Sun B2B Suite ebXML Protocol Manager User's Guide

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–2131–10
December 2007

071206@19028

Contents

Preface

The ebXML Protocol Manager product integrates with SeeBeyond eGate™ Integrator and SeeBeyond eXchange Integrator to enable you to design Java CAPS Projects that process and validate ebXML messages. eXchange provides an open framework to support standard B2B business protocols and communication protocols, such as ebXML.

ebXML stands for Electronic Business Extensible Markup Language, a modular set of specifications for standardizing XML to enable trade between organizations regardless of size. ebXML specifications give businesses a standard method they can use to exchange XML-based messages, carry on trading relationships, communicate data, and define and register eXchange B2B Protocols.

This guide explains how to install, configure, deploy, and use SeeBeyond® Integrated Composite Application Network Suite™ (ICAN) ebXML Protocol Manager Composite Application for eXchange™ Integrator.

This guide also describes and explains how to install and use ebXML Protocol Manager with eGate and eXchange, to function within the Java CAPS products. Additional detailed information, such as detailed steps required to create sample integration Projects are not included in this guide.

This Preface provides a brief introduction to the purpose, scope, and organization of the document plus additional reference information.

Who Should Use This Book

This guide is intended for computer users who have the ability and responsibility of setting up and maintaining a fully functioning Java CAPS system. These persons must also understand any operating systems on which the current Java CAPS is installed, for example Windows or Solaris UNIX, and must be thoroughly familiar with Windows-style user interface operations.

Before You Read This Book

For more information on ebXML, including the Requirements Specifications, see the following Web site:

`http://www.ebxml.org`

For more information on the Extensible Markup Language (XML), see the following Web site:

`http://www.xml.com/`

Sun Microsystems, Inc. Web Site

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

`http://www.sun.com`

How This Book Is Organized

This document provides information about installing, configuring, and using ebXML Protocol Manager and includes the following chapters:

- Chapter 1, "Overview of ebXML Protocol Manager," gives an overview of ebXML Protocol Manager, ebXML, and eXchange.
- Chapter 2, "Installing ebXML Protocol Manager," explains installation procedures, before and after installation, as well as system requirements.
- Chapter 3, "Configuration," explains ePartner Manager (ePM) configuration steps necessary to allow ebXML Protocol Manager to operate with the ePM.
- Chapter 4, "Quick Start Guide," provides a brief overview of how to create, configure and run an ebXML Protocol Manager Project.
- Chapter 5, "Implementation Scenario," explains in detail, the sample ebXML Protocol Manager Project provided with the installation CD-ROM.

Related books

Use the following related SeeBeyond guides as a reference for additional information in using ebXML Protocol Manager, if needed:

- *Sun Java CAPS Installation Guide*
- *Sun Java CAPS Deployment Guide*

- *eGate Integrator User's Guide*
- *eGate Integrator System Administration Guide*
- *eGate Integrator JMS Reference Guide*
- *Oracle eWay Intelligent Adapter User's Guide*
- *HTTP(S) eWay Intelligent Adapter User's Guide*
- *Sun B2B Suite eXchange Integrator User's Guide*
- *Sun B2B Suite eXchange Integrator Protocol Designer's Guide*
- *Secure Messaging Extension User's Guide*
- **Readme.txt** file for ebXML Protocol Manager

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX® system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell	`machine_name%`
C shell for superuser	`machine_name#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell for superuser	`#`

◆ ◆ ◆ CHAPTER 1

Overview of ebXML Protocol Manager

This chapter provides a general overview of ebXML Protocol Manager and its place in the Java CAPS, including system descriptions, ebXML information, general operation, basic features, and how it operates with eXchange.

What's in This Chapter

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

---

ebXML Protocol Manager: Introduction

ebXML Protocol Manager works primarily with eGate and eXchange. You can use ebXML Protocol Manager to design eGate Projects to process and validate messages that employ the ebXML delivery protocol.

Note – For more information on how eXchange operates with eGate, see the **eXchange Integrator User's Guide** .

---

ebXML Protocol Manager is designed to work with the eXchange framework to expose all of its related Project components. This feature allows for you to expand and customize your Projects and provides end-to-end visibility of the business logic implemented by the current Project.

ebXML Protocol Manager has the following primary characteristics:

- Uses a messaging service (also called business service) that is, a sequence of events incorporating the rules set by the protocol specifications
- Uses information in the message itself and in the eXchange Trading Partner profile to prepare messages according to ebXML standard
- Works with eXchange common services to prepare and deliver messages, employing the following features:
  - Error handling
  - Message tracking
  - Trading partner profile database lookup
  - PKI security services, such as encryption and signature creation and verification

About ebXML

Electronic Data Interchange (EDI) has given companies the ability to eliminate paper documents, reduce costs, and improve efficiency by exchanging business information with Trading Partners in electronic form. However, in the last few years, the Extensible Markup Language (XML) has rapidly become a first choice for defining data interchange formats for new eBusiness applications on the Internet.

EDI implementations represent substantial investments in encoding B2B Protocols (B2B Protocols). Companies with a large stake in EDI integration do not want to abandon EDI, for obvious cost reasons.

XML does enable more open, flexible business transactions than EDI, and XML might also allow more flexible and innovative business models. But there are still challenges in standardizing the semantics of message design and creating uniform B2B Protocol requirements. These problems are independent of the syntax in which messages are encoded.

ebXML specifications provide a framework in which EDI's investments in business processes (B2B Protocols in ebXML Protocol Manager) can be preserved in an architecture that also maintains and extends XML's technical capabilities.

For more information on ebXML, including the Requirements Specifications, see the following Web site:

`http://www.ebxml.org`

For more information on the Extensible Markup Language (XML), see the following Web site:

`http://www.xml.com/`

ebXML Overview

This model provides an example of the operations that may be required to configure and deploy ebXML applications and related components. These components can be implemented after initial system deployment, as needed.

Of course, there are more ebXML specifications than are shown in this simplified diagram. This model only provides a basic introduction to essential ebXML concepts. The numbers in the diagram call out steps in the process described under the next section.

Description of Model

The model operates as follows:

1. Company A has become aware of an ebXML registry accessible on the Internet.
2. Company A, after reviewing the contents of ebXML registry, decides to build and deploy its own ebXML-compliant application.

   Custom software development is not a necessary prerequisite for ebXML participation. ebXML-compliant applications and components that provide necessary solutions are commercially available and easily found.
3. Company A then submits its own business profile information (including implementation details and reference links) to ebXML registry.

   The business profile submitted to ebXML registry describes the company's ebXML capabilities and constraints, as well as its supported business scenarios. These business scenarios are XML versions of the B2B Protocols and associated information bundles (for example, a sales tax calculation) in which the company is able to engage. After receiving verification that the format and usage of a business scenario is correct, ebXML registry sends an acknowledgment to Company A.
4. After additional communication is established, Company B accesses, in ebXML registry, the business scenarios supported by Company A.

   If they decide they want to, Company B sends a request to Company A stating they want to engage in a business scenario using ebXML. Company B acquires the necessary ebXML-compliant applications in the same way as Company A.
5. Before starting the scenario, Company B submits a proposed business arrangement directly to Company A's ebXML-compliant software interface. The proposed business arrangement outlines the mutually agreed-upon business scenarios and specific agreements.

   The business arrangement also contains information pertaining to the messaging requirements for transactions to take place, contingency plans, and security-related requirements. Company A then accepts the business agreement.
6. Company A and B are now ready to engage in eBusiness transactions using ebXML.

Advantages of ebXML

As you can tell from this simple model, ebXML specifications provide the following advantages:

- Standardized way of describing B2B Protocols and their associated information models
- Registration and storing B2B Protocol and Information Meta Models so they can later be shared and reused
- Discovery and sharing of information about each participant including:
  - Supported B2B Protocols
  - Business service interfaces offered in support of each B2B Protocol
  - Business messages exchanged between their respective business service interfaces
  - Technical configuration of any supported transport, security, and encoding protocols

  Way to register the previously described information for later access and retrieval
- Way to describe the execution of mutually agreed-upon business arrangements derived from information provided by each participant from the discovery of information; these arrangements are called Collaboration Protocol Agreements (CPAs)
- Standardized business messaging service framework enabling interoperable, secure, and reliable exchanges of messages between Trading Partners
- Reliable way for configuring the respective messaging services to engage in agreed-upon B2B Protocols, in accordance with constraints defined in any predefined business arrangement

Design Conventions for ebXML Specifications

To ensure consistent capitalization and naming conventions across all ebXML specifications, use the Upper Camel Case (UCC) and Lower Camel Case (LCC) capitalization styles.

These styles employ the following conventions:

- UCC style capitalizes the first character of each word and compounds the name.
- LCC style capitalizes the first character of each word except the first word.

When you are producing ebXML documents that follow DTD, XML Schema, and XML instance conventions, use the following rules:

- Element names are in the UCC convention, for example:

  <UpperCamelCaseElement/>
- Attribute names are in the LCC convention, for example:

  <UpperCamelCaseElement lowerCamelCaseAttribute="Item"/>

When you use UML and Object Constrained Language (OCL) to specify ebXML naming capitalization, observe the following rules:

- For Class, Interface, Association, Package, State, Use Case, and Actor names, use the UCC convention, for example, **ClassificationNode**, **Active**, or **InsertOrder**.
- For Attribute, Operation, Role, Stereotype, Instance, Event, and Action names, use the LCC convention, for example, **name**, **notifySender**, or **orderArrived**.

General rules for all names are:

- Avoid acronyms, but in cases where you have to use them, keep the capitalization, for example, **XMLSignature**.
- Do not use underscores (_), periods (.), or dashes (-). For example, instead of using **title.document**, **stock_sale_6,** or **dollar-value**, use **TitleDocument**, **stockSale6**, and **DollarValue**.

About eXchange Integrator

eXchange Integrator provides an open Business Protocol framework to support standard EDI and B2B protocols, as well as packaging protocols. The eXchange product supports existing standard protocols, using an extensive set of prebuilt eInsight Business Processes (BPs). 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 eInsight Business Rules can be added and tailored to address the particular needs of providing eBusiness solutions. The tight integration with the rest of Java CAPS provides validation, logging, and reporting capabilities. Because each logical step within any Business Rule is accessible anywhere along the entire eInsight BP, the design tools provide complete end-to-end visibility.

Note – For a complete explanation of eXchange and eInsight, as well as their operation, see the **eXchange Integrator User's Guide** and **eInsight Business Process Manager User's Guide** .

Understanding BPs

An eInsight BP is a collection of actions or operations that take place in your company, revolving around a specific business practice. These processes can involve a variety of participants and may include internal and external computer systems or employees.

In eXchange, you create a graphical representation of a business process called a BP model. When you are using the sample for a PM's implementation scenario, the system uses the BPs necessary for scenario's operation. The BPs specific to the sample scenario provided with the product have already been created for this scenario.

Trading Partner Overview

The architecture of eXchange centers around the concept of sending and receiving messages relative to one or more TPs. Each TP that you import or create and then configure corresponds to one of your business trading partners.

These TPs contain configurable transaction profiles for each individual TP relationship. You can configure TPs with Transaction Profiles and Schedules, within ePM, for use by run-time components.

Each Transaction Profile specifies which one or more BPs to use for the current transaction, where and how to receive inbound messages, how to configure and secure messages in their channels, and how and where to deliver outbound messages.

Process Overview

Using eXchange to create a business solution consists of the following phases:

- Design phase within Enterprise Designer
- Configuration/design phase within ePM
- Run-time phase

eXchange Partner Manager

ePM is a feature of eXchange you can use as a tool that allows you to configure eXchange for use with any of your PMs. You must set specific parameter values within ePM to ensure the correct operation of your Projects, for each protocol. This guide explains how to use and configure ePM, to set parameters relevant to this guide's PM.

For more information on how to use ePM, see the *eXchange Integrator User's Guide*.

B2B Suite, eXchange, and the Java CAPS

eXchange is one of the products that make up the B2B Suite within the Java CAPS. The B2B Suite products, including eXchange, provide a Web-based TP configuration and management solution for automating and securely managing business partner relationships. The products also facilitate real-time interaction between the enterprise and its TPs, suppliers, and customers.

As a part of the Java CAPS, the B2B Suite provides the following benefits and features:

- B2B services via eXchange
- Protocol managing, specifically by the PMs
- Protocol formats contained in the OTD Libraries

- Trading partner management facility, that is, the ePM interface
- Archiving tool, the Message Tracking feature

The B2B Suite is tightly integrated with the Java CAPS and runs as a group of components within the Java CAPS environment.

CHAPTER 2

Installing ebXML Protocol Manager

This chapter explains how to install ebXML Protocol Manager, as well as supported operating systems and system requirements. The chapter also includes necessary post-installation procedures.

What's in This Chapter

Supported Operating Systems

ebXML Protocol Manager for eXchange is available for the following operating systems:

- Microsoft Windows Server 2003, Windows XP, and Windows 2000
- Sun Solaris 8 and 9
- HP Tru64 V5.1A and V5.1B
- HP-UX 11.0 and 11i (PA-RISC)
- IBM AIX 5.1L and 5.2
- Red Hat Linux 8 (Intel x86) and Linux Advanced Server 2.1 (Intel x86)

For more details on exact supported operating systems, see the **Readme.txt** file for eXchange Integrator.

System Requirements

To use ebXML Protocol Manager, you need:

- eGate Logical Host
- TCP/IP network connection
- SeeBeyond applications as listed under "Before Installing ebXML Protocol Manager" on page 24

Logical Host Requirements

ebXML Protocol Manager must have its configuration properties set and be administered using Enterprise Designer. For complete information on Enterprise Designer system requirements, see the *Sun Java CAPS Installation Guide*.

Supported External Applications

ebXML Protocol Manager, in conjunction with eXchange, requires external database support, as described in this section.

Database for eXchange Partner Management and Message Tracking

The eXchange database is required by ebXML Protocol Manager. This database provides a run-time persistent store for Trading Partner (TP) management and message tracking. For eXchange, the following databases are supported:

- Oracle version 8.1.7
- Oracle version 9.01
- Oracle version 9.2
- Oracle version 10.1g

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 feature provides for recovery and also enables some monitoring and reporting capabilities in Enterprise Manager. The eInsight engine supports the following databases:

- Oracle version 8.1.7, 9.0.1, 9.2, and 10.1g
- Sybase 12.5
- Microsoft SQL Server 2000

- IBM DB2 Universal Database version 8.1

Before You Install

Open and review the **Readme.txt** file for Java CAPS to gain current information you may need, for example for eGate or eInsight, before installing ebXML Protocol Manager. You can find this file in the root directory of the Java CAPS installation's Repository CD-ROM.

