IBM WebSphere Business Integration Adapters

A Technical Introduction to Adapters

Add Custom Cover Content Here

IBM WebSphere Business Integration Adapters

A Technical Introduction to Adapters

Note!
Before using this information and the product it supports, read the information in "Notices," on page 19.

25June2004

This edition of this document applies to *WBIA Adapter Framework*, version *2.4*.
To send us your comments about IBM WebSphere Business Integration documentation, e-mail doc-comments@us.ibm.com. We look forward to hearing from you.
When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

About this document

The IBM(R) WebSphere(R) Business Integration adapter portfolio and framework is a suite of software integration products that supply connectivity for leading e-business technologies and enterprise applications. It includes:

- Prebuilt components for common business integration processes
- Tools and templates for customizing and creating components
- A flexible, easy-to-use platform for configuring and managing the components

This document provides an introduction to the concepts and architecture underlying IBM WebSphere Business Integration adapters and describes their structure and operation.

Audience

This document is for consultants, developers, and system administrators who use the adapter at customer sites.

Related documents

The complete set of documentation available with this product describes the features and components common to all WebSphere Business Integration Adapters installations, and includes reference material on specific components.

You can install related documentation from the following sites:

- For general adapter information; for using adapters with WebSphere message brokers (WebSphere MQ Integrator, WebSphere MQ Integrator Broker, WebSphere Business Integration Message Broker); and for using adapters with WebSphere Application Server, see the IBM WebSphere Business Integration Adapters InfoCenter:
  http://www.ibm.com/websphere/integration/wbiadapters/infocenter
- For using adapters with WebSphere InterChange Server, see the IBM WebSphere InterChange Server InfoCenters:
  http://www.ibm.com/websphere/integration/wicserver/infocenter
  http://www.ibm.com/websphere/integration/wbicollaborations/infocenter
- For more information about WebSphere message brokers:
  http://www.ibm.com/software/integration/mqfamily/library/manualsa/
- For more information about WebSphere Application Server:
  http://www.ibm.com/software/webservers/appserv/library.html

These sites contain simple directions for downloading, installing, and viewing the documentation.

Typographic conventions

This document uses the following conventions:

`courier font`	Indicates a literal value, such as a command name, file name, information that you type, or information that the system prints on the screen.

italic	Indicates a new term the first time that it appears, a variable name, or a cross-reference.
blue text	Blue text, which is visible only when you view the manual online, indicates a cross-reference hyperlink. Click any blue text to jump to the object of the reference.
{ }	In a syntax line, curly braces surround a set of options from which you must choose one and only one.
|	In a syntax line, a pipe separates a set of options from which you must choose one and only one.
[ ]	In a syntax line, square brackets surround an optional parameter.
...	In a syntax line, ellipses indicate a repetition of the previous parameter. For example, `option[,...]` means that you can enter multiple, comma-separated options.
< >	Angle brackets surround individual elements of a name to distinguish them from each other, as in `<server_name><connector_name>tmp.log`.
/, \	In this document, backslashes (\) are used as the convention for directory paths. For UNIX installations, substitute slashes (/) for backslashes.
%text% and *$text*	Text within percent (%) signs indicates the value of the Windows `text` system variable or user variable. The equivalent notation in a UNIX environment is *$text*, indicating the value of the *text* UNIX environment variable.

Chapter 1. Overview

As today's businesses transform themselves into on demand e-businesses, responsiveness and flexibility are increasingly high-priority issues. Companies must be able to quickly adapt and respond to a changing marketplace, and the consequences of slow response can be severe. In an effort to support, accelerate and optimize business processes and to meet new business requirements, companies acquire and develop new applications. The integration of these applications within various, disparate systems has itself become a serious challenge.

One aspect of today's business integration requirements is the need to leverage existing enterprise information system (EIS) assets in a heterogeneous, integrated enterprise. Whether the business need is to share business data across disparate Enterprise Information System (EIS) assets, automate business processes across EISs, employees, and business partners, connect new e-business applications to existing EISs, or provide users with access to EIS business data, there is a common requirement to provide interfaces to EISs.

IBM WebSphere Business Integration adapters provide a large suite of ready-to-use adapters for interfacing to various EISs. These adapters are based on a common framework that allows you to adapt, configure and operate the adapters in a consistent manner, while using them in different deployment configurations within the WebSphere software platform to meet varying business integration needs.

IBM WebSphere Business Integration adapters offer a basic communication interface into or out of the application for which they were developed. They also act as a first abstraction layer for the integration server through the representation of business transactions in the form of business objects.

WebSphere Business Integration adapters can be used in simple or complex business integration solutions, providing access to EIS business data for application connectivity, process integration, e-business applications, and business portals.

