

SeeBeyond ICAN Suite

eXchange Integrator User's Guide

Release 5.0.1



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

SeeBeyond, e*Gate, and e*Way are the registered trademarks of SeeBeyond Technology Corporation in the United States and select foreign countries; the SeeBeyond logo, e*Insight, and e*Xchange are trademarks of SeeBeyond Technology Corporation. The absence of a trademark from this list does not constitute a waiver of SeeBeyond Technology Corporation's intellectual property rights concerning that trademark. This document may contain references to other company, brand, and product names. These company, brand, and product names are used herein for identification purposes only and may be the trademarks of their respective owners.

© 2003 by SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

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

Version 20031118080709.

Contents

List of Figures	7
-----------------	---

List of Tables	10
----------------	----

Chapter 1

Introduction	11
Overview	11
Contents of This Guide	12
Writing Conventions	12
Additional Conventions	13
Supporting Documents	13
Online Documents	13
The SeeBeyond Web Site	13

Chapter 2

Overview	14
SeeBeyond Integrated Composite Application Network	14
ICAN Suite Components	15
Projects	15
Environments	15
eGate Integrator Components	15
Servers	16
Business Integration Applications	16
About eXchange Integrator	18
Architectural Overview	19
Overview of Features	20
Summary of Features	23

Chapter 3

Installing eXchange	24
System Requirements	24
Platform Support	24
Database Support	24
Database for eXchange Partner Management and Message Tracking	24
Database for eInsight Engine	25
Installation Steps	25
Uploading eXchange to the Repository	25
Refreshing Enterprise Designer with eXchange	27
Persistence — Overview	29
Creating and Configuring the eXchange Database Instance	29
Extracting and Customizing Database Scripts for eXchange	30
Running Database Scripts to Set Up the eXchange Database	30
Upgrading an eXchange 5.0 Database to 5.0.1	31
Configuring the eInsight Run-Time Engine	33
Extracting and Running Database Scripts for eInsight Engine	34

Chapter 4

Using eXchange in Enterprise Designer	36
Setting Up Protocol Attribute Definitions	36
Creating and Configuring Transport Attribute Definitions	36
Creating and Configuring Enveloping Attribute Definitions	39
Setting Up B2B Business Protocols	42
Setting Up B2B Hosts	45
Activating a B2B Host	51
Creating an Environment	52
Creating a Map with a B2B Host and a MessageTracker	53
Activating the B2B Host	54

Chapter 5

Designing B2B Protocol Pipelines	57
Overview	57
Building a B2B Protocol Pipeline	57
Modeling a Business Process in eXchange	58
eXchange Protocol Designer	58
Using the eXchange Protocol Designer GUI	59
Modeling Elements	60
Activity Elements	60
Branching Activities	62

Intermediate Events	63
Scope	64
While	64
Validating a B2B Protocol Pipeline	64
Saving an Unfinished B2B Protocol Pipeline	65
Using B2B Protocol Pipelines in a Connectivity Map	65
Deploying a Project With a B2B Protocol Pipeline	66
Deployment Profiles Containing Protocol Pipelines	66

Chapter 6

Exception Handling	68
Overview	68
Exception Handling Configuration	68
Identifying Component or System Failures	69
Scope	69
Scope or Process level exceptions	69
Compensation	70
Using Scope and Exceptions to Trigger Compensation	70
Validate the B2B Protocol Pipeline	70

Chapter 7

Using eXchange in Enterprise Manager	71
eXchange Partner Management (ePM)	71
Overview	71
Accessing eXchange Partner Manager (ePM)	71
Creating Trading Partners	72
Creating Bindings to External Delivery Channels	73
Creating Bindings to Internal Delivery Channels	75
Creating Profiles and Activating Trading Partners	76
Monitoring B2B Protocol Pipelines	79
Message Tracking	80
Accessing eXchange Message Tracking	80

Chapter 8

Implementation Scenarios	81
Overview	81
Configuring the B2B Host and Trading Partner Profiles	82
Setting Up the Environment	82
Setting Up Attribute Definitions for Protocols	83
Setting Up the Business Protocols	85
Setting Up the B2B Host	87

Contents

Setting Up Trading Partners for BorkHost Using ePM	99
AS2 Design-Time Scenario for Greenfield	110
Setting Up the Components	111
Using the Prebuilt AS2 Pipelines	119
Designing the Outbound Message Flow	124
Designing the Inbound Message Flow	127
Deploying Components to Servers in the Environment	133
Method Palette	136
Operators	136
String	139
Number	143
Boolean	145
Nodes	146

List of Figures

Figure 1	SeeBeyond Integrated Composite Application Network (ICAN) Suite	18
Figure 2	eXchange Architecture	20
Figure 3	B2B Host Designer in Enterprise Designer	21
Figure 4	Prebuilt B2B Protocol Pipeline (for AS2 Inbound) in Enterprise Designer	21
Figure 5	eXchange Trading Partner Configuration	22
Figure 6	eXchange Message Tracking	22
Figure 7	Enterprise Manager ADMIN page	26
Figure 8	Update Center Wizard: Select Modules to Install	27
Figure 9	Update Center Wizard: Progress Bars	28
Figure 10	Update Center Wizard: Restart Enterprise Designer	28
Figure 11	eInsight Engine Configuration	33
Figure 12	Business Property Sheet	35
Figure 13	Properties of Database Scripts	35
Figure 14	Transport Attribute Definition	37
Figure 15	Properties for Transport Attributes Definition	38
Figure 16	Attribute Definitions for a Custom Protocol	39
Figure 17	Extended Attribute Definitions for AS2 Version 1.1	41
Figure 18	B2B Business Protocol and Dialog Designer	43
Figure 19	Properties for B2B Business Protocol, Showing Enveloping Protocol Choices	43
Figure 20	B2B Business Protocol With Outbound and Inbound Activities	44
Figure 21	B2B Host Designer	45
Figure 22	Business Protocols Tree Organized by Enveloping Protocol	46
Figure 23	Business Protocols Tree Populated with Standard and Custom Protocols	47
Figure 24	External Delivery Channels Populated and Configured	47
Figure 25	Default Property Settings for External Delivery Channels – General	48
Figure 26	Default Property Settings for External Delivery Channels – Packaging	49
Figure 27	Default Property Settings for External Delivery Channels – Unpackaging	50
Figure 28	Default Property Settings for External Delivery Channels – Partner Transport	50
Figure 29	Internal Delivery Channels Populated and Configured	51
Figure 30	Environment Configuration for Oracle External System	52
Figure 31	Map Showing B2B Host and Message Tracker to be Deployed	53
Figure 32	Deployment Profile Showing B2B Host Being Assigned to Integration Server	54

Figure 33	Activated B2B Host in Environment and Deployment Profile	55
Figure 34	Importing a Certificate for a Delivery Channel in a B2B Host Environment	56
Figure 35	Importing a TrustStore for a Delivery Channel in a B2B Host Environment	56
Figure 36	Sample B2B Protocol Pipeline	58
Figure 37	B2B Protocol Pipeline	59
Figure 38	Toolbar Options	59
Figure 39	Selected Activity	61
Figure 40	Connectivity Map with B2B Protocol Pipeline	65
Figure 41	Connectivity Map: Protocol Pipeline Binding	66
Figure 42	Deployment Profile with Some Components Assigned	67
Figure 43	eXchange Partner Management (ePM) – No Configuration	72
Figure 44	New Trading Partner with Unique ID	73
Figure 45	Trading Partner Newly Associated with an External Delivery Channel	74
Figure 46	Trading Partner Newly Associated with an External Delivery Channel	76
Figure 47	Newly Created Trading Partner Profile	77
Figure 48	Business Protocols Organized Under Enveloping Protocols	77
Figure 49	Business Protocol in Explorer Tree with Actions Shown on Canvas	78
Figure 50	Monitor View	79
Figure 51	Preliminary Setup of BorkEnvironment	83
Figure 52	Properties of Transport Attribute Definition BorkFTP	84
Figure 53	Transport Protocol Attribute Definitions for BorkFTP	84
Figure 54	Business Protocol for BorkAS2Service	86
Figure 55	Project Tree with Both Business Protocols Checked In	87
Figure 56	B2B Host Designer	88
Figure 57	B2B Host with Enveloping Protocols	89
Figure 58	B2B Host with Business Protocols	89
Figure 59	Property Sheet for External Delivery Channel AS2_xdc_Greenfield	90
Figure 60	Connectivity Map for Preactivating BorkHost	92
Figure 61	BorkPreDeployment Profile Ready for Activation	93
Figure 62	BorkPreDeployment Profile After Activation	93
Figure 63	Import Certificate Dialog Box for Greenfield (AS2) signingCertificate	95
Figure 64	Imported “bork_key” signingCertificate	95
Figure 65	Import Certificate Dialog Box for signingTrustStore	96
Figure 66	Imported “seebeyondca” signingTrustStore	97
Figure 67	Selecting an Existing Certificate	98
Figure 68	BorkHost Channels Both Configured with Certificates and TrustStores	98
Figure 69	eXchange Partner Manager Showing BorkEnvironment and BorkHost1	99
Figure 70	New Trading Partner with Unique ID	100

Figure 71	General Properties of Binding for AS2_xdc_Greenfield	101
Figure 72	Importing a Certificate for Delivery Channel AS2_xdc_Greenfield	101
Figure 73	Packaging Values for Delivery Channel AS2_xdc_Greenfield	102
Figure 74	Associating a Trading Partner Profile With a Specific Business Protocol	104
Figure 75	Parameters for Service Actions TPPurchaseOrder, TPPurchaseOrderResponse	105
Figure 76	Confirmation Message Upon Successful TP Activation	105
Figure 77	General Properties of Binding for Custom_xdc_Johnson	106
Figure 78	Sample ToPartnerTransport Parameters for BorkFTP	107
Figure 79	Importing a Certificate for Delivery Channel Custom_xdc_Johnson	108
Figure 80	Associating a Trading Partner Profile With a Specific Business Protocol	109
Figure 81	Parameters for Service Actions PurchaseOrder and POResponse	110
Figure 82	Document Elements (Top Nodes) for the Six XSD-Based OTDs	112
Figure 83	The 29 EX* Table Records in the Oracle Database Instance for eXchange	113
Figure 84	Creating a Java Collaboration Definition with New Web Service Operation	115
Figure 85	Importing .jar Files Into a Java Collaboration Definition	116
Figure 86	Specifying Names and Order of .jar Files to Be Imported	116
Figure 87	Locating the Prebuilt AS2 Pipelines	119
Figure 88	Modifying the Operation of the StoreAS2ObjectServiceInvoke Activity	120
Figure 89	Successful Drag-and-Drop	121
Figure 90	Deleting Unused Attributes	121
Figure 91	Modifying AS2UnpackagerPipeline Activity “StoreAS2ObjectServiceInvoke”	122
Figure 92	Modifying the AS2MDNAssociationInvoke Activity in AS2UnpackagerPipeline	123
Figure 93	Modifying the StoreAS2ObjectInvoke Activity in AS2PackagerPipeline	123
Figure 94	Modifying the TPLookup Activity in AS2InboundBP	124
Figure 95	Outbound Connectivity Map With Three Externals	124
Figure 96	OutboundCMap with Four Protocol Pipelines Partly Linked	125
Figure 97	OutboundCMap with AS2PackagerPipeline Linked to Six JCDs	126
Figure 98	Fully Linked OutboundCMap	127
Figure 99	InboundCMap with Externals Linked to Protocol Pipelines and JCDs	129
Figure 100	InboundCMap Showing Pipelines and JCDs, Not Yet Linked	130
Figure 101	InboundCMap Showing Links To and From JCDs	131
Figure 102	Final Components and Links for InboundCMap	132
Figure 103	The 46 Greenfield Components Assigned to External Servers	135
Figure 104	Method Palette: Operator tab	136
Figure 105	Method Palette: String tab	139
Figure 106	Method Palette: Number tab	143
Figure 107	Method Palette: Boolean tab	145
Figure 108	Method Palette: Nodes tab	146

List of Tables

Table 1	Writing Conventions	12
Table 2	Properties for Transport Attribute Definition	37
Table 3	Properties for Enveloping Attribute Definition	40
Table 4	Properties for External Delivery Channels — General	48
Table 5	Properties for External Delivery Channels — Packaging/Unpackaging	49
Table 6	Activity Elements	61
Table 7	Branching Activities	63
Table 8	Intermediate Events	63
Table 9	Certificates and Trust Stores for External Delivery Channels	94
Table 10	How the Greenfield OTDs Are Used by Its Java Collaboration Definitions	114
Table 11	Operator Methods	136
Table 12	String Methods	140
Table 13	Number Methods	143
Table 14	Boolean Methods	145
Table 15	Nodes Methods	147

Introduction

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

In this chapter

- [Overview](#) on page 11
- [Contents of This Guide](#) on page 12
- [Writing Conventions](#) on page 12
- [Supporting Documents](#) on page 13
- [Online Documents](#) on page 13
- [The SeeBeyond Web Site](#) on page 13

1.1 Overview

The User's Guide provides instructions and background information for all users of the eXchange Integrator application. This guide is designed for managers, system administrators, and others who use eXchange Integrator.

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

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

1.2 Contents of This Guide

This guide is arranged as follows:

- **Chapter 1, “Introduction”** provides an overview of this document’s purpose, contents, writing conventions, and supported documents.
- **Chapter 2, “Overview”** discusses general features and architecture of eXchange.
- **Chapter 3, “Installing eXchange”** provides step-by-step instructions for installing the eXchange product and setting it up for use.
- **Chapter 4, “Using eXchange in Enterprise Designer”** provides step-by-step procedures for working with eXchange at design time and deploying projects.
- **Chapter 5, “Designing B2B Protocol Pipelines”** provides step-by-step procedures for designing and deploying protocol pipelines using eXchange Protocol Designer.
- **Chapter 6, “Exception Handling”** explains the use of protocol pipelines for handling exceptions.
- **Chapter 7, “Using eXchange in Enterprise Manager”** provides step-by-step procedures for working with eXchange’s Web-based GUIs—Partner Manager (ePM) and Message Tracking—as well as using Monitor for B2B protocol pipelines.
- **Appendix A, “Method Palette”** lists and describes the tools available to you in eXchange Protocol Designer.

1.3 Writing Conventions

The following writing conventions are observed throughout this document.

Table 1 Writing Conventions

Text	Convention	Example
Names of buttons, files, icons, parameters, variables, methods, menus, and objects	Bold text	<ul style="list-style-type: none"> ▪ Click OK to save and close. ▪ From the File menu, select Exit. ▪ Select the logicalhost.exe file. ▪ Enter the timeout value. ▪ Use the getClassname() method. ▪ Configure the Inbound File eWay.
Command-line arguments, code samples	Fixed font. Variables are shown in <i>bold italic</i> .	<code>bootstrap -p <i>password</i></code>
Hypertext links	Blue text	For more information, see “Writing Conventions” on page 12.

Additional Conventions

Windows Systems

For the purposes of this guide, references to “Windows” will apply to Microsoft Windows Server 2003, Windows XP, and Windows 2000.

1.4 Supporting Documents

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

Title	Filename
<i>SeeBeyond ICAN Suite Primer</i>	Primer.pdf
<i>eGate Integrator Release Notes</i>	eGate_Release_Notes.pdf
<i>eGate Tutorial</i>	eGate_Tutorial.pdf
<i>eGate Integrator Installation Guide</i>	eGate_Install_Guide.pdf
<i>eGate User’s Guide</i>	eGate_User_Guide.pdf
<i>eGate Integrator JMS Reference Guide</i>	eGate_JMS_Reference.pdf
<i>Oracle eWay Intelligent Adapter User’s Guide</i>	Oracle_eWay.pdf
<i>SeeBeyond ICAN Suite Deployment Guide</i>	Deployment_Guide.pdf
<i>eXchange Integrator Release Notes</i>	eXchange_Release_Notes.pdf
	readme.txt

1.5 Online Documents

The documentation for the SeeBeyond ICAN Suite is distributed as a collection of online documents. These documents are viewable with the Acrobat Reader application from Adobe Systems. Acrobat Reader can be downloaded from:

<http://www.adobe.com>

1.6 The SeeBeyond Web Site

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

<http://www.seebeyond.com>

Overview

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

In this chapter

- [SeeBeyond Integrated Composite Application Network](#) on page 14
- [About eXchange Integrator](#) on page 18
- [Architectural Overview](#) on page 19
- [Overview of Features](#) on page 20
- [Summary of Features](#) on page 23

2.1 SeeBeyond Integrated Composite Application Network

Businesses face tremendous challenges in managing dynamic partner relationships and the many processes that control inventory, absorb shifting transaction volumes, and help to transform raw data into capital. Among other things, organizations and their trading partners must manage disparate component applications and align proprietary software requirements while conforming to common data exchange and security standards.

The SeeBeyond® Integrated Composite Application Network (ICAN) Suite merges traditional Enterprise Application Integration (eAI) and Business-to-Business (B2B) interactions into a unified eBusiness infrastructure that connects, integrates, and optimizes data flow across the extended enterprise for customers, suppliers, and partners alike.

ICAN enables you to:

- Leverage your existing technology and applications.
- Create an application consisting of component applications that are managed by your organization for exchanging messages with your trading partners.
- Rapidly execute business strategies.
- Create and manage virtual organizations across the entire value chain.
- Rapidly implement industry-standard business protocols.
- Quickly and easily establish new business partners, or update existing ones.

- Secure transmissions sent through the public domain.

The ICAN Suite also provides:

- Extensive back-office connectivity.
- Powerful data transformation and mapping.
- Content-based routing.
- Unparalleled scalability based on a fully distributed architecture.

2.1.1. ICAN Suite Components

The ICAN Suite includes eGate™ Integrator components, together with servers and business integration applications.

Projects

SeeBeyond ICAN Suite components are used in Projects. A Project represents a logical solution for a business problem.

Environments

Project components are hosted by a collection of physical resources and their configurations, called an environment.

eGate Integrator Components

eGate Integrator (“eGate”) is an Enterprise Application Integration (eAI) platform that runs on a distributed and open architecture. Depending on the communications protocols and adapters you choose, eGate can communicate with and link multiple applications and databases across a variety of operating systems.

Enterprise Designer

Enterprise Designer is the primary application for configuring eGate to create Projects. It includes subsidiary applications such as a Connectivity Map Editor and Collaborations that you use to develop Projects.

eGate Enterprise Manager

eGate Enterprise Manager is a client-side web application that communicates with the Integration Server and is used to manage service status (stopped, running, starting, or stopping). Enterprise Manager generates “alert,” “event” and “trace” notifications.

Enterprise Manager also includes the JMS Administrator, which enables you to manage messages for both the SeeBeyond Message Server and the Nonstop Server for Java Message Service (NSJMS). With JMS Administrator, you can add, delete, refresh, connect to, or disconnect from message servers. You can also bind objects and show bindings.

Diagnostic tools

Diagnostic tools such as Impact Analyzer and Version Control are applications accessed through Enterprise Designer that can be used to examine individual Project components.

Servers

Servers handle back-end processing and storage of various kinds for ICAN Suite components.

SeeBeyond Integration Server

The SeeBeyond Integration Server is a J2EE-compliant software platform that provides application server features such as transaction services, persistence, load balancing, and external connectivity. Integration Server houses the business logic container used to run Collaborations, which are encoded business rules and the information that enables them to interact with other components.

SeeBeyond Security Server

SeeBeyond Security Server is a standalone server that enables disparate security frameworks to be integrated across an enterprise.

SeeBeyond Message Server for eGate

The message server is a JMS-compliant, guaranteed delivery, store, forwarding, and queueing service.

Business Integration Applications

Business integration applications are eGate add-ons that extend ICAN Suite functionality.

eWay Intelligent Adapters

eWay Intelligent Adapters (“eWays”) connect applications and databases with eGate, communicating with both external applications and message servers. When integrating different systems, the appropriate eWay on each end of the route provides the adaptation necessary for seamless flow of data.

eXchange Integrator

eXchange Integrator (“eXchange”) runs on the Integration Server, where it tracks messages and manages trading partner profiles through support for such business message format and enveloping protocols as AS2, and ebXML.

eTL Integrator

eXtract Transform and Load Integrator (“eTL”) is an integrated Collaboration editor used to perform high-volume, high-speed extractions, transformations, and loads, primarily from databases and flat files in tabular format. eTL Collaborations are deployed through eGate’s Enterprise Designer.

eInsight Business Process Manager

eInsight Business Process Manager (“eInsight”) facilitates the automation and administration of business process flow across business activities. Real-time graphical modeling and monitoring enables users to assess the state of a business process instance and identify any bottlenecks in that process.

eInsight ESB

eInsight ESB facilitates the automation and administration of business process flow across Web services (services written in the WSDL format of XML) to the exclusion of all other types of objects. eInsight ESB does not recognize eWays other than the SOAP (Simple Object Access Protocol) eWay.

eVision Studio

eVision Studio (“eVision”) is SeeBeyond’s Web application development product. eVision applications enable users to interact with eInsight activities through the ePortal Composer framework. eVision applications also enable users to send data to or receive data from eGate.

eView Studio

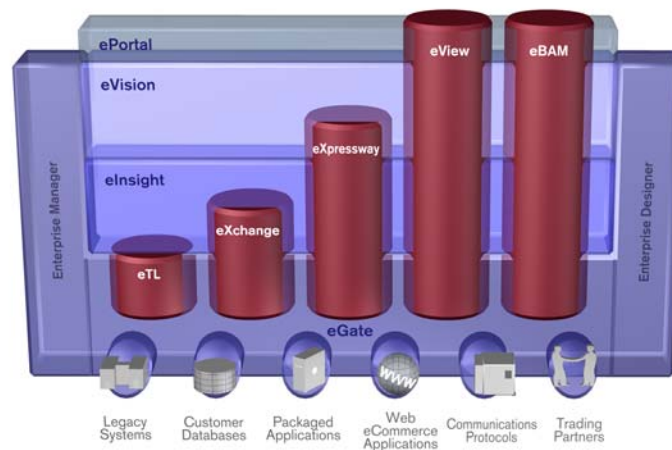
eView Studio (“eView”) is a plug-in for eGate, and an application-building tool that enables you to design, configure, and create a master index for uniquely identifying and cross-referencing the business objects (customers, vendors, members, hardware parts, and so on) stored in your system databases. Using eView, you can define the structure of your business objects and the logic that determines how business object data is updated, standardized, weighted, and matched in the master index database.

eIndex Global Identifier

eIndex Global Identifier (“eIndex”) is a highly customized instance of eView with added plug-in modules to facilitate person matching. eIndex automates identification and matching of member data across disparate source systems to simplify the process of sharing that data. eIndex creates a single view of all member data by providing a common identification process regardless of the system from which data originates.

ePortal Composer

ePortal Composer (“ePortal”) is a plug-in for eGate. ePortal enables people to interact with designated business processes over the Intranet or Internet.

Figure 1 SeeBeyond Integrated Composite Application Network (ICAN) Suite

For more information about the ICAN Suite, see the documentation associated with each product.

2.2 About eXchange Integrator

eXchange Integrator is an application that runs on the SeeBeyond Integration Server in distributed computing environments. eXchange manages interactions with trading partners by facilitating the reception, validation, transmission, and tracking of messages in supported formats.

Design-time, run-time, and administrative tools

eXchange provides three major sets of tools, for design-time, run-time, and monitoring.

