

*SeeBeyond ICAN Suite*

# ebXML Manager Composite Application User's Guide

*Release 5.0.5*



The information contained in this document is subject to change and is updated periodically to reflect changes to the applicable software. Although every effort has been made to ensure the accuracy of this document, SeeBeyond Technology Corporation (SeeBeyond) assumes no responsibility for any errors that may appear herein. The software described in this document is furnished under a License Agreement and may be used or copied only in accordance with the terms of such License Agreement. Printing, copying, or reproducing this document in any fashion is prohibited except in accordance with the License Agreement. The contents of this document are designated as being confidential and proprietary; are considered to be trade secrets of SeeBeyond; and may be used only in accordance with the License Agreement, as protected and enforceable by law. SeeBeyond assumes no responsibility for the use or reliability of its software on platforms that are not supported by SeeBeyond.

SeeBeyond, e\*Gate, e\*Way, and e\*Xchange are the registered trademarks of SeeBeyond Technology Corporation in the United States and/or select foreign countries. The SeeBeyond logo, SeeBeyond Integrated Composite Application Network Suite, eGate, eWay, eInsight, eVision, eXchange, eView, eIndex, eTL, ePortal, eBAM, and e\*Insight are trademarks of SeeBeyond Technology Corporation. The absence of a trademark from this list does not constitute a waiver of SeeBeyond Technology Corporation's intellectual property rights concerning that trademark. This document may contain references to other company, brand, and product names. These company, brand, and product names are used herein for identification purposes only and may be the trademarks of their respective owners.

© 2004 by SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

**This work is confidential and proprietary information of SeeBeyond and must be maintained in strict confidence.**

Version 20041119091900.

# Contents

List of Figures	5
-----------------	---

List of Tables	6
----------------	---

---

## Chapter 1

<b>Introduction</b>	<b>7</b>
About ebXML Protocol Manager	7
What's New in This Release	7
What's in This Document	8
Organization of Information	8
Document Conventions	8
Screenshots	8
Supporting Documents	9
SeeBeyond Web Site	9
SeeBeyond Documentation Feedback	9

---

## Chapter 2

<b>Introducing ebXML Protocol Manager</b>	<b>10</b>
About eXchange Integrator	10
eXchange and the ICAN Suite	11
eXchange Architectural Overview	11
Process Overview	12

---

## Chapter 3

<b>Installing ebXML Protocol Manager</b>	<b>14</b>
System Requirements	14
Supported Operating Systems	14
Database Support	15
Database for eXchange Partner Management and Message Tracking	15
Database for Persistence and Monitoring via eInsight Engine	15
Before You Install	15

<b>Installing the Product Files</b>	<b>15</b>
Uploading ebXML Protocol Manager to the Repository	16
Refreshing Enterprise Designer	18
<b>Database Scripts</b>	<b>20</b>
<b>Additional Policy JAR Files Required to Run SME</b>	<b>20</b>
Additional JAR Files for ebXML Encryption	21

## Chapter 4

<b>Implementation Scenario: ebXML</b>	<b>22</b>
Overview of the Sample ebXML Implementation	22
<b>Initial Setup Steps</b>	<b>23</b>
Installing the Sample Files for ebXML	23
Importing the Sample Projects	24
<b>Design Steps in Enterprise Designer</b>	<b>24</b>
Setting Up the Sample Environment for Berlin (Company B)	24
Viewing and Activating the B2B Host Project for Berlin	26
Configuring the eXchange Service with Crypto Information	28
Creating and Activating the Project Deployment Profiles	29
Setting Up the Atlanta (Company A) Projects	31
<b>Design Steps in eXchange Partner Manager</b>	<b>32</b>
Importing the Trading Partners	32
Configuring Trading Partner Parameters for the “Berlin” TP	34
Parameters for the Delivery Channel	34
Parameters for the Internal Delivery Channel	35
Activating the “Berlin” Trading Partner	35
Configuring Trading Partner Parameters for the “Atlanta” TP	35
Parameters for the Delivery Channel	36
Parameters for the Internal Delivery Channel	37
Activating the “Atlanta” Trading Partner	37

## Glossary

<b>Glossary of Acronyms</b>	<b>38</b>
	<b>40</b>
<b>Index</b>	<b>41</b>

# List of Figures

Figure 1	eXchange and the ICAN Suite	11
Figure 2	eXchange Architecture	13
Figure 3	Enterprise Manager ADMIN Page: Uploading the ProductsManifest.xml File	17
Figure 4	Update Center Wizard: Select Modules to Install	18
Figure 5	Update Center Wizard: Progress Bars	19
Figure 6	Update Center Wizard: Restart Enterprise Designer	19
Figure 7	Sample Environment, Before Configuration	26
Figure 8	Deployment Profile for B2B Host, Before Activation	28
Figure 9	Configuring Private Keys and Trust Stores	28
Figure 10	Deployment Profile for ebXML_Protocol (BerlinHost)	30
Figure 11	Activating the ebXML Error-Handler Project	31
Figure 12	Initial ePM Screen	33

# List of Tables

Table 1	Document Conventions	8
Table 2	JRE Versions Listed by Operating System	20
Table 3	B2B Protocol Processes Used in the ebXML Sample Implementation	22
Table 4	Delivery Channel Parameters for Trading Partner “Berlin”	34
Table 5	Delivery Channel Parameters for Trading Partner “Berlin”	36

# Introduction

This chapter introduces you to this guide, its general purpose and scope, and its organization. It also provides sources of related documentation and information.

### What's in This Chapter

- [About ebXML Protocol Manager](#) on page 7
- [What's New in This Release](#) on page 7
- [What's in This Document](#) on page 8
- [Supporting Documents](#) on page 9
- [SeeBeyond Web Site](#) on page 9
- [SeeBeyond Documentation Feedback](#) on page 9

---

## 1.1 About ebXML Protocol Manager

This user's guide provides instructions and background information for all users of the ebXML Manager Composite Application. This guide is designed for managers, system administrators, and others who use ebXML Protocol Manager.

The purpose of this guide is to help you do the following:

- Understand the nature of ebXML Protocol Manager.
- Understand the function of ebXML Protocol Manager.
- Understand the relationship of ebXML Protocol Manager to other components of the SeeBeyond Integrated Composite Application Network (ICAN) Suite.
- Learn about the ebXML Protocol Manager components and editors and how to use them in your environment.

---

## 1.2 What's New in This Release

This release of ebXML Protocol Manager has been updated for compatibility with the architecture of eXchange 5.0.5.

## 1.3 What's in This Document

This section provides a brief outline of this user's guide.

### 1.3.1. Organization of Information

This guide is arranged as follows:

- **Chapter 1, "Introduction"** provides an overview of this document's purpose, contents, writing conventions, and supported documents.
- **Chapter 2, "Introducing ebXML Protocol Manager"** discusses general features and architecture of eXchange.
- **Chapter 3, "Installing ebXML Protocol Manager"** provides step-by-step instructions for installing the eXchange product and setting it up for use.
- **Chapter 4, "Implementation Scenario: ebXML"** provides step-by-step procedures for creating implementation scenarios that provide a sample of how eXchange can be used to achieve B2B solutions.

In addition, there is a **Glossary of Acronyms** on page 38 that lists and explains special acronyms and initialisms that occur in this guide.

### 1.3.2. Document Conventions

The following conventions are observed throughout this document.

**Table 1** Document Conventions

Text	Convention	Example
Names of buttons, files, icons, parameters, variables, methods, menus, and objects	<b>Bold</b> text	<ul style="list-style-type: none"> <li>▪ Click <b>OK</b> to save and close.</li> <li>▪ From the <b>File</b> menu, select <b>Exit</b>.</li> <li>▪ Select the <b>logicalhost.exe</b> file.</li> <li>▪ Enter the <b>timeout</b> value.</li> <li>▪ Use the <b>getClassname()</b> method.</li> <li>▪ Configure the <b>Inbound</b> File eWay.</li> </ul>
Command line arguments, code samples	Fixed font. Variables are shown in <b><i>bold italic</i></b> .	<code>bootstrap -p <b><i>password</i></b></code>
Hypertext links	<b>Blue</b> text	See " <b>Document Conventions</b> " on page 8
Hypertext links for Web addresses (URLs) or email addresses	<b>Blue underlined</b> text	<a href="http://www.seebeyond.com">http://www.seebeyond.com</a> <a href="mailto:docfeedback@seebeyond.com">docfeedback@seebeyond.com</a>

### 1.3.3. Screenshots

Depending on what products you have installed, and how they are configured, the screenshots in this document may differ from what you see on your system.



---

## 1.4 Supporting Documents

For more information about eXchange and the ICAN Suite, refer to the following:

<b>Title</b>	<b>Filename</b>
<i>SeeBeyond ICAN Site Installation Guide</i>	ICAN_Install_Guide.pdf
<i>SeeBeyond ICAN Suite Deployment Guide</i>	Deployment_Guide.pdf
<i>eGate Integrator User's Guide</i>	eGate_User_Guide.pdf
<i>eGate Integrator System Administration Guide</i>	Sys_Admin_Guide.pdf
<i>eGate Integrator JMS Reference Guide</i>	eGate_JMS_Reference.pdf
<i>Oracle eWay Intelligent Adapter User's Guide</i>	Oracle_eWay.pdf
<i>HTTP(S) eWay Intelligent Adapter User's Guide</i>	HTTPS_eWay.pdf
<i>eXchange Integrator User's Guide</i>	eXchange_User_Guide.pdf
Readme for ICAN 5.0.5	Readme.txt

---

## 1.5 SeeBeyond Web Site

The SeeBeyond Web site is your best source for up-to-the-minute product news and technical support information. The site's URL is:

<http://www.seebeyond.com>

---

## 1.6 SeeBeyond Documentation Feedback

We appreciate your feedback. Please send any comments or suggestions regarding this document to:

[docfeedback@seebeyond.com](mailto:docfeedback@seebeyond.com)

# Introducing ebXML Protocol Manager

This chapter provides a general overview of eXchange Integrator and its place in the ICAN Suite, including system descriptions, general operation, and basic features.

## What's in This Chapter

- [About eXchange Integrator](#) on page 10

For more information about eGate and eXchange, see the corresponding user's guides.