Structure of adapters

IBM WebSphere Business Integration Adapters represent a collection of software programs, tools and application programming interfaces (APIs) an enterprise can use to enable EISs to exchange business data with an integration server. This can include IBM WebSphere Business Integration Server, IBM WebSphere InterChange Server, IBM WebSphere MQ Integrator Broker, IBM WebSphere MQ Integrator and IBM WebSphere Application Server Enterprise. Each EIS requires its own application-specific adapter to participate in the business integration system.

WebSphere Business Integration Adapters include:

- A connector that links the application to the integration server.
- Tools with graphical user interfaces to help the user configure the connector and create the business object definitions needed for the EIS.
- An object discovery agent (ODA) for some of the adapters, which introspects EIS metadata to generate business objects.

- An Adapter Development Kit (ADK), which provides a framework for developing custom adapters in Java or C++ technology if an adapter for a particular legacy or specialized EIS is unavailable

Connectors

A connector mediates interactions between an EIS and the integration server over the network. It can be specific to an application — such as SAP R/3, Version 4 — or to a data format or protocol, such as XML over HTTP. All connectors share certain common behaviors, differing only in the manner in which they interact with applications and with business objects.

Each connector consists of two parts — the adapter framework and the application-specific component (shown in Figure 1 on page 4).

- The adapter framework communicates with the integration server by means of the transport layer.
- The application-specific component interacts directly with an EIS.

Chapter 2. Adapter architecture

This chapter describes the components of an adapter and how they work together to process business objects.

The WebSphere Business Integration adapters offer a basic communication interface into or out of the EIS for which they were developed. They expose business transactions to the integration server in a simple and consistent manner, and they rely upon the integration server to provide transformation, routing, and process logic.

This lightweight architecture simplifies maintenance and upgrades because the artifacts do not require code changes in the adapter, and the complexity of artifacts within the adapter is minimized. Furthermore, this architecture delivers a very small footprint, providing the flexibility to situate the adapter near or even on the target EIS for optimum performance.

The chapter contains the following sections:

Overview

In general, WebSphere Business Integration adapters can be categorized as either application adapters or technology adapters.

- **Application adapters** are designed to interface with a specific application programmable interface (API) for a specific version of an EIS. These include adapters for mySAP.com, Siebel, PeopleSoft, and Oracle Applications.
- **Technology adapters** are designed to support a standard technology interface to any EIS that supports the same interface. These include adapters for JDBC (database), JText (file system), and Web Services.

It is important to note that these are functional distinctions, not architectural differences; all the adapters are still based on the WebSphere Business Integration Adapter Framework.

However, one minor architectural difference that does exist across adapters arises from the use of **data handlers**.

Some adapters use business object and attribute metadata directly to assemble API calls to an EIS. For example, a business object may represent an SAP BAPI, a Siebel

Business Component, a PeopleSoft Component Interface, or a database SQL statement. In these cases, business object data handlers provide complete business object processing.

Understanding connectors

The connector component of an adapter mediates interactions between an EIS and the integration server over the network. It can be specific to an application, such as SAP R/3, version 4; or specific to a data format or protocol, such as XML over HTTP. All connectors share certain common behaviors, differing only in the manner in which they interact with applications and with business objects.

Each connector consists of two parts:

- The **connector framework**, which communicates with the integration server by means of the transport layer.
- The **application-specific component**, which interacts directly with an application.

The subcomponents of a connector are shown in Figure 1.

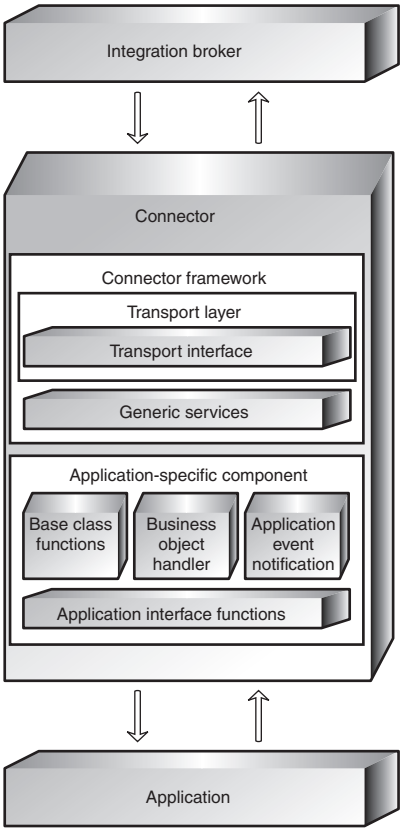

Figure 1. Subcomponents of a connector

The WebSphere Business Integration adapters are built upon a common adapter framework, which provides many features that are consistent across the adapters. They include:

- An adapter runtime infrastructure, based in Java
- Business objects, which constitute a metadata model for business transactions

- Communication with integration servers through the transport layer
- Standard interaction patterns for the exchange of business objects
- Quality of service for transactions, including assured delivery, fault tolerance and recovery, and failed transaction handling
- Vertical and horizontal scalability
- Configuration and deployment through common tooling
- Administration, monitoring, trace and log facilities

Each connector can be deployed in a distributed environment, separate from the integration server. The adapter framework communicates with the integration server over a **messaging transport**. The adapter can even be deployed remotely from the integration server across the Internet and through firewalls using WebSphere MQ over SSL or WebSphere MQ Internet Pass-Through.

