server)*」を参照してください。これらのインプリメンテーション・ガイドは両方とも WebSphere Business Integration Adapters 資料セットの一部です。

InterChange Server モデル

InterChange Server を実装した WebSphere Business Integration システムは、モジュラー・コンポーネントおよびアプリケーションに依存しないビジネス・ロジックを使用します。この方法では、InterChange Server は、分散されたハブ・アンド・スポーク方式のインフラストラクチャーを使用します。これにより、インターネット上、ローカル・ネットワークのアプリケーション上、あるいはその両方に分散されたさまざまなビジネス・プロセス (例えば、e-business 注文の配送、返品処理、在庫管理) を実行することができます。このシステムは分散型で柔軟性があり、サイト固有およびアプリケーション固有の要求に合わせることができる再使用可能なコンポーネントおよびカスタマイズ機能を備えています。

InterChange Server 環境では、WebSphere Business Integration Adapters 製品で提供されるコネクターを、複数のタイプのモジュラーおよびカスタマイズ可能なコンポーネント (コラボレーション、ビジネス・オブジェクト、マップ、データ・ハンドラーなど) とともに使用します。次の各章では、InterChange Server をブローカーとして実装した WebSphere Business Integration システムでこれらのコンポーネントがどのように使用されるかについて説明します。

IBM WebSphere InterChange Server は、IBM Tivoli License Manager とともに使用できるように対応しています。 IBM WebSphere InterChange Server は、サーバー・コンポーネントとツール・セットにライセンス管理機能およびインベントリー機能を提供します。(詳しくは、「*アクセス開発ガイド (Enterprise Java Beans)*」および「*アクセス開発ガイド (J2EE Connector Architecture)*」を参照してください。

WebSphere Business InterChange Server Toolset

InterChange Server では、WebSphere Business Integration Toolset を使用できます。これにより、以下の操作が可能になります。

- コンポーネントの開発およびカスタマイズ
- 次の開発および実動段階で配置するコンポーネントのプロジェクトへの編成
- 配置した InterChange Server 統合システムのモニター

これらについては、45 ページの『第 2 章 InterChange Server で使用するツール』で説明します。

コラボレーション、ビジネス・オブジェクト、および接続性

注: 本書において「アプリケーション」という用語は、統合されたエンタープライズ・ソフトウェア製品のことであり、IBM WebSphere InterChange Server 製品のコンポーネントのことではありません。

IBM WebSphere InterChange Server システムは、中央インフラストラクチャー (InterChange Server)、およびハブ・アンド・スポーク設計のモジュラー・コンポーネントを以下のように使用します。

- ビジネス・プロセス・ロジックは、ハブの InterChange Server **Collaborations** 内にあります。

  InterChange Server Collaborations は、分散ビジネス・プロセスを記述するロジックを含むソフトウェア・モジュールです。基本的なビジネス・プロセスが異なると、そのコラボレーションも異なります。例えば、ContactManager コラボレーションや InventoryMovement コラボレーションなどがあります。コラボレーションは各種のビジネス・プロセスの機能を調整し、ビジネス・プロセス間でデータ交換できるようにします。コラボレーションはハブであり、データはそれらを介してビジネス・オブジェクトの形式で交換されます。
- データは、**ビジネス・オブジェクト**の形式でハブとスポークの間で交換されます。

  ビジネス・オブジェクトは、IBM WebSphere InterChange Server システムで使用されるデータ交換用のメッセージです。**データ・ハンドラー**は、シリアル・アプリケーション・データをビジネス・オブジェクトに変換するために使用されます。**マップ**は、特定のアプリケーションのデータ・モデルのために構成されるビジネス・オブジェクトと、ハブのコラボレーションが使用するために汎用的に構成されるビジネス・オブジェクトの間で使用されます。
- アプリケーションおよびテクノロジー・**コネクター**は、IBM WebSphere Integration Adapters 製品で使用可能であり、スポークのアプリケーション (または Web サーバーや他のプログラマチック・エンティティー) に接続性を提供します。

  コネクターは、ネットワーク内部で、またはファイアウォールを越えてインターネット上で機能するように構成できます。

コネクターは、それぞれ**コネクター・コントローラー**と**コネクター・エージェント**という 2 つのパーツで構成されます。コネクター・コントローラーは WebSphere InterChange Server Collaborations オブジェクトと直接対話し、IBM WebSphere InterChange Server システムを実装したサーバー (ハブ・アンド・スポーク関係におけるハブ) に置かれます。コネクター・エージェントはアプリケーションと直接対話し、どのサーバーのアプリケーションにも置くことができます。リモート・エージェント・テクノロジーは、ハブ・サイトのコネクター・コントローラーと、別のサイトに置かれているエージェントとの間のインターネットを介した通信を実装するために使用されます。

コネクターの中には、特定のアプリケーションと対話するように設計されているものもあります。アプリケーション・コネクター (例えば Clarify コネクター) は、コラボレーションとアプリケーションを媒介します。これらのコネクターは、アプリケーションからのデータをコラボレーションで扱えるビジネス・オブジェクトに変換し、コラボレーションからのビジネス・オブジェクトを特定のアプリケーションが受信できるデータに変換します。

その他のコネクターは、特定のテクノロジー規格に準拠する対話用に設計されます。(例えば、XML コネクターは、InterChange Server Collaborations から Web サーバーにデータを送信するために使用されます。このコネクターは、Web サーバーがコネクター・エージェントまたはその他の IBM WebSphere ソフトウェアを実行していないネットワークのファイアウォールの外部に置かれている場合でも有効です。)

- **サーバー・アクセス・インターフェース**によって、WebSphere InterChange Server を実装していないリモートのスポーク・サイトは、**アクセス・クライアント**を使用して、InterChange Server を持つハブ・サイトへの呼び出しをインターネット経由で実行できるようになります。

  サーバー・アクセス・インターフェースは InterChange Server の一部です。サーバー・アクセス・インターフェースは CORBA 準拠の API であり、内部ネットワークまたは外部ソースからの同時データ転送を受け入れます。受け入れられたデータは、コラボレーションで扱えるビジネス・オブジェクトに変換されます。サーバー・アクセス・インターフェースは外部エンティティー (リモート顧客サイトの Web ブラウザーなど) からの呼び出しの受信を可能にします。これらの呼び出しはコネクター・エージェントを介してではなく、Web サーブレットを介してサーバー・アクセス・インターフェースで受信されます。

  サーバー・アクセス・インターフェースとコネクターは、どちらもデータ・ハンドラーを使用します。InterChange Server 環境では、**データ・ハンドラー・フレームワーク**と呼ばれる基底クラスのモジュラー・グループから新しいデータ・ハンドラーを作成できます。InterChange Server ソリューションには、**プロトコル・ハンドラー・フレームワーク**も含まれます。これらのフレームワークを使用すると、ソリューションをカスタマイズしたり、将来新たなデータ・フォーマットやプロトコルを追加するのが容易になります。

実装ソリューションのサンプル

一般的な InterChange Server ソリューションには、1 つ以上のコラボレーションと、企業に関連するビジネス情報を表すビジネス・オブジェクトのセットが組み込

まれます。コラボレーションとビジネス・オブジェクトは、コネクター、サーバー・アクセス・インターフェース、またはその両方とともに使用します。

コネクターは 1 つ以上のコラボレーションと対話でき、その結果さまざまなビジネス・プロセスを実行できます。また、各コラボレーションは任意の数のコネクター、つまり任意の数のアプリケーションと対話できます。

例えば、顧客情報が顧客対話管理 (CIM) アプリケーションで変更されたときにエンタープライズ・リソース・プランニング (ERP) アプリケーションを自動的に更新するための InterChange Server ソリューションは、CustomerSync コラボレーション、CIM アプリケーションと ERP アプリケーションのコネクター、および顧客情報を表すビジネス・オブジェクトの定義から構成されます。図 1 に、このソリューションを示します。

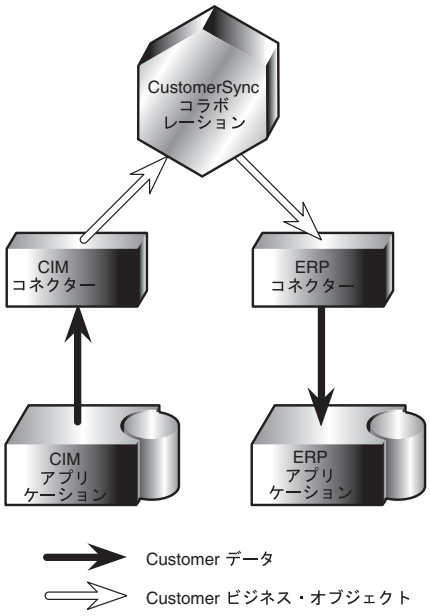

図 1. CIM から ERP Customer データ・ソリューション

図 1 のソリューションは、ネットワーク上でローカルに実装することができます。また、インターネットを介して実装することもできます。

図 2 では、IBM WebSphere InterChange Server システムもコネクターも実装していないサイトで、顧客担当者が Web ブラウザーを使用して、WebSphere InterChange Server システムを使用するサイトにある ERP アプリケーション (SAP など) からインターネット経由の購入注文状況を把握する必要がある場合があります。このために、InterChange Server ソリューションは (この例の仮想の) 購入注文ビジネス・ロジック用のコラボレーション、SAP コネクター、購入注文状況情報を表すビジネス・オブジェクトの定義とともに、サーバー・アクセス・インターフェースを使用します。

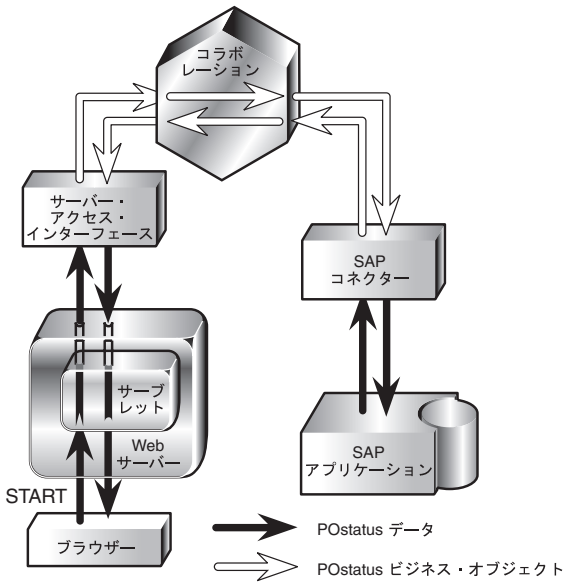

図 2. サーバー・アクセス・インターフェースを介する呼び出しの実行

複数のサーバーの配置

ハブ・アンド・スポーク環境では、単一のアプリケーションが複数の InterChange Server と対話できます。複数の InterChange Server は、共同して動作できるように展開できます。コネクター・エージェントは、分割したりアプリケーションからさまざまなサーバーにイベントを送信したりするように構成できます。このように構成することにより、複数の InterChange Server と対話するアプリケーションの処理負荷のバランスを取ることができます。さらに複数の InterChange Servers は、コネクターおよびサーバー・アクセス・インターフェースを介して互いに対話するように構成できます。

インターネット経由の接続

コネクターは、インターネットを経由したデータ交換をさまざまな方法で可能にします。次のような方法があります。

- リモート側に実装されるアプリケーション・コネクター・エージェント

  この方法 (リモート・エージェント・テクノロジーと呼ばれる) では、JMS をネイティブ WebSphere MQ の SSL か WebSphere MQ Internet Pass-Thru (MQIPT) のどちらか一方とともに使用して、インターネットを介してコネクターを実装します。IBM WebSphere InterChange Server システムを実装した単一のサイトはハブとして機能し、リモート・コネクター・エージェントを実装したスポーク・サイトとインターネット経由でデータ交換を行います。

- サーバー・アクセス・インターフェースとともに使用されるテクノロジー・コネクター

  この方法では、IBM WebSphere InterChange Server システムに同期呼び出しを渡すためにサーバー・アクセス・インターフェースが使用され、IBM WebSphere

InterChange Server システムからのデータを送信するためにインターネット・テクノロジー規格を使用するコネクターが使用されます。

– 外部ブラウザー (またはインターネット上の他のソース) からの呼び出しは Web サーブレットで受信され、Web サーブレットはサーバー・アクセス・インターフェースを介して呼び出しを送信し、コラボレーションのビジネス・プロセスを起動します。
– コラボレーションの代わりに、特定のインターネット・テクノロジー規格 (XML など) 用のコネクターが、特定のデータ・フォーマットを理解する外部宛先との要求/応答対話を行います。

サーバー・アクセス・インターフェースは、InterChange Server (ICS) のハブ・サイトに置かれます。サーバー・アクセス・インターフェースが呼び出しを受信すると、特定のデータ・フォーマットのハンドラー (XML データ・ハンドラーなど) にデータを送信します。データ・ハンドラーはデータを汎用ビジネス・オブジェクトに変換します。サーバー・アクセス・インターフェースはそのビジネス・オブジェクトをコラボレーションに送信します。コラボレーションはビジネス・オブジェクトに処理を施して応答し、その応答がプロセスの最初に使用された特定のデータ・フォーマットに変換されます。

外部プロセスからの呼び出しを受信して、その呼び出しをビジネス・オブジェクトとしてコラボレーションに送信するために、サーバー・アクセス・インターフェースには次のものが必要です。

– サーブレットまたはその他の処理を使用してデータをサーバー・アクセス・インターフェースに送信する Web サーバー。データは、InterChange Server でデータ・ハンドラーが構成されている MIME タイプである必要があります。サーブレットはサーバー・アクセス・インターフェースを使用して、ビジネス・オブジェクト形式にデータを変換するデータ・ハンドラーを呼び出します。次にサーバー・アクセス・インターフェースを使用して、ビジネス・オブジェクトを呼び出しとしてコラボレーションに送信します。
– サーバー・アクセス・インターフェースを介して、呼び出しとして受信される要求を扱えるように構成されたコラボレーション。

この方法の例は、図 3 に示されています。

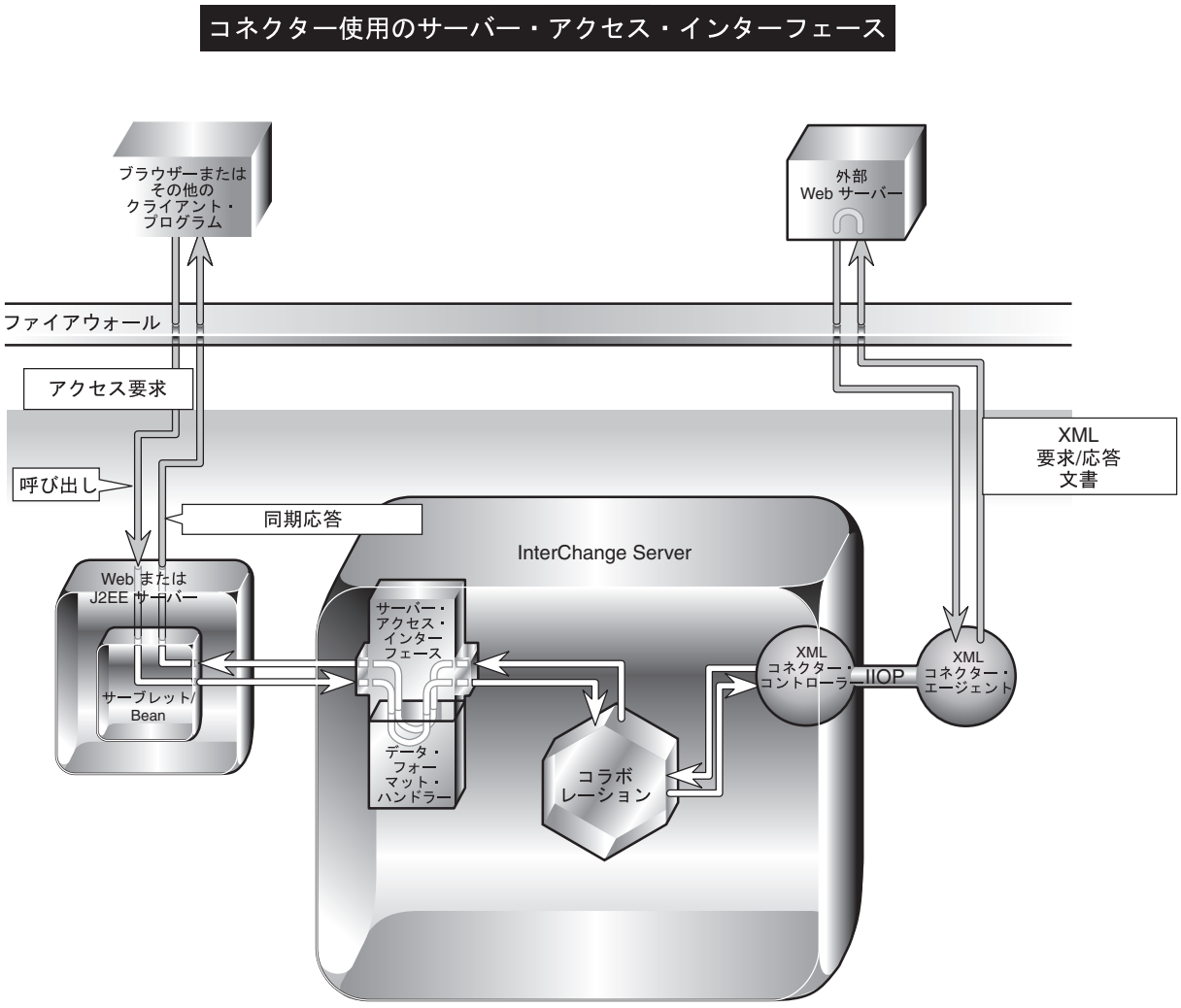

図 3. サーバー・アクセス・インターフェースを使用したインターネット・データ交換

- インターネット交換用に設計されたテクノロジー・コネクター

  コネクターの中には、リモート・エージェント構成を使用せずに、パブリッシュ・アンド・サブスクライブ対話と要求/応答対話によってインターネット経由のデータ交換を行えるものもあります。例えば TPI コネクターは、リモート取引先との XML、EDI、またはバイナリー・フォーマットのインターネットを介したデータ交換を実現します。同様に、Email コネクターは SMTP プロトコルを使用して、インターネットを介したデータ変換を行います。

  この方法の例は、図 4 に示されています。

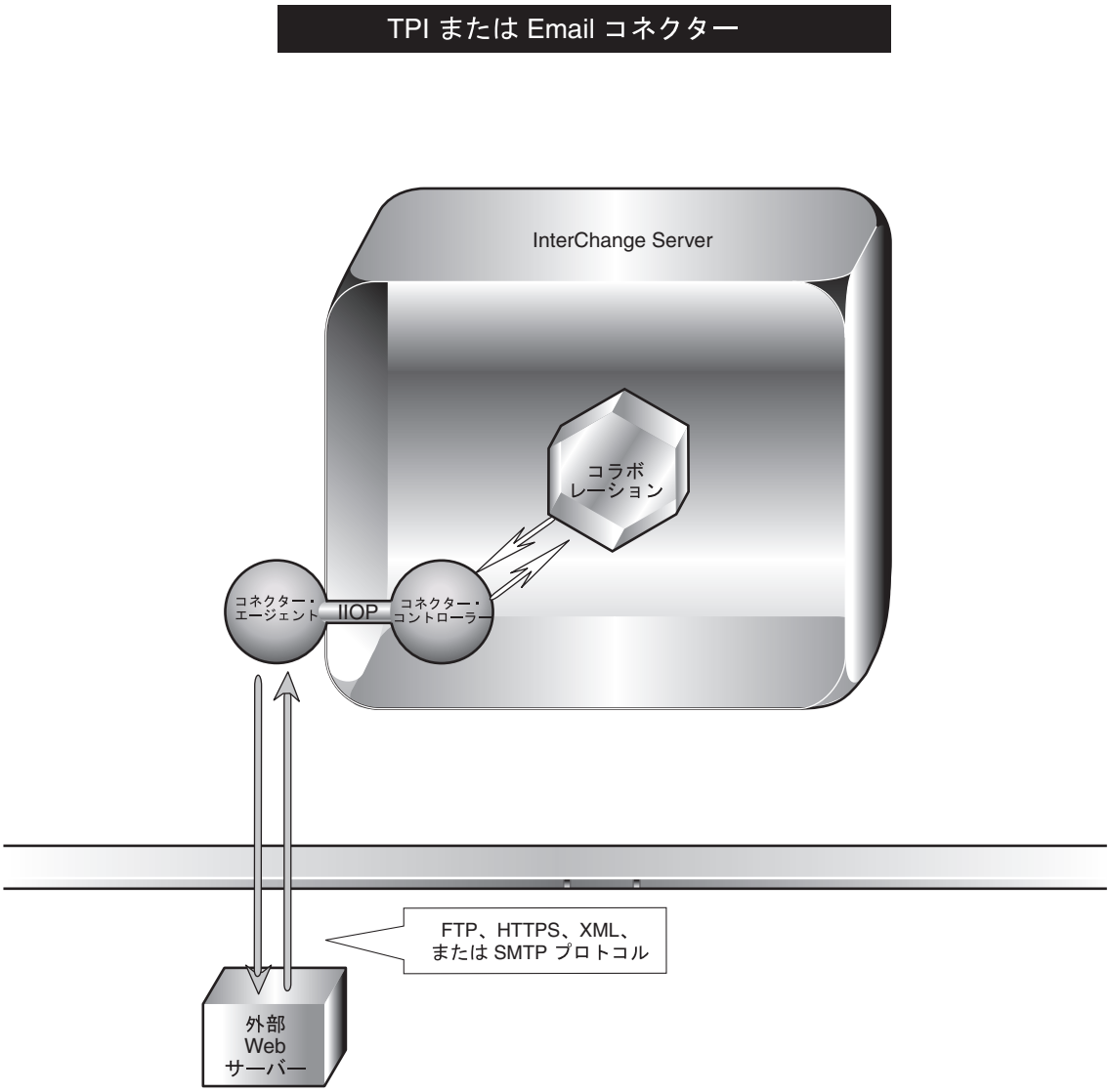

図 4. TPI または Email コネクターを使用したインターネット・データ交換

InterChange Server 環境におけるデータ・フロー

InterChange Server 環境において、データ・フロー (1 つのアプリケーションまたはエンティティーから別のアプリケーションまたはエンティティーへのデータの移動および処理) には次のすべてが含まれます。

- ローカル・ネットワーク上のさまざまなアプリケーション間での非同期交換または同期交換
- インターネット上のさまざまなアプリケーション間での非同期交換または同期交換
- ローカル・ネットワークおよび外部 Web サーバー上のアプリケーション間での非同期交換および同期交換
- 以上のすべてを含む包括的なビジネス・プロセス

データ・フローは、パブリッシュ・アンド・サブスクライブ対話またはサービス呼び出し対話のどちらかの対話によって開始されます。どちらの対話も、コラボレーションのビジネス・プロセスの実行を開始するトリガーを提供します。次に、コラボレーションは要求/応答対話という 3 つ目の対話を使用して、宛先とのデータ交換を終了します。

パブリッシュ・アンド・サブスクライブ対話

コネクターとコラボレーションは、**パブリッシュ・アンド・サブスクライブ**対話を使用して、アプリケーション・イベントに関する情報を処理するために IBM WebSphere InterChange Server に移動させます。

パブリッシュ・アンド・サブスクライブ対話では、コラボレーションがアプリケーション動作を表す特定タイプの**トリガー・イベント**のためのビジネス・オブジェクト (例えば Employee.Create) を受信して起動されると、ビジネス・プロセスを開始します。ビジネス・オブジェクトの名前 (*Employee*) は、ビジネス・エンティティーのタイプを表します。動詞 (*Create*) はそのエンティティーに対して実行された操作を表します。つまり、Employee.Create イベントは従業員エンティティーの作成を報告します。

パブリッシュ・アンド・サブスクライブ対話によって、トリガー・イベントは次のようにしてコラボレーションに到達します。

- コラボレーションはそれ自体の実行を起動するイベントに**サブスクライブ**します。コラボレーションはイベントを要求することによってそのイベントにサブスクライブし、待機します。Employee.Create イベントのビジネス・オブジェクトを受信すると、Employee.Create にサブスクライブしているコラボレーションが実行を開始します。
- アプリケーション内でイベントが発生し、そのイベントがアプリケーション・コネクターの**イベント通知**機構によって検出されます。コネクターはイベントをパブリッシュして 1 つ以上のコラボレーションに提供します。これにより、イベントはビジネス・オブジェクトとして使用可能になります。

イベントは、使用するコネクターによって、非同期または同期でコラボレーションにパブリッシュされます。また、コラボレーションの長期存続ビジネス・プロセス機能を使用可能にしているときは、コラボレーションは着信イベントが定義済みマッチング基準を満たしている場合にそのイベントを待ち状態で保持できます。

アクセス要求

コラボレーションは、直接呼び出しが**アクセス・クライアント**によって送信され、サーバー・アクセス・インターフェースで受信されて、そのコラボレーションにビ

ジネス・オブジェクトとして送信されると起動するように設計できます。InterChange Server 環境では、サーバー・アクセス・インターフェース経由でコラボレーションに送信された呼び出しは**アクセス要求**と呼ばれます。アクセス要求は、外部ソースまたは InterChange Server 環境内に構成されたソースから発信されます。

アクセス要求対話は、同期通信が重要な場合に役立ちます。例えば、顧客担当者が Web ブラウザーを使用してインターネット経由で在庫状況情報を要求するような場合です。

要求/応答対話

コラボレーションは、トリガー・ビジネス・オブジェクトを受信して起動されるとデータの処理を開始します。トリガー・ビジネス・オブジェクトは、アクセス要求またはイベント通知のいずれかの結果で生成されます。コラボレーションが起動されると、バインドされたコネクターに要求を送信したり応答を受信したりできます。

コラボレーションは、それ自体の要求 (**サービス呼び出し要求**と呼ばれます) を汎用ビジネス・オブジェクトの形式で作成します。コネクターは、汎用ビジネス・オブジェクトを特定のアプリケーションで理解されるデータ・エンティティーまたはコネクターが設計されたものと同じデータ・フォーマットに変換します。

コラボレーションがコネクターから受信する応答 (**サービス呼び出し応答**と呼ばれます) は、ビジネス・データ (検索要求の場合) または状況報告 (成功または不成功) を含んだビジネス・オブジェクトです。

図 5 は、パブリッシュ・アンド・サブスクライブ対話によって開始され、要求/応答対話によって完了するビジネス・プロセスを簡単に示しています。(この簡易図では、コネクター・コントローラーとコネクター・エージェントが単体として表されています。また、ローカル・ネットワーク上のアプリケーション用コネクターとインターネット・ファイアウォールの外部にあるアプリケーション用コネクターを区別していません。) この例は、顧客サービス担当者が 1 つのケースの作業を終えると自動的に送り状が生成される、分散ビジネス・プロセスを示しています。この例で、仮想の Service Billing コラボレーションは、コネクターを使用してビジネス・データを 3 つの異なるアプリケーションに変換しています。

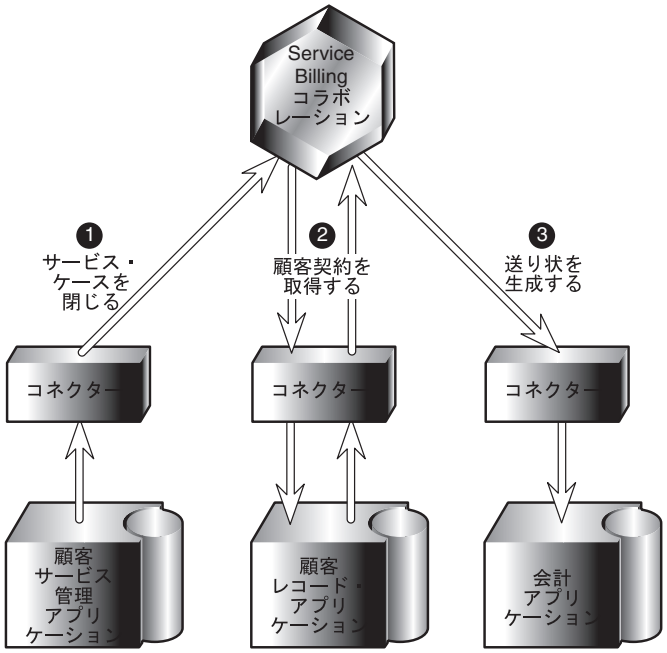

図 5. パブリッシュ・アンド・サブスクライブ対話、要求/応答対話

図 5 には、次のビジネス・プロセスの順序を示します。

1. 顧客サービス担当者が 1 つのケースの作業を終了します。コネクターは顧客サービス管理アプリケーションからケースの終了をイベントとして検出し、関連するケース・データを取り出します。次に、コネクターはイベントをパブリッシュして、そのイベントにサブスクライブしているコラボレーションが使用できるようにします。この一連のアクションは、パブリッシュ・アンド・サブスクライブ対話を構成します。
2. 送り状の金額を計算するには、コラボレーションに顧客契約条件が必要です。コラボレーションはサービス呼び出し要求を送信して、顧客レコード・アプリケーション用のコネクターから必要なデータを検索します。コネクターが要求に応答します。この一連のアクションは、コラボレーションと、顧客レコード・アプリケーションおよびコネクター間の要求/応答対話を構成します。
3. ケース情報と顧客契約の両方を使用して、コラボレーションは送り状の作成に必要な情報を生産します。コラボレーションが送り状作成要求を会計アプリケーション用のコネクターに送信すると、コネクターは要求をアプリケーションに転送し、コラボレーションに成功または失敗通知を送信します。この一連のアクションは、コラボレーションと、会計アプリケーションおよびコネクター間の要求/応答対話を構成します。

サイトでは、コラボレーションとコネクターが結合する度合いを調整できます。例えば、コラボレーションが 24 時間体制で実行している場合に、アプリケーションと通信するコネクターに要求を送信する時間を午前 12 時から午前 2 時の間に限定できます。コラボレーションは、応答を必要とすることなく要求を送信し、応答を受信すると単にその応答を処理するように設計および構成できます。

また、長期存続ビジネス・プロセスでコラボレーションを使用可能にしているときは、コラボレーションは、タイムアウト値 (応答による保管された処理フローの再開が可能な期間) とともに要求のフロー・コンテキストを保存したり、要求を送信したりできます。

サンプル・データ・フロー

図 6 は、アプリケーション間 (Web サーバーおよびブラウザーなど、他のエンティティー) でデータをやり取りするための一般的なアクセス要求対話、パブリッシュ・アンド・サブスクライブ対話、および要求/応答対話で使用される、IBM WebSphere InterChange Server システム・コンポーネントのハイレベル表示を示しています。

データはハブにある InterChange Server を通るフロー内を移動し、ローカル・ネットワーク上のアプリケーション、インターネット・ファイアウォールの外側のコネクター・エージェントとともに構成されたアプリケーション、および Web サーバーやブラウザーなどの外部エンティティーと交換されます。

フローは、アクセス要求やイベント、またはあらゆるタイプの外部および内部プロセスによって開始されます。フローが開始されると、コラボレーションは要求/応答対話を使用して、ローカルやリモートのアプリケーションまたは他のエンティティーとのビジネス・プロセスを完了します。

このダイアグラムで、1 つのコネクターはアプリケーション・イベントをパブリッシュし、もう 1 つのコネクターはコラボレーションと宛先 Web サーバー間の要求/応答対話を行っている状態が示されています。この図は、可能な構成の一例にすぎません。ほとんどのコネクターは、イベントのパブリッシュとコラボレーション要求への応答の両方を実行するように構成できます。

このダイアグラムには示されていませんが、インターネット上でデータを交換するために、コネクターがローカルに構成されたテクノロジー・サーバー (TPI サーバーや E メール・サーバーなど) と対話する構成も可能です。

また、コネクター・エージェントとそれに関連するアプリケーションはローカルに設置する必要がありません。リモート・エージェント・テクノロジーを使用することによって、リモート URL ロケーションにあるコネクター・エージェントは、イベントのパブリッシュとコラボレーション要求への応答の両方を実行できます。

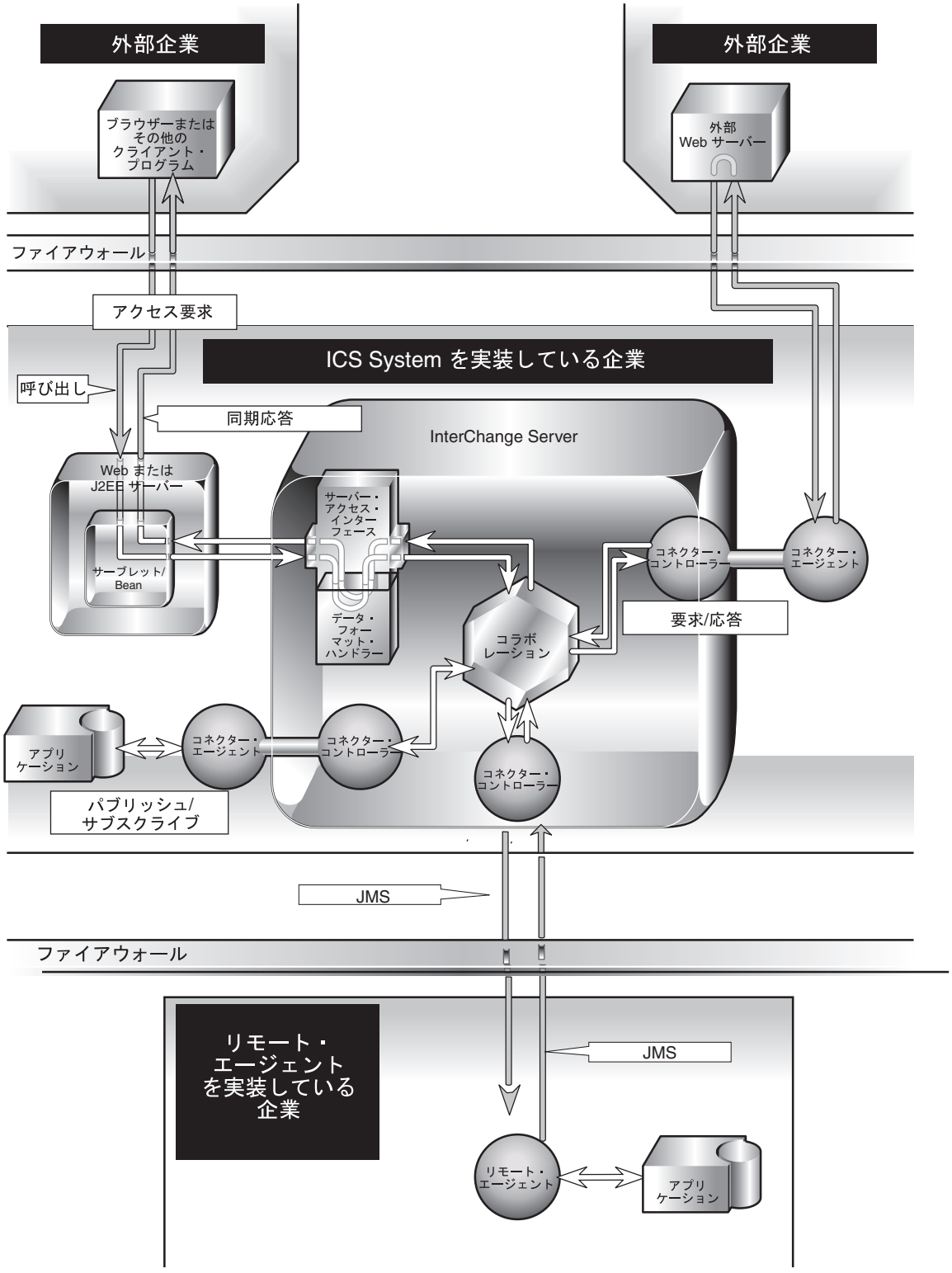

図 6. ビジネス・データ・フロー

図 7 は、可能ないくつかのデータ・パス上で交換されるデータのタイプを示しています。

数字は、データ・パスの特定の順序を 1 つのみ示しています。この例では、外部 Web ブラウザーからサーバー・アクセス・インターフェースを経由してコラボレーションに到達し、コラボレーションのビジネス・プロセスの完了後、外部 Web サーバーに送信されます。