Also, ebXML Protocol Manager has its own **Readme.txt** file that contains additional information specific to this application, including required ESRs.

Note – See the **Sun Java CAPS Installation Guide** for details on how to obtain the **Readme.txt** , ebXML Protocol Manager sample scenario, and documentation files.

Configuring eGate Projects for Large Messages

If an eGate Project uses SeeBeyond JMS IQ Manager and is estimated to process messages or transactions over 8 MB for Windows, or 16 MB for UNIX, you must increase the **Segment Size** property of JMS IQ Manager as explained in the *eGate Integrator JMS Reference Guide*.

Installing ebXML Protocol Manager Product Files

During the Java CAPS installation operation, use Enterprise Manager, a Web-based application, to select and upload ebXML Protocol Manager and add-on application **.sar** files from the Java CAPS installation CD-ROM to the Repository.

When the Repository is running on a UNIX operating system, you must still install ebXML Protocol Manager using Enterprise Manager and Microsoft Internet Explorer on a Windows computer connected to the Repository server.

Follow the general instructions for installing the Java CAPS, which you can find in the *Sun Java CAPS Installation Guide*. You must begin by installing eGate.

eGate Installation

The eGate (**eGate.sar**) installation process includes the following operations:

- Installing the eGate Repository
- Uploading products to the Repository
- Downloading the necessary components (including the eGate Enterprise Designer and the Logical Host)

- Viewing the product information home pages

Before Installing ebXML Protocol Manager

Before you install ebXML Protocol Manager ensure that:

- An eGate Repository server must be running on the machine where you are uploading the product files.

Note – For the exact versions of SeeBeyond eWays and other products you need to use, see the **Readme.txt** file (and/or references within that file) provided on the installation CD-ROM that accompanies ebXML Protocol Manager.

- In addition to **eGate.sar**, you upload the Java CAPS .sar files listed in the **Readme.txt** file.

Installing ebXML Protocol Manager

This section explains how to install ebXML Protocol Manager, that is, how to upload the ebXML_Manager.sar file to the Repository.

▼ To install ebXML Protocol Manager

1. **Make sure the eGate.sar file has been uploaded to the Repository using Enterprise Manager.**

2. **Make sure you have selected and uploaded, to the Repository, the .sar files listed under "eGate Installation" on page 23 and "Before Installing ebXML Protocol Manager" on page 24.**

3. **Upload the ebXML_Manager.sar file. This operation installs ebXML Protocol Manager.**

4. **Upload ebXML_ManagerDocs.sar (if you have not done so already). This file contains:**

5. **ebXML_Manager_Guide.pdf file with the product's documentation**

6. **ebXML_Manager_Sample.zip file containing the sample scenario Projects and related files (see "Run-time Steps" on page 69 for details on this file)**

7. **Readme.txt file**

   To obtain these files, follow the instructions provided by the *Sun Java CAPS Installation Guide* and Enterprise Manager.

After You Install

Once ebXML Protocol Manager is installed and configured, it must then operate in conjunction with an eGate Project before it can perform its intended functions. You must create these Projects specifically for ebXML Protocol Manager.

See the *eXchange Integrator User's Guide* and *eGate Integrator User's Guide* for detailed information on incorporating these types of Projects into eGate. See "Run-time Steps" on page 69 for a sample business scenario with Projects already created, using eXchange and ebXML Protocol Manager.

The rest of this section provides important information on steps you must take to ensure that ebXML Protocol Manager operates correctly with eGate and eXchange.

Database Scripts

eXchange allows you to collect database information and persist data about your TP profiles. eXchange provides database scripts to create and upgrade this database feature for eXchange. For more information, see the eXchange Integrator User's Guide.

You can also track message delivery histories, using the eXchange Message Tracking feature. See "Using Message Tracking" on page 100 for more information.

Additional Policy .jar Files Required To Run SME

To correctly use ebXML Protocol Manager, you must download and apply additional policy .jar files. This operation provides reliable cryptographic features that use libraries supplied with SeeBeyond Secure Messaging Extension (SME), contained in the **SMEWebServices.sar** installation file.

The type of .jar files required depends on the version of Java Virtual Machine (JVM) you are using. Refer to your JVM vendor for exact details on the specific policy .jar file requirements.

Use Table 2–1 to determine which JRE is included in your eGate Logical Host.

TABLE 2–1 JRE Versions Listed by Operating System

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

▼ To download the required policy.jar files

1. **Scroll to the bottom of the appropriate Web page listed in Table 2–1 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 as follows:**
   - For JRE 1.4.2, follow the link **Download jce_policy-1_4_2.zip**.
   - 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. **For each of your Logical Hosts, using the extracted .jar files, replace the existing versions of these files in the following location:**

   ```
   <Logical Host>\jre\lib\security\
   ```

6. **If you are running a repository on AIX, also replace, using the extracted .jar files, the existing versions of these files in the following location:**

   ```
   <AIX Repository>/jre/1.4.x/lib/security/
   ```

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

Additional File for ebXML Cryptographic Features

For ebXML, you must also download and apply an additional .jar file, **xss4j.jar**, for XML security. This file 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\
```

Note – The directories `stcis\lib\` are not created by the initial installation. They are created automatically when you first run the Logical Host. If you are copying this file to its correct location, and you have not yet run the Logical Host, you must first create the folders manually.

Configuring eGate Projects for Large Messages

For eGate Projects that are expected to process large messages and transactions, or large numbers of transactions, you can configure the Project in the following ways to increase throughput and improve performance:

- Increasing the heap size for the Logical Host
- Increasing the maximum concurrent instances and run-time thread pool size properties for the eInsight engine
- Increasing the segment size for SeeBeyond JMS IQ Manager
- Increasing the number of concurrent processes to be handled by the Oracle database

Increasing Logical Host Heap Size

To avoid memory errors on the Logical Host, increase the heap size for the Logical Host to at least 1024 MB as explained in this section.

▼ To increase the Logical Host heap size

1. **On the Environment Explorer tab of the Enterprise Designer, select and expand the Logical Host folder.**

2. **Right-click the Logical Host, and click Properties.**

   The **Properties** dialog box appears. See "Increasing Logical Host Heap Size" on page 27.

3. **Under Configuration > Logical Host Configuration, enter the heap size 1024 in the Heap Size text box.**

4. **When you are finished, click OK.**

Improving Performance

The eInsight engine has the following properties that affect performance:

- **Max Concurrent Instances**
- **Max Runtime Thread Pool Size**

The **Max Concurrent Instances** property indicates the amount of inbound messages that each BP processes simultaneously. For example, if you set this property to 10, the BP processes up to ten inbound messages at the same time. When the eleventh message is received, the BP does not process the message until the processing of one of the ten messages has been completed.

The **Max Runtime Thread Pool Size** property defines how many threads that BPs can invoke simultaneously.

These settings constitute a trade-off between performance and memory. Depending on how much memory your system has, you can specify these settings accordingly. There are no recommended settings for these properties because they depend on the specific memory capacity of your system. If your Project processes too slowly or messages are lost, increase these settings as your system allows.

For instructions on configuring run-time properties for the eInsight engine, see the *eXchange Integrator User's Guide*. For additional information about these properties, see the *eInsight Business Process Manager User's Guide*.

▼ To Improve Performance

1. **On the Environment Explorer tab of the Enterprise Designer, select and expand the Logical Host folder.**
2. **Right-click the Integration Server, and click Properties.**

   The **Properties** dialog box appears (see "Improving Performance" on page 27).
3. **Expand the IS Configuration, Sections, and eInsight Engine Configuration folders.**
4. **Increase the value of the setting for the Max Concurrent Instances property, as necessary. See "Improving Performance" on page 27.**
5. **Increase the value of the setting for the Max Runtime Thread Pool Size property, as necessary. See "Improving Performance" on page 27.**
6. **When you are finished, click OK.**

Increasing the JMS IQ Manager Segment Size Property

If a Java CAPS Project that uses the SeeBeyond JMS IQ Manager needs to process messages or transactions over 8 MB for Windows or 16 MB UNIX, increase the **Segment size** property of the JMS IQ Manager, as explained in the *eGate Integrator JMS Reference Guide*.

Increasing the Oracle Number of Processes

To increase Oracle throughput performance, increase the Oracle number of processes to handle simultaneous database process requests, such as 500. For specific information, refer to your Oracle documentation.

CHAPTER 3

Configuration

This chapter explains how to configure and design ePM and Enterprise Designer for use with ebXML Protocol Manager and ebXML.

What's in This Chapter

Configuration for ebXML Protocol Manager: Overview

This chapter explains the configuration parameters required in ePM for ebXML Protocol Manager Projects and their operation with ebXML. You can configure these parameter values for ebXML Protocol Manager using the eXchange ePM interface.

Chapter 5, "Implementation Scenario," explains a sample business scenario with Projects containing specific parameters already set and values already configured correctly in ePM. The ebXML Protocol Manager CD-ROM contains the files that make up this sample scenario.

You must use ePM to set configuration parameters for your ebXML Protocol Manager Projects' Trading Partners (TPs). Entering the correct values in ePM allows these TPs to operate correctly with eXchange, ebXML Protocol Manager, and ebXML.

Accessing and Using ePM

The eXchange ePM tool allows you to create and enable TPs for your ebXML Protocol Manager eXchange Projects.

Using ePM: Overview

In eXchange, each TP contains information identifying the values for the messaging, enveloping, and/or transport parameters to be used for sending and receiving B2B information.

Parameter Types

ePM contains the following types of parameters:

- General eXchange parameters common to all protocol managers
- ebXML Protocol Manager-specific parameters present only for this particular application

See "Configuring TPs" on page 34 for a complete explanation of parameter types.

Creating and Enabling TPs

To create and enable TPs you can do the following operations using ePM:

- Create bindings:
  - For external delivery channels (XDCs); required.
  - For internal delivery channels (IDCs); optional.

  Create one or more TPs.
- Configure your TPs.
- Activate your TPs.

For complete information on how to do these operations in ePM, see the *eXchange Integrator User's Guide*. Also, "Run-time Steps" on page 69 contains a sample scenario with specific examples of TPs set up in ePM.

The following sections provide basic instructions on how to use ePM:

- "Importing and Configuring TPs in ePM" on page 88
- "TP Activation" on page 97

The rest of this section provides a general summary of how to access ePM, as well as use it with TPs for ebXML Protocol Manager Projects.

Before Starting

Before you can access ePM for a Project, make sure that, using Enterprise Designer, you have created and activated the eXchange Project Deployment Profiles, including any for the necessary B2B hosts.

Accessing ePM

After you have ensured these conditions have been met, you can use your Web browser to access ePM. By default, you can access ePM using a predefined URL, for example:

```
http://localhost:12000/epm
```

If access to ePM has been set up using a different URL, consult your system administrator for this URL.

Using Bindings

A binding is an association of metadata parameters with a particular set of values you define, using ePM. This metadata can be for either XDCs or IDCs.

Creating Bindings for XDCs

The top of the Delivery Channels (XDCs) subtab in ePM lists all bindings currently defined for the current TP's XDCs.

In the case of XDCs, the transport parameters are those specified in a transport attribute definition (TAD). The packaging (messaging) parameters are those specified in a message attribute definition (MAD).

▼ To create bindings for XDCs

1. **Access an XDC and create a binding to it.**

2. **Import a signature certificate and encryption key, if necessary.**

Note – Using certificates and encryption keys in ebXML Protocol Manager is done in the same way for all eXchange protocol managers. For complete information, see the **eXchange Integrator User's Guide** .

Creating Bindings for IDCs

The top of the Internal Delivery Channels subtab in ePM lists all bindings currently defined for the TP's IDCs.

In the case of IDCs, the transport parameters are those specified in a TAD. The use of IDCs is optional.

To create the bindings for IDCs you must:

- Access an IDC and create a binding to it.

Optionally, you can also:

- Add a new binding for your IDC.

Note – To correctly use ebXML Protocol Manager and eXchange, you must understand the concepts and features explained in this section. For a quick reference that provides definitions of features, such as TADs, MADs, IDCs, and XDCs, see the Glossary.

Creating TPs and Messaging Service Bindings

To use ebXML Protocol Manager, you must create and activate one or more TPs. You must also create a new Messaging Service binding based on one of the Messaging Services shown on the ePM window.

You can also find the Messaging Service as the bottom leaf of the **ePM Explorer** pane's tree (on the left), under the messaging attributes definition, standard or custom, with which it is associated.

When you are finished, all the new elements you have created appear in this tree, on the **ePM Explorer** pane.

For details on these components and operations, see the *eXchange Integrator User's Guide*. Also, see the sample scenario in "Run-time Steps" on page 69.

Configuring TPs

After you have set up and activated your TPs, you must configure them for ebXML and your current system. Each set of specific configurations you assign to a TP make up its profile. TP configuration parameters include the following settings:

- **Properties**: Under this tab, you supply values for the necessary eXchange parameters not ebXML-specific, but needed by eXchange. These parameters are basic eXchange properties and enable the essential eXchange operation of its Projects.
- Under other tabs, you find parameters that allow you to supply values for the following types of eXchange and ebXML Protocol Manager-specific configuration settings that define:
  - **Delivery Channels**: Under this subtab, you supply XDC values for the **ToPartner** and **FromPartner** transport attributes, as well as the **ToPartner** and **FromPartner** packaging attributes.

- **Internal Delivery Channels**: Under this subtab, you supply values for the **ToPartner** and **FromPartner** transport attributes for each IDC.
- **Messaging Actions**: Under this tab, you supply values for each of the messaging actions (inbound and outbound).

Note – The ePM user interface refers to XDCs under tabs and headings named **Delivery Channels** .

"Importing and Configuring TPs in ePM" on page 88 explains a sample scenario with specific examples of how to configure parameters for an individual TP using ePM.

Note – After you have configured each TP, you must activate it then check for its successful activation. See "TP Activation" on page 97 for details.

The rest of this chapter explains in detail how to configure each ePM parameter setting for ebXML Protocol Manager's TPs, summarized under the types shown in the previous list. For more information, including information on the various tabs/subtabs, what they contain, and how to navigate through them, see the *eXchange Integrator User's Guide*.

Configuring Basic eXchange Properties

This section describes the ebXML Protocol Manager's properties configuration used to set basic eXchange parameters. For an example of how these settings appear in the ePM user interface.

These configuration parameters define basic eXchange settings. See the *eXchange Integrator User's Guide* for information on these parameters.

Configuring Bindings for Delivery Channels

This section describes how to configure ebXML Protocol Manager's bindings for XDCs. You set up and define these bindings by supplying values for the **ToPartner** transport attributes, the **FromPartner** transport attributes, and the combined **To/FromPartner** packaging attributes.

This section explains in detail how to configure ePM's ebXML-specific parameters. You can find the **Delivery Channels** (XDCs) subtab example shown under the **Components** tab.

Note – The ePM user interface refers to XDCs under tabs called **Delivery Channels** .

These configuration parameters define settings that allow ebXML Protocol Manager to deliver messages to TPs. Their categories are:

- "General" on page 36
- "ToPartner Transport" on page 36
- "FromPartner Transport" on page 38
- "ToPartner Packaging" on page 38
- "FromPartner Packaging" on page 44

General

These parameters allow you to define general eXchange settings. See the *eXchange Integrator User's Guide* for information on these parameters.

ToPartner Transport

These parameters allow you to supply information needed to transport data to the TP. Find these parameters under the **ToPartner Transport** subtab.

Note – For a complete description of how to enter these parameters for both **ToPartner Transport** and **FromPartner Transport** parameters, see"XDC Parameters" on page 91.

All Purpose End Point

Description

An endpoint specifies the logical address of where messages can be received. The all- purpose end point is used to designate the receipt address of all ebXML messages.

Values

A valid URL, according to the syntax given under "XDC Parameters" on page 91; required.

Default

None

Use Synchronous Channel

Description

Specifies whether the communication between the message service handler (MSH) servers is synchronous or asynchronous. When set to **true**, the MSH response message is returned on the same HTTP connection as the inbound request, with an appropriate HTTP response code.

This parameter is not required and is not used in the current release of ebXML Protocol Manager. However, it is provided for compatibility with previous releases.

Ext Attribute SyncReplyMode replaces this parameter in the current release, since ebXML requires a string (for example, **None** or **mshSignalsOnly**) and not a Boolean value.

Values

Select **true, false,** or **None**; not required.

Default

false

Track for Auditing

Description

Specifies whether the message is tracked using Enterprise Manager's Message Tracker. Choosing **true** enables the Message Tracker.

Values

Select **true, false,** or **None**; not required.

Default

false

Destination URL

Description

Allows you to enter the destination URL for the current TP.

Values

A valid URL according to the syntax given under "XDC Parameters" on page 91; required.

Default

None

Header Names/Values

Description

Allows you to enter any needed message headers and/or values that may be required by the current TP to reach its destination.

Values

Valid message header names and/or values; not required.

Default

None

FromPartner Transport

These parameters allow you to supply information needed to transport data from the TP. See "ToPartner Transport" on page 36 for an explanation of these parameters. Find these parameters under the **FromPartner Transport** tab.

ToPartner Packaging

These parameters allow you to define the packaging for data being transported to the TP. Find these parameters under the **ToPartner Packaging** tab. .

The initial set of parameters before **SOAP Media Type** are eXchange parameters common to all protocol managers. See the *eXchange Integrator User's Guide* for information on these parameters.

SOAP Media Type

Description

Specifies the Simple Object Access Protocol (SOAP) media type. In accordance with the SOAP specification, the multipurpose Internet mail extension (MIME) media type of the SOAP message has the value **text/xml**.

SOAP media type means the content type for SOAP with **Attach**, which is multipart and related.

Values

Multipart/Related; required.

Default

Multipart/Related

Digital Envelope Protocol

Description

Provides the sender's requirements for message encryption using the digital-envelope (DIGENV) method. Digital enveloping is a procedure in which the message is encrypted by symmetrical encryption (using a shared secret key), and the secret key is sent to the message recipient encrypted, along with the recipient's public key.

Values

Select **http://www.w3.org/2001/04/xmlenc#**, **SMIME2.0**, or **SMIME3.0**; not required.

Default

http://www.w3.org/2001/04/xmlenc#

Non-Repudiation Protocol

Description

Identifies the technology used to digitally sign a message. This parameter has a single implied version attribute whose value is a string that identifies the version of the specified technology.

Value

Select **XMLDSIG**; not required.

Default

XMLDSIG

Signature Algorithm

Description

Identifies the algorithm used to compute the value of the digital signature.

Values

Select **http://www.w3.org/2000/09/xmldsig#dsa-sha1**, **http://www.w3.org/2000/09/xmldsig#rsa-sha1**, or **http://www.w3.org/2000/09/xmldsig#hmac-sha1**; not required.

Default

http://www.w3.org/2000/09/xmldsig#dsa-sha1

Digest Method/HashFunction

Description

Identifies the algorithm used to compute the digest of the message being signed.

Value

Select **http://www.w3.org/2000/09/xmldsig#sha1**; not required.

Default

http://www.w3.org/2000/09/xmldsig#sha1

Canonicalization Method

Description

Identifies the canonicalization method applied to the data to be signed.

Values

Select **http://www.w3.org/TR/2001/REC-xml-c14n-20010315** or **http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments**; not required.

Default

http://www.w3.org/TR/2001/REC-xml-c14n-20010315

Encryption Algorithm

Description

Identifies the encryption algorithm to be used.

Value

Select **DES3**; not required.

Default

DES3

Signature Type

Description

Specifies the XML digital signature type to be used (see `http://www.w3.org/TR/xmldsig-core/`).

Values

Select **Enveloped**, **Enveloping**, **Detached**, or **SoapEnveloped**; not required.

Default

SoapEnveloped

Ack Signature Required

Description

Designates how the signed attribute within the **AckRequested** element in the SOAP message header is set.

Values

Select **Always**, **Never**, or **perMessage**; not required.

Default

None

Note – When ebXML Protocol Manager sends an encrypted but unsigned message to a TP then gets a signed acknowledgment from the TP, the system cannot verify the signature and logs an error.

Ack Request is Per Message

Description

Allows you to designate whether the **AckRequested** element within the SOAP message header is operating.

The value **true** requests an acknowledgment for each message. Setting it to **false** disables this feature.

Values

true or **false**; not required.

Default

false

Additional Information

Requesting acknowledgments per message depends on the following configuration parameters in ePM:

- **Send Acknowledgments**: Provided in ePM by eXchange.
- **Ack Request is Per Message**: ebXML-specific.

If the **Ack Request is Per Message** value is **true**, regardless of what is specified in the **Send Acknowledgments** parameter, the system uses the eXchange setting provided at run-time for outbound messages, that is, the setting under EX_INTERNAL.

If the **Ack Request is Per Message** value is **false**, the system checks the value set under **Send Acknowledgments**. If **Send Acknowledgments** is **true**, the system checks for negative acknowledgments. If this value is **false**, the system ignores negative acknowledgments.

Duplicate Elimination is Per Message

Description

Allows you to designate whether the **DuplicateElimination** element within the SOAP message header is operating.

If you set this parameter to **true**, the system keeps each message from being duplicated. Setting it to **false** disables this feature.

Values

true or **false**; not required.

Default

false

Additional Information

Eliminating duplicates per message depends on the following configuration parameters in ePM:

- **Check For Duplicates**: Provided in ePM by eXchange.
- **Duplicate Elimination is Per Message**: ebXML-specific.

If the **Duplicate Elimination is Per Message** value is **true**, regardless of what is specified in the **Check For Duplicates** parameter, the system uses the eXchange setting provided at run-time for outbound messages, that is, the setting under EX_INTERNAL.