Note: The only exception to this distributed deployment exists when the adapter is deployed as a J2EE Connector Architecture resource adapter within WebSphere Application Server.

The connector behaves as a client to the EIS to which it interfaces, and as such it can reside on any machine on the network from which it can connect to the EIS.

This distributed deployment capability provides maximum flexibility to the user. For example, you may deploy all the components of the integration infrastructure on one set of servers, or you may maximize performance by locating the adapter close to the target EIS server, and using WebSphere MQ across a wide area network (WAN) to communicate with the integration server.

Understanding the data flow

In the business integration system, data flow—the movement and processing of data sent from one application or entity to another— can occur either as an asynchronous or a synchronous exchange between applications over the network.

For instance, an application might need to exchange data with another application to obtain data or to communicate changes in its data store. WebSphere Business Integration adapters facilitate this flow by exchanging data between the integration server and adapter in the form of **business objects** (described in detail below).

Business objects are an abstraction of application data structures and their associated operations, and they encapsulate and transmit business data for several purposes. For instance, they may convey new or changed data, a request for data, or data returned in response to a request or operation. Furthermore, either the integration server or the adapter may originate business objects.

A business object can report the occurrence of an EIS event, an operation that affected a data entity in an EIS. The EIS event might be the creation, deletion, or change in value of that collection of data. For example, an adapter might poll an application for new employee entities on behalf of the integration server. If the EIS creates a new employee entity, the adapter sends an event business object to the integration server.

In an EIS-initiated request, the EIS invokes the adapter synchronously through a call-back mechanism, and the adapter sends a business object to the integration server to represent the operation. The adapter takes the corresponding response

business object from the integration server, and provides a synchronous response to the EIS. For example, an application may issue a real-time pricing look-up to the integration infrastructure.

In addition to such interactions for the exchange of business data, the adapter and integration server may also exchange administrative messages, such as status changes and administrative operations.

Understanding business objects

New instances of business objects are created based on templates called **business object definitions**. Business object definitions are the templates from which the adapter or integration server creates a particular instance of a business object. This definition specifies the structure and organization of the business object's attributes, values, and metadata.

A business object definition is identified by its name. The name indicates the business object definition type, such as Customer, Order, or Invoice.

Because application-specific information and other metadata in the business object definition guide the actions of the application-specific component, such behavior can be described as **metadata-driven**.

An application-specific component that is metadata-driven is flexible because it has no hard-coded instructions for each type of business object that it supports. Without recoding or recompiling, the application-specific component automatically supports new business object definitions, as long as the corresponding application data can be accurately described by the adapter's metadata syntax.

Business object structure

A business object reflects a data entity, a collection of data treated as a unit. For example, a data entity can be equivalent to an employee record, containing all the basic information about the employee, such as the name, address, telephone number, employee number, position code, and so forth. A business object may also have application-specific information (metadata) that helps the application-specific component process it. All business objects also contain attributes and verbs.

Each business object has a **type name** that identifies it within the business integration system. For example, the type might be Customer, Employee, Item, or Contract.

Verbs indicate actions on the data in the business object. While a business object definition contains a list of verbs, a business object contains only one verb. The most common verbs associated with business object definitions are Create, Retrieve, Update, and Delete. The meaning of a verb differs according to the role of the business object; it can describe an application event, make a call, make a request, or identify the result of a previous request.

Attributes in a business object definition describe the values connected with the entity, such as Last Name, Employee ID, Case Number, Amount, or Date Initiated. At runtime, attributes are filled in with actual data. For example, an Employee business object definition might contain attributes for the employee's name, address, employee ID, and other relevant information. The attributes of a business object are analogous to the fields of a form or columns in a database table.

The following figure is an example of a simple business object, showing its type, verb, and attribute values.