---

## 2.1 About eXchange Integrator

eXchange provides an open B2B protocol framework to support standard EDI and B2B business protocols and enveloping protocols. Not only does it support existing standard protocols, with an extensive set of prebuilt B2B protocol processes, it also provides the tools and framework to create and adopt new protocols and to build custom BPs.

B2B modeling semantics are exposed so that business rules can be added and tailored to address the particular needs of each eBusiness challenge. The tight integration with the rest of the ICAN Suite provides validation, logging, and reporting capabilities, and because each logical step within any business rule is accessible anywhere along the entire BP, the design tools provide complete end-to-end visibility.

The trading partner management facility, eXchange Partner Manager (ePM), is provided via a Web interface. For easy interoperability, trading partners can be configured by importing Collaboration Protocol Agreements (CPAs); or trading partner profiles can be configured manually. Each trading partner profile is identified by a unique ID determined by the enterprise, and delivery channels can be configured for acknowledgments, compression, industry-standard encryption and decryption, and nonrepudiation.

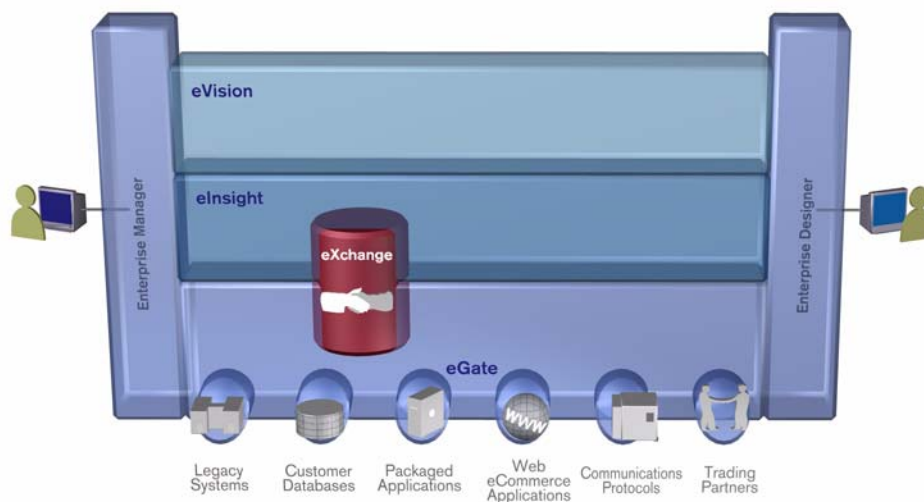
At run time, all steps in the business process, from initial receipt of the message to final delivery to the trading partner, are tracked in real time and also stored in the eXchange database. The Web-based message/package tracker provides tools for retrieving and filtering tracked message and envelope information. Used in conjunction with the other monitoring tools of the ICAN suite, this provides the enterprise with a complete solution for troubleshooting and managing all eBusiness activities.

## 2.1.1. eXchange and the ICAN Suite

eXchange is part of the SeeBeyond ICAN Suite of products. eXchange provides a Web-based trading partner configuration and management solution for automating and securely managing business partner relationships for real-time interaction between the enterprise and its partners, suppliers, and customers.

eXchange is tightly integrated with the ICAN Suite and runs as a component within the ICAN Suite environment. Figure 1 illustrates how eXchange and other ICAN Suite components work together.

**Figure 1** eXchange and the ICAN Suite



## 2.1.2. eXchange Architectural Overview

eXchange centers around the concept of a *Delivery Channel Profile* for each trading partner relationship. Delivery Channel Profiles are configured within eXchange for use by runtime components. Each profile specifies which B2B protocol(s) to use, where and how to receive inbound messages from trading partners, how to configure and secure messages in this channel, and how and where to deliver outbound messages to trading partners.

eXchange uses the following key components:

- **B2B Host Designer** — Using the **Enterprise Designer** GUI framework, eXchange provides an editor for setting up B2B environments, called the *B2B Host Designer*. Each B2B Host provides one or more protocol-specific delivery channels that are exposed to the eXchange database via the Repository. Delivery channels provided by the B2B Host can then be accessed by specific trading partners and reused.
- **B2B Services and Protocols** — eXchange provides two other special editors: The *eXchange Service Designer*, for modeling the choreography of B2B interactions between the internal system and an external trading partner, as mediated by the

B2B Host; and the *eXchange Protocol Designer*, for setting up the message flow logic (*B2B protocol processes*) required to address a specific business challenge, using such activity elements as branching activities, timers, and exception handling.

Prebuilt B2B protocol processes for such industry-standard B2B protocols as AS2 and ebXML are available as separately installable add-on products. eXchange also provides the flexibility of allowing the enterprise to create and configure custom protocol processes.

eXchange also supplies **Channel Manager**, a collection of eXchange Services that provides trading partner-specific integration to the enterprise. Industry-standard transport protocols (FTP, HTTP, HTTPS, SMTP) are supported by Channel Manager and other eWays.

- **Trading Partner Configuration** — eXchange provides a Web-based GUI, eXchange Partner Manager (ePM), for configuring and managing B2B trading partners. Each trading partner has one or more delivery channels that specify the protocols to be used, with corresponding transport mechanisms—encryption parameters such as certificate, signature, and keystore information, acknowledgment-handling preferences, and so forth.
- **eXchange Database** — eXchange uses an Oracle database to mediate retrieval of trading partner information and to store run-time information on message tracking.
- **Message Tracking**— eXchange provides a specific MessageTracker application that can be combined with other processes in a project just by dragging it into the Connectivity Map, as well as a Web-based message tracking GUI with powerful filtering and searching capabilities.

### 2.1.3. Process Overview

Using eXchange to create a business solution consists of three phases:

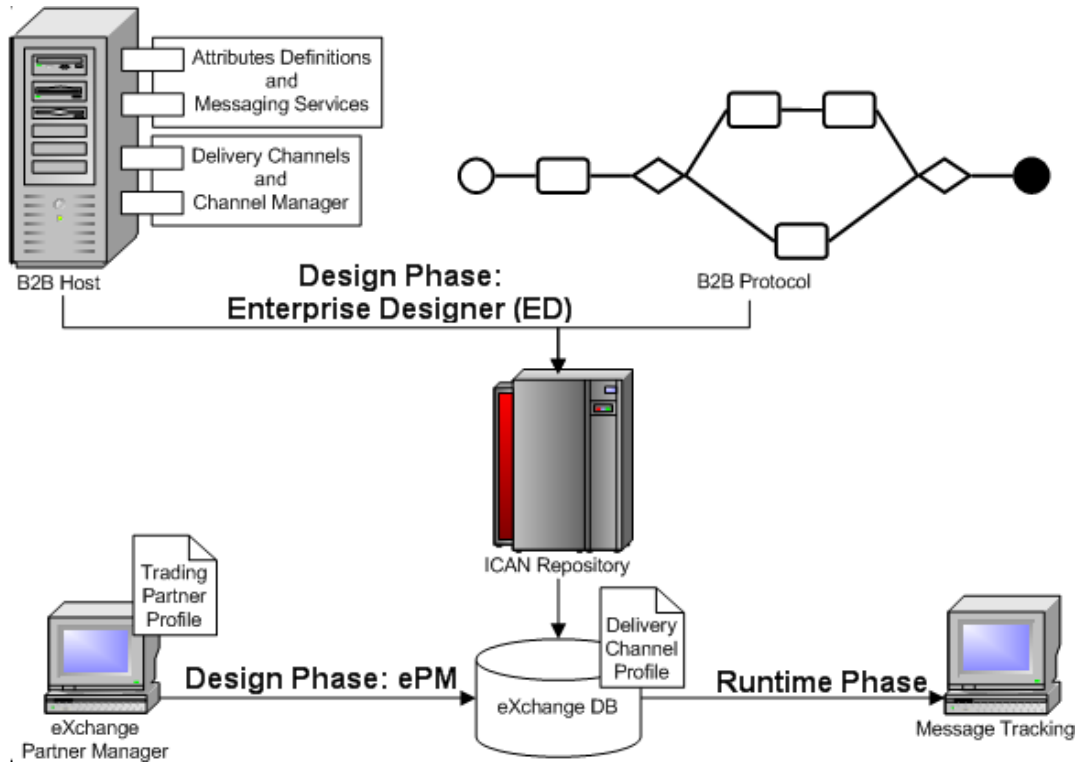
- Design phase within Enterprise Designer
- Design phase within eXchange Partner Manager
- Runtime phase

The purpose of the design phases is to: Create metadata for Delivery Channel Profiles; set up business logic for B2B protocol processes; configure connections with external systems; create and configure trading partners; and associate each trading partner relationship with a Delivery Channel Profile (DCP) configuration. Activating a trading partner exposes its DCP configuration settings to the eXchange database.

At run time, the Logical Host reads the DCP configuration from the database to determine: How to receive and process inbound messages; which business logic to run; and how to process and deliver outbound messages. Results are written to the database, where they can be filtered and viewed by the Message Tracker facility.

These phases are illustrated in Figure 2.

Figure 2 eXchange Architecture



# Installing ebXML Protocol Manager

This chapter provides the prerequisites and steps for installing ebXML Manager Composite Application.

## What's in This Chapter

- [System Requirements](#) on page 14
- [Before You Install](#) on page 15
- [Installing the Product Files](#) on page 15
- [Database Scripts](#) on page 20
- [Additional Policy JAR Files Required to Run SME](#) on page 20

---

## 3.1 System Requirements

This section lists system requirements and database requirements. The *SeeBeyond ICAN Suite Installation Guide* and the **Readme.txt** file, available on the product media and via Enterprise Manager (Documentation tab), contain up-to-date operating system requirements for each supported platform.