If the **Duplicate Elimination is Per Message** value is **false**, the system checks the value set under **Check For Duplicates**. If **Check For Duplicates** is **true**, the system checks for duplicates. If this value is **false**, the system ignores duplicates.

SOAP Actor

Description

Allows you to specify the current message's final destination. By using **toPartyMSH**, you make the receiving party the final destination.

Note – Values are listed in the drop-down menu.

Values

Select **urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH**; required.

Default

None

Sync Reply

Description

Specifies the mode of the reply to the current message.

Values

Select **None** or **mshSignalsOnly**; required.

Default

None

Language

Description

Specifies the language in which the description, errors, and so on, are sent out. Refer to the appropriate ebXML documentation on their Web site for details.

Values

A valid string supplied by the user; required.

Default

en-us (English, U.S.)

FromPartner Packaging

These parameters allow you to define the packaging for data being transported from the TP. Find these parameters under the **FromPartner Packaging** tab. .

The initial set of parameters before **Digital Envelope Protocol** are eXchange parameters common to all protocol managers. See the *eXchange Integrator User's Guide* for information on these parameters.

Digital Envelope Protocol

Description

Provides the sender's requirements for message encryption using the digital-envelope (DIGENV) method. Digital enveloping is a procedure in which the message is encrypted by symmetrical encryption (using a shared secret key), and the secret key is sent to the message recipient encrypted, along with the recipient's public key.

Values

Select **http://www.w3.org/2001/04/xmlenc#**, **SMIME2.0**, or **SMIME3.0**; not required.

Default

http://www.w3.org/2001/04/xmlenc#

Non-Repudiation Protocol

Description

Identifies the technology used to digitally sign a message. This parameter has a single implied version attribute whose value is a string that identifies the version of the specified technology.

Value

Select **XMLDSIG**; not required.

Default

XMLDSIG

Signature Type

Description

Specifies the XML digital signature type to be used (for more information, see `http://www.w3.org/TR/xmldsig-core/`).

Values

Select **Enveloped**, **Enveloping**, **Detached**, or **SoapEnveloped**; not required.

Default

SoapEnveloped

Ack Signature Required

Description

Designates how the signed attribute within the **AckRequested** element in the SOAP message header is set.

Values

Select **Always**, **Never**, or **Per Message**; not required.

Default

None

Note – When a signed acknowledgment is received from a TP and its associated outbound request message is not signed, ebXML Protocol Manager is unable to verify the signature and logs an error.

Ack Request is Per Message

Description

Allows you to designate whether the **AckRequested** element within the SOAP message header is operating.

If you set this parameter to **true**, the system requests an acknowledgment for each message. Setting it to **false** disables this feature (see "Additional Information" on page 42).

Values

true or **false**; not required.

Default

false

Duplicate Elimination is Per Message

Description

Allows you to designate whether the **DuplicateElimination** element within the SOAP message header is operating.

If you set this parameter to **true**, the system keeps each message from being duplicated. Setting it to **false** disables this feature (see "Additional Information" on page 43).

Values

true or false; not required.

Default

false

SOAP Actor

Description

Allows you to specify the current message's final destination. By using **toPartyMSH**, you make the receiving party the final destination.

Note – Values are listed in the drop-down menu.

Values

Select **urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH**; required.

Default

None

Sync Reply

Description

Specifies the mode of the reply to the current message.

Values

Select **None** or **mshSignalsOnly**; required.

Default

None

Language

Description

Specifies the language in which the description, errors, and so on, are sent out. Refer to the ebXML documentation on their Web site for details.

Values

A valid string supplied by the user; required.

Default

en-us (English, U.S.)

Configuring Certificates

No ebXML-specific parameter configuration for certificates is required. You must import them. See "Configuring Companies' Cryptographic Features" on page 85 for details.

Configuring Bindings for IDCs

This section describes how to configure ebXML Protocol Manager's bindings for IDCs. You set up and define these bindings by supplying values for the **ToPartner** and **FromPartner** transport attributes for each IDC used by the B2B host.

These configuration parameters are used by the B2B host, and their categories are:

- "Sender Transport" on page 48
- "Receiver Transport" on page 48

Sender Transport

The parameters under this tab allow you to define the sender transport attributes for individual IDC bindings used by the B2B host.

Note – See the **eXchange Integrator User's Guide** for an explanation of IDC configuration properties for sender and receiver transport. Under **Internal Delivery Channels**, select the **Transport Name** , as appropriate, for example **JMS** .

Receiver Transport

The parameters under this tab allow you to define the receiver transport attributes for individual IDC bindings used by the B2B host. Set these parameters in the same way as you do those for the sender transport channels. For details on how to set these parameters, see "Sender Transport" on page 48.

For an example of how the parameters appear in the ePM user interface.

Configuring Enveloping Channels

No ebXML-specific parameter configuration is required. For information on configuring enveloping channels, see the appropriate user's guide for the enveloping protocol you are using.

Configuring Messaging Actions

This section describes how to configure ebXML Protocol Manager's messaging actions under the **ePM Messaging Service Configuration**. These settings control messaging actions (inbound and outbound) for the ebXML MAD.

Note – Many of the explanations of these parameters refer to the ebXML Message Service Specification, version 2.0. To find this specification, see the following ebXML Web site:

`http://www.ebxml.org`

These configuration parameters define settings that allow ebXML Protocol Manager to set up, control, and order the sending and receiving of messages. These setting categories are:

- "Outbound" on page 49
- "Inbound" on page 62

This section explains in detail how to configure ePM's ebXML-specific parameters.

When you first click the **Messaging Actions** tab, each action appears collapsed as a separate line within a bar. The individual parameters for an action appear when you click the action's plus sign (+) and expand it.

Note – The individual Messaging Actions you see in ePM reflect those created for the current B2B host in Enterprise Designer. messaging actions in the sample scenario are specific to that scenario. The messaging actions you create depend on your system setup and configuration.

Outbound

These MAD parameters allow you to define messaging actions for outbound messages. For an example of how the parameters appear in the ePM user interface for ebXML Protocol Manager.

The initial set of parameters (before **Service Type**; see "Service Type" on page 50) are eXchange settings common to all protocol managers. See the *eXchange Integrator User's Guide* for information on how to set them, including how to set up messaging actions as either inbound or outbound.

Service Type

Description

Use this parameter when the parties sending and receiving the message know how to interpret the value of the Service element. The two parties may use the value of the **Type** attribute to assist the interpretation.

Values

A valid string supplied by the user; not required.

Default

None

Host Role in Transaction

Description

Associates the B2B host with a specific role in the business Collaboration, such as a "buyer."

Values

A valid string supplied by the user; not required.

Default

None

Partner Role in Transaction

Description

Associates a TP with a specific role in the business Collaboration, such as a "seller."

Values

A valid string supplied by the user; not required.

Default

None

HostPartyID Value

Description

The **PartyID** element provides an identifier for the B2B host. The value of the **PartyID** element is any string that provides a unique identifier. If this value is not a URN, the **HostPartyID** type *must* be provided.

Values

A valid string supplied by the user; required.

Default

None

HostPartyID Type

Description

The **PartyID** element provides an identifier for the B2B host. The type of the **PartyID** element provides a scope or namespace for the content of the **PartyID** element.

Values

A valid string supplied by the user; not required.

Default

None

PartnerPartyID Value

Description

The **PartyId** element provides an identifier for a TP. The value of the **PartyID** element is any string that provides a unique identifier. If this value is not a URN, the **PartnerPartyID** type *must* be provided.

Values

A valid string supplied by the user; required.

Default

None

PartnerPartyID Type

Description

The **PartyId** element provides an identifier for a TP. The type of the **PartyID** element provides a scope or namespace for the content of the **PartyID** element.

Values

A valid string supplied by the user; not required.

Default

None

Sequence Number

Description

Not yet implemented; the parameter is reserved for future use and is ignored in the current release.

Note – This parameter is part of a planned Message Sequencing feature and indicates the sequence with which a receiving MSH must process the message. The value entered is the start number for sequencing messages.

Values

A valid string supplied by the user; not required.

Default

None

Envelope ContentID

Description

Used to specify a unique, identifying label for each envelope, referred to as the content ID. This parameter is mandatory and must be a global universal identification (GUID).

For details, see the request for comments (RFC) document 2392, Section 2, on the following Web site:

`http://www.ietf.org/rfc/rfc2392.txt`

Values

A valid GUID identifying the content ID; required.

Default

Envelope

Envelope ContentLocation

Description

A URI designating the content-location of the MIME body part of an envelope object. Specify this parameter (URI) when the content/payload is not a package along with the message, but instead is referenced using the URI location.

Values

A valid string supplied by the user; not required.

Default

None

Envelope TransferEncoding

Description

Specifies the transfer encoding, for example, base64, to be used for the SOAP envelope. It is recommended that you set this value.

If the message is in a different character format than the one you specify, or if transfer encoding is not specified, the message may become corrupted during data transport. See RFC 2045, Section 6.1, on the following Web site:

`http://www.ietf.org/rfc/rfc2045.txt`

If an entity is the type "multipart" the content transfer encoding is not permitted to have any value other than **7bit**, **8bit**, or **binary**. See RFC 2045, Section 6.4, on the previous Web site.

The output of base64 encoding confirms to the charset US-ASCII. See RFC 2045, Section 6.2, on the previous Web site.

Values

Valid values are **7bit**, **8bit**, **binary**, **quoted-printable**, **base64**, **ietf-token**, and **x-token**, and the parameter entries are not case-sensitive; not required.

Note – Although this parameter is not required, it is recommended.

Default

7bit

Envelope MIME Parameters

Description

Allows for the specification of additional MIME parameters for the envelope, which are in conformance with the MIME (RFC 2045) specification.

See RFC 2045 on the following Web site:

`http://www.ietf.org/rfc/rfc2045.txt`

Note – Implementations may ignore any MIME header not defined in the ebXML Message Service Specification, version 2.0 (see the ebXML Web site). For example, an implementation could include **content-length** in a message. However, a recipient of a message with **content-length** could ignore it.

Values

A valid string supplied by the user; not required.

Default

boundary=MIME_boundary, **type=text/xml**, **start=Envelope**

Envelope ContentType

Description

Specifies the MIME content type to be used for the envelope.

Values

Must either be **text/xml** or **application/xml**; required

Default

text/xml

Envelope Role

Description

Identifies whether the role of the party sending and/or receiving the message is authorized.

Values

fromAuthorizedRole or **toAuthorizedRole**; not required.

Default

None

Envelope SignFlag

Description

When set to **true**, specifies that the envelope is digitally signed.

Values

true or **false**; required.

Default

None

Envelope Sign Transforms1 Algorithm

Description

Specifies the URI that designates the algorithm to be performed when creating or validating a signature.

You must enter the following value:

```
http://www.w3.org/2000/09/xmldsig#enveloped-signature
```

For more information see the ebXML Message Service Specification, version 2.0, Section 4.1.3 (see the ebXML Web site).

Value

http://www.w3.org/2000/09/xmldsig#enveloped-signature; required.

Default

Same as above

Envelope Sign Transforms2 Algorithm

Description

Enter this parameter value for use in conjunction with the **Sign Transforms XPath** value, to exclude other elements within the SOAP envelope.

You must enter the following value:

```
http://www.w3.org/TR/1999/REC-xpath-19991116
```

For more information see the ebXML Message Service Specification, version 2.0, Section 4.1.3 (see the ebXML Web site).

Value

http://www.w3.org/TR/1999/REC-xpath-19991116; required.

Default

Same as above

Envelope Sign Transforms XPath

Description

Specifies the **XPath** expression to be used with the transform element.

You must enter this value as shown below:

```
not(ancestor-or-self::node()[@soap:actor='urn:oasis:names:tc:ebxml-
msg:actor:nextMSH'] | ancestor-or-self::node()[@soap:actor='http://
schemas.xmlsoap.org/soap/actor/next'] )
```

For more information see the ebXML Message Service Specification, version 2.0, Section 4.1.3 (see the ebXML Web site).

Values

not(ancestor-or-self::node()[@soap:actor='urn:oasis:names:tc:ebxml-msg:actor:nextMSH'] | ancestor-or-self::node()[@soap:actor='http://schemas.xmlsoap.org/soap/actor/next']); required.

Default

Same as above

Envelope Sign Transforms3 Algorithm

Description

Specifies the URI that designates the algorithm to be performed when canonicalizing the payload's XML (if necessary).

When used, you must enter this value as shown below:

```
http://www.w3.org/TR/2001/REC-xml-c14n-20010315
```

Value

http://www.w3.org/TR/2001/REC-xml-c14n-20010315; not required

Default

None

Payload1 ContentID

Description

Allows you to specify a unique, identifying name for each payload. For more information, see "Envelope ContentID" on page 52.

Values

A valid string supplied by the user; not required.

Default

None

Payload1 ContentLocation

Description

A URI designating the content-location of the MIME body part of the payload object.

Values

A valid URI supplied by the user; not required.

Default

None

Payload1 TransferEncoding

Description

Specifies the transfer encoding, for example, base64, to be used for the payload.

If the message is in a different character format than the one you specify, or if transfer encoding is not specified, the message may become corrupted during data transport. See RFC 2045, Section 6.1, on the following Web site:

`http://www.ietf.org/rfc/rfc2045.txt`

If an entity is the type "multipart" the content transfer encoding is not permitted to have any value other than **7bit**, **8bit**, or **binary**. See RFC 2045, Section 6.4, on the previous Web site.

The output of base64 encoding confirms to the charset US-ASCII. See RFC 2045, Section 6.2, on the previous Web site.

Values

Valid values are **7bit**, **8bit**, **binary**, **quoted-printable**, **base64**, **ietf-token**, and **x-token**, and the parameter entries are not case-sensitive; not required.

Default

None

Payload1 MIME Parameters

Description

Allows for the specification of additional MIME parameters for the payload, which are in conformity with the MIME (RFC 2045) specifications.

Note – Implementations may ignore any MIME header not defined in the ebXML Message Service Specification, version 2.0 (see the ebXML Web site). For example, an implementation could include **content-length** in a message. However, a recipient of a message with **content-length** could ignore it.

Values

A valid string supplied by the user; not required.

Default

None

Payload1 ContentType

Description

Specifies the MIME **content-type** to be used for the payload.

Values

A valid string supplied by the user; not required.

Default

None

Payload1 Role

Description

Identifies whether the role of the party sending and/or receiving the message is authorized.

Values

fromAuthorizedRole or **toAuthorizedRole**; not required.

Default

None

Payload1 SignFlag

Description

When set to **true**, specifies that the payload is digitally signed.

Values

true or **false**; not required.

Default

false

Payload1 Sign Transforms1 Algorithm

Description

Specifies the URI that designates the algorithm to be performed when creating or validating a signature.

Values

A valid URI string supplied by the user; not required.

Default

None

Payload1 Sign Transforms1 XPath

Description

Used in conjunction with the **Sign Transforms XPath** parameter to exclude elements within the payload.

Values

A valid string supplied by the user; not required.

Default

None

Payload1 EncryptFlag

Description

Specifies whether the current payload is encrypted.

Values

true or **false**; not required.

Default

false

Payload1 Encrypt Transforms1 Algorithm

Description

Allows you to specify transformations you want to be executed on the current payload, before encryption is applied to it. This parameter only applies to XML types of payloads.

Values

A valid XML string supplied by the user; not required.

Default

None

Payload1 Encrypt Transforms1 XPath

Description

Allows you to specify the **XPath** to be executed on the current payload.

Values

A valid **XPath** expression supplied by the user; not required.

Default

None

Note – The parameters for any additional payloads (**Payload2** , **Payload3** , and so on), if present, are the same as those for **Payload1** , and their values are set in the same way.

Inbound

These MAD parameters allow you to define messaging actions for inbound messages. For an example of how the parameters appear in the ePM user interface.

The initial set of parameters (before **Service Type**; see "Service Type" on page 50) are eXchange parameters common to all protocol managers. See the *eXchange Integrator User's Guide* for information on these parameters.

The rest of these parameters operate in the same way as those for the outbound. For details on their operation, settings, and values, see "Outbound" on page 49.

◆ ◆ ◆ CHAPTER 4

Quick Start Guide

This chapter provides a basic, general description of how to successfully operate ebXML Protocol Manager in eXchange. Its purpose is to provide a summary of how to get started using ebXML Protocol Manager.

What's in This Chapter

- "Overview: Basic Operations and Concepts" on page 63
- "Design Steps in Enterprise Designer" on page 64
- "Creating and Configuring TPs in ePM" on page 68
- "Run-time Steps" on page 69

For more information on any of the topics discussed in this chapter, see the *eXchange Integrator User's Guide*. You can also refer to the steps, as provided in "Run-time Steps" on page 69, for setting up the sample Projects that come with the ebXML Protocol Manager installation CD-ROM.

Overview: Basic Operations and Concepts

The following list describes the basic operations necessary to use ebXML Protocol Manager successfully:

- Create, design, deploy, and activate the appropriate Projects and their Deployment Profiles (DPs), using Enterprise Designer.
- As part of the previous operations, create and set up an Oracle external system instance on a B2B host's Environment using Enterprise Designer. For more information on Oracle database requirements for eXchange and ebXML Protocol Manager, see "System Requirements" on page 22.
- Also using Enterprise Designer, set up the appropriate features on the current Project's Environment. This operation includes some configuration steps.

- Configure and activate eXchange external and internal delivery channels (XDCs and IDCs) in ePM for the Trading Partners (TPs). This operation includes setting general eXchange configuration parameters and configuration steps specific to ebXML.

  Note – See "Increasing the Oracle Number of Processes" on page 29 for information on how to configure ebXML Protocol Manager in ePM.

- Perform the required initial run-time steps and run the Project.

This chapter briefly and generally describes the installation, setup, design, and run-time steps required to get an ebXML Protocol Manager Project and its ePM configuration set up and running. The chapter's purpose is to provide a quick, easy set of procedures for reaching these goals, for the user who does not necessarily want or need all of the details.

For a complete, more detailed explanation of how to perform these steps, using the sample scenario provided with the CD-ROM with ebXML Protocol Manager, see "Run-time Steps" on page 69. The rest of this chapter describes, in general and without detail, how you can quickly perform the steps shown in the previous list.

Design Steps in Enterprise Designer

For ebXML Protocol Manager sample Project implementation, design-time procedures in Enterprise Designer consist of the following basic operations:

- "Setting Up B2B Host Projects" on page 64
- "Creating and Activating Project DPs" on page 67

The rest of this section describes these operations.

Setting Up B2B Host Projects

You must create a B2B host Project for each ebXML Protocol Manager installation. This section provides a summary of this operation.

For complete information on how to perform these operations, see the *eGate Integrator User's Guide* and the *eXchange Integrator User's Guide*, as well as the sample scenario provided in Chapter 5, "Implementation Scenario."

Using Projects

Before you can use the ebXML Protocol Manager with eXchange, perform the following actions:

- Create and set up at least one ebXML B2B host Project

- Create and set up other Projects, as needed
- Deploy and activate the Project Profiles

The sample scenario provided in Chapter 5, "Implementation Scenario," assumes you use the sample Projects provided and the default configurations where possible.

Using Environments

Each ebXML Protocol Manager installation requires at least one Environment. The sample scenario provided in Chapter 5, "Implementation Scenario," assumes you use the sample Environments provided and the default configurations for all servers where possible.

You must also make any changes where needed, for example:

- Ports: Run the SeeBeyond Integration Server on ports 18000 through 18009; to use a different application server, you must use different port numbers.
- HTTP(S): To configure your HTTP server or client to use SSL, see the HTTP(S) eWay Intelligent Adapter User's Guide for eWay settings and Integration Server configuration.
- Oracle: You must create a new outbound Oracle external system instance and configure it. The eXchange database instance runs on Oracle.

Each of the companies in the sample scenario has an eXchange installation with one Environment.

Creating B2B Host Projects' Environments

Your first step is to create the Project's Environment and its basic setup as follows:

- Using Enterprise Designer, create a new Environment.
- Create at least one new Logical Host, as follows:
  - Create the Integration Server for the Logical Host.
  - Create the SeeBeyond JMS IQ Manager for that Logical Host.

  Create a keystore then import at least one key file and certificate file to activate the protocol's cryptographic features.

Create the following external system instances:

- Batch eWay in local file mode
- Inbound and outbound File eWays
- HTTP(S) Server
- HTTP(S) (client)
- Oracle (must also be configured; see "Setting Up Oracle External System" on page 66)

Creating and Activating B2B Host Project DPs

Activating a Project DP creates an eXchange service that provides a connection to Enterprise Manager's Message Tracking feature and the eXchange database.

The B2B host Project contains the following components:

- At least one message service used by the B2B host. It contains messaging actions, alternating inbound and outbound.
- The B2B host component, which performs the following functions:
  - Has business protocols that reference the Project's services.
  - Defines XDCs for the Project's messaging attribute definitions (MADs). For transport to and from TPs, these channels reference the standard SeeBeyond-supplied HTTP transport attribute definitions (TADs).
  - Defines IDCs (optional). These channels use the Java Messaging Service (JMS) to communicate in the Sender direction.

  A Connectivity Map with the following components:
  - Input, an instance of the B2B host, with two outbound connections
  - Output, an instance of the eXchange database (Oracle), with two inbound connections
  - Instance of Message Tracking, connecting to the other two components

  Note – For more information on the Message Tracking feature, see the **eXchange Integrator User's Guide.**

  You must activate the B2B host's DP. This action creates an eXchange service for the B2B host. The creation and activation of this service is necessary for data communication between TPs.

  Note – Before you activate a B2B host Project DP, this DP must be pointed to the appropriate Environment, and this Environment must contain a correctly configured Oracle external system instance.

Setting Up Oracle External System

You must set up an Oracle external system instance in a B2B host's Environment in Enterprise Designer. Perform this operation using the following general steps:

- Create an Oracle external system instance for the current Project on the Environment.
- Set the instance up, that is, customize it, for your intended uses.

For complete information on how to create a new external system and its instance in Enterprise Designer's Environment, see the *eGate Integrator User's Guide*. Also, see the *Oracle eWay Intelligent Adapter User's Guide*.

The procedure in the rest of this section explains how to set up an Oracle external system instance for ebXML Protocol Manager.

▼ To set up an Oracle external system instance

1. **In the Environment Explorer tree, navigate to the appropriate Project.**

2. **Correctly configure the following properties in the eWay's Properties dialog box:**

   - **DatabaseName**
   - Delimiter
   - Description
   - **Password**
   - **PortNumber**
   - **ServerName**
   - **User**

   Note – Other properties are optional.

   ---

3. **When all properties have been configured correctly for your site, click OK.**

See Also The Project's Environment now has its Oracle external system instance configured correctly for activation of the B2B host. For details, see the sample scenario explained in "Run-time Steps" on page 69.

Creating and Activating Project DPs

To run all ebXML Protocol Manager Projects, you must create, set up, and activate DPs for the following types of Projects:

- B2B host
- Inbound and outbound
- Error-handler
- Additional Projects, as required by your data communication setup (see the sample scenario explained in "Run-time Steps" on page 69 for examples of additional types of supporting Projects that may be required)

Note – There are also DPs named **dpMain** <*Project name*> that reside inside the Project folder named **eXchange** , under the sub-Project **Deployment** .

Create, set up, and configure additional **dpMain** Projects in the same way as you do the Project for the B2B host. Keep in mind that details of these steps may vary, depending on the reason for and functioning of any additional Project.

Note – Only the B2B host Project's Environment requires an Oracle external system instance.

Use Enterprise Designer to activate each DP. See "Run-time Steps" on page 69 and the *eXchange Integrator User's Guide* for details.

Creating and Configuring TPs in ePM

This section describes the steps for creating and configuring TPs using ePM.

Basic ePM Steps

Before you begin this operation, your Repository and your eXchange Oracle database must be running and accessible to eXchange.

Note – Enterprise Designer does not need to be running, and you do not need to have any Logical Hosts running.

Using ePM to create TPs includes the following basic operations:

- Starting eXchange ePM
- Creating (or importing) the necessary TPs in ePM
- Configuring the TPs in ePM, including XDCs, named **Delivery Channels** in ePM, and IDCs; IDCs are optional
- Activating the TPs

For ePM configuration information, see "Increasing the Oracle Number of Processes" on page 29. For an implementation example, including importing and activating TPs, see the sample scenario in "Run-time Steps" on page 69.

Configuring eXchange Service With Cryptographic Features

Your use of the ebXML protocol assumes you are also using its cryptographic features (encryption, decryption, signatures, and verifications). Additional steps are required in setting up the eXchange service to use these features.

You must associate encryption information with each XDC eXchange service. For complete information on setting up an eXchange service with cryptographic features for Protocol Managers, see the *eXchange Integrator User's Guide*. For specific examples, see the encryption setup used in the sample Project explained in "Run-time Steps" on page 69.

Run-time Steps

You run an ebXML Protocol Manager Project using the Logical Host. Running a Project requires the following basic steps:

- Starting and running the Logical Hosts
- Making sure your TPs have been activated
- Supplying input data
- Checking output data
- Making sure all client Projects and eXchange deployments deployed successfully

You can check the output data either by checking the configured output file directory or using the eXchange Message Tracking feature, whichever is appropriate. For more information on Message Tracking, see "Using Message Tracking" on page 100.

Note – You can use Enterprise Designer to make changes to a Project while it is running. See "Applying Changes Without Shutting Down Logical Host" on page 99 for details.

Once you do these steps, your Project is active, and you are ready to send and receive data. For detailed information on these steps, see the sample scenario provided with ebXML Protocol Manager and supplied with the installation CD-ROM, as explained in "Run-time Steps" on page 69.

◆ ◆ ◆ CHAPTER 5

Implementation Scenario

This chapter provides an explanation of the sample Projects provided with ebXML Protocol Manager, showing how you can use eXchange to achieve B2B solutions using the ebXML protocol.

What's in This Chapter

Overview of Sample Implementation

The sample ebXML Protocol Manager Projects' implementation scenario demonstrates inbound and outbound message processing between the following parties:

- Atlanta Company
- Berlin Company

For the sample's scenario, each company has an eXchange installation and trades data. The scenario also demonstrates ebXML in conjunction with SME to illustrate cryptographic features.

Overview of Basic Sample Setup

For complete information, see the *eXchange Integrator User's Guide*.

Sample Scenario

The sample Projects, Environments, and related files illustrate the following scenario:

- The Atlanta Company's Trading Partner (TP) is **tpBerlin**, that is, the Berlin Company.
- The Berlin Company's TP is **tpAtlanta**, that is, the Atlanta Company.
- In the sample scenario, the Atlanta Company sends data (outbound files) to the Berlin Company, which processes it, then sends the processed data (inbound files) back to the Atlanta Company.
- You can change the sample's scenario (as explained in this chapter) to reverse the companies' sender and receiver roles, if you want.

For more information on the file transactions between the companies, see the following sections:

- "Configuring eWays on Connectivity Maps" on page 82
- "Supplying Input Data" on page 99

Sample Data

The sample data contains the following files:

- Sent from the Atlanta Company:
  - **Payload.xml**
  - **Blue hills.jpg**
  - **Outbound_NoPayload.txt**
  - **Outbound_Ping.txt**
  - **Outbound_PO_AckReq.txt**
  - **Outbound_PO_Binary.txt**
  - **Outbound_PO_DupCheck.txt**
  - **Outbound_PO_NoAckReq.txt**
  - **Outbound_StatusRequest.txt**
- Sent from the Berlin Company:
  - **Payload.xml**
  - **Blue hills.jpg**
  - **Outbound_NoPayload.txt**
  - **Outbound_Ping.txt**
  - **Outbound_POAccep_Binary.txt**
  - **Outbound_POAccep_NoAckReq.txt**
  - **Outbound_POAcceptance.txt**
  - **Outbound_StatusRequest.txt**

For a complete list of files contained in the sample, see "Extracted Files and Directories" on page 73.

Importing Sample Projects and Environments

This section explains how to import the ebXML Protocol Manager sample Project, Environment, and related files. It also lists these files.

Importing the sample's files requires the following steps:

- "Extracting Sample Files" on page 73
- "Importing Sample Projects" on page 74
- "Importing Sample Environments" on page 75

Extracting Sample Files

This section explains how to extract the ebXML Protocol Manager sample files from the file **ebXML_Manager_Sample.zip** that contains them. You must perform this operation before you can import an ebXML Protocol Manager sample Project or Environment.

ebXML_Manager_Sample.zip is provided on the CD-ROM with ebXML Protocol Manager. See Chapter 2, "Installing ebXML Protocol Manager,"for instructions on how to obtain (download) this file.

▼ To extract the sample files

● **Extract the contents of the ebXML_Manager_Sample.zip file to a temporary directory, for example,** `C:\temp\eXchange`**. Make sure you preserve the extracted file paths.**

Note – It is recommended that you first create a temporary eXchange directory for the sample files and extract these files to this directory for your convenience. This action allows you to easily access the files while performing procedures provided later in this chapter.

Extracted Files and Directories

If you use the `C:\temp\eXchange` top-level directories, the following directories and files are created:

```
 C:\temp\eXchange\Sample\Certificates\CompanyA-Cert.der
C:\temp\eXchange\Sample\Certificates\CompanyB-Cert.der
C:\temp\eXchange\Sample\Data\Atlanta\payload\Blue hills.jpg
C:\temp\eXchange\Sample\Data\Atlanta\payload\payload.xml
```

```
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_NoPayload.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_Ping.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_PO_AckReq.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_PO_Binary.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_PO_DupCheck.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_PO_NoAckReq.txt
C:\temp\eXchange\Sample\Data\Atlanta\Outbound_StatusRequest.txt
C:\temp\eXchange\Sample\Data\Berlin\payload\payload.xml
C:\temp\eXchange\Sample\Data\Berlin\payload\Blue hills.jpg
C:\temp\eXchange\Sample\Data\Berlin\Outbound_NoPayload.txt
C:\temp\eXchange\Sample\Data\Berlin\Outbound_Ping.txt
C:\temp\eXchange\Sample\Data\Berlin\Outbound_POAccep_Binary.txt
C:\temp\eXchange\Sample\Data\Berlin\Outbound_POAccep_NoAckReq.txt
C:\temp\eXchange\Sample\Data\Berlin\Outbound_POAcceptance.txt
C:\temp\eXchange\Sample\Data\Berlin\Outbound_StatusRequest.txt
C:\temp\eXchange\Sample\Environments\CommonEnvironments.zip
C:\temp\eXchange\Sample\Keys\CompanyA-Key.p12
C:\temp\eXchange\Sample\Keys\CompanyB-Key.p12
C:\temp\eXchange\Sample\Projects\SampleProjects.zip
C:\temp\eXchange\Sample\TradingPartners\envBerlin_AtlantaSME.xml
C:\temp\eXchange\Sample\TradingPartners\envAtlanta_BerlinSME.xml
```

Caution – If you use the **\temp** directory, and your system does any kind of auto-delete operation on any directory with this name, you must disable this function.

As shown in the previous list, in addition to files for the Projects and Environments, **ebXML_Manager_Sample.zip** also contains key, certificate, TP, and data files.

Note – Before you import the Environments, you must first extract them from the **CommonEnvironments.zip** file.

If You Use Different Path Locations

If you extract the data files to different locations, modifications are necessary in the appropriate Enterprise Designer configurations. You must also modify the appropriate eXchange ePM fields under the **ToPartner** and **FromPartner** tabs.

Importing Sample Projects

Before you import the sample Projects, your Repository must be running, and you must be logged on to Enterprise Designer. If your Repository already has a Project at the root level whose name is identical to any of the Projects you are importing, you must delete or rename such Projects before you start.

▼ To import sample Projects

1. **In Project Explorer, right-click the Repository and, on the context menu, click Import.**

2. **In the Import Manager dialog box, browse to the directories where you installed the sample files (such as `C:\temp\eXchange\Sample\ebXML\Projects`).**

   The name of the Project file you must import is **SampleProjects.zip**.

3. **Select this file and click Import.**

   A message appears saying the import operation was successful.

4. **Click OK.**

5. **Close the Import Manager dialog box.**

   The **Project Explorer** tree now displays the following Projects under **ClientProjects**:

   - **ClientInebXML**
   - **ClientOutebXMLAtlanta**
   - **ClientOutebXMLBerlin**
   - HostAtlanta
   - HostBerlin
   - **JmsToFile**
   - **ToInternal**

Using Deployment Profiles

To run an ebXML Protocol Manager Project, you must create and activate at least one Deployment Profile for the Project. The general types of Projects are:

- B2B host Projects
- Main Projects
- Outbound and inbound Projects
- Error-handler Projects

Importing Sample Environments

Before you import the sample Environments, your Repository must be running, and you must be logged on to Enterprise Designer. If your Repository already has an Environment at the root level whose name is identical to any of the Environments you are importing, you must delete or rename such Environments before you start.

▼ To Import Sample Environments

1. **In Environment Explorer, right-click the Repository and, on the context menu, click Import.**

2 In the Import Manager dialog box, browse to the directories where you installed the sample files (such as `C:\temp\eXchange\Sample\ebXML\Environments`).

The name of the Environment file you must import is **CommonEnvironment.zip**.

3 Select this file and click Import.

A message appears saying the import operation was successful.

4 Click OK.

5 Close the Import Manager dialog box.

The **Environment Explorer** tree now displays the following Environments:

- **envAtlanta**
- **envBerlin**

Enterprise Designer Steps

For the ebXML Protocol Manager sample Project implementation, design-time procedures in Enterprise Designer for setting up and activating the sample's scenario consist of the following operations:

- "Setting Up Companies' Environments" on page 76
- "Updating Files for Java Collaboration Definitions" on page 80
- "Setting Up Companies' B2B Host Projects" on page 81
- "Configuring Companies' Cryptographic Features" on page 85
- "Creating Additional Deployment Profiles" on page 88

The rest of this section explains these operations.

Setting Up Companies' Environments

This section explains how to set up the Atlanta and Berlin Companies' sample Environments, **envAtlanta** and **envBerlin**.

The sample assumes you use default configurations for all servers, where possible, and that you make any changes in Enterprise Designer, where needed, for example:

- Oracle: You must create a new outbound Oracle external system instance for each Environment and configure it, even if you imported the sample Environments. Sample parameters are for reference only. Any Oracle database used by eXchange must be accessible to eGate, and you must know its Oracle SID, user name, and password.
- Ports: To run anything other than Integration Server on ports 18000 through 18009, you must make adjustments to configuration values for port numbers, depending on the type of application server you are using.

- HTTPS: For information on how to configure your HTTP server or client to use SSL, see the HTTP(S) eWay Intelligent Adapter User's Guide, for eWay settings and Integration Server configuration.

Note – After you have created all of the external system instances you need for the current B2B host Project, another external system instance, an eXchange service, is created in this Project automatically when you activate the Project. See "Setting Up Companies' B2B Host Projects" on page 81 for details.

Creating Atlanta Company's Environment

This section explains how to create the sample's Environment for the Atlanta Company.

▼ To create the basic components

1. **In Enterprise Designer, near the lower left of the window, click the Environment Explorer tab.**
2. **On the Environment Explorer tree, right-click the Repository and, on the context menu, click New Environment.**
3. **Name the newly created Environment envAtlanta.**
4. **Right-click envAtlanta and, on the menu, click New Logical Host and name the Logical Host lhAtlanta.**

Note – If you want to create any additional Logical Host, right-click it and open its **Properties** dialog box. Click Logical Host Configuration and change the value Logical Host Base Port to a larger multiple of 1000 (28000 if ports 28000-28009 are unused; otherwise 28100 or 29000). When you are finished, close the **Properties** dialog box.

---

5. **Create an Integration Sever under lhAtlanta and name it isAtlanta.**
6. **Create a Seebeyond JMS IQ Manager under lhAtlanta and name it jmsAtlanta.**

▼ To create and configure the nondatabase external system instances

1. **In Enterprise Designer, on the Environment Explorer tree, right-click envAtlanta and, on the context menu, click New BatchFTP External System.**
2. **Name the new external system instance extBatchFtpAtlanta, and click OK.**

   These operations create, for the Environment, an external system instance for the Batch eWay in FTP mode.

3. **On the new external system's Properties dialog box, allow the default ports for SOCKS, FTP, and SSH Tunneling.**

4. **Create a Batch eWay (local file mode) external system under envAtlanta and name it extBatchLocalFileAtlanta.**

5. **Create a File eWay (inbound mode) external system under envAtlanta and name it extFileInAtlanta.**

6. **Create a File eWay (outbound mode) external system under envAtlanta and name it extFileOutAtlanta.**

7. **Create an HTTP eWay (client mode) external system under envAtlanta and name it extHttpAtlanta.**

8. **On the new HTTP eWay's Properties dialog box, enter the in the following values:**
   - **URL Syntax**: `http://www.seebeyond.com`
   - **Content Type**: Leave blank.

9. **Create an HTTP Server eWay external system under envAtlanta and name it extHttpServerAtlanta.**

10. **Create a new keystore under envAtlanta and name it envAtlanta_ks_store. See "Configuring Companies' Cryptographic Features" on page 85 for details on how to add keys and certificates to the keystore.**

▼ To create and configure the Oracle external system instance

1. **In Enterprise Designer, on the Environment Explorer tree, right-click envAtlanta and, on the context menu, click New Oracle External System.**

   Note – The eXchange database instance runs on Oracle. For more information Oracle requirements for eXchange and ebXML Protocol Manager, see "System Requirements" on page 22.

   ---

2. **Name the new component extOracleOutAtlanta, designate it Outbound Oracle eWay, and click OK.**

   These operations create, for the Environment, an external system instance for the Oracle eWay in outbound mode.

3. **Right-click extOracleOutAtlanta and configure properties appropriately, as follows:**
   - **DatabaseName**: SID for your current Oracle system.

- **DataSourceName**: **local**.
- **Delimiter**: Symbol #.
- **Description**: Oracle thin driver Connection Pool Datasource.
- **Driver Properties**: Blank, for this sample.
- **Password**: Valid password for the current Oracle system.
- **PortNumber**: **1521** (change this value only if your Oracle system administrator changed the default).
- **ServerName**: **myMachine**: Host name of the Oracle server machine.
- **TNS Entry**: Blank, for this sample.
- **User**: Valid user ID for the current Oracle system.

4 **When all properties have been configured correctly for your site, click OK.**

When You Are Finished

- Your result appears as shown in "When You Are Finished" on page 79.
- Collapse the **envAtlanta Environment Explorer** tree, click Save All, and close all canvases.

Creating Berlin Company's Environment

This section explains how to create the sample's Environment for the Berlin Company.

▼ To create the basic components

1 **Follow the steps provided under the "Creating Atlanta Company's Environment" on page 77.**

2 **Name the appropriate components as follows:**

- envBerlin
- **lhBerlin**
- **isBerlin**
- **jmsBerlin**

▼ To create and configure the nondatabase external system instances

1 **Follow the steps provided under the "Creating Atlanta Company's Environment" on page 77, under envBerlin.**

2 **Name the appropriate external system instances as follows:**

- **extBatchFtpBerlin** (use default properties)

- **extBatchLocalFileBerlin**
- **extFileInBerlin**
- **extFileOutBerlin**
- **extHttpBerlin** (use the same properties values as **extHttpAtlanta**)
- **extHttpServerBerlin**.
- **envBerlin_ks_store**.

▼ To create and configure the Oracle external system instance

1. **Follow the steps provided under the "Creating Atlanta Company's Environment" on page 77, using envBerlin.**

2. **Name the new component extOracleOutBerlin, designate it Outbound Oracle eWay.**

3. **Configure extOracleOutBerlin as shown in the list provided in step "Creating Atlanta Company's Environment" on page 77 under the "Creating Atlanta Company's Environment" on page 77"Creating Atlanta Company's Environment" on page 77.**

When You Are Finished

- Your result appears as shown in "When You Are Finished" on page 80.
- Collapse the **envBerlin Environment Explorer** tree, click Save All, and close all canvases.

Updating Files for Java Collaboration Definitions

Special steps are required to update the ebXML Protocol Manager's sample Java Collaboration Definitions (JCDs) to take advantage of the cryptographic .jar files used in SME and the sample.

▼ To update the JCDs used for signing and verifying

1. **In Enterprise Designer's Project Explorer tree open this Project folder: SeeBeyond > eXchange > User Components > Crypto > Sign > JCDs.**

2. **For each of the four elements under the JCDs folder, perform the following steps:**

   a. **Right-click the JCD and check it out.**

   b. **Double-click the JCD to edit it in the Collaboration Editor (Java).**

   c. **On the Tool Palette, click Import JAR file.**

   d. **In the Add/Remove Jar Files dialog box, click Add.**

e. **Navigate up four levels, then down to SeeBeyond > SME > External JARs.**

f. **Select the file com.stc.smeapi.jar.**

g. **Click Import and click Close.**

h. **Click Save.**

See Also For more information, see the *eGate Integrator User's Guide*.

Setting Up Companies' B2B Host Projects

On the **Project Explorer** tree, you can open a B2B host Project (HostBerlin or HostAtlanta) to display its components. This section provides a summary of a B2B host's contents.

Activating a B2B host Project creates an eXchange service that acts as a channel manager and provides a connection to Message Tracking and the eXchange database instance.

See "Using Message Tracking" on page 100 and the *eXchange Integrator User's Guide* for more information on Message Tracking.

Components of Companies' B2B Host Projects

The following list describes the B2B host Projects' components and their functions:

- bhAtlanta and bhBerlin are the actual B2B hosts. These hosts have the following components and functions:
  - **ebXML**, under **Business Protocols** (BPs), the only messaging attribute definition (MAD) contained in the Project, for each B2B host.
  - BPs that reference two message services (both under **ebXML**): **PurchaseService** and urn:oasis:names:tc:ebxml-msg:service, for each B2B host.
  - **PurchaseService** message service contains the Projects' messaging actions, inbound and outbound.
  - There are two external delivery channels (XDCs) for the single **ebXML** MAD in each B2B Host. These XDCs are named xdcAtlanta_ebXML_via_HTTP and xdcBerlin_ebXML_via_HTTP.
  - For transport to and from TPs, these XDCs reference the standard SeeBeyond-supplied HTTP transport attribute definitions (TADs).
  - Only bhAtlanta has internal delivery channels (IDCs). These channels use JMS to communicate in the sender (**ToInternal**) direction. These IDCs are:
    - **idcAtlanta_ebXML_Send_via_File**
    - **idcAtlanta_ebXML_Recv_via_File**

The Enterprise Designer canvas cmHostAtlanta is a Connectivity Map. It shows a graphical view of the **cmHostAtlanta** components.

This Connectivity Map has the following components:

- **bhAtlanta1**: Only input, an instance of **bhAtlanta**, with two outbound connections.
- extOracleOut: Only output, an instance of extOracleOutAtlanta, with two inbound connections.
- **mtrk**: Instance of Message Tracking, connecting to the other two components.

Note – For more information on eXchange's Message Tracking, see the **eXchange Integrator User's Guide.**

The Enterprise Designer canvas cmHostBerlin is also a Connectivity Map (similar to "Components of Companies' B2B Host Projects" on page 81) and has the following components:

- **bhBerlin1**: Only input, an instance of **bhBerlin**, with two outbound connections.
- extOracleOut: Only output, an instance of extOracleOutBerlin, with two inbound connections.
- **mtrk**: Instance of Message Tracking, connecting to the other two components.

Configuring eWays on Connectivity Maps

This section explains how to configure the properties of eWay external system instances on the sample Projects' Connectivity Maps.

Note – If directories listed in this section are not already present as subdirectories of `C:\temp\eXchange\Sample\`, you must create them, according to the path locations shown.

Use the default properties, except in the following cases:

File eWay: From Atlanta

The Atlanta Company's **clientOutebXMLAtlanta** Project Connectivity Map **cmClientOutebXML** has an eWay external system instance named **extFileIn**. Modify the default properties as follows:

- **Directory**: `C:\temp\eXchange\Sample\Data\Atlanta\OutboundebXML\`
- **Input file name**: **Outbound*.txt**

File eWay: From Berlin

Inbound: The Berlin Company's **clientOutebXMLBerlin** Project Connectivity Map **cmClientOutebXML** has an eWay external system instance named **extFileIn**. Modify the default properties as follows:

- **Directory**: `C:\temp\eXchange\Sample\Data\Berlin\OutboundebXML\`
- **Input file name**: **Outbound*.txt**

Batch eWay

Configure the **Target Directory Name** property for the external system instance **extBatchLocalFileOut** on the Connectivity Map **cmToInternal** as follows:

```
C:\temp\eXchange\Sample\Data\ToInternal
```

Activating Companies' B2B Host Deployment Profiles

Before performing this operation, make sure you are using the current Project's Environment (**envAtlanta** or **envBerlin**) and that it contains a correctly installed and configured Oracle external system instance (see the "Creating Atlanta Company's Environment" on page 77). Your Oracle system must be running, because the eXchange database instance runs on Oracle.

Note – When you activate a B2B host, eXchange automatically creates another external system instance, an eXchange service. For more information on this service, see the **eXchange Integrator User's Guide** .

▼ To activate the B2B host's Deployment Profile for the Atlanta Company

1. **On Enterprise Designer's Project Explorer tree, right-click HostAtlanta and, on the context menu, point at New and click Deployment Profile.**

2. **Name the new Deployment Profile dpHostAtlanta, point it at envAtlanta, and click OK.**

   The Deployment Editor opens. Its left pane shows two services and two Oracle external system instances (representing Oracle eWays).

   Its right pane contains windows representing the basic components and external system instances in **envAtlanta**.

3. **On the right pane, minimize all windows except lhAtlanta and extOracleOutAtlanta.**

4. **Click the Automap button. See "Activating Companies' B2B Host Deployment Profiles" on page 83.**

   The services and eWays automatically map to the appropriate windows on the right pane. The result looks like "Activating Companies' B2B Host Deployment Profiles" on page 83, except *without* the eXchange service.

5. **Click Save All.**

6. **Click Activate to activate the Deployment Profile.**

   A dialog box appears, indicating the activation is successful. A new external system instance is created, named bhAtlanta1 eXchange Service. You can view this service on **envAtlanta** on the **Environment Explorer** tree, as well as on the right pane of **Deployment Editor**.

   A dialog box appears with a message communicating the status of the activation. If the activation is not successful, repeat the steps in this procedure, carefully rechecking every action. If the activation is successful, go on to the next step.

7. **After the Deployment Profile is activated successfully, a dialog box with the message Apply Environment updates as well. [LogicalHost may be restarted] appears, prompting you to apply the changes to all running Logical Hosts. You can take one of the following actions:**

   - **If no Logical host is running, click No.**
   - **If one or more Logical Hosts are running, click Yes, then decide whether there are Environment changes to be applied and:**
     - **If yes, check the check box and press Enter.**
     - **If no, clear the check box and press Enter.**

     Note – For the purpose of this sample Project, at this point in the procedure, leave the dialog box's check box checked and press **Esc** .

     ---

8. **When you are finished, click Save All, close all canvases, and click Refresh All from Repository.**

▼ To activate the B2B host's Deployment Profile for the Berlin Company

1. **On Enterprise Designer's Project Explorer tree, right-click HostBerlin and, on the context menu, point at New and click Deployment Profile.**

2. **Name the new Deployment Profile dpHostBerlin, point it at envBerlin, and click OK.**

   The Deployment Editor opens. Its right pane shows two services and two Oracle external system instances (representing Oracle eWays).

Its left pane contains windows representing the basic components and external system instances in **envBerlin**. The result is similar to that shown in "Activating Companies' B2B Host Deployment Profiles" on page 83.

3 **On the right pane, minimize all windows except lhBerlin and extOracleOutBerlin.**

4 **Click the Automap button (see "Activating Companies' B2B Host Deployment Profiles" on page 83).**

The services and eWays automatically map to the appropriate windows on the right pane.

5 **Click Save All.**

6 **Click Activate to activate the Deployment Profile.**

A dialog box appears, indicating the activation is successful. A new external system instance is created, named bhBerlin1 eXchange Service. You can see this service on **envBerlin**, under **extOracleOutBerlin** on the **Environment Explorer** tree.

The result is similar to that shown in "Activating Companies' B2B Host Deployment Profiles" on page 83.

7 **Click or press the appropriate button or key to close the dialog box.**

Note – For the purpose of this sample Project, at this point in the procedure, leave the dialog box's check box checked and press **Esc** .

---

8 **When you are finished, click Save All, close all canvases, and click Refresh All from Repository.**

Troubleshooting Tips

If **extOracleOutAtlanta** or **extOracleOutBerlin** refuses to accept eWays, this issue may be an indication that the referenced eXchange database instance is:

- Inaccessible, in which case ensure that Oracle is running and that the Environment's **extOracleOutAtlanta** or **extOracleOutBerlin** properties match the instance's host name, SID, user name, and password. If necessary, see the "Creating Atlanta Company's Environment" on page 77 or the *eXchange Integrator User's Guide* for more information.
  - Misdefined as inbound, in which case delete the Environment's **extOracleOutAtlanta** or **extOracleOutBerlin** and re-create either instance as outbound. Then, click Save All, followed by Refresh All from Repository.

Configuring Companies' Cryptographic Features

This section explains how to configure the eXchange Service's cryptographic features.

Note – To use ebXML Protocol Manager's cryptographic features, make sure you have installed the appropriate . **jar** files, as explained under "After You Install" on page 25.

Since the sample assumes you are using cryptographic features (encryption, decryption, signatures, and verifications), additional steps are required for configuring these features for bhAtlanta1 eXchange Service and bhBerlin1 eXchange Service.

Note – If the Logical Host is running while you are configuring private keys, you must apply any changes you make by selecting the **Environment Explorer** tree, then right-clicking lhAtlanta or lhBerlin (whichever is current) and, on the context menu, click Apply.

▼ To configure the private key for the Atlanta Company

1. **On the envAtlanta Environment Explorer tree, right-click envAtlanta_ks_store.**

2. **On the context menu, click Manage Private Keys.**

   A **Private Keys** dialog box appears.

3. **Click Import.**

   An **Import Private Keys** dialog box appears.

4. **Enter the following information, all lower-case:**

   - **Alias: privatekey1**
   - **Password: companya**

5. **Browse to and select the CompanyA-Key.p12 file (see "Extracted Files and Directories" on page 73).**

6. **Click OK, then click Close.**

7. **Right-click bhAtlanta1 eXchange Service and, on the context menu, click Properties.**

   A **Properties** dialog box appears. This dialog box allows you to configure the public and private keys for the current B2B host's XDC named xdc_Atlanta_ebXML_via_HTTP.

8. **Using the drop-down menus, select privatekey1 under the Signature Key and Decryption Key columns. See "Configuring Companies' Cryptographic Features" on page 85.**

9. **Click OK.**

Note – If **privatekey1** does not appear in the drop-down list, click the ellipsis [...] and click Import. Using the alias **privatekey1** , import the **CompanyA-Key.p12** file with the previously given password.

---

10 **When you are finished, click Save All.**

▼ To configure the public certificate for the Atlanta Company

1 **On the envAtlanta Environment Explorer tree, right-click envAtlanta_ks_store.**

2 **On the context menu, click Manage Public Certificates.**

A **Public Certificates** dialog box appears.

3 **Click Import.**

An **Import Public Certificates** dialog box appears.

4 **Enter the following information, all lower-case:**

5 **Alias: signkey**

6 **Browse to and select the CompanyB-Cert.der file (see "Extracted Files and Directories" on page 73).**

7 **Click OK, then click Close.**

8 **When you are finished, click Save All.**

Note – Additionally, if the Logical Host is running, you must, on the **Environment Explorer** tree, right-click lhAtlanta or **lhBerlin** and, on the context menu, click Apply.

▼ To configure the private key and public certificate for the Berlin Company

● **Follow the envAtlanta steps provided under the "Configuring Companies' Cryptographic Features" on page 85 and the "Configuring Companies' Cryptographic Features" on page 85 with envBerlin, with the following exceptions:**

- For the **envBerlin** key store password, enter **companyb**.
- Import the **CompanyB-Key.p12** file
- Import the **CompanyA-Cert.der** file.

Note – See "Extracted Files and Directories" on page 73.

Results

The appropriate cryptographic information is now configured and associated with the following XDCs for each B2B Host:

- xdc_Atlanta_ebXML_via_HTTP for bhAtlanta1 eXchange Service
- xdc_Berlin_ebXML_via_HTTP for bhBerlin1 eXchange Service

Creating Additional Deployment Profiles

On **Project Explorer**, you must create, automap, and activate additional Atlanta and Berlin Company Deployment Profiles for the rest of the Projects in the sample scenario, in the same way as described previously.

Create additional Deployment Profiles for the appropriate Environments and with the names, as shown in Table 5–1.

TABLE 5–1 Additional Deployment Profiles

Projects	Deployment Profiles and Names	Environments
ClientOutebXMLAtlanta	dpOutEbXMLAtlanta	envAtlanta
ClientOutebXMLBerlin	dpOutEbXMLBerlin	envBerlin
ClientInebXML	dpInEbXMLAtlanta	envAtlanta
	dpInEbXMLBerlin	envBerlin
JmsToFile	dpFileAtlanta	envAtlanta
	dpFileBerlin	envBerlin
Main (created by eXchange)	dpMainAtlanta	envAtlanta
	dpMainBerlin	envBerlin

Importing and Configuring TPs in ePM

This section explains how import TPs, as well as how to enter values and use the ePM configuration parameters for the TPs in the sample scenario.

Note – For procedures on how to create TPs, see the eXchange Integrator User's Guide.

Getting Started

Before you begin, your Repository and your eXchange database (Oracle) must be running and accessible. Enterprise Designer does not need to be running, and you do not need to have any Logical Hosts running.

In addition, you must perform the following operations:

- "Starting ePM" on page 89
- "Importing TPs" on page 90

The rest of this section explains these operations. For detailed information on configuring ePM, see "Increasing the Oracle Number of Processes" on page 29.

Starting ePM

This section explains how to start running ePM.

▼ To start eXchange ePM

1. **Start a new browser session (that is, do *not* clone a new window of an existing session)**

2. **Enter a Repository URL, with `epm` appended, for example:**

   - If your Repository were running local on port 12000, the URL is:

     ```
     http://localhost:12000/epm
     ```

   - For a Repository running on machine **herMachine** on port 33000, the URL is:

     ```
     http://herMachine:33000/epm
     ```

   - IP addresses are also permissible, for example:

     ```
     http://10.18.75.85:36271/epm
     ```

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

3. **When the sign-in window appears, enter Enterprise Manager user name and password and click Sign In.**

   The initial ePM window appears.

Importing TPs

Your next step is importing the Atlanta and Berlin Company TPs (TP files), as explained under this section. Keeping track of the TPs, where they are sent from and where they are received depends on which company is the current company. See “Current Company Viewpoint and TPs” on page 91.

This sample scenario has the following TPs:

- **tpBerlin**: For the Atlanta Company.
- **tpAtlanta**: For the Berlin Company.

▼ To import the tpBerlin to envAtlanta

1. **From the initial ePM window, on the upper left side, click Import.**

   The **Import New Trading Partner** window appears.

2. **Open B2B Repository then open envAtlanta.**

3. **Select bhAtlanta1.**

4. **Enter tpBerlin as the name for the new TP.**

5. **Browse to**

   ```
   C:\temp\eXchange\Sample\ebXML\TradingPartners
   and select the envAtlanta_BerlinSME.xml file and click Import.
   ```

   On the Explorer tree **tpBerlin** appears under **envAtlanta**.

▼ To locate tpBerlin in the ePM window

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

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

2. **Open the B2B Repository and envAtlanta and click bhAtlanta1.**

3. **Click Search, then click OK.**

   On the Explorer tree **tpBerlin** reappears under **envAtlanta**.

▼ To import tpAtlanta to envBerlin

- **Follow the same steps as those in the “Importing TPs” on page 90, with the following exceptions:**

- Under **B2B Repository** open envBerlin.
- Select **bhBerlin1** and enter **tpAtlanta** as the TP name.
- Import the file envBerlin_AtlantaSME.xml.

▼ To locate tpAtlanta in the ePM window

● **Search for tpAtlanta under envBerlin and bhBerlin1.**

Current Company Viewpoint and TPs

When you are configuring a TP and pointing it to **envAtlanta**, you must take the viewpoint of that Environment's company. So, the Atlanta Company's TP is tpBerlin.

Therefore, in terms of the companies, **ToPartner** means from the Atlanta Company and **FromPartner** to the Atlanta Company.

Conversely, When you are configuring a TP and pointing it to **envBerlin**, you must take the viewpoint of that Environment's company. So, the Berlin Company's TP is tpAtlanta.

Therefore, in terms of the companies, **ToPartner** means from the Berlin Company and **FromPartner** to the Berlin Company.

Configuring TP Parameters

When you import a TP, its related configuration parameter values are set in part based on parameters stored in the export file and in part based on the name of the TP. This section explains how to update these parameters under the following sections:

- "XDC Parameters" on page 91
- "IDC Parameters" on page 95

XDC Parameters

You must set the configuration properties governing the Atlanta or Berlin Company's message exchange with the appropriate TP using the companies' XDCs, both named ebXMLProtocol_delivery_channel1. These properties are the XDC parameters.

Before you begin to configure these parameters, be sure to display the appropriate TP's name in the ePM **Explorer** (lower left side of the window).

Note – The channels named **External Delivery Channels** (XDCs) in Enterprise Designer are named **Delivery Channels** in ePM. When referring to this component in ePM, the text still uses the term XDC, except when referring to specific text in the user interface.

This section explains how to set the **Delivery Channel** (external) parameters governing message exchange for **tpAtlanta** and **tpBerlin**.

▼ To configure the XDC parameters for the current TP

1 **In the Explorer (lower left) side of ePM, click tpAtlanta or tpBerlin.**

The canvas displays the TP's general properties.

2 **Click the Components tab.**

The selected TP's XDC parameters for that tab appear (see the following table.).

Binding Name	**tpAtlanta**: xdc_Berlin_ebXML_via_HTTP **tpBerlin**: xdc_Atlanta_ebXML_via_HTTP
ToPartner Transport Name	HTTP
FromPartner Transport Name	HTTP
Packager Name	ebXML

3 **Click the appropriate binding name. You are now entering parameters for this current binding.**

Note – The **General** parameters allow you to define general eXchange settings. See the *eXchange Integrator User's Guide* for information on these parameters.

4 **Click the ToPartnerTransport tab.**

The tab's parameters appear. These parameters allow you to supply information needed to transport data to the TP, that is, allowing the TP to *receive* data.

5 **For Use Synchronous Channel and Track For Auditing, enter false for both TPs.**

6 **For All Purpose End Point, enter:**

```
http://<machine_ID><port_no>
```

Where:

- *machine_ID*: The name or IP address of the TP's destination machine.
- *port_no:* The name of the TP's destination port number.

7 Enter the value for Destination URL. The URL syntax used for this parameter is:

`http://`**`<machine_ID><port_no>/<dep_profile>`**`_servlet_<`**`MSH_type>/`**   **`<MSH_type>`**

Where:

- *machine_ID*: The name or IP address of the TP's destination machine.
- *port_no:* The TP's destination machine port number.
- *dep_profile*: The current deployment profile name.
- *MSH_type*: The message service handler type, in this case, `ebXMLMSH`.

  For the sample, enter:

```
tpBerlin: http://<Atlanta_machine_ID>:<Atlanta_port_no>/dpInEbxmlBerlin_servlet_ebXMLMSH/ebXMLMSH