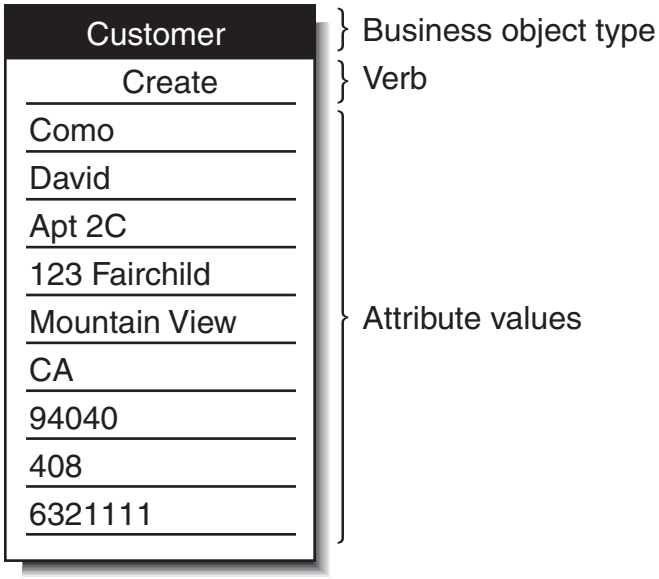

Figure 2. Business object components

An attribute can also refer to a child business object or to an array of child business objects, such as an array of line items in a contract or part references in an invoice. A business object that contains child business objects or arrays of child business objects is a hierarchical business object. A business object whose attributes contain only data is a flat business object.

Within the adapter environment, it is important to draw a distinction between business object definitions and instances of the business objects themselves. To summarize:

- A business object definition specifies the types, structure, and order of information for each entity within a WebSphere Business Integration Adapter, the verbs that it supports, and the metadata associated with object, attributes, and verbs. The local repository for the adapter stores business object definitions.
- A business object is an instance of the definition, containing actual data. Business objects are created at runtime and are not stored in the repository.

Figure 5 illustrates the relationship between a business object definition and a business object.

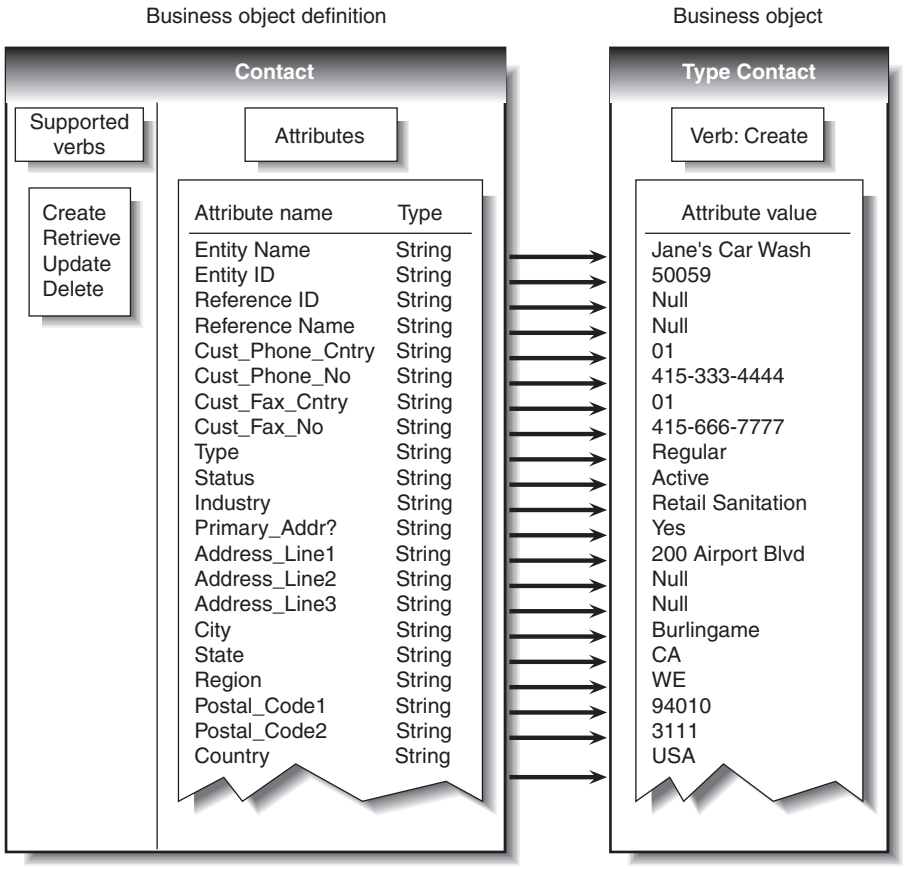

Figure 3. Business object definition and business object

Creating business object definitions

Business object definitions are developed with a graphical tool called the Business Object Designer. The Business Object Designer allows a developer to directly edit the structure and properties of a business object's elements, including its attributes, verbs, associated metadata, and child business objects.

To enable rapid development of new business transactions for an adapter, the Business Object Designer leverages wizard-driven Object Discovery Agents (ODAs), which are available for many of the adapters. Each ODA is designed for its corresponding adapter and EIS. The ODA introspects the target EIS to expose available functions. Then it generates business objects with the structure and metadata syntax required by the adapter to execute the selected functions. Once generated, the business objects can be extended within the Business Object Designer.