- Integrated with the Enterprise Designer framework, eXchange's design-time tools, libraries, and prebuilt B2B protocol pipelines supplement and extend the eGate tool set with the following editors and components:
 - ♦ B2B Host Designer, along with Channel Manager, allow you to create and manage internal and external delivery channels tailored to the exact needs of each trading partner.
 - ♦ Protocol attribute definitions allow you to extend or modify standard transport and enveloping protocols.
 - ♦ The eXchange Protocol Designer is a powerful graphical modeling tool that lets you apply business logic to message flow through a B2B protocol pipeline.
 - ♦ The Dialog Designer, for editing B2B business protocols, helps you choreograph message-exchange interactions between your enterprise and trading partners.
- Running on the SeeBeyond Integration Server, eXchange's B2B Host, protocol pipelines, and message-tracker application enable your enterprise to:
 - ♦ Receive, validate, process, and route inbound and outbound messages.

- ♦ Look up trading partner configurations and access custom-tailored delivery channels that use agreed-upon protocols to envelope/de-envelope, encrypt/decrypt, and compress/decompress messages.
- ♦ Generate and reconcile acknowledgments; handle and report errors; and store message history, acknowledgments, and errors in a database.
- On the Web, plugged into the Enterprise Manager framework, eXchange's administrative and monitoring tools allow you to define and manage trading partner profiles, track message access, view message history, and record/monitor the status of completed and running B2B protocol pipelines.

2.3 Architectural Overview

eXchange uses the following key components:

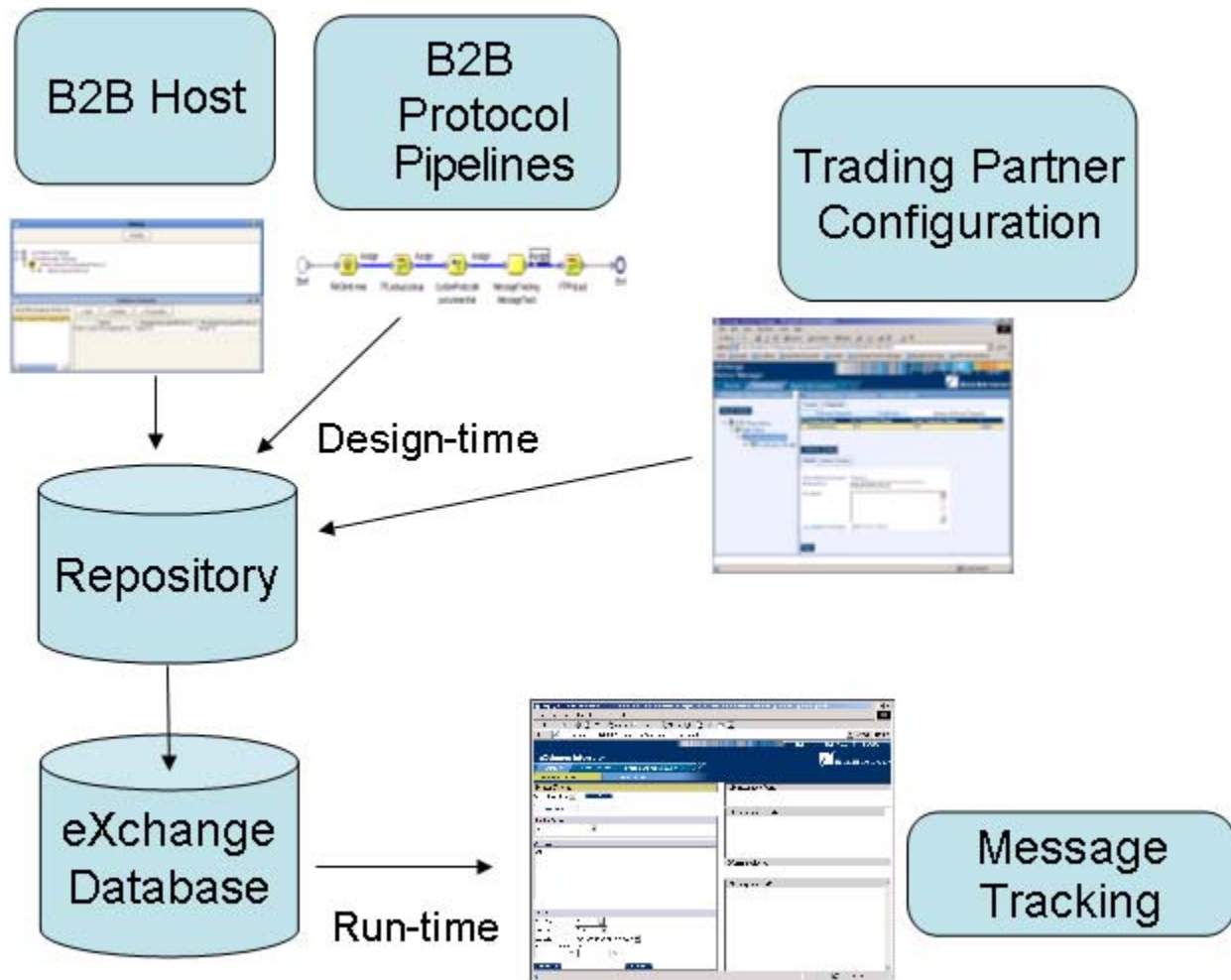
- **B2B Host Designer** — eXchange provides an additional editor within **Enterprise Designer** for setting up B2B environments. Each B2B Host provides one or more protocol-specific delivery channels that are exposed to the eXchange database via the Repository. Delivery channels defined within the B2B Host can then be accessed when specifying trading partner configurations. See [Figure 3 on page 21](#).
- **Enveloping Protocols and Transport Protocols** — eXchange supplies prebuilt B2B protocol pipelines for industry-standard protocols such as AS2 and ebXML, and it also provides the flexibility of allowing the enterprise to create and configure custom protocols and protocol pipelines. See [Figure 4 on page 21](#).

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

- **Trading Partner Configuration** — eXchange provides a Web-based GUI, eXchange Partner Management (ePM), for configuring and managing B2B trading partners. Each trading partner has one or more delivery channels that specify protocols to be used, with corresponding transport mechanisms—parameters such as encryption and signature, acknowledgment-handling preferences, and so forth. See [Figure 5 on page 22](#).
- **eXchange database** — eXchange uses an Oracle database to mediate retrieval of trading partner information and store run-time information for message tracking.
- **Message Tracking**— eXchange provides a MessageTracker application that can be combined with other processes in a project by dragging it into the connectivity map, as well as a Web-based message tracking GUI with powerful filtering and searching capabilities. See [Figure 6 on page 22](#).

The interaction of these key components is illustrated in Figure 2.

Figure 2 eXchange Architecture



2.4 Overview of Features

The illustrations below indicate some of the features provided by the various GUIs.

Figure 5 eXchange Trading Partner Configuration

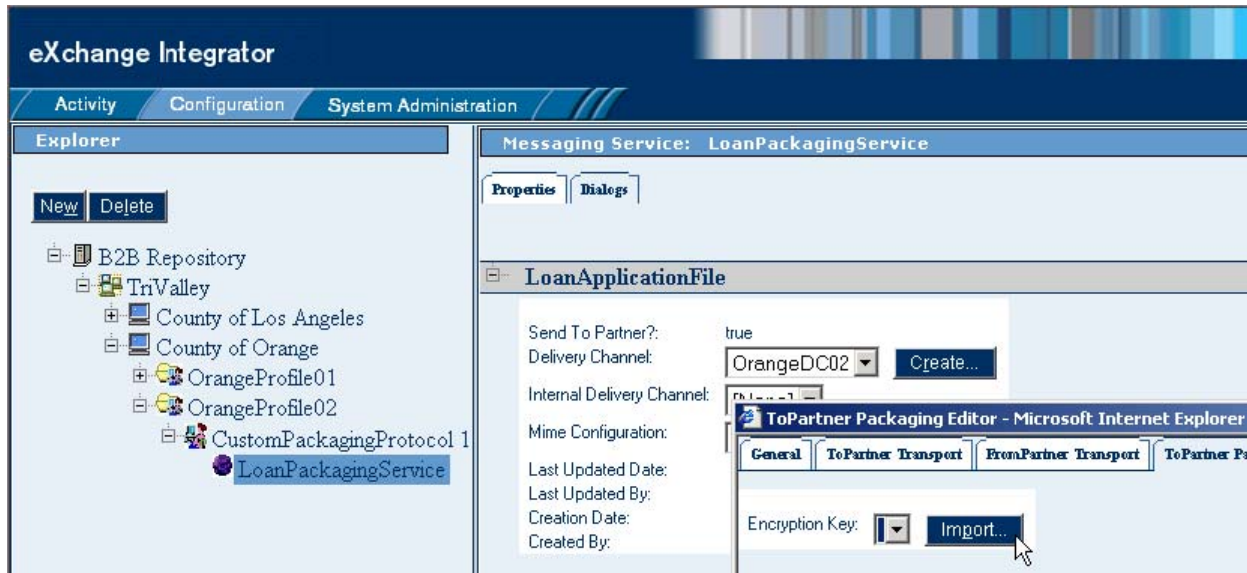
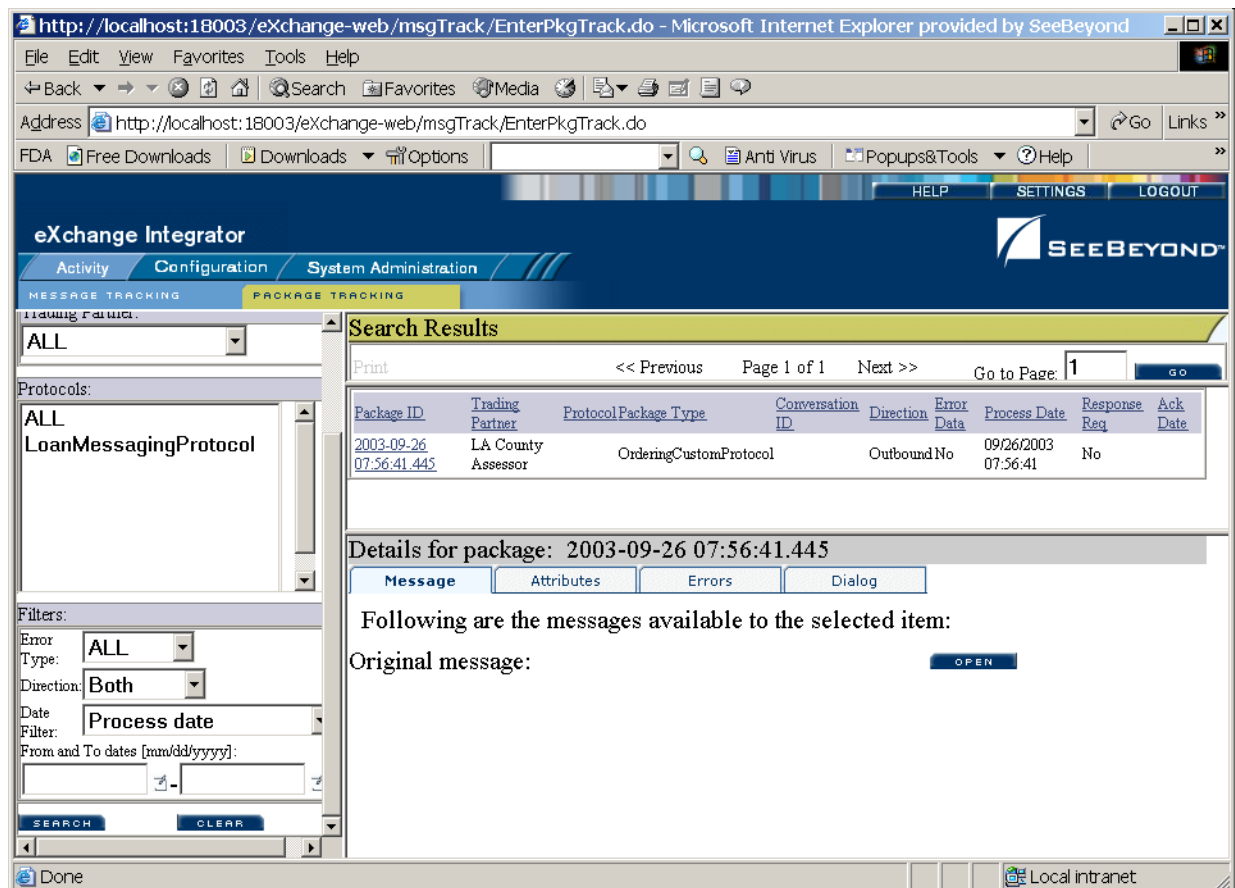


Figure 6 eXchange Message Tracking



2.5 Summary of Features

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

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

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

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

Chapter 3

Installing eXchange

This chapter provides the prerequisites and steps for installing eXchange Integrator.

3.1 System Requirements

This section lists system requirements and database requirements. The Readme.txt file on the product media contains the most up-to-date operating system requirements for each supported platform.

3.1.1. Platform Support

eXchange supports the following operating systems:

- Microsoft Windows Server 2003, Windows XP SP1a, and Windows 2000 SP3
- Sun Solaris 8 and 9 with required patches
- IBM AIX 5.1 and 5.2 with required maintenance level patches
- HP-UX 11.0 and 11i with required patches
- HP Tru64 V5.1A with required patches
- Red Hat Linux 8 (Intel) and Linux Advanced Server 2.1 (Intel version)

3.1.2. Database Support

Database for eXchange Partner Management and Message Tracking

The eXchange database provides a run-time persistent store for trading partner management and message tracking. eXchange supports the following databases:

- Oracle 8.1.7
- Oracle 9.01
- Oracle 9.2

Database for eInsight Engine

In addition, eXchange uses the eInsight engine (supplied with eXchange) to collect and persist data from your B2B protocol pipelines, so that Enterprise Manager can monitor business processes and B2B protocol pipelines. The eInsight engine supports the following databases:

- SQL Server 2000
- Sybase 12.5
- Oracle 8.1.7
- Oracle 9i

3.2 Installation Steps

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

3.2.1. Uploading eXchange to the Repository

Before you begin

- A Repository server must be running on the machine where you will be uploading the eXchange product files.
- The following ICAN products must have already been uploaded to this Repository:
 - ♦ eGate Enterprise Designer (eGate.sar)
 - ♦ Batch eWay adapter (BatcheWay.sar)
 - ♦ File eWay adapter (FileeWay.sar)
 - ♦ HTTP eWay adapter (HTTPeWay.sar)
 - ♦ Oracle eWay adapter (OracleeWay.sar)

To upload eXchange product files to the Repository

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

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

where

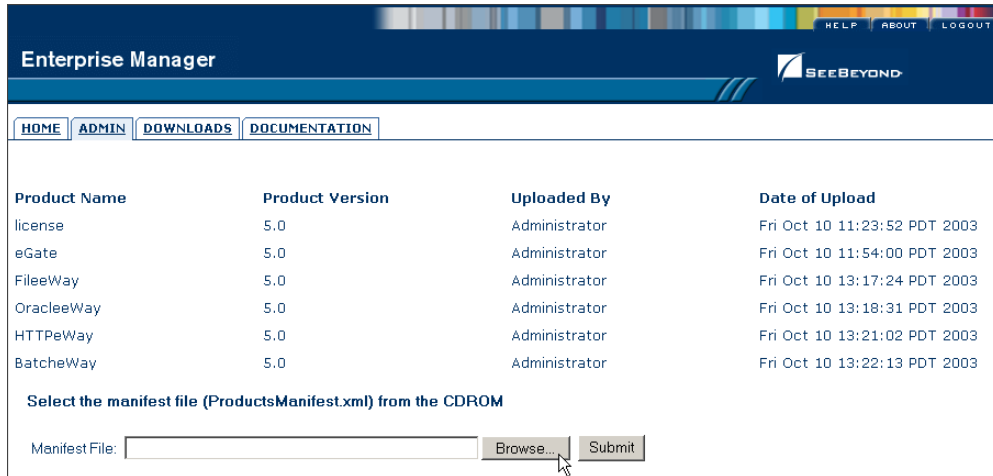
- ♦ <hostname> is the name of the machine running the Repository server.
- ♦ <port> is the starting port number assigned when the Repository was installed.

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

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

- 2 On the Enterprise Manager **SeeBeyond Customer Login** page, enter your username and password.
- 3 When Enterprise Manager responds, click the **ADMIN** tab. See **Figure 7 on page 26**.

Figure 7 Enterprise Manager ADMIN page



- 4 In the ADMIN page, click **Browse**.
- 5 In the **Choose file** dialog, click **ProductsManifest.xml**, and then click **Open**.
- 6 In the ADMIN page, click **Submit**.

The lower half of the ADMIN page lists the product files you are licensed to upload.

- 7 In the Products column, find **eXchange**, and then click the **Browse** button for it.
- 8 In the **Choose file** dialog, click **eXchange.sar**, and then click **Open**.
- 9 Repeat the previous two steps for other eXchange-related **.sar** files you are licensed to upload, such as SME Web Services (for **Secure Messaging Exchange**) or OTD libraries (such as for X12).

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

- 10 In the ADMIN page, click the **|upload now ::|** button.

3.2.2. Refreshing Enterprise Designer with eXchange

Before you begin

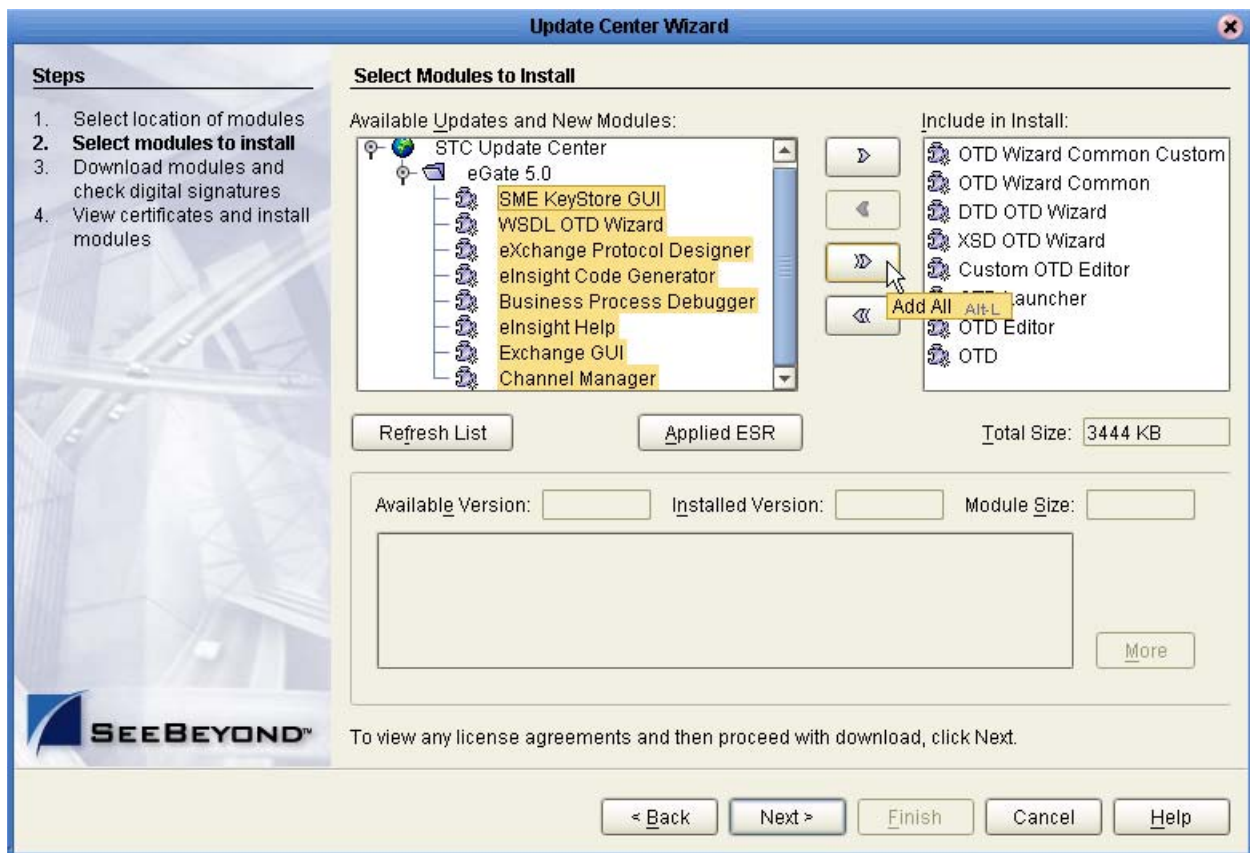
- You must have already downloaded and installed Enterprise Designer.
- A Repository server must be running on the machine where you uploaded the eXchange product files.

To refresh an existing installation of Enterprise Designer

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

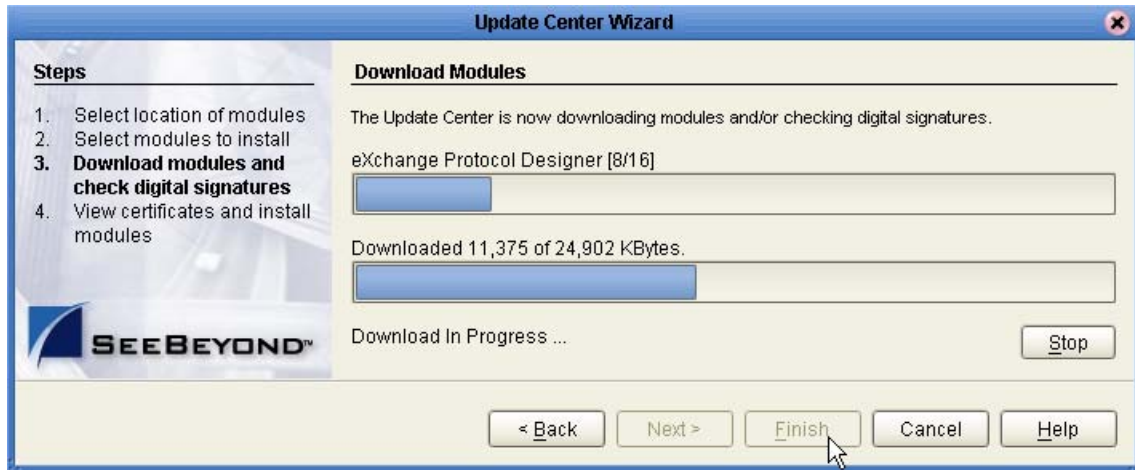
The Update Center shows a list of components ready for updating. See Figure 8.