### 3.1.1. Supported Operating Systems

ebXML Protocol Manager is available on the following operating systems:

- Microsoft Windows 2000 SP3 or SP4, Windows XP SP1a, and Windows Server 2003
- Sun Solaris 8 and Solaris 9, with required patches
- HP Tru64 V5.1A, with required patches
- HP-UX 11.0, 11i (PA-RISC), and 11i v2.0 (11.23) with required patches and parameter changes
- IBM AIX 5.1L and AIX 5.2 (either 64-bit kernel or 32-bit kernel with 64-bit extension), with required maintenance level patches
- Red Hat Linux 8 (Intel x86) and Linux Advanced Server 2.1 (Intel x86)

## 3.1.2. Database Support

### Database for eXchange Partner Management and Message Tracking

The eXchange database is required. It provides a run-time persistent store for trading partner management and message tracking. For eXchange, the following databases are supported:

- Oracle 8.1.7
- Oracle 9.0.1
- Oracle 9.2

### Database for Persistence and Monitoring via eInsight Engine

In addition, eXchange can optionally use the eInsight engine (supplied with eXchange) to collect and persist data from your B2B protocol processes. This provides for recovery, and also enables some monitoring and reporting capabilities in Enterprise Manager. The eInsight engine supports the following databases:

- Oracle 8.1.7, 9.0.1, and 9.2
- Sybase 12.5
- Microsoft SQL Server 2000
- IBM DB2 Universal Database 8.1

---

## 3.2 Before You Install

Before you begin installing ebXML Protocol Manager, make sure of all the following:

- All projects that will be re-used after the eXchange installation have been exported. Installing this release of eXchange.sar introduces changes to system projects (that is, projects such as SeeBeyond > eXchange and eXchange > Deployment) that delete or permanently modify previous contents of these system projects.
- A Repository server is running on the machine where you will be uploading the product files.

---

## 3.3 Installing the Product Files

The steps for installing ebXML Protocol Manager are the same as for other products in the ICAN Suite. You can find general product installation instructions in the *ICAN Suite Installation Guide*, which is available on the product media and can also be accessed via Enterprise Manager (Documentation tab).

### 3.3.1. Uploading ebXML Protocol Manager to the Repository

#### Before you begin

- A Repository server must be running on the machine where you will be uploading the product files.
- The following ICAN **.sar** files must have already been uploaded to this Repository:
  - ♦ eGate Enterprise Designer (**eGate.sar**) 5.0.5
  - ♦ Batch eWay adapter (**BatcheWay.sar**)
  - ♦ Oracle eWay adapter (**OracleWay.sar**)
  - ♦ File eWay adapter (**FileeWay.sar**) — This not an installation requirement, but it is required by the sample implementation
  - ♦ eXchange Integrator (**eXchange.sar**) 5.0.5
  - ♦ HTTP(S) eWay adapter (**HTTPeWay.sar**)
  - ♦ Secure Message Extension (**SMEWebServices.sar**)

**Note:** *SMEWebServices.sar* is required for such features as encryption/decryption, signature verification, certificate authentication, and compression/decompression.

#### To upload ebXML Protocol Manager product files to the Repository

- 1 On a Windows machine, start a Web browser and point it at the machine and port where the Repository server is running:

```
http://<hostname>:<port>
```

where

- ♦ *<hostname>* is the name of the machine running the Repository server.
- ♦ *<port>* is the starting port number assigned when the Repository was installed.

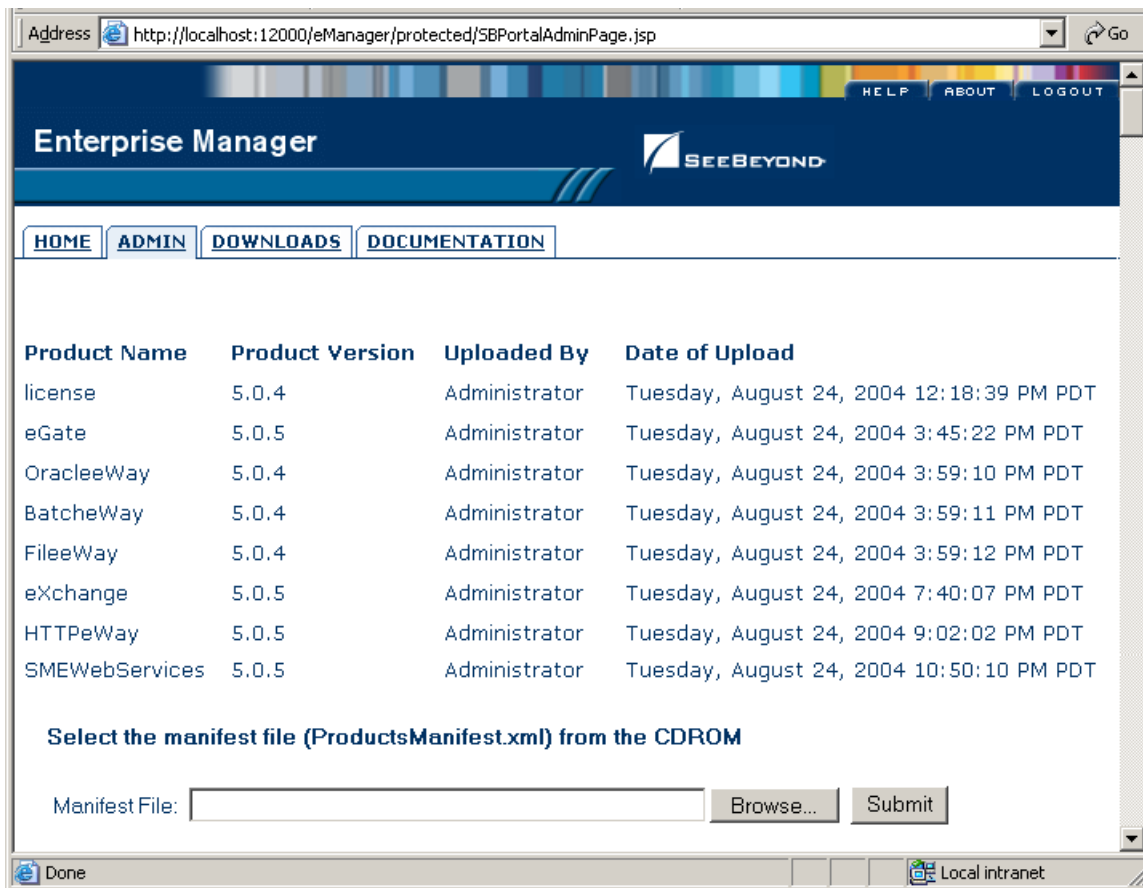
For example, the URL you enter might look like either of the following:

```
http://localhost:12001  
http://serv1234.company.com:19876
```

- 2 On the Enterprise Manager **SeeBeyond Customer Login** page, enter your username and password.
- 3 When Enterprise Manager responds, click the **ADMIN** tab. See Figure 3.



**Figure 3** Enterprise Manager ADMIN Page: Uploading the ProductsManifest.xml File



- 4 In the ADMIN page, under "Select the manifest file [...]", click **Browse**.
- 5 In the **Choose file** dialog, navigate to the correct directory for the products you want to upload (for example: **CoreProducts** for eXchange, or **Add-ons** for eWays), click the **ProductsManifest.xml** filename, and then click **Open**.
- 6 In the ADMIN page, click **Submit**.  
The lower half of the ADMIN page lists the product files you are licensed to upload.
- 7 In the Products column, find **ebXML Manager Composite Application**, and then click the **Browse** button for it.
- 8 In the **Choose file** dialog, click **ebXML\_Manager.sar** and then click **Open**.
- 9 Repeat the previous two steps for other eXchange-related product **.sar** files you are licensed to upload, such as OTD libraries for ASC X12 or UN/EDIFACT.
- 10 For documentation and samples, also upload the corresponding [...] **Docs.sar** files, such as **ebXML\_ManagerDocs.sar** (and X12\_OTD\_Docs.sar, and so forth).

### 3.3.2. Refreshing Enterprise Designer

The following steps are needed only if you have newly uploaded (or re-uploaded) eGate.sar or any other .sar file that affects the Enterprise Designer GUI framework.

**Tip:** How can you determine whether to use the Update Center? Start Enterprise Designer and, on the **Tools** menu, click **Update Center**; if there are any items under “SeeBeyond 5.0” besides “**Base ESR**”, you need to follow these steps.

#### Before you begin

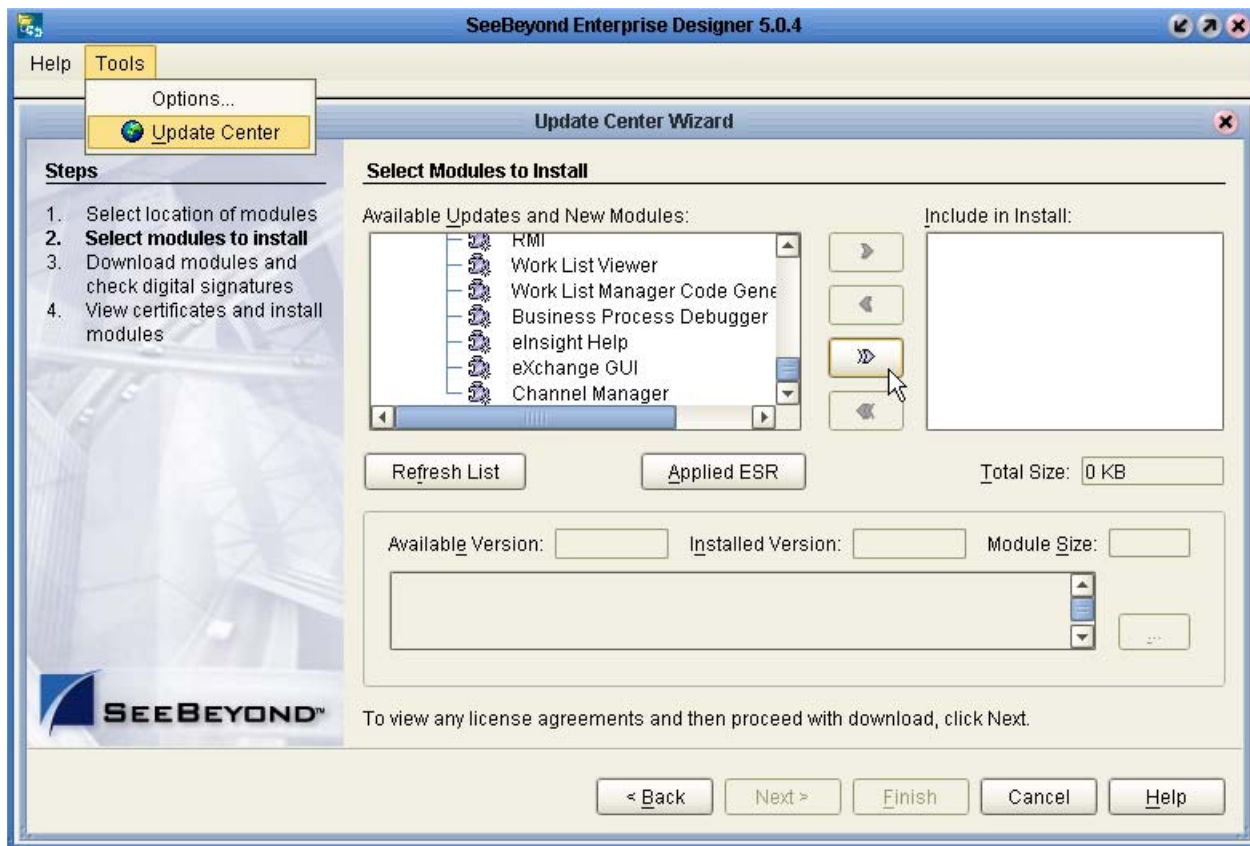
- You must have already downloaded and installed Enterprise Designer.
- A Repository server must be running on the machine where you uploaded the ebXML Protocol Manager product files.