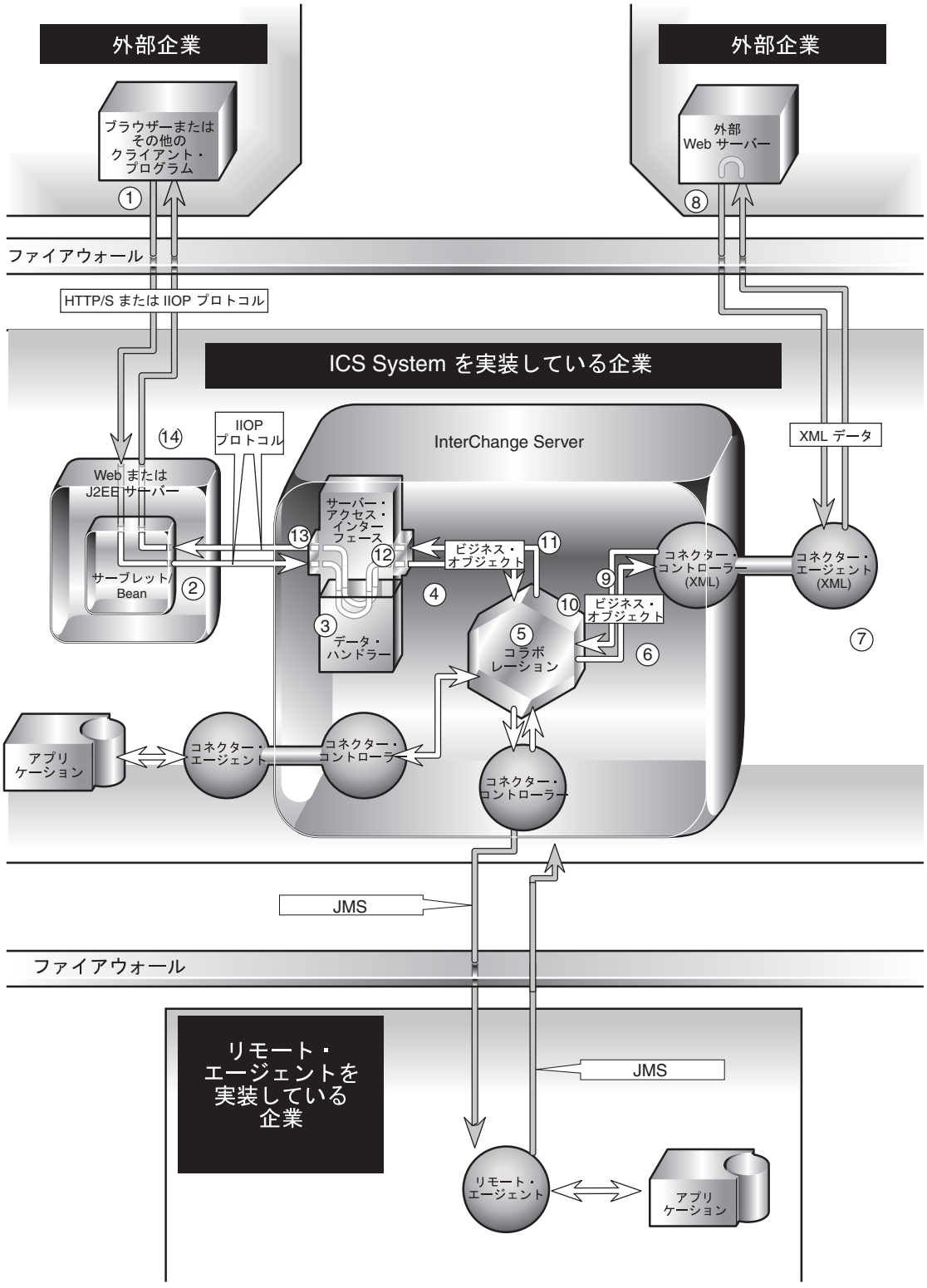

図 7. ビジネス・データ・フローの例

図 7 のサンプルでは、次の処理が行われます。

1. クライアント・プログラムがサーバーと通信し、アクセス要求をコラボレーションに送信します。クライアント・プログラムは、HTTP/HTTPS プロトコルを

使用して Web サーバーと通信する Web ブラウザーや、RMI/IIOP を使用して J2EE アプリケーション・サーバー上で J2EE 方式の呼び出しを行う J2EE クライアントの場合があります。

2. 呼び出しが、IBM WebSphere InterChange Server システムと通信するよう構成された適切なサーバー側コンポーネント (Web サーバーの場合はサーブレット、J2EE アプリケーション・サーバーの場合は EJB) によって処理されます。コンポーネントは、作成されるビジネス・オブジェクトと、呼び出しによって起動されてサーバー・アクセス・インターフェースに呼び出しを送信するコラボレーションを指定します。
3. サーバー・アクセス・インターフェースが呼び出しを受信し、それを適切なデータ・ハンドラー (この例では XML データ・ハンドラー) に渡します。データ・ハンドラーはデータをビジネス・オブジェクトに変換し、それをサーバー・アクセス・インターフェースに戻します。
4. サーバー・アクセス・インターフェースがビジネス・オブジェクトを指定されたコラボレーションに渡します。
5. コラボレーションは、(a) ビジネス・オブジェクトに対してビジネス・プロセスを実行し、(b) 同時にビジネス・オブジェクトをサーバー・アクセス・インターフェースに戻して、元のクライアント・ブラウザーまたはアプリケーションに転送されるようにします。ビジネス・オブジェクトには、プロセスの結果として生成されたビジネス・データまたは例外通知が含まれます。
6. この例で、コラボレーションのビジネス・ロジックは、ビジネス・オブジェクトを要求としてテクノロジー・コネクターに送信するよう指定します。テクノロジー・コネクターは、データをインターネット経由で外部 Web サーバーに送信するために使用されます。

   また別のシナリオでは、コラボレーションがテクノロジー・コネクターではなくアプリケーション・コネクターにビジネス・オブジェクトを送信する場合があります。アプリケーションはローカルに置くことができますが、リモート・エージェント・テクノロジーが使用されている場合には、アプリケーションとそのコネクター・エージェントはハブから離してインターネット上に置くことができます。
7. この例では、XML コネクターがビジネス・オブジェクトを XML 文書に変換し、それを Web サーバーに送信します。
8. Web サーバーが応答を XML コネクターに送信します。
9. XML コネクターが応答をビジネス・オブジェクトに変換し、それをコラボレーションに送信します。
10. コラボレーションがそのビジネス・オブジェクトにビジネス・プロセスを実行します。
11. コラボレーションがビジネス・オブジェクトをサーバー・アクセス・インターフェースに送信します。
12. サーバー・アクセス・インターフェースがビジネス・オブジェクトを適切なデータ・ハンドラーに送信し、そのフォーマットのデータを受信します。
13. サーバー・アクセス・インターフェースが IIOP プロトコルを使用して、データをサーブレットに送信します。
14. サーブレットが HTTP/HTTPS プロトコルを使用して、データを元の Web ブラウザーまたは他のエンティティーに送信します。

コネクター

コネクターは、IBM WebSphere Business Integration Adapters 製品の一部として提供されています。コネクターは IBM WebSphere InterChange Server システムに分散変換サービスを提供し、コラボレーションと次のいずれかとの間でデータをやり取りします。

- アプリケーション
- コネクターが使用する XML などのテクノロジー規格を理解するプログラマチック・エンティティー (例えば、リモート Web サーバー)

InterChange Server 環境では、コネクターは次の分散構造を持ちます。

- **コネクター・コントローラー**はコラボレーションと直接対話し、InterChange Server プロセス内でコンポーネントとして実行します。
- **クライアント・コネクター・フレームワークは** InterChange Server から分離したプロセスとして実行し、アプリケーション固有のコンポーネントとともに使うと、アプリケーションやその他のプログラマチック・エンティティーと直接対話します。本書では、クライアント・コネクター・フレームワークとアプリケーション固有のコンポーネントを合わせて**コネクター・エージェント**と呼びます。

コネクターの 2 つの部分は、同一システム上でも 2 つの異なるシステム上でも実行できます。コネクター・コントローラーは、InterChange Server の一部として実行するため、このサーバーのあるシステム上に置きます。一方、コネクター・エージェントは、そのアプリケーションとコネクター・コントローラーの両方と通信できるあらゆるシステム上に置くことができます。

コネクターとアプリケーション間の通信

アプリケーションの各バージョンには、それぞれ 1 つのコネクターがあります。コネクターはアプリケーション・インターフェースを使用してアプリケーションと通信するので、それぞれ一意のコネクターになります。アプリケーション・プログラミング・インターフェース (API) がある場合、コネクターはそれを使用できます。しかし、API を持たないアプリケーションのコネクターでも、ユーザー出口や E メール・メッセージなど、アプリケーションが提供するあらゆる方式を使用できます。

コラボレーションに関連するアプリケーション・イベントを検出するために、コネクターはアプリケーションをポーリングしたり、アプリケーション・イベントのコールバック通知機構 (ある場合) を使用します。また、コネクターはコラボレーションのコマンドでアプリケーションと対話したり、コラボレーションの直前の要求結果を確認したりできます。

要素間のバインド

ビジネス・プロセスを実行するため、コラボレーションはコネクター、他のコラボレーション、および外部プロセスと通信して、サーバー・アクセス・インターフェース経由でアクセス要求を受信できます。コラボレーションを構成するときには、コラボレーションとこれらの要素間の通信を設定するために**バインド**を行います。

トリガーのバインド

コラボレーションのビジネス・プロセスを起動するには、構成時に、トリガー・イベントや呼び出しを提供する要素をコラボレーションにバインドする必要があります。

バインドは、コラボレーションのビジネス・プロセスに参加するあらゆる要素とコラボレーション間で実行できますが、トリガーとしてバインドされるのは 1 つの要素のみです。

イベントのためのバインド

コラボレーションがイベントによって起動されるようにサイトで構成する場合、トリガー・イベントをパブリッシュできるコネクターにコラボレーションをバインドします。例えば、コラボレーションの Employee.Delete イベントが PeopleSoft コネクターからパブリッシュされるように指定できます。

この関係をより詳しく見てみるには、コネクターは、実際にはコネクター・コントローラー (コラボレーションと同様に InterChange Server に置かれる) とコネクター・エージェント (クライアント・コネクター・フレームワークとアプリケーション固有のコンポーネントが組み込まれ、InterChange Server から分離している) の 2 つで構成されていることを思い出してください。コネクター・コントローラーは、バインド情報を保持し、コラボレーションがサブスクライブしているイベントのリストをコネクター・エージェントに提供します。

コネクター・エージェントは、関係のあるアプリケーション操作が実行されると、リストにあるイベントをコネクター・コントローラーにパブリッシュします。コネクター・エージェントがイベントをコネクター・コントローラーに送信するときには、コラボレーションの最終的な宛先については一切認識していません。

図 8 に示されているように、コネクター・コントローラーはコネクター・エージェントとコラボレーション間を媒介します。

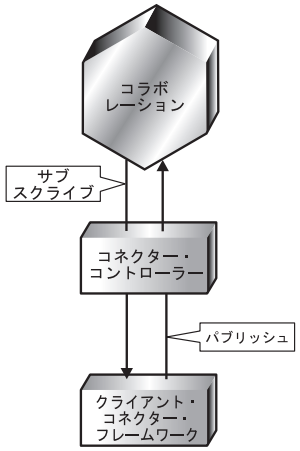

図 8. トリガー・イベントを提供するコネクター

複数のコラボレーションが同一のイベントにサブスクライブする場合があります。コネクター・コントローラーがイベントをパブリッシュするとき、同時にすべてのサブスクライバーにパブリッシュすることができます。

アクセス要求を受信するためのバインド

トリガーのためにコラボレーションをコネクターにバインドする代わりに、コラボレーションが外部プロセスからアクセス要求をトリガーとして受信するように指定できます。

バインドの宛先

コラボレーションは、トリガー要素へのバインドに加えて、コラボレーションが要求/応答対話を行う宛先要素へもバインドできます。宛先要素は、コネクターの場合と他のコラボレーションの場合があります。1 つのコラボレーションを複数の宛先要素にバインドできます。

ビジネス・オブジェクト

コラボレーションとコネクターは、InterChange Server を介した**ビジネス・オブジェクト**の送受信によって対話します。

ビジネス・オブジェクトは、データ・エンティティー (1 つの動作単位として扱われるデータの集合) を反映します。例えば、データ・エンティティーがフォームと同等で、フォームのフィールドをすべて含める場合があります。フォームは通常、アプリケーション内または Web 上で使用され、顧客、従業員、または送り状についてのビジネス情報が取り込まれます。

ビジネス・オブジェクトは、コラボレーションの実行中に高速アクセスのためにメモリー内のキャッシュに入れられます。また、トランザクション状態の永続的ストアにも格納され、障害後のサーバーの再始動の際に耐久力のあるリカバリー、ロールバック、およびコラボレーションの再実行を提供します。

IBM WebSphere InterChange Server システムは、エンティティーに含まれる情報を反映したビジネス・オブジェクトを作成します。本書では、データ・エンティティーという用語はそこに含まれるビジネス情報の関連でしばしば使用されます。例えば、*従業員エンティティー* や*顧客エンティティー* などがあります。

このセクションでは、ビジネス・オブジェクトについて簡単に説明します。以降の章では、ビジネス・オブジェクト、ビジネス・オブジェクトの内容、さらに IBM WebSphere InterChange Server システムがビジネス・オブジェクトを管理および処理する方法について説明します。

ビジネス・オブジェクトの役割

ビジネス・オブジェクトは、イベント、要求、または応答として動作できます。

イベント

ビジネス・オブジェクトは、アプリケーション内のデータ・エンティティーに関係する操作である**アプリケーション・イベント**の発生を報告します。アプリケーション・イベントは、そのデータ集合の値の作成、削除、または変更である可能性があります。コネクターがアプリケーション・イベントを検出して、ビジネス・オブジェクトを関連するコラボレーションに送信するとき、ビジネス・オブジェクトはイベントの代わりを務めるので、IBM WebSphere InterChange Server システムでは**イベント**と呼ばれます。

例えば、コネクターがコラボレーションのために新しい従業員エンティティーのアプリケーションをポーリングすることがあります。アプリケーションが新しい従業員エンティティーを作成する場合、コネクターはイベント・ビジネス・オブジェクトをコラボレーションに送信します。

要求

要求は通常、次の 2 つの方法のうちどちらかで生成されます。

- コラボレーションはビジネス・オブジェクトを**要求**としてコネクターに送信でき、コネクターがアプリケーション内のデータを挿入、変更、削除、または検索するように指示します。例えば、図 5 の Service Billing コラボレーションでは、コラボレーションは契約の検索および送り状の作成という 2 つのビジネス・オブジェクトをコネクターに送信しています。両方とも要求です。
- コラボレーションが Retrieve 動詞をトリガーとして受け入れるように設計またはカスタマイズされている場合、サーバー・アクセス・インターフェースはビジネス・オブジェクトを要求としてそのコラボレーションに送信できます。

応答

コネクターが要求の処理を終了すると、通常、**応答**を戻します。例えば、コネクターがアプリケーションから従業員データを検出する要求を受信すると、コネクターは従業員データを含むビジネス・オブジェクトを送信します。

ビジネス・オブジェクトの構造

ビジネス・オブジェクトは、タイプ (名前)、処理命令 (動詞)、およびデータ (属性値) を含んだ自己記述ユニットです。図 9 に、単純なビジネス・オブジェクトの例として、タイプ、動詞、および属性値を示します。

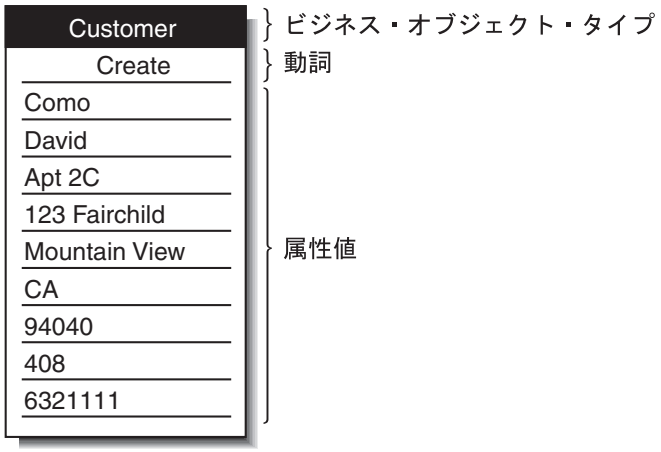

図 9. ビジネス・オブジェクト・コンポーネント

次のセクションでは、これらのコンポーネントについて説明します。

ビジネス・オブジェクト・タイプ

各ビジネス・オブジェクトには、IBM WebSphere InterChange Server システム内で識別するためのタイプ名があります。このタイプは、ビジネス・オブジェクト定義によって定義されます。例えば、Customer、Employee、Item、Contract などのタイプがあります。

ビジネス・オブジェクトの動詞

ビジネス・オブジェクトの動詞は、属性値に対するアクションを指定します。動詞は、ビジネス・オブジェクトの役割に応じてさまざまなタイプのアクションを表すことができます。表 1 は、3 つのビジネス・オブジェクトの役割をリストし、各役割を持つビジネス・オブジェクトの動詞の意味を説明しています。

表 1. ビジネス・オブジェクトの動詞の意味

ビジネス・オブジェクトの役割	動詞の意味
イベント	アプリケーション内で発生したことを説明します。例えば、イベントにおける Create 動詞は、ソース・アプリケーションが新しいデータ・エンティティーを作成したことを表します。
要求	ビジネス・オブジェクトを処理するためにアプリケーションと対話する方法をコネクターに伝えます。例えば Update 動詞は、コネクターにデータ・エンティティーを更新させる要求です。
応答	直前の要求の結果を提供します。例えば、応答における Retrieve 動詞は、コネクターがアプリケーションから属性値を取得したことを表します。

注: 命名規則は、*ビジネス・オブジェクト・タイプ.動詞* の形式で特定のタイプのビジネス・オブジェクトと特定の動詞を表すためのものです。例えば Customer.Create は、Customer ビジネス・オブジェクトと Create 動詞を表します。

ビジネス・オブジェクト属性値

ビジネス・オブジェクトには、Last Name、First Name、Employee ID、Invoice Status などのデータ・エンティティーに関連するデータ・フィールドを表す**属性値**が含まれます。

一部の属性には、データが含まれる代わりに、**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**が含まれます。 図 10 に、Contract ビジネス・オブジェクトの構造を示します。 Contract 内の品目名の情報は、子ビジネス・オブジェクトの配列に含まれます。

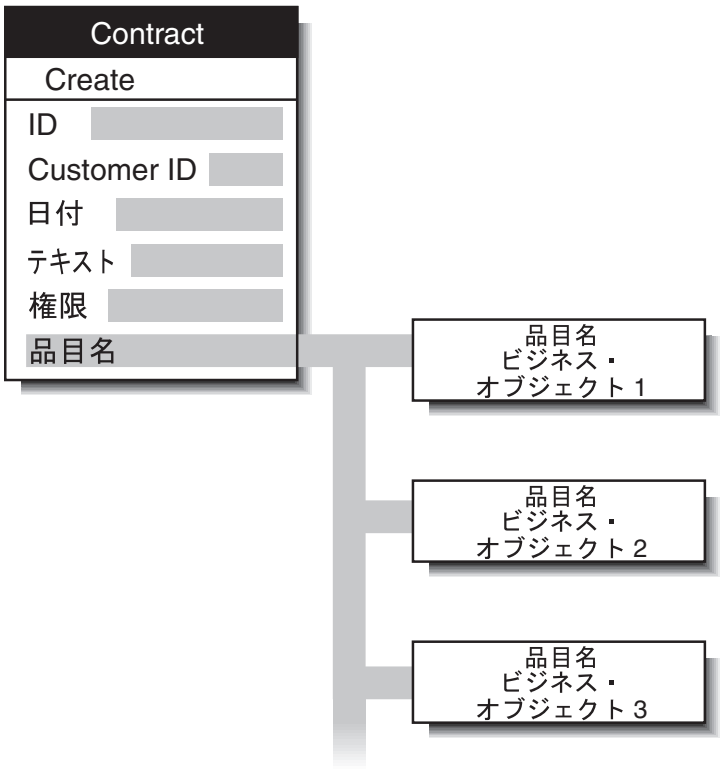

図 10. 子ビジネス・オブジェクトを持つビジネス・オブジェクト

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含むビジネス・オブジェクトは、**階層ビジネス・オブジェクト**といいます。属性にデータのみを含むビジネス・オブジェクトは、**フラット・ビジネス・オブジェクト**といいます。

アプリケーション固有のビジネス・オブジェクトおよび汎用ビジネス・オブジェクト

IBM WebSphere InterChange Server システムには、アプリケーション固有と汎用という 2 種類のビジネス・オブジェクトがあります。

- **アプリケーション固有のビジネス・オブジェクト**は、特定のアプリケーションまたはプログラマチック・エンティティーのデータ・エンティティー属性とデータ・モデルを反映します。
- **汎用ビジネス・オブジェクト**には、広範囲のアプリケーションに共通する一連のビジネス関連属性が含まれ、特定のアプリケーションのデータ・モデルに拘束されません。

コネクター・エージェントが更新などのアプリケーション・イベントを (アプリケーション固有のコンポーネントを介して) 検出すると、アプリケーションから適切なデータ・エンティティーを検索して、それを**アプリケーション固有のビジネス・オブジェクト**に変換します。

注: 本書では、Clarify_Contact や Oracle_Customer のようにアプリケーション名が含まれたビジネス・オブジェクトは、アプリケーション固有のビジネス・オブジェクトを指します。例えば Clarify_Contact ビジネス・オブジェクトには、

Clarify アプリケーションが連絡先に関連して保管している一連の情報が含まれます。別のアプリケーションでは、連絡先エンティティーに一連の異なる情報が保管されていたり、異なる順序または形式で情報が保管されていたり、異なる名前が付けられていたりする可能性があります。

コネクター・エージェントがアプリケーション固有のビジネス・オブジェクトの作成を終了すると、そのビジネス・オブジェクトを InterChange Server 内のコネクター・コントローラーに送信します。

コネクター・コントローラーは、コラボレーションとコネクター・エージェント間でビジネス・オブジェクトを交換します。コラボレーションは一般的にアプリケーションに中立なので、コネクター・コントローラーがコラボレーションとの間で交換するビジネス・オブジェクトは**汎用ビジネス・オブジェクト**である必要があります。汎用ビジネス・オブジェクトのビジネス・ロジックは特定のアプリケーションの特定のバージョンにバインドされていないので、汎用ビジネス・オブジェクトを使用するとコラボレーションの再使用可能性が高まります。

注: 汎用ビジネス・オブジェクトの名前には、企業名や製品名は含まれません。例えば、Contact、Employee、Customer などです。

図 11 は、IBM WebSphere InterChange Server システム内で 2 種類のビジネス・オブジェクトが置かれる場所を示しています。コラボレーションは汎用ビジネス・オブジェクトを使用して対話し、コネクター・エージェントは特定のアプリケーション用に設計されたビジネス・オブジェクトをサポートします。

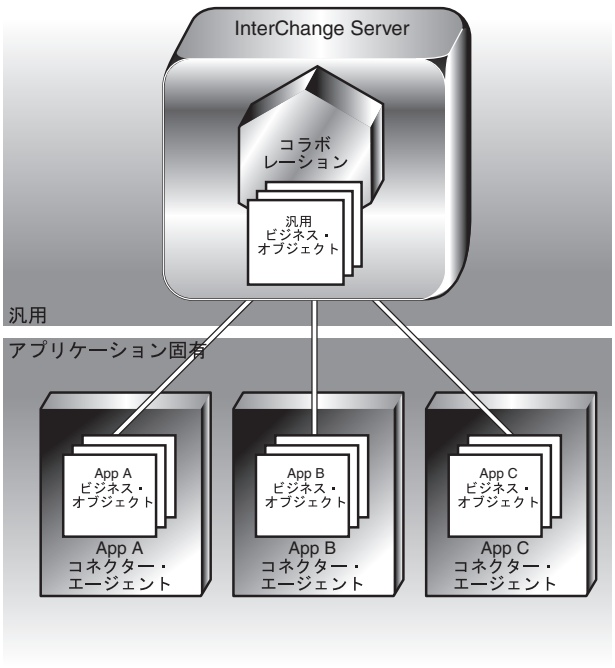

図 11. 汎用ビジネス・オブジェクトとアプリケーション固有のビジネス・オブジェクト

コネクターが複数のタイプのコラボレーションによって使用されているビジネス・オブジェクトをサポートしている場合は、同じコネクターを使用して、これらのコラボレーションを実行できます。

データ・マッピング

図 11 は、各アプリケーションとコラボレーションにあるさまざまなタイプのビジネス・オブジェクトを示しています。したがって、IBM WebSphere InterChange Server システムはビジネス・オブジェクトをそれぞれの形式に変換して、イベントやデータをアプリケーションとコラボレーション間でやり取りできるようにする必要があります。あるタイプのビジネス・オブジェクトを別のタイプに変換するプロセスを**データ・マッピング**といいます。データ・マッピングは、IBM WebSphere InterChange Server システムがデータをやり取りするソースと宛先の間で同一のデータ・モデルが共有されていない場合には、常に必要です。

1 つのアプリケーションから別のアプリケーションにデータを直接マッピングするカスタム・アプリケーション統合ソリューションとは違って、InterChange Server Collaborations は、一般的に複数のアプリケーション固有のデータ・モデル間に汎用ビジネス・オブジェクトを使用します。汎用ビジネス・オブジェクトは、共通のクロス・アプリケーション・データ・セットとして機能します。将来アプリケーションを変更する場合には、新規のコネクターを入手して、新規のアプリケーション固有のビジネス・オブジェクトを汎用ビジネス・オブジェクトにマッピングするだけで済みます。これでコラボレーションは以前と同様に動作します。

コラボレーションがビジネス・オブジェクトを異なるアプリケーションに転送するときは、マッピングによってビジネス・オブジェクトを共通のデータ・セットに、およびその逆に変換します。ビジネス・オブジェクトの変換は、次のように実行されます。

- ビジネス・オブジェクトがコネクターからコラボレーションに渡されるときは、アプリケーション固有から汎用へ
- ビジネス・オブジェクトがコラボレーションからコネクターに渡されるときは、汎用からアプリケーション固有へ

例えば、Clarify_BusOrg データと SAP_CustomerMaster データを同期するコラボレーションでは、マッピングは 2 回実行されます。

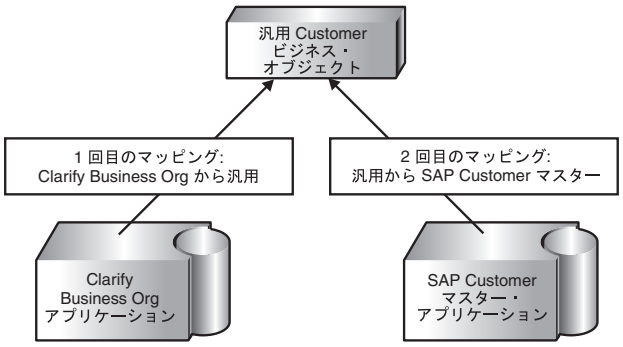

図 12. マッピング変換

コネクター・エージェントと InterChange Server 間で渡されるビジネス・オブジェクトのマッピングは、コネクター・コントローラーが管理します。ただし、データ・マッピングを実際に行うには、システムが、マッピング・ツールである Map Designer および Relationship Designer を呼び出します。これらのツールによって、マッピング仕様の詳細を作成および修正したり、実行時にマッピングを実行したりできます。

コネクター・コントローラーは、マッピングを必要とするビジネス・オブジェクトを受信すると、マッピング機能を呼び出します。図 13 に、コネクター・コントローラーからのマッピングの呼び出しを示します。

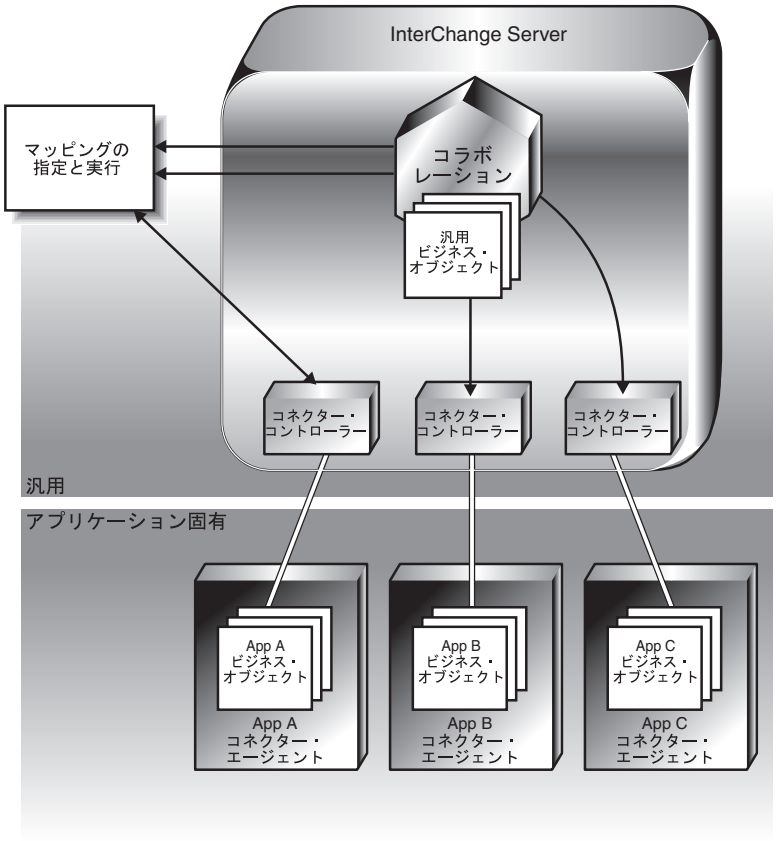

図 13. マッピングの制御と実行

InterChange Server

InterChange Server は、コラボレーションのためのマルチスレッド化された Java ベースの実行フレームワークです。InterChange Server は、自身の Java 仮想マシン (JVM) で実行します。 InterChange Server は Windows と UNIX のシステム上で実行します。このセクションでは、InterChange Server の以下のサービスや機能について説明します。

- 27 ページの『イベント管理サービス』

- 『コネクター・コントローラー』
- 『リポジトリー』
- 『データベース接続サービス』
- 28 ページの『データベース接続プール』
- 28 ページの『高可用性』
- 29 ページの『トランザクション・サービス』
- 31 ページの『リカバリー機能』

イベント管理サービス

InterChange Server は、コラボレーションの実行時に受信するすべてのビジネス・オブジェクトを永続的に保管します。これによって InterChange Server は、イベント通知や呼び出しを失うことなく、予期しない終了またはコラボレーションの障害から回復することができます。

コネクター・コントローラー

コネクター・コントローラーは、クライアント側のコネクターと InterChange Server 間のインターフェースを指します。コネクター・コントローラーは、ビジネス・オブジェクトが IBM WebSphere InterChange Server システムをトラバースするときにその経路を定め、クライアント側のコネクターとコラボレーションのリンクおよびマッピング・プロセスの管理を行います。

コネクター・コントローラーを使用することにより、管理者は次のことを実行できます。

- InterChange Server とコネクター・エージェント間の対話をトレースします。
- InterChange Server とコネクター・エージェント間の対話を使用可能および使用不可にします。
- マッピングのタイプを指定して、コネクター・エージェントとの間でビジネス・オブジェクトを送受信できるようにします。

リポジトリー

InterChange Server は、すべてのオブジェクトの構成情報および定義を、**InterChanger Server リポジトリー**という永続的ストアに保持します。このリポジトリーは、関連するデータベース内の表の集合から構成されます。これらの表には、オブジェクト定義および構成情報が XML 文書の形式で保管されています。

データベース接続サービス

データベース接続サービスは、InterChange Server とリポジトリー間の対話を管理します。図 14 に示すように、データベース接続サービスは、JDBC (Java Database Connectivity) API を使用してリポジトリーと対話します。

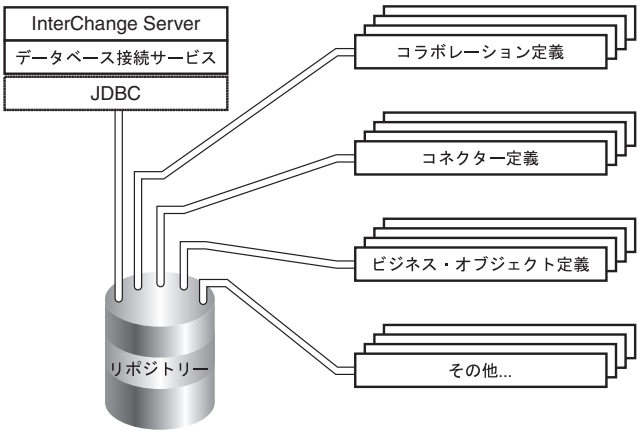

図 14. データベース接続サービスおよびリポジトリー

注: ODBC (Open Database Connectivity) ドライバーを持っているが固有の JDBC ドライバーを持っていないデータベースもあります。このようなデータベースの場合、有効な JDBC-ODBC ブリッジを使用できます。

InterChange Server は多数のデータベース・ベンダーをサポートしており、新たなデータベースの認証を継続します。現在サポートされているデータベースのリストについては、「*システム・インストール・ガイド (Windows 版)*」または「*システム・インストール・ガイド (UNIX 版)*」を参照してください。

データベース接続プール

IBM WebSphere InterChange Server システムの System Manager ツールを使用すると、InterChange Server でデータベース接続プールを定義できます。ユーザー定義のデータベース接続プールによって、開発者がコラボレーションまたはマップから関連するデータベースに直接アクセスできるようになります。この機能は、次のことをサポートします。

- 自動化されたデータベース接続ライフ・サイクルの管理
- SQL ステートメントおよびストアード・プロシージャー実行のための単純化された API
- コンテナー管理のトランザクション・ブラケット

高可用性

IBM WebSphere InterChange Server システムは、InterChange Server (ICS) に高可用性 (HA) を提供するように構成できます。

Windows システムでは、HA 構成は Microsoft Cluster Server (MSCS) ソフトウェアで実行します。HA 構成には、Microsoft 認証のクラスター・サーバー・マシンが 2 台必要です。2 台のサーバーは同一の InterChange Server システム構成でセットアップされ、MSCS ソフトウェア内でクラスター・システムとして指定されます。1 台のマシンは MSCS にプライマリー (障害が発生するまでアクティブなサーバー) として構成し、もう 1 台はバックアップとして構成します。2 台のマシンは、外部プロセスがアクティブ・サーバーにアクセスするために使用する、クラスター名と

クラスター IP アドレスを共有します。プライマリー・サーバーおよびバックアップ・サーバーはともに、アクティブ・サーバーが制御する共有 RAID ストレージ・システムへのアクセス権を持ちます。

ハードウェア障害が検出されると、HA 構成はクラスター・バックアップ・システム上で ICS のシャットダウン、自動マイグレーション、および ICS (ならびにサード・パーティー製ソフトウェアと HA がサポートするコネクター) の再始動を実行します。クラスター・バックアップ・サーバーはアクティブ・サーバーになり、プライマリー・サーバーのクラスター名と IP アドレスを引き継ぎます。プライマリー・サーバーの障害が修復されてフェイルバック (つまり、手動でプライマリー・サーバーに処理を戻すこと) が開始されるまで、バックアップ・サーバーがシステム処理を自動的に引き継ぎます。

HA 環境の概要については、「*システム管理ガイド*」を参照してください。HA 環境用に ICS を構成する方法の詳細については、「*システム・インストール・ガイド*」を参照してください。

トランザクション・サービス

いかなるアプリケーション統合ソリューションでも、データをあるアプリケーションから別のアプリケーションに移動させるときにはリスクに直面します。データがアプリケーションおよびそのデータベースの保護領域を出て、ネットワーク経由で別のアプリケーションに送られるときには、問題が発生する可能性は数多くあります。例えば、次のような問題が考えられます。

- ネットワークがダウンする。
- データを受信する前に、宛先アプリケーションが破損する。
- 宛先アプリケーションが新規データを処理する機会を得る前に、エラーが発生する。

人事、給与計算、および製品別原価計算という 3 つのアプリケーションを含むコラボレーションがあるとします。各アプリケーションには、従業員給与が格納されています。図 15 に、従業員の昇給を処理するためのクロス・アプリケーション・ビジネス・ロジックをハイレベル表示で示します。