Figure 8 Update Center Wizard: Select Modules to Install



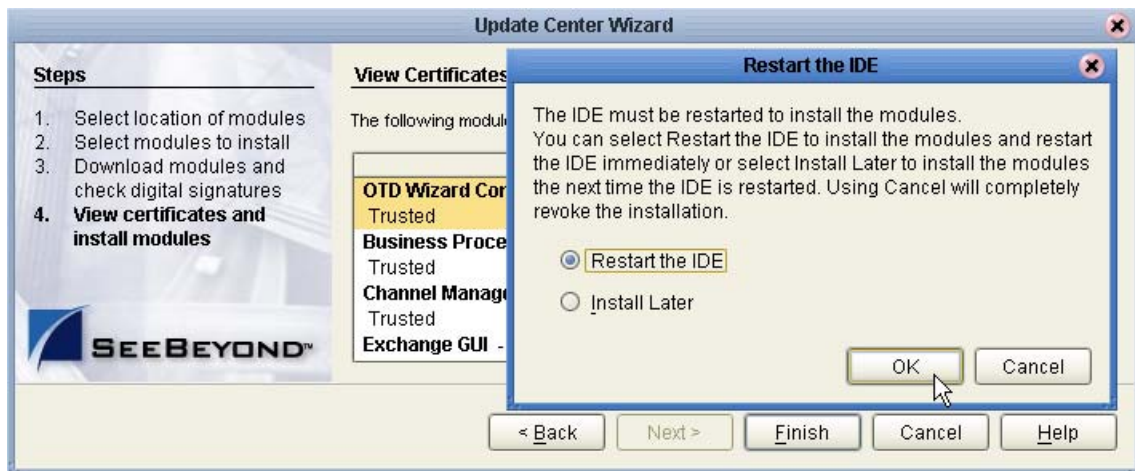
- 3 Click **Add All** (the button with a doubled chevron pointing to the right).
All modules move from the Available/New pane to the **Include in Install** pane.
- 4 Click **Next** and, in the next window, click **Accept** to accept the license agreement.
The wizard shows you the progress of the download. See Figure 9.

Figure 9 Update Center Wizard: Progress Bars



- 5 When the progress bars indicate the download has ended, click **Next**.
- 6 Review the certificates and installed modules, and then click **Finish**.
- 7 When prompted to restart Enterprise Designer, click **OK**. See Figure 10.

Figure 10 Update Center Wizard: Restart Enterprise Designer



When Enterprise Designer restarts, the installation of eXchange Integrator is complete, and you can use all eXchange tools provided on the Enterprise Designer framework. However, to take advantage of the Web-based GUIs for monitoring B2B protocol pipelines, you must also enable *persistence* for the eInsight engine.

3.3 Persistence — Overview

eXchange provides database scripts to create the database schemas for both the eInsight engine and eXchange itself.

- The eXchange database schema allows you to collect and persist data about your trading partner profiles, and also allows you to track message delivery history.

For eXchange, the areas to be configured are:

- ♦ [Creating and Configuring the eXchange Database Instance](#) on page 29
- ♦ [Extracting and Customizing Database Scripts for eXchange](#) on page 30
- ♦ [Running Database Scripts to Set Up the eXchange Database](#) on page 30

- The eInsight database schema allows you to collect and persist data from your B2B protocol pipelines; because the data is persisted, you can also use Enterprise Manager to monitor business processes and B2B protocol pipelines.

For eInsight, the areas to be configured are:

- ♦ [Configuring the eInsight Run-Time Engine](#) on page 33
- ♦ [Extracting and Running Database Scripts for eInsight Engine](#) on page 34

3.3.1. Creating and Configuring the eXchange Database Instance

Before you begin: You need to have already created an Oracle database instance with an entry in the `tnsnames.ora` file. Your TNSListener service must be running, and you will need to know the name of the database instance (default: **eXchange**) and to temporarily use the system username and password (default: **sys,manager**).

If you have never installed an Oracle database, ask your Oracle database administrator for help. The following constitutes a brief reminder of how to use the Oracle 9i wizard.

To create a new Oracle database instance

- 1 Step 1 (“Operations”): Choose **Create a database**.
- 2 Step 2 (“Database Templates”): Choose **New Database**.
- 3 Step 3 (“Database Identification”): Enter (for example) **eXchange**
- 4 Step 4 (“Database Features”): Deselect all checkboxes and reply **Yes** to all prompts.
- 5 Step 5 (“Database Connection Options”): Choose **Dedicated** [...].
- 6 Step 6 (“Initialization Parameters”): Keep all values unchanged.
- 7 Step 7 (“Database Storage”): Under Datafiles, click `\{DB_Name\}\undotbs01.dbf` (the fifth entry). In the **General** tab, reduce **File Size** from 200 to 100.
- 8 Step 8 (“Creation Options”): Choose **Create Database**, and then click Finish.

After this, to set up the eXchange database, you will extract the files supplied in the Database Scripts project folder, edit scripts so they have the correct parameters for your Oracle database setup, and run the scripts to set up and initialize the database.

Extracting and Customizing Database Scripts for eXchange

Note: *If you have already created and populated an eXchange database for 5.0, you will need to upgrade it to 5.0.1. Skip past this section, and continue to “[Upgrading an eXchange 5.0 Database to 5.0.1](#)” on page 31.*

To extract and customize the database scripts

- 1 In Enterprise Explorer, in the project tree, expand the following folders:
SeeBeyond > eXchange > DBScripts
- 2 Right-click each of the following files and, on the popup context menu, click **Export**; then use the **Save** dialog box to save each file to a local directory.
 - ♦ **createdb.sql**
 - ♦ **createtablespace.sql**
 - ♦ **createuser.sql**
 - ♦ **eXchange50Runtime.sql**
 - ♦ **in_user_seq.sql**
 - ♦ **x12_datamodel.sql**
- 3 If your Oracle location is not **c:\oracle\oradata**, or if your database instance name (SID) is other than **eXchange**, then open the **createtablespace.sql** file and make the appropriate change or changes in the first line.

Running Database Scripts to Set Up the eXchange Database

Important: *The database user who runs these .sql scripts must have permission to create tables.*

To run the database scripts that install the schema

- 1 Open a command prompt, change directories to the location where you saved the .sql scripts, and enter the following SQL*Plus command:

```
sqlplus system/<SYSTEMPWD>@<TNSNAME> @createtablespace.sql
```

where:

<SYSTEMPWD> is the password for the **system** login ID

<TNSNAME> is the name of the Oracle database instance you created for eXchange.

Here are two examples of valid commands, depending on the password and name:

```
sqlplus system/manager1@eX50 @createtablespace.sql  
sqlplus system/oraclePW@eXchange @createtablespace.sql
```

When this finishes, you will have created new tablespaces.

- 2 In the command prompt, enter the following SQL*Plus command:

```
sqlplus system/<SYSTEMPWD>@<TNSNAME> @createuser
```

where, as before, <SYSTEMPWD> is the password for the **system** login ID and

<TNSNAME> is the name of the Oracle database instance you created for eXchange.

Here is an example of a valid command:

```
sqlplus system/myPassWd@eX500DB @createuser.sql
```

- 3 In response to the system prompt, enter the username. For example: **ex_admin**
- 4 After the system finishes dropping tables, it asks for username and password so it can create the eXchange Administrator for the database instance). For example: **ex_admin ex_admin**
- 5 After the system finishes creating the eXchange Administrator, it asks a third time for the username, this time so it can grant resources and connect. Enter: **ex_admin** (or whatever you use as the username).
- 6 After running the previous two SQL script, there is one more. In the command prompt, enter the following SQL*Plus command:

```
sqlplus ex_admin/ex_admin@<TNSNAME> @createdb
```

where, as before, *TNSNAME* is the name of the eXchange Oracle database instance, and your eXchange administrator username and password are both **ex_admin**.

The system populates the tables, and you are now ready to use the database instance as your eXchange database. You can create Oracle OTDs based on this database, and use Enterprise Manager GUIs for trading partner configuration and message tracking.

Upgrading an eXchange 5.0 Database to 5.0.1

Note: The following steps should be performed only if you have a pre-existing eXchange database for 5.0.

To extract and customize the database scripts

- 1 In Enterprise Explorer, in the project tree, expand the following folders:
SeeBeyond > eXchange > DBScripts > Upgrade_500_to_501
- 2 Right-click each of the following files and, on the popup context menu, click **Export**; then use the **Save** dialog box to save each file to a local directory.
 - ♦ **copy_ex_dcp_service_from_temp.sql**
 - ♦ **copy_ex_dcp_service_to_temp.sql**
 - ♦ **eXchange_500_2_501_script.sql**
- 3 In a command prompt, change to the directory where you extracted these files, start a SQL*Plus session and log into the eXchange database instance. For example:

```
G:\> cd \ican50\dbscripts
G:\ican50\dbscripts> C:\oracle\ora92\bin\sqlplus
SQL*Plus: Release 9.2.0.1.0 - Production on Tue Nov 18 06:38:26 2003
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
Enter user-name: ex_admin
Enter password:
Connected to:
Oracle9i Enterprise Edition Release 9.2.0.1.0 - Production
With the Partitioning, OLAP and Oracle Data Mining options
JServer Release 9.2.0.1.0 - Production
SQL>
```

- 4 Execute the `eXchange_500_2_501_script.sql` script, wait for the prompt to reappear, and then exit. The console should resemble the following:

```
SQL> @eXchange_500_2_501_script.sql
Table altered.
Table altered.
Table altered.
1 row created.
1 row created.
Table created.
Input truncated to 1 characters
Procedure created.
PL/SQL procedure successfully completed.
2 rows deleted.
Commit complete.
Input truncated to 1 characters
Procedure created.
PL/SQL procedure successfully completed.
Table dropped.
Commit complete.
Procedure dropped.
Procedure dropped.
Table altered.
Table created.
Table altered.
5 rows updated.
Commit complete.
Table created.
Index created.
SQL> exit
```


3.3.2. Configuring the eInsight Run-Time Engine

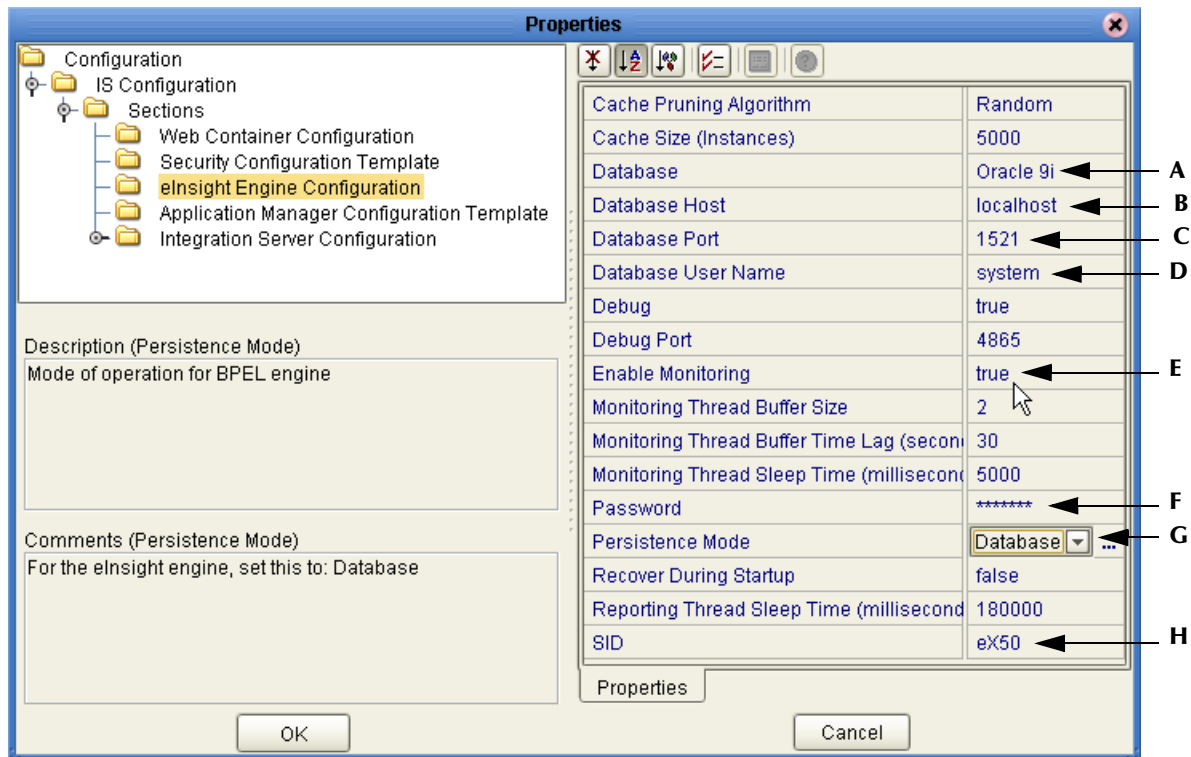
To configure the eInsight engine in an integration server

- 1 In Environment Explorer, open the environment and logical host, right-click the **Integration Server (IS)**, and, on the popup context menu, click **Properties**.

The Properties dialog opens.

- 2 Open the Configuration > IS Configuration > **Sections** tree and select **eInsight Engine Configuration**. See [Figure 11 on page 33](#).

Figure 11 eInsight Engine Configuration



- 3 Configure the following parameters appropriately for your database connection:

- A **Database**—Select one of Sybase 12.5, Oracle 8.1.7, SQL Server 2000, or Oracle 9i.
- B **Database Host**—Enter the name of the machine where your database resides.
- C **Database Port**—Enter your database connection port number (default=1521).
- D **Database User Name**—Enter the user name for your database.
- E **Enable Monitoring**—For persistence, change this from the default to: **true**.
- F **Password**—Enter the password for your database user.
- G **Persistence Mode**—For persistence, change this from the default to: **Database**.
- H **SID**—Enter the database name or SID.

Note: Leave other settings with their default values if you have no reason to override them.

3.3.3. Extracting and Running Database Scripts for eInsight Engine

To create the run-time recoverability database schema, extract and run the database scripts that are automatically installed with eXchange.

To extract the database schema scripts

- 1 In Enterprise Explorer, in the project tree, expand the following folders:
SeeBeyond > eInsight > Download Database Scripts
- 2 Right-click the **.zip** file associated with the appropriate database (oracle.zip, sqlserver.zip, or sybase.zip) and, on the popup context menu, click **Export**.
- 3 Save the **.zip** file to a local folder.
- 4 Extract the **.zip** file contents to a local folder, which will contain:
 - ♦ **install_db.bat** or **install.sh**—Creates the tablespace, users, tables, stored procedures, and any initial value.
 - ♦ **uninstall_db.bat** or **uninstall_db.sh**—Reverses what the **install_db** script creates; that is, it drops tables and users, and deletes stored procedures.
 - ♦ **clear_db.bat** or **clear_db.sh**—Truncates all tables without performing any other uninstall actions.
 - ♦ *database-specific .sql scripts*—Called by the **install_db.bat** and **uninstall_db.bat** commands. For example:, **create_tables.sql**, **drop_tables.sql**.
 - ♦ **Readme.txt**—Additional instructions specific to your database application.

To install the database schema

Important: *The database user that executes this script must have permission to create tables.*

- 1 Open a command prompt (on Windows) or (on UNIX) shell.
- 2 Change to the directory where the **install_db** script is located.
- 3 Enter the following command:

```
install_db <user> <password> <tnsname>
```

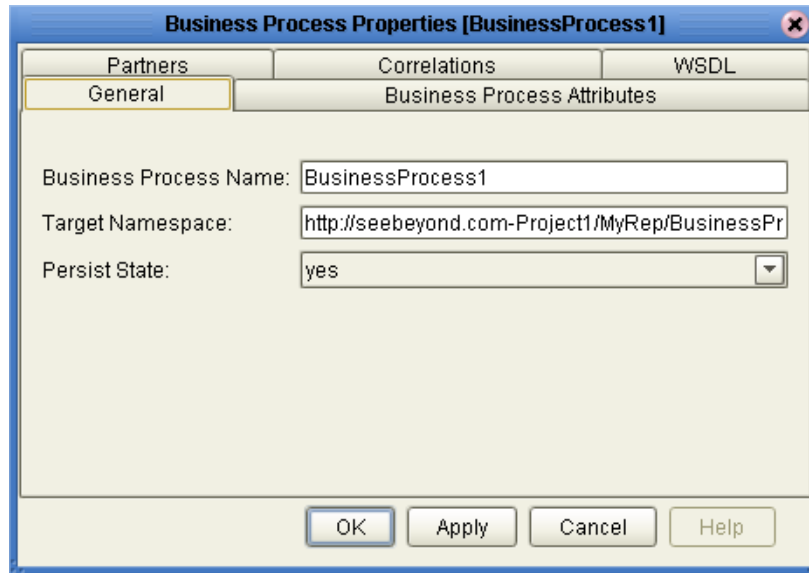
- ♦ <user> is the database username
- ♦ <password> is the password for this database user
- ♦ <tnsname> is the database or TNS name

Note: *The default user and password created from these scripts is: **einsight**. You can modify the user, password, and disk space allocated for tables and user permissions, but the table and column definitions should not be modified.*

To create specific tables for B2B protocol pipelines

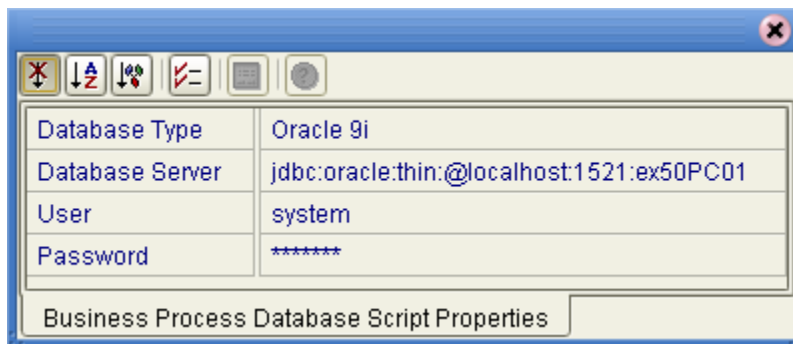
- 1 In Enterprise Designer, in the project tree, right-click the protocol pipeline and, on the popup context menu, click **Open Property Sheet**.
You set persistence state for individual protocol pipelines. The default setting is **no**.
- 2 Change the **Persist State** value to **yes**, as shown in Figure 12, and then click **OK**.

Figure 12 Business Property Sheet



- 3 On the main Enterprise Designer toolbar, click **Save All**.
The project tree displays a new **Database Scripts** under your protocol pipeline.
- 4 Right-click **Database Scripts** and fill in the correct information for your database from the dialog that appears (as shown in [Figure 13 on page 35](#)).

Figure 13 Properties of Database Scripts



To run the database script for a B2B protocol pipeline

- 1 In the project tree, open your protocol pipeline and its **Database Scripts** folder.
- 2 Right-click the appropriate database and, on the popup context menu, click **Run**.

The scripts complete the database creation process. Once persistence is configured, you can use the Enterprise Manager to monitor your protocol pipeline instances.

Using eXchange in Enterprise Designer

This chapter provides step-by-step procedures for using the components, tools, editors, and pre-supplied protocols and libraries provided by eXchange in Enterprise Designer.

In this chapter

- Steps to set up *attribute definitions* for transport and enveloping protocols
- Steps to set up *B2B business protocols*, which reference enveloping protocols
- Steps to set up *B2B hosts* (which reference B2B business protocols) and their *delivery channels* (which reference enveloping and transport protocols)
- Steps to preactivate a *B2B host environment* with MessageTracker, to set the stage for later activation of projects using Channel Manager.
- Steps to configure eXchange deployment profiles and environment components, such as the B2B host's certificates and truststores.

4.1 Setting Up Protocol Attribute Definitions

The section explains how to create and configure attribute definitions, and how to generate the corresponding OTDs, for the following project-level components:

- Transport attribute definitions
- Enveloping attribute definitions

4.1.1. Creating and Configuring Transport Attribute Definitions

In general, a *protocol* is a code of behavior; a framework for interpretation and communication that is agreed upon by all parties. It prescribes rules for interacting with others who are using the same protocol.

A *transport protocol* provides a way of specifying how data is to be delivered from one system to another. Some of the common standard transport protocols presupplied with eXchange include: FILE, FTP, HTTP, HTTPS, and SMTP. For example, FTP (file transfer protocol) requires the client to specify a transfer mode (such as ASCII or binary), a target directory, a target filename or file pattern, and so forth.

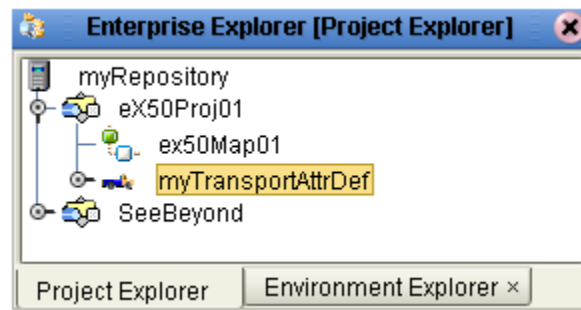
In addition to the standard transport protocols noted above, eXchange users can employ *transport attribute definitions* to create modifications or extensions of the standard transport protocols. These variants are called *custom* transport protocols.

To create a transport attribute definition

- 1 In Enterprise Designer with the Project Explorer tab active, in the project tree, right-click the project or subproject where the transport attribute definition will reside.
- 2 On the popup context menu, point at **New**, and then click **B2B Transport Attributes Definition**.

The project tree displays the new component, whose icon suggests a blue transport vehicle. Its default name is **B2BTransportAttributesDefinition<n>**, but you can rename it however you like.

Figure 14 Transport Attribute Definition



This new component is useful only by virtue of its configuration—you will need to add and define attributes that govern the nodes in the OTD that will be generated from it. Once attributes are defined, they can be exposed to eXchange Partner Management for delivery channel configuration.

To configure a transport attribute definition

- 1 In the project tree, right-click the transport attribute definition you want to modify.

Note: *If the component is locked, you must check it out before you can modify it.*

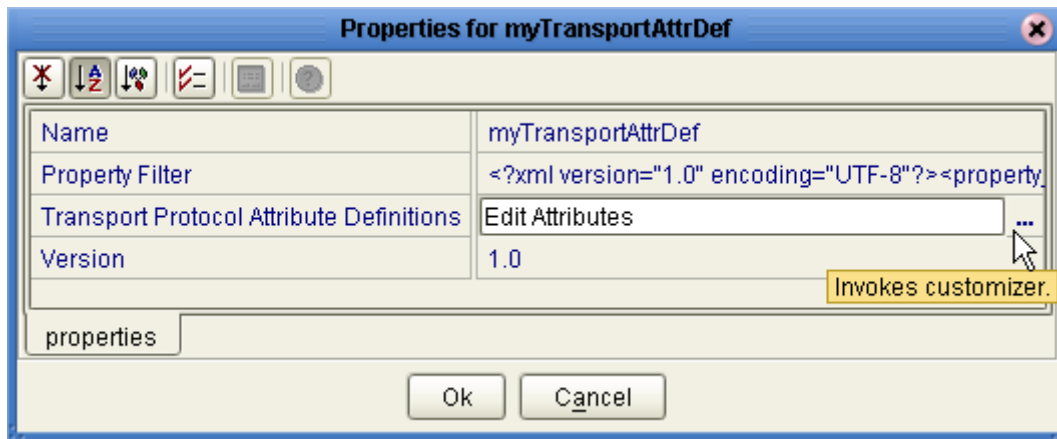
- 2 On the popup context menu, click **Properties**.

The properties dialog appears. See Table 2 and Figure 15.

Table 2 Properties for Transport Attribute Definition

Property Name	Initial or Default Value	Description
Name	(component name)	Free text; this is the component name displayed in the project tree.
Version	1.0	Free text; reflects whatever version naming conventions you use. This is independent of the Repository-assigned versioning, which tracks check-outs/check-ins.
Transport Protocol Attribute Definitions	null (initially)	An XML string that holds the values displayed in the Attribute Definitions dialog. To open the dialog, click the ellipsis [...] on the far right.
Property Filter	(XML string)	An XML string that holds the values displayed in the Property Filter dialog. To open the dialog, click the ellipsis [...] on the far right.