Processing business objects

Now that we've reviewed the components of the adapter and the basic interactions with an integration server, we can examine more closely how the adapter processes business objects for specific operations.

Request processing

Request processing is initiated by the integration server. The integration server sends requests, in the form of business objects, to the adapter framework. A request can ask the destination application to do either of the following:

- Retrieve business data and return it to the integration server
- Update the application's data store

For example, the integration server might send an adapter a request message to delete a contract, update a part, create an order, or retrieve a customer.

When the adapter framework receives an integration server's request, it converts the message into a suitable business object and forwards it to the application-specific component.

For example, if the integration server sends a request to delete a contract, the application-specific component receives the request in the form of a "Contract.Delete" business object. The application-specific component translates the business object into an application request – typically a set of calls to the API – and then returns the results, if needed.

When an application-specific component receives a request, it determines how to process the request based on three types of information:

- The business object verb
- Metadata that is contained in the business object definition and used in the construction and deconstruction of the business object
- Application-specific information for the verb

An adapter's application-specific component responds to the Create, Retrieve, Update, or Delete verb in a request according to the logic and API of its application. The application-specific components of different adapters might handle the same type of request differently, although the result is logically the same.

For some adapters, only one method is required for performing operations on a business object, regardless of what verb the request contains. But for many adapters, each verb requires a different method.

When an application-specific component receives a request, it invokes the method in the EIS that matches the business object's active verb. For example, when an application-specific component receives an "AppAEmployee.Update" business object, it invokes the Update method on the "AppAEmployee" object. The Update method interacts with the EIS in order to perform the update.

When the operation is complete, the application-specific component will optionally populate the business object request with data from the EIS and the status of the operation, and forward it to the adapter framework for return to the integration server. This may be the data retrieved from the EIS in a retrieve operation, or it may be data generated by the application, such as object keys or default values, in a create or update operation.

Event notification

The process of conveying changed EIS data to the integration server is called event notification. Most of today's EISs either do not provide an event notification mechanism for external systems (such as an adapter) or have only rudimentary support for it. To enable automated integration, event notification is an absolute necessity. If the EIS does not provide event notification, the enterprise must either purchase an adapter from a business integration vendor that provides this feature, or it must spend the time, money and development resources (with deep application knowledge) necessary to develop this event notification mechanism for the EIS.

To an adapter, an EIS event is any operation that affects the data of an EIS entity associated with a business object definition. There are other types of events in an EIS – a mouse click is an event to an EIS's window system or forms interface. The adapter, however, is interested only in a pre-defined subset of the data-level events that create, update, delete, or otherwise affect the content of the EIS's data store. Some EISs can explicitly trap and report events, providing user-friendly event management. Other EISs, without a concept of discrete, reportable events, might silently update their databases when something happens.

Where an EIS provides a facility for events, the adapter's event detection mechanism generally leverages it to populate the event store. For such applications, setting up the connector's event-notification mechanism is a normal application setup task. For example, an EIS may allow the installation of a script that executes when a particular type of event occurs, and that the script can place a notification in an event store. This mechanism is frequently referred to as a user exit. Another EIS might have an internal workflow system that can register an event in the EIS and write to an event store when executed.

Essentially all IBM WebSphere Business Integration Adapters provide the event notification for the EIS for which the adapter is developed. This event notification mechanism is typically developed with the tools the EIS vendor provides and is certified by the EIS vendor for IBM WebSphere Business Integration Adapters. Many competitive adapter products do not provide this event detection mechanism for the EIS, and the enterprise must invest its own resources to solve this problem.

The ways in which application-specific components detect and retrieve events differ from one adapter to another. However, the way in which application-specific components send events to the adapter framework, and the way in which the adapter framework delivers those events to the integration broker, is standard across all adapters. The following describes general concepts regarding the event notification mechanism of most adapters, but does not describe the specific implementation of any particular adapter.

The event notification mechanism generally consists of the following elements:

- An event store located within the EIS
- A means to populate the event store with EIS events
- An event detection mechanism for the adapter to identify new EIS events in the event store
- A mechanism to retrieve new events from the EIS
- A mechanism to deliver events to the integration server

The following figure shows an adapter and its supporting infrastructure detecting a change to the EIS data store and constructing a business object to convey the changed data to the integration server.

The event store provides a persistent mechanism, such as additional tables in the EIS data store, for logging the EIS's data changes. The data store provides an ordered list of operations that take place in the EIS. It might have the physical form of an application event queue, a database table, a directory on the file system, or an e-mail inbox.

