BEA WebLogic Integration™

Implementing ebXML for B2B Integration

Release 7.0
Document Date: June 2002

Implementing ebXML for B2B Integration

Part Number	Date	Software Version
N/A	June 2002	7.0

Contents

About This Document

This document describes how to implement the ebXML business protocol in a BEA WebLogic Integration^{TM} application.

It is organized as follows:

- Chapter 1, "Introduction," provides information about the support for the ebXML business protocol provided by WebLogic Integration, and it describes the ebXML architecture in WebLogic Integration applications.
- Chapter 2, "Administering ebXML," describes how you can configure your WebLogic Integration system to support the exchange of ebXML business messages by trading partners.
- Chapter 3, "Using Workflows with ebXML," describes how you can design and manage ebXML business processes using workflows.

What You Need to Know

This document is intended primarily for:

- Business process designers who use the WebLogic Integration Studio to design workflows that conform to the ebXML standards for the exchange of business messages.
- System administrators who set up and administer WebLogic Integration ebXML solutions.

Before reading this document, we recommend that you become familiar with the following documents:

- *Introducing BEA WebLogic Integration*
- *Learning to Use BEA WebLogic Integration*
- *Starting, Stopping, and Customizing BEA WebLogic Integration*
- *Introducing B2B Integration*
- *Administering B2B Integration*

Before you begin designing your own ebXML solutions, we recommend that you become familiar with the following documents:

- *Designing BEA WebLogic Integration Solutions*
- *Deploying BEA WebLogic Integration Solutions*

e-docs Web Site

BEA product documentation is available on the BEA corporate Web site. From the BEA Home page, click on Product Documentation or go directly to the "e-docs" Product Documentation page at `http://e-docs.bea.com`.

How to Print the Document

You can print a copy of this document from a Web browser, one file at a time, by using the File—>Print option on your Web browser.

A PDF version of this document is available from the BEA WebLogic Integration documentation Home page, which is available on the documentation CD and on the e-docs Web site at `http://e-docs.bea.com`. You can open the PDF in Adobe

Acrobat Reader and print the entire document, or a portion of it, in book format. To access the PDFs, open the BEA WebLogic Integration documentation Home page, click the PDF Files button, and select the document you want to print.

If you do not have the Adobe Acrobat Reader installed, you can download it for free from the Adobe Web site at `http://www.adobe.com/`.

Contact Us!

Your feedback on the BEA WebLogic Integration documentation is important to us. Send us e-mail at **docsupport@bea.com** if you have questions or comments. Your comments will be reviewed directly by the BEA professionals who create and update the WebLogic Integration documentation.

In your e-mail message, please indicate which version of the product documentation you are using.

If you have any questions about this version of BEA WebLogic Integration, or if you have problems installing and running BEA WebLogic Integration, contact BEA Customer Support through BEA WebSUPPORT at **www.bea.com**. You can also contact Customer Support by using the contact information provided on the Customer Support Card, which is included in the product package.

When contacting Customer Support, be prepared to provide the following information:

- Your name, e-mail address, phone number, and fax number
- Your company name and company address
- Your machine type and authorization codes
- The name and version of the product you are using
- A description of the problem and the content of pertinent error messages

Documentation Conventions

The following documentation conventions are used throughout this document.

Convention	Item
Ctrl+Tab	Indicates that you must press two or more keys simultaneously.
italics	Indicates emphasis or book titles.

`monospace text`	Indicates code samples, commands and their options, data structures and their members, data types, directories, and file names and their extensions. Monospace text also indicates text that you must enter from the keyboard.
	Examples:
	`#include <iostream.h> void main ( ) the pointer psz`
	`chmod u+w *`
	`\tux\data\ap`
	`.doc`
	`tux.doc`
	`BITMAP`
	`float`

`monospace italic text`	Identifies variables in code. *Example*: `String` *`expr`*
UPPERCASE TEXT	Indicates device names, environment variables, and logical operators. *Examples*: LPT1 SIGNON OR
`{ }`	Indicates a set of choices in a syntax line. The braces themselves should never be typed.

Convention	Item
`[ ]`	Indicates optional items in a syntax line. The brackets themselves should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
|	Separates mutually exclusive choices in a syntax line. The symbol itself should never be typed.
`...`	Indicates one of the following in a command line: ■ That an argument can be repeated several times in a command line ■ That the statement omits additional optional arguments ■ That you can enter additional parameters, values, or other information The ellipsis itself should never be typed. *Example*: `buildobjclient [-v] [-o name ] [-f file-list]... [-l file-list]...`
. . .	Indicates the omission of items from a code example or from a syntax line. The vertical ellipsis itself should never be typed.

CHAPTER

1 Introduction

WebLogic Integration supports a routing architecture that allows it to manage the exchange of XOCP, RosettaNet, cXML, and ebXML messages. Trading partners using WebLogic Integration can therefore exchange business messages using any of these business protocols.

WebLogic Integration supports the *ebXML Message Service Specification v1.0*, which defines the message enveloping and header document schema used to transfer ebXML messages with a communication protocol such as HTTP. In addition, WebLogic Integration supports the creation and execution of workflows that model ebXML business messages.

The following sections provide an overview of the ebXML business protocol and its implementation in WebLogic Integration:

- About ebXML
- WebLogic Integration Architecture and ebXML

About ebXML

The ebXML business protocol is sponsored by UN/CEFACT and OASIS. The ebXML Web site (`http://www.ebxml.org`) describes ebXML as "a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms and define and register business processes."

The ebXML Message Service Specification supported by WebLogic Integration is a set of layered extensions to the base Simple Object Access Protocol (SOAP) and SOAP Messages with Attachments specifications. The ebXML Message Service provides security and reliability features that are not provided in the specifications for SOAP and SOAP Messages with Attachments.

The *ebXML Message Service Specification v1.0*:

- Focuses on defining a communications protocol-neutral method for exchanging electronic business messages.
- Defines specific enveloping constructs that support reliable, secure delivery of business information.
- Defines a flexible enveloping technique that permits ebXML-compliant messages to contain payloads of any format. This capability allows legacy electronic business systems employing traditional syntaxes, such as UN/EDIFACT, ASC X12, and HL7, to leverage the advantages of the ebXML infrastructure along with users of emerging technologies.

Information about the ebXML business protocol, including *ebXML Message Service Specification v1.0*, can be found at the ebXML Web site (`http://www.ebxml.org`).

Information about SOAP, including the following documents, can be found at the World Wide Web Consortium (W3C) Web site (`http://www.w3c.org`):

- *Simple Object Access Protocol (SOAP) 1.1*
- *SOAP Messages with Attachments*

WebLogic Integration Architecture and ebXML

WebLogic Integration support for ebXML consists of the following components:

- ebXML protocol layer

  Business protocol definitions provide support for the ebXML protocol in WebLogic Integration.

- ebXML plug-in

  The ebXML plug-in extends the functionality of the WebLogic Integration Studio and the process engine by supporting the modeling and execution of workflows that, in turn, support ebXML messaging.