Figure 15 Properties for Transport Attributes Definition



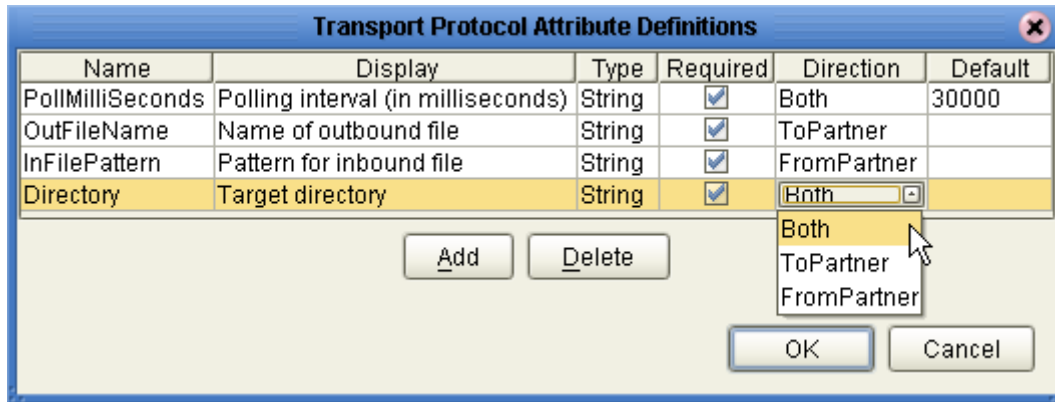
- 3 To the far right of the value for Transport Protocol Attribute Definitions, click the ellipsis [...].

The **Transport Protocol Attribute Definitions** dialog appears; see Figure 16. You use this dialog to create and set attributes. It is these attributes that determine the OTD nodes. The values in these nodes are subsequently used to dynamically configure the eWay properties. These values also govern the appearance and behavior of the parameters displayed in eXchange Partner Management when configuring external delivery channels for a trading partner (in the ToPartner Transport and FromPartner Transport subtabs).

- 4 Click the **Add** button as many times as needed and then, for each row created:
 - ♦ Change **Name** to a meaningful node name for the OTD you will generate.
 - ♦ Change **Display** to the text you want to display as a prompt or label for the parameter in eXchange Partner Management.
 - ♦ For **Required**, select or clear the box according to whether you want the parameter to be a required or optional entry. (In the ePM GUI, parameters that have been designated as required are flagged with a red asterisk.)
 - ♦ For **Direction**, choose ToPartner, FromPartner, or Both according to whether you want the parameter to appear with the ToPartner parameters, FromPartner parameters, or both.
 - ♦ For **Default**, you can optionally enter a default value that will appear in eXchange Partner Manager (ePM). This is the value that will be used if it is not overridden during parameter configuration in the ePM GUI.

Attributes for a sample file-based transport attribute definition are shown in Figure 16.

Figure 16 Attribute Definitions for a Custom Protocol



- 5 When you have finished adding and modifying attributes, click **OK**.
- 6 For **Property Filter**: If you want to change the packaging filters for this custom protocol (by default, both encryption truststore and signing certificates are filtered for both FromPartner bindings and ToPartner bindings), click the ellipsis at the far right and then, in the **Property Filter** dialog, clear the checkboxes for the filters you want disabled. When you have finished, click **OK**.
- 7 Click **OK** to close the properties dialog.

After you have completed these steps, the transport attribute definition will appear as a choice in the drop-down list of transport protocols when you configure the delivery channels of your B2B host, and it can now be used to generate an OTD. This OTD is used in the Business Rules Designer (lower pane of the eXchange Protocol Designer) so you can assign business logic from one activity to the next in your protocol pipeline.

To generate or refresh the OTD for a transport attribute definition

Before you begin: If the OTD has already been generated, make sure it is checked out.

- 1 In Enterprise Designer with the Project Explorer tab active, in the project tree, right-click the transport attribute definition whose OTD you want to generate or refresh.

Note: If the component is locked, you must check it out before you can modify it.

- 2 On the popup context menu, click **Generate OTD**.

The project tree displays a newly generated OTD. If the name of your transport attribute definition was *abcd*, the name of the new OTD will be *abcdOTD_abcd*.

4.1.2. Creating and Configuring Enveloping Attribute Definitions

An *enveloping protocol* (sometimes called a *packaging protocol* or *messaging protocol*) provides a way of specifying how data is bundled and unbundled—for example, it is at this level that encryption, acknowledgment, and nonrepudiation are addressed. SeeBeyond provides two standard enveloping protocols (AS2 and ebXML).

In addition to these standard enveloping/packaging protocols, eXchange users can employ *enveloping attribute definitions* to create modifications or extensions of the standard enveloping protocols. These variants are called *custom* enveloping protocols.

To create an enveloping attribute definition

- 1 In Enterprise Designer with the Project Explorer tab active, in the project tree, right-click the project or subproject where the enveloping attribute definition will reside.
- 2 On the popup context menu, point at **New**, and then click **B2B Enveloping Attributes Definition**.
- 3 In **Create B2B Enveloping Attributes Definition**, enter a name and click **OK**.

The project tree displays both the new component, whose icon suggests a brown paper package, or envelope. It also displays a new subproject; if the name of your enveloping attribute definition was *xyz*, the subproject will be named *xyz_Project* and will contain two B2B protocol pipelines: *xyz_FromPartner* and *xyz_ToPartner*. Do not alter these names.

This new component is useful only by virtue of its configuration—you will need to add and define attributes that govern the nodes in the OTD that will be generated from it.

To configure an enveloping attribute definition

- 1 In Enterprise Designer, with the Project Explorer tab active, right-click the enveloping attribute definition you want to modify.

Note: If the component is locked, you must check it out before you can modify it.

- 2 On the popup context menu, click **Properties**. See Table 3.

Table 3 Properties for Enveloping Attribute Definition

Property Name	Initial or Default Value	Description
Name	(component name)	Free text; this is the component name displayed in the project tree.
Version	1.0	Free text; reflects whatever version naming conventions you use. This is <i>independent</i> of the Repository-assigned versioning, which tracks check-outs/check-ins.
Protocol Attribute Definitions	null (initially)	An XML string that holds the values displayed in the Protocol Attribute Definitions dialog. To open the dialog, click the ellipsis [...] on the far right.
Enveloping Pipeline	null	(not yet supported)
Deenveloping Pipeline	null	(not yet supported)
Property Filter	(XML string)	An XML string that holds the values displayed in the Property Filter dialog. To open the dialog, click the ellipsis [...] on the far right.

- 3 To the far right of the value for **Protocol Attribute Definitions**, click the ellipsis [...] on the far right.

The **Protocol Attribute Definitions** dialog appears. You use this dialog to create and set the attributes; these attributes determine the OTD nodes. The values in these nodes are subsequently used to provide the trading partner–specific values needed for a message envelope. These values also govern the appearance and behavior of the parameters displayed in ePM when configuring external delivery channels for a trading partner (the ToPartner Transport and FromPartner Transport subtabs).

- 4 Click the **Add** button as many times as needed and then, for each row created:
 - ♦ Change **Name** to a meaningful node name for the OTD you will generate.
 - ♦ Change **Display** to the text you want to display as a prompt or label for the parameter in eXchange Partner Management.
 - ♦ For **Required**, select or clear the box according to whether you want the parameter to be a required or optional entry. (In the ePM GUI, parameters that have been designated as required are flagged with a red asterisk.)
 - ♦ For **Direction**, choose ToPartner, FromPartner, or Both according to whether you want the parameter to appear with the ToPartner parameters, FromPartner parameters, or both.
 - ♦ For **Default**, enter the default value you want this parameter to assume if it is not changed in trading partner management.

Figure 16 on page 39 showed an example of extended attributes for a custom protocol; as a different example, the extended attributes of a standard enveloping protocol (in this case, AS2 version 1.1) are shown in Figure 17.

Figure 17 Extended Attribute Definitions for AS2 Version 1.1

Name	Display	Type	Req...	Direction	Default
AS2_FROM	AS2_FROM	String	<input checked="" type="checkbox"/>	Both	
AS2_TO	AS2_TO	String	<input checked="" type="checkbox"/>	Both	
AS2_HOST	AS2_HOST	String	<input checked="" type="checkbox"/>	ToPartner	
AS2_VERSION	AS2_VERSION	String	<input checked="" type="checkbox"/>	ToPartner	1.1
HTTP_FROM	AS2_HTTP_FROM	String	<input checked="" type="checkbox"/>	ToPartner	
SIGNATURE_REQ	AS2_SIGNATURE_REQ	Boolean	<input checked="" type="checkbox"/>	ToPartner	TRUE
ENCRYPT_REQ	AS2_ENCRYPT_REQ	Boolean	<input checked="" type="checkbox"/>	ToPartner	TRUE
MDN_REQ	AS2_MDN_REQ	Boolean	<input checked="" type="checkbox"/>	ToPartner	
MDN_SIGNATURE_REQ	AS2_MDN_SIGNATURE_REQ	Boolean	<input checked="" type="checkbox"/>	ToPartner	TRUE
MDN_RESP_TYPE	AS2_MDN_RESP_TYPE	String	<input checked="" type="checkbox"/>	ToPartner	ASYN
MDN_DELIVERY_URL	AS2_MDN_DELIVERY_URL	String	<input type="checkbox"/>	ToPartner	
AS2_SUBJECT	AS2_SUBJECT	String	<input checked="" type="checkbox"/>	ToPartner	
COMPRESSED	AS2_COMPRESSED	Boolean	<input checked="" type="checkbox"/>	ToPartner	TRUE
COMPRESSED_BEFORE_SIGNED	AS2_COMPRESSED_BEFORE_SIGNED	Boolean	<input checked="" type="checkbox"/>	ToPartner	FALSE
PAYLOAD_TYPE	AS2_PAYLOAD_TYPE	String	<input checked="" type="checkbox"/>	ToPartner	x12
MESSAGE_FORMAT	AS2_MESSAGE_FORMAT	String	<input checked="" type="checkbox"/>	ToPartner	SMIME
ENCODING	AS2_ENCODING	String	<input checked="" type="checkbox"/>	ToPartner	base64
REPORTING_UA	AS2_REPORTING_UA	String	<input checked="" type="checkbox"/>	FromPartner	
POSITIVE_MDN_DISPOSITION_MESSAGE	AS2_POSITIVE_MDN_DISPOSITION_MESSAGE	String	<input type="checkbox"/>	FromPartner	positive disposition message

- 5 When you have finished adding and modifying attributes, click **OK**.

- 6 For **Property Filter**: If you want to change the packaging filters for your enveloping attribute definition (by default, all truststores and certificates are filtered for both FromPartner bindings and ToPartner bindings), click the ellipsis at the far right and then, in the **Property Filter** dialog, clear the checkboxes for the filters you want disabled. When you have finished, click **OK**.
- 7 Click **OK** to close the properties dialog.

The enveloping attribute definition can now be used to generate an OTD.

To generate or refresh the OTD for an enveloping attribute definition

Before you begin: If the OTD has already been generated, be sure it is checked out.

- 1 In the project tree, right-click the enveloping attribute definition whose OTD you want to generate or refresh.

Note: *If the component is locked, you must check it out before you can modify it.*

- 2 On the popup context menu, click **Generate OTD**.

The project tree displays a newly generated OTD. If the name of your enveloping attribute definition was *abcd*, the name of the new OTD will be *abcdOTD_abcd*.

After you have completed these steps, the enveloping attribute definition will appear as a choice in the drop-down list of enveloping protocols when you configure the delivery channels of your B2B host.

4.2 Setting Up B2B Business Protocols

A B2B business protocol is, in brief, a project-level (logical) component that sets forth rules of exchange between an enterprise and its trading partners.

The section explains how to create, populate, configure, and compile a B2B business protocol, using the **Dialog Designer** editor. These steps must be followed to allow the B2B protocol to be used by a B2B host.

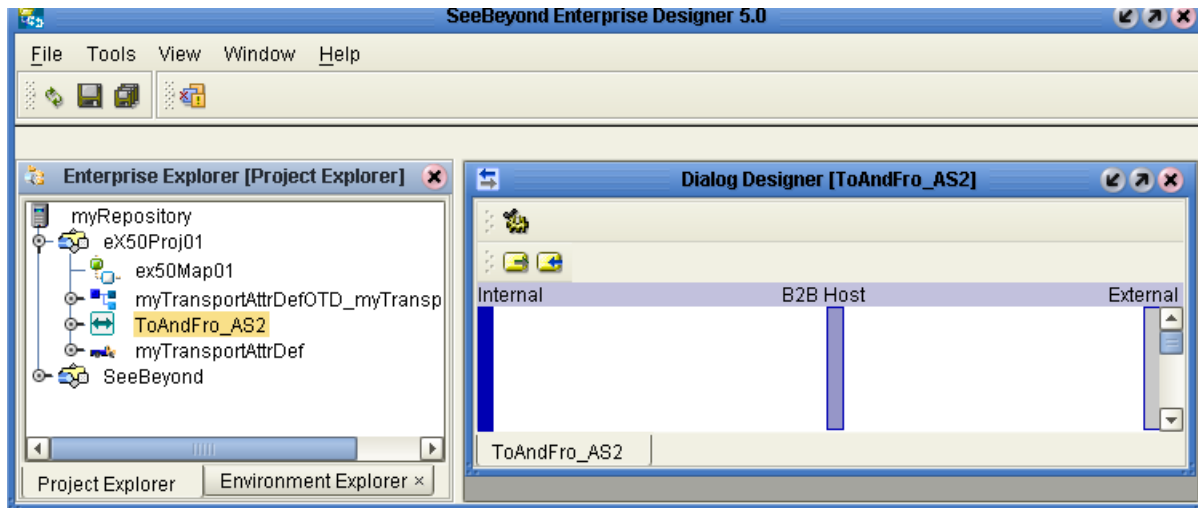
To create a B2B business protocol

- 1 In Enterprise Designer with the Project Explorer tab active, in the project tree, right-click the project or subproject where the B2B business protocol will reside.
- 2 On the popup context menu, point at **New**, and then click **B2B Business Protocol**.

The project tree displays the new component, whose icon suggests bidirectional communication. By default, it is named **B2B Business Protocol<n>**, but you can rename it as you like.

The Dialog Designer opens to display a blank canvas. See Figure 20.

Figure 18 B2B Business Protocol and Dialog Designer

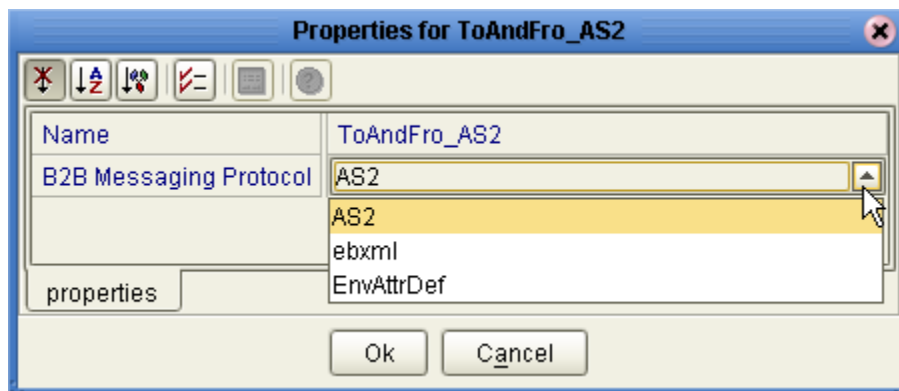


This new component still needs to be configured, populated, and compiled—you will need to add activities that describe the interchange of messages to and from the trading partner, configure it to reference an enveloping protocol, and then compile it.

To configure, populate, and compile a B2B business protocol

- 1 In the project tree, right-click the B2B business protocol and, on the popup context menu, click **Properties**.
- 2 In the Properties dialog, select an enveloping protocol from the choices displayed in the drop-down list. This list will include all standard enveloping protocols supplied by SeeBeyond (such as AS2 and ebXML), and will also show you any attribute definitions that have been created for custom enveloping protocols. See Figure 19.

Figure 19 Properties for B2B Business Protocol, Showing Enveloping Protocol Choices



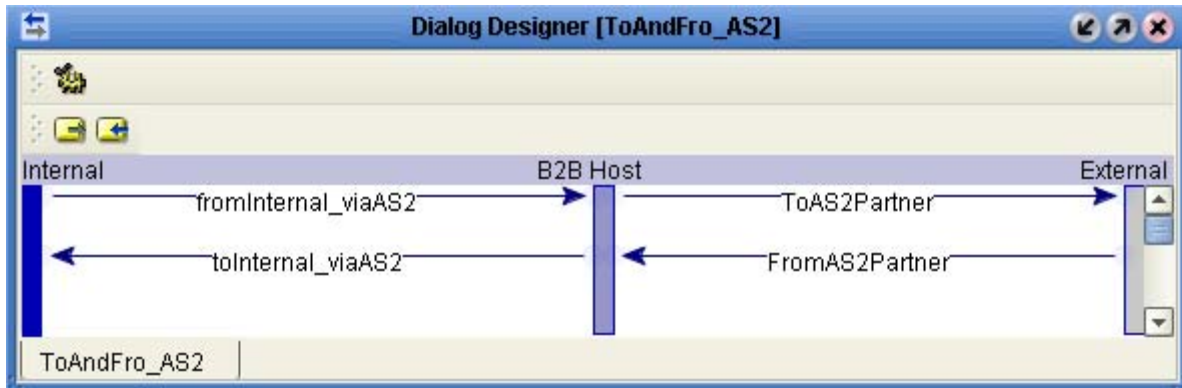
- 3 When you have chosen an enveloping protocol, click **OK**.
- 4 If Dialog Designer is not already open, open it by double-clicking the B2B business protocol you want to populate.
- 5 Do at least one of the following:
 - ♦ Click **from_internal** (the right-pointing arrow icon) to add an activity that originates in the enterprise and passes through the B2B host to the trading

partner. Then, double-click, the label on each of the two right-pointing arrows, editing the names to whatever you want.

- ♦ Click **message_in** (the left-pointing arrow icon) to add an activity that originates in the trading partner and passes through the B2B host to the enterprise. Then, double-click the label on each of the two left-pointing arrows, editing the names to whatever you want.

6 Optionally, add other activities as needed. See Figure 20.

Figure 20 B2B Business Protocol With Outbound and Inbound Activities



Depending on the complexity of interaction between the enterprise and the trading partner, a B2B business protocol might have only one activity, or five fromInternal-toPartner activities followed by one fromPartner-toInternal reply, or an interleaved series of to-and-fro exchanges. The characteristic quality of a B2B business protocol is that each activity passes through the B2B host; there is no separate back-and-forth interactive communication between just the B2B host and either side.

7 Click **compile** (button in the upper left; the icon suggests a gear with a checkmark).

Note: *You cannot compile a B2B business protocol that does not reference a valid enveloping protocol; see steps 1 through 3 in this procedure.*

Once the B2B business protocol is successfully compiled, the project tree treats it as a container of the activities you modeled in the Dialog Designer. Although these activities can be referenced, they are not web service operations, and are therefore ineligible for dragging onto canvases. These are also the activities that appear at the lowest level of the tree in eXchange Partner Manager.

4.3 Setting Up B2B Hosts

A B2B Host is, in brief, both a logical component that oversees delivery channels and also an environment component that is exposed to web GUIs such as eXchange Partner Management (ePM) and message tracking. It provides bindings for Channel Managers.

The section explains how to create, populate, configure, and preactivate a B2B Host. These steps must be followed before Channel Manager can be activated, and before eXchange Partner Management (ePM) can be used for trading partner configuration.

The editor used for setting up B2B hosts is the **B2B Host Designer**.

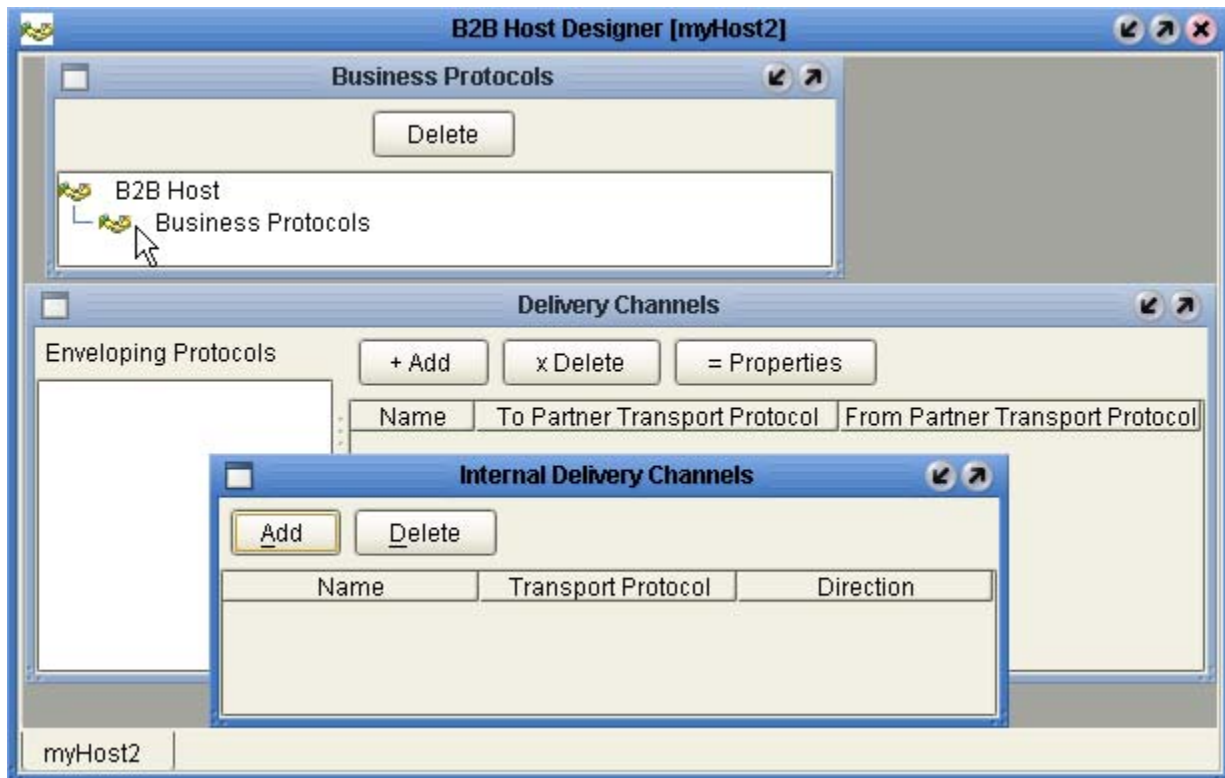
To create a B2B Host

- 1 In Enterprise Designer with the Project Explorer tab active, in the project tree, right-click the project or subproject where the B2B host will reside.
- 2 On the popup context menu, point at **New**, and then click **B2B Host**.
- 3 In the **Create B2B Host** dialog, enter a name for the host, and then click **OK**.

The project tree displays the new component, whose icon suggests two partners shaking hands.

The B2B Host Designer opens to display three blank windows: Business Protocols, Delivery Channels, and Internal Delivery Channels. See Figure 21.