#### To refresh an existing installation of Enterprise Designer

- 1 Start Enterprise Designer.
- 2 On the **Tools** menu, click **Update Center**.

The Update Center shows a list of components ready for updating. See Figure 4.

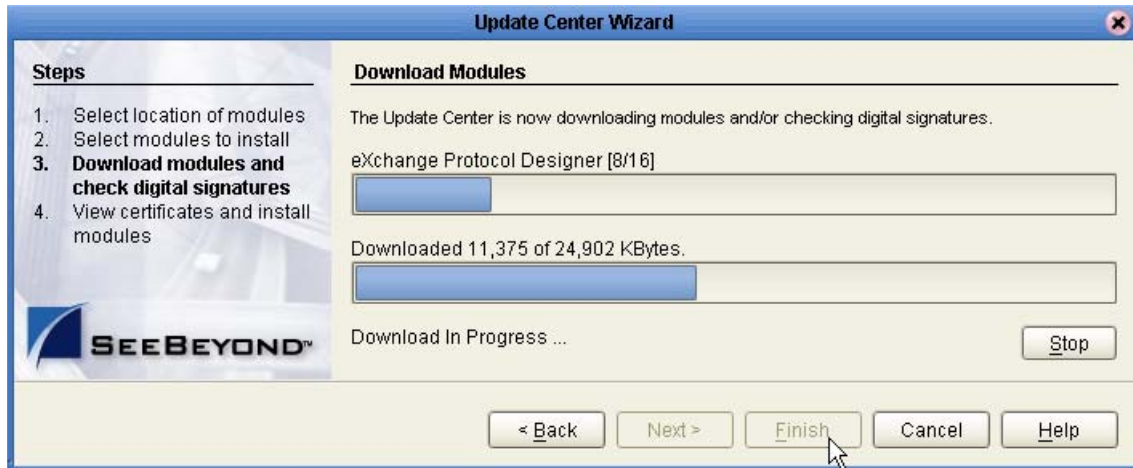
**Figure 4** Update Center Wizard: Select Modules to Install



**Note:** Depending on what products you have installed, and how they are configured, the screenshots pictured may differ from what you see on your system.

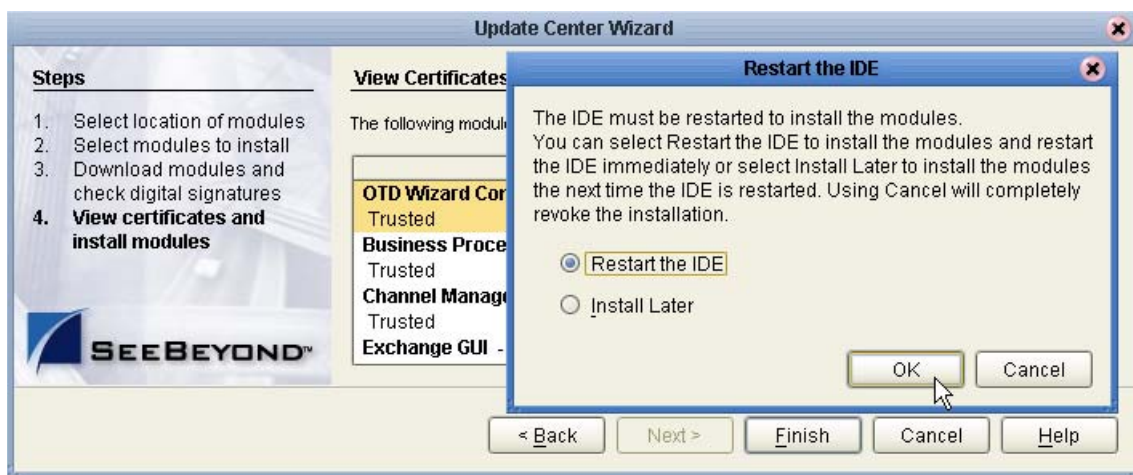
- 3 Click **Add All** (the button with a doubled chevron pointing to the right).  
All modules move from the Available/New pane to the **Include in Install** pane.
- 4 Click **Next** and, in the next window, click **Accept** to accept the license agreement.  
The wizard shows you the progress of the download. See Figure 5.

**Figure 5** Update Center Wizard: Progress Bars



- 5 When the progress bars indicate the download has ended, click **Next**.
- 6 Review the certificates and installed modules, and then click **Finish**.
- 7 When prompted to restart Enterprise Designer, click **OK**. See Figure 6.

**Figure 6** Update Center Wizard: Restart Enterprise Designer



When Enterprise Designer restarts, the installation of ebXML Manager Composite Application is complete, and you can use all eXchange tools provided on the Enterprise Designer framework.

## 3.4 Database Scripts

The eXchange database schema collects and persists data about your trading partner profiles, and also allows you to track message delivery history. eXchange provides database scripts to create and upgrade these database schemas for eXchange. For information, see the *eXchange Integrator User's Guide*.

## 3.5 Additional Policy JAR Files Required to Run SME

For strong encryption/decryption and signatures/verification, using libraries supplied with Secure Messaging Extension (SME), you must download and apply additional policy .jar files. The type of .jar files required depends on the JVM are using. Refer to your JVM vendor for exact details on the specific policy .jar file requirements.

Use Table 2 to determine which JRE is included in the eGate logical host.

**Table 2** JRE Versions Listed by Operating System

Operating System	JRE	URL
Windows, Solaris, HP-UX, Linux, Tru64	1.4.2	<a href="http://java.sun.com/j2se/1.4.2/download.html">http://java.sun.com/j2se/1.4.2/download.html</a>
AIX	1.4.1	<a href="http://java.sun.com/products/archive/j2se/1.4.1_07/index.html">http://java.sun.com/products/archive/j2se/1.4.1_07/index.html</a>

To download the required policy.jar files

- 1 Scroll to the bottom of the web page listed in Table 2 for your logical host's JRE.
- 2 Click the [DOWNLOAD](#) link for **Unlimited Strength Jurisdiction Policy Files 1.4.2**. (or, for AIX, **Unlimited Strength Jurisdiction Policy Files 1.4.1**).
- 3 Click the link to download the .zip file containing the required policy .jar files (in other words: for JRE 1.4.2, follow the link [Download jce policy-1 4 2.zip](#); or, for JRE 1.4.1, follow the link [Download jce policy-1 4 1.zip](#))
- 4 Extract the following required policy .jar files:
  - ♦ local\_policy.jar
  - ♦ US\_export\_policy.jar
- 5 Then, for each of your logical hosts, replace the versions of these files in:
 

```
<logicalhost>\jre\lib\security\
```
- 6 In addition, if you are running a repository on AIX, also replace the versions of these files in:
 

```
<AIXrepository>/jre/1.4.x/lib/security/
```

For complete information on SME, see the *Secure Messaging Extension User's Guide*.

### 3.5.1. Additional JAR Files for ebXML Encryption

For ebXML, you must also download and apply an additional **.jar** file, **xss4j.jar**, for XML security. This can be downloaded, as part of the XML Security Suite, from the following Web site:

<http://www.alphaworks.ibm.com/tech/xmlsecuritysuite>

The only file needed is **xss4j.jar**. After extracting it, copy it to the following location for each logical host:

```
<logicalhost>\stcis\lib\
```

# Implementation Scenario: ebXML

This chapter provides an implementation scenario showing how eXchange can be used to achieve B2B solutions using the ebXML protocol.

## What's in This Chapter

The steps for the sample implementation occur in these phases:

<b>Initial Setup Steps</b>	In these steps, you ensure that prerequisites are met, obtain the necessary sample materials, extract sample files, and import the sample projects. See section 4.1 <a href="#">on page 23</a> .
<b>Design Steps in Enterprise Designer</b>	In these steps, you use Enterprise Designer to add and configure externals, view components in the B2B host project, and activate the host, creating an eXchange Service. Then you view the components in the main project and activate it. See section 4.2 <a href="#">on page 24</a> .
<b>Design Steps in eXchange Partner Manager</b>	In these steps, you use the eXchange Partner Management (ePM) facility to import trading partners, view and specify parameter values, and activate them. See section 4.3 <a href="#">on page 32</a> .

## Overview of the Sample ebXML Implementation

The sample implementation combines inbound and outbound message processing. It makes use of the following ebXML protocol processes supplied by SeeBeyond:

**Table 3** B2B Protocol Processes Used in the ebXML Sample Implementation

For Inbound Messages	For Outbound Messages	Other
ebXML_Inbound	DcpFinderCollab	
ebXmllIdentification	ToInternalDeliverySelector	ToInternalJMSDeliveryProtocol
ValidateCollab	RetrieveAckCollab	ToInternalFileDeliveryProtocol
DecryptCollab	EncryptCollab	SendExternalCollab
VerifyCollab	SignCollab	
SignAckCollab	VerifyAckCollab	
ExtractPayloadCollab	PackCollab	
	TrackerCollab	JMSErrorHandler
	PackMshSignalCollab	ErrorHandlerSelector
	TrackAckCollab	OutboundHTTPClient

**Table 3** B2B Protocol Processes Used in the ebXML Sample Implementation (Continued)

For Inbound Messages	For Outbound Messages	Other
Base64EncodeCollab		

## 4.1 Initial Setup Steps

Initial setup steps:

- [Installing the Sample Files for ebXML](#) on page 23
- [Importing the Sample Projects](#) on page 24

### 4.1.1 Installing the Sample Files for ebXML

These steps assume the existence of a temporary eXchange directory for sample files, such as `C:\temp\exchange\`. You will extract the sample files to this directory so that you can conveniently access the files in later procedures.