- Interoperability with WebLogic Integration – Business Connect

  A trading partner that deploys WebLogic Integration can use ebXML to interoperate with a trading partner that deploys WebLogic Integration – Business Connect. WebLogic Integration – Business Connect is a lightweight client that can be deployed in a few hours, enabling enterprises to enlist new business partners quickly by eliminating their barriers to entry.

  For information about WebLogic Integration – Business Connect, see *Using WebLogic Integration – Business Connect*.

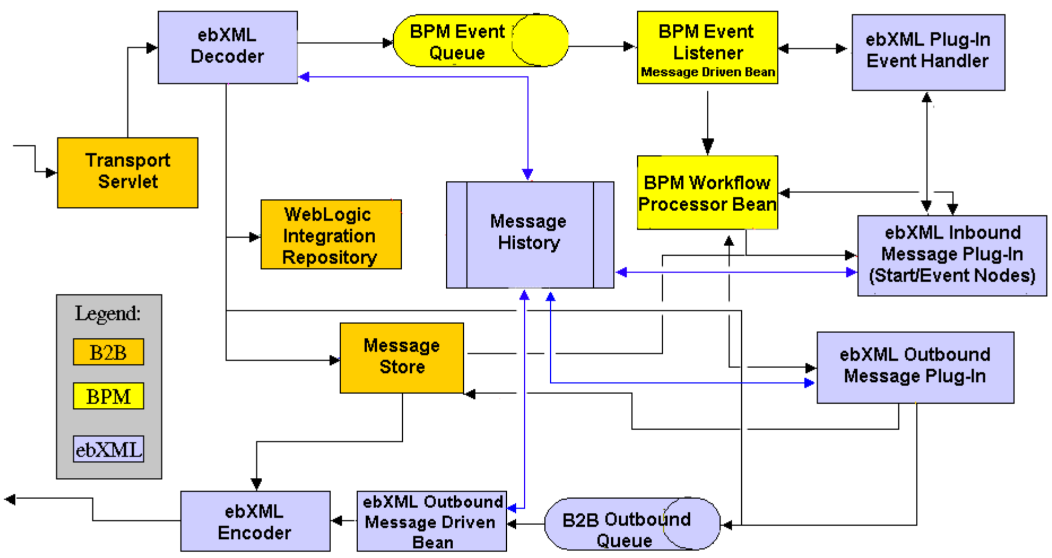
In WebLogic Integration, the ebXML plug-in, the workflows, the process engine, and the B2B engine work in concert to provide the following capabilities:

- Trading partner delivery channels are bound to the ebXML business protocol as part of the B2B configuration.
- Conversation definitions and collaboration agreements defined in the B2B configuration associate delivery channels with the appropriate workflows.
- Workflow actions send ebXML messages that are routed through the B2B engine to remote trading partner delivery channels or to WebLogic Integration – Business Connect.

- Workflow template properties indicate which role in a conversation is implemented by a workflow. The roles in ebXML conversations have predetermined names (*initiator* and *participant*).
- Workflow events wait for ebXML messages sent by other workflow actions.

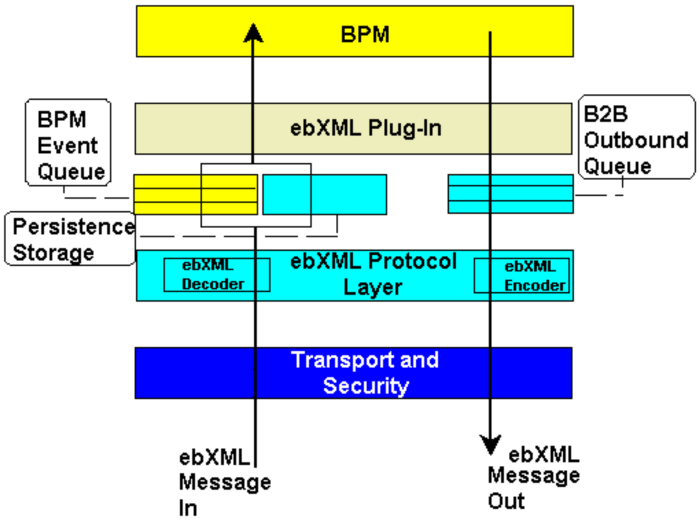
The following figure illustrates WebLogic Integration ebXML architectural components.

Figure 1-1 WebLogic Integration ebXML Architecture


ebXML Decoder
BPM Event Queue
BPM Event Listener
Message Driven Bean
ebXML Plug-In Event Handler
Transport Servlet
BPM Workflow Processor Bean
WebLogic Integration Repository
Message History
ebXML Inbound Message Plug-In (Start/Event Nodes)
Legend:
B2B
BPM
ebXML
Message Store
ebXML Outbound Message Plug-In
ebXML Encoder
ebXML Outbound Message Driven Bean
B2B Outbound Queue

The following figure illustrates how ebXML messages are processed by WebLogic Integration. (For information about how WebLogic Integration – Business Connect processes messages, see "Using ebXML" in *Using WebLogic Integration – Business Connect*.)

Figure 1-2 ebXML Message Processing in WebLogic Integration



The following sections describe the components illustrated in the preceding figures.

ebXML Protocol Layer

The ebXML protocol layer provides the ability to send and receive messages via the Internet according to the ebXML Message Service specifications for transport, message packaging, and security.

Logic plug-ins are Java classes that can intercept and process business messages at run time. (For information about logic plug-ins, see *Programming Logic Plug-Ins for B2B Integration*.) The ebXML business protocol is associated with two logic plug-ins: an ebXML decoder and an ebXML encoder.

When a WebLogic Integration trading partner receives an ebXML message, the transport servlet forwards the message to the ebXML decoder. The ebXML decoder processes the protocol-specific message headers, identifies the trading partner that sent the message, and forwards the ebXML message to a BPM event queue. When a WebLogic Integration trading partner sends an ebXML message, the ebXML encoder takes the message from the send-side B2B outbound event queue and forwards it to the transport service.

The *ebXML Message Service Specification v1.0* is independent of the communication protocol used. WebLogic Integration supports the HTTP(S) communication protocol.

Security

The SSL protocol with mutual authentication (HTTPS) can be used to support an ebXML conversation between trading partners. The SSL protocol provides secure connections by:

- Enabling each of two applications linked through a network connection to authenticate the other's identity
- Encrypting the data exchanged between applications

For information about configuring security for ebXML business transactions, see "Configuring Security" on page 2-10.

ebXML Plug-In to BPM

The WebLogic Integration Studio is a key tool that supports business process management (BPM) in the WebLogic Integration environment. It provides the functionality needed by integration specialists to design, run, and maintain complex e-business processes that integrate existing enterprise systems, cross-enterprise applications, and human decision makers.

As described in Chapter 3, "Using Workflows with ebXML," an ebXML plug-in to BPM provides the functionality needed to design and monitor workflows that can choreograph ebXML business processes and the associated message exchange.

WebLogic Integration Repository

The WebLogic Integration repository is the database that stores the information required by WebLogic Integration. For information about the WebLogic Integration repository, see "Working with the Repository" in *Administering B2B Integration*.