The information in the event store generally includes the business object type, verb, the key identifying the changed data entity in the EIS data store, timestamp, and priority. The event store may also contain the complete data for the event itself. The event store is usually provided with the adapter as an artifact specific to the EIS development environment, and is readily imported into the system.

If an EIS does not provide native support for events, the event notification mechanism can utilize the EIS database. A table would be created for the adapter's event store, and database triggers would be installed on the EIS tables of interest to insert event records into the event table.

For example, you can set up a trigger on an "Employee" table that detects updates to the rows. When an update occurs, the trigger inserts information about the update into an event table. Each new row that appears in the event table represents an event notification.

An adapter's application-specific component identifies new EIS events through its event detection mechanism, the most common of which is polling for new events in the event store. The adapter framework generally initiates a poll call at periodic intervals. The polling method is specific to the application, based on the event notification mechanism that the connector uses.

Polling behavior is configurable, including poll frequency and the maximum number of events processed per poll call. The poll call asks the application-specific component to check for changes to the EIS's event store. The interface to the event store may be through the EIS APIs where possible, or it may be provided through database queries to retrieve new events directly from an event table.

If there has been a change since the last poll call, the application-specific component determines if a business object definition exists to represent the changed data. After detecting an event, the adapter's application-specific component:

- Associates the application event with a business object definition and creates an instance of that business object.
- Sets the verb and key value attributes in the business object.
- Retrieves application data and populates the business object's attributes, generally by invoking the same method used for integration server-initiated requests.
- Forwards the business object to the adapter framework.
- Archives the event (optional).

Once the adapter framework receives the business object from the application-specific component, it delivers the event to the integration server.

Why use this process?

There are two arguments you might hear in objection to this technology for event notification:

1. The event notification and event persistence are intrusive to the application.
2. Why use polling when an application provides event notification?

This section explains why IBM architected its adapters this way and how this architecture benefits WebSphere Business Integration adapter customers.

Intrusiveness

If an EIS does provide an event notification mechanism, the WebSphere Business Integration Adapters typically leverage this mechanism. One example is the IBM WebSphere Business Integration adapter for mySAP.com, which can use the SAP internal workflow event notification mechanism. The adapter provides the link into this SAP internal event notification mechanism.

The information that is provided by the event notification mechanism (including the ID of the data that caused the event in the application, timestamp of the event, and the priority of the event,) is usually stored in the event table within the EIS.

The event table is created specifically for the adapter in the EIS. By storing the event in the additional event table, no events are lost – even when the adapter is down or the network connection between the adapter and the integration server is lost. If this persistence mechanism did not exist, application events could be lost and the result would be data inconsistencies across the enterprise.

Polling

The WebSphere Business Integration Adapter Framework provides the technology to bind specific methods within the adapter to the event detection mechanism of an application. This enables a “publish and subscribe” mechanism for application events within the adapter.

Most EISs do not provide robust error handling of their event notification mechanism, and sometimes the adapter will not leverage this direct link into the event notification mechanism of the application.

For example, suppose four external applications, including an adapter, subscribe to a specific EIS event, such as the creation or change of customer data. The event within the EIS occurs and gets published to all four subscribing applications, but the adapter is unavailable or an error occurs during the event submission to the adapter. Ideally, the event notification mechanism of the EIS would be intelligent enough to re-submit the events to any external application that has subscribed to it.

Furthermore, the EIS would take care of the error handling and persistence of the event until all subscribing applications have successfully processed it. This sophisticated event notification typically doesn’t exist in today’s EISs, since they were not built with integration in mind. Additionally, EISs that do provide this capability often require extensive configuration and development within the EIS accommodate the deployment. WebSphere Business Integration adapters provide this valuable capability in a simple and reliable way.

Request/Response processing

One common business integration scenario is an EIS initiated synchronous request. In this case, the EIS issues a request to retrieve or update data outside the bounds of the EIS, and expects a synchronous response.

Examples would be a web application that must check product availability before placing an order, or a call center service application that must access scheduling information before booking a service appointment. In both cases, the EIS user makes an inquiry, which requires real-time access to backend business data or processes, so that the user can make use of the result before proceeding.

We have already discussed how an adapter can accommodate synchronous requests from an integration server to fulfill such a situation. We have also discussed how an adapter can provide event detection within an EIS to asynchronously deliver events to the integration server. The difference here is that we need to facilitate synchronous requests from an EIS to the integration server.

This type of interaction is usually handled through a standards-based interface, such as XML over HTTP, web services, request/reply messaging, CORBA, or those provided for J2EE. Or the integration server may provide direct interfaces for standards-based synchronous requests, bypassing an adapter altogether.

As a consequence, while the WebSphere Business Integration system as a whole provides support for EIS-initiated synchronous request processing across many standard interfaces, only a few of the adapters need to provide this support. These are mainly adapters that support open standards for synchronous processing, and those designed for specific EISs that conform to their proprietary interfaces.