Figure 21 B2B Host Designer



- The **Business Protocols** window maintains a tree of Business Protocols that are exposed to the B2B host. These protocols are not added directly to the **Business Protocols** branch of the tree, but are instead organized by enveloping protocol.
- The **Delivery Channels** window lists these enveloping protocols in its left pane; the right pane lists delivery channels defined for the highlighted enveloping protocol, and allows you to add, delete, and modify the properties of those delivery channels.
- The **Internal Delivery Channels** window lists names, transport protocols, and direction (Sender=ToInternal; Receiver=FromInternal), and allows you to add and delete IDCs.

To add B2B business protocols to a B2B host, organized by enveloping protocol

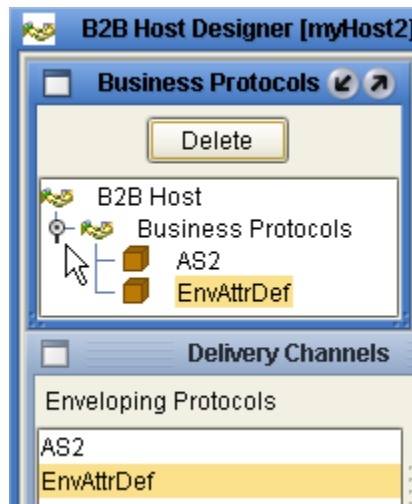
Before you begin: The B2B Host Designer must be open and the project tree available.

- 1 Into the **Business Protocols** window, under the Business Protocols branch, drag in an enveloping protocol (either custom or standard) from the project tree.

Note: Standard enveloping protocols are located under the *SeeBeyond* project folder; each resides at the bottom of its corresponding eXchange > **MessagingProtocols** folder.

The enveloping protocol appears both in the business protocols tree and in the Enveloping Protocols pane of the Delivery Channels window. See Figure 22.

Figure 22 Business Protocols Tree Organized by Enveloping Protocol

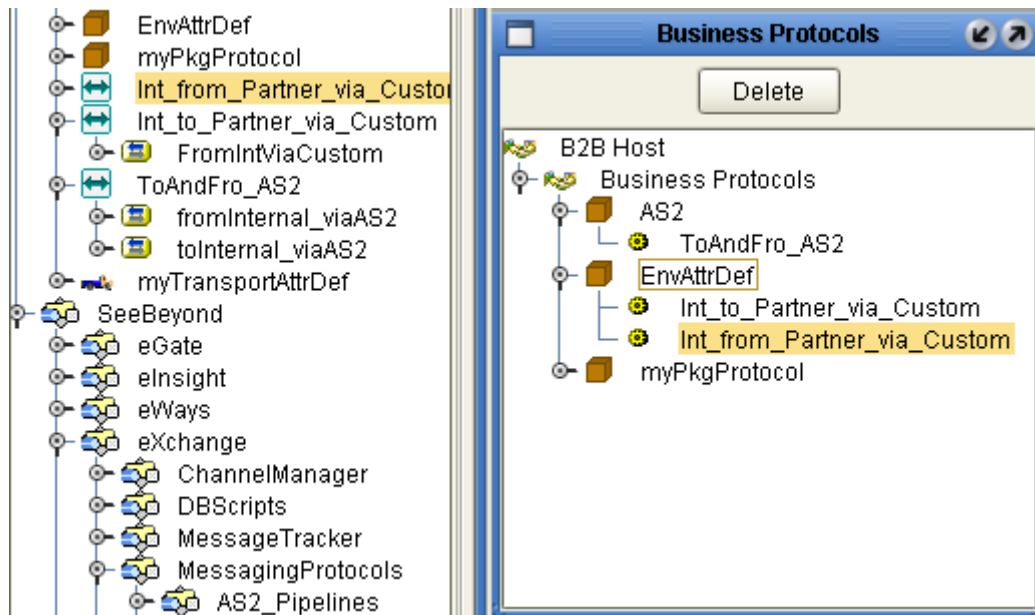


- 2 Under this enveloping protocol, drag in a B2B business protocol whose properties reference it. (The GUI does not allow you to drag in a B2B business protocol that references a different enveloping protocol, or no protocol.)

The B2B business protocol appears in the business protocols tree, subordinate to the enveloping protocol it references.

- 3 Repeat the previous step as needed, or repeat all the previous steps. See Figure 23.

Figure 23 Business Protocols Tree Populated with Standard and Custom Protocols



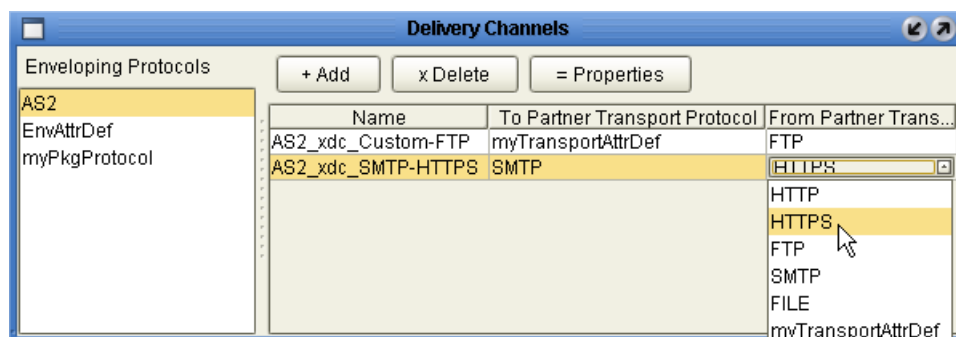
4 If you need to delete a leaf or a branch from the tree, highlight it and click **Delete**.

To add external delivery channels to a B2B host, organized by enveloping protocol

Before you begin: The Delivery Channels window must already display at least one item in its Enveloping Protocols pane; if it does not, refer to the previous procedure.

- 1 In the **Delivery Channels** window, in the Enveloping Protocols pane, click the enveloping protocol for which you want to add an external delivery channel.
- 2 Click **Add**. When the new external delivery channel appears on the right, rename it.
- 3 In the **ToPartner Transport Protocol** list, choose from the list of standard and custom transport protocols displayed. This specifies transport protocol attributes for messages outbound from the B2B host to the trading partner on this channel.
- 4 In the **FromPartner Transport Protocol** list, choose from the list of standard and custom transport protocols displayed. This specifies transport protocol attributes for messages inbound on this channel from trading partner to B2B host.
- 5 Repeat the last three steps as needed, or repeat all previous steps. See Figure 24.

Figure 24 External Delivery Channels Populated and Configured



Note: From the Delivery Channels dialog box, you can view or edit the MIME and Simple Parts properties of a particular enveloping protocol, including AS2 and ebXML (if you have checked them out), by double-clicking its name in the far left column.

- For any external delivery channel for which you want to override default properties, click the channel to highlight it, click the **Properties** button to open its property sheet, and navigate to the property or properties you want to modify. See Figure 25 and Table 4 through [Figure 28 on page 50](#).

Figure 25 Default Property Settings for External Delivery Channels – General

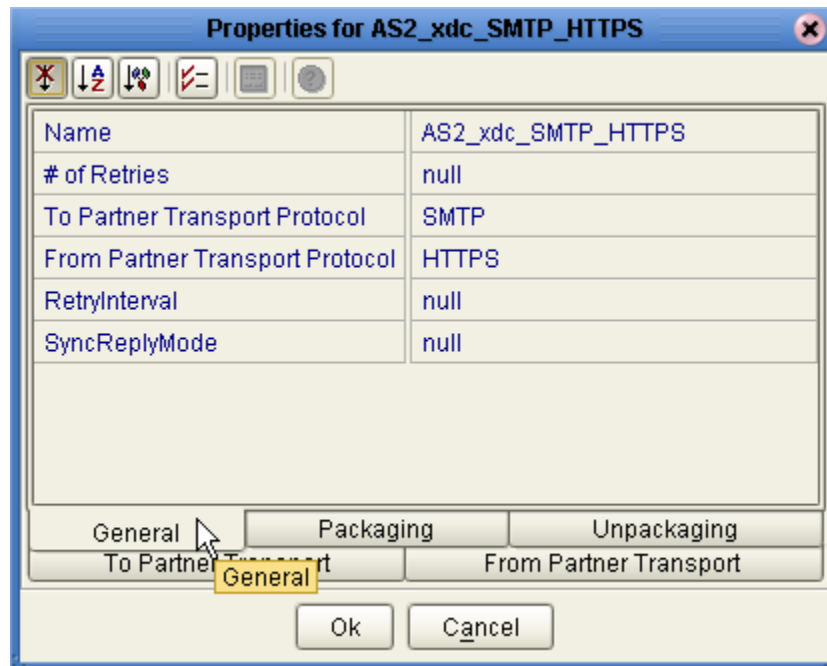


Table 4 Properties for External Delivery Channels – General

Property Name	Initial or Default Value	Description
Name	(component name)	Free text; this is the component name displayed in the project tree.
# of retries	null	Number of times to retry a send of the message.
To Partner Transport Protocol	(as specified in setup)	Transport protocol to be used when sending messages outbound to the trading partner.
From Partner Transport Protocol	(as specified in setup)	Transport protocol to be used when receiving messages inbound from the trading partner.
Retry interval	null (initially)	Interval to use for retrying a message send.
SyncReplyMode	null (initially)	The mode (synchronous or asynchronous) to be used for messaging.

Figure 26 Default Property Settings for External Delivery Channels – Packaging

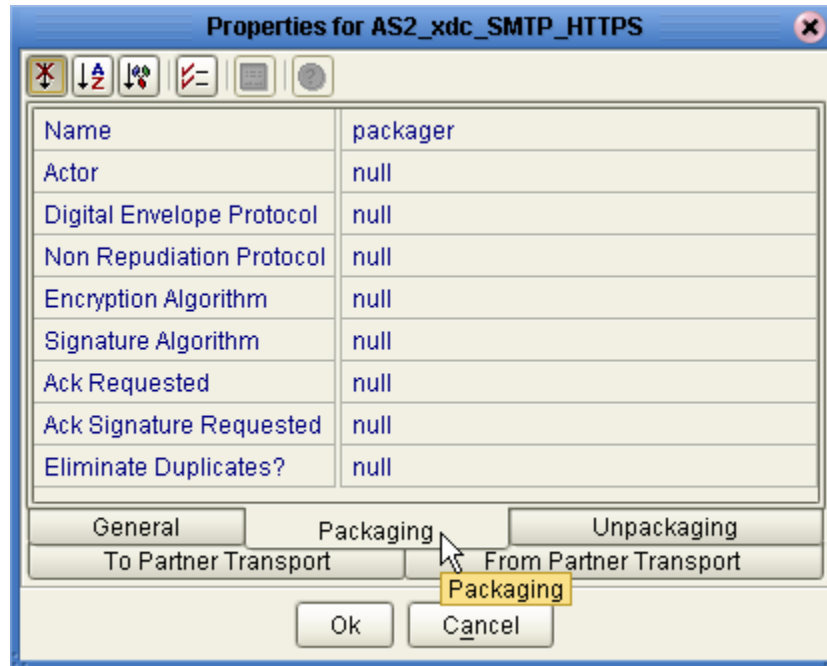
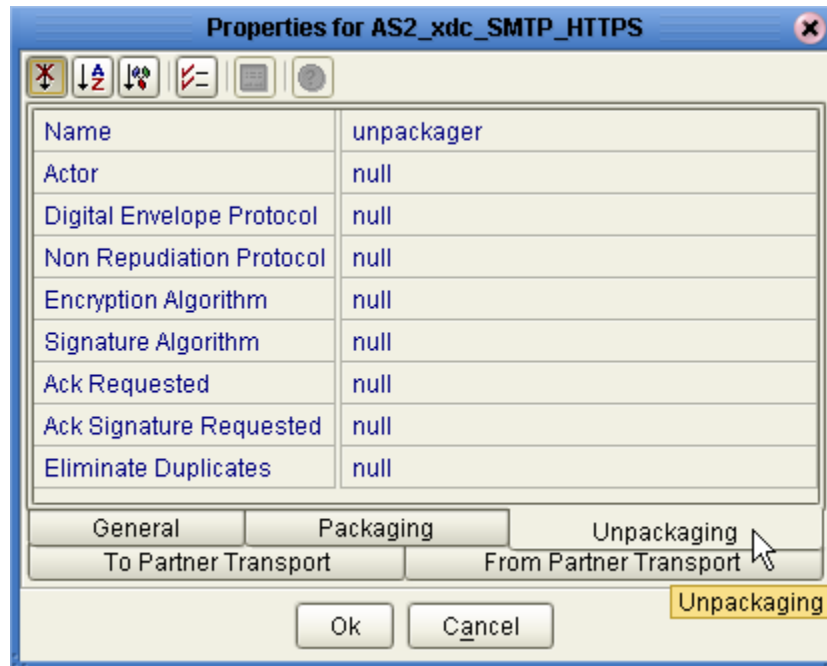


Table 5 Properties for External Delivery Channels – Packaging/Unpackaging

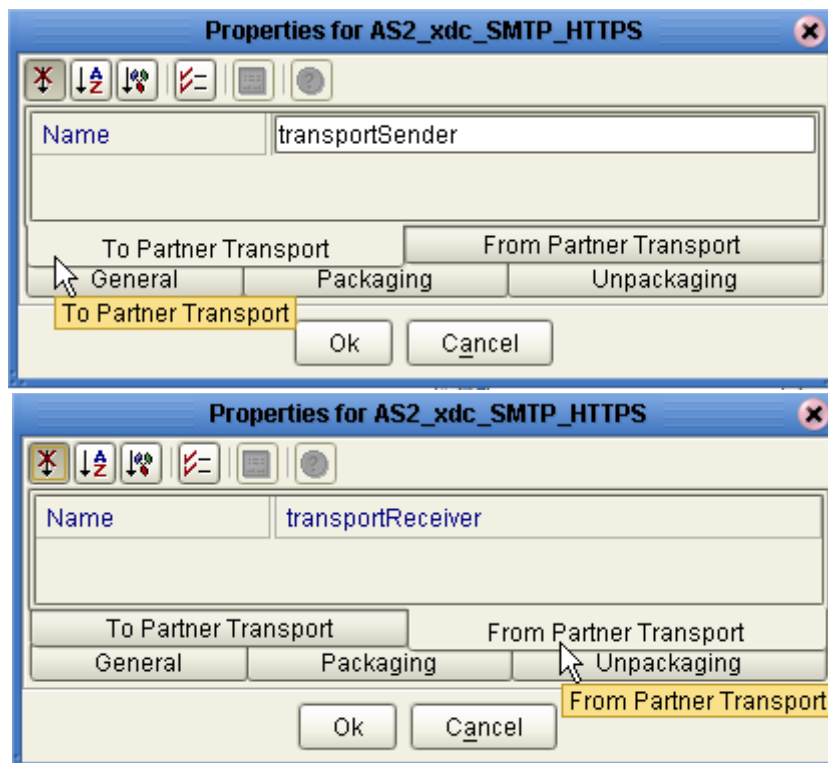
Property Name	Initial or Default Value	Description
Name	(component name)	Free text; this is the component name displayed in the project tree.
Actor	null (initially)	Namespace identifying the actor, such as urn:oasis:names:tc:ebxml-msg:actor:nextMSH
Digital Envelope Protocol	null (initially)	Digital envelope protocol and version to be used, of: SMIME/2.0 , SMIME/3.0 , or DSIG/1.0 .
Nonrepudiation Protocol	null (initially)	Nonrepudiation protocol and version to be used, of: SMIME/2.0 , SMIME/3.0 , or DSIG/1.0 .
Encryption Algorithm	null (initially)	Encryption algorithm to be used, of: RC2 or DES3 .
Signature Algorithm	null (initially)	Encryption algorithm to be used. If set, must be xmldsig#dsa-sha1 .
Ack Requested	null (initially)	Specifies circumstances in which a message acknowledgment is requested (such as TA1), of: Always , Never , or Per Message .
Ack Signature Requested	null (initially)	Specifies circumstances in which a signature is required on the message acknowledgment, of: Always , Never , or Per Message .
Eliminate Duplicates?	null (initially)	Specifies circumstances in which duplicate messages are to be eliminated, of: Always , Never , or Per Message .

Figure 27 Default Property Settings for External Delivery Channels — Unpackaging



For unpackaging properties, names, values, and descriptions are the same as for packaging properties; see [Table 5 on page 49](#).

Figure 28 Default Property Settings for External Delivery Channels — Partner Transport

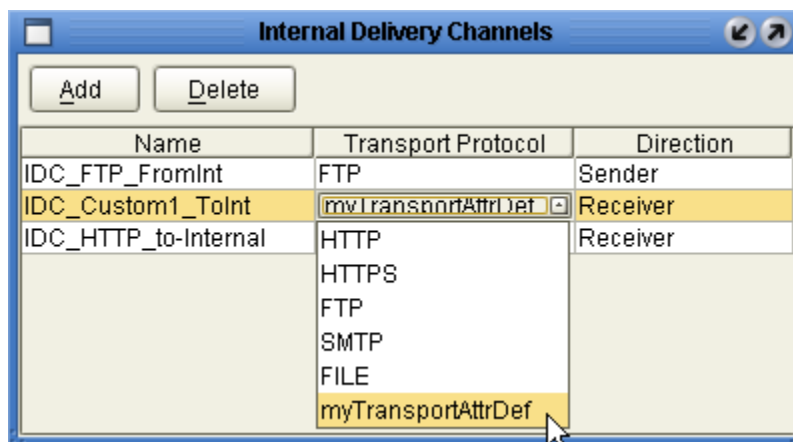


To add internal delivery channels to a B2B host

- 1 In the B2B Host Designer, in the **Internal Delivery Channels** window, click **Add**.
- 2 Rename the new internal delivery channel to something meaningful.
- 3 In the **Partner Transport Protocol** list, choose from the list of standard and custom transport protocols displayed.
- 4 In the **Direction** list, choose a designation of the role played by the Internal system:
 - ♦ **Sender** designates this internal delivery channel as coming from Internal.
 - ♦ **Receiver** designates this internal delivery channel as going to Internal.
- 5 As needed, repeat previous steps to add and configure internal delivery channels.

See Figure 29.

Figure 29 Internal Delivery Channels Populated and Configured



4.4 Activating a B2B Host

The B2B host plays a dual role: It functions both as an *object*—that is, a project-level (logical) component in the project tree that can be dragged into a Connectivity Map—and also as a *server*—that is, an environment (physical) component. As a server, it hosts Channel Managers and is exposed to the eXchange Partner Management GUI.

Unlike most other environment components, it is not offered as a template in the environment to be statically configured. Instead, since it has a dynamic dependency on the Oracle database that persists eXchange trading partner information (and message tracking information, if used), it must be preactivated for a particular environment. This section provides the steps required for preactivation: creating an environment, creating a map containing a B2B host, and activating a deployment profile that binds the map’s components to the environment.

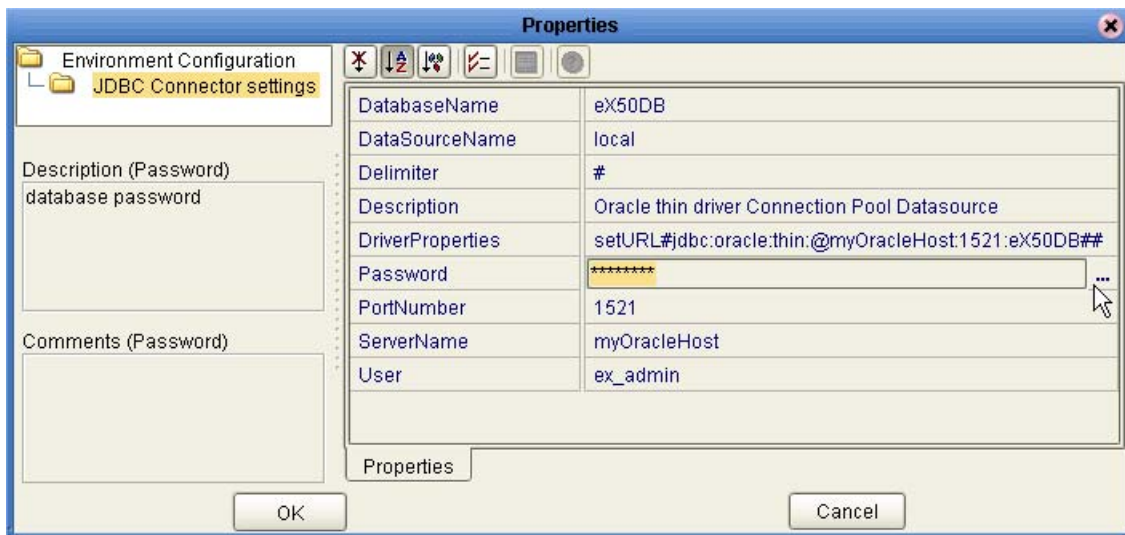
4.4.1. Creating an Environment

These steps create the minimal environment required to activate a B2B host.

To create and populate the environment prior to B2B host deployment

- 1 In Enterprise Designer with the **Environment Explorer** tab active, right-click the repository and, on the popup context menu, click **New Environment**.
The explorer tree displays a new environment, and the Environment Editor opens. Optionally, you can rename the environment to something meaningful.
- 2 In the environment explorer tree, right-click the new environment and, on the popup context menu, click **New Oracle External System**.
- 3 In the **Create an External System** dialog, enter a meaningful name, set the system type to **Outbound Oracle eWay**, and then click **OK**.
- 4 Configure the Oracle external with the values for your eXchange database instance. For details on configuring the Oracle external, see the Chapter 3 of the *Oracle eWay User's Guide*. For sample settings typical of an eXchange database, see Figure 30.

Figure 30 Environment Configuration for Oracle External System



- 5 In the environment explorer tree, right-click the new environment again and, on the popup context menu, click **New Logical Host**.
The explorer tree and editor canvas display the new logical host. Optionally, you can use the tree to rename the logical host to something meaningful.
- 6 In the environment explorer tree, right-click the new logical host and, on the popup context menu, click **New SeeBeyond Integration Server**.
The explorer tree displays the new integration server, and the canvas displays it inside the logical host. Optionally, you can rename it to something meaningful.
- 7 If appropriate, right-click the integration server, click **Properties**, and configure its parameters as needed for use at your site. For example, see [“Configuring the eInsight Run-Time Engine” on page 33](#) for instructions on configuring eInsight engine settings for database connections and persistence.

Once you finish these last steps, the environment now has all you need to deploy a B2B host: An integration server and an Oracle external system. However, you still need to create a map that links the B2B host to an Oracle eWay.

4.4.2. Creating a Map with a B2B Host and a MessageTracker

These steps create a map that allows you to activate a B2B host with a Message Tracker.