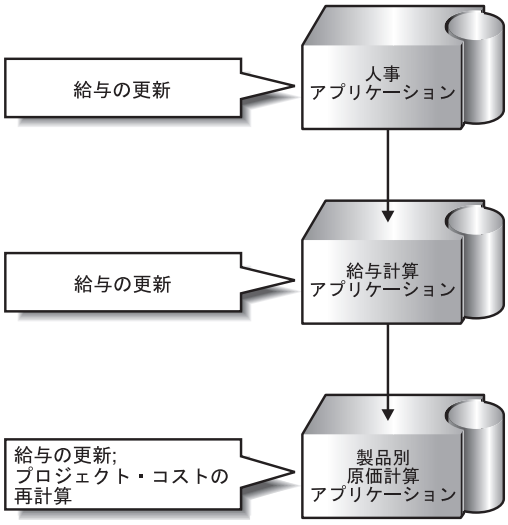

図 15. 3 つのアプリケーションによる従業員給与の処理

ある従業員が昇給すると、担当者が人事アプリケーションに新規給与を入力します。コラボレーションは自動的に給与計算アプリケーションの従業員給与を更新し、製品別原価計算アプリケーションのプロジェクト費用を調整します。

システム・エラーによって給与更新が製品別原価計算アプリケーションに伝達されない場合、データ不整合が生じます。従業員給与の値がアプリケーションごとに異なるので、製品原価が誤って計算されます。

従来は、入念なクロスチェックが必要でした。これを行わないと、誰かが問題に気が付くまでデータは誤ったままでした。しかし、IBM WebSphere InterChange Server システムは、エンタープライズ・データを信頼できるよう厳密に制御された高度なサービスを提供するので、コラボレーションを一種の**トランザクション**であるかのように実行できます。

トランザクション・コラボレーション

トランザクションの特性は、アプリケーション全体でデータの整合性が重要であるコラボレーションにとって理想的であり、目的を達成できます。他のトランザクションのように、トランザクション・コラボレーションには一連のステップが含まれます。エラーが発生すると、InterChange Server はトランザクションのようにロールバックを実行して、完了した各ステップを元に戻すことができます。

ただし、コラボレーションと従来のトランザクションは、次の点で大きく異なります。

- コラボレーションのアクションは分散しており、関係するデータベースを中央制御することはできません。
- イベントに応答するコラボレーション (パブリッシュ・アンド・サブスクライブ・モデルなど) は長期存続型です。つまり、こうしたコラボレーションが実行中にアプリケーション・データを分離するのはアプリケーション・ユーザーに悪影響を与えるので、非同期に実行します。

- アプリケーションはコラボレーションによるデータ変更を保管することにより、分散したクロス・アプリケーション形式の持続性を提供します。ただし、コラボレーションをロールバックする必要がある場合、直前に保管した操作を取り消す必要があります。

この結果、トランザクション・コラボレーションをサポートするために InterChange Server が使用する手法は、従来のトランザクションをサポートするために使用する手法とは異なります。コラボレーションに関するトランザクション・レベルは、InterChange Server がトランザクションのセマンティクスを強化する度合いを定義します。

リカバリー機能

InterChange Server 環境は、障害後のリブートに ICS が要する時間を改善する機能、すべてのフローが回復する前に ICS で他の作業を行えるようにする機能、および失敗したイベントの再実行依頼を制御する機能を備えています。

- 非同期リカバリー

  InterChange Server は、コラボレーションとコネクターが回復するのを待たずにブートを完了します。つまり、コラボレーションとコネクターは、InterChange Server のブート後に非同期に回復できます。これによって、コネクターとコラボレーションが回復中である一方で、System Monitor や「未解決のフローを管理」ダイアログなどの System Manager トラブルシューティング・ツールを使用できるようになります。

- 据え置きリカバリー

  この機能の使用はオプションであり、コラボレーション・オブジェクト・プロパティーを使用して構成します。この機能を使用可能にすると、ICS の障害が発生すると、コラボレーションのプロセス中の作業フローの回復は、サーバーがリブートするまで据え置かれます。このため、これらのフローに関連するメモリー使用量を節約できます。サーバーのリブート後、System Manager の「未解決のフローを管理」ダイアログを使用してイベントを再実行依頼できます。

- 転送中状態でサービスの呼び出しを持続

  非トランザクション・コラボレーションが宛先アプリケーションに重複してイベントを送信するのを回避するため、障害の発生時に、リカバリーに自動的に転送中のすべてのサービス呼び出しを再実行依頼させたくない場合があります。この場合には、障害およびリカバリーが発生したときにコラボレーションが転送中のすべてのサービス呼び出しを持続するように、コラボレーションを構成します (サーバーの障害に先立って実行します)。InterChange Server が回復するとき、サービス呼び出しを処理していたフローは転送中状態を維持するので、「未解決のフローを管理」ダイアログを使用して個々の未解決のフローを調査したり、それらのフローを再実行依頼するときに (またはする場合には) 制御したりできます。

- イベント・デリバリーの保証機能

  JMS 対応のコネクター (JMS をトランスポート機構として使用するコネクター) では、リカバリー状態でイベント・デリバリーを保証するには、次の機能が有効です。

  – コンテナー管理下のイベント機能

コンテナー管理下のイベント機能は、JMS イベント・ストアを使用する JMS 対応のコネクターに対して有効です。この機能により、システム破壊やリカバリーが発生したときに、イベント・ストアとコネクター・フレームワーク間で処理が行われていたイベントが、1 度だけコネクター・フレームワークに配信されます (2 度配信されることはありません)。この機能はオプションで、コネクター・プロパティーを介して構成されます。なお、この機能は、コネクターが JMS をトランスポート機構として使用している場合にのみ有効です。

– 重複イベントの除去機能

重複イベントの除去機能は、JMS 対応のコネクターに対して有効です。この機能は、コネクターのアプリケーション固有コードに固有のイベント ID を使用することにより、デリバリー・キューへのイベントのデリバリーが重複しないようにするものです。この機能はオプションで、コネクター・プロパティーを介して構成されます。なお、この機能は、コネクターが JMS をトランスポート機構として使用している場合にのみ有効です。

通信トランスポート・インフラストラクチャー

InterChange Server 環境のモジュラー・アーキテクチャーは、多様なサイト構成に適応する分散システムを構成します。InterChange Server 環境のアーキテクチャーによって、ネットワーク上の複数のマシンとの分散対話、およびインターネット・ファイアウォールを越えた分散対話が可能になります。

ネットワーク上の分散

ネットワーク上の分散 InterChange Server コンポーネント間の対話は、Common Object Request Broker Architecture (CORBA) およびメッセージング・テクノロジー (ネイティブ WebSphere MQ メッセージングや Java Message Service (JMS) など) により可能になります。

さまざまな構成が考えられます。CORBA は、コネクター・エージェントと ICS 間の要求/応答対話や管理通信に使用できます。ネイティブ WebSphere MQ は、パブリッシュ・アンド・サブスクライブ操作におけるイベントのデリバリーを実行します。一方、Java Message Service は、コネクター・エージェントと ICS 間のすべての通信および対話に使用できます。

CORBA

Common Object Request Broker Architecture (**CORBA**) は、ネットワーク上の分散オブジェクトのための規格およびインターフェースを定義します。オブジェクト・リクエスト・ブローカー (**ORB**) はライブラリーのセットであり、クライアント・アプリケーションとオブジェクト・サーバーが通信に使用するコンポーネントでもあります。 InterChange Server は IBM Java ORB 製品を使用します。

ORB によって、InterChange Server からクライアント、コネクター・エージェント、および System Manager にアクセスできるようになります。InterChange Server は ORB のネーム・サービスに登録し、そこからクライアントがサーバーとの対話を検出および開始するのに必要な情報を取得します。クライアントとサーバーは、ORB のインターフェース定義言語 (IDL) によってオブジェクト対オブジェクトの

対話を実行します。トランスポート・レベルでは、クライアントとサーバーは Internet Inter-ORB Protocol (IIOP) を使用して通信します。 ORB ベースの通信は、通常、次の目的で使用します。

- サーバー・アクセス・インターフェースは、ORB ベースの通信を使用して呼び出しを処理します。
- 要求/応答対話において、コラボレーションとコネクターは ORB ベースの通信を使用してビジネス・オブジェクトを交換します。
- コネクター・エージェントは、ORB ベースの通信を使用して次のことを実行します。
  - 始動時に、InterChange Server と対話する場合に自身の初期構成を取得します。
  - 操作中に、コネクター・コントローラーから指示を受け取ると、自身の状況の報告、一時停止、停止、または再開を実行します。
- オプションで、ORB ベースの通信は、パブリッシュ・アンド・サブスクライブ対話におけるイベントのデリバリーにも使用されます。

IIOP 要求/応答プロトコルでは、通信が即時に成功または失敗するので、コンポーネントが通信するために両方のプログラムが実行されている必要があります。一方、InterChange Server 環境における要求/応答対話では、コネクター・プロパティーを使用してストア・アンド・フォーワード・モードの設定を行えるため、コネクター・エージェントが使用できないときにコネクター・コントローラーがコラボレーションの要求に対してどのように応答するかを指定できます。

- プロパティーを true に設定した場合は、コネクター・エージェントが使用不可の場合でもコネクター・コントローラーはいかなるコラボレーション要求も失敗しません。要求は、コネクター・エージェントが操作可能になるまでブロックされます。これにより、コネクター・エージェントが操作可能になるまで、コラボレーションは要求のフロー処理の完了を待機することになります。
- false に設定した場合、コネクター・エージェントが使用不可の場合に、コネクター・コントローラーはすべてのコラボレーション要求に失敗します。コラボレーションは、失敗した要求処理のためのビジネス・ロジックによって要求の処理を完了します。

図 16 は、InterChange Server 用として IBM Java ORB を構成するコンポーネントを示します。InterChange Server コンポーネントが ORB ドライバーにアドレッシングするさまざまな言語には、それぞれ異なるコンポーネントが存在しますが、C++ および Java のコンポーネントは単一システムのように互いに通信します。IBM Transient Naming Server はネーミング・サービスを提供します。

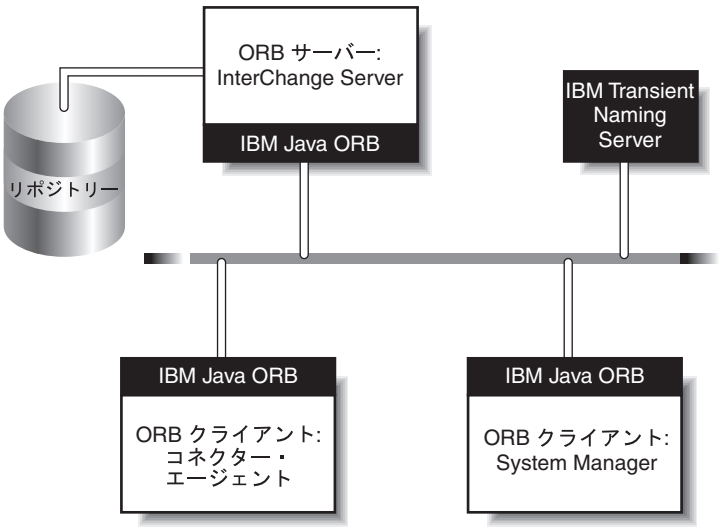

図 16. WebSphere InterChange Server システムの ORB コンポーネント

これらの ORB コンポーネントは分離する必要がありません。サイトによっては、すべての ORB コンポーネントが同一マシン上に存在する場合があります。

メッセージング・テクノロジー

メッセージングは、プログラムが非同期に離散的データ単位を交換する通信スタイルを形成します。メッセージング・トランスポートを使用するプログラムは、接続を確立したりメッセージを待つ必要がありません。それぞれのプログラムがメッセージング・サービスと対話することによって、メッセージを非同期に送受信します。メッセージング・サービスは、宛先プログラムが使用不可の場合にメッセージを保管して、使用可能になるまで再試行することにより、デリバリーを保証します。

InterChange Server 環境でサポートされているメッセージング・システムには、ネイティブ IBM WebSphere MQ メッセージングと Java Messaging Service (JMS) ソフトウェアの両方が含まれます。一般的に、メッセージング・システムはパブリッシュ・アンド・サブスクライブ対話でのイベント・デリバリーに使用され、ORB は要求/応答対話に使用されます。ただし、JMS を両方のタイプの対話に使用する場合もあります。

JMS がデリバリー・トランスポート機構である場合は、長期存続ビジネス・プロセス機能によってデータの永続性が与えられます。この機能が使用されているときは、コラボレーションの要求によって開始されたプロセスはタイムアウト値とともに待ち状態で置かれるため、指定されたデータ応答を受信した時点でこのプロセスが再開されます。この機能を使用するには、コラボレーション・テンプレートの作成時にこの機能を使用可能にしておく必要があります。

インターネット上の分散

InterChange Server システムは、コネクター・コントローラー/コネクター・エージェント対話をインターネット・ファイアウォールを越えて分散させるためにセキュア・ソケット・レイヤー (SSL) 通信をサポートしています。

実装はハブ・アンド・スポーク関係をとります。ハブとは IBM WebSphere InterChange Server システムを完全インストールしたサイトであり、スポークとはファイアウォールを越えてハブとデータを交換するリモート・サイトです。スポーク・サイトは、リモート・コネクター・エージェントを必要としますが、IBM WebSphere InterChange Server システム全体を必要とはしません。

次の 2 つの代替構成を使用します。

- SSL を使用した WebSphere MQ
- HTTP/HTTPS を使用した WebSphere MQ Internet Pass-Thru

InterChange Server の保護

WebSphere Business Integration システムは、保護される必要のある重要なビジネス・データを使用する分散システムです。転送中またはディスクに保管されたデータを、無許可の第三者が表示または変更するのを防ぐ必要があります。システムでのデータの保護には、必ず認証、メッセージ保全性、およびプライバシーの 3 つのセキュリティー・サービスが関係します。認証は、許可されていると主張するユーザーを検証することです。認証によるアクセス制御は、認証済みユーザーが誰であるか、およびそのユーザーに付与されている許可が何であるかに基づいて、システムの部分に対するアクセスを制限します。メッセージ保全性は、データが変更されていないことを保証します。プライバシーは、許可ユーザーのみがデータを表示できることを保証します。保全性、プライバシー、および認証は、暗号化により実現できます。一方、RBAC は、ユーザー ID とパスワードを設定することにより実現できます。

暗号化

3 つのセキュリティー・サービスのほかに、InterChange Server の保護に関係する 3 つのセキュリティー機構があります。これらの機構のうち、2 つの機構は、暗号化によるデータのエンコードに関係します。これらは**対称暗号化**および**非対称暗号化**です。対称暗号化では、暗号化アルゴリズムまたは鍵を使用してデータをエンコードします。データのエンコードとデコードでは、同じ鍵が使用されます。あるプロセスがデータを暗号化し、別のプロセスがそのデータを暗号化解除する必要がある分散システムでは、無許可ユーザーが鍵を使用してデータにアクセスできないようにするために、セキュアな方法で鍵を共用および交換する必要があります。鍵を共用できるようにすることが、対称暗号化における重要なポイントです。

非対称暗号化の機構は、鍵を共用または交換するために一般的に使用されている方法です。この方法では、データは**鍵ペア**を使用して暗号化および暗号化解除されます。鍵ペアの一方の鍵は秘密に保持される**秘密鍵**であり、鍵ペアのもう一方の鍵は他の人に共用および配布される**公開鍵**です。公開鍵を使用して暗号化されたデータは、鍵ペアの対応する秘密鍵によってのみ暗号化解除できます。送信側は受信側の公開鍵を使用してデータを暗号化します。このため、適切な受信側のみが自身の秘密鍵を使用してそのデータを暗号化解除できます。非対称暗号化は低速なため、データの暗号化には使用されません。その代わり、秘密鍵を非対称的に暗号化および暗号化解除するために使用され、この秘密鍵を使用して、データが対称的に暗号化されます。システムでのプライバシーを確保するために重要となるのが鍵の強度であるため、このプロセスにより、セキュリティーと速度の最適な組み合わせが実現されます。

秘密鍵は、データの**ディジタル署名**を作成するためにも使用されます。ディジタル署名は、送信側の識別 (認証) と、送信されたデータの保全性 (メッセージ保全性) を検査するために使用できます。ディジタル署名を作成するために、**ハッシュ**という処理でデータが小規模な固定長の**メッセージ・ダイジェスト**に変換されます。これがマップ (メッセージ・ダイジェスト) を作成する片方向の機能です。メッセージ・ダイジェストは送信側の秘密鍵を使用して暗号化され、これによりディジタル署名が作成されます。ディジタル署名はデータに追加され、受信側に送信されます。受信側は公開鍵を使用してディジタル署名を暗号化解除してメッセージ・ダイジェストに戻し、これにより署名が検証されます。なぜなら、公開鍵は、送信側のみが所有している秘密鍵を使用して暗号化されたデータのみを暗号化解除できるからです。その後、受信側はデータをハッシュしてメッセージ・ダイジェストを作成し、これを送信されたメッセージ・ダイジェストと比較します。両方のメッセージ・ダイジェストが同じであれば、受信側は、送信側が署名した後にデータが変更されていないことを確認できます。

重要なビジネス・データを保護するために、InterChange Server には、次のセキュリティー機能が組み込まれています。

- エンドツーエンド・プライバシー (エンドツーエンド・セキュリティーとも呼びます) は、ソース・アダプター・プロセスから InterChange Server を介して宛先アダプター・プロセスに流れるデータを保護します。エンドツーエンド・プライバシーでは、非対象および動的セキュリティーを使用して認証、保全性、およびプライバシーに対応します。
- 役割ベースのアクセス制御は、認証済みユーザーが誰であるか、およびそのユーザーに付与されている許可が何であるかに基づいて、システムの部分に対するアクセスを制限します。役割ベースのアクセス制御は、アクセス制御および認証を扱います。

エンドツーエンド・プライバシー

アダプター・プロセスと InterChange Server は、メッセージを送信することにより通信します。メッセージは双方向であり、InterChange Server とアダプター・プロセスは、どちらもメッセージの送受信を行うことができます。SSL は、ノード間、またはリンク・レベルのセキュリティーを提供します。メッセージは、ノード間を移動する際は保護されますが、ノードでディスクに保管されている時は保護されません。メッセージおよびデータは平文であり、無許可ユーザーがこれらを表示したり変更したりすることができます。エンドツーエンド・プライバシーでは、発信元のノードから送信されたメッセージが中間ノードを経由して宛先ノードに到達するまでの間、メッセージを保護することができます。非対象および動的セキュリティーを使用すると、転送中のメッセージが保護され、処理を待つ間、ディスク上のキュー・マネージャーで保管されている間も保護されます。メッセージは、暗号化や**エンドポイント認証**など、さまざまなレベルのセキュリティーで保護されます。エンドポイント認証は、通信におけるエンドポイント (InterChange Server やアダプターなど) が、ディジタル署名を使用して、自分の身元が、自分が主張しているものであることを証明するときに発生します。

非対称および動的セキュリティー

分散システムでは、送信側と受信側の両方が、通信の通話者になり得ます。通信が保護される必要がある場合、着信メッセージと発信メッセージのセキュリティー・

レベルが同じであれば、これは対称セキュリティーとみなされます。このタイプのセキュリティーは、サーバーとアダプターの間に密結合を作成します。SSL は、対称セキュリティーの代表的な形式です。

サード・パーティー製セキュリティー製品の多くは、**Secure Sockets Layer** (SSL) セキュリティー・プロトコルに基づいています。SSL は、TCP/IP 接続に関して、データ暗号化、サーバー認証、メッセージの保全性、およびオプションでクライアント認証を提供します。通常、SSL は対称セキュリティーを使用する密結合されたシステムに関連するため、データが重要かどうかに関係なく、すべてのデータが暗号化されます。SSL では、非セキュアなデータとセキュアなデータを区別することができません。着信メッセージと発信メッセージとのセキュリティー・レベルが違ってもよい場合、これは非対称セキュリティーとみなされます。例えば、あるユーザーがサーバーから大きな文書を受信する前にユーザー ID とパスワードを使用してサーバーで認証する必要があるとします。ユーザー ID およびパスワードは、転送中に無許可ユーザーによって見られないように暗号化する必要がありますが、転送中の文書を暗号化する必要はありません。SSL を使用する場合、ユーザー ID およびパスワードの両方、並びに大きな文書が暗号化されます。大きな文書を暗号化すると、サーバーのパフォーマンスに影響を与える可能性があります。

非対称セキュリティーを使用する場合、ユーザーからのメッセージ・フローではユーザー ID およびパスワードを暗号化するように設定し、サーバーからユーザーへのメッセージ・フローでは暗号化を行わないように設定することができます。ユーザーからのユーザー ID およびパスワードは暗号化によって保護され、一方、ユーザーに大きな文書を送信する前にこの文書を暗号化する必要がないので、サーバーのパフォーマンスは高まります。非対称セキュリティーでは、通信の通話者間のセキュリティー・パラメーターに対する動的更新も使用可能です。SSL などの非対称セキュリティーを使用する場合、セキュリティー・パラメーターは、通信開始時に設定されます。セキュリティー・パラメーターに対する変更は、新しい通信セッションが開始されるまで有効にはなりません。非対称セキュリティーを使用する場合、通信の各通話者が個々のセキュリティー・パラメーター定義を所有しているため、セキュリティー・パラメーターに対する変更を動的にすることができます。一方の通話者のセキュリティー・パラメーターに対する変更は、現在のセキュリティー・セッションを中断せず、他方の通話者に対する更新を要求します。非対称セキュリティーは柔軟ですが、エンド・ノードと直接結合していても、弱く (または間接的に) 結合している中間ノードでは許可されていないソースによってデータが操作されてしまうぜい弱性があります。

エンドツーエンド・プライバシーを使用すると、アプリケーション層でセキュリティーを設定し、アプリケーションに出入りする前にメッセージを保護できます。このタイプのセキュリティーにより、中間ノードにメッセージがあるときに許可されていないソースがメッセージにアクセスする危険性を回避できます。エンドツーエンド・プライバシーにより、メッセージ・タイプごとにセキュリティーのレベルを変えることもできます。

セキュリティーのレベル

セキュリティーのレベルは、個々のアダプターおよび InterChange Server について、それぞれのメッセージ・タイプごとに構成されます。次の 4 つのセキュリティー・レベルがあります。

なし セキュリティー・レベルは設定されず、アダプターまたは InterChange Server からのメッセージは標準どおりに送信されます。これはデフォルトのセキュリティー・レベルです。

保全性 送信側 (アダプターまたは InterChange Server) は、メッセージにディジタル署名を追加します。受信側は、送信側の公開鍵を使用して署名を検証します。送信側の秘密鍵を使用できるのは送信側だけであり、送信側だけがこれを使用して署名を生成できるので、受信側は署名を検証することにより、送信側を認証します。署名はデータのメッセージ・ダイジェストであり、受信側がメッセージ・ダイジェストを生成して、これがメッセージと一緒に送信されたメッセージ・ダイジェストと一致することを確認すると、メッセージの保全性が保証されます。データが変更されていないことを検証するプロセスでは、送信側の身元も検証されるため、保全性には認証も暗黙的に含まれます。

プライバシー

送信側はメッセージを完全に暗号化します。受信側はメッセージを暗号化解除して、その後の処理にメッセージを渡します。アダプターまたは InterChange Server が「プライバシー」を構成してメッセージを送信する場合、送信側は秘密鍵を生成し、秘密鍵を使用してメッセージを暗号化し、受信側の公開鍵を使用して秘密鍵を暗号化し、暗号化された秘密鍵をメッセージに組み込み、それからメッセージを送信します。受信側は、自身の秘密鍵を使用して秘密鍵を暗号化解除し、暗号化解除された秘密鍵を使用してメッセージを暗号化解除し、その後の処理にメッセージを転送します。

秘密鍵は、受信側の公開鍵を使用して暗号化されます。暗号化された秘密鍵は、受信側の秘密鍵でのみ暗号化解除できます。受信側の秘密鍵を使用できるのは受信側だけなので、メッセージはセキュアになります。

保全性とプライバシー

送信側は、『保全性』で説明したように各メッセージにディジタル署名を追加し、次に、『プライバシー』で説明したようにメッセージを完全に暗号化します。

InterChange Server とアダプターは、自身の鍵ストアを保持します。鍵ストアは、パスワード保護されたファイルであり、公開鍵と秘密鍵をセキュアに保管するために使用します。InterChange Server は、自身の公開鍵と秘密鍵のペア、およびインストールされているすべてのアダプターの公開鍵が格納された鍵ストアを保持します。個々のアダプターは、自身の公開鍵と秘密鍵のペア、InterChange Server の公開鍵、自身が通信するすべてのプロセスの公開鍵が格納された鍵ストアを保持します。

個々のアダプターおよび InterChange Server について、セキュリティーのレベルと、保護するメッセージのタイプを組み合わせることができます。例えば、InterChange Server からの管理メッセージについて保全性を構成し、SAP アダプターからのビジネス・オブジェクト・メッセージについてプライバシーを構成し、E メール・アダプターからのメッセージにはセキュリティーを設定しないことができます。

メッセージ・タイプ

アダプターまたは InterChnage Server からの保護されるメッセージのタイプは次の 4 つです。

すべてのメッセージ (すべて)
アダプターまたは InterChange Server からのすべてのメッセージが保護されます。

管理メッセージ (管理)
アダプターまたは InterChange Server からのすべての管理メッセージが保護されます。

すべてのビジネス・オブジェクト (BO)
アダプターまたは InterChange Server からのすべてのビジネス・オブジェクト・メッセージが保護されます。

個々のビジネス・オブジェクト (BO 仕様名)
保護するビジネス・オブジェクト仕様の名前を指定することにより、アダプターまたは InterChange Server からの特定のビジネス・オブジェクト・メッセージが保護されます。

エンドツーエンド・プライバシーを使用してメッセージを保護する場合、管理者は、保護するメッセージのタイプおよび必要なセキュリティー・レベルを決定した上で、InterChange Server および個々のアダプターの両方を構成する必要があります。InterChange Server に対してエンドツーエンド・プライバシーを構成した場合、これは InterChange Server から送信されるメッセージにのみ適用されます。個別のアダプターに対してエンドツーエンド・プライバシーを構成した場合、これはそのアダプターから送信されるメッセージにのみ適用されます。エンドツーエンド・プライバシーをオンにする前に、既存のすべてのイベントをシステムから除去する必要があります。既存のイベントは、エンドツーエンド・プライバシーを構成している間は適切に処理されません。エンドツーエンド・プライバシーの構成方法の詳細については、「*システム管理ガイド*」を参照してください。

役割ベースのアクセス制御

無許可ユーザーがシステムにアクセスするのを防止するには、ユーザー ID とパスワードが、基本的なセキュリティー要件に対応します。ユーザー ID はシステムにアクセスする必要のあるユーザー (個人またはプロセス) に対して発行されます。ユーザーを認証することは、そのユーザーがシステムのすべての部分に対してアクセスできなければならないことを意味するわけではありません。システムには、選択されたユーザーのみがアクセスできる重要な、つまり機密性の高い部分と、すべてのユーザーがアクセス可能な部分がある可能性があります。アクセス制御は、認証済みユーザーが誰であるか、およびそのユーザーに付与されている許可が何であるかに基づいて、システムの部分に対するアクセスを制限することを意味します。個々のユーザーに対してさまざまなコンポーネントへのアクセスを許可することが可能ですが、このような方法で許可を管理するのは、管理者にとって時間がかかる作業です。役割ベースのアクセス制御は、InterChange Server へのアクセスを求めるユーザーを認証した上で、ユーザーに割り当てられた役割に基づいて、個々のコンポーネントへのユーザーのアクセスを制限します。

役割

役割は、いくつかの機能的関連を持つ 1 人以上のユーザーの集合です。役割に基づいて許可を割り当てることにより、管理の負担が大幅に削減されます。管理者は、保護される必要のあるそれぞれの操作ごとに、その操作の実行を許可された役割を定義します。許可された役割のメンバーであるユーザーだけが、その操作の実行を

許可されます。役割に許可を割り当てると、その役割許可が役割のセキュリティー・ポリシーとして参照されます。ユーザーには、複数の役割を割り当てることができます。

操作すなわちアクションに役割が割り当てられていない場合、そのアクションは保護されておらず、すべての認証済みユーザーがそのアクションを実行できます。保護されたアクションへのアクセスを許可された役割がユーザーに割り当てられている場合にのみ、ユーザーはそのアクションの実行を許可されます。

InterChange Server には、定義済みの管理者役割がデフォルトの役割として組み込まれています。管理者役割のメンバーは、サーバー上でのすべての操作 (ユーザーの作成、役割の作成、役割への許可の付与など) を実行する許可を持っています。管理者は、組織の構造をより適切に反映する追加の役割を作成することができます。例えば、ある組織では、異なるグループによって開発されたコンポーネントへのアクセスを制限するために、Finance Developers や Redevelopers などといった複数の開発者役割がある可能性があります。ユーザーは、複数の役割に所属することができます。デフォルトの役割は削除できません。

管理者は、システムを照会して、システムに現在ログインしているユーザーを調べることができます。1 人のユーザーは、同時に最大で 20 個のセッションにログインできます。管理者は、現在ログインしている任意のユーザーの特定のセッションをログアウトすることができます。また、管理者は、あるユーザーが開いているすべてのセッションをログアウトすることもできるので、ユーザーがログアウトするのを忘れた場合、管理者がユーザーの代わりにログアウトすることができます。ユーザーがすでに 20 個のセッションを開いている場合、ログインしようとすると例外がスローされます。管理者は、ユーザーのパスワードを再設定することもできます。

ゲスト・ユーザーがデフォルトのユーザーとして事前定義されており、ゲストのパスワードが割り当てられています。デフォルトのユーザーは削除できません。ゲストがログインできるセッションの数に制限はありません。

InterChange Server 管理者は、コンポーネントに割り当てられた許可であるセキュリティー・ポリシーを security.xml としてインポートまたはエクスポートすることができます。適切な役割を持つユーザーは、System Manager およびコマンド行ツール *repos_copy* を使用して、役割定義および役割メンバーシップに関する情報をファイルにエクスポートしたり、ファイルからインポートすることができます。このファイルの名前は、`RoleMembership.xml` です。ユーザーおよびパスワードは、バイナリー・ファイルにエクスポートしたり、バイナリー・ファイルからインポートすることが可能であり、このバイナリー・ファイルは別のサーバーにインポートすることができます。パスワードをエクスポートすることには、セキュリティー上のリスクがあります。このファイルは、オペレーティング・システムのセキュリティーによって保護されているディレクトリーにエクスポートする必要があり、インポート後に削除する必要があります。

デフォルトのセキュリティー・ポリシーでは、管理者役割を持つユーザーのみが、InterChange Server のサーバーの開始と停止、セキュリティーの管理、構成ファイルのエクスポート、および構成ファイルの配置を行うことができます。新規コンポー

ネントに対するデフォルトの許可では、認証済みユーザーは、すべてのアクションを実行できることになっています。デフォルトでは、コンポーネントは保護されていません。

役割ベースのアクセス制御は、InterChange Server のインストール時に自動的には*構成されません*。役割ベースのアクセス制御を構成する方法については、「システム*管理ガイド*」を参照してください。役割ベースのアクセス制御をオンに設定する前に、少なくとも 1 人のユーザーを管理者役割に追加する必要があります。InterChange Server でリポジトリー実装を使用する場合は、ユーザーを作成して管理者役割に追加する必要があります。役割ベースのアクセス制御がオンに設定されている場合、管理者役割に属するユーザーが 1 人もいない状態でサーバーが開始されると、サーバーはエラー・メッセージをログに記録して、役割ベースのアクセス制御をオフにした状態で開始されます。サーバーを開始するためにユーザー ID およびパスワードは必要なく、すべてのユーザーがすべての操作に対してアクセス権を持ちます。役割ベースのアクセス制御がオンに設定されている場合は、サーバーを開始するためにはユーザー ID とパスワードが必要であり、すべての許可またはアクセス・チェックが機能します。

監査

監査により、管理者は、ログイン、ログアウト、ユーザー/役割/許可の管理などのセキュアな操作を実行したユーザーを追跡できるようになります。収集される監査データには、日時、ユーザー ID、実行した操作、および操作の状況 (成功、アクセス否認、例外発生など) があります。監査ログは、通常の Interchangesystem.log ファイルには書き込まれません。これは、セキュアでない *your_ics_dir_name/log/audit directory* ディレクトリーにある別のファイルに書き込まれます。監査ログ・ファイルには、無許可ユーザーがこのファイルを表示できないように、オペレーティング・システムによって保護される必要のある情報が含まれています。ログ・ファイルのサイズおよび新規ログ・ファイルを作成する頻度は、構成可能です。ファイルの名前は、ログ・ファイルがファイル・サイズ設定値を越えて新規ログ・ファイルを作成する必要の生じた頻度に応じて決まります。頻度のオプションには、毎日、毎週、および毎月があります。デフォルト値は毎週です。監査ログ・ファイル名の形式は、以下のようになります。

```
Daily file: icsaudit_yyyymmmdd_nnn.log
Weekly file: icsaudit_yyyymmmwx_nnn.log
    Where x = 1,2,3.... Week numbers
Monthly file : icsaudit_yyyymmm_nnn.log
```

ここで、yyyy は年、mmm は月、dd は日、wx は週番号を表します。nnn は 001.....999 であり、ログ・ファイルがファイル・サイズ設定値を超えたために 1 つの期間内に複数のファイルを作成する必要がある場合に使用されます。監査できる操作には、以下が含まれます。

- ログインおよびログアウト
- サーバーの開始および停止、セキュリティー、役割およびユーザーの管理、モニター、失敗したイベントの表示、MMS - 配置/エクスポート/削除/コンパイル/構成ファイルのエクスポート/構成ファイルの配置
- コラボレーション・オブジェクトの開始/停止/一時停止/シャットダウン/実行 (AccessFramework 呼び出し)、トランザクション状況の解決、失敗したイベントのサブミット、失敗したイベントの削除、LLBP フローの取り消し

- コネクターの開始/停止/一時停止、エージェントのシャットダウン、失敗したイベントのサブミット、失敗したイベントの削除
- マップの開始および停止
- 関係の開始および停止
- ベンチマークの開始および停止

ユーザー・レジストリー

ユーザー・レジストリーは、ユーザー名およびパスワードを保管するシステムです。デフォルトでは、ユーザーおよびパスワードは、ローカルの WBI リポジトリーに保管されます。userRegistry 構成パラメーターを使用して、InterChange Server が Lightweight Directory Access Protocol (LDAP) ディレクトリーをユーザー・レジストリーとして使用するように構成できます。LDAP は、エンタープライズ・ディレクトリー・サービスにアクセスするためのプロトコルです。エンタープライズ・ディレクトリーは、複数のアプリケーション間で共用できる情報を保管するために使用されます。例えば、すべてのエンタープライズ・アプリケーションで必要なユーザー情報には、ユーザー名、パスワード、および E メール・アドレスなどがあります。この情報を個々のアプリケーションの専有データベースに保管し、データを複写する場合、同期を保つのが困難です。新しい従業員が会社に加わったときは、それぞれのアプリケーションでその従業員用のユーザー ID およびパスワードを作成する必要があります。しかし、ディレクトリー・サービスを使用すると、ユーザー ID とパスワードはこのディレクトリーで一度作成するだけで済み、エンタープライズ・アプリケーションはこの情報を再利用できます。

プロバイダーがユーザー・リポジトリーとして LDAP ディレクトリーを使用するようにするにはプロバイダーを LDAP に設定し、ユーザー・レジストリーとしてローカルの WBI リポジトリーを使用するようにするには REPOS に設定します。WBI リポジトリーを使用してユーザーおよびパスワードを保管する場合は、それ専用の個別のデータベースを使用することを強くお勧めします。ユーザー・レジストリーをホスティングするデータベースの URL を指定することができます。これは、構成ファイル interchangesystem.cfg 内の USER_REGISTRY という名前の階層プロパティーです。構造は既存の REPOSITORY プロパティーと同じであり、データベースの URL、ユーザー名、およびパスワードを指定できます。このプロパティーは、サーバーの構成ツールを使用して編集できます。LDAP を使用する場合も WBI データベースを使用する場合も、1 つのエンタープライズごとに 1 つのユーザー・レジストリーを設定し、InterChange Server の複数のインスタンスがユーザーやパスワードをインポートおよびエクスポートする必要なしに同じユーザー・レジストリーを共用できるようにすることをお勧めします。InterChange Server は、RFC 2798 に記述されている InetOrgPerson スキーマを使用して、LDAP ディレクトリーのユーザー名およびパスワードを使用できます。InetOrgPerson は、各スキーマを固有に識別するために IETF によって割り当てられたオブジェクト ID (OID) 2.16.840.1.113730.3.2.2 を持っています。InterChange Server は、ディレクトリー内のディレクトリー・ツリー構造について何の推測も行いません。ユーザーは、ユーザーおよび役割の両方の構成の一部として基本識別名 (baseDN) を指定できます。この baseDN は、すべての検索および更新のルートとして使用されます。