#### To install the sample files

*Before you begin:* Your repository must already be running, and you must be logged in to Enterprise Manager. If you have already uploaded the documentation for eXchange, you can skip steps 1 and 2 and start with step 3.

- 1 In the ADMIN tab, if you have not already done so, browse to the [...] \Documentation \ **ProductsManifest.xml** file and submit it.
- 2 In the ADMIN tab, if you have not previously done so, browse to the **ebXML\_ManagerDocs.sar** file, select it, and click the **upload now** button.
- 3 In the DOCUMENTATION tab, in the Products window, click **ebXML Manager Composite Application**.
- 4 In the window that appears on the right side, click **Download Sample**.
- 5 Preserving file paths, extract the files to your `C:\temp\exchange` directory.

*Result:* The following directories and files are created:

```
C:\temp\exchange\Sample\Projects\ebXML_All.zip
C:\temp\exchange\Sample\Projects\FileToJmsFeeder.zip
C:\temp\exchange\Sample\Projects\JmsErrorToFileReceiver.zip
C:\temp\exchange\Sample\Projects\JmsPayloadToInternalReceiver.zip
C:\temp\exchange\Sample\Projects\Environments.zip
C:\temp\exchange\Sample\TradingPartners\ebXML_AtlantaHost_TP.xml
C:\temp\exchange\Sample\TradingPartners\ebXML_BerlinHost_TP.xml
C:\temp\exchange\Sample\Data\ebXML_Atlanta_1-XML.~in
C:\temp\exchange\Sample\Data\ebXML_Atlanta_2-XML-binary.~in
C:\temp\exchange\Sample\Data\ebXML_Berlin_1-XML.~in
C:\temp\exchange\Sample\Data\ebXML_Berlin_2-XML-binary.~in
C:\temp\exchange\Sample\Data\payload.binary
C:\temp\exchange\Sample\Data\payload.xml
C:\temp\exchange\Sample\Data\Out\
```

## 4.1.2 Importing the Sample Projects

### To import the sample projects

*Before you begin:* Your repository must already be running, and you must be logged in to Enterprise Designer. If your repository already has a project at the root level whose name is identical to one of the seven projects you will be importing, you must delete or rename it before you start.

- 1 In Project Explorer, right-click the repository and, on the popup menu, click: **Import**
- 2 In the **Import Manager** dialog, browse to the folder where you installed the sample files (such as C:\temp\exchange\Sample\Projects), select **ebXML\_All.zip**, and click **Open**.
- 3 Select all four projects, click **Import**, and then click **OK** to clear the confirmation.
- 4 Repeat the previous two steps to import the projects in: **FileToJMSFeeder.zip**; **JmsErrorToFileReceiver.zip**; and **JmsPayloadToInternalReceiver.zip**.
- 5 Close the Import Manager dialog.

*Result:* The Project Explorer tree now displays seven new projects: ebXML\_AtlantaHost, ebXML\_B2B Templates, ebXML\_BerlinHost, ebXML\_Protocol; and FileToJMSFeeder, JmsErrorToFileReceiver, and JmsPayloadToInternalReceiver.

---

## 4.2 Design Steps in Enterprise Designer

For the ebXML sample implementation, design-time steps in Enterprise Designer consist of the following:

- [Setting Up the Sample Environment for Berlin \(Company B\)](#) on page 24
- [Viewing and Activating the B2B Host Project for Berlin](#) on page 26
- [Configuring the eXchange Service with Crypto Information](#) on page 28
- [Creating and Activating the Project Deployment Profiles](#) on page 29
- [Setting Up the Atlanta \(Company A\) Projects](#) on page 31

### 4.2.1 Setting Up the Sample Environment for Berlin (Company B)

The sample assumes you will use default configurations for all servers where possible, and that you will make any changes where needed. For example:


- You must create a new outbound Oracle external and configure it, even if you imported the sample environment; the sample parameters are for reference only.
- If you use anything other than a SeeBeyond Integration Server on ports 18000–18009, make adjustments in steps 3 (ports) and/or 4 (type of Integration Server).
- If you want your HTTP client to use SSL, see the *HTTP(S) eWay Intelligent Adapter User's Guide* for eWay settings and Integration Server Web Server configuration.



### To create the sample environment

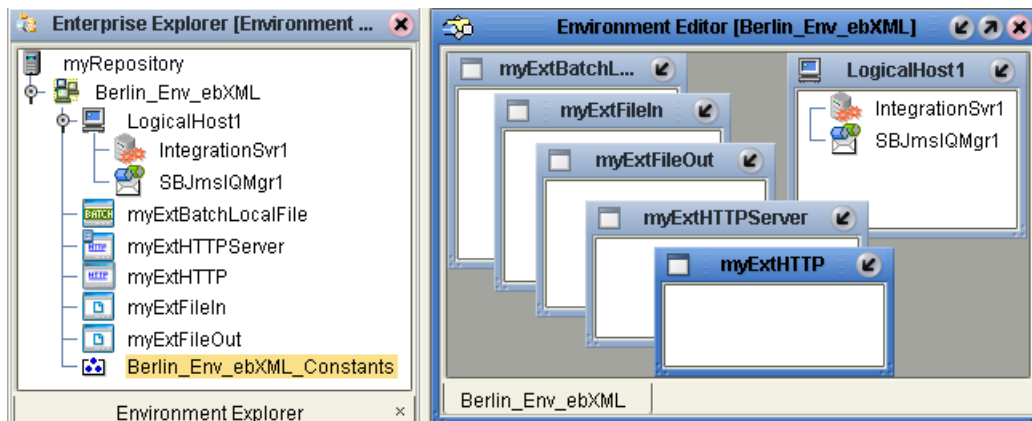
- 1 In Enterprise Designer, near the lower left of the window, click the **Environment Explorer** tab.
- 2 In the Environment Explorer tree, right-click the repository and, on the popup context menu, click **New Environment**
  - ♦ Rename the newly created environment to **Berlin\_Env\_ebXML**.
- 3 Right-click Berlin\_Env\_ebXML and, on the menu, click: **New Logical Host**
  - ♦ Retain the default name: **LogicalHost1**

**Tip:** For a second or subsequent logical host: Right-click it and open its properties, click **Logical Host Configuration** and change the value **Logical Host Base Port** to a larger multiple of 1000 (19000 if ports 19000-19009 are unused; otherwise 20000, or 21000), and then close the properties sheet.

- 4 Right-click LogicalHost1 and click: **New SeeBeyond Integration Server**
  - ♦ Retain the default name: IntegrationSvr1
- 5 Right-click LogicalHost1 and click: **New SeeBeyond JMS IQ Manager**
  - ♦ Retain the default name: SBJmsIQMgr1
- 6 Right-click Berlin\_Env\_ebXML > **New BatchLocalFile External System**
  - ♦ Name it **myExtBatchLocalFile** and click OK.
- 7 Right-click Berlin\_Env\_ebXML > **New File External System**
  - ♦ Name it **myExtFileIn**, set it to **Inbound File eWay**, and click OK.
- 8 Right-click Berlin\_Env\_ebXML > **New File External System**
  - ♦ Name it **myExtFileOut**, set it to **Outbound File eWay**, and click OK.
- 9 Right-click Berlin\_Env\_ebXML > **New HTTP Server External System**
  - ♦ Name it **myExtHTTPServer** and click OK.
- 10 Right-click Berlin\_Env\_ebXML > **New HTTP External System**
  - ♦ Name it **myExtHTTP** and click OK.
- 11 Right-click Berlin\_Env\_ebXML > **New Constant**
  - ♦ In the **Create a Constant** dialog, name it **DataRootDirectory**, give it the value **C:\temp\exchange\Sample\Data\Berlin** (if you installed the samples to a location other than C:\temp\exchange, substitute appropriately), and click **OK**; then close the **Variables and Constants** dialog box.
- 12 On the main toolbar, click  Save All.


*Result:* The environment, named **Berlin\_Env\_ebXML**, now has all but two of the externals needed by the projects. Steps for the outbound Oracle external are provided in the following procedure, and the final external will be created by activating the project containing the B2B host.

**Figure 7** Sample Environment, Before Configuration



### To create and configure the Oracle external

*Before you begin:* Your eXchange 5.0.5 Oracle database must be accessible, and you must know its SID, username, and password.

- 1 In the Environment Explorer tree, right-click `Berlin_Env_ebXML` and, on the popup context menu, click **New Oracle External System**
  - ◆ Name it **myExtOracleOut**, designate it **Outbound Oracle eWay**, and click OK.
- 2 Right-click `myExtOracleOut` and configure properties appropriately. For example:
  - ◆ **DatabaseName:** *exch50* (change this to the SID for your eXchange Oracle database)
  - ◆ **DataSourceName:** **local**
  - ◆ **Password:** (replace this with the password for your eXchange 5.0.5 database user)
  - ◆ **PortNumber:** **1521** (change this only if your Oracle administrator changed the default)
  - ◆ **ServerName:** *myMachine* (change this to the hostname of the Oracle server machine)
  - ◆ **User:** *ex\_B* (change this to the username for your "Berlin" eXchange database user)
- 3 When all properties have been configured correctly for your site, click OK.
- 4 Collapse the `Berlin_Env_ebXML` tree, click  **Save All**, and close all canvases.

*Result:* `Berlin_Env_ebXML` now has all but one of the externals needed by the projects.

The final external, an eXchange service, will be created by activating the project that contains the B2B host.

## 4.2.2 Viewing and Activating the B2B Host Project for Berlin

In the Project Explorer tree, open the B2B host project (named `ebXML_BerlinHost`) to display its four components. Below is a summary of the B2B host's contents. Activating

this project will create an eXchange service that acts as a channel manager and provides a connection to the message tracking application and the eXchange database.