In general, an adapter that provides EIS initiated synchronous request handling may have listeners to receive requests, or it may register itself as a server upon startup to be invoked by clients. When the adapter receives a request, it translates it into a business object, and forwards it to the adapter framework, which submits a synchronous request to the integration server.

When the integration server returns the response business object, the adapter translates it back to the native interface, and provides a synchronous response to the originating client. It is important to note that while not many adapters need to provide this capability, the adapter framework does provide support for this type of interaction between an EIS and the integration server.

Using data handlers

In cases where serial data is used with an EIS interface, adapters frequently make use of data handlers to transform between the serial data and business object. Examples of data handlers include those for EDI, XML, delimited, fixed-width, SWIFT, and HL7 data formats.

An adapter may use multiple data handlers; for example, an adapter for JText may be configured to read and write some files containing EDI records, and others containing fixed-width records. And the same data handler can be used with multiple adapters. The delimited data handler may be used to transform data in an adapter for JText or the adapter for Email. New data handlers can easily be developed from the Data Handler Framework for use with adapters. Data handlers thus form a component of adapters that provide flexibility and extensibility to a deployment.

Where data handlers are used, business object handlers implement the technology interface, and in turn call data handlers to transform between business objects and serial data. Because the data handler interprets the business object metadata to perform the transformation, the metadata syntax for the business object is specific to the data handler, not the adapter.

Advanced capabilities

Several advanced capabilities follow from the metadata driven nature of the adapters. They include the following.

- The adapter may handle multiple operations (Create, Retrieve, Update, Delete) with the same business object.
- The adapter may handle multiple business objects because the instructions that identify each object in the target system are stored in the object metadata.

Also, we saw earlier that the adapter framework can handle synchronous and asynchronous operations in either direction. Business objects, in turn, may be used in either direction. To create a business object, you simply use graphical tools and wizards to introspect application metadata.

Taken together, these features provide a sophisticated yet simple means of accommodating the exchange of business data with an EIS using only one adapter, and without coding.

Chapter 3. Deploying adapters

This chapter describes how WebSphere Business Integration adapters can be deployed in various configurations to support business integration solutions that involve different integration servers.

In every case, the components used by the adapter runtime are developed in a consistent manner through the WebSphere Business Integration System Manager perspective within the Eclipse Workbench, though their deployment to the integration server may vary.

Also, the application-specific component of the adapter is unchanged in the different deployments. The adapter framework is simply configured to accommodate communications with the appropriate integration server.

This chapter contains the following sections:

WebSphere InterChange Server

The processes that execute within WebSphere InterChange Server to choreograph automated interactions across distributed EISs are called **collaborations**. Collaborations interact with each adapter through a connector controller, which is a runtime service that mediates communication between collaborations and adapters.

While all connector controllers are identical except for their configuration, WebSphere InterChange Server instantiates a separate connector controller for each adapter. The connector controller exchanges business objects with the adapter to support all the interaction patterns described previously. The connector controller also exchanges administrative messages with the adapter, such as the adapter status, and commands to pause, start, or shutdown.

The communication transport between WebSphere InterChange Server and the adapter may be provided through Java Messaging Service (JMS) queues, IIOP, or a combination of the two. When used in combination, JMS is used for asynchronous event notification, and IIOP is used for WebSphere InterChange Server-initiated request processing and administrative messages.

Business object definitions are represented as XML schemas, but because both WebSphere InterChange Server and the adapters use business objects, they are simply serialized in communication. The adapter is configured through the WebSphere Business Integration System Manager perspective within the Eclipse Workbench, which deploys the adapter artifacts together with other integration artifacts to WebSphere InterChange Server.

The adapter may be configured to receive all of its configuration information from WebSphere InterChange Server upon start-up, or from a local repository of XML schemas.

Figure 4shows how the adapter is deployed with WebSphere InterChange Server.

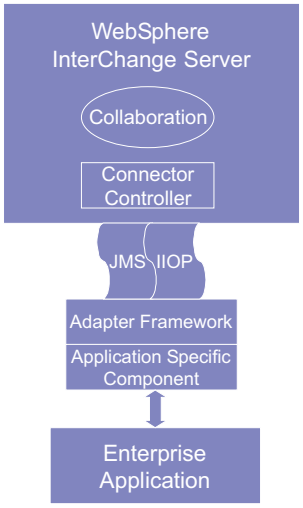

Figure 4. Adapter deployed with WebSphere InterChange Server

The WebSphere InterChange Server provides administration and monitoring for the adapters by means of the web-based System Monitor, the System manager perspective in the WebSphere Studio Workbench, and the WebSphere InterChange Server SNMP Agent.