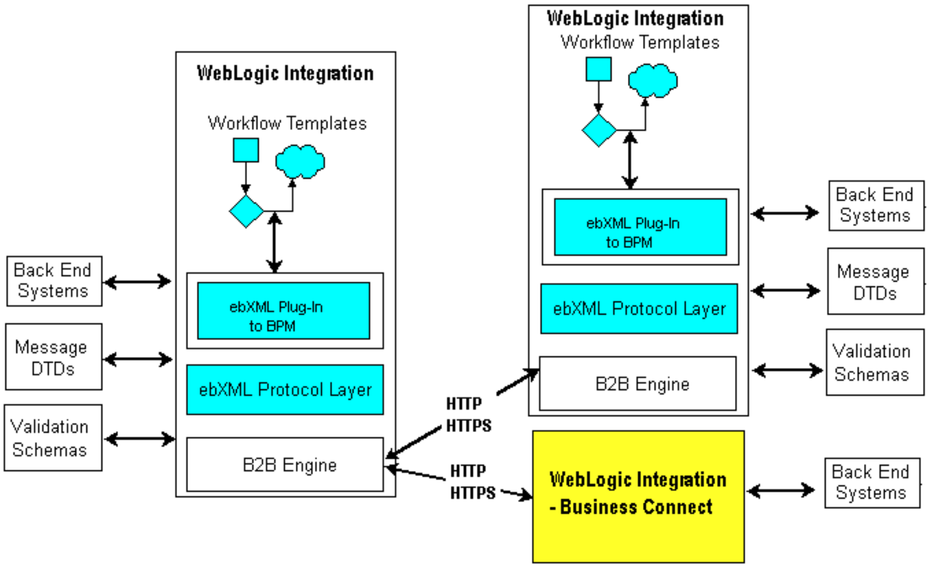
WebLogic Integration – Business Connect

The ebXML business protocol is supported by WebLogic Integration in two scenarios:

- Conversations between two trading partners that deploy WebLogic Integration
- Conversations between one trading partner that deploys WebLogic Integration and another that deploys WebLogic Integration – Business Connect

The following figure shows possible trading partner configurations for ebXML message exchange.

Figure 1-3 WebLogic Integration ebXML Trading Partner Configurations



WebLogic Integration – Business Connect is a lightweight client that implements the ebXML business protocol. It can be deployed in a few hours, enabling BEA customers to rapidly establish and develop B2B relationships with trading partners with a minimal investment in time, technology, and resources. Thus trading partners of all sizes can participate in e-commerce trading communities.

For information about installing and using WebLogic Integration – Business Connect, see *Using WebLogic Integration – Business Connect*.

2 Administering ebXML

In the WebLogic Integration environment, you use the B2B Console to configure your system to support the exchange of ebXML business messages between trading partners, and to design and manage ebXML business processes using workflows.

This section describes how the architecture of WebLogic Integration provides a robust framework supporting the ebXML business protocol. It also describes how to configure your system to support the exchange of ebXML business messages between trading partners. It includes the following topics:

- Introduction
- Configuring Your Environment for ebXML Messaging
- Reliable Messaging
- Configuring Security

For information about business processes and workflows, see Chapter 3, “Using Workflows with ebXML.”

Introduction

A trading partner for which WebLogic Integration is deployed can engage in ebXML-based e-business transactions with other trading partners deploying WebLogic Integration, or with trading partners for which the lightweight client, WebLogic Integration – Business Connect, is deployed. WebLogic Integration – Business Connect offers small or mid-size trading partners a low-cost opportunity to join trading networks quickly and participate in the rapidly growing e-commerce community.

When two WebLogic Integration trading partners exchange ebXML business messages, they must be configured in peer-to-peer mode. In a peer-to-peer configuration, applications for two trading partners communicate through their respective delivery channels. Peer-to-peer configurations are described in "Configuration Models" in" Getting Started with B2B Integration "in *Introducing B2B Integration*.

For information about the WebLogic Integration – Business Connect lightweight client, see *Using WebLogic Integration – Business Connect*.

Configuring Your Environment for ebXML Messaging

In the WebLogic Integration environment, the user-defined elements required to implement the exchange of ebXML messages between trading partners include the following:

- Conversation definitions
- Workflow templates
- Trading partners and delivery channels
- Collaboration agreements

All these elements are configured with the WebLogic Integration B2B Console and WebLogic Integration Studio. A trading partner that deploys WebLogic Integration stores this configuration data in its repository. Specifically, each enterprise populates its own WebLogic Integration repository with the trading partners, conversation definitions, collaboration agreements, and workflows necessary to engage in e-business interactions with other WebLogic Integration trading partners.

Note: A trading partner for which WebLogic Integration – Business Connect is deployed stores trading partner configurations for both itself and its trading partners. Other data, such as the conversation definition and collaboration agreement necessary to engage in an ebXML business transaction with WebLogic Integration, is configured by the WebLogic Integration trading partner through the B2B Console.

For a sample WebLogic Integration ebXML configuration, see "ebXML Applications" in "Configuration Requirements" in *Administering B2B Integration*. This sample configuration describes the ebXML options available for performing the following tasks:

- Configuring trading partners and their associated delivery channels
- Configuring conversation definitions to implement the required roles
- Associating the required trading partner delivery channels with the appropriate roles in the collaboration agreements
- Ensuring your WebLogic Integration system is correctly configured to support the exchange of ebXML messages between your system and a trading partner using the lightweight client, WebLogic Integration – Business Connect

Using the WebLogic Integration B2B Console

For information about how to use the B2B Console to configure the conversation definitions, trading partners, delivery channels, and collaboration agreements needed to support B2B integration functions, see "Configuring B2B Integration" in *Online Help for the WebLogic Integration B2B Console.*

To facilitate the exchange of data and business processes among trading partners, the import and export functions of the B2B Console allow you to export WebLogic Integration repository data to an XML file, and import data from an XML file to your WebLogic Integration repository.

Note: The import and export functions of the B2B Console can also be used to facilitate the exchange of configuration data between one trading partner that deploys WebLogic Integration and another that deploys the lightweight client. Options for facilitating the import and export of data to be shared by such trading partners are provided in the B2B Console.

Warning: If the Initialize Database option is selected, then when you subsequently import repository data, existing data is destroyed. Be careful about selecting the Initialize Database option.

For information about using the B2B Console to import and export repository data, see "Importing and Exporting B2B Integration Components" in *Administering B2B Integration*.

Exchanging Configuration Data Files with WebLogic Integration – Business Connect

Trading partners are business entities that are authorized to send and receive business messages in a conversation. Each WebLogic Integration trading partner stores trading partner data for itself and its trading partners in its WebLogic Integration repository.

XML configuration files containing configuration data required for B2B conversations can be imported and exported to and from the WebLogic Integration repository, as described in the previous section. Trading partner profile XML files can be imported and exported to and from a WebLogic Integration – Business Connect trading partner. This functionality supports the exchange of trading partner data between a trading partner that deploys WebLogic Integration and another that deploys WebLogic Integration – Business Connect.

When you import a trading partner profile created by a WebLogic Integration – Business Connect application into the WebLogic Integration repository, your repository is populated with trading partner configuration data. Other configuration data (such as conversation definitions and collaboration agreements) necessary to support e-business transactions between the trading partners must be entered through the B2B Console.

Warning: If the Initialize Database option is selected, then when you subsequently import repository data, existing data is destroyed. Be careful about selecting the Initialize Database option.