### Components of the B2B host project


- **myHost\_MsgingService** is the only message service used by the B2B host. It contains eight messaging actions, alternating inbound and outbound.
- **BerlinHost** is the B2B host itself:
  - ♦ Its Business Protocols references two services (both under the ebXML MAD): **myHost\_MsgingService** (from the current project), and **urn:oasis:names:tc:ebxml-msg:service**
  - ♦ It defines only one external delivery channel (for the ebXML MAD), named **ebxml\_delivery\_channel1**. For transport to and from trading partners, this channel references the standard SeeBeyond-supplied HTTP transport attributes definitions (TADs).
  - ♦ It defines only one internal delivery channel (IDC): **Sender\_IDC**. This uses JMS to communicate in the Sender (toInternal) direction.
- **BerlinHost\_CMap** is the map whose activation will create the eXchange service.
  - ♦ Its only input is an instance of BerlinHost, with two outbound connections.
  - ♦ Its only output is an instance of Oracle, with two inbound connections.
  - ♦ Connecting to both is an instance of a SeeBeyond-supplied tracking application.

### To activate the B2B host, creating the eXchange service

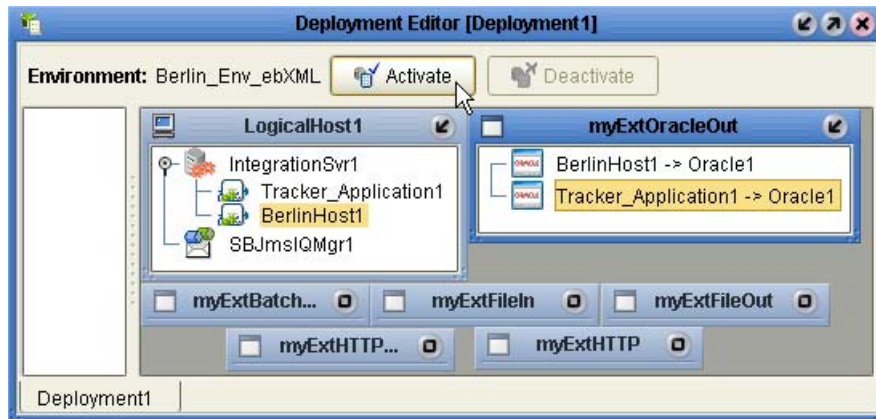
*Before you begin:* Your environment must contain a well-configured Oracle external (see the preceding procedure), and the environment must be named Berlin\_Env\_ebXML— that is, it must correspond to the name of your host project.

- 1 Right-click **ebXML\_BerlinHost** and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Keep the default name (Deployment1), point it at Berlin\_Env\_ebXML, and click OK. The Deployment Editor opens. Its left pane has two services and two Oracle eWays.
- 3 On the right side, minimize all windows except LogicalHost1 and myExtOracleOut.
- 4 One by one, drag the two services into LogicalHost1 and under **IntegrationSvr1**.
- 5 One by one, drag the two Oracle eWays into **myExtOracleOut**. See Figure 8.

**Tip:** *If myExtOracleOut refuses to accept eWays, it may be an indication of:*

- ♦ *The Oracle database instance it references is inaccessible. Ensure it is running and that the myExtOracleOut properties match its hostname, SID, username, and password. If necessary, see “**To create and configure the Oracle external**” on page 26.*
- ♦ *It was misdefined as inbound. Delete myExtOracleOut and re-create it as outbound. Then: Click **Save All**, followed by  **Refresh All from Repository**.*

**Figure 8** Deployment Profile for B2B Host, Before Activation



- 6 Click Save All, and then click **Activate**.
- 7 In response to the dialog box, click **No**; that is, do *not* apply to Logical Host(s).

*Result:* A new external is created, named **BerlinHost1 eXchange Service**. The projects now have all the externals they need. Save all of your work, close all canvases, and click **Refresh All from Repository**.

### 4.2.3 Configuring the eXchange Service with Crypto Information

Since the sample assumes you will be using cryptographic features (encryption, decryption, signatures and verifications), additional steps are required for configuring the eXchange service.

To associate the eXchange service with the private key for “Company B”

- 1 In the Environment Explorer tree, right-click **BerlinHost1 eXchange Service** and, on the popup context menu, click **Properties** to configure the public and private keys for the B2B host’s delivery channel (**ebxml\_delivery\_channel1**). See Figure 9.

**Figure 9** Configuring Private Keys and Trust Stores




- 2 For Signature Key, click **companyb\_pvtkey**

**Tip:** If *companyb\_pvtkey* does not appear in the drop-down list, click the ellipsis [...] and, in the Signature Key dialog, click **Import**. Using alias *companyb\_pvtkey*, import C:\temp\exChange\Sample\Crypto\CompanyB-Key.p12 with the following password (all-lowercase): **companyb**

- 3 For Signature Trust Store, click **mytrust**

**Tip:** If *mytrust* does not appear in the drop-down list, click the ellipsis [...] and, in the Signature Trust Store dialog, click *New*, enter the alias **mytrust**, and click *OK*.

- 4 For Decryption, click **companyb\_pvtkey**
- 5 For Encryption Trust Store, click **mytrust**
- 6 Click **OK** to close the dialog box.
- 7 If a logical host were running, you would also need to do the following:  
In the environment tree, right-click **LogicalHost1** and, on the menu, click **Apply**
- 8 Click  **Save All**, and then close all canvases.

*Result:* Cryptographic information is now associated with the ebxml\_delivery\_channel1 delivery channel for this eXchange service.

## 4.2.4. Creating and Activating the Project Deployment Profiles

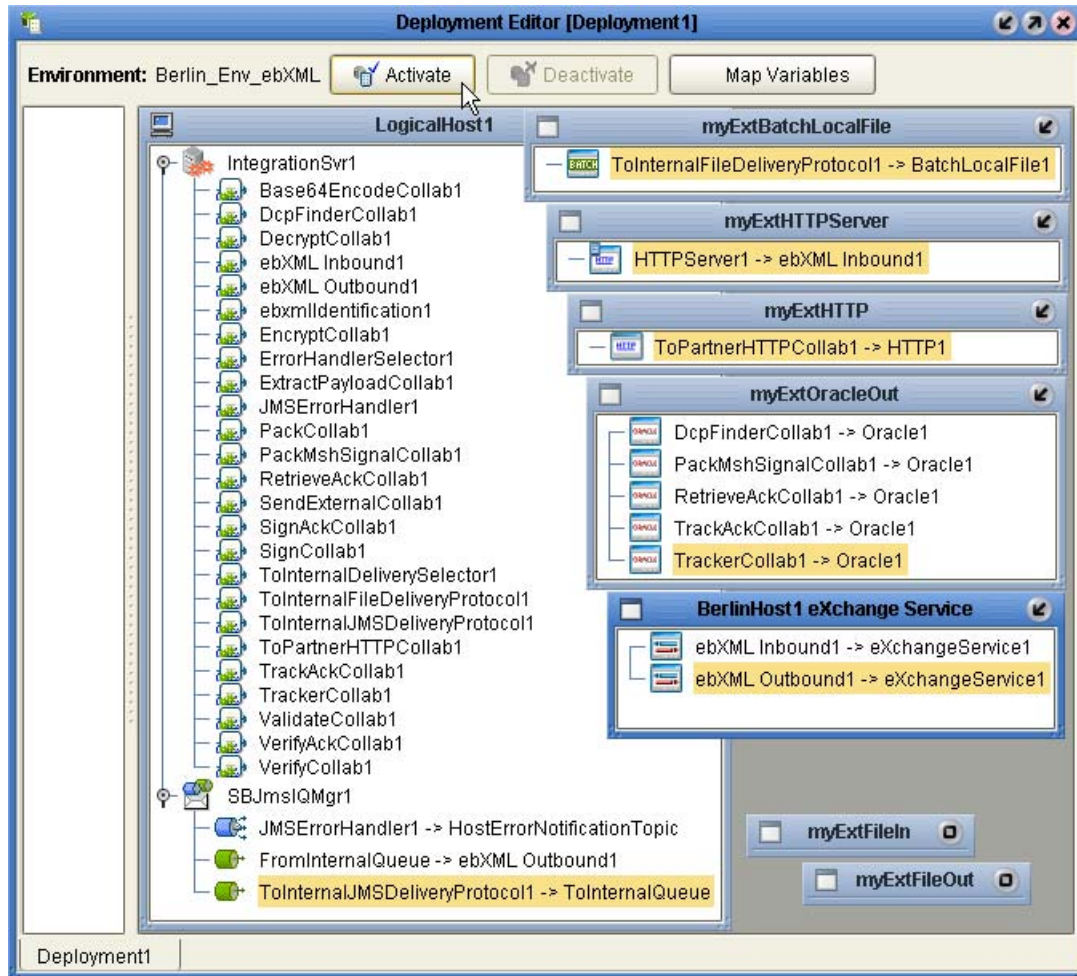
To activate the main ebXML project

*Before you begin:* Your environment must be named Berlin\_Env\_ebXML (that is, it must correspond to the name of your host project), and it must contain an eXchange service; if necessary, see the [procedure on page 27](#).

- 1 In the Project Explorer tree, right-click **ebXML\_Protocol** and, on the popup context menu, point at *New* and click **Deployment Profile**
- 2 Keep the default name (Deployment1), point it at Berlin\_Env\_ebXML, and click *OK*.  
The Deployment Editor's left pane displays: A long list of services; a topic (JMSErrorHandler1->Host...) and two queues; an inbound eWay for HTTPServer; outbound eWays for HTTP and BatchLocalFile; and eWays for one inbound and four outbound eXchange services.
- 3 One by one, drag all services into LogicalHost1 and under **IntegrationSvr1**.
- 4 One by one, drag the topic and queues into LogicalHost1 and under **SbJmsIQMgr1**.
- 5 Drag the inbound HTTPServer eWay into myExtHTTPServer.
- 6 Drag the outbound HTTP eWay into myExtHTTP.
- 7 Drag the outbound BatchLocalFile eWay into myExtBatchLocalFile.
- 8 Drag the eXchange Service eWays into Berlin2Host1 Exchange Service. When you drag and drop the inbound service, specify **ebXML** as the protocol.
- 9 When Deployment1 is complete—that is, when all components in the ebXML\_Protocol project are associated with corresponding servers—click **Activate**. See Figure 10.

**Tip:** Successful activation will take some time. If activation fails because of an improper configuration of myExtHTTP, return to the environment, open the properties of the HTTP external, and provide it with dummy values for truststore and password.