コンポーネントおよびアクションの保護

保護可能なコンポーネントおよびアクションのリストを以下に示します。

表 2. 保護可能なコンポーネント

保護可能なコンポーネント	保護可能なアクション
InterChange Server	• 開始 • シャットダウン • セキュリティー - 役割およびユーザーの管理 • モニター • 失敗したイベントの表示 • MMS - 配置 • MMS - エクスポート • MMS - 削除 • MMS - コンパイル • MMS - 構成ファイルのエクスポート • MMS - 構成ファイルの配置
コラボレーション・オブジェクト	• 開始 • 停止 • 一時停止 • シャットダウン • 実行 (Access Framework 呼び出し) • トランザクション状況の解決 • 失敗したイベントのサブミット • 失敗したイベントの削除 • LLBP フローの取り消し
コネクター	• 開始 • 停止 • 一時停止 • エージェントのシャットダウン • 失敗したフローのサブミット • 失敗したフローの削除
マップ	• 開始 • 停止
関係	• 開始 • 停止
ベンチマーク	• 開始 • 停止

要約

この章では、IBM WebSphere InterChange Server システムの主要なコンポーネントについて説明しました。覚えておくべき重要な点は次のとおりです。

- コラボレーションはコネクターおよびサーバー・アクセス・インターフェースと対話して、分散ビジネス・プロセスを作成します。コラボレーションには、ビジネス・プロセス・ロジックが含まれます。コネクターは IBM WebSphere InterChange Server システムとアプリケーション環境をブリッジして、アプリケーション全体にビジネス・プロセス・ロジックを実装できるようにします。サーバー・アクセス・インターフェースと XML コネクターを使用すると、ビジネス・プロセス・ロジックをインターネット上に拡張できます。
- コラボレーションを含むすべての対話は、ビジネス・オブジェクトの交換によって実行します。サーバー・アクセス・インターフェースが呼び出しを受信するか、コネクターがアプリケーション・イベントについての情報を受信すると、ビジネス・オブジェクトが作成され、コラボレーションに送信されます。
- ビジネス・オブジェクトには、汎用とアプリケーション固有があります。コラボレーションは汎用ビジネス・オブジェクトを処理し、コネクターはアプリケーション固有のビジネス・オブジェクトを処理します。
- InterChange Server は、コラボレーション実行の中枢です。InterChange Server には、リポジトリー、コネクター・コントローラー、イベント管理サービス、データベース接続サービス、およびトランザクション・サービスが含まれます。
- System Manager は、各 InterChange Server に保管されているオブジェクトの表示、構成、および実行の制御を行います。
- コンポーネントは分散しています。コラボレーション/コネクター・プロトコル、メッセージング、および ORB によって、コンポーネント間が結ばれます。
- InterChange Server のメッセージは、エンドツーエンド・プライバシーを使用して保護することができます。InterChange Server は、役割ベースのアクセス制御を使用して保護することができます。

次の章では、コラボレーションについて詳しく説明します。

第 2 章 InterChange Server で使用するツール

InterChange Server で使用できるのは、以下の Toolset です。

- 『WebSphere Business Integration Toolset』
- 47 ページの『開発ツール』
- 48 ページの『管理ツール』

WebSphere Business Integration Toolset

WebSphere Business Integration Toolset は、InterChange Server とともに使用できる管理および開発ツールを提供します。WebSphere Business Integration Toolset は、Windows 2000 および Windows XP でのみサポートされます。管理ツールには、以下が含まれます。

- System Manager
- System Monitor
- Flow Manager
- Log Viewer
- Relationship Manager

管理ツールと管理タスクの詳細については、「システム*管理ガイド*」を参照してください。

開発ツールには、以下が含まれます。

- Map Designer
- Relationship Manager
- Process Designer
- Business Object Designer
- Connector Configurator

これらの開発ツールの詳細は、「*WebSphere InterChange Server システム・インプリメンテーション・ガイド*」および各種コンポーネントの開発ガイドを参照してください。

WebSphere Business Integration Toolset ツールは Windows ショートカットを介してアクセス可能であり、このためには、「**スタート**」 >「**プログラム**」>「**WebSphere Business Integration Toolset**」を選択してから、管理ツールの場合は「**管理**」を、開発ツールの場合は「**開発**」を選択します。

IBM WebSphere Business Integration Toolset は、IBM Tivoli License Manager (ITLM) とともに使用できるように対応しています。これは、インベントリー機能のサポートを提供します。ITLM を使用すると、ソフトウェアの利用料に関してさらなる制御および柔軟性が提供され、顧客はマシンの全容量ではなく、ソフトウェアの使用率に基づいてソフトウェアの対価を支払うことができます。ILTM は製品の使用量を測定し、これを IBM に報告します。以下の機能が含まれます。

- インベントリー・データの収集、並びに各サイトで使用およびインストールされている製品に関する情報の抽出。
- さまざまな顧客または独立した組織によって購入された製品のソフトウェア調達および契約情報、並びにライセンス条件に関する情報の保管と保守。
- 調達されているライセンス、インストールされているライセンス、および使用されているライセンスの比較と、これらの 3 つの異なる視点から派生した情報の調整。
- 課金およびインベントリー結果を編集して、各製品に必要な同梱のライセンスを識別して必要な数のライセンスを調達すること。
- ソフトウェア製品が顧客とベンダーとの間の契約に従って使用されていることの監視。

System Manager

System Manager インターフェースは Eclipse プラットフォームに基づいています。Eclipse プラットフォームは、ツール作成用のオープン・ソースの統合開発環境です。Eclipse プラットフォームは、ユーザーが特定のタイプのリソースで作業できるプラグインをツール開発者が作成するための開発キットとランタイムを提供します。

注: System Manager は、Eclipse プラットフォームの IBM ブランド・バージョンで使用できるプラグインとしてインストールされます。

System Manager には Eclipse ベースのパースペクティブ としてアクセスします。System Manager パースペクティブは、システムの使用やツールへのアクセス全般に役立つビューやエディターを提供し、さらに、開発する統合プロジェクトを展開するためのインターフェースも提供します。このパースペクティブには、以下のタスクのサポートが含まれます。

- IBM WebSphere InterChange Server 開発ツールにアクセスする
- IBM WebSphere InterChange Server システムで使用される個々のモジュラー・コンポーネントに対する構成タスクを実行する
- InterChange Server のインスタンスを処理する
- 統合コンポーネントのグループをユーザー・プロジェクトとして処理し、コンポーネントをリポジトリーに展開する
- Business Integration システムとそのコンポーネントのパフォーマンス特性のベンチマーク試験を実行する

System Manager のタスクにアクセスするには、メニューおよび対応するツールバーを使用します。WebSphere Business Integration System Manager ビューを使用することもできます。System Manager の左端のパネルに表示される WebSphere Business Integration System Manager ビューには、モジュラー統合コンポーネントのライブラリーのフォルダー形式のリスト (ビジネス・オブジェクト定義、データベース接続プール、およびコラボレーション・テンプレートとオブジェクトなど) が表示されます。開発および構成タスクを実行するには、ライブラリー内のコンポーネントのアイコンを右クリックします。

System Manager からコラボレーション・テンプレート、ビジネス・オブジェクト定義、マップ、および関係を作成するための GUI ツールにアクセスできます。一部の開発ツールは System Manager を介さずにアクセスおよび実行できます。

使用可能な開発ツールの一部のリストについては、47 ページの『開発ツール』を参照してください。

コンポーネント構成

System Manager を使用してモジュラー IBM WebSphere InterChange Server コンポーネント (**統合コンポーネント**と呼ばれます) を操作できます。このコンポーネントは IBM WebSphere InterChange Server 環境でビジネス・データを交換するために使用されます。これらのコンポーネントには、コラボレーション、コネクター、ビジネス・オブジェクト、マップ、関係、およびデータベース接続プールがあります。System Manager から、コンポーネントに関連した以下の構成タスクを実行できます。

- コラボレーション・テンプレートからコラボレーション・オブジェクトを構成する。ポート・バインディングおよび InterChange Server での実行時に使用される一般プロパティーなど。
- 標準コネクター・プロパティーおよび特殊コネクター・プロパティーのリポジトリー定義を表示および改訂し、コネクターが使用するビジネス・オブジェクトとマップを指定するためのツールにアクセスする。

System Manager および InterChange Server モード

統合コンポーネントの開発時には、System Manager のサーバー・モード選択機能を使用して、作成したコンポーネント定義および構成をリポジトリーにロードする形式と範囲を制御できます。サーバー・モード選択機能は、開発の設計、テスト、および実動の各段階で使用できます。サーバー・モードの使用に関する詳細は、「*WebSphere InterChange Server インプリメンテーション・ガイド*」を参照してください。

開発ツール

コラボレーション、コネクター、ビジネス・オブジェクト、マップ、および関係を作成および変更することができます。開発ツールは、システムが実働状態になる前に開発環境でシステム・コンポーネントの作成と構成に使用されます。これらのツールの詳細は、「*WebSphere InterChange Server インプリメンテーション・ガイド*」および各種コンポーネントの開発ガイドを参照してください。表 3 には、開発環境で使用可能なソフトウェア・ツールの一部を説明とともにリストしています。

表 3. 開発ツール

対象読者	ツール	説明
コラボレーション開発者	Process Designer	コラボレーション・テンプレートの定義に使用するグラフィック・ツール。 Process Designer の詳細、およびコラボレーションの開発方法については、「*コラボレーション開発ガイド*」を参照してください。

表 3. 開発ツール (続き)

対象読者	ツール	説明
コラボレーションおよびマップ開発者	Test Environment および Virtual Test Connector	開発したビジネス・インテグレーション・インターフェースをテストする環境を提供します。コネクターのエミュレーション、必要なコンポーネントの開始、ビジネス・オブジェクト・データの検査を実行するためのグラフィカル・インターフェースを提供します。Test Environment の詳細、およびコラボレーション、マップ、コネクターのテスト方法については、「*WebSphere InterChange Server インプリメンテーション・ガイド*」を参照してください。
コネクター開発者	Connector Configurator	アプリケーション固有のプロパティーをコネクター定義に追加し、プロパティー値を設定し、ビジネス・オブジェクトとマップにコネクター定義を構成するために使用します。 Connector Configurator の詳細、およびコネクターの開発方法については、「*コネクター開発ガイド (Java 用)*」または「*コネクター開発ガイド (C++ 用)*」を参照してください。
マップ開発者	Map Designer	データ形式変更を指定するグラフィック・ツール。Map Designer の詳細、およびマップの開発方法については、「*マップ開発ガイド*」を参照してください。
マップ開発者	Relationship Designer	複数タイプのオブジェクト間の関係を定義するグラフィック・ツール。これらの関係はマッピングにおいて重要です。例えば、あるビジネス・オブジェクトと異なるタイプのビジネス・オブジェクト間の関係を指定することが必要です。Relationship Designer の詳細、および関係の開発方法については、「*マップ開発ガイド*」を参照してください。
ビジネス・オブジェクト開発者	Business Object Designer	ビジネス・オブジェクト定義の作成 (手動または Object Discovery Agents (ODA) を使用した作成) に使用するフォーム・ベースのインターフェース。Business Object Designer の詳細、およびビジネス・オブジェクト定義の設計方法については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。
ビジネス・オブジェクト開発者	Object Discovery Agent Development Kit (ODK)	Object Discovery Agent (ODA) の作成を容易にする API。ODA はデータ・ソースに固有のビジネス・オブジェクト要件を識別し、これらの要件に基づいて定義を生成します。Business Object Designer の詳細、および Object Discovery Agent (ODA) の開発方法については、「*ビジネス・オブジェクト開発ガイド*」を参照してください。
すべての開発者	ベンチマーク・ウィザード	ベンチマーク・ツールを使用すると、さまざまな IBM WebSphere コンポーネント、インターフェース、およびシステムをテストしてスループットを測定できます。ベンチマーク・ツールは System Manager を使用してアクセスできます。ベンチマーク・ウィザードの詳細については、「*Benchmarking Guide*」を参照してください。

管理ツール

System Manager から一部の管理ツールにアクセスできます。System Monitor にアクセスして統合コンポーネントの状態を管理することもできます。管理ツールと管理タスクの詳細については、「*システム管理ガイド*」を参照してください。

第 3 章 コラボレーション

InterChange Server 環境では、**コラボレーション**という用語は、コードと複数のアプリケーション間の対話を促進するビジネス・プロセス・ロジックを含むソフトウェア・モジュールを指します。単純なコラボレーションはいくつかのステップのみから構成され、複雑なコラボレーションはいくつかのステップと他のコラボレーションから構成されます。

コラボレーションは、複数のアプリケーションにわたって分散させることができ、同期および非同期のサービス呼び出しの処理を行います。また、コラボレーションは、長期存続ビジネス・プロセスをサポート可能です。この章では、コラボレーションの概要、構成、およびコラボレーションとの対話方法について説明します。

この章では、コラボレーション、およびコラボレーションの表示、変更、作成に使用するコンポーネントの概要について説明します。この章は次のセクションから構成されます。

IBM WebSphere InterChange Server コラボレーション・ツールを使用したコラボレーション・テンプレートの作成方法の詳細については、「*コラボレーション開発ガイド*」を参照してください。

コラボレーションのテンプレートとオブジェクト

コラボレーションをインストールするときは、**コラボレーション・テンプレート**をインストールします。コラボレーション・テンプレートにはコラボレーションのすべての実行ロジックが含まれますが、テンプレートを実行することはできません。コラボレーションを実行するためには、まずテンプレートから**コラボレーション・オブジェクト**を作成する必要があります。コラボレーション・オブジェクトは、コネクターまたは他のコラボレーション・オブジェクトにバインドし、その他の構成プロパティーを指定することによって構成すると、実行可能になります。

注: InterChange Server で使用可能な Toolset には、**Process Designer** と呼ばれるコラボレーション設計ツールがあります。このツールを使用すると、コラボレーション・テンプレートをカスタマイズおよび作成できます。

1 つのコラボレーション・テンプレートを使用して、複数のコラボレーション・オブジェクトを作成することができます。例えば、Contact Manager コラボレーションを使用して、フロント・オフィスとバック・オフィスのアプリケーションを統合するとします。企業内でフロント・オフィス・アプリケーションに Clarify を使用している部門と Siebel を使用している部門があり、両部門ともバック・オフィス・アプリケーションに SAP を使用している場合、ContactManager コラボレーション・テンプレートから 2 つのコラボレーション・オブジェクトを作成できます。

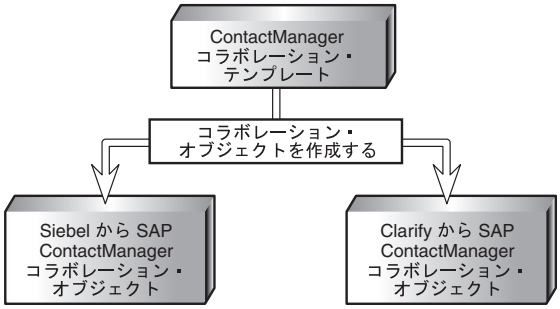

図 17. テンプレートからのコラボレーション・オブジェクトの作成

これで、2 つのコラボレーション・オブジェクトがそれぞれのソース・アプリケーションのコネクターにバインドされていながら、宛先アプリケーションの同一コネクターにバインドされるように構成できます。

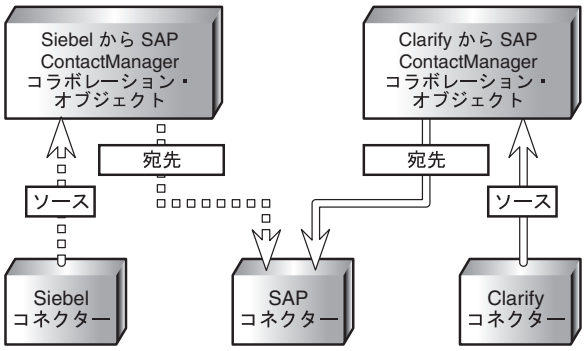

図 18. コラボレーション・オブジェクトの構成

注: 本書では、動作、設計、および機能を説明するときに「コラボレーション」という用語を使用します。「コラボレーション・テンプレート」および「コラボレーション・オブジェクト」という用語は、通常、構成プロセスまたはリポジトリーに関連して区別が必要な場合にのみ使用します。

コラボレーション処理

コラボレーションは、ビジネス・オブジェクトを受け取ることによって起動されると、処理フローを開始します。トリガーとなるのは以下のいずれかです。

- パブリッシュ・アンド・サブスクライブ対話において、コネクターによってパブリッシュされたイベントから受け取るビジネス・オブジェクト (第 1 章で説明)

- サーバー・アクセス・インターフェースを介して受け取るアクセス要求 (第 1 章で説明)
- コネクターから受け取るサービス呼び出し (この章で説明)

サービス呼び出し処理および長期存続ビジネス・プロセス

コラボレーションは、サービス呼び出しデータ・フローの内容を指定された期間保持できるため、サービス呼び出しプロセスを同期的に待機させることなくコネクターと対話できます。指定された期間にサービス呼び出しに対する適切な応答を受け取ると、データ・フローを再開し、処理を続行します。この機能はコラボレーションの**長期存続ビジネス・プロセス**と呼ばれ、コラボレーション・テンプレートの作成時に使用可能にすることができます。

InterChange Server 実装におけるコラボレーションは、以下のタイプのサービス呼び出しをサポートできます。

- コラボレーションからの同期アウトバウンド
- コラボレーションからの非同期アウトバウンド
- コラボレーションへの非同期インバウンド

同期アウトバウンド・サービス呼び出し

同期アウトバウンド・サービス呼び出しでは、同期要求/応答機構が使用されます。このサービス呼び出しは、要求を送信した後に、応答が届いて処理されるまでプロセスを完了できません。

同期サービス呼び出しでは、差し戻しがサポートされます。また、長期存続ビジネス・プロセスのタイムアウト値もサポートされます。

非同期アウトバウンド・サービス呼び出し

非同期アウトバウンド・サービス呼び出しでは、コラボレーションは要求を送信しますが、応答を受け取るまで待機せずに処理を継続できます。

非同期アウトバウンド・サービス呼び出しでは、差し戻しはサポートされますが、長期存続ビジネス・プロセスのタイムアウト値はサポートされません。

非同期インバウンド・サービス呼び出し

非同期インバウンド・サービス呼び出しでは、コラボレーションは受信イベントを受け取るまで待機し、このサービス呼び出しを長期存続ビジネス・プロセスとともに使用します。非同期インバウンド・サービス呼び出しを作成すると、タイムアウト値を取得します。このサービス呼び出しがタイムアウトの期限が切れる前に受信イベントを受け取らなかった場合は、例外が発生します。

非同期インバウンド・サービス呼び出しは、コラボレーション・テンプレートの「長期存続ビジネス・プロセスのサポート」オプションを使用可能にしている場合にのみ使用できます。この機能は、コネクターとのデータ交換のみに使用されます。

非同期インバウンド・サービス呼び出しでは、差し戻しはサポートされません。

コラボレーションと並行処理

コラボレーションは、ビジネス・オブジェクトを受け取ることによって起動されると、処理フローを開始します。デフォルトでは、コラボレーションは 1 つのフローの処理が完了してから、次のトリガー・ビジネス・オブジェクトのフローを処理します。

ただし、イベントにより起動された複数のフローを並行して処理するようにコラボレーションを構成することができます。このようにコラボレーションを構成すると、コラボレーションはビジネス・オブジェクト依存関係を持つフローを識別し順番に処理しますが、競合しないフローについては並行処理を許可します。

コラボレーション・グループ

コラボレーション・オブジェクトは、別のコラボレーション・オブジェクトにバインドさせて**コラボレーション・グループ**を形成することができます。コラボレーション・グループは、離散した複数のコラボレーションの能力を利用して、関連するビジネス・プロセスを統合します。

コラボレーション・グループ内のコラボレーション間の関係は、1 つのコラボレーションが他のコラボレーションにビジネス・オブジェクトを提供しているという事実によってのみ制限されます。コラボレーション・グループは、チェーンや Web など任意の数のトポロジーで接続されます。

コラボレーション・グループは、データ分離機能の提供にも役立ちます。詳細は、「*コラボレーション開発ガイド*」を参照してください。

ポート

コラボレーションは、**ポート**と呼ばれる外部へのインターフェースのセットを持っています。コラボレーション・テンプレートでは、各ポートはコラボレーション・オブジェクトが実行時に受信または作成するビジネス・オブジェクトを表す変数です。例えば、顧客サービス管理アプリケーションにおいてある事象 (case) の終結によって起動される、仮想の Service Billing コラボレーションを考えてみましょう。このコラボレーションは、顧客レコード・アプリケーションから顧客の契約 (Contract) をビジネス・オブジェクトとして取り出し、契約にある情報を使用して会計アプリケーションに送り状 (Invoice) ビジネス・オブジェクトを送信します。

このようなコラボレーションは、対話するコネクターに 1 つずつ、計 3 つのポートを使用します。図 19 に示すように、各ポートは特定のタイプのビジネス・オブジェクトに関連付けられています。

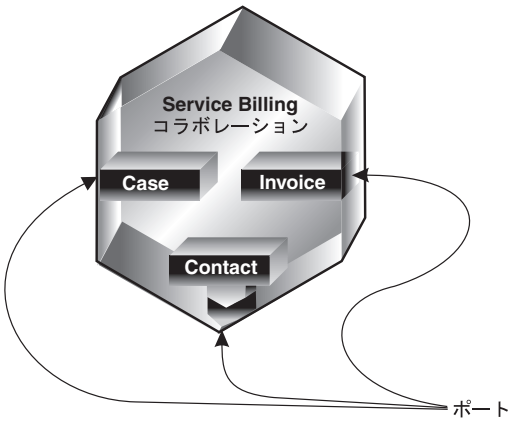

図 19. Service Billing コラボレーションのポート

構成時に、管理者はテンプレートのポートを含むコラボレーション・オブジェクトを作成します。管理者は、その特定のコラボレーション・オブジェクトのポートを**バインド**して、各ポートをコネクターまたは別のコラボレーション・オブジェクトに関連付けます。 Retrieve 動詞を受け入れるテンプレートを持つコラボレーションでは、ポートを*外部* として構成できるので、トリガー・ビジネス・オブジェクトをアクセス要求 (サーバー・アクセス・インターフェースの外部呼び出し) から受信できます。

ポートはバインドされたエンティティー間の通信を可能にするので、コラボレーション・オブジェクトはビジネス・プロセスを起動するビジネス・オブジェクトを受信でき、また要求および応答としてビジネス・オブジェクトを送受信できます。

図 20 に示す仮想コラボレーションでは、起動するポートが Case ビジネス・オブジェクトを外部プロセスからの呼び出しとして受信するようバインドされ、他のポートは Contract ビジネス・オブジェクトおよび Invoice ビジネス・オブジェクトを使用するコネクターにバインドされています。

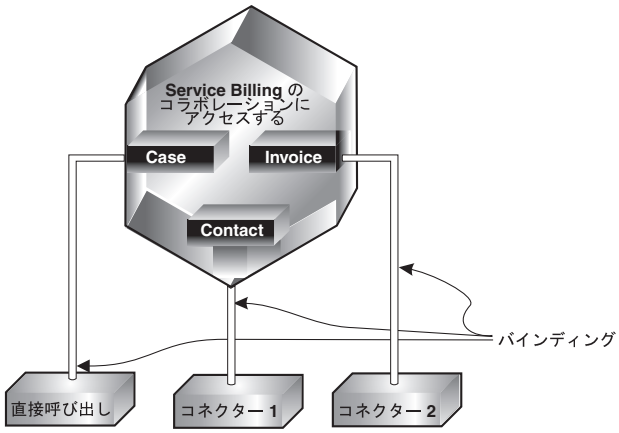

図 20. コネクターにバインドされているポート

例えば、コンピューターの場合はさまざまなハードウェア・インターフェースがあります。コンピューターにスピーカーやモニターなどの周辺装置を接続するには、コンピューターのインターフェースの形状に合うケーブルやワイヤーを接続しま

す。コラボレーション・ポートの「形状」とは、ビジネス・オブジェクトが通過できるタイプを指します。コンピューター・ケーブルが形状の合うインターフェースにしか接続できないように、管理者がポートをバインドできるのは、特定のタイプのビジネス・オブジェクトをサポートするコネクターに限られます。

同一タイプのビジネス・オブジェクトをサポートするポートを持つ 2 つのコラボレーションは、バインドできます。図 21 は、Item ビジネス・オブジェクトをサポートするポートでバインドされた 2 つのコラボレーション・オブジェクトを示しています。

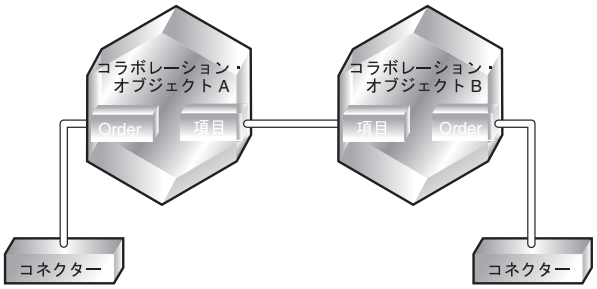

図 21. 2 つのコラボレーション・オブジェクトのバインド

ポート名は、ポートでサポートされる、コラボレーションのビジネス・オブジェクトの役割を示すことがよくあります。例えば、コラボレーションが顧客情報を受信して契約情報を生成する場合、そのポートは「InCustomer and OutContract」と呼ぶことができます。

動的サービス呼び出し

コラボレーションは、明示的バインディングによりサービス呼び出しをインプリメントします。これは、コラボレーション・テンプレートからコラボレーションを開発する際に、既知の宛先 (コネクターや別のコラボレーションなど) に接続するようにコラボレーションのサービス呼び出しを構成する必要があることを意味します。明示的バインディングは静的であり、これは、ポートの割り当てが変更された場合、コラボレーション・オブジェクトを使用できるようにするには、コラボレーション・オブジェクトを InterChange Server に再配置する必要があることを意味します。また、コラボレーションは、すべてのポートがバインドされるまで配置できません。

InterChange Server には、動的サービス呼び出しを使用可能にする API が組み込まれています。これにより、コラボレーションは、コラボレーション開発時に事前定義されなかった、明示的にバインドされていない宛先に対するサービス呼び出しを行うことができます。動的サービス呼び出し API はコンシューマーに関する必要な詳細情報をすべて取得するため、コラボレーションは多数のコネクターと相互作用することが可能であり、コネクターの数とタイプは動的に変更することができます。動的サービス呼び出しを受信するためのコラボレーションを作成するときは、System Manager ツールでポートをバインドする際にインバウンド・ポートを外部ポートとしてバインドし、コラボレーションが外部ポートを介してトリガー・ビジネス・オブジェクトを受信できるようにする必要があります。API は、クライアントの名前やタイプなど、クライアントに関する基本的な情報を渡すことができます。

また、API は、ソース・アダプター情報を使用して、サービス要求の送信先を判別することもできます。動的サービス呼び出し API は同期です。

動的サービス呼び出し API を使用すると、ツールの必要なしに動的サービス・バインディングが可能になります。API および関連するパラメーター・セットをコラボレーション・テンプレート内で使用する場合、既存のコラボレーションに対する変更と影響は最小限ですみます。API 自体は、例外を処理したり、障害時の振る舞いを定義することはないので、これらの問題は、コラボレーション・テンプレートで処理する必要があります。また、API は、トランザクション・サポートの補正などのサービス品質を提供しません。API の詳細については、「*コラボレーション開発ガイド*」を参照してください。

動的サービス呼び出しでは、以下のサポートは制限されるか、またはサポートされません。

- あるコラボレーションから別のコラボレーションへの動的サービス呼び出しでは、2 つのコラボレーション間に明示的ポート・バインディングがないため、コラボレーション・グループは形成されません。
- 動的サービス呼び出しは補正をサポートしないため、動的サービス呼び出しのロールバック・セマンティクスはありません。
- LLBP コラボレーションは、コネクターに対する動的サービス呼び出しを行うことのみ可能であり、トランスポート・タイプとして JMS を使用する必要があります。LLBP コラボレーションは外部ポートにバインドすることができないため、動的サービス呼び出しを受信できません。

シナリオ

コラボレーション内では、1 つ以上の**シナリオ**に、ビジネス・プロセスを実装する「コード」が含まれます。各シナリオは、特定のタイプのイベントを表す特定のタイプのビジネス・オブジェクトを受信したときに応答する内容を指定します。シナリオには、コラボレーションの処理作業がすべて含まれます。

コラボレーションとシナリオの関係は、従来のプログラムとルーチンの関係に似ています。プログラマーは、いくつもの方法でプログラムのロジックをルーチンによって、分解することができます。例えばプログラムには、さまざまな入力引き数、またはそれぞれが異なる入力引き数を処理する複数のルーチンを含めることができます。

同様に、コラボレーション開発者は 1 つまたは複数のシナリオを使用して、コラボレーションの作業を実行できます。例えば、従業員を同期化させるコラボレーションは、次のどちらの方法でも設計できます。

- Employee.Create、Employee.Delete、および Employee.Update のイベント通知を 1 つのシナリオが処理する。
- Employee.Create、Employee.Delete、および Employee.Update をそれぞれ別のシナリオが処理する。

図 22 に、これら 2 つの方法を示します。

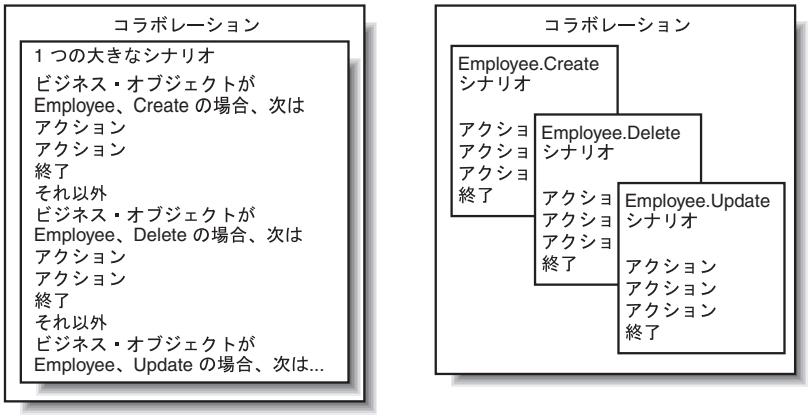

図 22. コラボレーションのシナリオ

シナリオはそれぞれ個別に実行されますが、個々のシナリオを個別に構成および管理する必要はありません。コラボレーションを構成すると、コラボレーションのすべてのシナリオがその設定を継承します。

ビジネス・プロセス・ロジック

シナリオには、コラボレーションのビジネス・プロセス・ロジックが含まれます。非常に単純な例として、図 23 はある仮想シナリオのロジックを示しています。コラボレーションは、変更を宛先アプリケーションに移動させることによって、ソース・アプリケーション内の従業員情報の変更を複製します。このシナリオは、Create 動詞を明確に処理します。

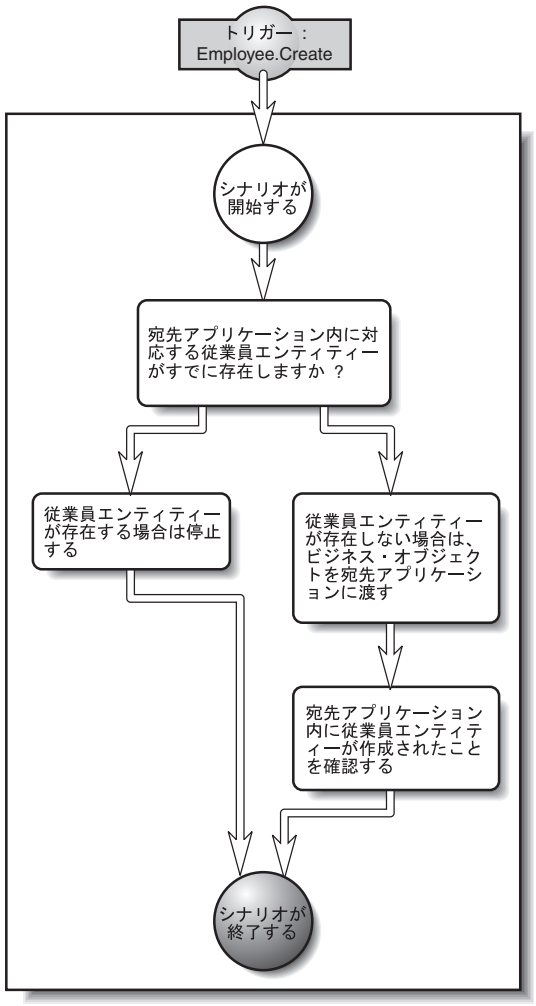

図 23. シナリオのハイレベル表示

図 23 では、シナリオのトリガーは Employee.Create イベント通知ビジネス・オブジェクトです。Employee.Create ビジネス・オブジェクトが受信されると、シナリオが開始します。まず、シナリオは従業員情報が宛先アプリケーションにすでに存在するかどうかをチェックします。

従業員情報がすでに宛先アプリケーションに存在する場合、シナリオは停止します。従業員情報がまだ存在しない場合、シナリオは Employee.Create 要求ビジネス・オブジェクトを宛先アプリケーションのコネクターに渡します。検証ステップにより、操作が正常に終了したことが確認されます。

実際のシナリオのビジネス・プロセス・ロジックは、通常はさらに多くのステップから構成され、それぞれのステップがコラボレーションによるアクションを実装しています。次に挙げるのは、コラボレーションに含まれるアクションのタイプの例です。

- 受信するビジネス・オブジェクトのタイプ、属性値、または動詞を取得する。
- 値なしで、または既存のビジネス・オブジェクトを複製して、新規のビジネス・オブジェクトを作成する。
- 属性値を定数または別の属性値と比較する。

- 2 つのビジネス・オブジェクトを比較して、等しいかどうか調べる。
- ビジネス・オブジェクトをコネクターまたは別のコラボレーションに送信して、操作を要求する (操作結果の処理)。
- コラボレーションの構成可能なプロパティーの値を 1 つ取得する。
- 情報メッセージ、警告メッセージ、またはエラー・メッセージを記録する。

コネクターとアプリケーションとの対話

コラボレーションはポートを介して、コネクター、アクセス・インターフェース、および他のコラボレーションと対話します。コラボレーションはポートを介して、トリガーの受信、要求の送信、および応答の取得を行います。これらはアプリケーション操作との間で変換されます。

図 24 には、図 23 で示したシナリオを簡単な例として使用し、コラボレーションがパブリッシュ・アンド・サブスクライブ対話によって操作を進める方法を示します。図 24 では、宛先アプリケーションに従業員情報が含まれない場合にとられる単一の実行経路のみが示されています。図 24 のグレーの矢印パスは、コネクター B がコラボレーション要求をアプリケーション API への要求に転送し、アプリケーション応答をコネクター B 自身のコラボレーションへの応答に転送するときに、コネクター B が行う変換を示します。