Trading partners defined by WebLogic Integration and those defined by WebLogic Integration – Business Connect contain different elements. A WebLogic Integration – Business Connect trading partner is defined by a smaller set of elements than a WebLogic Integration trading partner.

When a trading partner profile created by WebLogic Integration – Business Connect is imported into a WebLogic Integration repository, the elements of the imported trading partner are mapped to equivalent WebLogic Integration-specific elements.

However, for several elements required to define trading partners in the WebLogic Integration repository, there are no equivalent elements in a WebLogic Integration – Business Connect trading partner profile. When there are no values to map from the imported data to WebLogic Integration-specific elements, WebLogic Integration assigns default values to those elements.

The following table shows:

- How imported trading partner elements are mapped to WebLogic Integration trading partner elements (compare the first two columns)
- The values assigned to WebLogic Integration-specific elements for which there are no WebLogic Integration – Business Connect equivalents (compare columns two and three)

Table 2-1 Mapping Trading Partner Elements

WebLogic Integration – Business Connect Trading Partner	WebLogic Integration Trading Partner	Values Assigned to Business Connect Trading Partner Elements on Import to WebLogic Integration Repository
Identity:Name	Name	
Identity:Address +City+State+Zip +Country	Address (Fields are concatenated)	
	Type	Remote
Identity:Phone	Phone	
Identity:Fax	Fax	
Identity:Notify Email	Email	
Identity:ID	Party:PartyID	
Identity:ID	Party:BusinessID	
Identity:ID:Type	Party:BusinessID:BusinessIDType	
	Doc Exchange:Name	TP_DocumentExchange
	Doc Exchange: Business Protocol Binding	ebXML 1.0
	Doc Exchange: Business Protocol Definition	ebXML
	Doc Exchange: Delivery Semantics	OnceAndOnlyOnce
	Doc Exchange: Retries	3

Table 2-1 Mapping Trading Partner Elements (Continued)

WebLogic Integration – Business Connect Trading Partner	WebLogic Integration Trading Partner	Values Assigned to Business Connect Trading Partner Elements on Import to WebLogic Integration Repository
	Doc Exchange: Interval	60000
	Doc Exchange: Time To Live	0
	Transport:Name	TP_Transport
	Transport Protocol	HTTP-1.1 **Note:** If you exchange security certificates and use SSL in interactions with the WebLogic Integration – Business Connect trading partner, you must use the B2B Console to change this transport security protocol specification to HTTPS-1.1 after you import the trading partner data. (For details, see "Configuring Security" on page 2-10.)
Outbound Protocol:URL	URI Endpoint	
	Delivery Channel: Name	TP_DeliveryChannel
	Delivery Channel: Document Exchange Name	TP_DocumentExchange
	Delivery Channel: Transport Name	TP_Transport

For information about WebLogic Integration trading partner data, see "Configuring Trading Partners" in *Online Help for the WebLogic Integration B2B Console*.

For information about WebLogic Integration – Business Connect trading partner data, see " Company Profiles " in *Using WebLogic Integration – Business Connect*.

Using the Bulk Loader

We recommend that you use the B2B Console to export and import WebLogic Integration repository data. WebLogic Integration also provides a command-line tool, called the Bulk Loader, that you can use to import and export these configuration files.

For more information, see "Working with the Bulk Loader" in *Administering B2B Integration*.

Exchanging Messages with WebLogic Integration – Business Connect

How ebXML messages are processed in the WebLogic Integration environment is described in "WebLogic Integration Architecture and ebXML" on page 1-3.

WebLogic Integration – Business Connect supports ebXML business processes using Message Control Documents (MCDs) as the interface between its ebXML engine and a back-end system. The MCDs are XML documents that contain zero or more payloads and information that WebLogic Integration – Business Connect uses to process outbound and inbound ebXML documents.

Mapping ebXML Messaging Elements

The following table shows how ebXML messaging elements are mapped to WebLogic Integration elements, and, how these WebLogic Integration elements are mapped, in turn, to WebLogic Integration – Business Connect elements.

Table 2-2 Mapping ebXML Messaging Elements

ebXML Element	WebLogic Integration Element	WebLogic Integration – Business Connect MCD Element	WebLogic Integration Default Settings
PartyId	Sender Business ID	mcd:SenderId	
	Receiver Business ID	mcd:ReceiverId	

Table 2-2 Mapping ebXML Messaging Elements (Continued)

ebXML Element	WebLogic Integration Element	WebLogic Integration – Business Connect MCD Element	WebLogic Integration Default Settings
Service	Conversation Definition Name	mcd:Service	
ConversationId	Conversation Instance ID	mcd:CorrelationId	
CPAId			`http://www.bea.com/wli/cpa`
Action		mcd:Action	SendMessage/ Acknowledgment
MessageId	Message ID (Unique)	mcd:MessageId	
RefToMessageId	Message ID (Reference to Original Message)	mcd:ReftoMessageId	

You can use the information in this table to understand how business messages between WebLogic Integration and WebLogic Integration – Business Connect trading partners are processed and tracked in an ebXML exchange. (For an example scenario, see "Correlating ebXML Messages in Conversations" in the next section.)

For information about WebLogic Integration – Business Connect MCDs, see "Using ebXML" in *Using WebLogic Integration – Business Connect*.

Correlating ebXML Messages in Conversations

The `Conversation Instance ID` (and `mcd:CorrelationId`) relates documents that belong to the same conversation between partners in an ebXML exchange. In other words, in a given conversation, all ebXML messages must reference the `ConversationId` element of the first message in that conversation. For example, when a WebLogic Integration – Business Connect trading partner receives an ebXML message from a WebLogic Integration trading partner, the Business Connect trading partner must use the `Correlation ID` from the incoming message when responding in the same conversation.

When the Business Connect trading partner uses an MCD to process the inbound ebXML message, the successfully processed document is placed in a directory for inbound XML documents. The WebLogic Integration – Business Connect XML application must extract the Correlation ID from the inbound XML message, and use it in the messages it returns in this conversation.

The default directory in which inbound XML documents are placed is:

`WLI-BC\data\company_profile_ID\xmlin`

In the preceding pathname:

- `WLI-BC` represents the directory in which you installed WebLogic Integration – Business Connect.
- `company_profile_ID` represents the directory specified in your Business Connect installation for the WebLogic Integration trading partner from which the message originated.

(For more details, see " Company Profiles " in *Using WebLogic Integration – Business Connect*.)

Reliable Messaging

In your WebLogic Integration environment, a reliable messaging service handles the delivery and acknowledgment of ebXML messages that require reliable delivery. The parameters that control reliable messaging are specified, using the WebLogic Integration B2B Console, as part of the document exchange definition for a trading partner. For information about configuring trading partners, see "Configuring Trading Partners" in *Online Help for the WebLogic Integration B2B Console*.

Whether a message must be sent reliably is determined by the value you specify for the delivery semantics option. Valid values are:

- OnceAndOnlyOnce—This is the default value. The recipient application or process receives the message only one time. Messages are resent in the case of failure, and duplicate messages are ignored.