tpAtlanta: http://<Berlin_machine_ID>:<Berlin_port_no>/dpInEbxmlAtlanta_servlet_ebXMLMSH/ebXMLMSH
```

8 Leave the rest of the parameters blank.

9 When you are finished, click Save.

Note – Be sure to click **Save** after you are finished configuring the parameters on each tab.

---

10 Click the FromPartnerTransport tab.

The tab's parameters appear. These parameters allow you to supply information needed to transport data from the TP.

11 Enter the values under the FromPartnerTransport tab in the same way as you do the ToPartnerTransport tab (see steps earlier in this procedure). Keep in mind that these are the values that allow you to *send* data from the TP.

12 Click the ToPartner Packaging tab, and enter values for tpBerlin as shown in the following table.

Note – The parameters listed in ePM before **Soap Media Type** are standard eXchange parameters. See the **eXchange Integrator User's Guide.** for information on how to configure these parameters.

Parameter	Value
SOAP Media Type	Multipart/Related
Digital Envelope Protocol	http://www.w3.org/2001/04/xmlenc#
Non-repudiation Protocol	XMLDSIG

Parameter	Value
Signature Algorithm	http://www.w3.org/2000/09/xmldsig#rsa-sha1
Digest Method/HashFunction	http://www.w3.org/2000/09/xmldsig#-sha1
Canonicalization Method	http://www.w3.org/TR/2001/REC-xml-c14n-20010315
Encryption Algorithm	DES3
Signature Type	SoapEnveloped
Ack Signature Required	perMessage
Ack Request is Per Message	true
Duplicate Elimination is Per Message	true
SOAP Actor	urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH
Sync Reply	mshSignalsOnly
Language	en-us

13 Click the FromPartner Packaging tab and edit the parameters as listed in the following table.

Note – The parameters listed in ePM before **Signature Type** are standard eXchange parameters. See the **eXchange Integrator User's Guide** for information on how to configure these parameters.

Parameter	Value
Digital Envelope Protocol	http://www.w3.org/2001/04/xmlenc#
Non-Repudiation Protocol	XMLDSIG
Signature Type	SoapEnveloped
Ack Signature Required	perMessage
Ack Request is Per Message	true
Duplicate Elimination is Per Message	true
SOAP Actor	urn:oasis:names:tc:ebxml-msg:actor:toPartyMSH
Sync Reply	mshSignalsOnly
Language	en-us

14 **With the ToPartnerPackaging tab open, click Import (to the right of the Encryption Key text box) to import the current company's public certificate.**

The **Import a Certificate** dialog box appears. Import the certificate as follows:

- For **Certificate Name**, enter the appropriate certificate alias.
- For Import from file, click **Browse** and navigate to the current company's certificate file. See "Extracted Files and Directories" on page 73 for file names.

Note – If you are importing a certificate using ePM, you must apply the changes made during this step to the current Environment or restart the Logical Host. Otherwise, the certificate does not appear in the . **keystore** file.

---

15 **Repeat step 13 for the FromPartnerPackaging tab.**

16 **When you are completely finished, click Save.**

IDC Parameters

You can set the parameters governing the current company's internal message processing, when handling messages received and preparing messages to be sent. These properties are the IDC parameters.

Before you begin to configure these parameters, be sure to display the appropriate TP's name in the ePM **Explorer** (lower left side of the window).

Note – IDC bindings are optional and not required for an ebXML Protocol Manager Project.

This section explains how to set the **Internal Delivery Channel** parameters used for the current company's internal message processing.

Note – In the sample scenario, **tpBerlin** (for the Atlanta Company) has no IDCs.

▼ To configure the IDC parameters for the tpBerlin

1 **In the Explorer, click tpBerlin.**

The canvas displays the TP's general properties.

2 **Click the Components tab.**

The current TP's delivery channel parameters are displayed.

3 **Click the Internal Delivery Channels tab for the current TP.**

The IDC parameters appear. See the following tables.

Binding Name	idcAtlanta_ebXML_Send_via_File
Direction	Sender
Transport Name	JMS

Binding Name	idcAtlanta_ebXML_Rcv_via_File
Direction	Receiver
Transport Name	JMS

4 **To continue IDC configuration for tpBerlin, click the binding name for the first IDC (idcAtlanta_ebXML_Send_via_File), click the Sender Transport tab, and make changes as shown in the following table.**

FilePattern: *	*.txt
Directory: *	`C:\temp\eXchange\Sample\Data\Atlanta\OutboundebXML\`

5 **To finish IDC configuration for tpBerlin, click the binding name for the second IDC (idcAtlanta_ebXML_Rcv_via_File), click the Sender Transport tab, and make changes as shown in the following table.**

FilePattern: *	*.txt
Directory: *	`C:\temp\eXchange\Sample\OutputebXML\`

6 **When you are finished, click Save.**

Note – See “Increasing the Oracle Number of Processes” on page 29 and the **eXchange Integrator User’s Guide** for more information.

Messaging Actions Parameters

You must also configure ebXML Protocol Manager’s messaging actions under the **ePM Messaging Service Configuration**. These settings control messaging actions (inbound and outbound) for an ebXML MAD.

Configure the **Properties** and **Messaging Actions** for the TPs, using the parameters under these tabs, for the appropriate Messaging Service.

These configuration parameters define settings that allow ebXML Protocol Manager to set up, control, and order the sending and receiving of messages. For information on how to configure these parameters, see "Configuring Messaging Actions" on page 49.

TP Activation

This section explains how to activate a TP. You must save all the configuration information to the eXchange database instance to make it available at run time. Before you begin to activate a TP, the eXchange database instance for the corresponding B2B host must be running.

▼ To activate a TP

1. **In the Explorer (lower left) side of the ePM window, click tp Berlin or tpAtlanta .**
2. **In the bottom lower left of the canvas, click Activate.**
3. **In response to the confirmation prompt, click Activate.**

   The canvas displays the following confirmation message:

   Trading Partner is successfully activated

Initial Run-time Steps

For ebXML Protocol Manager sample Project implementation, initial run-time steps consist of:

- "Starting Logical Hosts" on page 97
- "Supplying Input Data" on page 99

Starting Logical Hosts

The procedures in this section assume you have installed two or more Logical Hosts, and that you have replaced the additional files as listed under the following sections:

- "Additional Policy .jar Files Required To Run SME" on page 25
- "Updating Files for Java Collaboration Definitions" on page 80

Applying Deployment Profiles to Logical Host

At this point, apply all Deployment Profiles to the appropriate Environment's Logical Host. Return to previous procedures in this chapter to troubleshoot any possible problems appropriately until all Deployment Profiles apply with no errors.

Running Logical Host

The directory for the Logical Host that runs the Atlanta Company is called **lhAtlanta** (associated with **envAtlanta**), and the directory for the Logical Host that runs the Berlin Company is called **lhBerlin** (associated with **envBerlin**).

▼ To run the current Logical Host for the first time

1. **Access a command prompt and change directories to the run executable file's location (for example, beginning with lhAtlanta) of the Logical Host:**

   ```
   cd lhAtlanta\bootstrap\bin
   ```

2. **Start the run script using the appropriate parameters, for example:**

   ```
   bootstrap- r http://<myMachine:12345>/<myRepository>- i <myID>
       -p <myPassword>- e envAtlanta- l lhAtlanta -f
   ```

   Where:

   - For the -r (Repository) parameter), supply the correct URL with the Repository name.
   - For the -i and -p (user ID and password) parameters, supply the appropriate values.
   - For -e (Environment name) parameter, use envAtlanta
   - For -l (Logical Host name) parameters, use lhAtlanta
   - You need not supply the -f (force) flag if all configuration changes have already been applied, or if this is the first time that the run script has been executed.

   After a brief time, the Logical Host starts running, and all activated Projects that reference **envAtlanta** are automatically applied to it.

3. **Repeat steps Step 1 and Step 2 on the envBerlin Logical Host, referencing the same Repository but pointing it at lhBerlin, for example:**

   ```
   cd lhBerlin\bootstrap\bin

   bootstrap- r http://<myMachine:12345>/<myRepository>- i <myID>
       -p <myPassword>- e envBerlin- l lhBerlin -f
   ```

▼ To run the Logical Host at later times

● **Enter the command bootstrap.**

Applying Changes Without Shutting Down Logical Host

This procedure is optional. You only need to use it if changes are made to parameters in an Environment while a Logical Host is running, for example, adding or modifying keystores for an eXchange service.

Use these steps to apply the changes without having to shut down and rerun the Logical Host. This procedure is the equivalent of shutting down the Logical Host and running it again using the -f flag.

▼ To apply Environment changes when a Logical Host is running

1. **In Enterprise Designer's Environment Explorer, open the Environment where the changes have occurred.**

2. **Right-click the Logical Host that is running and, on the context menu, click Apply.**

3. **Repeat the previous step, as needed, for other Logical Hosts in the same Environment.**

   In the Logical Host, a brief shutdown and restart occur, and the changes are applied to the running Logical Host.

Supplying Input Data

Before you begin, your Logical Hosts must both be running and the Oracle system used by the eXchange database instance must be accessible to eXchange.

The originating B2B host sends an outbound status request and receives an inbound status response. The originating ePM sends an outbound "ping" and receives an inbound "pong."

Both companies can trade data back and forth with one another, so you can reverse the inbound and outbound settings, depending on which company you want to be the sender and which the receiver. The sample's scenario (as supplied) is set up according to the procedure in this section.

See "Sample Data" on page 72 for a list of the sample scenario's data files and the directory locations created when the sample scenario's files are first extracted.

Note – You may have given different names to the `\temp\eXchange\` directories shown in the path locations listed under "Sample Data" on page 72. If so, substitute those names for the `\temp\eXchange\` directories given in this procedure.

▼ To send input data to the HostBerlin Project

1. **Before you start, change the .txt file extensions for your outbound data to .~in.**

2. **Copy the outbound data files to the following directory:**

   `C:\temp\eXchange\Sample\Data\Atlanta\OutboundebXML\`

3. **Make sure the file names of your outbound data (any of the files you want to send) have .txt file extensions. Any file with a .txt file extension changes to .~in as the file is picked up for delivery by eXchange.**

   This operation also copies the information in the payload data files and sends it along with the outbound files, so you see no change to the payload data file names. The Berlin Company returns the inbound data files to the following directory:

   `C:\temp\eXchange\Sample\Data\ToInternal`

   You can use the Message Tracking tool in eXchange to view the acknowledgement message. See the next section for information on how to use Message Tracking.

Using Message Tracking

eXchange provides a special feature, Message Tracking, allowing you to monitor the status of messages as they are received and processed through eXchange and ebXML Protocol Manager.

Before You Begin

- You must already have activated a project whose Connectivity Map contains one or more instances of the eXchange **Tracker_Application**.
- Your Oracle system for eXchange must already be running, and you must already have begun running a Logical Host before you can run this project.
- For Message Tracking to be useful, there must be one or more messages that have already been picked up by the current Logical Host’s Integration Server.

Accessing Message Tracking

This section explains how to access an instance of Message Tracking.

▼ To access Message Tracking

1 **Start a new browser session (that is, do *not* clone a window of an existing session).**

2 **Point your browser at the following URL:**

`http://<`**`loghostname`**`>:<`**`port`**`>/<`**`appname`**`>/msgTrack/EnterPkgTrack.do`

Where:

- *loghostname*: The host name or IP address of a Logical Host running your Project, that is, the current Logical Host.
- *port*: is The Web server connector port configured in your Integration Server. To discover this information, use **Environment Explorer** to open the current Logical Host. Right-click the Integration Server and select **Properties**. Open **IS Configuration > Sections > Web Container > Web Server > Default Web Server**; *port* is the value set for **Connector Port**. If you have several Web server configurations, check them also.

  The default port-number value is 18004 for the first Integration Server in the first-created Logical Host (or **19004** for the first Integration Server in the second-created Logical Host, and so on).
- *appname*: The name of the Message Tracking instance as it appears on the current Connectivity Map.

Example 5–1 Message Tracking Initial Window, on Startup

Example: To access Message Tracking for the ebXML Protocol Manager sample scenario, use the following URL:

```
http://localhost:18004/mtrk1/
```

As stated previously, the sample must be running before you access Message Tracking, and messages must have been passed before any become accessible. When you first run Message Tracking, its initial window appears.

Message Tracking Operation

This section explains how to use the Message Tracking feature.

▼ To search by B2B host, TP, and protocol

1 **Under Search Criteria, use the current B2B host's drop-down list to choose the B2B host whose messages you want to examine, then click GO.**

2 **Under Trading Partner, either click ALL or choose a particular TP from the drop-down list.**

3 **Under Protocols, either click ALL or choose a protocol from the drop-down list.**

4 **At the lower left of the window, click SEARCH.**

The canvas (right side), under **Search Results**, displays a page containing the Package IDs of the previous ten tracked messages fitting the criteria you specified. Navigation links (**Previous**, **Next**, and **Go to Page**) allow you to view other pages of ten results each.

▼ To search by B2B host, protocol, and Package ID

1 **Under Search Criteria, use the current B2B host's drop-down menu to choose the B2B host whose messages you want to examine, and click GO.**

2 **Under Protocols, either click ALL or choose a particular protocol from the list.**

3 **Under Package Type (to search by Package ID) either click ALL or choose a particular packaging protocol from the drop-down list. Perform the next steps:**

a. **Under ID, enter a string for matching the message ID.**

b. **At the lower left of the window, click SEARCH.**

The canvas displays a page containing the Package IDs of the previous ten tracked messages fitting the criteria you specified.

▼ To filter results by error type, direction, and/or date

● **After performing a search, or after setting up a search using either of the two previous procedures, you can specify one or more further search criteria.**

▼ To Specify Additional Search Criteria

1 **Near the bottom of the left pane, under Filters, specify one or more of the following characteristics:**

- Error Type: If you do not choose **ALL**, you can restrict your search either to display error messages only, or to display nonerror messages only.
- Direction: If you do not choose **ALL**, you can restrict your search either to display inbound messages only, or to display outbound messages only.
- Date: You can choose to include only those messages whose processing date exists within a range you specify, or only those messages whose acknowledgment date exists within the range.

2 **At the lower left of the window, click SEARCH.**

The canvas displays a page containing the Package IDs of the previous ten tracked messages fitting the criteria you specified.

On a package-by-package basis, you can also view the message text. After you view the text in a new window, you can preform cut, copy, and paste operations on any text in the window.

▼ To obtain details of a specified package

1 **After obtaining results from a search, using any of the procedures provided earlier, click the Package ID name for any of the returned results.**

2 **In the Details for package <*package-ID*> pane, click Open to view the contents of the current message in a new window.**

Note – Keep in mind that the current message could be encrypted, depending on your selection for the encryption parameters in the ePM configuration.

Glossary

AD, *AD, xAD	In eXchange an Attributes Definition defines the metadata attributes of parameters used in a business protocol, delivery protocol, or transport. Examples of xADs include: BPAD=BAD+EAD; DPAD=MAD+PAD; and TAD.
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.
B2B	Business-to-business (B2B) interactions are those that occur between business partners in the context of e-commerce.
BAD	In eXchange, Business Attribute Definitions (BADs) define the metadata attributes of message payload parameters used in business protocols such as X12, HIPAA, EDIFACT, or CIDX. Each BAD combines with one EAD to constitute a BPAD.
BPAD	In eXchange, Business Protocol Attribute Definitions (BPADs) define metadata for business protocols such as X12, HIPAA, EDIFACT, or CIDX. A BPAD consists of one Business Attributes Definition (BAD) and one Enveloping Attributes Definition (EAD).
CAPS	The Sun Java Composite Application Platform Suite (Java CAPS) includes eGate Integrator, eInsight Business Process Manager eXchange Integrator, eWay Intelligent Adapters, OTD Libraries, and Protocol Managers, as well as many other products.
CIDX	The Chemical Industry Data Exchange (CIDX) is a non-profit organization dedicated to improving the ease, speed and cost of securely conducting business electronically in the chemical industry. CIDX focuses on the development of eBusiness standards, called Chem eStandards.
DPAD	In eXchange, Delivery Protocol Attribute Definitions (DPADs) define metadata for delivery protocols such as AS2, ebXML, or RNIF. A DPAD consists of one Messaging Attributes Definition (MAD) and one Packaging Attributes Definition (PAD).
EAD	In eXchange, Enveloping Attribute Definitions (EADs) define the metadata attributes of message envelope parameters used in business protocols such as X12, HIPAA, EDIFACT, or CIDX. Each EAD combines with one BAD to constitute a BPAD.
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.
LDAP	The Lightweight Directory Access Protocol is a standard networking protocol for querying and modifying information stored as a distributed nonrelational database in directory servers (informally called “LDAP servers”) accessed via TCP/IP. Specifications for LDAP include RFCs 1777-1779 and 2251-2255.
MAD	In eXchange, Messaging Attribute Definitions (MADs) define the metadata attributes of messaging parameters used in delivery protocols such as AS2, ebXML, or RNIF. Each MAD combines with one PAD to constitute a DPAD.
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 [Please define the SuiteName0Short text entity], 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.
PAD	In eXchange, Packaging Attribute Definitions (PADs) define the metadata attributes of packaging parameters used in delivery protocols such as AS2, ebXML, or RNIF. Each PAD combines with one MAD to constitute a DPAD.
RNIF	The purpose of the RosettaNet Implementation Framework (RNIF) is to allow trading partners to configure their business processes in such a way as to operate with other trading partners adhering to the same framework, allowing electronic business transactions to be conducted securely over the Internet.
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.
SME	In [Please define the SuiteName0Short text entity], Secure Messaging Extensions (SME) uses advanced cryptographic techniques to ensure security, verifiability, and nonrepudiation of messages exchanged electronically.
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.
TCP/IP	The Transmission Control Protocol/Internet Protocol is a standard suite of communication protocols for connecting hosts and transmitting data over the Internet.
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.
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