**Figure 10** Deployment Profile for ebXML\_Protocol (BerlinHost)



10 In response to the dialog box, click **No**; that is, do *not* apply to Logical Host(s).

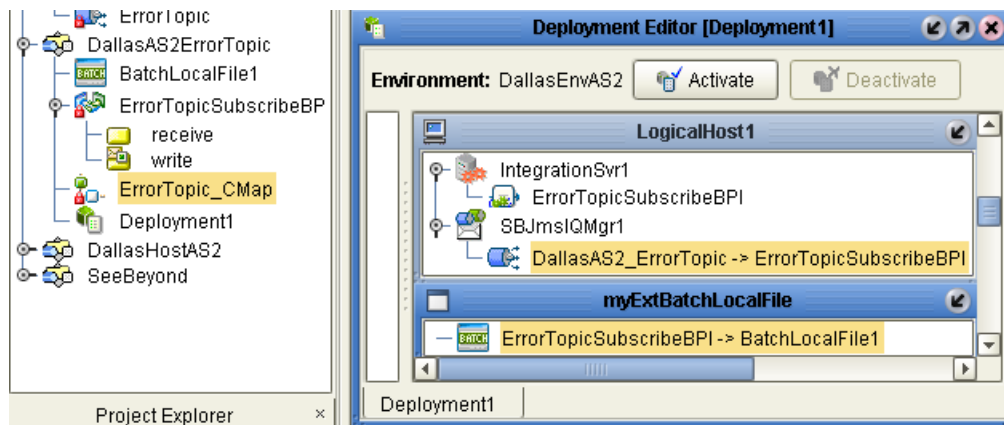
### To activate the ebXML error-handler project

*Purpose:* The purpose of the JmsErrorToFileReceiver project is to provide a durable subscriber for ErrorTopic in the ebXML\_Protocol project, allowing Enterprise Manager to monitor messages in it. A simple B2B protocol, ErrorTopicSubscribeBP, receives its output and writes to C:\temp\exChange\Sample\Data\Berlin\.

*Before you begin:* Your environment must be named Berlin\_Env\_ebXML (that is, it must correspond to the name of your host project), and it must contain an eXchange service; if necessary, see the [procedure on page 27](#).

- 1 In the Project Explorer tree, right-click **JmsErrorToFileReceiver** and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Keep the default name (Deployment1), point it at Berlin\_Env\_ebXML, and click **OK**.
- 3 Drag the BP service into LogicalHost1 and under IntegrationSvr1, and then drag the topic and under SbjmsIQMgr1.
- 4 Drag the BatchLocalFile eWay into myExtBatchLocalFile. See Figure 11.

**Figure 11** Activating the ebXML Error-Handler Project



5 Click **Activate**. Save all, close canvases, and click **Refresh All from Repository**.  
*Result:* All Berlin projects are activated, and will run when the logical host is started.

#### 4.2.5. Setting Up the Atlanta (Company A) Projects

*Purpose:* Because this implementation demonstrates messages between two different B2B hosts, it requires a second complete setup. On the same machine or another, repeat all the preceding steps, starting from section 4.2.1, with the following differences:

- Wherever you see “Berlin”, substitute: “Atlanta”.
- In the Oracle properties, the users for Berlin and Atlanta must be different. For example, in step 2 on page 26, you might substitute user “ex\_A” for “ex\_B”.
- Wherever a reference to company B appears, replace it with company A. For example, in section 4.2.3 “Configuring the eXchange Service with Crypto Information” on page 28, use **companya\_pvtkey** instead of **companyb\_pvtkey**.
- If the same physical machine will be running both logical hosts, ensure that the logical hosts use different base ports. See the tip on page 25.

## 4.3 Design Steps in eXchange Partner Manager

For this sample implementation, design-time steps in eXchange Partner Manager (ePM) consist of the following:

- [Importing the Trading Partners](#) on page 32
- [Configuring Trading Partner Parameters for the “Berlin” TP](#) on page 34
- [Activating the “Berlin” Trading Partner](#) on page 35
- [Configuring Trading Partner Parameters for the “Atlanta” TP](#) on page 35
- [Activating the “Atlanta” Trading Partner](#) on page 37

### 4.3.1 Importing the Trading Partners

*Before you begin:* Your repository and your eXchange 5.0.5 Oracle database must be running and accessible. Enterprise Designer does *not* need to be running, and you do *not* need to have any logical hosts running.

To start eXchange Partner Manager (ePM)

- 1 Start a *new* browser session (that is, do *not* clone a new window of an existing session) pointing it at a repository URL, with **epm** appended. For example:
  - ♦ If your repository were running local on port 12000, the URL would be:  
`http://localhost:12000/epm`
  - ♦ For a repository running on machine herMachine on port 33000, it would be:  
`http://herMachine:33000/epm`
  - ♦ As usual, IP addresses are also permissible:  
`http://10.18.75.85:36271/epm`

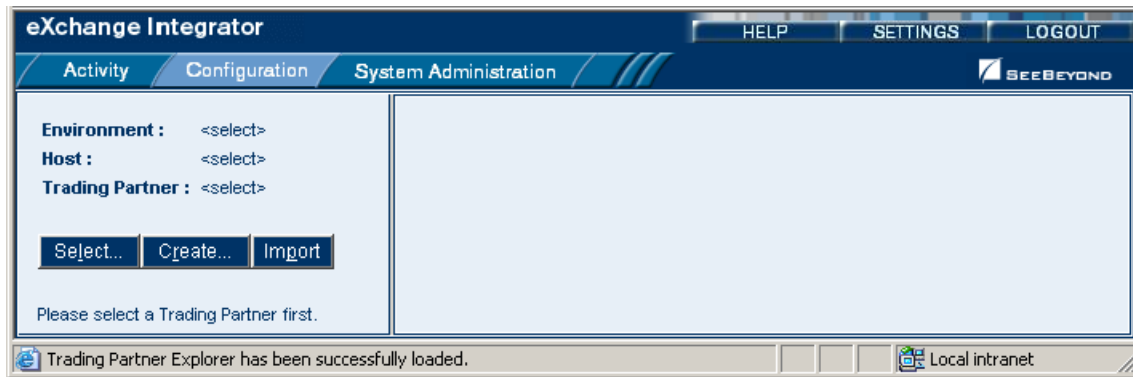
The string **epm** is case sensitive. In other words, ePM, Epm, and EPM are all errors.

- 2 When the sign-in screen appears, enter the Enterprise Manager username and password and click **Sign In**.

*Result:* The status bar (along the lower margin of the window) confirms that Trading Partner Explorer has loaded successfully, and the initial ePM screen appears, with no environment, host, or trading partner. See Figure 12.



Figure 12 Initial ePM Screen



### To import the “Atlanta” trading partner to the Berlin host

- 1 From the initial ePM screen, in the upper left side, click **Import**.  
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **Berlin\_Env\_ebXML** and click **BerlinHost1**.
- 3 Enter **Atlanta**, browse to C:\temp\exchange\Sample\TradingPartners and open **ebXML\_BerlinHost\_TP.xml**, and then click **Import**.

*Result:* In the explorer tree, under Berlin\_Env\_ebXML, new trading partner **Atlanta** appears.

### To find the “Atlanta” trading partner

- 1 In the upper left side of the ePM screen, click **Select**.  
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **Berlin\_Env\_ebXML** and click myAS2Host1.
- 3 Click **Search**, and then click OK.

*Result:* In the explorer tree, under Berlin\_Env\_ebXML, trading partner **Atlanta** reappears.

### To import the “Berlin” trading partner to the Atlanta host

- 1 In the upper left side of the ePM screen, click **Import**.  
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **Atlanta\_Env\_ebXML** and click **AtlantaHost1**.
- 3 Enter **Berlin**, browse to C:\temp\exchange\Sample\TradingPartners and open **ebXML\_AtlantaHost\_TP.xml**, and then click **Import**.

*Result:* In the explorer tree, under Atlanta\_Env\_ebXML, new trading partner **Berlin** appears.

### 4.3.2 Configuring Trading Partner Parameters for the “Berlin” TP

When you imported the trading partner, parameter settings were valuated in part based on parameters stored in the export file, and in part based on the name of the trading partner. In this section, you will set or update the following:

- [Parameters for the Delivery Channel](#) on page 34
- [Parameters for the Internal Delivery Channel](#) on page 35

#### Parameters for the Delivery Channel

*Purpose:* To set the parameters governing Atlanta’s message exchange with “Berlin”. You are configuring a trading partner for the Atlanta environment, and so take the viewpoint of the Atlanta B2B host: “ToPartner” means “to Berlin”; “FromPartner” means “from Berlin”.

To configure the delivery channel parameters for trading partner “Berlin”

- 1 In the explorer (lower left) side of the ePM screen, click **Berlin**.  
The canvas displays the trading partner’s general properties.
- 2 Click the **Components** tab.  
The trading partner’s delivery channel parameters are displayed. See Table 4.

**Table 4** Delivery Channel Parameters for Trading Partner “Berlin”

Binding Name	ebxml_delivery_channel1
ToPartner Transport Name	HTTP
FromPartner Transport Name	HTTP
Packager Name	

- 3 Click the binding name, **ebxml\_delivery\_channel1**.  
The delivery channel’s general properties are displayed.
- 4 Click the **ToPartnerTransport** tab and edit the All Purpose End Point so that it contains the correct URL for Atlanta. For example:  
`http://localhost:18004/Deployment1_servlet/Inbound`  
If the logical host running the Atlanta B2B host is not named localhost, or if its integration server uses a port other than 18004 for its web server, or if the name of the deployment profile is not Deployment1, make the appropriate changes. When you are done, click **Save**.
- 5 Click the **FromPartnerTransport** tab and edit the All Purpose End Point so that it contains the correct URL for Berlin. For example:  
`http://localhost:18004/Deployment1_servlet/Inbound`  
If the logical host running the Berlin B2B host is not named localhost, or if its integration server uses a port other than 18004 for its web server, or if the name of the deployment profile is not Deployment1, make the appropriate changes. When you are done, click **Save**.

- 6 Click the **ToPartnerPackaging** tab, make changes as needed, and then click **Save**.
- 7 Click the **FromPartnerPackaging** tab, make changes as needed, and then click **Save**.

*Result:* For trading “Berlin”, the parameters for external delivery channel `ebxml_delivery_channel1` are now set correctly.

## Parameters for the Internal Delivery Channel

*Purpose:* To set the parameters governing Atlanta’s internal message processing when handling messages received from Berlin or preparing messages to be sent to Berlin. You are configuring a trading partner for the Atlanta environment, and so take the viewpoint of the Atlanta B2B host: “ToInternal” means “to the Atlanta internal (having been received from Berlin)”; “FromInternal” means “from the Atlanta internal (and destined for sending to Berlin)”.