  When you specify OnceAndOnlyOnce delivery semantics, you can also specify other quality of service parameters using the B2B Console. They include:

  - Number of Retries—Number of times WebLogic Integration resends a message in specific situations, such as timeouts, network failures, and so on. The default is 3.
  - Interval—Amount of time, in milliseconds, WebLogic Integration waits before trying to resend a message. The default is 60,000 milliseconds.
  - Time to Live—The life span of a message (in seconds). If the message is not delivered and processed by the recipient during this time period, WebLogic Integration stops trying to send the message. The default is zero seconds. A value of zero specifies that the message never expires. If you specify a nonzero value for Time to Live, ensure that the quality of service parameters maintain the following relationship:

    ```
    number_of_retries * interval < time_to_live
    ```

- BestEffort—Reliable delivery semantics are not used. When an application receives a message that it cannot deliver, it does not take any action to recover or to notify anyone of the problem. The sender of the message does not attempt to recover from any failure.

Configuring Security

You configure SSL security for ebXML through the WebLogic Integration B2B Console and the WebLogic Server Administration Console. The following sections outline the steps to follow to configure security in each of two scenarios:

- Both Trading Partners Deployed on WebLogic Integration
- One Trading Partner Deployed on WebLogic Integration and Another Deployed on WebLogic Integration – Business Connect

Both Trading Partners Deployed on WebLogic Integration

For a scenario in which each trading partner in a given ebXML conversation deploys WebLogic Integration, you must configure the following:

- SSL protocol

  Use the WebLogic Server Administration Console to configure your SSL protocol, as described in "Configuring the SSL Protocol and Mutual Authentication" in "Configuring Security" in *Implementing Security with B2B Integration*.

- Trading partner security

  In the B2B Console, configure the following:

  - Client certificate and an associated private key for the local trading partner
  - Client certificate for the remote trading partner
  - Server certificate for the remote trading partner
  - Secure transport for the HTTPS endpoint

  For information about how to use the B2B Console to configure the certificates and the secure transport, see "Configuring Trading Partner Security" in "Configuring Security" in *Implementing Security with B2B Integration*.

One Trading Partner Deployed on WebLogic Integration and Another Deployed on WebLogic Integration – Business Connect

This section describes the HTTPS configuration for a scenario in which one trading partner in an ebXML conversation deploys WebLogic Integration, and the second trading partner deploys WebLogic Integration – Business Connect. Both client authentication and server authentication are enabled on both sides.

You can export WebLogic Integration trading partner data from the B2B Console to provide to your trading partners, as described in "Using the WebLogic Integration B2B Console" on page 2-3. However, certificate information is not included in the trading partner XML file exported from the B2B Console.

Certificate information must be exchanged outside the scope of the trading partner XML file. That is, a WebLogic Integration trading partner must provide certificate files that can be imported by a WebLogic Integration – Business Connect trading partner, and vice versa.

Configuring Security for a WebLogic Integration Trading Partner

This section explains how to configure the following:

- SSL Protocol
- Trading Partner Security

SSL Protocol

Use the WebLogic Server Administration Console to configure the SSL protocol for your WebLogic Integration trading partner, as described in "Configuring the SSL Protocol and Mutual Authentication" in "Configuring Security" in *Implementing Security with B2B Integration*.

The following table describes the parameters you must define through the WebLogic Server Administration Console.

Table 2-3 SSL Configuration Fields in WebLogic Server Administration Console

Tab	Field	Description
SSL	Server Private Key Alias	Specify the alias for the Keystore entry for the server private key.
	Server Private Key Passphrase	Specify a passphrase for the server private key. Such a passphrase is required for every key in the Keystore. (The Keystore has a passphrase, and each entry has its own passphrase.)
	Server Certificate File Name	Specify the location of the public key file for WebLogic Server. You must obtain this file from a trusted security vendor, as described in "Configuring the SSL Protocol and Mutual Authentication" in "Configuring Security" in *Implementing Security with B2B Integration*.

Table 2-3 SSL Configuration Fields in WebLogic Server Administration Console (Continued)

Tab	Field	Description
	Client Certificate Enforced check box	Select this option to enable mutual authentication between WebLogic Integration and the WebLogic Integration – Business Connect trading partner.
	Cert Authenticator	Specify `com.bea.b2b.security.WLCCertAuthenticator` in this field. The `WLCCertAuthenticator` class is an implementation of the WebLogic Server `CertAuthenticator` class. It maps the digital certificate for a trading partner to the corresponding trading partner user defined in the WebLogic Integration repository.
	Key Encrypted check box	Select this option to indicate that the private key for WebLogic Server has been encrypted with a password.
	Use Java check box	Select this option to enable the use of native Java libraries. WebLogic Server provides a pure-Java implementation of the SSL protocol. Native libraries enhance performance for SSL operations on the Solaris, Windows NT, and IBM AIX platforms.
	Handler Enabled check box	Skip this option; it is no longer used.
	Export Key Lifespan	Enter the number of times WebLogic Server can use an exportable key between a domestic server and an exportable client before generating a new key. The more secure you want WebLogic Server to be, the fewer times the key should be used before a new key is generated.
	SSL Login Timeout	Enter the duration (in milliseconds) allowed for a login sequence. If the specified duration is exceeded, the login is timed out. Enter 0 to disable.
	Certificate Cache Size	Check the number of certificates being held that have not been redeemed by tokens.
	Hostname Verification Ignored	Set this option to disable the installed implementation of the `weblogic.security.SSL.HostnameVerifier` class when WebLogic Server is acting as a client to another application server.
	Hostname Verifier	Check the name of the class that implements the `weblogic.security.SSL.HostnameVerifier` class.
SSL Ports	Enable SSL Listen Port check box	Select this option to enable the use of the SSL protocol on the listen port on the server. To select the port on which WebLogic Integration listens for SSL connections, set the Listen Port attribute.

Table 2-3 SSL Configuration Fields in WebLogic Server Administration Console (Continued)

Tab	Field	Description
	SSL Listen Port	Specify the dedicated port on which WebLogic Integration listens for SSL connections.
	Enable Domain Wide Administration Port	Check whether a secure administration port for the server has been enabled. This value displayed here is set in the DomainMBean for this server.
	Domain Wide Administration Port	Check the secure administration port number for the server. The value displayed here is set in the DomainMBean for this server.
	Local Administration Port Override	Enter a new secure administration port number for the server. The setter is used to override the same field in the DomainMBean for this server. If the value of this field is not zero, then the corresponding field in the DomainMBean is used for the server. TIf you want to use this port, SSL must be configured and enabled.

When you set the parameters described in the previous table, your specifications are written to the `config.xml` file for your domain. The following code listing is an excerpt from a `config.xml` file that shows a sample SSL configuration created for a WebLogic Integration trading partner.

Listing 2-1 SSL Settings in config.xml File

```
<SSL CertAuthenticator="com.bea.b2b.security.WLCCertAuthenticator"
     CertificateCacheSize="10" ClientCertificateEnforced="true"
     Enabled="true" ListenPort="7002"
     Name="myserver"
     ServerCertificateFileName="config/wli-server_cert.pem"
     ServerKeyFileName="config/wli-server_key.pem"/>
```

For information about the `config.xml` files in your WebLogic Integration domains, see "WebLogic Integration Sample Configuration Files" in *Starting, Stopping, and Customizing BEA WebLogic Integration*.