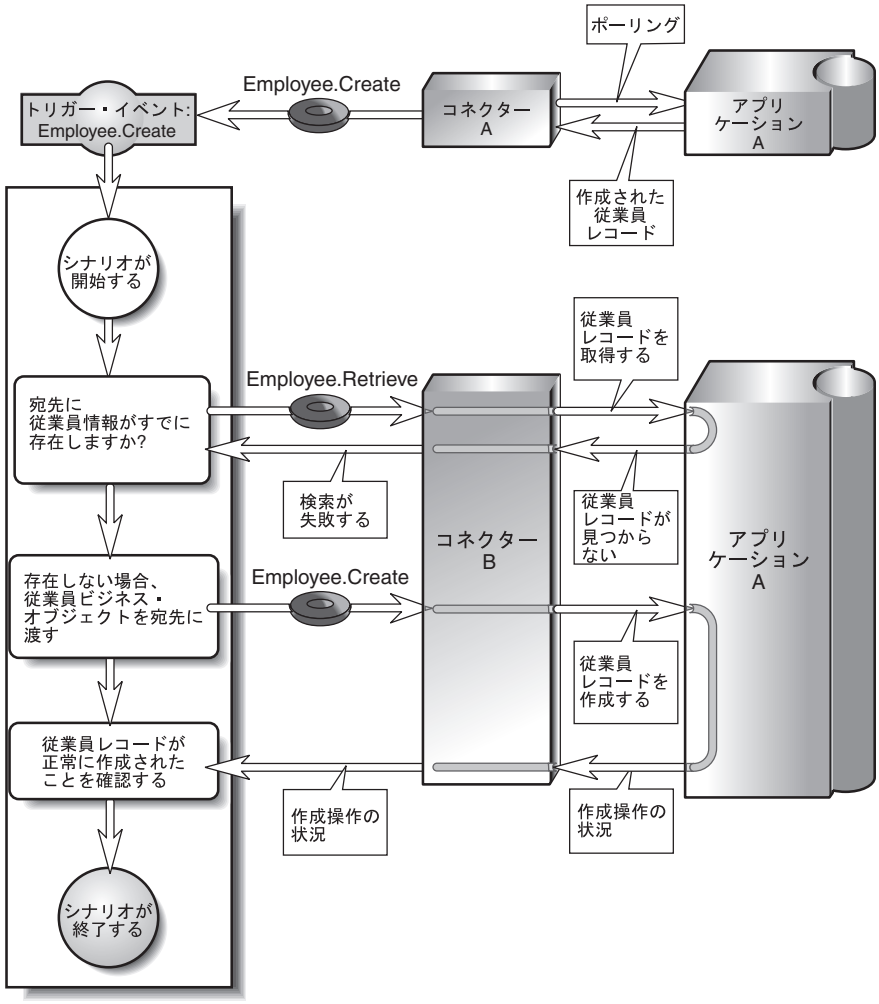

図 24. 実行時の処理

この図で、コネクターとはコネクター・エージェントおよびコネクター・コントローラー機能の両方を表します。

図 24 には、次のプロセスを示します。

1. アプリケーション A が新規の従業員レコードを作成すると、コネクター A がそれを検索します。
2. コネクター A は Employee.Create ビジネス・オブジェクトを作成し、このイベント通知をコラボレーションに送信します。これがコラボレーションで受信されると、シナリオが開始されます。
3. シナリオは、宛先アプリケーションで同一の従業員情報を検索することによって、従業員情報が新規のものかどうかを調べます。この検索のために、シナリオは Employee.Retrieve 要求をコネクター B に送信します。
4. コネクター B は、従業員情報の検索を試みますが、その従業員がアプリケーション B に存在しないことがわかります。コネクターはコラボレーションに障害状況を戻します。
5. シナリオは Employee.Create 要求の送信を開始します。コネクター B が Employee.Create 要求を受信すると、アプリケーション B の API を使用して新規の従業員を作成します。

6. コネクター B が操作の成功を示す成功状況を戻します。
7. シナリオが成功状況を受信し、終了します。

コラボレーションの開始

設計時に、コラボレーション開発者は、コラボレーションで受信されると各シナリオが実行されるビジネス・オブジェクトを指定します。コラボレーションを構成して使用可能にすると、コラボレーションが始動します。コラボレーションは、シナリオを開始するすべてのビジネス・オブジェクトにサブスクライブし、待機します。

コネクター・エージェントが、サブスクライブされたビジネス・オブジェクトの 1 つにコントローラーを送信すると、次のようになります。

1. コネクター・コントローラーがビジネス・オブジェクトをコラボレーションに渡します。
2. コラボレーションがアプリケーション・シナリオを呼び出し、実行を開始します。

コラボレーションを構成して使用可能にすると、明示的に使用不可にしない限り、再度使用可能にする必要はありません。InterChange Server がシャットダウンして再始動する場合、直前にアクティブだったすべてのコラボレーションも再始動されます。

要約

この章では、コラボレーションのいくつかの機能について説明しました。覚えておくべき重要な点は次のとおりです。

- コラボレーション・テンプレートからコラボレーション・オブジェクトを作成し、それをコネクターおよび他のコラボレーション・オブジェクトにバインドして、実行できるようにします。
- コラボレーションは、ポートを介して間接的にビジネス・オブジェクトを表します。コラボレーション・オブジェクトは、各ポートをビジネス・オブジェクト定義およびコネクター、別のコラボレーション・オブジェクト、または外部呼び出しにバインドすることにより構成します。
- シナリオによって、コラボレーションのビジネス・プロセス・ロジックが実装されます。シナリオは、ビジネス・オブジェクトを受信すると実行される、イベント・ハンドラーです。シナリオが実行されると、ビジネス・オブジェクトがコネクターに送信され、コネクターからビジネス・オブジェクトが受信されます。
- コラボレーションを実行可能にするには、コラボレーションを構成および始動します。コラボレーションは、シナリオを起動できるタイプのビジネス・オブジェクトにサブスクライブします。サブスクライブされたビジネス・オブジェクトを受信すると、コラボレーションは適切なシナリオを実行します。

次の章では、ビジネス・オブジェクトについて詳しく説明します。

第 4 章 ビジネス・オブジェクト

ビジネス・オブジェクトは、データおよびアクション要求とともにコラボレーションとコネクターの間を移動して、ビジネス・プロセスの内容をアプリケーションから別のアプリケーションに伝えます。この章では、ビジネス・オブジェクトの構造およびコンポーネントについて説明します。この章は次のセクションから構成されます。

ビジネス・オブジェクト定義とビジネス・オブジェクト

1 ページの『第 1 章 IBM WebSphere InterChange Server の概要』では、ビジネス・オブジェクト定義とビジネス・オブジェクトのインスタンスを区別せずに、ビジネス・オブジェクトについて説明しました。ここでは、その相違点を説明します。

- **ビジネス・オブジェクト定義**は、IBM WebSphere InterChange Server が取り扱う各エンティティーの情報のタイプと配列、およびサポートする動詞を指定します。ビジネス・オブジェクト定義は、InterChange Server リポジトリーに格納されます。
- **ビジネス・オブジェクト**とは、実際のデータを含んだ、定義のインスタンスのことです。ビジネス・オブジェクトは実行時に作成され、リポジトリーには格納されません。

図 25 に、ビジネス・オブジェクト定義とビジネス・オブジェクト間の関係を示します。

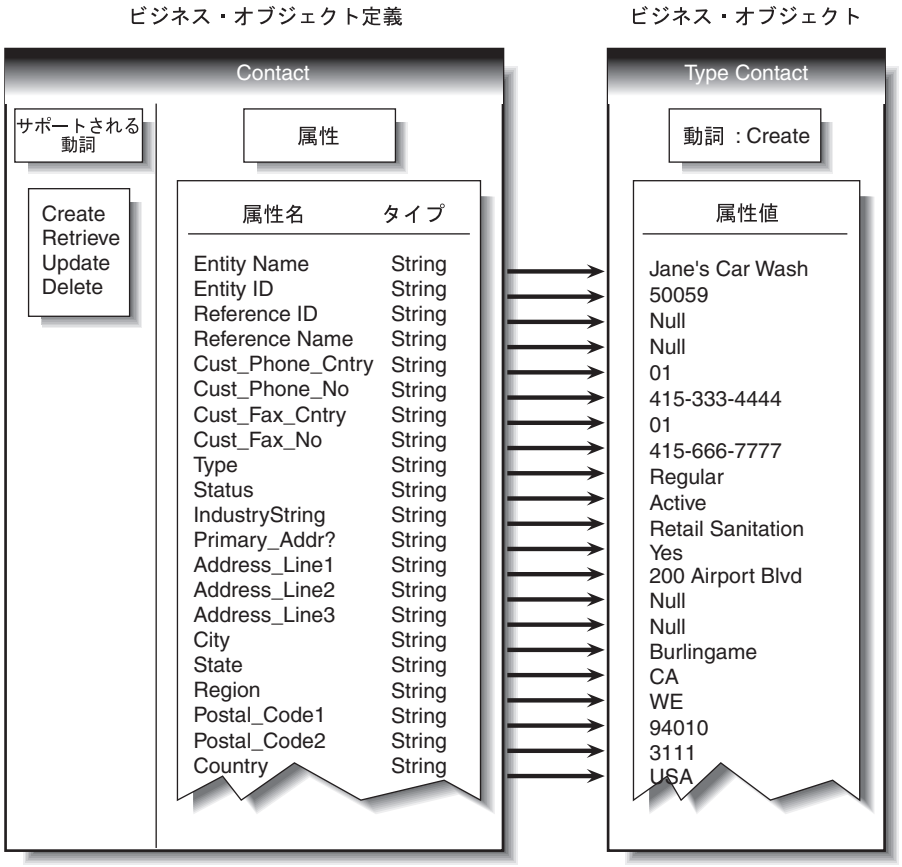

図 25. ビジネス・オブジェクト定義およびビジネス・オブジェクト

コネクター・エージェントおよびコラボレーション・シナリオは、両方ともビジネス・オブジェクト定義を使用してビジネス・オブジェクトを作成します。

ビジネス・オブジェクト定義のコンポーネント

1 ページの『第 1 章 IBM WebSphere InterChange Server の概要』 では、ビジネス・オブジェクトのコンポーネントをタイプ、属性値、動詞のように簡単な用語で説明しました。このセクションでは、ビジネス・オブジェクト定義の主要なコンポーネントについて詳細に説明します。

全般的に、ビジネス・オブジェクト定義は名前で識別されます。名前は Customer、VantiveCase、Invoice など、ビジネス・オブジェクト定義のタイプを表します。アプリケーション固有のビジネス・オブジェクトには、コネクター・エージェントが処理するときに役立つアプリケーション固有情報も含まれます。また、すべてのビジネス・オブジェクトには、次のセクションで説明する属性や動詞も含まれます。

属性

ビジネス・オブジェクト定義の**属性**は、Last Name、Employee ID、Case Number、Amount、Date Initiated などのエンティティーに結びついた値を表します。実行時に、属性に実際のデータが入ります。

例えば、Employee ビジネス・オブジェクト定義には、従業員の名前、住所、従業員 ID、その他の関連情報の属性が含まれます。ビジネス・オブジェクトの属性は、フォームのフィールドまたはデータベース表の列に類似しています。

また属性は、契約の項目の配列や送り状の部品参照など、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参照する場合もあります。

ObjectEventId 属性

ObjectEventId 属性は必須属性であり、すべてのビジネス・オブジェクト内で最後の属性です。コネクターがイベントをパブリッシュするとき、ビジネス・オブジェクト定義の ObjectEventId 属性を使用し、作成中の特定のビジネス・オブジェクト・インスタンスを識別する一意の値を保管します。

ObjectEventId 属性の値はシステムによって生成および処理されます。システムはこの値を使用して、システム全体における特定のイベントのフローを識別および追跡します。開発者は、コネクター・エージェントまたはデータ・ハンドラーを使用して ObjectEventId 属性をマップしたり取り込んだりすることはできません。

単純属性タイプと複合属性タイプ

属性タイプが String、Boolean、Double、Float、Integer などの基本データ型の場合、属性値はデータベースのフィールド値のような離散的なデータ項目です。このような属性は、**単純属性**と呼ばれます。例えば、LastName、CustomerID、PartNumber、AssignedTo、Price などです。

属性タイプが別のビジネス・オブジェクト定義名 (複合タイプ) である場合、属性値は子ビジネス・オブジェクトまたは子ビジネス・オブジェクトの配列です。このような属性は、**複合属性**と呼ばれます。例えば、Customer、Contract、Oracle_Contact などです。

属性プロパティー

いくつかの**プロパティー**は、属性が表す値を定義します。ここでは可能性のあるプロパティーをすべては示していないが、図 26 にビジネス・オブジェクト定義内の属性プロパティーの位置を示します。

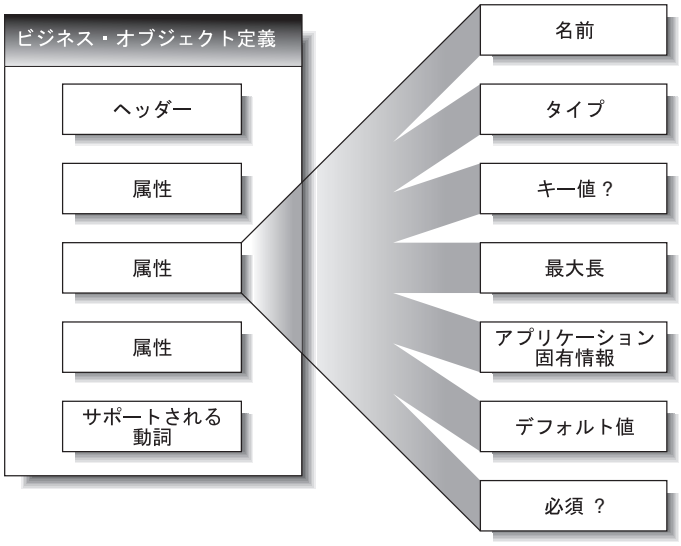

図 26. 属性プロパティー

特定の属性に対するプロパティーのセットは、属性タイプが基本か複合かに依存します。つまり属性のプロパティーは、属性が単一ユニットのデータと子ビジネス・オブジェクトのどちらを参照するかによって異なります。

動詞

動詞は、ビジネス・オブジェクトのデータに対して行うアクションを示します。ビジネス・オブジェクト定義には動詞のリストが含まれます。1 つのビジネス・オブジェクトに含まれる動詞は 1 つのみです。ビジネス・オブジェクト定義に関連した最も一般的な動詞は、Create、Retrieve、Update、Delete です。動詞の意味は、ビジネス・オブジェクトの役割によって異なります。動詞は、アプリケーション・イベントの記述、呼び出しの作成、要求の作成、直前の要求結果の確認などを実行できます。

注: 物理的な削除要求をサポートしないアプリケーションもあります。このようなアプリケーションに対応するため、IBM WebSphere InterChange Server システムは同等の論理削除を実行します (通常、非アクティブ状態にする更新を実行します)。さらに、アプリケーションが物理的な削除をサポートしている場合でも、要求をアプリケーションに送信するときに Delete 動詞を Update 動詞に変換するように IBM WebSphere InterChange Server システムを構成できます。

アプリケーション固有のビジネス・オブジェクトの詳細

アプリケーション固有のビジネス・オブジェクトには、コネクター・エージェントと特定のアプリケーション間でやり取りされるデータが含まれます。そのため、各アプリケーション固有のビジネス・オブジェクト定義には、それぞれのアプリケーションのデータ・モデルおよびコネクター・エージェントのアクセス方式が反映されます。

2 つのアプリケーション固有のビジネス・オブジェクトが類似のアプリケーション・エンティティーを参照する場合でも、属性の編成方法やアプリケーション固有情報に相違が現れます。

属性の編成

アプリケーションは、同一の情報を異なる方法で編成することがよくあります。例えば、アプリケーション A は連絡先用に電話番号と FAX 番号を 4 つのフィールドに格納していますが、アプリケーション B はこれらの番号を 2 つのフィールドに格納しています。

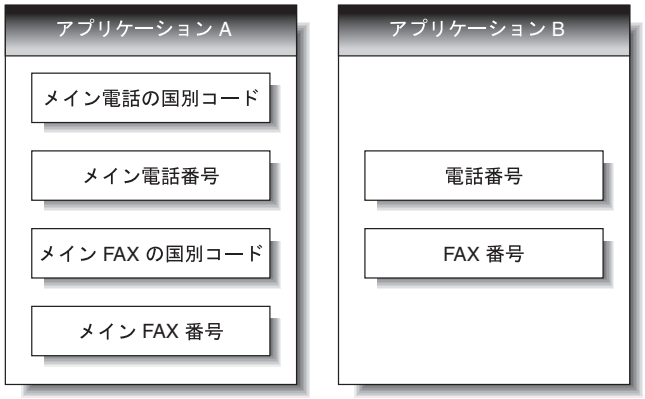

図 27. 2 つのアプリケーションでの電話データ

アプリケーション A ビジネス・オブジェクトとアプリケーション B ビジネス・オブジェクトのビジネス・オブジェクト定義は、この差異を反映して異なる属性を持ちます。

アプリケーション固有情報

アプリケーション固有のビジネス・オブジェクトが異なるのは、各ビジネス・オブジェクトにコネクター・エージェントのための処理命令 (アプリケーション固有情報) を任意で組み込むことができるためでもあります。アプリケーション固有情報は、コネクター・エージェントがビジネス・オブジェクトを処理するために必要なあらゆる情報から構成されます。

ビジネス・オブジェクト定義には、ビジネス・オブジェクト全体、各属性、および各動詞に適用されるアプリケーション固有情報を含めることができます。ビジネス・オブジェクト定義内のアプリケーション固有情報が表示される各場所で、コネクターがアプリケーションとの対話に使用する情報が提供されます。

ビジネス・オブジェクトのアプリケーション固有情報

ビジネス・オブジェクトのアプリケーション固有情報は、コネクター・エージェントがビジネス・オブジェクト全体を処理するときに使用する情報を提供します。

属性のアプリケーション固有情報

属性に適用するアプリケーション固有情報は、アプリケーション内での属性値の位置を示すことがよくあります。コネクター・エージェントがアプリケーションへの API 呼び出しを構築して属性値を検索または入力するときに、この情報を使用します。

アプリケーション固有情報は、アプリケーションごとに異なる形式をとります。コネクター・エージェントは、アプリケーションの形式およびフィールド名によって属性の位置を参照できる場合もありますが、参照がより複雑になる場合もあります。

表 4 は、Customer ビジネス・オブジェクトのアプリケーション固有情報の 2 つの例を示しています。

表 4. サンプル属性のアプリケーション固有情報

属性	サンプル・アプリケーション 1 のアプリケーション固有情報	サンプル・アプリケーション 2 のアプリケーション固有情報
Customer ID	CUST:CID	cust:site_id::3::
Customer name	CUST:CNAM	cust:name::3::
Status	CUST:CSTAT	cust:status::1:0:Status
Industry	CUST:CIND	cust:industry_type::3::Industry

値には次の意味があります。

- サンプル・アプリケーション 1 の値は、CUST 表内のフィールドを指定します。
- サンプル・アプリケーション 2 の値は、次の形式を使用してアプリケーション・データベース内の表、フィールド、および関係を指定します。

  table:*field*:*relation*:*type*:*default-value*

  ここで、*type* は、1 (int)、2 (float)、または 3 (string) を表します。

例外的に、属性のアプリケーション固有情報が必要ない場合があります。例えば、アプリケーションの中にはデータ単位の指定が直接的で使いやすいものがあります。アプリケーションが表 5 に示すようなサンプル・フィールドを識別することを考えてください。

表 5. サンプル・アプリケーション ID

属性	値を持つフィールドのアプリケーション ID
Customer ID	XCustomerID
Customer name	XCustomerName
Status	XStatus
Industry	XIndustry

表 5 に示した例では、変換規則が規則的 (X を追加するか除く) なので、コネクター・エージェントが属性とアプリケーションの ID を容易に関連付けられます。このため、このアプリケーションのビジネス・オブジェクトの属性は、アプリケーション固有情報を必要としないこともあります。

動詞に関するアプリケーション固有情報

ビジネス・オブジェクト定義には、サポートされている動詞のアプリケーション固有情報を含めることができます。アプリケーション固有情報は、動詞がアクティブな場合、ビジネス・オブジェクトを処理する方法をコネクター・エージェントに指示します。

変更オプション

汎用ビジネス・オブジェクト定義またはアプリケーション固有のビジネス・オブジェクト定義は変更できます。場合によっては、その両方の変更が必要になります。例えば、コラボレーションがいくつか追加のアプリケーション・データを処理するように変更するには、両方のタイプのビジネス・オブジェクト定義を変更して属性を追加し、それぞれのビジネス・オブジェクトがデータを転送するようにする必要があります。ビジネス・オブジェクト定義を変更する場合、マッピング・コンポーネントも変更する必要があります。

要約

この章では、IBM WebSphere InterChange Server システムにおけるビジネス・オブジェクトの定義と使用について説明しました。覚えておくべき重要な点は次のとおりです。

- ビジネス・オブジェクト定義とは、データの操作を起動するアプリケーション・データおよび動詞のセットのタイプと配列を定義する仕様のことです。ビジネス・オブジェクトとは、実際のデータと動詞を含んだ、定義のインスタンスのことです。
- 同じタイプのエンティティーを持つアプリケーション固有のビジネス・オブジェクトが異なるのは、アプリケーション・モデルが異なるためです。この違いは、属性の配列と数、およびアプリケーション固有情報に顕著に現れます。
- ビジネス・オブジェクト定義には、動詞と属性が含まれます。基本データ型の属性はデータを参照し、複合データ型の属性は子ビジネス・オブジェクトを参照します。
- ビジネス・オブジェクト定義内では、関連するアプリケーション・データがアプリケーション固有情報に含まれます。コネクター・エージェントはアプリケーション固有情報を使用して、アプリケーションとの対話を促進します。
- サイトでデータ・フローを変更する必要がある場合、アプリケーション固有および一般の両方のビジネス・オブジェクトを変更できます。

第 5 章 コネクター

コネクターは、1 つのアプリケーション (または Web サーバーなどのプログラマチック・エンティティー) と 1 つ以上のコラボレーションを媒介します。コネクターは、SAP R/3 バージョン 4 などのアプリケーションや、データ・フォーマットまたはプロトコル (XML や EDI) などのテクノロジーに固有の場合もあります。コネクターがコラボレーションと通信するときは、次の 2 つの形式をとります。

- アプリケーション・**イベント通知**。特定のアプリケーションのコネクターがコラボレーションに渡します。
- **要求処理**。コネクターがコラボレーションに代わり実行します。

コネクターは、ローカル・ネットワークにあるアプリケーション、およびインターネット・ファイアウォールの外部にあるリモート・アプリケーションに構成できます。

大部分のコネクターには共通する動作があり、異なるのはアプリケーションやアプリケーション固有のビジネス・オブジェクトと対話する方法のみです。この章では、コネクターの共通の動作と相違点の両方について説明します。この章は次のセクションから構成されます。

多くのアプリケーションおよびテクノロジーのコネクターが WebSphere Business Integration Adapters 製品の一部として提供されています。WebSphere Business Integration Adapter には、コネクター自身およびそのメッセージ・ファイルと構成ツールが含まれますが、そのコネクターに固有のビジネス・オブジェクトの作成および処理を行うための機構が含まれることもあります。ただし、カスタム・コネクターを作成したり、既存のコネクターを変更する必要があるときは、コネクターの動作についての詳細な知識が必要になります。カスタム・コネクターの作成方法の詳細については、「*コネクター開発ガイド (Java 用)*」または「*コネクター開発ガイド (C++ 用)*」を参照してください。

コネクターの始動

InterChange Server システムのコネクターには以下のコンポーネントがあり、コネクターを実行するにはこれらの各コンポーネントを始動する必要があります。

- **コネクター・コントローラー**は、InterChange Server と同時に始動します。コネクターの構成がすでに完了しているときは、コネクターのコントローラーが InterChange Server によって自動的に作成されます。
- **コネクター・エージェント**は、コマンド行またはショートカットを使用して明示的に始動する必要があります。ただし、異常シャットダウンや障害の後にコネクター・エージェントが自動的に再始動されるように、オプションで構成できます。

コネクター・エージェントは、始動時および実行中は常に、特定の InterChange Server と通信します。コネクター・エージェントを始動するときには、InterChange Server の名前を指定します。コネクター・エージェントは最初に InterChange Server にアクセスして、自身の構成設定と、サポートするよう構成されているビジネス・オブジェクト定義を要求します。

コネクター・エージェントを始動するときには、InterChange Server を実行していなくても構いません。ただし、コネクター・エージェントの初期化に必要な情報を取得するまでは、有効な作業を実行できません。InterChange Server を実行しないでコネクター・エージェントを始動すると、コネクター・エージェントはサーバーへのアクセスを繰り返し試みます。

コネクター・エージェントがアプリケーションに接続するには、InterChange Server リポジトリーから自身の構成を受け取り、ビジネス・オブジェクト定義をダウンロードする必要があります。また、サポートする必要があるコラボレーション・イベント・サブスクリプションのリストも取得します。

以下の図は、コネクター・エージェントの始動の 3 つのステップを示しています。

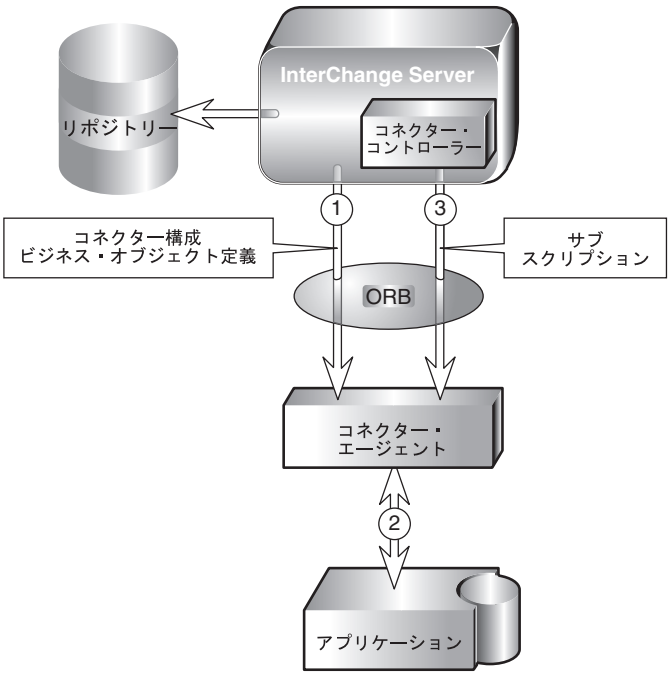

図 28. コネクターの始動

コネクター・エージェントは、次の順序で始動タスクを実行します。

1. InterChange Server にアクセスして、構成情報とビジネス・オブジェクト定義を取得します。
2. アプリケーションに接続します。
3. コネクター・コントローラーにアクセスして、サブスクリプションのリストを取得します。

コネクターは、自身のビジネス・オブジェクト定義と特定の構成プロパティー (アプリケーションのユーザー名やパスワードなど) を始動時にのみダウンロードします。他のプロパティー (コネクター・トレースを制御するプロパティーなど) の値は、動的に変化します。

イベント通知

多くの (すべてではない) コネクターは、コラボレーションとパブリッシュ・アンド・サブスクライブ対話を行い、イベント通知を使用してそれらのコラボレーションのイベントを起動します。InterChange Server Collaborations にトリガー・イベントを提供するアプリケーションのコネクターは、これらのイベントについて認識し、関連するデータをサブスクライブしているコラボレーションに送信する必要があります。以下の図は、イベント通知に関連したコネクターの対話を示しています。

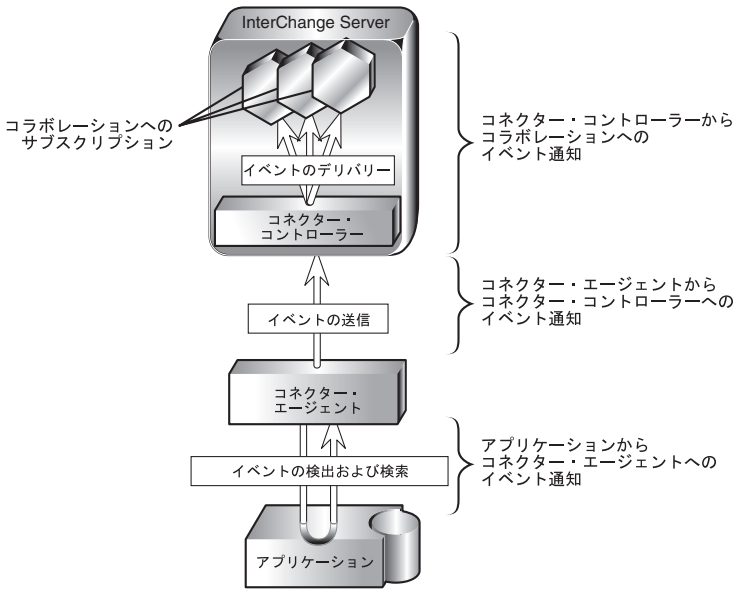

図 29. コネクター内のイベント通知

コネクター・エージェントがイベントを検出および取得する方法は、コネクターごとに異なります。しかし、コネクター・エージェントがイベントをコネクター・コントローラーに送信する方法、およびコネクター・コントローラーがイベントをコラボレーションに送信する方法は、すべてのコネクターで共通です。

次のサブセクションでは、次の項目を中心に、大半のコネクターの操作に関する一般的概念について説明します。

- コネクターがアプリケーション・イベント通知機構を使用する方法
- コネクターがイベントを検出および処理する方法

この記述は、特定のコネクターの特定の実装について説明するわけではありません。

アプリケーションのイベント通知メカニズムのセットアップ

コネクターにとって、**アプリケーション・イベント**とは、ビジネス・オブジェクト定義に関連するアプリケーション・エンティティーのデータに影響を及ぼすすべての操作を指します。アプリケーション・イベントには、他のタイプもあります。例えばマウスのクリックは、アプリケーションのウィンドウ・システムまたはフォーム・インターフェースに対するイベントです。ただしコネクターは、作成、更新、削除など、アプリケーションのデータ・ストアの内容に影響を及ぼすデータ・レベルのイベントにのみ反応します。

アプリケーションの中には、使いやすいイベント管理および構成可能なイベント・テキストを提供して、明示的にトラップしたりイベントを報告するものもあります。また、離散的で報告可能なイベントの概念を持たず、何かが起きたときにデータベースを黙って更新してしまうアプリケーションもあります。

ほとんどのコネクターでは、実装者がいくつかのアプリケーション構成を実行して、**イベント通知機構**をコネクターの使用のために設定する必要があります。イベント通知機構は、アプリケーションで実行される操作の番号付きリストです。リストには、アプリケーション・イベント・キュー、E メール受信箱、データベース表などの物理形式が含まれます。

コネクターが使用するイベント通知機構はどのようなものでしょうか。以降のセクションでは、いくつかの一般的なアプローチについて説明します。

アプリケーションにイベント・サポートがある場合

アプリケーションがイベントに基づく場合は、クライアント・アプリケーションが使用するためのイベント通知インターフェース (コネクターなど) を持っていると考えられます。また、実装者は、アプリケーションにイベント報告テキストを構成できます。このようなアプリケーションにとって、コネクターのイベント通知機構の設定は通常のアプリケーション設定タスクです。

例えば、あるアプリケーションで、特定タイプのイベントが発生したときに実行するスクリプトをインストールでき、スクリプトは通知をイベント受信箱に入れられるとします。このアプリケーションのコネクターをインストールするには、コネクターのユーザー・アカウントを作成し、追跡するイベントを扱うスクリプトを書き込むか取得します。次にスクリプトをインストールし、各スクリプトを起動するイベント・タイプを指定して、受信箱を作成します。完了すると、コネクター・エージェントは定期的に受信箱の内容を検索し、新規イベントをチェックします。

以下の図は、イベント受信箱を含むアプリケーション構成を示しています。

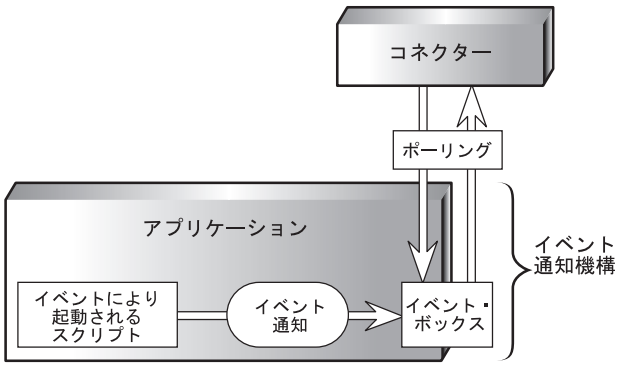

図 30. イベント通知でのイベント受信箱の使用例

特定の操作が発生したときに、E メール・メッセージを作成したりイベント・キューに書き込むことができる内部ワークフロー・システムを持つアプリケーションもあります。図 31 に、ビジネス・オブジェクトとイベントが定義されているビジネス・オブジェクト・リポジトリーを持つアプリケーションを示します。この図で、Customer はアプリケーション・ビジネス・オブジェクトを示し、Create、Delete、および Update はビジネス・オブジェクトに関連するイベントのタイプを示します。

Customer.Update などのビジネス・オブジェクト・イベントが発生すると、ワークフロー・システムに送信されます。ワークフロー・システムは、エントリーをアプリケーション・データベースのイベント表に配置します。

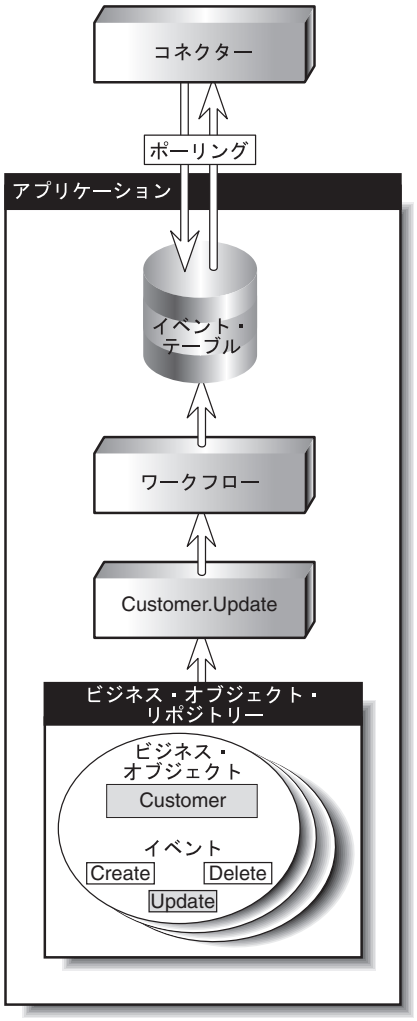

図 31. 例: イベント通知でのアプリケーション・ワークフローの使用

アプリケーションにイベント・サポートがない場合

コネクターがアプリケーション・イベントと対話する方法で適切なのは、アプリケーションの API を介する方法です。API は、アプリケーションのデータ・モデルとロジックを強化するフレームワークを提供します。ただし、アプリケーション API の中には、イベント通知のネイティブ・サポートを提供しないものもあります。

このようなアプリケーションからコネクターがイベント通知を受け取る 1 つの方法は、アプリケーション・データベースとの対話です。例えば実装者は、行への更新を検出するトリガーを Employee 表に設定できます。更新が発生すると、トリガーは更新情報を特殊な表に挿入します。このイベント表内の新規の行は、イベント通知を表します。コネクターは SQL 照会を使用して、この表から新規イベントを検索できます。

図 32 に、この方法を示します。

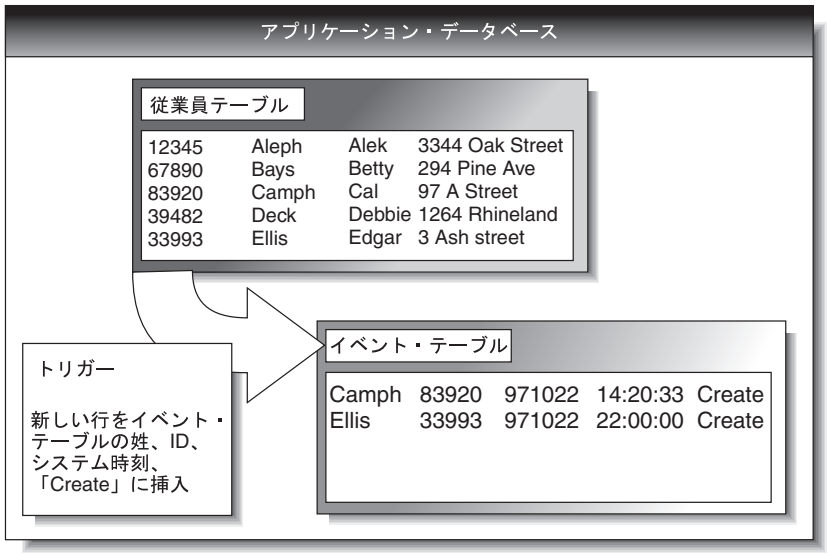

図 32. 例: イベント通知でのデータベースの使用

図 32 にあるアプリケーション・データベースには、Employee 表のレコード作成から生じるトリガーがあります。アプリケーションが新規レコードを挿入するたびに、トリガーはイベント表に行を作成します。行には、新規従業員レコード (姓および従業員 ID)、システム時刻、およびイベント・タイプ Create というキー値が含まれます。