To create and populate the map prior to B2B host deployment

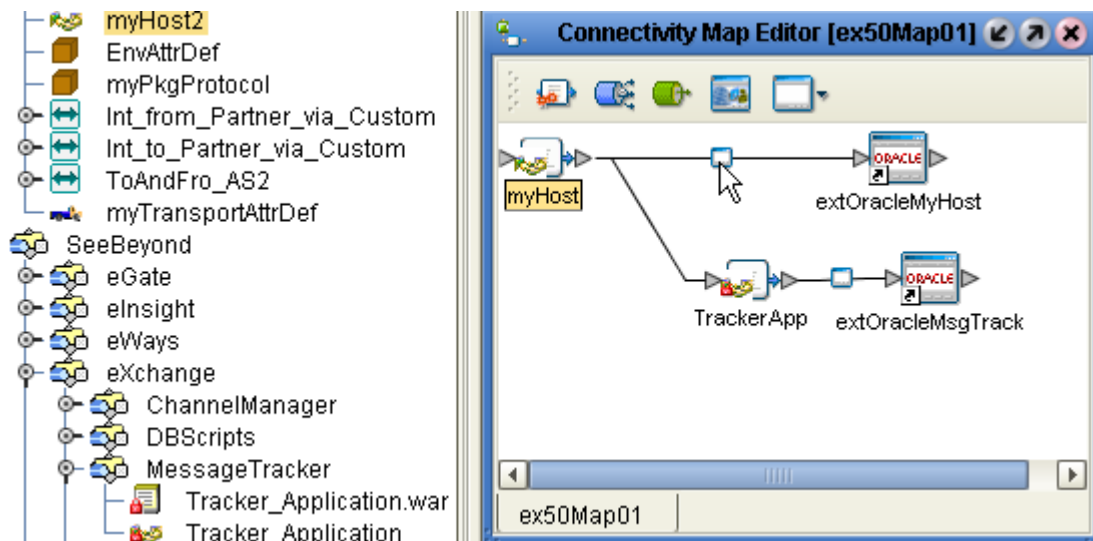
Before you begin: Your project must already contain a B2B host with at least one validly configured external delivery channel; see [Setting Up B2B Hosts](#) on page 45.

- 1 With the **Project Explorer** tab active, in the project tree, right-click the project and, on the popup context menu, point at click **New** and click **Connectivity Map**.

The project tree displays a new map, and the Connectivity Map Editor opens. Optionally, you can rename the map to something meaningful.

- 2 In the toolbar along the top of the canvas, click the **External Applications** tool and, from the drop-down list, select the checkbox for **Oracle External Application**.
- 3 Drag two Oracle externals onto the canvas, towards the right side.
- 4 From the project tree, drag your B2B host onto the upper left side of the canvas.
- 5 In the project tree, open the **SeeBeyond > eXchange > MessageTracker** folder and drag its **Tracker_Application** protocol pipeline onto the lower center of the canvas.
- 6 Rename components to something meaningful, and connect as shown in Figure 31.

Figure 31 Map Showing B2B Host and Message Tracker to be Deployed



- 7 Configure both eWays, designating them outbound to external Oracle applications. Now that the map is populated and configured, the B2B host is ready to be activated.

4.4.3. Activating the B2B Host

These steps create a deployment profile—a binding of particular logical components to an environment—and then activates it. Project activation always makes components available to external servers (such as the SeeBeyond integration server and Oracle), but activation of a B2B host makes it available as an external server. Activation also exposes the B2B host and MessageTracker application so they can communicate with the web-based eXchange GUIs, eXchange Partner Management and Message Tracking.

To create, configure, and activate a deployment profile

Before you begin: Your project must already contain a validly configured B2B host used in a map that connects it, via an outbound eWay, with a configured Oracle external.

- 1 With the Project Explorer tab active, in the project tree, right-click the project and, on the popup context menu, point at click **New** and click **Deployment Profile**.

The project tree displays a new deployment, and the Deployment Editor opens, showing a canvas whose left pane shows all components on the map and whose right pane shows all external servers in the environment.

- 2 One by one, drag both Oracle eWays to the Oracle server. Then, one by one, drag the B2B host and the tracker application to the integration server. See Figure 32.

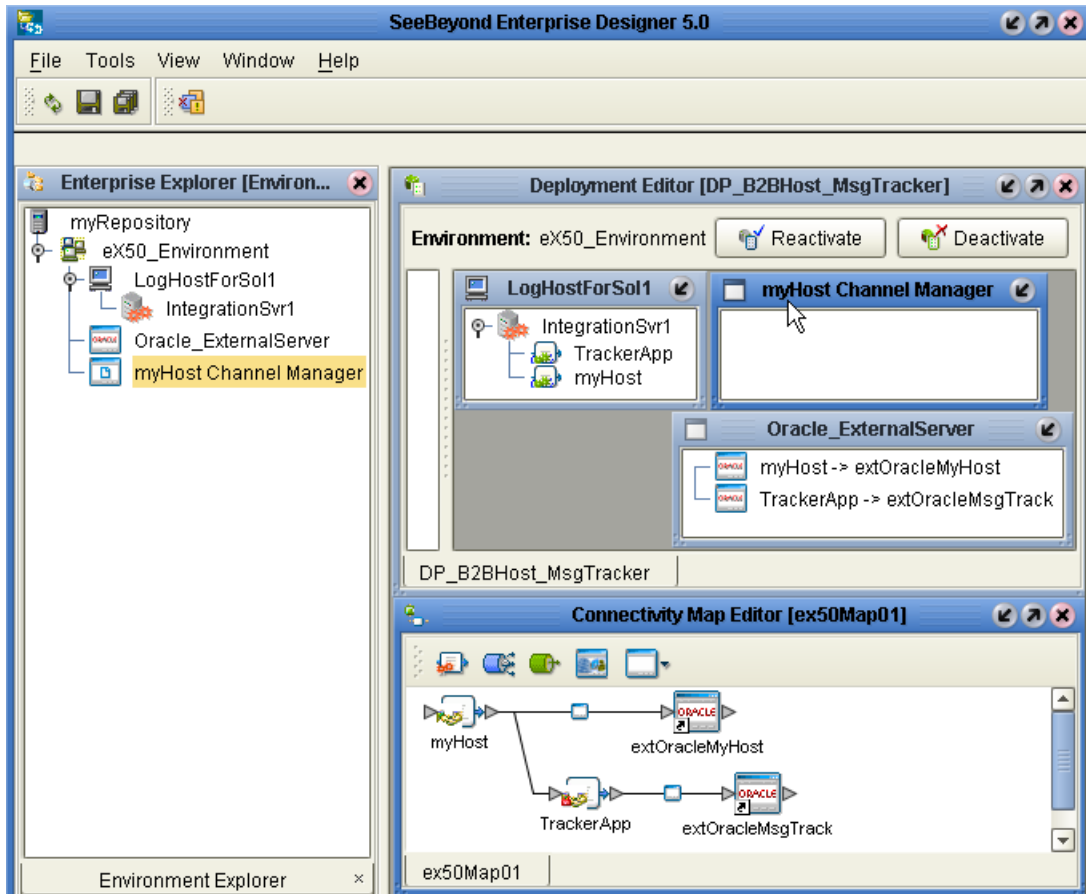
Figure 32 Deployment Profile Showing B2B Host Being Assigned to Integration Server



- 3 After all components have been assigned to external servers, click **Activate**.

Upon successful activation, the Environment Editor and Deployment Editor are updated to show a new B2B host server. See Figure 33.

Figure 33 Activated B2B Host in Environment and Deployment Profile



Once deployed, the B2B host server is capable of hosting Channel Managers; it is also available to Enterprise Manager facilities that perform pipeline monitoring, trading partner management, and message tracking.

To associate certificate and keystore information with a B2B host's delivery channels

Before you begin: You must have already installed SMEWebServices.jar (see step 9 in the [procedure on page 26](#)), and you must have access to the locations and names of the necessary certificates and keystores.

- 1 In Enterprise Designer with the **Environment Explorer** tab active, in the tree, right-click the B2B host Channel Manager and, on the popup menu, click **Properties**.

The properties page opens, showing all external delivery channels for this host.

- 2 One by one, as needed, click the ellipsis [...] for the type of certificate or truststore you want to define for a particular external delivery channel and then either select from the drop-down list or else use Import to import a new certificate or truststore. For certificate import, see Figure 34; for truststore import, see Figure 35.

Figure 34 Importing a Certificate for a Delivery Channel in a B2B Host Environment

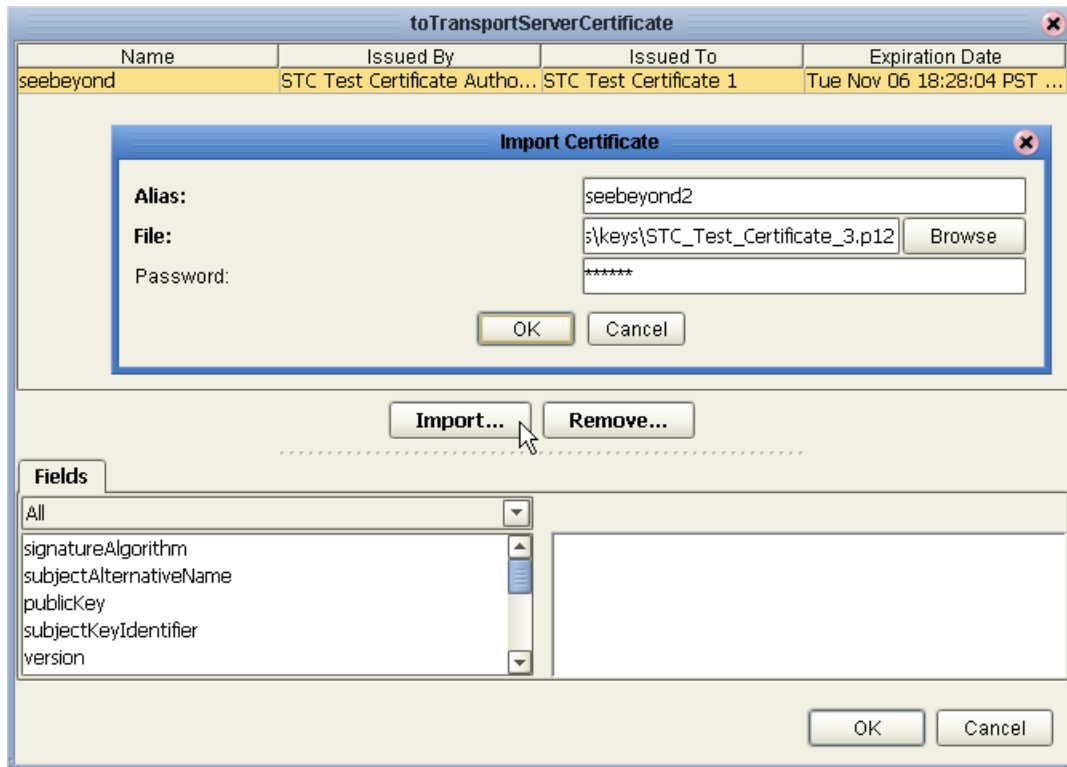
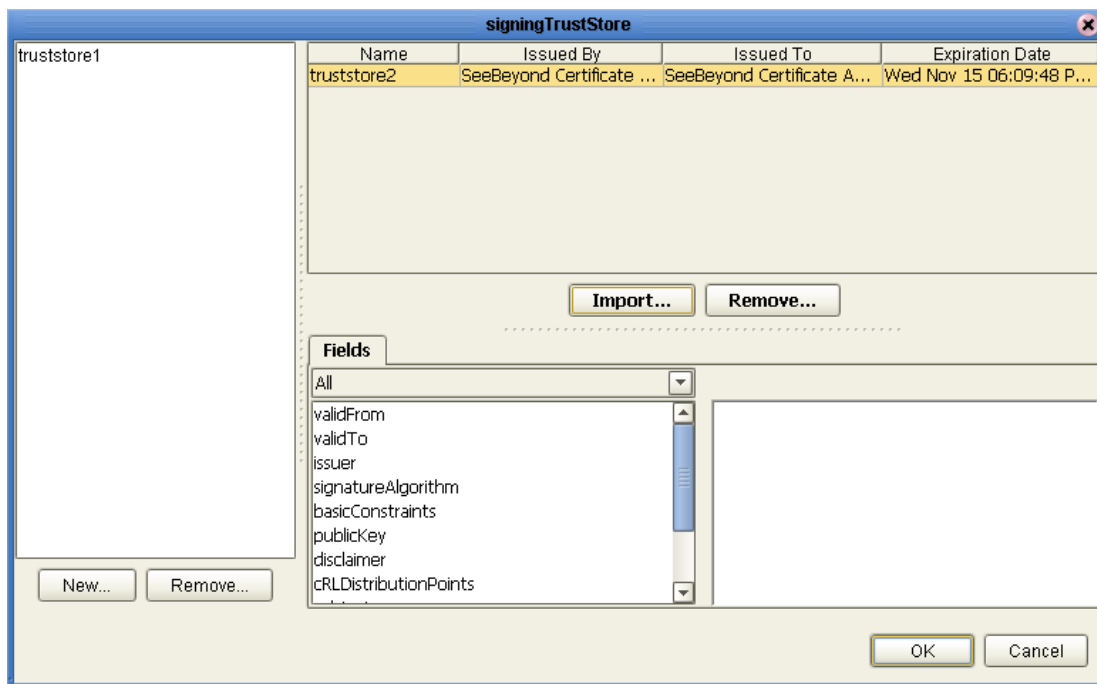


Figure 35 Importing a TrustStore for a Delivery Channel in a B2B Host Environment



Chapter 5

Designing B2B Protocol Pipelines

You can use eXchange to configure the components depicted by each activity in your B2B protocol pipelines. This chapter provides the background information you need to create and understand eXchange B2B protocol pipelines.

5.1 Overview

Topics in this chapter are:

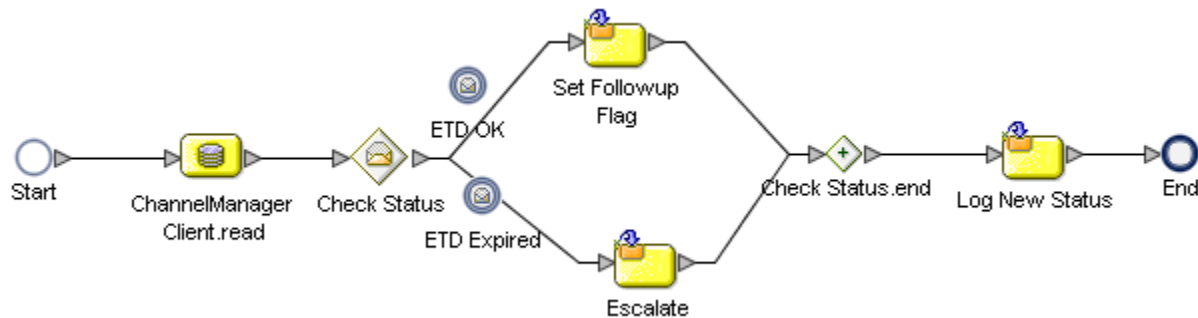
- [“Building a B2B Protocol Pipeline” on page 57](#)
- [“Using the eXchange Protocol Designer GUI” on page 59](#)
- [“Modeling Elements” on page 60](#)
- [“Using B2B Protocol Pipelines in a Connectivity Map” on page 65](#)
- [“Deploying a Project With a B2B Protocol Pipeline” on page 66](#)

5.2 Building a B2B Protocol Pipeline

A business process is a collection of actions 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 the business process called a *B2B protocol pipeline*.

A business process modeled in eXchange may look something like Figure 36.

Figure 36 Sample B2B Protocol Pipeline



Add a B2B Protocol Pipeline to your Project

Adding a B2B protocol pipeline to your project provides an empty modeling canvas where you add and manipulate items on the canvas, called *activities*. Before you can model your business process, you must add a new protocol pipeline to your project.

Note: The *eGate User's Guide* has detailed information on creating a project.

- 1 In Enterprise Designer, in Project Explorer, right-click the project and, on the popup context menu, point at **New** and click **B2B Protocol Pipeline**.
- 2 Enter a new name for your B2B protocol pipeline.

5.2.1. Modeling a Business Process in eXchange

To model a business process in eXchange, drag and drop modeling elements on the Exchange Protocol Designer, and then link these components to reflect the logical flow of the business process. eXchange provides the tools you need to quickly develop B2B protocol pipeline models, including graphical editing tools to help you adjust, size, and align model components.

eXchange Protocol Designer

Once you create a new protocol pipeline, you will build your model in the eXchange Protocol Designer (as shown in Figure 37). The eXchange Protocol Designer is the area in the Enterprise Designer where you view, create, and edit your protocol pipelines.

Create a B2B Protocol Pipeline

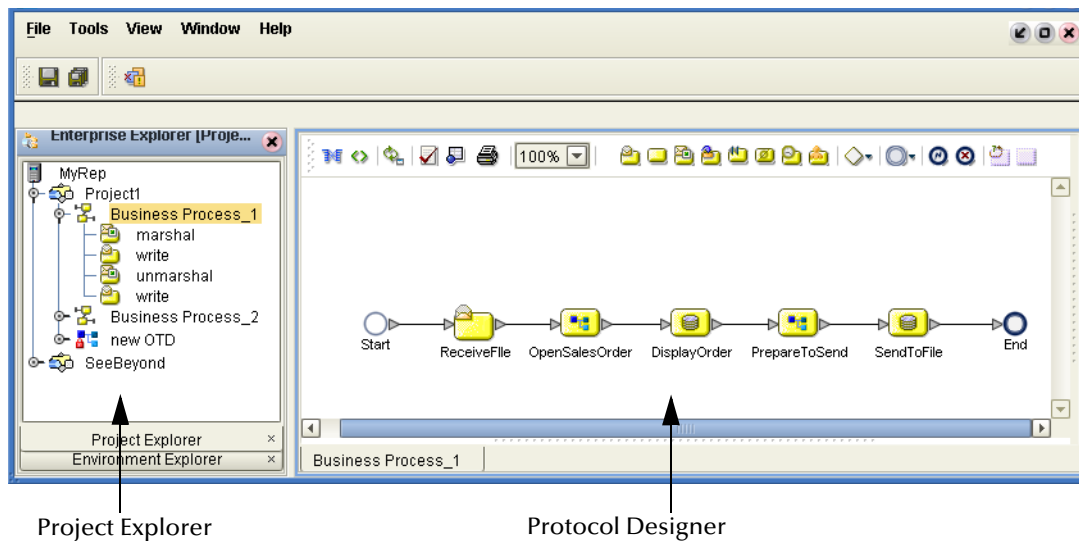
Begin designing your protocol pipeline by dragging and dropping modeling elements from the toolbar onto the Exchange Protocol Designer canvas.

The **Start** and **End** activity appear on the blank canvas by default. There is only one starting point for any protocol pipeline. (There can be multiple end points.)

- 1 Drag the appropriate modeling elements to your blank protocol pipeline to the Exchange Protocol Designer canvas. See Figure 37.

Note: See [Appendix A](#) for a complete list of modeling element options.

Figure 37 B2B Protocol Pipeline

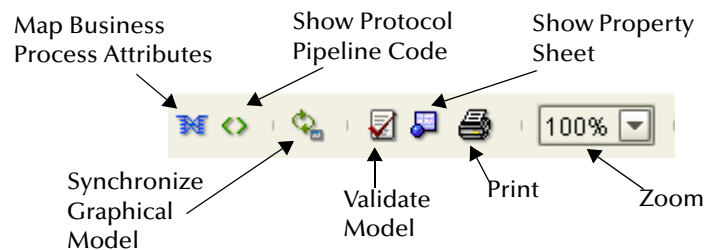


- 2 Draw links between the modeling elements to show the process flow (Figure 37)
- 3 On the main toolbar, click **Save** to save your changes to the Repository.

This will validate your protocol pipeline, generate the code to run it, and save your changes to the SeeBeyond Repository.

5.3 Using the eXchange Protocol Designer GUI

Figure 38 Toolbar Options



- **Map Business Process Attributes**—Opens the **Business Rule Designer**.
- **Show Protocol Pipeline Code**—Toggles display of underlying BPEL code.
- **Synchronize Graphical Model and Protocol Pipeline Code**—Causes the graphic model, the Business Rules, and the underlying BPEL code to match.
- **Validate Protocol Pipeline Model**—Runs application to check syntactic validity.
- **Show Property Sheet**—Toggles display of property list and graphical overview.
- **Print**—Prints the protocol pipeline graphic. Options allow you to control the scale.
- **Zoom**—Enlarges or shrinks the displayed graphic in the canvas.

5.4 Modeling Elements

The eXchange Protocol Designer is where the user creates the protocol pipeline flow. It provides a palette of modeling elements for designing your B2B protocol pipeline. Like other logical components in a project, protocol pipelines appear in the Project Explorer tree.

Elements from the Enterprise Explorer can either be dropped onto empty canvas or onto an Activity. Many elements provide custom settings so that you can model every detail of your process. Each B2B protocol pipeline you create consists of basic elements as described in the following sections:

- **Activity Elements** on page 60
- **Branching Activities** on page 62
- **Intermediate Events** on page 63
- **Scope** on page 64
- **While** on page 64

Activity Elements

You can include several different kinds of activities and subprocesses in a B2B protocol pipeline. For examples of each of the different kinds of activities, see Table 6.

To add an activity

- 1 Either drag a modeling element from the toolbar or drag a web service operation from the Project Explorer, and then drop it where you want it on the canvas.
- 2 Click the activity name and begin typing to rename it from the default.

Note: *Every activity name must contain at least one character (A-Z, a-z, or 0-9); it must start with a letter or an underscore (_), and it may contain spaces.*

The activity appears on the modeling canvas.

Link modeling elements

eXchange supports orthogonal and diagonal link styles – this setting applies to all links in a model and is an automated application of the style.

To link modeling elements, do the following:

- 1 Move your cursor over the connector portion of your modeling element.
- 2 Hold the cursor over the outside edge of the modeling element until it changes from the arrow pointer to a hand (see Figure 39).

Figure 39 Selected Activity



- 3 Click down, and drag a line from the first activity to the connector of the second activity. When the link attaches, release the mouse button.

Table 6 Activity Elements













Button	Command	Function
 Start	Start Node	<p>The Start Node is a modeling element indicating the start of the process. This element appears in the eXchange Protocol Designer Editor by default, when you create a new B2B protocol pipeline.</p> <p>A Start Node can only link to an activity that has a receive or read capability, signaled by a subicon in the upper left resembling an opened envelope (see Receive Activity just below).</p>
 	Link Link with Business Rule	<p>Links indicate the flow of the protocol pipeline by connecting elements together. When you select a link, a context menu allows you to configure how data is going to be passed to and from the underlying component or web service operation using protocol pipeline attributes.</p> <p>eXchange ensures the model is being properly linked because it does not allow invalid links to connect. Links can also accept mapped values. A link with mapped values is highlighted in blue.</p>
 End	End Node	<p>The End modeling element indicates the completed state of a protocol pipeline. This element appears in the B2B Protocol Designer by default, when you create a new B2B protocol pipeline.</p>
	Receive Activity	<p>The Receive activity indicates the invocation of a protocol pipeline or a wait state pending the arrival of an inbound message.</p> <p>The Receive activity represents the actual method by which a B2B protocol pipeline is initiated. For example:</p> <ul style="list-style-type: none"> ▪ An eWay polls a file system or database and retrieves data that is passed to the engine, along with the indication that a protocol pipeline instance has started. ▪ A user types a URL into their browser and a servlet initiates a protocol pipeline by sending a message to eGate or eInsight.
	Activity	<p>An activity is a step in the protocol pipeline in which the engine will invoke a web service operation or an eGate component. Depending upon the configuration of the component, a response may or may not be required. One example would be a synchronous extraction process from a database to return the credit status of a trading partner.</p>

Table 6 Activity Elements

Button	Command	Function
	Reply Activity	<p>The Reply activity allows a protocol pipeline to respond to the external system or user that originally invoked the protocol pipeline. The original receive at the beginning of the protocol pipeline is paired with the Reply at the end of the process. In cases where a message must be sent back to the caller of the protocol pipeline, the Reply uses information that correlates the message in the calling system.</p> <p>A Reply acts as the last step in a protocol pipeline in which the protocol pipeline is acting as a web service operation or subprocess. A Reply correlates the outbound message back to the calling process; for example, it can reply to an external system as a web service operation.</p>
	Business Rule Activity	The Business Rule activity sets data values, including task assignments. It is used when imported models have multiple data mappings between the invocation of human tasks or automated systems.
	Compensate	The Compensate element invokes compensation on an inner scope that has already completed normally. This construct can be invoked only from within a fault handler or another compensation handler.
	Empty Activity	The Empty activity allows data to pass through without any changes.
	Wait Activity	The Wait activity acts as a timer. The user will build a model in which there are two simultaneous paths within a set scope, one for the protocol pipeline and one for the timer. If the timer condition takes place first, an exception will be thrown and handled, and the protocol pipeline path will then be abandoned.
	User Activity	The User activity is used only by eInsight, and should not be placed on a canvas unless your site is licensed for eInsight as well as eXchange. It is used when assigning, escalating, or otherwise using human intervention to complete eInsight business process tasks.