Your configuration of the internal delivery channel depends on your setup.

*Note:* Internal Delivery Channel bindings are always required for a project to run. Although it is possible to activate a trading partner whose messaging actions lack IDCs, the Channel Manager would have no instructions to read any input.

### 4.3.3 Activating the “Berlin” Trading Partner

To activate the “Berlin” trading partner

*Purpose:* To save all the configuration information to the Oracle database to make it available at run time.

*Before you begin:* Your eXchange 5.0.5 Oracle database for the corresponding B2B host must be running.

- 1 In the explorer (lower left) side of the ePM screen, click **Berlin**.
- 2 In the bottom lower left of the canvas, click the **Activate** button.
- 3 In response to the confirmation prompt, click **Activate**.

The canvas displays a confirmation: **Trading Partner is successfully activated**.

*Result:* The **Berlin** trading partner is entirely complete and ready to be run.

### 4.3.4 Configuring Trading Partner Parameters for the “Atlanta” TP

When you imported the trading partner, parameter settings were valuated in part based on parameters stored in the export file, and in part based on the name of the trading partner. In this section, you will set or update the following:

- [Parameters for the Delivery Channel](#) on page 34
- [Parameters for the Internal Delivery Channel](#) on page 35

To find the “Atlanta” trading partner

- 1 In the upper left side of the ePM screen, click **Select**.

A new window opens, prompting you to select a B2B host and trading partner.

- 2 Open the B2B Repository and **Berlin\_Env\_ebXML** and click myAS2Host1.
- 3 Click **Search**, and then click OK.

*Result:* In the explorer tree, under Berlin\_Env\_ebXML, trading partner **Atlanta** reappears.

## Parameters for the Delivery Channel

*Purpose:* To set the parameters governing message exchange with the “Atlanta” trading partner. You are configuring a trading partner for the Berlin environment, and so take the viewpoint of the Berlin B2B host: “ToPartner” means “to Atlanta”; “FromPartner” means “from Atlanta”.

### To configure the delivery channel parameters for trading partner “Atlanta”

- 1 In the explorer (lower left) side of the ePM screen, click **Atlant**.  
The canvas displays the trading partner’s general properties.
- 2 Click the **Components** tab.  
The trading partner’s delivery channel parameters are displayed. See Table 4.

**Table 5** Delivery Channel Parameters for Trading Partner “Berlin”

Binding Name	ebxml_delivery_channel1
ToPartner Transport Name	HTTP
FromPartner Transport Name	HTTP
Packager Name	

- 3 Click the binding name, **ebxml\_delivery\_channel1**.  
The delivery channel’s general properties are displayed.
- 4 Click the **ToPartnerTransport** tab and edit the All Purpose End Point so that it contains the correct URL for Berlin. For example:  
`http://localhost:19004/Deployment2_servlet/Inbound`  
If the logical host running the Berlin B2B host is not named localhost, or if its integration server uses a port other than 19004 for its web server, or if the name of the deployment profile is not Deployment1, make the appropriate changes. When you are done, click **Save**.
- 5 Click the **FromPartnerTransport** tab and edit the All Purpose End Point so that it contains the correct URL for Berlin. For example:  
`http://localhost:18004/Deployment1_servlet/Inbound`  
If the logical host running the Berlin B2B host is not named localhost, or if its integration server uses a port other than 18004 for its web server, or if the name of the deployment profile is not Deployment1, make the appropriate changes. When you are done, click **Save**.
- 6 Click the **ToPartnerPackaging** tab, make changes as needed, and then click **Save**.
- 7 Click the **FromPartnerPackaging** tab, make changes as needed, and then click **Save**.

*Result:* For trading “Atlanta”, the parameters for external delivery channel `ebxml_delivery_channel1` are now set correctly.

## Parameters for the Internal Delivery Channel

*Purpose:* To set the parameters governing Berlin’s internal message processing when handling messages received from the “Atlanta” TP or preparing messages to be sent to the “Atlanta” TP. You are configuring a trading partner for the Berlin environment, and so take the viewpoint of the Berlin B2B host: “ToInternal” means “to the Berlin internal (having been received from the Atlanta TP)”; “FromInternal” means “from the Berlin internal (and destined for sending to the Atlanta TP)”.

Your configuration of the internal delivery channel depends on your setup.

*Note:* Internal Delivery Channel bindings are always required for a project to run. Although it is possible to activate a trading partner whose messaging actions lack IDCs, the Channel Manager would have no instructions to read any input.

### 4.3.5 Activating the “Atlanta” Trading Partner

To activate the “Atlanta” trading partner

*Purpose:* To save all the configuration information to the Oracle database to make it available at run time.

*Before you begin:* Your eXchange 5.0.5 Oracle database for the corresponding B2B host must be running.

- 1 In the explorer (lower left) side of the ePM screen, click **Atlanta**.
- 2 In the bottom lower left of the canvas, click the **Activate** button.
- 3 In response to the confirmation prompt, click **Activate**.

The canvas displays a confirmation: **Trading Partner is successfully activated.**

*Result:* The **Atlanta** trading partner is entirely complete and ready to be run.

# Glossary of Acronyms

**AS2**

*Applicability Statement 2 (AS2)* is an Internet Draft security standard defined by the IETF (Internet Engineering Task Force), designed to allow business transactions to move securely over the Internet.

**BAD**

In eXchange, *Business Attribute Definitions (BADs)* define the metadata attributes of parameters used in business protocols such as X12, HIPAA, or EDIFACT.

**B2B**

*Business-to-business (B2B)* interactions are those that occur between business partners in the context of e-commerce.

**DCP**

In eXchange, a *Delivery Channel Profile (DCP)* is an association between a particular messaging service and a particular transport attributes definition (TAD). Also see "IDC" and "XDC".

**EAD**

In eXchange, *Enveloping Attribute Definitions (EADs)* define the metadata attributes of parameters used in enveloping protocols such as X12, HIPAA, or EDIFACT.

**ebXML**

A well-recognized e-business XML (extensible markup language; see "XML") whose implementation includes specifications for messaging, collaboration profiles, business processes, and metadata registry.

**ePM**

*eXchange Partner Manager (ePM)* is a Web-based GUI for defining and managing Trading Partner (TP) information.

**FTP**

*File Transport Protocol (FTP)* is a transport protocol for sending and receiving files. Specifications for FTP include RFCs 959, 1635, 2228, and 2577.

**HTTP**

*Hypertext Transport Protocol (HTTP)* is a transport protocol for transmitting information referenced in a URL of the form **http://<hostname>:<port>/.../....** Specifications for HTTP include RFCs 2068, 2616, 2617, 2660, and 3310.

### **ICAN**

See Beyond's *Integrated Composite Application Network (ICAN)* Suite includes eGate Integrator, eXchange Integrator, various eWay Intelligent Adapters, OTD Libraries, and Protocol Manager Composite Applications, as well as many other products.

### **IDC**

In eXchange, an *Internal Delivery Channel (IDC)* is an association between a particular transport attributes definition (TAD) and a direction (either Sender or Receiver). Compare with "XDC".

### **MAD**

In eXchange, *Messaging Attribute Definitions (MADs)* define the metadata attributes of parameters used in messaging protocols such as AS2 or ebXML.

### **MIME**

*Multipurpose Internet Mail Extensions (MIME)* extends the format of basic Internet mail to allow non-textual messages, multipart message bodies, and so forth. Specifications for MIME include RFCs 2045–2049.

### **OTD**

In ICAN, an *Object Type Definition (OTD)* contains the data structure and rules that define an object. OTDs are used in Java collaborations to transform data interface with external systems.

### **SME**

In ICAN, *Secure Messaging Exchange (SME)* uses advanced cryptographic techniques to ensure security, verifiability, and nonrepudiation of messages exchanged electronically.

### **S/MIME**

*Secure/Multipurpose Internet Mail Extensions (S/MIME)* provides a consistent way to send and receive secure MIME data, using digital signatures for authentication, message integrity and non-repudiation and encryption for privacy and data security. Specifications for S/MIME version 2 include RFCs 2311–2315.

### **SMTP**

*Simple Mail Transfer Protocol (SMTP)* is a transport protocol for transmitting e-mail messages between servers or from client to server. Specifications for SMTP include RFCs 1651, 2821, and 3461.

### **TAD**

In eXchange, *Transport Attribute Definitions (TADs)* define the metadata attributes of parameters used in transport protocols such as FTP or HTTP.

### **TP, TPP**

In eXchange, a *Trading Partner (TP)* has one or more *Trading Partner Profiles (TPPs)* that contain information identifying the values of messaging, enveloping, and/or transport parameters to be used for sending and receiving B2B information.

**URL**

A *Uniform Resource Locator* (URL) is a string that identifies information, such as a particular piece of information shared by a particular host.

**XDC**

In eXchange, an *External Delivery Channel* (XDC) is an association between three items:

- (1) either a messaging service (passthrough) or a business service (dialog);
- (2) a transport attributes definition (TAD) for the ToPartner (Sender) direction; and
- (3) a TAD for the FromPartner (Receiver) direction.

**XML**

An *Extensible Markup Language* (XML) is a language whose syntax obeys an official schema, called “the XML schema”, but whose semantics (“vocabulary”) are open.



# Index

## C

conventions, document 8

## D

database for eInsight engine  
    supported levels 15  
database for eXchange  
    supported levels 15  
document conventions 8

## E

ebXML sample implementation 22–37  
eInsight engine  
    supported databases 15  
encryption .jar files  
    updating 20  
    where to download 20  
eXchange database  
    system requirements 15  
eXchange support  
    for Oracle databases 15  
    for platforms 14

## L

local\_policy.jar, updating 20

## O

operating systems supported by eXchange 14  
Oracle database for eXchange  
    supported levels 15

## P

platforms supported by eXchange 14  
policy .jar files  
    updating 20  
    where to download 20

## S

sample implementations  
    ebXML 22–37  
Screenshots 8  
Secure Messaging Extension (SME)  
    updating policy .jar files for 20  
supporting documents 9

## U

US\_export\_policy.jar  
    updating 20  
    where to download 20

## X

xss4j.jar, required for ebXML security 21