Note: Add the following line to the `startWeblogic` script in your domain if you get the error message listed after it:

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true

Error: Host name doesn't match DN name
```

Trading Partner Security

In the B2B Console, configure the following:

- Client certificate and an associated private key for the local trading partner
- Client certificate for the remote trading partner
- Server certificate for the remote trading partner
- Secure transport for the HTTPS endpoint

Note: When you configure remote trading partner client and server certificates, you are prompted to enter the location and filename of each certificate on the WebLogic Integration machine where the client and server certificates are stored. Your WebLogic Integration – Business Connect trading partner may provide a single certificate for both client and server authentication (that is, a self-signed certificate). In this case, you should specify the same location for both the client certificate and the server certificate.

For information about how to use the B2B Console to configure the certificates and the secure transport, see “Configuring Trading Partner Security” in “Configuring Security” in *Implementing Security with B2B Integration*.

Configuring Security for a WebLogic Integration – Business Connect Trading Partner

To configure your WebLogic Integration – Business Connect trading partner to use HTTPS to exchange ebXML messages, complete the following tasks:

1. Create your certificates and export them using the Export Certificate functionality in the WebLogic Integration – Business Connect Administrator tool. (For details, see “Exporting Your Certificate for Backup or Distribution” in “Keys and Certificates ” in *Using WebLogic Integration – Business Connect*.)

   The certificates can be self-signed or signed by an external certificate authority (CA). If self-signed, a single certificate is generated for both client and server authentication. Export the certificate in DER encoded binary X.509 (.cer) format. (PKCS #7 format is not supported by WebLogic Integration.)

2. Import the certificate or certificates provided by your WebLogic Integration trading partner using the procedure described in “Importing a Partner’s Certificate” in“ Keys and Certificates” in *Using WebLogic Integration – Business Connect*.

   Note: When you import more than one certificate file, only one is marked Active. Select the appropriate certificate to mark as Active.

CHAPTER

3 Using Workflows with ebXML

This section describes how you design and manage ebXML business processes using workflows, and it provides the steps you must take to start the exchange of ebXML business messages between trading partners. It includes the following topics:

- ebXML Business Messages
- Participating in an ebXML Conversation
- ebXML Sample Application
- ebXML Sample Application

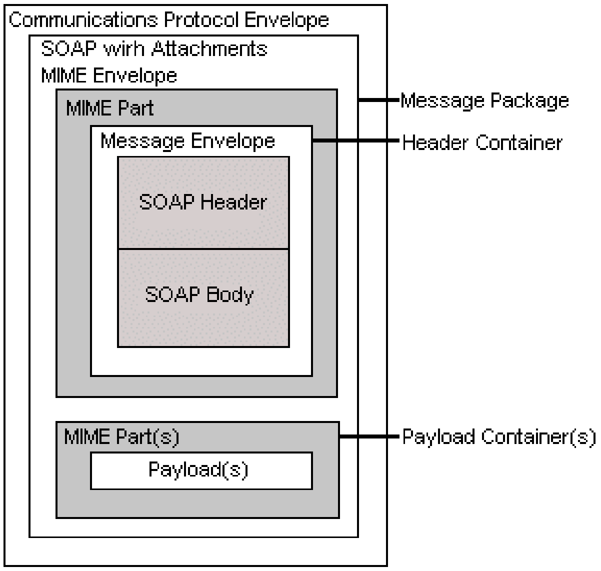
ebXML Business Messages

A business message is the basic unit of communication between trading partners. Business messages are exchanged as part of a conversation. The roles in a conversation are implemented by workflows, which choreograph the exchange of business messages.

An ebXML business message contains one XML business document and zero or more attachments. An ebXML message is a communication protocol-independent MIME/Multipart message envelope, referred to as a *message packag*e. All message packages are structured in compliance with the SOAP Messages with Attachments specification.

The following figure represents the structure of a business message exchanged in a conversation based on the ebXML business protocol.

Figure 3-1 ebXML Business Message



The message package shown in the preceding figure illustrates the following logical MIME parts:

- Header Container—Logical container in which one SOAP 1.1-compliant message is stored. This SOAP message is an XML document; its root element is the SOAP Envelope, which, in turn, contains the following elements:
  - SOAP Header—ebXML-specific header elements. The SOAP Header is a generic mechanism for adding features to a SOAP message.
  - SOAP Body—Container for message service handler control data and information related to the payload parts of the message.
- Payload Container—Zero or more MIME parts containing application-level payloads. Payloads are not file-type-specific; they may contain XML or binary data.

Participating in an ebXML Conversation

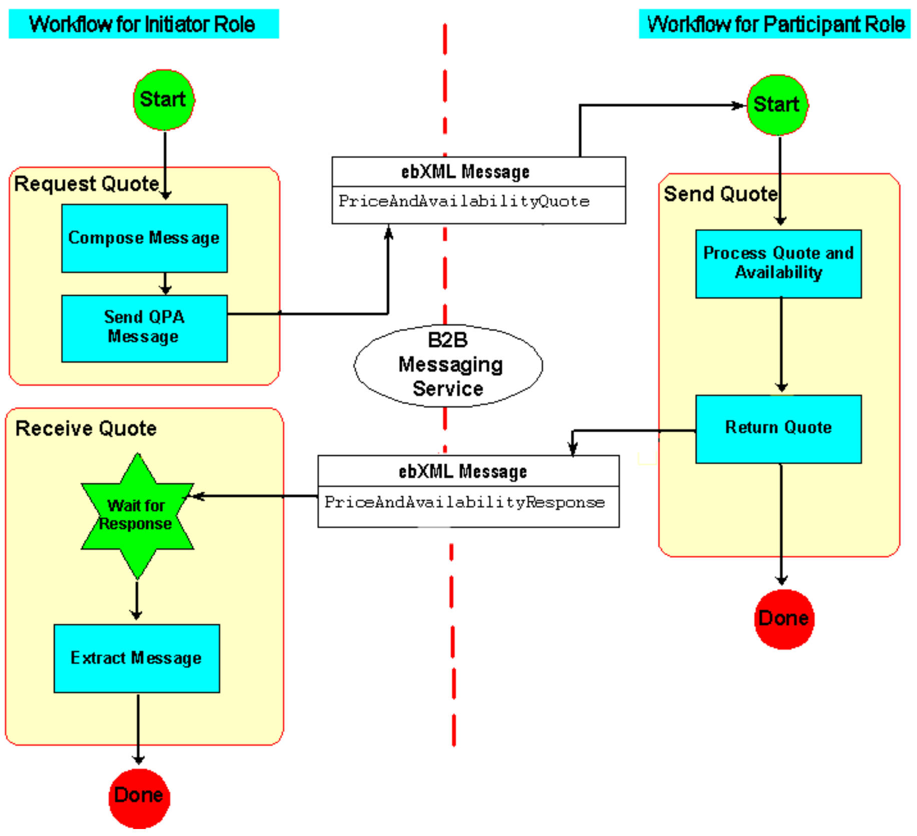
A conversation is a series of business message exchanges between trading partners that is predefined by a conversation definition. A fundamental step in defining a conversation is creating the workflows that execute the roles in the conversation. A conversation definition that references the ebXML business protocol always contains two roles: *initiator* and *participant*. Each workflow template associated with an ebXML conversation performs only one of the two roles.

An ebXML plug-in extends the functionality of the WebLogic Integration Studio and process engine to allow you to design and execute the workflows that implement the roles in an ebXML conversation. Workflows are business processes. ebXML-based business processes can be designed as public or private processes.

Note: The information in this section is based on the assumption that the reader is familiar with the WebLogic Integration Studio. A detailed discussion of how to use the Studio is beyond the scope of this section. For information about starting and using the Studio, see *Using the WebLogic Integration Studio* and *Learning to Use BPM with WebLogic Integration*.

The following figure represents a simple ebXML Query Price and Availability (QPA) conversation that contains two roles, initiator and participant, and a high-level, hypothetical workflow for each role. Each trading partner implements one workflow for its role in the conversation. Each role has its own set of tasks required to send and receive the right ebXML messages at the right time.