Branching Activities

Branching activities are objects you add to your B2B protocol pipelines to specify the logical flow of information. eXchange provides three different kinds of branching activities—Decisions, Event Based Decisions, and Flow.




Add a Branching activity

To add a Branching activity to the modeling canvas:

- 1 On the toolbar, click the **Branching Activities** drop-down icon, and then release the mouse button.
- 2 Point at the type of Branching activity you want to add, click, and then drag the activity from the toolbar to the eXchange Protocol Designer canvas.

The selected Branching activity appears on the modeling canvas.

Table 7 Branching Activities

	Decision	<p>A Decision allows one of several possible paths to execute, based on expression logic. This element is used to create complex expressions that determine the path of the protocol pipeline. It also contains the expression and connection names.</p> <p>Decisions allow you to define expressions that are evaluated to determine the proper protocol pipeline flow. Expressions are built using the mapping interface and protocol pipeline attributes.</p>
	Event Based Decision	<p>Multiple possible messages can be juxtaposed against a timeout condition to allow the type of message received to determine the appropriate protocol pipeline path.</p>
	Flow	<p>Allows you to specify one or more activities to be performed concurrently.</p>

Intermediate Events

Intermediate events are those activities that can interrupt the flow of a protocol pipeline. Some intermediate events handle exceptions that may occur during your protocol pipeline or compensate for exceptions that occur.

Add an Intermediate event

To add an **Intermediate event** to the modeling canvas:

- 1 On the toolbar, click the **Intermediate Events** drop-down icon, and then release the mouse button.
- 2 Point at the type of Intermediate event you want to add, click, and then drag the activity from the toolbar to the eXchange Protocol Designer canvas.

The selected Intermediate event appears on the modeling canvas.

Table 8 Intermediate Events







	Compensation Handler	<p>Used when something in a protocol pipeline fails and requires a rollback or upstream activities (like money has to be returned to the customer). On an automatic basis in the protocol pipeline, upstream steps in the protocol pipeline are notified that the failure has occurred and certain transactions need to be reversed, sometimes in a sequential order. The compensation handler allows you to design the process and circumstances in which the compensation takes place.</p>
	Catch Named Exception	<p>Each automated system (back-end system) or web service operation can publish their possible error codes (for instance, fault 15 is “bad data”). Those codes can be mapped to exception handlers. Each exception handler is connected to the scope that surrounds one or more steps in a protocol pipeline. The components within that scope will throw the exceptions when things go wrong and the exception handler will automatically initiate the appropriate process to handle the problem.</p>

Table 8 Intermediate Events


	Catch All Exceptions	This exception handler is configured to handle all exceptions that occur in a scope.
	Message Event	This is similar to a Receive Activity, but it occurs only in the middle of a pipeline. Each of these elements can be a different message.
	Timer Event	A timeout condition is set upon Activities, sets of Activities, or a protocol pipeline as a whole, to ensure that processes complete within given amount of time. Timeout conditions also allow you to design the pipeline branch to take after a timeout condition takes place.

Scope

The behavior for one or more activities can be defined by a scope. A scope can provide exception handlers, event handlers, a compensation handler, and data variables. The exception handlers for the scope can be used to catch the faults caused by the possible exception responses.

	Scope	The Scope element allows you to apply exception handlers, compensation and transactionality to a set of sequential or simultaneous steps in a protocol pipeline.
---	-------	--

While

	While	This allows you to create a looping process within a protocol pipeline (for instance, a negotiation process may take several weeks, but the manager wants to review the daily status). The loop continues until the negotiation is complete, and then the protocol pipeline continues.
---	-------	--

5.4.1. Validating a B2B Protocol Pipeline

After creating a B2B protocol pipeline, you can check to see if there are any problems such as activities that are not connected or an incorrect number of output links from an activity.

To check the protocol pipeline for errors

- On the toolbar, click **Validate Protocol Pipeline Model**.

If an error is encountered, a message box displays information about the error. If there are no errors, a message appears stating that there were no errors.

Note: *If an error message displays, see “Saving an Unfinished B2B Protocol Pipeline” for information on repairing errors. Repairing the error may entail such items as adding logic to Decisions or adding attributes to activities.*

5.4.2. Saving an Unfinished B2B Protocol Pipeline

Even if a B2B protocol pipeline is not complete and/or contains errors, you can save it as a work in progress and return to it later.

To save a B2B protocol pipeline

- Do one of the following:
 - ♦ On the **File** menu, click **Save**
 - ♦ On the main toolbar, click **Save**.
 - ♦ On the keyboard, press **Ctrl+S**.

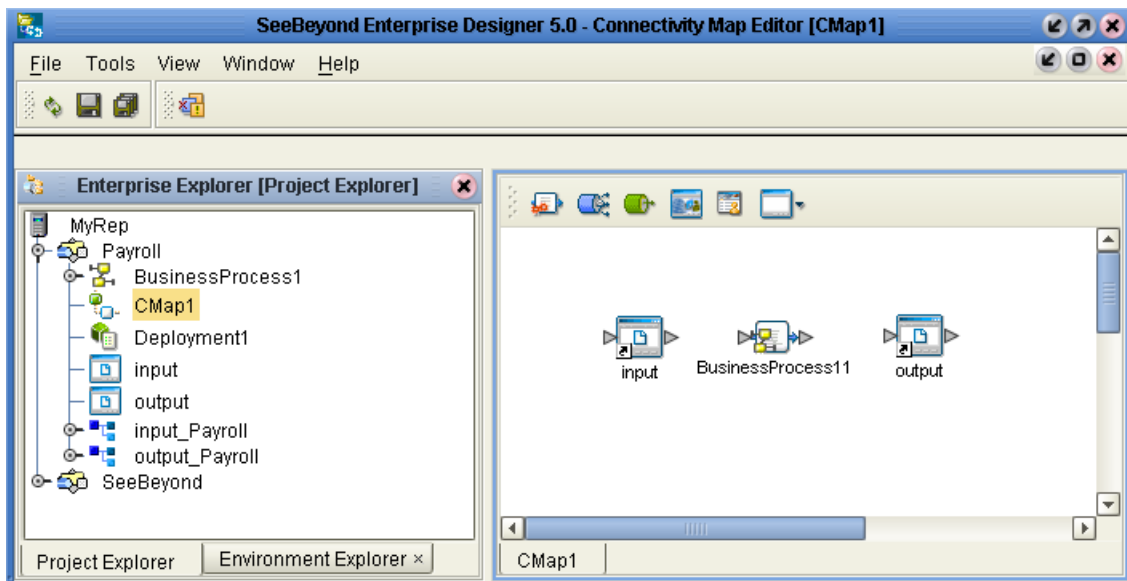
5.5 Using B2B Protocol Pipelines in a Connectivity Map

The connectivity map represents connection information in the ICAN Suite. The flow is represented at a higher level than in the B2B protocol pipeline. eXchange also uses the information in the connectivity map to establish and maintain connections to systems for the correct step in a protocol pipeline.

To include a protocol pipeline as a service on a connectivity map

- 1 In the Connectivity Map Editor, drag a protocol pipeline onto the canvas.
- 2 Add and connect other components and external systems as needed. See Figure 40.

Figure 40 Connectivity Map with B2B Protocol Pipeline

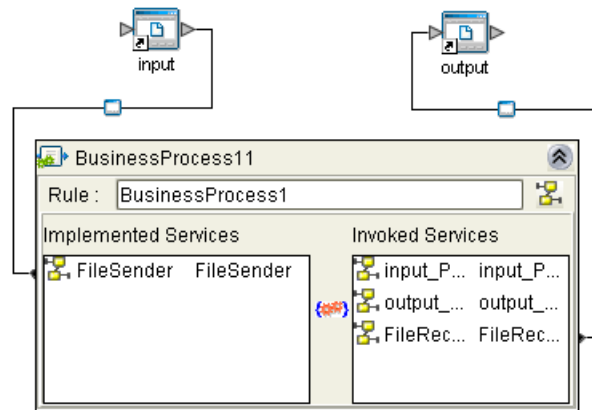


To connect the B2B Protocol Pipeline activities to the externals

- 1 In the map, double-click the protocol pipeline to open the Binding Dialog.
- 2 Connect the appropriate activities to the corresponding external.

Note that **Receive** activities appear in the left pane, and **Invoke** and **Reply** activities appear in the right pane. See Figure 41.

Figure 41 Connectivity Map: Protocol Pipeline Binding



5.6 Deploying a Project With a B2B Protocol Pipeline

5.6.1. Deployment Profiles Containing Protocol Pipelines

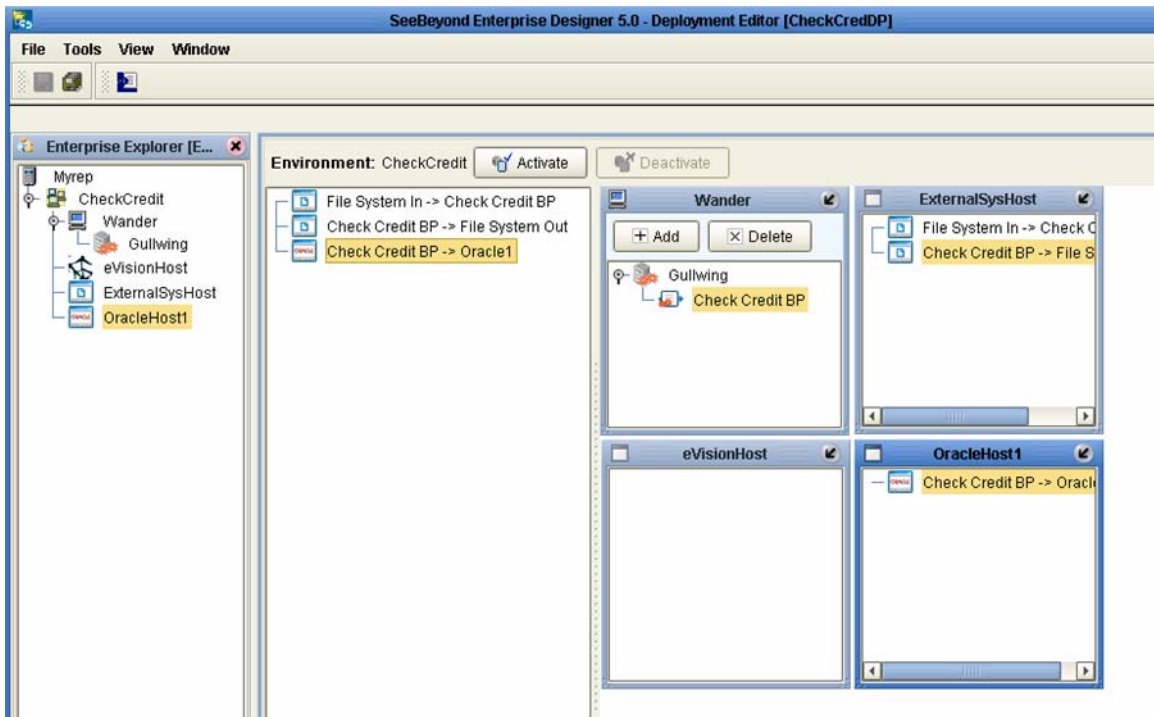
An environment becomes useful only after it has been populated with:

- One or more logical hosts, each containing one or more integration servers. Configuration steps may be required for integration servers, depending on the nature of the project and run-time environment.
- All necessary external hosts, properly configured.

Note: *If a B2B host is present in a pipeline or map, it requires special preactivation steps. See [Activating a B2B Host](#) on page 51.*

When these conditions are met, a deployment profile that references the environment is used to assign each logical component, eWay, and Channel Manager to a corresponding integration server and external host system. See Figure 42.

Figure 42 Deployment Profile with Some Components Assigned



- Each B2B protocol pipeline and each B2B host component should be dragged under the correct integration server of the appropriate logical host.
- Each eWay should be dragged into the correct external host system.
- Each Channel Manager, if present, should be dragged into a preactivated B2B host.

After all logical components have been assigned, click **Activate** to generate the code that will be run by the integration server within the logical host engine.

Chapter 6

Exception Handling

This chapter explains the concept of exception handling and how to configure various methods of handling errors.

- [“Scope” on page 69](#)
- [“Compensation” on page 70](#)

6.1 Overview

Exception handling is the identification of failed components or systems. In eXchange, exception handling allows one or more components to throw an exception that is caught by eXchange within a *scope*. Using the **scope** element, you can configure eXchange to catch all exceptions or certain exceptions that you specify. The elements that you use to configure exception handling in your model are:

- **Catch Named Exceptions**
- **Catch All Exceptions**

Exception handling in protocol pipelines relies heavily on the concept of *compensation*. Compensation is an application-specific activity that reverse the effects of a previous activity that was carried out as part of a larger unit of work that is being abandoned.

Protocol pipelines are often of long duration and use asynchronous messages for communication. They also manipulate sensitive business data in back-end databases and line-of-business applications. As a result, the overall business transaction may fail or be cancelled after many transactions have been committed during its progress. In these cases, the partial work may need to be reversed.

6.1.1. Exception Handling Configuration

Exception handlers are configured to catch errors that are thrown by eGate components and/or Web Services. These systems can be configured to publish one or more exceptions.

- **Manual Exception Handling:** The model can contain protocol pipeline logic designed to handle the exception.
- **Automatic Exception Handling:** Pre-packaged functionality guides the user to create multiple types of catches for thrown exceptions.

Each exception can be handled differently. This is one example:

- 1 Build the exception handling logic as a protocol pipeline.
- 2 Select the exception handler to configure which exception triggers the exception handling process.
- 3 Drag the Scope element onto the eXchange Protocol Designer canvas.
- 4 Drag the Exception modeling element into the scope for which it should take effect.
- 5 Define a protocol pipeline that appropriately handles each exception.
- 6 Model manual exceptions in a protocol pipeline.
- 7 Configure the exception handler to take place when one of the components within the Scope throws the appropriate exception.

Identifying Component or System Failures

Exception management allows users to quickly identify and correct problems with components or systems.

Users can filter the list of displayed instances to quickly identify exceptions.

Users can easily navigate to particular versions of a protocol pipeline to monitor the progress of instances.

A Web-based interface allows users to securely access the monitoring environment over the Internet.

Identification of troubled instances (i.e. time-outs or bad messages).

Failed components/systems create visual alerts via the protocol pipeline monitoring interface. The integrated monitoring environment allows you to identify the problem, assign a resource to fix the problem, and if necessary, restart the affected instances.

Users can quickly identify troubled instances from a large number of instances, repair and restart that instance for continued processing.

6.2 Scope

Scope allows you to define a range

- For handling of exceptions
- For creating compensation logic

The range of the scope can span one or more activities in the protocol pipeline or even the entire protocol pipeline.

Scope or Process level exceptions

Either the **Named** or **Catch All** exceptions can be used at the protocol pipeline level.

Named exception

- 1 Drag the **Catch Named Exception** element into the scope for which the exception handler applies.
- 2 In the Exception Handler properties, configure the following:
 - ♦ Fault Container—The output Attribute that will be containing the run-time name of the thrown fault.
 - ♦ Fault Name—The run-time value for the exception that will be passed from the component to the engine at run time.

Note: *The fault name is auto-populated with values based on the components dragged to the editor.*

- 3 Select the configuration control for the Exception Handler – the properties pane will appear to select the Fault name and container.
- 4 Drag the **Catch Named Exception** into the associated scope.

Catch All Exceptions

No configuration of the Catch All Exceptions element is required – any thrown exception not previously caught is caught with the Catch All Exception element.

6.3 Compensation

Compensation allows the modeler to create the process flow for executing complex compensations. Exception Handlers for parent scopes invoke the correct Compensation Handlers in the appropriate order.

Using Scope and Exceptions to Trigger Compensation

- Compensation Activity—In an exception handler, initiates the compensation process. It models the compensation as a protocol pipeline, and indicates the Compensation for “DB Insert” should be initiated.
- Compensation Handler— This is dropped within a scope to create the compensation logic for a given scope.

6.4 Validate the B2B Protocol Pipeline

After generating the business process code (BPEL), you can click the **Validation** button on the toolbar to identify any issues with the model. The validation results now appear in a wizard, listing any issues one by one with clear and understandable descriptions for the issues. You can fix each issue, regenerate the business process code, and again view the validation results until each of the issues has been fixed, and the model validates as correct.

Using eXchange in Enterprise Manager

This chapter provides step-by-step procedures for using the Web-based facilities in Enterprise Manager to configure and monitor eXchange components and processes.

In this chapter

- “eXchange Partner Management (ePM)” on page 71
- “Monitoring B2B Protocol Pipelines” on page 79

Each of these three facilities requires that you have already created a valid project and activated it.

7.1 eXchange Partner Management (ePM)

Before you begin: You must have already configured a B2B host with one or more delivery channels (see [procedure on page 45](#), “Setting Up B2B Hosts”) and you must have already preactivated it (see [procedure on page 51](#), “Activating a B2B Host”). If you want to use encryption in the trading partner, you must have already configured the proper certificates and truststores in the environment properties of the B2B host.

Overview

In this procedure, you will add a trading partner, bind it to at least one external delivery channel, add a profile beneath it, configure the profile, assign a business protocol to it, and configure all of the actions within the business protocol.

Accessing eXchange Partner Manager (ePM)

To access eXchange Partner Manager and locate a B2B host in the explorer tree

Before you begin: Your Repository server must already be running.

- 1 Open a browser window and point it at the hostname and port where your Repository server is running. For example:

`http://localhost:16271`

- 2 Log in to Enterprise Manager.
- 3 When you have logged in, point your browser at a new URL that is the same as the previous one but with the string `/epm` appended. For example:

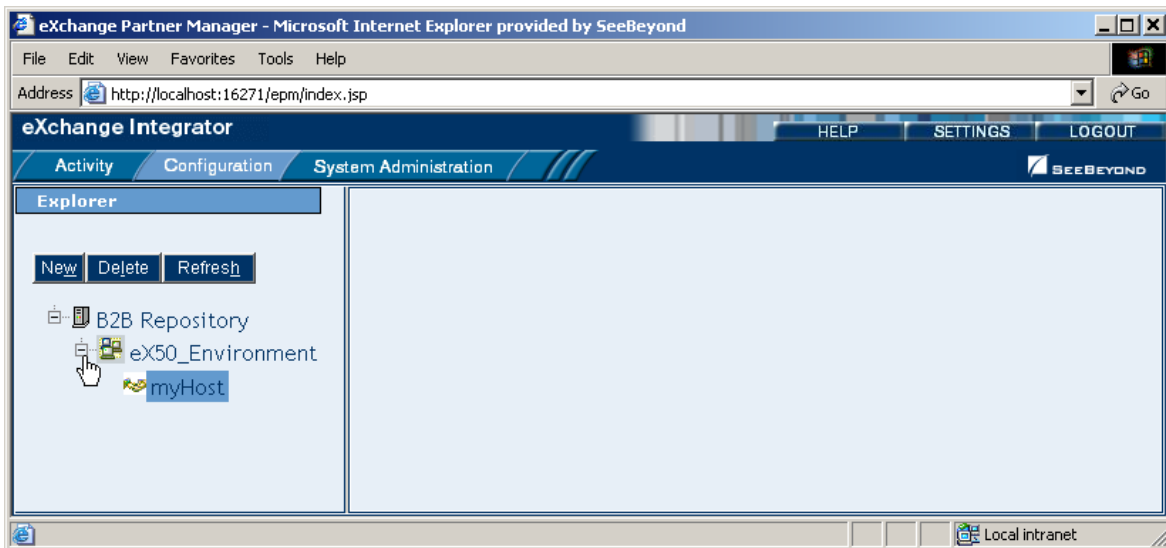
`http://localhost:16271/epm`

Note: This URL is case-sensitive. If your browser warns you, “Requested resource not available,” double-check that you entered the suffix *lep*m in all-lowercase.

After a pause, the window displays the eXchange Partner Manager GUI, a two-pane window with an explorer tree on the left and a canvas on the right.

- 4 In the tree, open the root (named **B2B Repository**) and then open the environment where you deployed your B2B host. So far, the canvas remains blank. See Figure 43.

Figure 43 eXchange Partner Management (ePM) – No Configuration

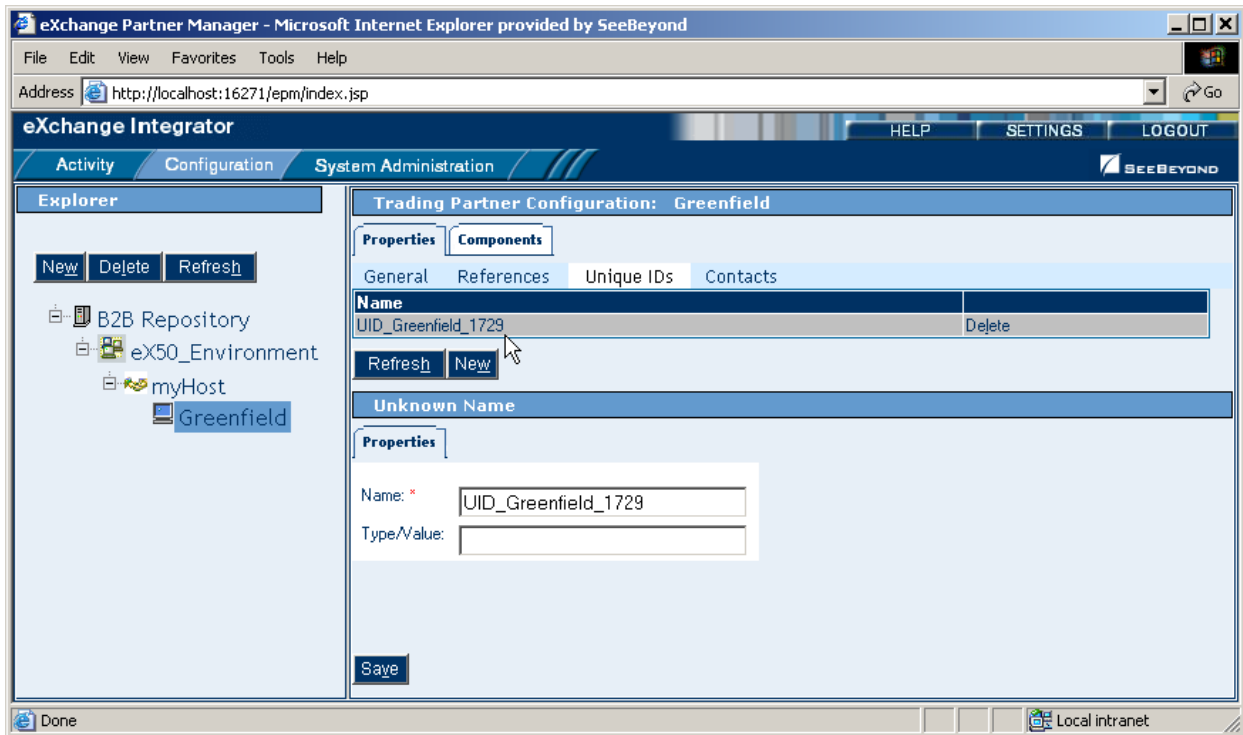


Creating Trading Partners

To create and name a trading partner

- 1 With the B2B host highlighted, click **New** (top left button in Explorer pane).
- 2 Enter a name for the trading partner, and then click **Save**.
In the explorer tree, the new trading partner appears under the B2B host.
- 3 Click the trading partner and then click **New** (upper left).
The **Trading Partner Configuration** canvas displays two tabs: **Properties** and **Components**.
- 4 In the Properties tab with the **General** subtab active, click **Save**.
- 5 Click **Unique IDs** (the third subtab), and then click **New** (button in lower center).
- 6 Enter a unique ID for this trading partner, and then click **Save**.
The new unique ID appears under the Name column. See Figure 44.