WebSphere Business Integration Message Broker

Message transformation rules and routing logic are contained in components called **message flow**s that execute within the WebSphere Business Integration message brokers. These include WebSphere MQ Integrator, WebSphere MQ Integrator Broker, and WebSphere Business Integration Message Broker.

Message flows are invoked by messages that the message listener receives, and these flows put new messages onto queues. Message flows interact with adapters by exchanging XML messages over WebSphere MQ queues.

The adapter framework interacts with the queues through the JMS interface, and translates between business objects in memory and the XML message representation of those objects. The business object definitions associated with an adapter are represented as XML schemas and are readily imported into the WebSphere message broker Message Repository Manager (MRM) for use in message flows.

The adapter is configured through the WebSphere Business Integration System Manager perspective within the Eclipse Workbench. The System Manager also provides basic monitoring and administration capabilities. The adapter receives all its configuration data from a local repository of XML schemas upon start-up.

Figure 5 on page 17shows how the adapter is deployed with WebSphere Business Integration Message Broker.

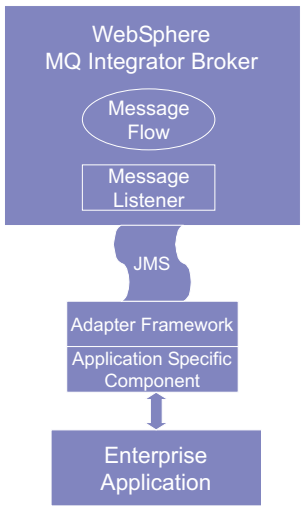

Figure 5. Adapter deployed with WebSphere Business Integration Message Broker

WebSphere Application Server

WebSphere Application Server Enterprise Edition provides the infrastructure for e-business applications. The e-business application interacts with an adapter by exchanging XML messages over JMS queues. The communications are encapsulated in Session Enterprise JavaBeans (EJBs) and Message-Driven Beans (MDBs) to accommodate the various interactions.

The actual interface used to make the communication calls and handle message formatting is Web Services Invocation Framework (WSIF). At runtime, WSIF uses Web Services Description Language (WSDL) definitions obtained from the adapter. These WSDL definitions describe the adapter interface and configuration (that is, the services available).

The adapter is configured through the WebSphere Business Integration System Manager perspective in WebSphere Studio Application Developer Integration Edition, which is based on the WebSphere Studio Workbench. The adapter definition and its associated business object definitions are represented as WSDL and XML schemas, and exported as a Service Project to the Business Integration perspective, where the runtime EJBs and MDBs are generated and deployed to WebSphere Application Server Enterprise.

The WebSphere Administrative Console enables monitoring and administration of the adapters. The WebSphere Business Integration Adapter Monitor perspective also provides basic monitoring and administration capabilities.

The adapter framework interacts with the message queues through the JMS interface, and translates between business objects in memory and the XML message representation of those objects. The adapter receives all of its configuration information from a local repository of XML schemas upon start-up.

Figure 6 on page 18shows how the adapter is deployed with WebSphere Application Server.

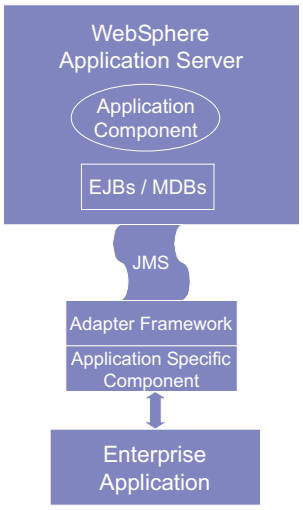

Figure 6. Adapter deployed with WebSphere Application Server

J2EE Connector Architecture

In this deployment, the adapter runs in-process with the J2EE application, and application components interact directly with the J2C resource adapter through the Common Client Interface (CCI). The adapter's business objects may be represented as custom records within the CCI. The adapters will continue to be configured and deployed through WebSphere Studio Application Developer Integration Edition, enabling rapid deployment of business integration solutions.

A J2C implementation of the adapter framework may have some architectural advantages over the JMS implementation, but it will not replace it. A form of the JMS implementation will remain to provide enhanced quality of service in a distributed environment deployed over a WAN or across the Internet.

Appendix. Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM RTP Laboratory
3039 Cornwallis Road
P.O. BOX 12195

Raleigh, NC 27709-2195
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both:

IBM
the IBM logo
AIX
CrossWorlds
DB2
DB2 Universal Database
MQIntegrator
MQSeries
Tivoli
WebSphere

Lotus, Domino, Lotus Notes, and Notes Mail are trademarks of the Lotus Development Corporation in the United States, other countries, or both.Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

MMX, Pentium, and ProShare are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product or service names may be trademarks or service marks of others.

IBM WebSphere Business Integration Adapter Framework V2.4.

Printed in USA