Figure 3-2 Sample ebXML Conversation



The preceding figure illustrates the exchange of ebXML messages between trading partners in a conversation:

- One trading partner workflow, represented on the left in the preceding figure, starts the ebXML conversation by sending the first message.
- One trading partner workflow, represented on the right in the preceding figure, participates in the conversation by sending an ebXML message in response to the message it receives.

Key components of the ebXML conversation are described in the following sections:

- Starting an ebXML Conversation
- Sending an ebXML Message
- Receiving an ebXML Message
- Ending an ebXML Conversation

Starting an ebXML Conversation

Each trading partner in an ebXML conversation is assigned to one of two roles: *initiator* or *participant*. An ebXML conversation is started when a workflow sends an ebXML message in a new conversation. (For details, see "Sending a Message in a New Conversation" on page 3-7.)

The workflow that sends the first message in a conversation can be started in any of four ways:

- Timed—The workflow is started at an exact date and time that you specify.
- Manual—The workflow is started from the WebLogic Integration Worklist, meaning that it is started by a Worklist user.
- Called—The workflow is started by being called from another workflow.
- Event—The workflow is started when an XML document arrives on a JMS queue. (This event must not be an event that occurs as a result of the receipt of an ebXML business message.)

Sending an ebXML Message

A workflow task can send an ebXML message by invoking an ebXML action called Send ebXML Message. To add this action to a task node in a workflow:

1. In the Studio, right-click a definition in the list of template definitions, and select Open.

   The specified workflow template is displayed in the Studio.

2. Right-click the task node, and select Properties from the shortcut menu.

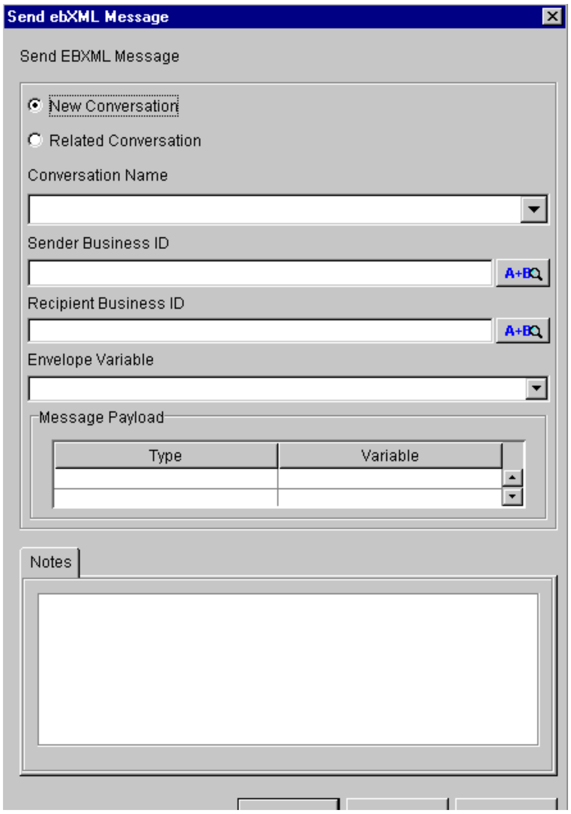
   The Task Properties dialog box is displayed.

3. Click Add to display the Add Action dialog box.

4. Choose ebXML Actions→Send ebXML Message.

5. Click OK to display the Send ebXML Message dialog box, as shown in the following figure.

Figure 3-3 Send ebXML Message Dialog Box



WebLogic Integration supports two modes for sending ebXML messages. Which mode is used depends on the circumstances under which the message is sent: the first message in a given conversation is sent in New Conversation mode; subsequent (response) messages are sent in Related Conversation mode.

For example, two ebXML messages are sent in the sample ebXML conversation illustrated in Figure 3-2. The first message (`PriceAndAvailabilityQuote`) is sent from the initiator workflow to the participant workflow; the second (`PriceAndAvailabilityResponse`), from the participant workflow to the initiator workflow.

The type of information required in the Send ebXML Message dialog box depends on which message type you select:

- Sending a Message in a New Conversation
- Sending a Message in a Related Conversation

Sending a Message in a New Conversation

In the case of `PriceAndAvailabilityQuote`, the first ebXML message in our sample conversation, select New Conversation in the Send ebXML dialog box, as shown in Figure 3-3, and then specify the following information:

- Conversation Name—Select the ebXML-based conversation from the drop-down list to specify the conversation in which the message is sent. The conversation definition is stored in your WebLogic Integration repository. (For information about configuring a conversation definition, see "ebXML Business Messages" on page 3-1.)
- Sender Business ID—Specify a business ID for the sender of the message. This is a unique value that identifies the trading partner sending the ebXML message. The value can be a static value or a value that is dynamically evaluated via the expression builder.

  Note: We recommend that you do not define business IDs as static values. Instead use a variable or an XPath expression to evaluate a business ID at runtime. In this way, your workflows can be used by multiple trading partners.
- Recipient Business ID—Specify a business ID for the recipient of the message. This is a unique value that identifies the trading partner receiving the ebXML message. The value can be a static value or a value that is dynamically evaluated via the expression builder. (See the **Note** for the previous bullet.)
- Envelope Variable—Specify a variable which will contain the ebXML message header (the SOAP envelope) *after* the composed ebXML message is sent. You

can retrieve information, such as the message ID or conversation ID, from this variable in order to trace the status of the message.

- Message Payload data—Specify zero or more payloads. For each payload, specify a type (XML or ebXML binary) and an associated variable.

Sending a Message in a Related Conversation

In the case of `PriceAndAvailabilityResponse` (our sample ebXML message), which is sent by the participant workflow to the initiator workflow in response to the `PriceAndAvailabilityQuote` ebXML message, select Related Conversation in the Send ebXML dialog box, which is shown in Figure 3-3.

When you select Related Conversation in the Send ebXML dialog box, the Conversation Name, Sender Business ID, and Recipient Business ID fields are not available for you to write. These attributes are set once for a given conversation; that is, when the first ebXML message is sent in a new conversation. (For details, see "Sending a Message in a New Conversation" in the preceding section.) You do not need to specify the Conversation Name, Sender Business ID, and Recipient Business ID when you send a message in a related conversation because the system obtains these attributes from the previous ebXML message exchange. (For details, see "Receiving an ebXML Message.")