イベントの検出

コネクター・エージェントは、イベント通知機構で新規イベントをポーリングすることによって、アプリケーション・イベントを認識します。ポーリング方式は完全にアプリケーション固有で、コネクターが使用するイベント通知機構に基づいています。

ポーリングは構成可能です。System Manager を使用してコネクターを構成したときは、次のことを行うことができます。

- コネクター・エージェントがポーリングする頻度の調整
- コネクター・エージェントがポーリングする時間の指定

コラボレーションがアプリケーション・イベントを要求しない場合、コネクター・エージェントはアプリケーションにポーリングする必要はありません。特定のアプリケーションがコラボレーションに参加していてもイベントのソースではない場合、ポーリング頻度を「no」に設定することによって、コネクター・エージェントのポーリングを停止できます。

イベントの処理

イベントを検出すると、コネクター・エージェントは次のことを実行します。

- アプリケーション・イベントと、そのイベントを表すビジネス・オブジェクト定義および動詞の関連付け
- イベントへのサブスクリプションのチェック
- アプリケーション・データの検索

- ビジネス・オブジェクトのコネクター・コントローラーへの送信 (必要に応じて)
- イベントのアーカイブ (オプション)

アプリケーション・イベントをビジネス・オブジェクト定義と関連付ける方法

コネクター・エージェントがイベントを検索するときには、そのイベントにコラボレーションがサブスクライブしているかどうかを判別する必要があります。ただし、コラボレーションがビジネス・オブジェクト・タイプ.*動詞* の形式でイベントをサブスクライブするので、コネクターはまずイベントを表すビジネス・オブジェクト定義と動詞を決定する必要があります。

表 6 に示されているように、コネクター・エージェントはイベント・テキストを使用して、イベントをビジネス・オブジェクト定義と動詞に関連付けます。

表 6. イベント・テキストとビジネス・オブジェクト形式

アプリケーション・イベント内のデータ型	例	使用法
アプリケーション・エンティティー・タイプ	Customer、Part、Item	関連付けられたビジネス・オブジェクト定義の決定
発生した動作	Create、Update、Delete	ビジネス・オブジェクトのアクティブな動詞の決定

例えば、コネクターは次のイベント・テキストを Employee.Create ビジネス・オブジェクトに関連付けることができます。

```
1997.10.19.12:50.22 employee created lname="como" id="101961"
```

このイベント・テキストには、次の内容が含まれます。

- イベントを一意的に識別するのに役立つタイム・スタンプ
- アプリケーション・エンティティー「employee」
- 動作「created」
- コネクター・エージェントが残りの従業員情報の検索に使用できるキー値、従業員の姓および ID

この例は単純ですが、他のタイプのイベント・テキストは、コネクター・エージェントによる処理をさらに要求する可能性があります。

サブスクリプションのチェック

コネクター・エージェントがイベントをビジネス・オブジェクト定義と動詞に関連付けると、内部のサブスクリプション・リストをチェックして、そのイベントにコラボレーションがサブスクライブしているかどうかを検索します。コネクター・コントローラーは、サブスクリプションが変更するたびにコネクター・エージェントを更新するので、サブスクリプション・リストは常に最新のものです。

イベントが現在のサブスクリプションと一致しない場合、コネクター・エージェントは警告メッセージをログに記録し、イベントを破棄します。

ビジネス・オブジェクトの作成

イベントにサブスクリプションがある場合、コネクター・エージェントはビジネス・オブジェクトを作成し、キー値を使用してアプリケーション・データを検索して、ビジネス・オブジェクトに加えます。80 ページの『ビジネス・オブジェクトの構成とデコンストラクション』では、ビジネス・オブジェクトを作成するプロセスについて説明します。

コネクター・コントローラーへのビジネス・オブジェクトの送信

コネクター・エージェントは、使用中のトランスポート (メッセージングまたは CORBA) によってビジネス・オブジェクトをコネクター・コントローラーに送信します。コネクター・エージェントが認識しているのは、ビジネス・オブジェクトにサブスクリプションがあるということのみで、どのコラボレーション (単数または複数) がサブスクライバーなのかは認識していません。コネクター・コントローラーは、コラボレーションへのデリバリーを行います。

イベントのアーカイブ

アプリケーション・イベント・アーカイブは、トラブルシューティングやレコードの保存に有効です。イベント・アーカイブには、各イベントについての次のような状況情報が含まれます。

- InterChange Server に正常に送信されました
- イベントにサブスクリプションがありません
- 処理に失敗しました

アプリケーションがイベント・アーカイブ機能を提供する場合、一般的にコネクターはそれを使用します。イベント・アーカイブ機能をサポートしないアプリケーションのコネクターは、独自のイベント・アーカイブ機能を持つ場合があります。例えば、コネクターのイベント通知機構が 図 32 のようなデータベース機構である場合、データベースのトリガーは削除されたイベントをアーカイブ表にコピーできます。

要求処理

コネクターは、コラボレーションから要求を受け取ったり、アプリケーションの代わりに要求を作成したりすることができます。例えば、コラボレーションはコネクターに対して、契約の削除、部品の更新、顧客の作成という要求を、それぞれ Contract.Delete、Part.Update、Customer.Create というビジネス・オブジェクト形式で送信できます。

コネクター・コントローラーがコラボレーションの要求を受信すると、その要求をコネクター・エージェントに転送します。コネクター・エージェントはビジネス・オブジェクトをアプリケーション要求 (通常は API への呼び出しのセット) に変換し、結果を戻します。

図 33 に、コラボレーション要求の処理に関するコネクターの対話を示します。

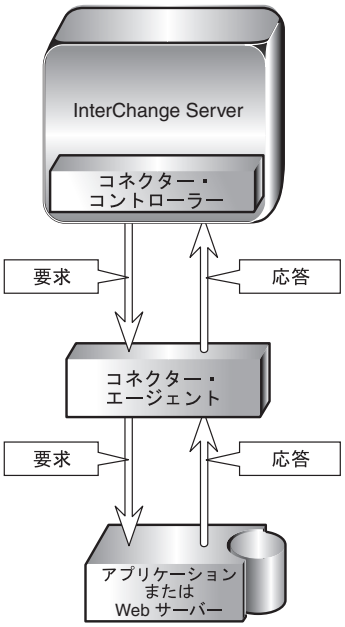

図 33. 要求の処理のためのコネクターの対話

注: イベント通知にパブリッシュ・アンド・サブスクライブ対話が使用されるのに対して、コラボレーションがビジネス・オブジェクトを特定のコネクターに送信するには、要求/応答対話を使用します。

コネクター・エージェントが要求を受信すると、次の 3 タイプの情報に基づいて要求の処理方法を決定します。

- ビジネス・オブジェクトの動詞
- 動詞のアプリケーション固有情報
- ビジネス・オブジェクト定義自身に含まれ、ビジネス・オブジェクトの構築および破棄に使用されるメタデータ

これらの要素は、以降のトピック内で説明します。

動詞ベースの処理

コネクター・エージェントは、ロジックおよびアプリケーションの API に基づいて、要求内の Create、Retrieve、Update、または Delete 動詞に反応します。2 つのコネクター・エージェントが同一タイプの要求を異なる方法で処理する場合がありますが、結果は論理的に同じになります。

一部のコネクターでは、要求に含まれる動詞に関係なく、ビジネス・オブジェクトでの操作の実行に 1 つの方法のみを要求します。しかし、多くのコネクターでは、各動詞にそれぞれ異なる方法が要求されます。

コネクター・エージェントが要求を受信すると、ビジネス・オブジェクトのアクティブな動詞に合うメソッドをアプリケーションから呼び出します。例えば、コネクター・エージェントが AppAEmployee.Update ビジネス・オブジェクトを受信すると、AppAEmployee オブジェクトの Update メソッドを呼び出します。Update メソッドは、更新を実行するためにアプリケーションと対話します。

図 34 には、いくつの動詞がメソッドを処理する様子を示しています。

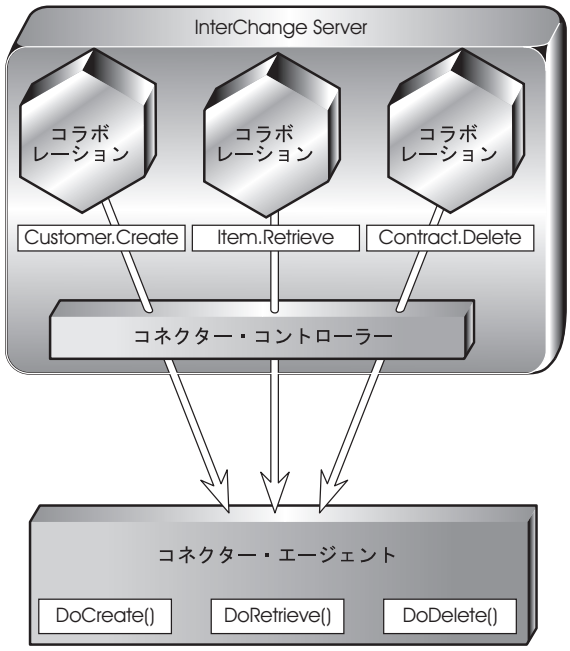

図 34. 要求の処理

図 34 のコネクターが Customer.Create、Item.Retrieve、または Contract.Delete 要求を受信すると、それぞれ DoCreate、DoRetrieve、または DoDelete メソッドを呼び出します。

動詞ベースのアプリケーション固有情報

ビジネス・オブジェクト定義において、アプリケーション固有情報は追加の入力データを、ビジネス・オブジェクトのインスタンスを処理するコネクター・エージェントに提供します。各動詞は、アプリケーション固有情報を持つことができます。アプリケーション固有情報の形式および内容は、完全にコネクター固有のものです。

例えば、あるビジネス・オブジェクト定義内の Retrieve 動詞のアプリケーション固有情報は、そのコネクター・エージェントの Retrieve メソッドに特別な入力引き数を提供します。

例として、MyApp アプリケーションに 3 つの情報形式があるとすると、InventoryItem に関する情報は次のように表示されます。

- InventoryItem-New
- InventoryItem-Change
- InventoryItem-Remove

MyApp コネクター・エージェントが在庫品目に対して操作を行うときには、その操作に合った形式を参照する必要があります。InventoryItem ビジネス・オブジェクト定義内の各動詞に関連したアプリケーション固有情報フィールドには、形式名を保管できます。

動詞固有のメソッドおよびこれらのメソッドに対するアプリケーション固有の入力データの組み合わせによって、コネクター・エージェントは固有の処理命令を取得します。

並行処理機能

Java ベースの InterChange Server が持つマルチスレッド性により、コネクター・コントローラーがコラボレーションにデリバリーするイベントは並行処理が可能です。

実装されているアプリケーションやコネクターによっては、コネクター・エージェントが受信する要求でも並行処理を実行できます。どちらの場合も、複数のスレッドまたは並列処理の使用によって実行します。この機能は、コラボレーションとアクセス・クライアントのどちらから発信される要求にも適用できます。

並行処理を実装する方法は、コネクターのタイプと設計によって異なります。

- コネクター・エージェントが C++ で作成されている場合、本質的に単一スレッド型です。ただし、C++ コネクター・エージェントは Connector Agent Parallelism を使用することによって、並行処理できるようになります。Connector Agent Parallelism とは、コネクター・エージェントの単一スレッド・マスター・プロセスから複数のスレーブ・プロセスのインスタンスを作成することによって並行処理を実行する機能です。この機能は、System Manager の構成可能な設定を使用してアクティブまたは非アクティブにできます。
- コネクター・エージェントが Java で作成されている場合、Connector Agent Parallelism 機能を使用しなくても、マルチスレッドによって並行処理する能力を本質的に持っています。
- Java Connector Development Kit (JCDK) 自体はマルチスレッド型ですが、Java で作成されたコネクター・エージェントの中には、単一スレッド処理を実行するよう設計されているものもあります。一般的に Java コネクター・エージェントがこのように設計されるのは、アプリケーションの API ライブラリーが、コネクター・エージェントを単一スレッドでのみ実行するよう制限している場合です。このようなコネクターでは、Connector Agent Parallelism 機能を使用して、コネクター・エージェントの単一スレッド・マスター・プロセスから複数のスレーブ・プロセスのインスタンスを作成することができます。

Connector Agent Parallelism を使用するには、アプリケーション自体が並行処理をサポートできることが必要です。アプリケーションによっては、並行処理を実行できないようにするアーキテクチャーや処理要件を持つ場合があります。

ビジネス・オブジェクトの構成とデコンストラクション

コネクター・エージェントは、ビジネス・オブジェクトを構築および破棄することによって、イベント通知タスクや要求処理タスクを完了します。

- コネクター・エージェントが InterChange Server に送信するイベントを検索するとき、そのイベントを表すビジネス・オブジェクトを構築します。
- コネクター・エージェントがコラボレーションの要求を表すビジネス・オブジェクトを受信すると、ビジネス・オブジェクトを破棄してアプリケーション要求を作成します。

ビジネス・オブジェクト・メタデータおよびコネクターの動作

コネクターによる、アプリケーション・イベントからビジネス・オブジェクトへの変換およびビジネス・オブジェクトからアプリケーション要求への変換は、ビジネス・オブジェクトの設計時に定義されるデータ定義 (**メタデータ**) によって駆動されます。

コネクター・エージェントとビジネス・オブジェクト・メタデータは、協働するように設計されます。コネクター・エージェントとそのビジネス・オブジェクトの設計は、特定の機能をソフトウェアとハードウェアのどちらを使用しても実装できる、コンピューター・デバイスの設計に類似しています。開発者は、主要な機能をどこに実装するかを決定するために、パフォーマンス、拡張性、およびその他の問題を考慮します。

同様に、コネクター・エージェントとそのビジネス・オブジェクト間の作業分配は、コネクター開発者の設計判断によって決定します。IBM WebSphere InterChange Server 設計原則では、ビジネス・オブジェクト・メタデータを使用して、コネクター・エージェントのハードコーディングされたロジックではなくコネクター・ロジックを駆動することを推奨します。

属性のタイプ、サイズ、およびデフォルト値を指定するプロパティーに加えて、ビジネス・オブジェクト定義はアプリケーション固有フィールドを使用して、ビジネス・オブジェクトの処理方法について特定の指示をコネクター・エージェントに与えます。

例えば、61 ページの『第 4 章 ビジネス・オブジェクト』で提示した、顧客を表すビジネス・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の例を思い出してください。表 7 に、これらの例をいくつか示します。

表 7. アプリケーション固有情報の例

属性	アプリケーション固有情報
Customer ID	CUST1:CID
Customer name	CUST1:CNAM
Status	CUST1:CSTAT

ビジネス・オブジェクトを処理するとき、コネクター・エージェントはその定義を読み取り、アプリケーション固有情報を使用してアプリケーション要求を作成します。

- Customer ID を取得するため、コネクター・エージェントは Form CUST1、Field CID の値を取得します。
- Customer name を取得するため、コネクター・エージェントは Form CUST1、Field CNAM の値を取得します。
- Customer status を取得するため、コネクター・エージェントは Form CUST1、Field CSTAT の値を取得します。

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およびその他のメタデータは、コネクター・エージェントのアクションを決定するので、コネクター・エージェントの動作は**メタデータ主導型**であるといえます。

メタデータ主導型コネクター・エージェントの利点

メタデータ主導型コネクター・エージェントは、サポートするビジネス・オブジェクト・タイプの処理方法についてのハードコーディングされた指示を持たないので、柔軟性があります。コネクターの仕様に一致するかぎり、コネクター・エージェントは新規ビジネス・オブジェクト定義を再コーディングや再コンパイルせずに、自動的にサポートします。

メタデータ主導型コネクター・エージェントは、ビジネス・オブジェクト定義内の新規または変更された属性もサポートします。コネクター・エージェントは、ビジネス・オブジェクト定義の属性をループしながら、属性を自動的に処理します。

ビジネス・オブジェクト作成の例

コネクター・エージェントがビジネス・オブジェクト定義からビジネス・オブジェクトを作成する方法は、次のとおりです。

1. コネクター・エージェントがビジネス・オブジェクト定義属性を順にループし、アプリケーション固有情報を使用して、API 呼び出しを用意するかアプリケーション・エンティティーを取得するための照会を作成します。
2. コネクター・エージェントが要求をアプリケーションに送信し、結果を取得します。
3. コネクター・エージェントが結果をループし、AppSpecificInfo の値を使用して、各ビジネス・オブジェクト属性を表す取得値を判断します。

図 35 には、ビジネス・オブジェクト定義からビジネス・オブジェクトを作成しているコネクター・エージェントの例を示します。コネクター・エージェントは、キー値である品目番号 123 を含むアプリケーション・イベントを取得しました。コネクター・エージェントは、Group、Description、Price、および ItemNum という 4 つの属性を含むビジネス・オブジェクト定義から、Item ビジネス・オブジェクトを作成する必要があります。

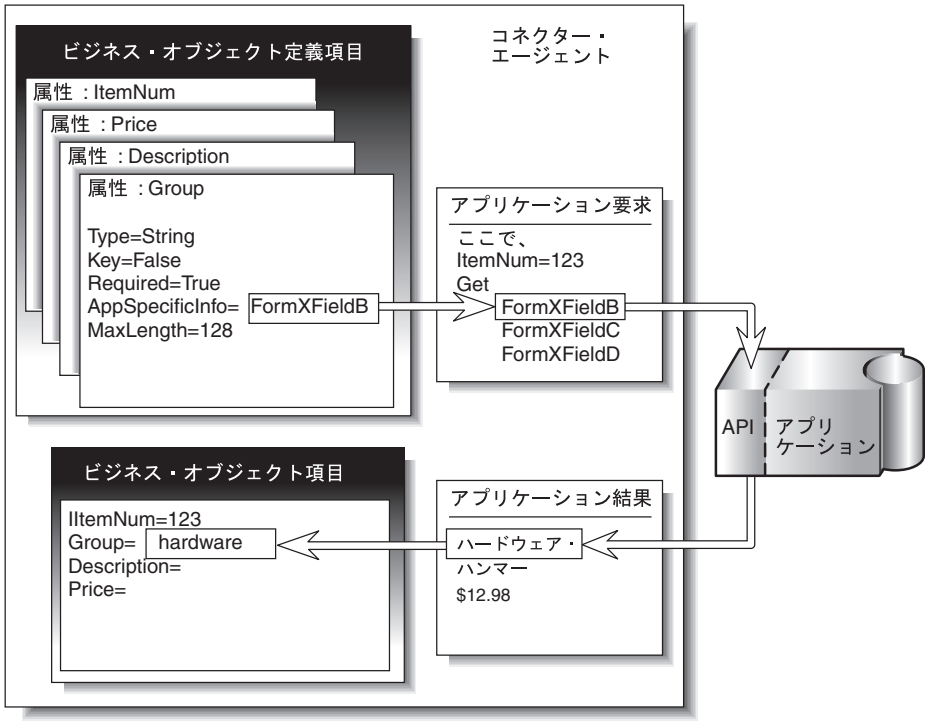

図 35. コネクター内でのビジネス・オブジェクトの作成

品目番号 123 を使用して品目を識別することにより、コネクター・エージェントは残りの属性値を検索します。アプリケーション固有情報は、必要なデータの形式およびフィールドを提供します。

例えば、FormXFieldB は Group データを識別します。コネクター・エージェントは、品目 ID 123 に対する、Form X 内の Field B の値を要求します。次にコネクター・エージェントは戻された値「hardware」を使用して、ビジネス・オブジェクトの Group 属性にその値を挿入します。

破棄のプロセスは、この逆の方法です。コネクター・エージェントはビジネス・オブジェクト定義を使用して、受信したビジネス・オブジェクトに含まれるデータからアプリケーション要求を作成する方法を決定します。

コネクターの構成

コネクターを使用できるようにするには、コネクター定義を用意する必要があります。コネクター定義には以下の情報が含まれます。

- コネクター・エージェントおよびコネクター・コントローラーによって使用されるコネクター構成プロパティーの値
- コネクターがサポートするビジネス・オブジェクト
- 場合により、指定されたビジネス・オブジェクト間で使用される特定マップ

Connector Configurator ツールは、この情報を設定するためのタブを提供します。

コネクター・プロパティー

コネクター・プロパティーには 2 つのタイプがあります。

- **標準プロパティー**

  標準プロパティーは、すべてのコネクターで使用できます。標準プロパティーには、次のものが含まれます。

  – コネクター・エージェントがメッセージをローカルに記録するか、メッセージを InterChange Server に送信して一般ログ・ファイルに入れるかの指示
  – コネクター・エージェントがローカルにログを記録するためのログ・ファイルの場所
  – アプリケーション・ログインおよびパスワード

- **コネクター固有のプロパティー**

  コネクター固有プロパティーは、一般的に、特定のコネクター・エージェントがアプリケーションとのセッションを確立するために必要な値です。さまざまなコネクターのコネクター固有プロパティーには、次のようなものがあります。

  – アプリケーションを実行するマシンの名前または IP アドレス
  – アプリケーション・ゲートウェイ・システムの ID
  – アプリケーション・データベースの名前
  – イベント受信箱の名前

コネクター・プロパティーは、Connector Configurator ツールを使用して設定できます。特定の制約事項がある場合、コネクター・プロパティーは、コネクター・エージェントがインストールされているマシンにあるローカル構成ファイルでも設定できます。コマンド行で設定できるコネクター構成プロパティーもあります。

関連付けられたマップ

コネクター・コントローラーは、マッピングが必要なビジネス・オブジェクトを受信すると、マップ参照を使用します。特定の汎用ビジネス・オブジェクトとアプリケーション固有ビジネス・オブジェクトの間でデータ変換が必要な場合は、この変換を実行するマップを指定する必要があります。関連付けられたマップは、Connector Configurator で指定します。

コネクターの開発

コネクターを開発または変更するときは、コネクター・エージェント自身と、それが使用するビジネス・オブジェクト定義を作成してから、コネクター・リポジトリー定義を作成します。コネクター・コントローラーを作成したり変更したりする必要はありません。コネクター・コントローラー・コンポーネントは InterChange Server の内部要素であり、リポジトリーに定義された各コネクターごとに InterChange Server によってインスタンスが作成されます。

コネクター開発には、コネクターと特定のアプリケーション間の関係の作成が含まれます。実際のコネクターのコーディングは、通常、かなり簡単なプロセスです。最も困難な作業は、次のとおりです。

- アプリケーションのイベント通知メソッドの設計

- アプリケーション固有のビジネス・オブジェクト定義の定義および汎用ビジネス・オブジェクト定義へのマッピング
- アプリケーション固有のビジネス・オブジェクトとコネクター間の関係の定義

コネクター・アーキテクチャーの変更および開発についての詳細は、「*コネクター開発ガイド (Java 用)*」または「*コネクター開発ガイド (C++ 用)*」を参照してください。

要約

この章では、コネクターとその動作の概要について説明しました。覚えておくべき重要な点は次のとおりです。

- コネクターには 2 つの主要な役割があります。それは、アプリケーション・イベントをコラボレーションに通知することと、コラボレーションに代わってアプリケーション要求を実行することです。
- イベント通知の役割では、コネクターはアプリケーションと対話して、アプリケーション内の変更の検出、およびそれらの変更に関連するデータの処理を実行します。
- コラボレーション要求の実装者としての役割では、コネクターは自身がサポートするビジネス・オブジェクト動詞を実装するという固有の機能を使用します。
- コネクター・エージェントがアプリケーション・イベントを構築したり、ビジネス・オブジェクトを破棄してアプリケーション要求を作成するとき、コネクター・エージェントはビジネス・オブジェクト定義内のアプリケーション固有情報およびその他のメタデータによって駆動されます。

次の章では、マッピングの仕組みについて詳しく説明します。

第 6 章 データ・マッピング

データ・マッピングとは、IBM WebSphere InterChange Server システムが、アプリケーション固有のビジネス・オブジェクトから汎用ビジネス・オブジェクトに、または汎用ビジネス・オブジェクトからアプリケーション固有のビジネス・オブジェクトにデータを渡すプロセスです。マッピングによって、IBM WebSphere InterChange Server システムはアプリケーションごとに異なるデータ・モデルを処理できるようになります。

注: コラボレーションが、複数の場所にインストールされた*同一* アプリケーションの間で情報を転送する場合、それぞれのインストール場所で異なるカスタマイズが行われていない限り、マッピングは必要ありません。

データ・マッピングは IBM WebSphere InterChange Server システムの実行時に発生し、実行時に先立って作成したマップを使用します。

この章では、マッピング・プロセス、およびマップと関係の表示、変更、作成のために使用するコンポーネントの概要について説明します。この章は次のセクションから構成されます。

IBM WebSphere InterChange Server マッピング・ツールを使用したマップの作成の詳細については、「*マップ開発ガイド*」を参照してください。

InterChange Server システムでのマッピングの使用方法

マッピングによって、コラボレーションは 1 つのアプリケーションのビジネス・オブジェクトからデータを取り出し、それを変換して異なるアプリケーションのビジネス・オブジェクトを生成できます。マッピング・プロセスにおいて、コラボレーションはコネクター、アクセス・インターフェース、またはその両方と対話します。

IBM WebSphere InterChange Server システムは、複数のビジネス・オブジェクト間のデータ・マッピングに対して、次の機能を含む広範囲のサポートを提供します。

- ソース・ビジネス・オブジェクトの 1 つ以上の属性のデータ値を、宛先ビジネス・オブジェクトの 1 つ以上の属性に変換します。
- 等価だが異なって表され、直接変換できない複数のデータ・エンティティー間の関係を確立および維持します。
- 照会実行用のデータベースやサード・パーティー製のマッピング製品などの外部マッピング・リソースにアクセスできるようにします。

IBM WebSphere InterChange Server 環境では、通常、マッピングはアプリケーション固有のビジネス・オブジェクトと汎用ビジネス・オブジェクトの間で実行されます。IBM WebSphere InterChange Server システムは、アプリケーション固有のビジネス・オブジェクトを他のアプリケーション固有のビジネス・オブジェクトに直接的にはマップしません。その代わり、汎用オブジェクトが 2 つのアプリケーション・データ・モデルの媒介として動作して、マップされた情報を 1 つのデータ・モデルから (コネクターかサーバー・アクセス・インターフェースのどちらかを経由して) コラボレーションに運びます。次に、マップされた情報をコラボレーションから別のデータ・モデルのコネクターに運びます。

図 36 に、実行時のデータ・マッピングが発生する状態を、仮想の Employee Management コラボレーションを例にして示します。Employee Management コラボレーションがソース・コネクターから従業員ビジネス・オブジェクトを受信すると、従業員ビジネス・オブジェクトを宛先コネクターに送信します。(この例で、コラボレーションはコネクターからビジネス・オブジェクトを受信します。コラボレーションがアクセス・インターフェースからビジネス・オブジェクトを受信する場合、類似のマッピング・プロセスが実行されます。)

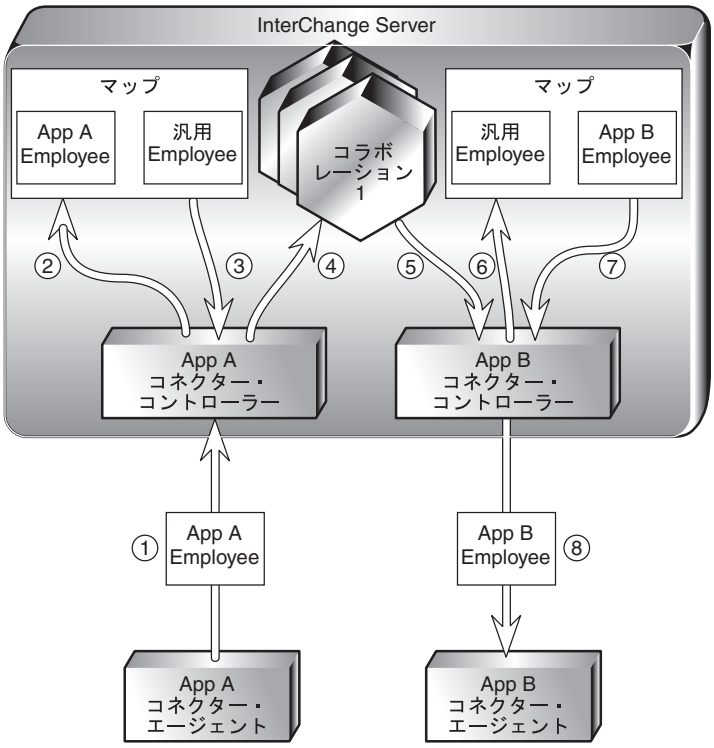

図 36. 実行時のデータ・マッピング

図 36 は、次の順序で示されています。

1. App A コネクター・エージェントが App A Employee ビジネス・オブジェクトを作成し、コネクター・コントローラーに送信します。
2. コネクター・コントローラーが App A Employee ビジネス・オブジェクトをマッピングのために InterChange Server に渡します。要求には、コネクター構成で指定されたマップ名に基づく、InterChange Server が使用する必要があるデータ・マップの名前が含まれます。

3. マップが汎用 Employee ビジネス・オブジェクトをコネクター・コントローラーに戻します。
4. コネクター・コントローラーが、汎用 Employee ビジネス・オブジェクトへのサブスクリプションを持つコラボレーションをチェックします。この例では、コラボレーション 1 がサブスクリプションを持っているので、コネクター・コントローラーはビジネス・オブジェクトをコラボレーション 1 に渡します。
5. コラボレーションは処理を実行した後、別の汎用 Employee ビジネス・オブジェクトを出力として作成し、コネクター・コントローラーに送信します。
6. コネクター・コントローラーが汎用ビジネス・オブジェクトを InterChange Server に渡して、App B Employee ビジネス・オブジェクトへのマッピングを要求します。
7. マップがアプリケーション固有のビジネス・オブジェクトをコネクター・コントローラーに戻します。
8. コネクター・コントローラーが App B ビジネス・オブジェクトを App B コネクター・エージェントに渡し、App B コネクター・エージェントがビジネス・オブジェクト内のデータを App B に渡します。

以上の例では、2 つのマップを使用しました。1 つはコラボレーションが使用する App A Employee ビジネス・オブジェクトから汎用 Employee ビジネス・オブジェクトへのマップ、もう 1 つは汎用 Employee ビジネス・オブジェクトから App B Employee ビジネス・オブジェクトへのマップです。Employee データは、App A から App B への 1 方向のみに移動しました。

Employee データを 2 つの異なるアプリケーション間で*両* 方向に交換する場合、次の 4 つのマップが必要になります。

- App A のアプリケーション固有のビジネス・オブジェクトから汎用ビジネス・オブジェクトへの、コラボレーションが使用するマップ
- 汎用ビジネス・オブジェクトから App B のアプリケーション固有のビジネス・オブジェクトへのマップ
- App B のアプリケーション固有のビジネス・オブジェクトから汎用ビジネス・オブジェクトへのマップ
- 汎用ビジネス・オブジェクトから App A のアプリケーション固有のビジネス・オブジェクトへのマップ

マップ・コンポーネントおよびツール

IBM WebSphere InterChange Server システムには、共通のデータ・フォーマット変更状況の処理方法を含む Java マッピング API が含まれます。 IBM WebSphere InterChange Server マッピングの 2 つの主要コンポーネント (マップおよび関係定義) を作成するために、グラフィカル設計ツールが提供されています。

- **マップ**には、1 つ以上のソース・ビジネス・オブジェクトから 1 つ以上の宛先ビジネス・オブジェクトへの属性の変換方法を指定する Java コードが含まれます。マップは、異なるアプリケーションとの変換を希望するすべてのビジネス・オブジェクトに必要です。ビジネス・オブジェクトを変更するときは、関連するマップも変更する必要があります。通常、変換するソース・ビジネス・オブジェクトごとに 1 つのマップを作成します。

マップの作成およびコンパイルを行うには **Map Designer** ツールを使用します。

- **関係定義**は、IBM WebSphere InterChange Server システムにおける複数のデータ・エンティティー間の関係を確立するものです。マップ内の関係定義は、類似の目的を持っているがアプリケーションごとに形式が異なるデータにおいて、ビジネス・オブジェクト属性を変換するために最も頻繁に使用されます。ほとんどのマップは、1 つまたは少数の関係定義を使用します。

  Relationship Designer ツールは関係定義の作成のために使用し、表スキーマは実行時の関係インスタンス・データの保管のために使用します。

マップと関係定義は、ともに InterChange Server のリポジトリーにあります。ビジネス・オブジェクト定義のように、関係定義は作成されたインスタンスの仕様またはテンプレートとして機能します。ビジネス・オブジェクトのインスタンスとは違って、関係インスタンスは永続的であり、関係ごとに特別な表に保管されます。

システムが要求を受信して指定されたビジネス・オブジェクトに変換するときは、関連するマップを実行して、変換の目的によっては関連する関係定義のインスタンスを 1 つ以上作成します。マップの実行中に作成された関係インスタンスには、関連する属性の実行時データが含まれます。このデータは関係表に保管されます。

マッピング・ツールの動作についての詳細は、「*マップ開発ガイド*」の第 1 章を参照してください。

マッピング変換

マップはソース・ビジネス・オブジェクトと宛先ビジネス・オブジェクトを関連付けます。また、変換される属性それぞれに 1 つずつ、一連の変換ステップが含まれます。各変換ステップには、属性の値を計算する Java コードが含まれます。

データ・マップでは、ソース属性から宛先属性への変換が単純な場合と、等価だが異なって表され、直接変換できないデータ・エンティティー間で、関係の確立および維持を要求できる場合があります。

単純な変換

単純なデータ変換の場合、属性の構造が異なるとしても、ソース属性と宛先属性の値は明確に対応し、類似の意味を持ちます。

単純なマッピングには、次のアクションが含まれます。

- 1 つのソース属性値を 1 つ以上の宛先属性値にコピーします。
- 1 つのソース属性値を複数の宛先属性値に分割します。
- 複数のソース属性値を 1 つの宛先属性値に結合します。
- 宛先ビジネス・オブジェクトに等価の属性がないソース属性値を無視します。

91 ページの図 37 に、これらの操作のいくつかを行う単純なマッピングの例を示します。

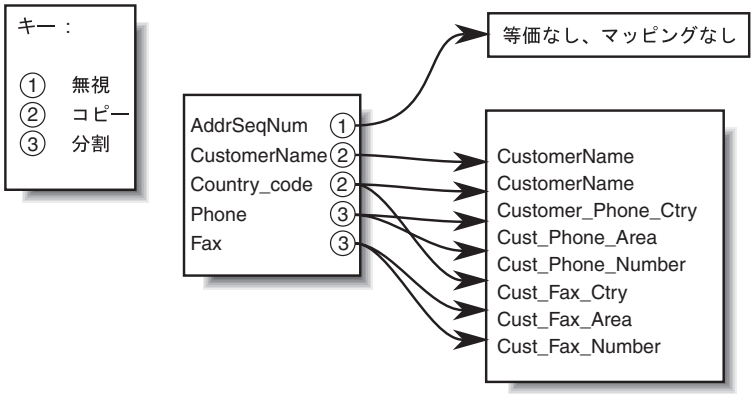

図 37. 単純なマッピング

関係変換

属性の中には、単純な変換を実行できないものもあります。異なるアプリケーションが、等価で類似する目的の情報を含むが、形式または値に互換性のない属性を持っている場合があります。例えば、Country 属性に、あるアプリケーションは 2 文字コード (US、FR、EG など) を使用し、別のアプリケーションは数表示 (1、2、3 など) を使用していることがあります。

異なるアプリケーション間でこのような属性を関連付けるには、**関係定義**を作成して、ソース属性と宛先属性のデータを関連付けます。

関係の種類

関係定義を作成するときには、関係に含まれる参加者 (ビジネス・オブジェクトまたはその他のデータ・エンティティー) をリストし、各参加者のタイプ (ビジネス・オブジェクト定義の名前または Data) を指定します。定義に関連している参加者のタイプと各参加者のインスタンス数に基づいて、関係は次のカテゴリーに分類されます。

- **一致関係**は、複数のビジネス・オブジェクト間、または 1 対 1 に基づいたその他のデータ間の関係を確立します。すべての関係インスタンスにおいて、各参加者に 1 つのインスタンスのみが存在します。一致関係は一般的に、ID 番号や製品コードなどビジネス・オブジェクトのキー属性を変換します。
- **非一致関係**は、複数のビジネス・オブジェクト間、または 1 対多、多対多に基づいたその他のデータ間の関係を確立します。すべての関係インスタンスにおいて、各参加者に 1 つ以上のインスタンスが存在します。非一致関係の例としては、RMA 対 Order の変換があります。単一の RMA (Return Materials Authorization) ビジネス・オブジェクトは、1 つ以上の Order ビジネス・オブジェクトを抱えることができます。
- **参照関係**は、ビジネス・オブジェクトの属性などのデータ間の関係を確立します。データは 1 対 1、1 対多、または多対多に基づいて関連付けられます。参照関係の参加者は、`Data` タイプです。参照関係は一般的に、既婚/独身や通貨コードなどのコードを表す値を持つ非キー属性を変換します。

関係定義についての詳細は、「*マップ開発ガイド*」の第 1 章を参照してください。

マップを使用したコネクターの構成

System Manager を使用してコネクターをインストールおよび構成するときは、コネクターがサポートする各ビジネス・オブジェクトを変換するために、InterChange Server のリポジトリーに保管されている IBM WebSphere InterChange Server マップの名前を指定します。

コネクター・コントローラーがマッピングのためにビジネス・オブジェクトを送信するとき、コネクター・コントローラーはマップを実行して、ビジネス・オブジェクトとマッピング情報を入力として送信します。

要約

この章では、ビジネス・オブジェクト・マッピングに関連する用語および概念について説明しました。覚えておくべき重要な点は次のとおりです。

- データ・マッピングはビジネス・オブジェクトを変換します。ビジネス・オブジェクトがソース・アプリケーションからコラボレーションに送られるとき、マッピングはアプリケーション固有のビジネス・オブジェクトを汎用ビジネス・オブジェクトに変換します。ビジネス・オブジェクトがコラボレーションから宛先アプリケーションに送られるとき、マッピングは汎用ビジネス・オブジェクトをアプリケーション固有のビジネス・オブジェクトに変換します。
- マップは、Map Designer および Relationship Designer ツールを使用して作成できます。
- 単純なマッピングは Map Designer で実行できます。等価だが互換性のない形式の属性を含むマッピングには、Relationship Designer を使用します。

ここまでのすべての章では、IBM WebSphere InterChange Server システムの基本機能について説明しました。次の章では、一部のユーザーに関係のある拡張機能について説明します。

第 7 章 トランザクション・コラボレーション

トランザクション・コラボレーションは、ビジネス・プロセスのためにデータの整合性の保証を提供します。この章では、トランザクション・コラボレーションの機能について説明します。この章は次のセクションから構成されます。

トランザクション・モデル

このセクションでは、IBM WebSphere InterChange Server トランザクション・モデルの基礎となる概念について簡単に説明します。データベースおよびトランザクション処理についての知識をお持ちの方は、次のセクションに進んでください。

トランザクションは、作業論理単位を形成する、関連のある相互依存ステップの集合です。複数のステップを 1 つのトランザクションにグループ化することによって、ステップの集合全体が成功するときも失敗するときも、ステップの集合が 1 つのアトミック単位として処理されます。

図 38 に、update 1 と update 2 の 2 つの更新を含むデータベース・トランザクションを示します。

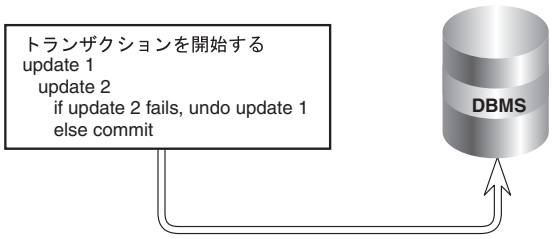

図 38. データベース・トランザクション

トランザクションが実行されているとき、トランザクションは変更中のデータベースをロックして、他のトランザクションが変更のためにそのデータを使用できないようにします。外部の干渉からデータを分離することは、トランザクションに特有の性質の 1 つです。

トランザクションの操作が成功した場合は、**コミット**操作で変更内容をディスクに書き込むことによりトランザクションが完了します。コミット操作はトランザクションのロックを解除し、他のトランザクションが更新済みのデータを使用できるようにします。

実行途中でエラーが発生した場合、トランザクション全体の成功は不可能になり、トランザクション全体が失敗します。トランザクションはデータベースに部分的な結果を残すのではなく、すでに実行された変更をバックアウトして、データベースをトランザクションが開始する前の値に戻します。このバックアウト・プロセスは、**ロールバック**と呼ばれます。

もう 1 つのタイプのデータベース・トランザクションは 2 フェーズ・コミットで、複数のデータベース上での分散使用に適しています。2 フェーズ・コミットはトランザクション・モニターを使用して、並行更新を調整します。トランザクション・モニターは、最初にすべてのデータベースが必要な変更を実行できるかどうかチェックします。すべてのデータベースが変更を実行できるとしても、トランザクション・モニターのシグナルがそれを表示するまで待機する必要があります。変更を実行できないデータベースが 1 つでもある場合は、変更はまったく実行されません。

すべてのトランザクションはデータベース内でデータの整合性を維持する必要がありますが、2 フェーズ・コミット・プロトコルは個々のデータベースを越えてこの要件の範囲を拡張します。一般的な例は資金の送金です。この場合、トランザクション・モニターは、あるアカウントから引き出された資金が他のアカウントに振り込まれたか、またはどちらのアカウントも変更されていないことを確認します。

トランザクション・コラボレーションとは

トランザクション・コラボレーションは、データ変更をロールバックできるコラボレーションです。データベース・トランザクションと同様に、トランザクション・コラボレーションは全か無かの操作です。つまり、コラボレーション全体が成功するか、コラボレーション全体が失敗するかのいずれかです。さらに、トランザクション・コラボレーションは、データ操作ロジックを損なう可能性のあるデータ分離違反を検出してそれに反応できます。

トランザクション・コラボレーションは、データベース・トランザクションと 2 フェーズ・コミット・プロトコルで確立している原則に基づいています。ただし、トランザクション・コラボレーションには、データベース・トランザクションとは異なるコラボレーション固有の特性 (任意の数のアプリケーションへの分散、非同期、長期存続、アプリケーションの内部ではなく外部に置かれる、など) があります。

トランザクション・コラボレーションと非トランザクション・コラボレーションは、実行方法が異なります。トランザクション・コラボレーションは、InterChange Server のトランザクション・サービスの制御下で実行します。トランザクション・サービスは、実行、ロールバック、および分離チェックを制御します。

このセクションでは、トランザクション・コラボレーションのコンポーネントについて説明します。

トランザクション・シナリオ

コラボレーション・テンプレートの「最小トランザクション・レベル」プロパティーは、コラボレーションがトランザクションかどうかを決定し、すべてのシナリオに適用されます。テンプレートから作成されたすべてのコラボレーション・オブジェクトは、最小トランザクション・レベルを継承します。

実際のトランザクション・ロジックは、シナリオ・レベルに適用されます。トランザクション・コラボレーションにおいて、各シナリオは**トランザクション・シナリオ**と呼ばれ、シナリオの開始はトランザクションを暗黙的に開始し、シナリオの正常終了はトランザクションを暗黙的にコミットします。

各シナリオは、単一の実行コンテキスト内で実行されます。コラボレーションがコラボレーション・グループの一部の場合、シナリオは別のコラボレーションを呼び出して作業を実行し、戻すことがあります。呼び出されたシナリオは呼び出し元の実行コンテキスト内で実行され、同一のトランザクションの一部となります。

サブトランザクション

シナリオ内のステップによってアプリケーションのデータが変更されるとき、ステップは自身でアプリケーションのトランザクションを開始します。アプリケーション・トランザクションを引き起こすステップを、**サブトランザクション・ステップ**といいます。これはトランザクション内のトランザクションで、大きなトランザクションはそれ自身がシナリオになります。

トランザクションの概念を示すため、この章では 図 39 のシナリオを使用します。このシナリオは、価格設定管理コラボレーションの一部として更新を扱います。コラボレーションは、ERP アプリケーションが製品の価格を変更すると、その変更を顧客サービス・アプリケーションおよび製品構成アプリケーションに突き合わせます。図では、トランザクションの意味に関係ない処理の詳細をすべて省略しています。

シナリオにはアクション 1 (A1) からアクション 4 (A4) まで 4 つのステップがあり、A、B、および C の 3 つのビジネス・オブジェクトと対話します。ビジネス・オブジェクト A は ERP システム内の特定の製品を表し、B は顧客サービス・アプリケーション内の同一製品を表し、C は製品構成アプリケーション内の同一製品を表します。

注: *BusinessObject.Attribute* の形式では、ビジネス・オブジェクトの特定の属性を表します。例えば A.Price は、ビジネス・オブジェクト A の Price 属性を表します。

シナリオを左側に示します。右側のコメントには、製品マネージャーが ERP システムにログインして製品価格を 700 ドルに変更したときに発生する内容が示されています。

注: 単純化するため、この章の図ではコネクターを省略します。

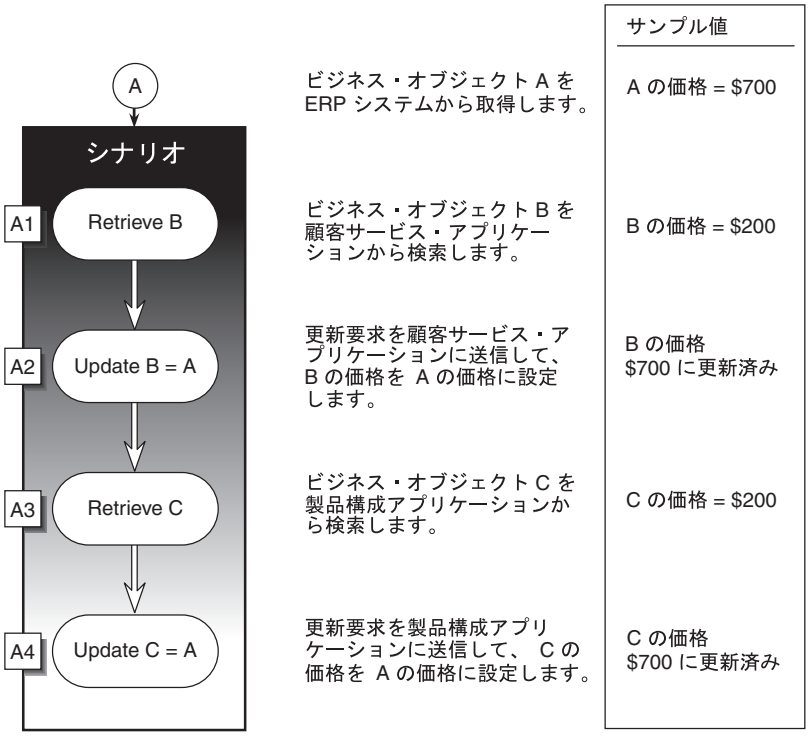

図 39. サンプル・シナリオ

シナリオをトランザクション処理するには、まずサブトランザクション・ステップを識別します。図 40 に、シナリオのサブトランザクション・ステップが A2 と A4 であることを示します。これは、A2 が顧客サービス・アプリケーション、A4 が製品構成アプリケーションのトランザクションをそれぞれ引き起こしているからです。

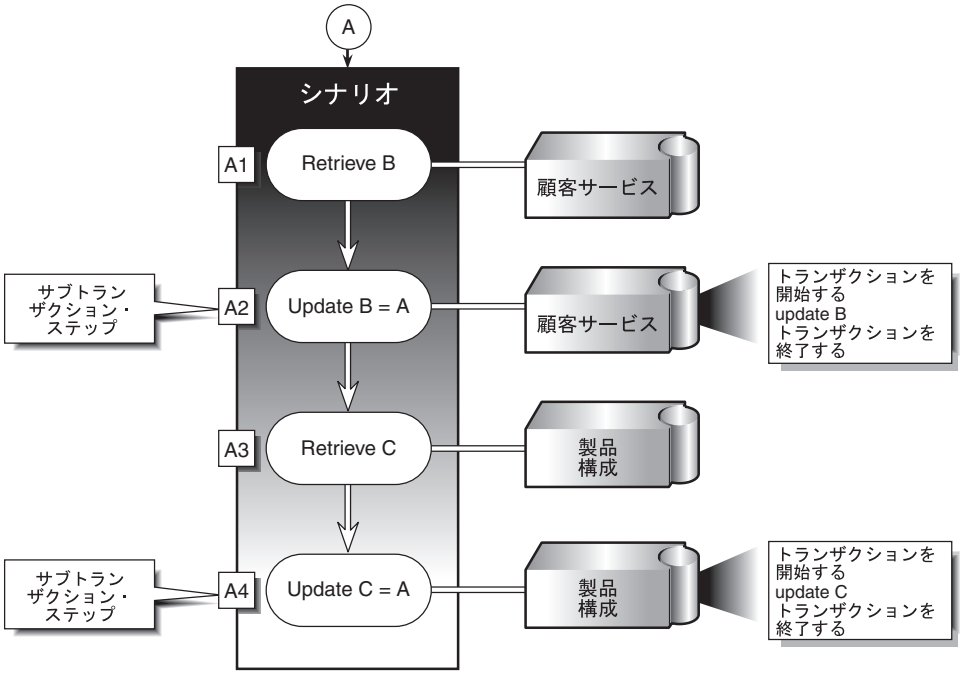

図 40. トランザクション・シナリオのサブトランザクション・ステップ

A1 や A3 のように要求の検索を実行するステップは、データの変更を伴わないので、サブトランザクション・ステップではありません。

差し戻しとロールバック

トランザクション・シナリオの障害への応答方法は、非トランザクション・シナリオの障害への応答方法とは異なります。非トランザクション・シナリオが失敗すると、コラボレーションは単にエラーを記録して終了します。トランザクション・シナリオが失敗すると、シナリオはロールバックして、関連するデータベース全体でデータを整合性のある状態にします。

ロールバックでの差し戻しの使用

エラーが発生したとき、サブトランザクションによってすでにアプリケーションでの作業のコミットが実行されている場合があります。このため、他のアクションの影響を取り消すアクションである**差し戻しステップ**を使用して、ロールバックを実行します。差し戻しステップは、ロールバック中にのみ実行されます。

ロールバック中、InterChange Server はすべての実行パスを逆方向にたどります。完了したそれぞれのサブトランザクション・ステップに対して、サーバーが関連する差し戻しを実行します。すべての実行済みトランザクション・ステップが差し戻されると、ロールバックが完了します。ロールバック中にエラーが発生した場合、InterChange Server は単にエラーを記録します。

差し戻しステップは、コラボレーション開発者が元のアクションを取り消すために使用する任意のアクションから構成されます。表 8 に、コラボレーション開発者が特定のアクションの差し戻しに使用する可能性のある共通の方法を、いくつかリストします。

表 8. 差し戻しの例

アクション	差し戻し
ビジネス・オブジェクトを作成	ビジネス・オブジェクトを削除
ビジネス・オブジェクトを削除	ビジネス・オブジェクトを作成
ビジネス・オブジェクトを更新	以前の値を復元して、ビジネス・オブジェクトを更新

一般的に、差し戻しは元のアクションによるデータ変更を元に戻す作業から構成されますが、別の場合もあります。例えば、作成要求に対する差し戻しは、別の作成要求であることがあります。この場合、レコードに監査ログが書き込まれます。したがって、差し戻しは**論理的な取り消し**操作であり、実際の取り消し操作であるとは限りません。

ロールバックの設計

コラボレーション設計者は、ロールバックを使用できるようにシナリオを設計する必要があります。図 41 に、サンプル・シナリオがロールバックを許可するように変更された状態を示します。

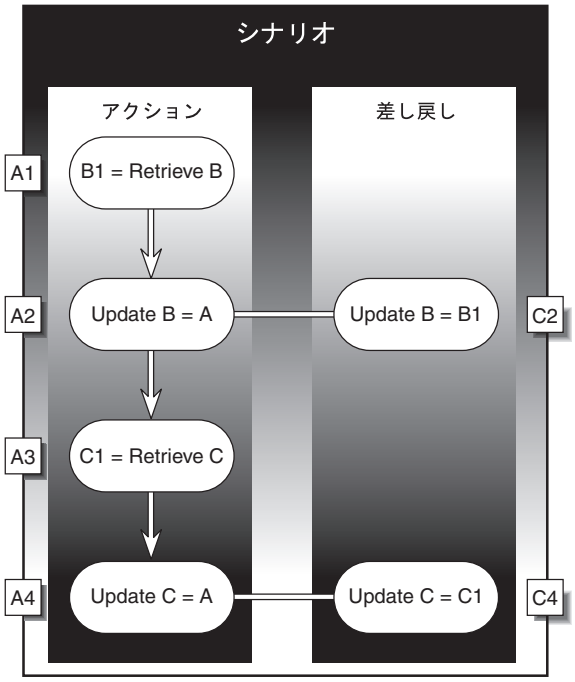

図 41. アクションと差し戻し

この例は、次のように変更されています。

- アクション・ステップ A1 と A3 は、それぞれ B^1 と C^1 を使用して B と C の元の値を保管します。
- 差し戻しステップ C2 と C4 は、B と C の元の値を復元するという新規の差し戻しステップです。

A2 の実行後に、シナリオにエラーが発生したとします。ロールバックの間、差し戻しステップ C2 が B^1 に保管された値を使用して、元の値を B に戻します。

実行時の図

以下の図は、サンプル・シナリオの実行例を示しています。図に示された実行時シーケンスでは、アクション・ステップ A1、A2、および A3 が正常に実行されます。しかし、A4 の実行中にアプリケーションでエラーが発生します。シナリオが A4 の正常状況ではなくエラーを受信すると、シナリオは失敗し、ロールバックが開始されます。

ステップをさかのぼり、トランザクション・サービスは次のことを実行します。

- A4 はデータ変更を完了していないので、差し戻しの必要はありません。
- A3 は検索操作なので、差し戻しの必要はありません。
- A2 はデータ変更を完了し、定義された差し戻しを持っているので、差し戻し C2 が実行されます。

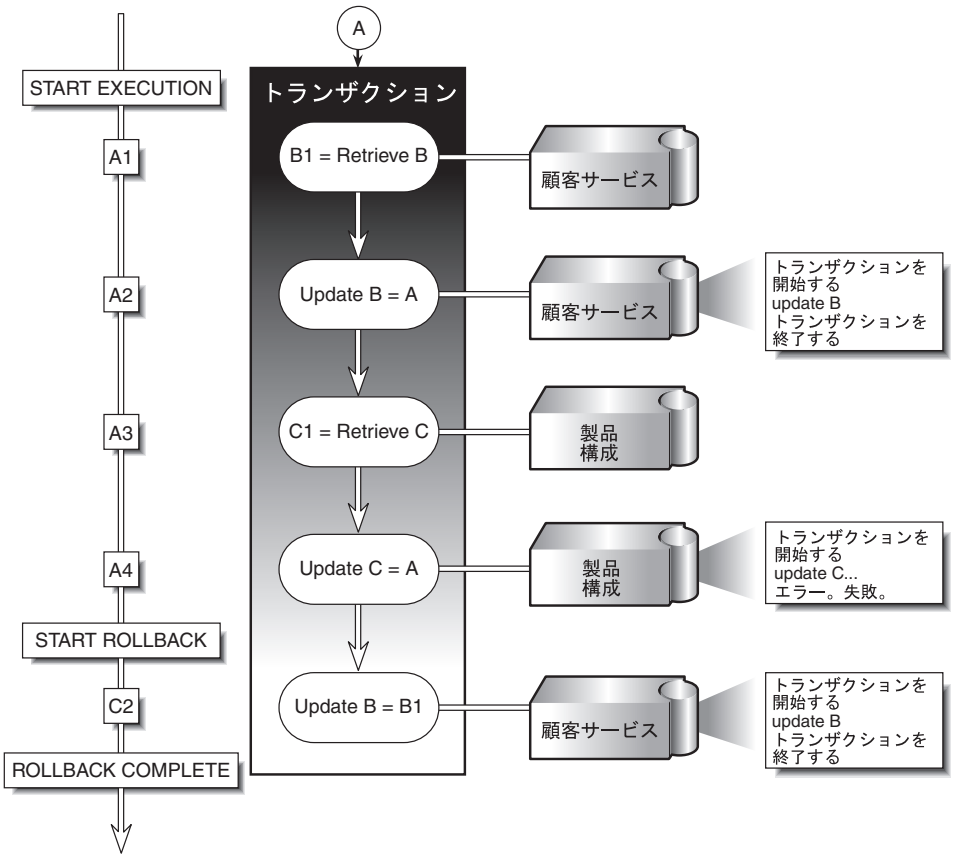

図 42. トランザクション・シナリオのロールバック

データ分離

IBM WebSphere InterChange Server システムは、コラボレーションにいくつかのトランザクション実行レベルを提供します。より高度なレベルでは、InterChange Server が分離違反を検出します。分離とは、トランザクションの実行中に、そのトランザクションがデータへの排他的アクセス権を持つことの保証です。トランザクションのデータはトランザクション内の操作間で変更されないので、トランザクションのロジックの影響は予測可能です。

データベース・トランザクションまたは 2 フェーズ・コミットでは、データベース・ロックによってトランザクション実行中の分離が保証されます。IBM WebSphere InterChange Server 環境では、トランザクション・シナリオはステップ間のデータをロックできません。各ステップがアプリケーション・トランザクションを引き起こし、完了するとロックを解除するからです。ロックが解除されると、他のプログラムは更新されたデータを表示したり変更したりできます。

シナリオで、同一のデータを複数回使用する必要がある場合があります。例えば、複数のアクション・ステップで同一のデータを更新することがあります。また、シナリオが失敗してロールバックする場合、差し戻しステップが直前のアクションで設定されたデータに再度アクセスして、元の値に復元することがあります。

シナリオが複数回使用するデータにとって、あるステップが引き起こしたアプリケーション・トランザクションの完了と別のステップが引き起こしたアプリケーション・トランザクションの開始との時間間隔は、**ぜい弱な時間帯**です。

図 43 に、同一のビジネス・オブジェクトを更新する 2 つのトランザクション・ステップ間のぜい弱な時間帯を示します。

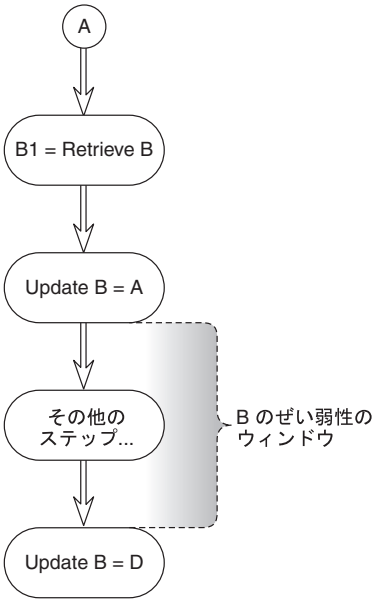

図 43. ぜい弱な時間帯

ぜい弱な時間帯の間、データの分離は保証されません。他のトランザクションによって、シナリオがアクセスしていない間にデータの変更が行われると、シナリオの結果にエラーが生じる場合があります。

コラボレーションによっては、分離を必要としないものもあります。例えば、データを変更するプログラムにすぎないコラボレーションです。ただし、シナリオが他のプログラムも変更を行うデータを変更する場合、分離違反が問題になります。

IBM WebSphere InterChange Server システムは、分離違反からの保護を必要とするコラボレーションに**分離チェック**を提供します。コラボレーションが「最大限の努力」または「緊急」トランザクション・レベルで実行されている場合、トランザクション・サービスとコネクターは協働して、データが複数のアクセスの間に損なわれていないか判断します。次のセクションでは、トランザクション・レベルおよび各レベルで分離チェックが発生する状態について説明します。

トランザクション・レベル

コラボレーションのトランザクション・レベルは、システムがコラボレーション内でシナリオを実行するときに使用する機構を決定します。

コラボレーション開発者がコラボレーションにトランザクションの実行が必要かどうかを決定する場合、コラボレーションは開発段階でトランザクションになります。必要だと決定すると、開発者は差し戻しステップをテンプレートのシナリオに追加して、コラボレーションのトランザクション・レベルを指定します。すべてのコラボレーションは、次のトランザクション・レベルのうちのいずれかに指定されます。

- なし
- 最小限の努力
- 最大限の努力 (このレベルは長期存続コラボレーションの場合はサポートされないので注意してください)
- 緊急

System Manager は構成中に、コラボレーション・テンプレートの最小トランザクション・レベルを表示します。これは、コラボレーション・オブジェクトがそのテンプレートの継承から作成されるからです。

しかし、コラボレーション・オブジェクトが実際に実行するトランザクション・レベルは、他の要素によって影響されます。構成時に、管理者はコラボレーション・オブジェクトをコネクターまたはコラボレーション・オブジェクト、あるいはその両方にバインドします。コネクターまたはコラボレーション・オブジェクトは、それぞれ特定のトランザクション・レベルを持っており、コラボレーションはすべてのバインディングがサポートするレベルで実行する必要があります。例えば、コネクターのトランザクション・レベルは、アプリケーションの機能に基づいて、コネクターが提供できるサポートのレベルを表します。コラボレーションは、コネクターのうち 1 つでもサポートしないレベルでは実行できません。

なし

トランザクション・レベルが「なし」のコラボレーションは、トランザクション・コラボレーションではありません。コラボレーションの実行中にエラーが発生した場合、エラー・メッセージが記録されます。

コラボレーション・テンプレートに差し戻しステップが含まれていても、そのテンプレートから作成されたコラボレーション・オブジェクトはトランザクション・レベルなしで実行できます。例えば、コネクター・オブジェクトのバインディングがより高度なトランザクション・レベルをサポートしない場合が考えられます。この場合、差し戻しは無視されます。

最小限の努力

「最小限の努力」トランザクション・レベルのコラボレーションには、シナリオのトランザクション・ステップに定義された差し戻しが含まれます。シナリオの実行中にエラーが発生すると、InterChange Server はシナリオをロールバックし、実行済みのトランザクション・ステップに対する差し戻しを実行します。

シナリオが異なるコラボレーション・オブジェクトを呼び出して作業の一部を実行させた場合、つまりコラボレーションがコラボレーション・グループの一部である場合、コラボレーション内で実行されたシナリオもロールバックします。

「最小限の努力」トランザクション・レベルは、次の条件下にあるコラボレーションに適しています。

- コミットされた作業を取り消すために、障害時のロールバックが重要である。
- コラボレーションのシナリオが、他のプログラムまたはコラボレーションが変更を行わないデータを変更するので、分離チェックの必要がない。

最大限の努力

「最大限の努力」トランザクション・レベルで実行するコラボレーションは、分離チェックを使用します。「最大限の努力」トランザクション・レベルは、データの不整合から保護する必要のあるコラボレーションに適しています。

コラボレーションが分離チェックを使用して実行すると、次のことが実行されます。

- InterChange Server のトランザクション・サービスが、シナリオの実行中にシナリオが再アクセスしたデータを、コネクターとともに動作してチェックします。
- システムが、後続の操作を適用する前に、データの現在の状態と最後にチェックした状態とを照合します。
  - データが直前にチェックしたときと同じである場合、データは仮想分離され、操作が続行します。
  - データが直前にチェックしたときと同じでない場合、操作はエラーになります。

分離チェックは適切なトランザクション・セマンティクスを提供しますが、InterChange Server での新たなデータベース・アクセスや、コネクターと InterChange Server 間のより一層の通信が必要になります。この追加のシステム作業は、パフォーマンスに影響を与えます。このため、トランザクション・レベルを選択するときは、パフォーマンスの問題とデータの整合性の問題を比較検討する必要があります。

しかし、「最大限の努力」トランザクション・レベルでさえ、小さなぜい弱な時間帯は存在します。分離チェックの開始とコラボレーションが操作を継続する時間の

間、別のトランザクションがデータを変更する可能性があります。それでも、「緊急」トランザクション・レベルをサポートするアプリケーション API は少数なので、「最大限の努力」トランザクション・レベルは、多くのアプリケーションの組み合わせで使用できる最も高度なトランザクション・レベルです。

注: 「最大限の努力」トランザクション・レベルは長期存続コラボレーションの場合はサポートされません。

緊急

「緊急」トランザクション・レベルは最も高度な分離保証ですが、これを使用できるのは、クライアント・アプリケーションにアトミック・テストの実行と操作の設定を許可する API を持つアプリケーションのみです。このようなアプリケーションとともに作動するコネクターは分離チェック中にデータをロックするので、他のプログラムがシナリオのデータを変更する可能性がなくなります。現在、この機能を提供するアプリケーションはごくわずかです。

リカバリー

あらゆるソフトウェア・プログラムは、予期せず実行を停止させるハードウェア・イベントまたはソフトウェア・イベントによる中断のリスクを抱えています。InterChange Server は、予期しない終了が発生した際に進行中のトランザクションをリカバリーする堅固な機構を持っています。

InterChange Server が予期しない終了から復帰すると、終了時にアクティブ・トランザクション状態だったコラボレーションをチェックします。次に、2 段階のリカバリーが開始します。

1. InterChange Server が、中断されたトランザクション・コラボレーションを再びアクティブ化およびロールバックします (コラボレーションが長期存続ビジネス・プロセス機能を使用する場合を除く)。この間、サーバーはコラボレーションに新規イベントを送達しません。
2. InterChange Server が、中断されたコラボレーションの元のトリガー・イベントをイベント管理サービスから検索し、それを再送達します。コラボレーションが実行し、イベントを再処理します。

リカバリーが完了すると、InterChange Server はコラボレーションでの新規イベントの処理を許可します。

トランザクション・コラボレーションと長期存続ビジネス・プロセス

トランザクション・コラボレーションは、長期存続ビジネス・プロセス機能を使用できます。長期存続ビジネス・プロセス機能は、コラボレーション・テンプレートの作成時にオプションで使用可能にできます。この機能を使用可能にしているテンプレートからコラボレーション・オブジェクトを作成した場合は、フローがサービス呼び出し応答を待機している間に InterChange Server が破損すると、リカバリー時にそのフローが、保管済みのコンテキストから継続されます。一方、その機能を使用可能にしていないときは、InterChange Server が破損すると差し戻しロールバックが起動されます。

要約

この章では、トランザクション・コラボレーションの概要について説明しました。覚えておくべき重要な点は次のとおりです。

- IBM WebSphere InterChange Server トランザクション・モデルは、DBMS 環境でのトランザクションの使用から派生しています。ただし、IBM WebSphere InterChange Server モデルはコラボレーションの固有の性質に適合しています。
- トランザクション・コラボレーションは、ロールバックを処理できるシナリオを備え、InterChange Server のトランザクション・サービスの制御下で実行します。
- トランザクション・コラボレーションのシナリオが失敗すると、InterChange Server はシナリオをロールバックし、完了したアクションの差し戻しを実行します。
- 高度なトランザクション・レベルでは、システムは分離チェック機能も提供します。
- トランザクション・コラボレーションのリカバリー機構は、システム障害やその他の予期しないイベントの後でも、InterChange Server がコラボレーションを不明のトランザクション状態のままにしないことを保証します。

第 8 章 言語固有の振る舞いのサポート

WebSphere InterChange Server および WebSphere Business Integration Adapters (WebSphere Business Integration 製品) は、*双方向使用可能化* により、ヘブライ語やアラビア語などの双方向言語をサポートします。双方向使用可能化は、WebSphere Business Integration 製品にバンドルされているコンポーネント (コネクター、アダプター・フレームワーク、コラボレーション、およびマップ) 内の双方向スクリプト・データを正しく表示および処理するための機構です。WebSphere Business Integration 製品はすべての双方向形式のデータを処理できますが、WebSphere Business Integration 製品ドメイン内のデータは、Windows 標準の双方向形式 (論理的に左から右) である 1 つの単一の双方向形式に設定されます。

この章では、WebSphere Business Integration 製品における双方向言語サポートおよび双方向言語サポートを使用可能化するために使用する処理の概要について説明します。この章は次のセクションから構成されます。

- 『WebSphere Business Integration 製品でのロケール・サポート』
- 109 ページの『双方向スクリプト・サポート』
- 113 ページの『レイアウト変換および属性』
- 114 ページの『WebSphere Business Integration 製品での双方向スクリプトの使用可能化』
- 119 ページの『設計上の制限』

WebSphere Business Integration 製品でのロケール・サポート

ロケール は、特定の国、言語、または地域に固有のデータの処理方法に関する情報をまとめたユーザーの環境の一部です。通常、ロケールは、オペレーティング・システムの一部としてインストールされます。

ロケールにより、ユーザー環境に次の情報が提供されます。

- 言語および国 (または地域) に応じた次の国/地域別情報。
  - データ・フォーマット:
    - 日付: 曜日と月の完全な名前と省略名、および日付の構造 (日付の分離文字を含む) を定義します
    - 数: 3 桁ごとの区切りと小数点に使用する記号、および数値内に 3 桁ごとの区切り記号を置くかどうかを定義します
    - 時刻: 12 時制の場合の標識 (午前と午後の標識など)、および時刻の構造を定義します
    - 金額: 数字と通貨記号、および金額内でのこれらの記号の配置位置を定義します
  - 照合順序: 特定の文字コード・セットおよび言語でのデータのソート方法
  - ストリングの取り扱い。大文字と小文字、比較、サブストリング、連結などの操作

- 文字エンコード: 文字 (アルファベット) から文字コード・セットの数値へのマッピング。例えば、ASCII 文字コード・セットでは、文字 A を 65 としてエンコードしますが、EBCIDIC 文字セットではこの文字を 43 としてエンコードします。*文字コード・セット* には、1 つ以上の言語のすべての文字のエンコードが含まれます。

ロケール名の形式は次のとおりです。

```
11_TT.codeset
```

ここで、11 は 2 文字の言語コード (通常は小文字)、TT は 2 文字の国および地域コード (通常は大文字)、codeset は関連する文字コード・セットの名前です。名前の codeset 部分はオプションです。通常、ロケールは、オペレーティング・システムの一部としてインストールされます。

ロケールの設定

ロケール依存データを正しく処理できるようにするには、デフォルトのユーザー・ロケールを双方向ロケール (アラビア語またはヘブライ語) に設定しておく必要があります。ユーザーのデフォルト・ロケールを双方向ロケールに設定しないと、双方向文字が正しく表示されません (詳しくは、「*システム・インストール・ガイド (Windows 版)*」を参照してください)。ヘブライ語の場合の Windows ロケール設定は、通常 iw_IL であり、アラビア語の場合は通常 ar_AE です。アラビア語の場合の ASCII コード・セットは 1256 であり、ヘブライ語の場合は 1255 です (双方向ロケールの設定方法について詳しくは、「*システム・インストール・ガイド (Windows 版)*」を参照してください)。

ロケール依存データの処理

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、ほとんどの既知の文字コード・セット (単一バイトとマルチバイトの両方) のエンコードが含まれています。WebSphere Business Integration 製品のほとんどのコンポーネントは Java で作成されています。したがって、ほとんどの WebSphere Business Integration 製品コンポーネント間でデータを転送する場合は、文字を変換する必要はありません。

外部アプリケーションと WebSphere Business Integration 環境の間でデータを転送する場合は、2 バイトの Unicode ではなく 1 バイトの文字になることがあるため、形式どうしで適切な文字変換を行う必要があります。この事態に対処するために、コネクターは国際化されて 2 バイトと 1 バイトの両方の文字セットをサポートしており、指定の言語にメッセージ・テキストを送達できるようになっています。統合ブローカーは Unicode を使用するため、コネクターは、文字コード・セットを使用するロケーションからブローカーにデータを転送するときに、1 バイトのコード・セットから 2 バイトのコード・セット (Unicode) にデータを変換します。

ログ・メッセージおよび通知メッセージには 1 バイト・コード・セットのデータが含まれるため、Unicode と 1 バイト・コード・セットの間で双方向テキストを正しく変換するには、ロケールを適切に設定する必要があります。したがって、適切な言語で適切な国または地域のためにエラー・メッセージおよび通知メッセージをログに記録するには、使用している環境のロケール標準構成プロパティーを特定の双

方向言語設定に構成してください。構成プロパティーについて詳しくは、「システム・インストール・ガイド *(Windows 版)*」のロケールの設定のセクションを参照してください。

注: 必ずデフォルトのコード・ページを DOS プロンプトで正しく設定してください。コード・ページが正しく設定されていないと、表示されるデータが標準の Windows エディターやビューアーで読み取れない場合があります (「システム・インストール・ガイド *(Windows 版)*」のヘブライ語およびアラビア語の場合の ASCII 設定の DOS プロンプト・コード・ページの変更のセクションを参照してください)。