However, in the Message Payload fields, you must specify the payload type (XML or ebXML binary) and associated variables for zero or more payloads.

Receiving an ebXML Message

WebLogic Integration supports two methods of receiving ebXML messages in workflows: Start nodes and Event nodes. Which node is used depends on the circumstances under which the message is received:

- A workflow for a trading partner in an ebXML-based conversation can be triggered by receiving an ebXML message from the workflow that initiates the conversation. This initial ebXML message triggers the start node of the participant workflow, which is defined as an ebXML Message start. The participant workflow in our sample conversation, illustrated in Figure 3-2, is an example of a workflow triggered in this way.

- Any ebXML workflow (the workflow for either the initiator role or the participant role) can wait at an event node for an ebXML message, such as a reply to a request. In our sample conversation, illustrated in Figure 3-2, the Wait for Response node in the initiator workflow is an example of such an event node.

Start Nodes

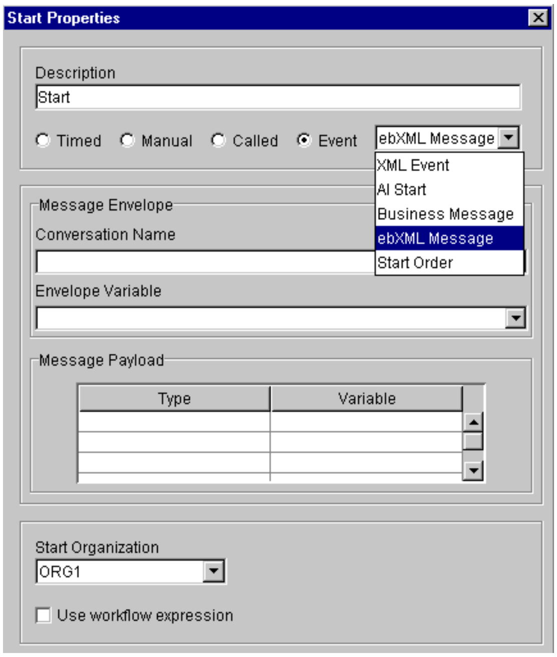
A workflow for a trading partner in an ebXML-based conversation can be started when it receives an ebXML message from another trading partner. The Studio provides an ebXML-specific option in the Start node to design these workflows.

To specify an ebXML event at the Start node in a workflow (for example, the Start node for the participant workflow illustrated in Figure 3-2):

1. Right-click the Start node, and then select Properties from the shortcut menu.

   The Start Properties dialog box is displayed.

Figure 3-4 Start Properties Dialog Box



Start Properties

Description: Start

Timed [✓ Event] Manual Called — ebXML Message

Start Organization: ORG1

Use workflow expression

2. Select the Event option and, in the drop-down list, select EBXML Message.
3. In the Message Envelope fields, from the appropriate drop-down list, select:
   - The conversation to which an incoming message applies
   - A variable in which to store the message envelope
4. In the Message Payload field, select a type (XML or ebXML binary) for zero or more message attachments, and an associated variable in which to store the data for each attachment.

   Note: Each row in the Message Payload field represents an attachment in the associated ebXML message. When an ebXML message includes multiple attachments, the order of the rows in this field represents the order of the attachments in the message. If you leave a row in the Message Payload field empty, the corresponding attachment in the ebXML message is ignored when the message is received.
5. Click OK to save your settings and close the dialog box.

Event Nodes

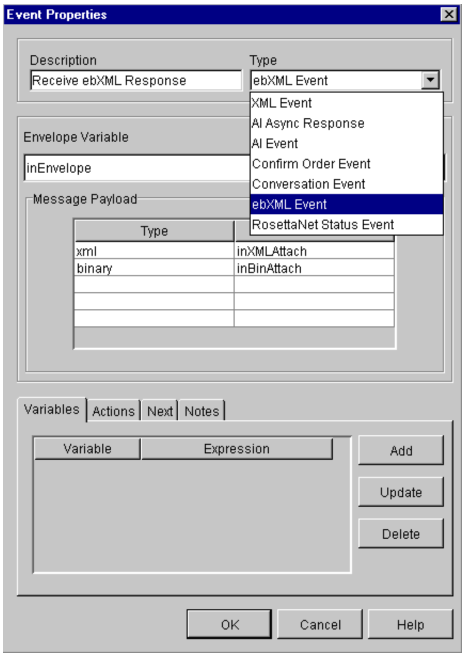
A workflow can contain events designed to be triggered when an ebXML message is received for a particular instance of the workflow. An event that waits to receive an ebXML message must be defined as type ebXML Event. Such an event is triggered at run time when the appropriate ebXML message is received in the conversation.

To define an event that waits to receive an ebXML message:

1. Right-click the Event node, and then select Properties from the shortcut menu.

   The Event Properties dialog box is displayed.

Figure 3-5 Event Properties Dialog Box



Description: Receive ebXML Response

Type: ebXML Event

Envelope Variable: inEnvelope

Message Payload

Type	
xml	inXMLAttach
binary	inBinAttach

Variable	Expression

2. Select EBXML Event from the Type drop-down list.

3. From the drop-down list in the Envelope Variable field, select a variable in which to store the message envelope.

4. In the Message Payload field, select a type (XML or ebXML binary) for zero or more message attachments, and an associated variable in which to store the data for each attachment.

Note: Each row in the Message Payload field represents an attachment in the associated ebXML message. When an ebXML message includes multiple attachments, the order of the rows in this field represents the order of the attachments in the message. If you leave a row in the Message Payload field empty, the corresponding attachment in the ebXML message is ignored when the message is received.

5. Click OK to save your settings and close the dialog box.

Ending an ebXML Conversation

An ebXML-based conversation ends when the exchange of ebXML messages is complete for both trading partners.

Contrast this behavior to the XOCP business protocol. WebLogic Integration supports an XOCP conversation management service, meaning that the workflow responsible for initiating the conversation also ends the conversation and sends an end-of-conversation message to each workflow in the conversation. (For details, see "Ending Collaborative Workflows" in *Creating Workflows for B2B Integration*.)

In the case of an XOCP conversation, to define a conversation termination property, you must select the Custom option on the Done node in the workflow that initiated the conversation. Do not select this option for ebXML-based conversations: to the contrary, you should ensure that in workflows for ebXML conversations, the Custom option for Done nodes is not selected.

ebXML Sample Application

An ebXML sample application is provided in the `SAMPLES_HOME\integration\samples\ebxml` directory, where `SAMPLES_HOME` represents the WebLogic Platform samples directory. This sample demonstrates two workflows: one designed for the role of initiator; the other, for the role of participant in a Query Price and Availability (QPA) conversation. Both workflows are designed and used to manage an ebXML-based business process between two trading partners, each of which deploys WebLogic Integration. For information about this sample, including instructions for running it, see "ebXML Sample" in *Running the B2B Integration Samples*.

As additional ebXML samples become available, they will be posted on the BEA dev2dev Online site at the following URL:

```
http://dev2dev.bea.com
```

Index