設計上の考慮事項

WebSphere Business Integration 環境では、エンコードは次の 3 つのレベルで処理されます。

- コンテンツ・データ
- メタおよび構成データ
- ログおよびトレース・データ

コンテンツ・データのレベルでは、ビジネス・オブジェクトの実行時インスタンス・データを処理します。メタおよび構成のレベルでは、外部アプリケーションと WebSphere Business Integration 環境の間の通信を確立して管理するためにコネクターが使用するデータを処理します。ログおよびトレース・データのレベルでは、さまざまなログ・ファイルおよびトレース・ファイルに記録される通知メッセージを処理します。一部のアダプターは柔軟なエンコード・サポートを提供しています。例えば、XML コネクターには、完全なエンコード・サポートを提供する XML ファイル・ヘッダーのエンコード仕様を処理する機能があります。一方、SAP 用コネクターには、SAP クライアント API が提供する限定されたエンコード・サポートしかありません。コンテンツ・レベルとメタおよび構成データ・レベルでは、コネクターに構成オプションが提供されている場合は、エンコード指定オプションによってロケール設定が上書きされます。ログおよびトレース・データ・レベルでは、デフォルトの DOS プロンプト・コード・セットを使用して出力データをエンコードします。

コンテンツ・データ・エンコード

コネクター間でコンテンツ・データ・エンコードをサポートするための一般化されたアプローチはありません。コネクターがコンテンツ・データ・エンコード・サポートを処理するさまざまな手法を扱うために使用できる基本的なアプローチは 3 つあります。これらのアプローチは次のとおりです。

- ビジネス・オブジェクト・データの直列化および非直列化に使用するミドルウェアまたはサード・パーティーの API (SAP/PS クライアント API や RDBMS 用 JDBC ドライバーなど) によるエンコードの自動処理
- コネクターまたはデータ・ハンドラー用の適切な構成を持つ下位テクノロジーの使用 (XML DH の XML ヘッダーの charset パラメーターや Email コネクターでの添付ファイルごとの MIME 文字指定の使用)

- WebSphere Business Integration 製品での内部的なエンコードの処理可能化、および JText コネクターで使用する MO_JTextConnector_Default メタ・ビジネス・オブジェクトの DataEncoding 属性などの提供されている機構による指定

これらの 3 つのアプローチにより、アダプターによってコンテンツ・データを処理する手段が提供されます (使用するアダプターに適用できるアプローチについては、アダプター・ガイドを参照してください)。

メタ構成データ・エンコード

多くの場合、メタ構成データのエンコード・サポートは、外部アプリケーションへの接続をセットアップするために使用するミドルウェア・アダプターのコードの機能に依存します。

(アダプター・テンプレートまたはサポートされるビジネス・オブジェクト・テンプレートの一部として) WebSphere Tools で ASCII でエンコードされたメタおよび構成データは、Unicode でリポジトリーに格納されます。実行時には、このデータをネイティブ Java API またはサード・パーティー API のいずれかの引き数として使用して、外部ソースとの通信リンクを確立します。アダプター・コードで通信リンクを確立するために使用するミドルウェアは、ネイティブの Java API やパッケージ (Email や WebSphere MQ Connector など) をはじめ、サード・パーティー API (PeopleSoft など) にすることもできます。さらに、ミドルウェアに複数のレイヤーを持たせ、非 Windows システムの File System フォルダーや JText のようにアダプター Java コードを外部ソースから分離できます。File System フォルダーに Windows プロセスからアクセスする場合は、Java API から出された呼び出しに対するサービスがソフトウェアによって間接的に提供され、ターゲット Windows File System の代わりに非 Windows 区画をマウントできます。ネイティブ Java API、サード・パーティー API、ミドルウェアのいずれも明示的なエンコード指定をサポートしていることが保証されないため、WebSphere Business Integration 製品は、メタおよび構成双方向データの場合は Unicode エンコードのみをサポートします。

ログおよびトレース・データ・エンコード

デフォルトでは、ログ・ファイルおよびトレース・ファイルに書き込まれる双方向データのエンコードは、プロセスを起動する DOS プロンプトのコード・ページと同じです。しかし、アラビア語およびヘブライ語の双方向ロケールに関連したデフォルトの DOS プロンプトのコード・ページは、Windows オペレーティング・システムでのこれらの言語に対する標準のコード・ページ (1256 および 1255) と異なります。結果として、WebSphere Business Integration Java プロセスの DOS プロンプト・コード・ページが変更されていない場合は、これらのプロセスによってログ・トレース・ファイルに出力された双方向データは、双方向言語データの表示に標準のコード・ページを使用するほとんどの Windows ビューアーでは正しく表示されません (詳しくは、「システム・インストール・ガイド *(Windows 版)*」の 『DOS プロンプト・コード・ページの変更』を参照してください)。

双方向スクリプト・サポート

WebSphere Business Integration 製品は、双方向スクリプトの言語に対する双方向言語サポートを提供します。双方向スクリプトには、右から左に書かれるテキストと、左から右に書かれるセグメントの両方が含まれます (左から右に書かれるテキストには、文中に埋め込まれる数字や、英語、フランス語、キリル文字の諸言語、ギリシャ語などのローマ字ベースのスクリプトのテキストがあります)。

アラビア語およびヘブライ語は、双方向スクリプトを使用する主要な 2 つの言語グループです。アラビア語スクリプト・グループには、アラビア語、ペルシア語、ウルドゥー語などの言語が含まれます。ヘブライ語スクリプト・グループには、ヘブライ語だけでなく、イディッシュ語やラディノ語も含まれます。いずれの言語にもアルファベット (27 文字のみ) があるため、単一バイト・エンコード・スキームに合わせることができます。

双方向言語の特性

双方向スクリプトと西洋諸言語のスクリプト (英語、フランス語、ドイツ語、ギリシャ語など) の間には、主に 2 つの特性の違いがあります。これらの 2 つの特性は*双方向性* と*整形* です。

双方向性

双方向性は、次の 7 つの主要な概念から構成されます。

- セグメンテーション
- ネスト
- グローバル方向
- 双方向テキストの論理順序と物理順序
- テキスト・タイプおよび関連した再配列方式
- 対称スワッピング
- 翻訳後の GUI のウィジェット・ミラーリング

注: 下記に示す例では、いずれも大文字 (DCBA など) を使用してアラビア語またはヘブライ語の文字を表しています。

セグメンテーション
セグメンテーションは、方向が異なるストリング内にテキストの一部を持つストリングとして定義します。したがって、スクリプトは、右から左の方向で構成される主要部分と、左から右の方向で構成されるその他の部分を持つことができます。双方向セグメンテーションの例として、番地 Entrance B 25 Maple Street があげられます。この住所は、ヘブライ語では B ECNARTNE 25 TEERTS ELPAM と書きます。この例では、ストリング・テキストの主要部分 B ECNARTNE および TEERTS ELPAM の方向は右から左ですが、番号 25 の方向は左から右です。

ネスト ある方向を持つテキスト・セグメントの中に、方向が反対の追加のセグメントも存在する場合に、このテキスト・セグメントをネストとして定義します。ここでも番地 B ECNARTNE 25 TEERTS ELPAM を使用します。この住所のネスト・レベルは 1 です。番地 TEERTS ELPAM は右から左の方向に書かれていますが、その後で方向が反転し、番地 25 (方向は左から右) が

正しく配置できるようになっています。番地の後、方向が再度反転し、B ECNARTNE では右から左になっています。

グローバル方向

グローバル方向は書き順、読み順、段落方向とも呼ばれ、テキストの書き込みを開始する画面、ウィンドウ、またはページの側を指定します。グローバル方向は*文脈にも依存*します。つまり、テキストの意味は文脈によって変わります。双方向スクリプトとして書かれた文を使用して文脈に依存する例を示します。文は次のようになっています。

```
FRED DOES NOT BELIEVE taht yas syawla i.
```

この文では、左から右に読んだときの意味 (Fred does not believe I always say that) と、右から左に読んだときの意味 (I always say that Fred does not believe) が異なります。グローバル方向は文脈から常に明らかとは限らないため、アプリケーション開発者は、テキストがどのように読まれるか (左から右か右から左か) を意識する必要があります。

物理および論理テキスト順序

双方向テキストは、論理順序または物理順序のいずれかで格納できます。ワークステーション環境では、双方向テキストの入力および処理に適した方法は論理順序です。論理順序によれば、テキストをローマ字テキストと同様に処理できます。格納に論理順序を使用する場合は、グローバル方向とは方向が逆のセグメントを反転する手段を提供する必要があります。例えば、グローバル方位が英語 (左から右) である場合は、アラビア語およびヘブライ語のセグメントでは、本来の右から左の方向で表示されるように特別な処理が必要になります。逆に、双方向テキストの入力および処理のためにメインフレームで双方向テキストを格納するために適した方法は物理順序です。したがって、メインフレームおよびワークステーション環境から双方向テキストを統合する場合は、すべてのテキストのテキスト順序が同じレイアウトになるようにテキストを変換する必要があります。

双方向スクリプトでテキスト順序を使用する場合は、物理順序と論理順序も重要です。物理順序はテキスト・セグメントの物理的な表現方法を指し、論理順序はテキスト・スクリプト・セグメントの入力方法 (声に出して読む場合の発音方法) を指します。状況によっては、一部のセグメントを論理順序または物理順序のいずれかで再配列する必要もあります。例として次の文を示します。

```
my wife's name is ILIN
```

全体として、この文の方向は左から右です。読み手は、最初の文字が m で次に y が続く (以下同様) テキストを読みます。物理順序では、文字 i および s の後には ILIN を含むセグメントの文字 I が続きますが、ヘブライ語では ILIN を NILI と発音するため、論理順序では名前セグメントの最初の文字は I ではなく N になります。

テキスト・タイプ

テキスト・タイプは、特定のテキストを記録するために最適なアプローチです。つまり、テキスト・タイプごとに異なる記録手法が必要です。記録に使用するテキスト・タイプには、ビジュアル、暗黙 (論理)、および明示の 3 種類があります。

ビジュアル・テキスト・タイプは最も古い記録形式であり、単なる画面全体のコピーです。この形式のテキスト・タイプでは、プログラマーが埋め込みセグメントの位置とその処理について知っている必要があります。ほとんどのレガシー・アプリケーションおよびそのファイルでは、このタイプのテキストを使用します。

暗黙のテキスト・タイプによる記録では、ローマ字アルファベットの文字には固有の左から右への方向があり、アラビア語、ウルドゥー語、およびヘブライ語のアルファベットには固有の右から左への方向があると想定します。双方向性に適合させるために、暗黙テキスト処理のアルゴリズムを使用して固有の方向特性に基づいてセグメントを認識し、自動的にセグメントを反転できるようにします。暗黙テキスト・タイプの主な制限は、部品番号のように数字と文字 (左から右と右から左) が混在するストリングを処理できないことです。

明示テキスト・タイプによる記録では、テキスト・ストリングに追加の制御文字が埋め込まれていることを前提とします。これにより、明示アルゴリズムに対してセグメント反転、整形、数表示の選択などの変換を指示します。明示テキスト・タイプの制限は、埋め込み制御を処理するための自動処理が必要になることです。暗黙テキスト・タイプと明示テキスト・タイプの橋渡しをする特定の技法があります。この技法は、*基本表示アルゴリズム* として Unicode 標準 BIDI アルゴリズムで定義されています。

対称スワッピング

対称スワッピングは、<、(、[、{ など、反対方向の意味を持つ対称的な文字 >、)、]、} がある文字を処理できる機能です。グローバルにスワッピングすると、例えば `A > B` が `B > A` に変換されてしまうため、これらの文字では問題が発生しやすくなります。対称スワッピングにより、この記号を `B < A` に文字変換できます。

ウィジェット・ミラーリング

翻訳後の GUI のウィジェット・ミラーリングでは、GUI を言語の方向に合わせてミラーリングします。例えば、ウィジェット・ミラーリングでは、メニュー・ボタンおよびナビゲーション・ツリーを左ではなく、右に移動できます。それ以外の場合は、フレームおよびウィンドウはミラーリングされません。 112 ページの図 44 に、ドロップダウン・メニューのウィジェット・

ミラーリングを示します。

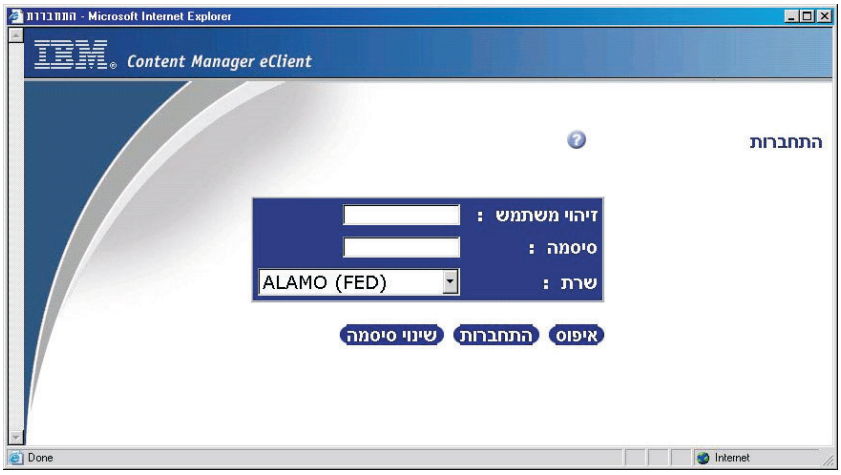

図 44. ウィジェット・ミラーリングされたウィンドウに表示された双方向ラベル

整形

整形は、多くの複雑な言語、特にアラビア語やヘブライ語などの**筆写体**の言語に見られる特性です。相互につながった単語中で隣接する文字がある場合、書記システムは筆写体となります。筆写体は、印刷よりも手書きに適しています。例えば、アラビア語では、一部の文字のみを右側の文字につなぐことができます。さらに、単語中の位置および隣接する文字との連結特性に応じて文字の形が異なる場合があります。これらの点から、整形は双方向テキストを分かりやすくするために重要です。整形処理では、文字の抽象表現を適切な形状で置換することにより、文字を適切な表示形式にレンダリングします。これを行うには、文字の基本型を使用し、形状を指定せずに特定の筆写体文字を選択できるようにします。

次に、形状決定ルーチンで文字の適切な形状を選択します。このルーチンでは、ソフトウェアかユーザーのいずれかによって指示された文脈に応じて適切な形状を自動的に (アルゴリズムによって) 選択できます。ほとんどの場合、筆写体言語のテキストの基本形状が格納されます。整形を構成する特性は、ほかにも 2 つあります。これらは文字の合成と固有の数字です。

文字の合成は、格納されているテキスト文字の番号と表示されるテキスト文字の番号の対応関係として定義されます。対応関係を管理するために、**合字**や**発音符**などの方法を使用します。合字は、複数の文字を、1 つの表示セルを占有する単一の文字で表現できる場合に使用します。双方向スクリプトでの発音符は、子音字の特定の方向 (上、中、下、近くのいずれか) に置いて母音を表すマークです。これらのマークを格納するときには、物理的な位置を占有しますが、表示に使用する場合には関連する子音字と同じセルを占有します。アラビア語では、現在スペーシング発音符が別個の文字として実装されており、発音符が属する文字の後に置くことになっています。

固有の数字も他の言語と使用方法が異なるため、特別な処理が必要です。例えば、ヘブライ語では、数字はアラビア数字 (1、2、3、...0) を使用して表記します。しかし、アラビア語、ペルシア語、ウルドゥー語などの筆写体言語では、固有の記号で数字を表記します。筆写体言語で使用する数字のラベルは、ヒンディ数字かアラ

ブ・インド数字のいずれかです。アラビア数字、ヒンディ数字、アラブ・インド数字のいずれであっても、単純な数値は左から右に表記されますが、数式の表記は言語ごとに異なります。例えば、アラビア語では数式を左から右に書きますが、ペルシア語では右から左に書きます。このため、通常、数字はアラビア数字でエンコードし、ユーザーまたは開発者の意図に応じて各国語固有の記号またはアラビア数字で表示します。

レイアウト変換および属性

双方向スクリプトを使用可能にするには、次のような特別な注意が必要です。

- 『テキスト・レイアウト』
- 『レイアウト属性』
- 114 ページの『レイアウト変換』

テキスト・レイアウト

双方向テキストではレイアウトが異なる場合があります。レイアウトは、テキストで使用するセグメントに応じて異なります。また、アラビア語スクリプトの場合は、使用する文字の整形および数字の形状も異なります。異なるレイアウト間で変換を行うには、変換機能 (レイアウト機能やレイアウト・サービス機能とも呼びます) が必要です。

レイアウト属性

双方向テキスト・レイアウトの特性を定義するには、一連の属性が必要です。双方向属性は、テキストの実際のレイアウトおよび変換方法を確定するために必要です。これらの属性は、通常テキストの外部に存在し、外部リソース・ファイルに格納されます。双方向属性には次の 5 つの属性があります。

- 方向
- テキスト・タイプ
- テキスト整形
- 対称スワッピング
- 数表示形状

方向 方向は、双方向テキストの書き込みを開始する表示領域 (ウィンドウ、フレーム、ページ) の境界を指定します。この指定は、テキストの先頭文字の方向に基づきます。方向は、右から左、左から右、またはコンテキスト依存にすることができます。)

テキスト・タイプ

テキスト・タイプは、テキスト・レイアウトの変換時に使用するアルゴリズムの種類を指定します。使用できるアルゴリズムには、ビジュアル、暗黙、明示の 3 種類があります。ビジュアル・アルゴリズムでは、埋め込まれている既存の方向セグメントに無関係に、テキストの行全体を出現するとおりにコピーします。暗黙アルゴリズムでは、文字の自然な方向 (例えばアラビア語では右から左、英語では左から右) に基づいて方向セグメントを認識し、それに応じてセグメントを反転します。明示アルゴリズムでは、方向セグメントを認識し、テキストに埋め込まれたビジュアル、明示、および方向制御に基づいて反転します。

テキスト整形

アラビア語スクリプトでは、単語中の位置や前後の文字の接続特性に応じて文字の形状が変わるため、テキスト整形が重要になります。

対称整形

対称整形は、いつ (、>、[、{ などの特定の文字を)、<、]、} と交換し、表示するテキストの論理的な意味を保存する必要があるかを指定します。

数表示形状

数表示形状は、アラビア語スクリプトに埋め込まれた数字を表示するときに、ヨーロッパ数字の形状を使用するかアラブ・インド数字の形状を使用するかを指定します。

双方向レイアウト属性に指定可能な値として一般的な組み合わせというものはありません。既存のアプリケーションは、これらの値のさまざまな組み合わせでデータを処理します。したがって、双方向データ・ストリームをアプリケーションに渡す場合は、アプリケーションが関連したテキスト属性を認識できることが重要です。

レイアウト変換

双方向テキストは、さまざまな環境 (プラットフォーム) やさまざまなレイアウトで格納および処理されます。レイアウト間の変換を作成するには、レイアウト変換機能が必要です。WebSphere Business Integration 製品は、Unicode BiDi アルゴリズムに基づくレイアウト変換機能を使用します。このアルゴリズムについては下記を参照してください。

`http://www.unicode.org/reports/tr9/)`

また、下記の IBM Java SDK 1.4.1 で実装されています。

`http://www-106.ibm.com/developmentworks/java/jdk/bidirectional/JAVABIDI.htm`

WebSphere Business Integration 製品での双方向スクリプトの使用可能化

WebSphere Business Integration 製品での双方向スクリプトの使用可能化は、さまざまなレベルおよびさまざまなコンポーネント構成で行われます。双方向使用可能化は、3 つのレベルで実現されています。

- WebSphere Toolset (Connector Configurator、BO Designer、System Manager など) を使用して作成した双方向スクリプト文字の表示、タイプ入力、格納、および検索
- コード・ページ変換の使用による (Unicode コード・セットと単一バイトコード・セットの間での) 双方向文字の形式の変換
- 双方向テキスト変換を使用した Windows 双方向形式 (WebSphere Business Integration 環境で使用する形式) と外部アプリケーションで使用する各種の双方向形式の間の変換。

双方向スクリプト・データを処理できる WebSphere Business Integration 製品は、Toolset、Adapter Framework、ICS broker、9 種のアダプター (JText、JDBC、Email、XML、WebSphere MQ、SAP、PeopleSoft、Web services、Lotus Domino)、および該当する ODA (JDBC、XML、WSDL) です。

上記のコンポーネントのうち、第 1 レベルに属するものは WebSphere Business Integration Toolset です。これらは、System manager などの Java ベースのツール、Business Object Designer や Connector Configurator などの C++ ベースのツール、Dashboard などの Web ベースのツールです。第 2 レベルに属するコンポーネントは双方向対応アダプターです。第 3 レベルに属するコンポーネントは、双方向対応アダプターおよび Adapter Framework です。

コネクター

コネクターは、アプリケーションと 1 つ以上のコラボレーションを媒介します。アプリケーションからのコンテンツの双方向データを処理し、Connector Configurator などの WebSphere Tools によって定義されるメタおよび構成データを使用して、ネイティブの双方向形式を使用するアプリケーションと 1 つ以上のコラボレーションの間の通信を確立します。

Adapter Framework

Adapter Framework は、アプリケーションを 1 つ以上のコラボレーションにリンクする構造を提供し、アプリケーションとコラボレーションの間でのコンテンツ・データの転送を処理します。Adapter Framework は双方向言語に対応しており、コンテンツ・データの双方向形式の整合性を確保します。

Toolset

WebSphere Business Integration Toolset は、InterChange Server とともに使用できる管理および開発ツールを提供します。Toolset のツールは、双方向スクリプト・データの処理に暗黙的に対応しており、双方向ロケール以外の追加の構成を必要としません (詳しくは、「*システム・インストール・ガイド (Windows 版)*」を参照してください。

次のセクションでは、次の各コンポーネントを使用可能にするための概要について説明します。

- 『コネクターでの双方向スクリプトの使用可能化』
- 116 ページの『Adapter Framework での双方向スクリプトの使用可能化』
- 116 ページの『コラボレーションでの双方向スクリプトの使用可能化』
- 117 ページの『マップでの双方向スクリプトの使用可能化』

コネクターでの双方向スクリプトの使用可能化

コネクターは、1 つのアプリケーション (または Web サーバーなどのプログラマチック・エンティティー) と 1 つ以上のコラボレーションを媒介します。コネクターは、SAP R/3 バージョン 4 などのアプリケーションや、データ・フォーマットまたはプロトコル (XML や EDI) などのテクノロジーに固有の場合もあります。コネクターは、次の 2 つの形式でコラボレーションと通信します。

- アプリケーション・**イベント通知**。特定のアプリケーションからコネクターに渡されます
- **要求処理**。コネクターがコラボレーションに代わり実行します。

(詳しくは、 69 ページの『第 5 章 コネクター』を参照してください。)

コネクターは、Connector Configurator などの WebSphere Tool によって定義されるメタおよび構成データを使用してデータ・ソース (アプリケーションまたはプログラミング・エンティティー) からのコンテンツ・データを処理します。WebSphere 製品環境では、メタおよび構成データは標準の Windows 双方向形式で表現および

格納されます。外部アプリケーションは、同じメタおよび構成データを、WebSphere Business Integration 製品で使用する Windows 双方向形式と異なる双方向形式で保持できるため、WebSphere Business Integration 環境と外部アプリケーションの間で正しく通信するためには変換が必要です。双方向言語に対応したコネクターをコネクターの BiDi.Metadata 標準プロパティーで構成すると、外部アプリケーションに固有のメタおよび構成データの双方向形式が適用されます。

Adapter Framework での双方向スクリプトの使用可能化

Adapter Framework は、InterChange Server および WebSphere Business Integration Adapters にコネクターをリンクし、アプリケーションまたはプログラミング・エンティティーと 1 つ以上のコラボレーションの間でコンテンツを流せるようにします。コネクターがメタおよび構成データを処理するのに対し、Adapter Framework はビジネス・オブジェクト属性の値などの実際のデータ内容を処理します。これは、Adapter Runtime コンポーネントが全コネクターに共通であるためです。

Adapter Framework の役割は、WebSphere Business Integration 環境での表現 (Windows の双方向形式でデータを表現します) と、外部アプリケーションでの表現 (別の双方向形式を使用する場合があります) の間でデータの整合性を確保することです。したがって、WebSphere Business Integration 環境から外部アプリケーションにデータが流れる場合は、Adapter Framework が必要に応じて Windows の双方向形式から外部の双方向形式への変換を実行します。逆に、外部の双方向形式が WebSphere Business Integration 環境で使用する Windows の双方向形式と異なる場合は、Adapter Framework が Windows の双方向形式への必要な変換を実行します。

Adapter Framework は、WebSphere Business Integration ブローカー以外のブローカーを使用する場合にもデータの整合性を確保できます。ブローカーが Windows と異なる双方向形式を使用する場合は、デフォルトの双方向形式をそのブローカーが使用する形式に変更することもできます。

コラボレーションでの双方向スクリプトの使用可能化

WebSphere 製品の実装では、**コラボレーション**という用語は、コードと複数のアプリケーション間の対話を促進するビジネス・プロセス・ロジックを含むソフトウェア・モジュールを指します。単純なコラボレーションはいくつかのステップのみから構成され、複雑なコラボレーションはいくつかのステップと他のコラボレーションから構成されます。

コラボレーションは、複数のアプリケーションにわたって分散させることができ、同期および非同期のサービス呼び出しの処理を行います。また、コラボレーションは、長期存続ビジネス・プロセスをサポート可能です。

WebSphere 製品の双方向スクリプト・サポートはコラボレーションも対象としています。コラボレーションは、WebSphere 環境 (コネクターやアクセス・インターフェース、他のコラボレーション) または Web サービスなどの外部ソースのいずれかから送信されたデータを受信します。双方向データ・フォーマットは、使用可能ないずれかのコネクターによって暗黙的に適用されるか、双方向言語サポートまたは API によって明示的に適用されます。双方向言語に対応していないコネクターや Web サービスなどの双方向言語サポートを適用しないコンポーネントからデータが

送信されている場合は、形式が整合せず、コラボレーションのビジネス・ロジックが失敗する可能性や、正しい結果を生成できない可能性があります。このようなエラーは、次のようにして回避できます。

- 双方向言語に対応したソースから入力を受け入れるか、WebSphere Business Integration 製品と同じ双方向形式で入力を受け入れる。
- 双方向言語に対応したコネクターで双方向言語サポートを呼び出し、コラボレーションに渡すデータに双方向形式を適用する。
- CwBidiEngine クラスの BiDi API を使用して、WebSphere 製品ドメインで使用するデータおよび外部データ・ソースから WebSphere 製品ドメインに導入するデータに双方向形式を適用する (「*コラボレーション開発ガイド*」の CwBidiEngine に関する章を参照してください)。

マップでの双方向スクリプトの使用可能化

WebSphere Business Integration 製品は、双方向スクリプトを持つマップをサポートします。マップは、コネクターまたは外部ソースのいずれかからデータを受信します。したがって、双方向言語に対応したコネクターから WebSphere Business Integration 製品環境に送信されるデータは、一様な双方向言語形式 (標準の Windows の双方向形式) であることが保証されます。しかし、Web サービス経由でエクスポートされるデータなど、不明な外部ソースからマップにデータが導入される場合もあります。Web サービスが Windows の双方向形式でない双方向データを操作する場合は、2 とおりの結果が考えられます。最初の結果として、サービスなどの接続が失敗する可能性があります。2 番目の結果として、双方向データが Windows の双方向形式と異なる形式であると、そのデータが Windows の双方向形式のデータと比較されるため、データ処理で予測不能な結果となってしまいます (「*マップ開発ガイド*」の Activity Editor での双方向機能の使用のについてのセクションを参照してください)。これらのエラーは、116 ページの『コラボレーションでの双方向スクリプトの使用可能化』で説明しているステップと同じステップを使用して回避できます。

双方向テキストの処理

双方向テキストの処理時に特別な注意が必要になるときがあります。特別な注意が必要となる場合の 1 つとして、双方向言語に対応していない旧バージョンの WebSphere Business Integration からリポジトリー・データをマイグレーションする場合があげられます (詳しくは、『データのマイグレーション』を参照してください)。特別な注意を払わないと、2 つの双方向形式のデータが同じリポジトリーに混在してしまい、ビジネス・ロジックの正常な処理および機能に影響が及ぶ可能性があります。別のケースは、FTP URL や E メール・アドレスなどの例外パターンがある双方向テキストに関するものです (詳しくは、118 ページの『BiDi API』を参照してください)。

データのマイグレーション

旧バージョンの WebSphere Business Integration 製品からデータをマイグレーションするときには、予防措置をとる必要があります。

マイグレーション処理中に、旧バージョンから格納された双方向データを保持し、双方向言語に対応したコネクターを経由して導入された新しい双方向データととも

に使用できます。この場合、サーバー・レベルで操作対象の双方向データが Windows 形式であることは保証されません。結果として、コラボレーションやマップなどでの双方向データの処理が破壊され、元に戻せない場合があります。

現行バージョンの WebSphere 製品にマイグレーションする前に、現行バージョンの BiDi API を使用して、リポジトリーにある双方向データをすべて Windows の双方向形式に変換することをお勧めします。(詳しくは、「*マップ開発ガイド*」の CxBidiEngine に関する章を参照してください。)

特別な双方向ストリング

FTP URL および E メール・アドレスに双方向変換を明示的に適用すると、データが正しく解釈されなくなる可能性があります。正確に解釈されるようにするために、このようなストリングを解析してから変換を開始し、ストリング値の中で問題が発生する可能性があるサブコンポーネントを識別します。問題が発生する可能性があるサブコンポーネントが識別された場合は、ストリングを分割し、各サブコンポーネントに双方向変換を適用します。変換処理が完了したら、サブコンポーネントを単一のストリングに組み立てなおし、正確な変換値を表現します。この値は、後で使用できるように格納されます。この処理は、メタ・ビジネス・オブジェクトの処理に使用されます。

BiDi API

WebSphere Business Integration 製品にバンドルされている BiDi API クラスにより、各種のコンポーネントに双方向形式を適用できます。BiDi API クラス関数を Adapter Framework で使用すると、コンテンツ・データに双方向形式を適用できます。また、コネクターで使用すると、メタおよび構成データに双方向形式を適用できます。コラボレーションおよびマップで使用すると、外部ソースからインポートするデータに双方向形式を適用できます。

BiDi メソッド

BiDi API クラスには、次の 3 つのメソッドがあります。

- BiDiBusObjTransformation
- BiDiBOTransformation
- BiDiStringTransformation

BiDiBusObjTransformation 関数

BiDiBusObjTransformation メソッドは、BusinessObject タイプのビジネス・オブジェクトの双方向形式を変換します。このメソッドはコラボレーションの場合に有用です。(詳しくは、「*コラボレーション開発ガイド*」の CxBiDiEngine に関する章を参照してください。)

BiDiBOTransformation 関数

BiDiBOTransformation 関数は BusinessObject インスタンスに適用します。この関数は Adapter Framework の場合に有用です。(詳しくは、「*コラボレーション開発ガイド*」または「*マップ開発ガイド*」の CxBiDiEngine に関する章を参照してください。)

BiDiStringTransformation 関数

BiDiStringTransformation 関数は String オブジェクトに適用します。この関数は、WebSphere 製品環境の内部と外部のいずれのオブジェクトにも使用できます。(詳しくは、「*コラボレーション開発ガイド*」または「*マップ開発ガイド*」の CxBiDiEngine に関する章を参照してください。)

設計上の制限

現行の設計では、双方向言語サポートに対するソリューションが限定されています。制限の一部を次に示します。

- 双方向メタデータのサポートは、すべてのメタデータ双方向プロパティーに適用可能な 1 つのパラメーターによってのみ提供されます。つまり、属性が双方向パラメーターに基づく場合は、すべてのメタデータ属性を変換するか、メタデータ属性をまったく変換しないかのいずれかです。したがって、メタデータ双方向プロパティーごとに異なる双方向形式を設定したり、この双方向プロパティーの影響範囲外とするメタデータ・パラメーターを指定したりすることはできません。
- WebSphere Business Integration 製品のすべてのコンポーネントが双方向言語に対応しているわけではありません。例えば、Server Access Interface や各種のコネクターは双方向言語に対応していません。このようなコンポーネントを双方向言語に対応したコンポーネントと併用すると、InterChange Server または WebSphere Business Integration Adapters に置かれるデータに複数の双方向形式が混在してしまう場合があります。結果としてこの場合は、InterChange Server または WebSphere Business Integration Adapters の双方向データの表現が整合しなくなり、それらに基づく処理が正しく行われなくなる可能性があります。しかし、InterChange Server または WebSphere Business Integration Adapters のいずれかのデータに一様な双方向形式を適用するための機構は提供されていません。この形式を適用する責任はユーザーにあります。双方向言語に対応していないコンポーネントから受信するデータに双方向形式を適用するために、BiDi API が提供されています。
- 双方向コンテンツのサポートはコネクター・レベルで提供されます。結果として、ユーザーがコネクターごとに使用できる双方向形式の指定は 1 つのみに制限されます。つまり、特定のコネクターでサポートされるビジネス・オブジェクトは、すべて 1 つの双方向形式で双方向データを保持する必要があります。
- WebSphere Business Integration 製品は、適切な手動インストール構成を行った場合に双方向言語に対応します (詳しくは、「*システム・インストール・ガイド (Windows 版)*」を参照してください)。
- WebSphere Business Integration 製品で双方向言語に対応するアダプターは、JText、JDBC、Email、XML、WebSphere MQ、SAP、PeopleSoft、Web services、および Lotus Domino の 9 つのみです。
- WebSphere Business Integration 製品の内部双方向形式には Windows の双方向形式と同じ形式を選択しているため、WebSphere 製品のツール (Connector Configurator、Map Designer、BO Designer など) は Windows プラットフォームでのみサポートされます。ツールのサポートを他のプラットフォームに拡張する場合は、現行の設計では不十分であり、将来変更する必要があります。
- InterChange Server および WebSphere Business Integration Adapters の内部形式として Windows の双方向形式を選択しているため、異なる双方向形式を使用する

アプリケーションとの通信が制限されます。例えば、2 つのアプリケーションがビジュアル双方向形式を使用して WebSphere Business Integration 製品経由で通信するとき、設計上、内部論理双方向形式を持ちます。これは、遷移的でない双方向変換の制限によるものです。

- WebSphere は、ビジネス・オブジェクトおよびビジネス・オブジェクト属性名の一部として双方向文字をサポートしません。このため、属性、要素、または列名に双方向文字がある XML/XSD ファイルやデータベース・テーブルに基づいてビジネス・オブジェクト・テンプレートを生成した後には、ローマ字以外の文字を含むビジネス・オブジェクトおよびビジネス・オブジェクト属性名をすべて名前変更し、ローマ字のみが含まれるようにする必要があります。
- 現在、XML および区切り付きデータ・ハンドラーのみがサポートされています。要求/応答データ・ハンドラーは、XML および区切り付きデータ・ハンドラーのいずれかを使用する場合、または要求/応答処理に別個に XML および区切り付きデータ・ハンドラーを使用する場合にのみサポートされます。EDI、固定幅、および名前値データ・ハンドラーはサポートされません。
- WebSphere は、基本エンティティー名 (コネクター、コラボレーション、マップ、ビジネス・オブジェクト、ビジネス・オブジェクト属性など) の一部として双方向文字などのローマ字以外の文字をサポートしません。

要約

この章では、双方向テキストおよび WebSphere Business Integration システムで双方向テキストを使用可能にする方法について説明しました。覚えておくべき重要な点は次のとおりです。

- 双方向スクリプトは、左から右に書くテキストを持ち、右から左に書くセグメントが埋め込まれています。
- 双方向テキストには、双方向性と整形の 2 つの主要な特性があります。双方向アプリケーションの開発時には、それぞれの特性に対処する必要があります。
- 双方向テキスト・スクリプトを利用するには、ロケールをセットアップする必要があります。
- InterChange Server は、適切なシステム構成を行った場合に双方向言語に対応します (詳しくは、「システム・インストール・ガイド *(Windows 版)*」を参照してください)。WebSphere Business Integration Adapters には、双方向言語に対応したアダプターが 9 つ含まれています。
- WebSphere Business Integration の内部双方向形式は Windows の双方向形式 (論理、左から右) です。

特記事項

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

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

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

以下の保証は、国または地域の法律に沿わない場合は、適用されません。

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

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

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

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

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

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

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

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

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

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

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

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

プログラミング・インターフェース情報

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

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

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

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

商標

以下は、IBM Corporation の商標です。

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

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

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

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

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。

System Manager および Adapter Monitor 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]

[タ行]

[ナ行]

[ハ行]

[マ行]

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