Figure 44 New Trading Partner with Unique ID

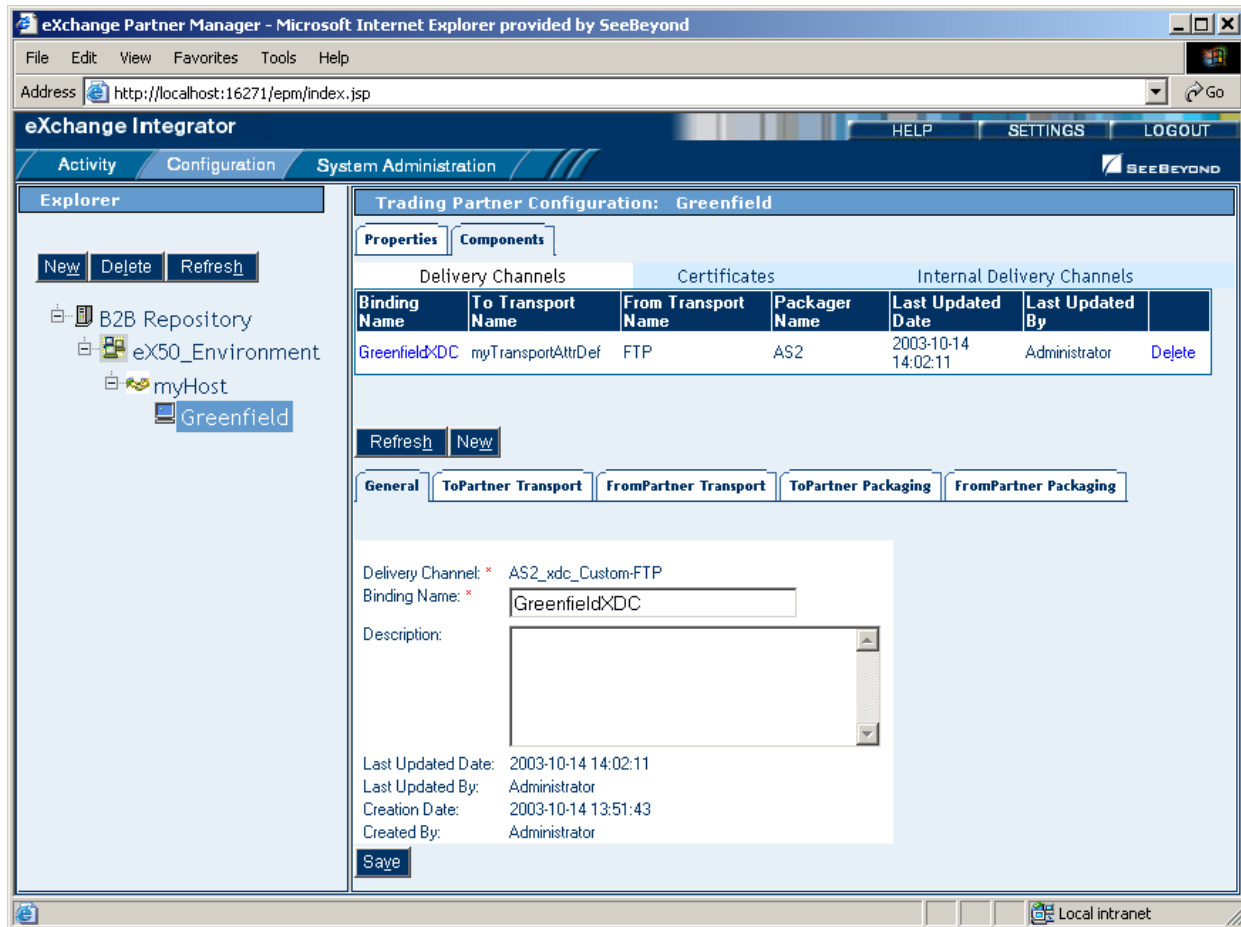


Creating Bindings to External Delivery Channels

To access an external delivery channel and create a binding to it

- 1 With a trading partner highlighted, in the configuration canvas, click **Components**. When the Components tab becomes active, it displays three subtabs.
- 2 With **Delivery Channels** active (the first subtab), click **New** (lower center button).
- 3 Select a delivery channel from the list, enter a binding name, and click **Continue**. Five new subtabs are displayed.

Figure 45 Trading Partner Newly Associated with an External Delivery Channel



You will need to provide specific parameter values to be used by the transport and enveloping (=packaging) protocols.

- 4 Click the **ToPartner** Transport tab and enter appropriate transport protocol values. The number and type of parameters depends on the particular transport protocol you are using. For example, a File transport protocol might require only three values, an FTP transport protocol might require six and permit twelve, and a protocol that uses user-defined attributes can require hundreds of values, or none.
- 5 When you have entered all values, click **Save**.
- 6 Repeat the previous two steps for the other three tabs (FromPartner Transport, ToPartner Packaging, and FromPartner Packaging). Be sure to click **Save** each time.
- 7 Optionally, if you are using Secure Messaging Exchange (SME), you can import an Encryption Key in the ToPartner Packaging tab and import a Signature certificate in the FromPartner Packaging tab.

To import a signature certificate or encryption key

- 1 Do one of the following:
 - ♦ In a Components Delivery Channels > []Partner Packaging tab, click **Import**.

- ♦ In the Components >**Certificates** tab, click **New**.
- 2 Enter a value for **Certificate Name.**, click **Browse**, locate and select the correct certificate, and then click **Open** to choose the file and return.
- 3 Click **Save**.

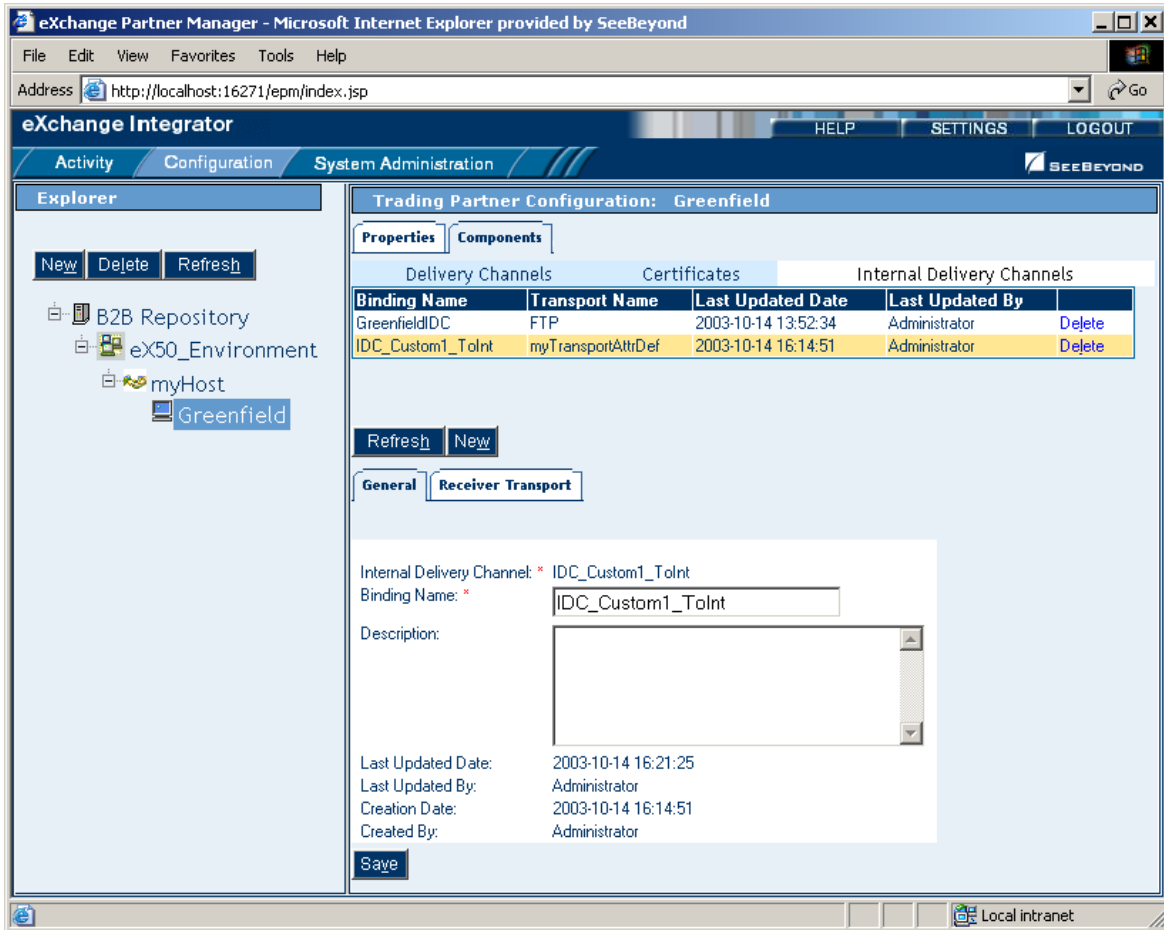
The imported certificate is saved. It is now displayed both under the **Certificates** tab and in the drop-down list.

Creating Bindings to Internal Delivery Channels

To access an internal delivery channel and create a binding to it

- 1 With a trading partner highlighted, in the configuration canvas, click **Components**.
When the Components tab becomes active, it displays three subtabs.
- 2 With **Internal Delivery Channels** active (rightmost tab), click **New** (lower center).
- 3 Select an IDC from the list, accept or enter a binding name, and click **Continue**.
Two new subtabs are displayed—in addition to **General**, one or the other of:
 - ♦ **Sender Transport** (for IDCs that the B2B host designated as From Internal)
 - ♦ **Receiver Transport** (for IDCs that the B2B host designated as To Internal)
- 4 Click the Sender or Receiver tab and supply values for that transport protocol.
- 5 Click **Save**. See Figure 46.

Figure 46 Trading Partner Newly Associated with an External Delivery Channel

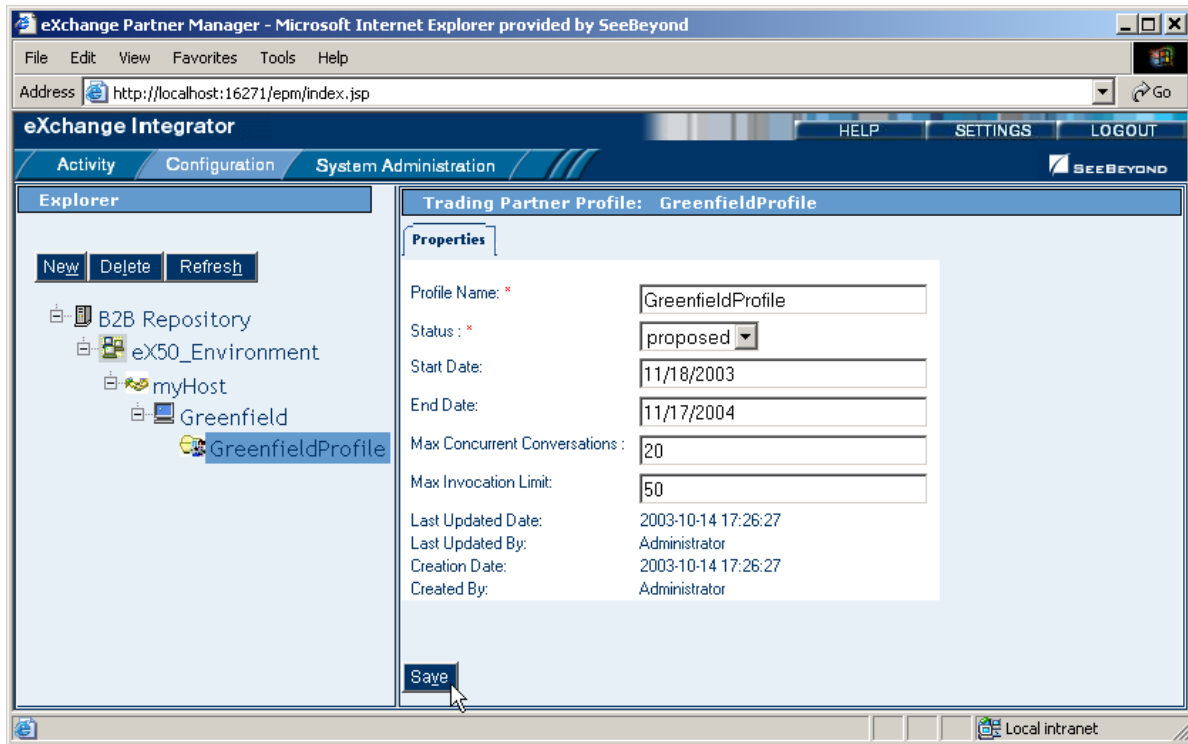


Creating Profiles and Activating Trading Partners

To create a trading partner profile and activate the trading partner

- 1 With a trading partner highlighted, click **New** (upper left).
- 2 Enter a profile name, select a status (if not **proposed**), and optionally enter a start date, end date, and maximum values concurrent conversations and invocations.
- 3 Click **Save**. See Figure 47.

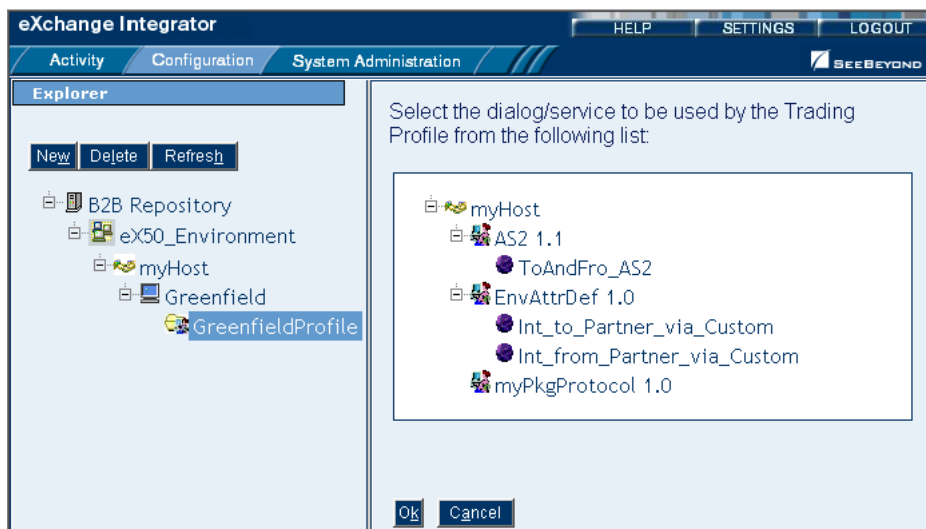
Figure 47 Newly Created Trading Partner Profile



The explorer tree displays a new trading partner *profile*.

- 4 In the explorer tree, click the newly created profile, and then click **New** (upper left). You are creating a new business protocol binding based on one of the business protocols that are the leaves of the tree now displayed on the canvas. See Figure 48.

Figure 48 Business Protocols Organized Under Enveloping Protocols



- 5 Click one of the business protocols and then click **OK**.

The business protocol appears as the bottommost leaf of the explorer tree, under the enveloping protocol (standard or custom) with which it is associated.

- 6 Click **Save**, and then click the **Dialogs** tab to display the actions that you defined for that business protocol.
- 7 For the actions, assign appropriate parameter values and click **Save**. See Figure 49.

Figure 49 Business Protocol in Explorer Tree with Actions Shown on Canvas



- 8 In the explorer tree, click the trading partner to display the Trading Partner Configuration canvas.
- 9 Click **Activate**. In response to the prompt, click **Activate**.

A success message confirms that the trading partner configuration is now in the database, and accessible to Channel Manager and other partner lookup facilities.

7.2 Monitoring B2B Protocol Pipelines

You use standard eGate tools within Enterprise Manager to monitor your B2B protocol pipelines.

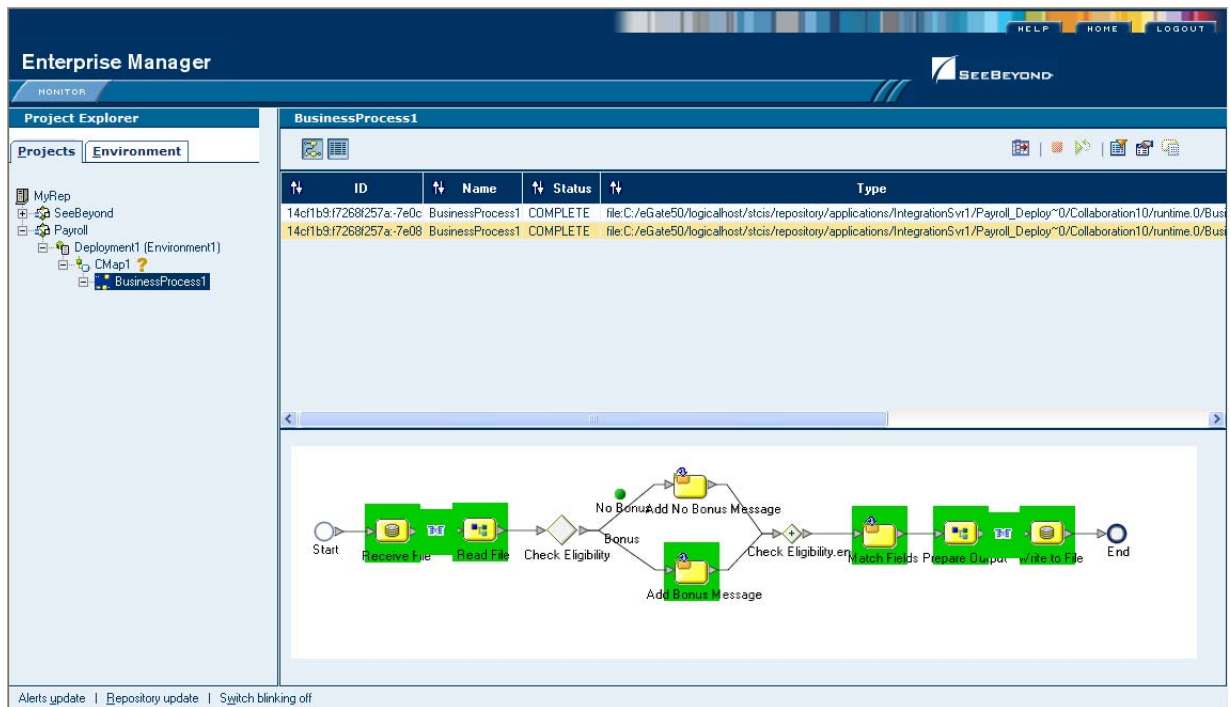
Before you begin

- You must already have activated a project whose connectivity map contains one or more services for B2B protocol pipelines.
- You must already have bootstrapped a logical host to run this project.

To monitor a B2B protocol pipeline

- 1 Start **Enterprise Manager**, and click the **Home** tab.
- 2 Select the **Monitor** icon to bring up the tree structure which allows you to navigate through projects or environments.
- 3 Select the **Projects** tab.
- 4 Navigate to the correct **Project/Deployment Profile/Connectivity Map**, and select the protocol pipeline name.

Figure 50 Monitor View



Monitor options

In the right pane of Enterprise Manager, you can see the model and/or the instance list (the toggle at the top of the pane controls the view).

The options are:

- show the model
- show the instance list
- show the instance list and model

7.3 Message Tracking

eXchange provides a special application, named MessageTracker, that allows you to monitor the status of messages as they are received and processed by eXchange.

Before you begin

- You must already have activated a project whose connectivity map contains one or more instances of the eXchange Tracker_Application.
- You must already have bootstrapped a logical host to run this project.
- For the facility to be useful, there must be one or more messages that have already been picked up by this logical host's integration server.

Accessing eXchange Message Tracking

To access Message Tracking

Before you begin: Your Repository server must already be running.

- 1 Open a browser window and point it at the hostname and port where your Repository server is running. For example:
`http://myBox:16271`
- 2 Log in to Enterprise Manager.
- 3 When you have logged in, point your browser at a new URL, substituting the port number where your integration server connects to the Web (default 18003; to learn this, see Properties > IS Configuration > Sections > Web Container Configuration) and appending this **string /Tracker_Application1/msgTrack/EnterMsgTrack.do**
For example:

`http://myBox:18003/Tracker_Application1/msgTrack/EnterPkgTrack.do`

Implementation Scenarios

This chapter provides step-by-step scenarios providing a sample of how eXchange can be used to achieve B2B solutions. It is assumed that you have already downloaded the documentation and sample files supplied with eXchange.

In this chapter

- Explanation of the scenario—see **“Overview”** on this page.
- Initial preparation—see **“Configuring the B2B Host and Trading Partner Profiles” on page 82.**
- Design and run the scenario for Greenfield, using the prebuilt protocol pipeline for AS2—see **“AS2 Design-Time Scenario for Greenfield” on page 110.**

Overview

Our company, Bork International, has eBusiness relationships with trading partners around the world. This exercise focuses on exchanging messages with two partners in particular:

- The Greenfield Group, which uses secure messaging with HTTP transport and AS2 enveloping.
- Johnson Limited, which requires custom transport and enveloping protocols.

In the exercises that follow, we will configure a B2B host with several different delivery channels; for secure messaging exchange, all outbound channels will use encryption, all inbound channels will use decryption, and all messages will be signed.

This one B2B host will participate in five projects:

- **projBorkConfigTest** sets up and activates the B2B host (BorkHost), and then sets up and activates trading partners.
- **AS2GreenfieldOut** and **AS2GreenfieldIn** use standard AS2 transport protocols for exchanging messages with the Greenfield trading partner.
- **CustomJohnsonOut** and **CustomJohnsonIn** use custom protocols for exchanging messages with the Johnson trading partner.

The repository in these scenarios is named **myRepository**.

8.1 Configuring the B2B Host and Trading Partner Profiles

In this section

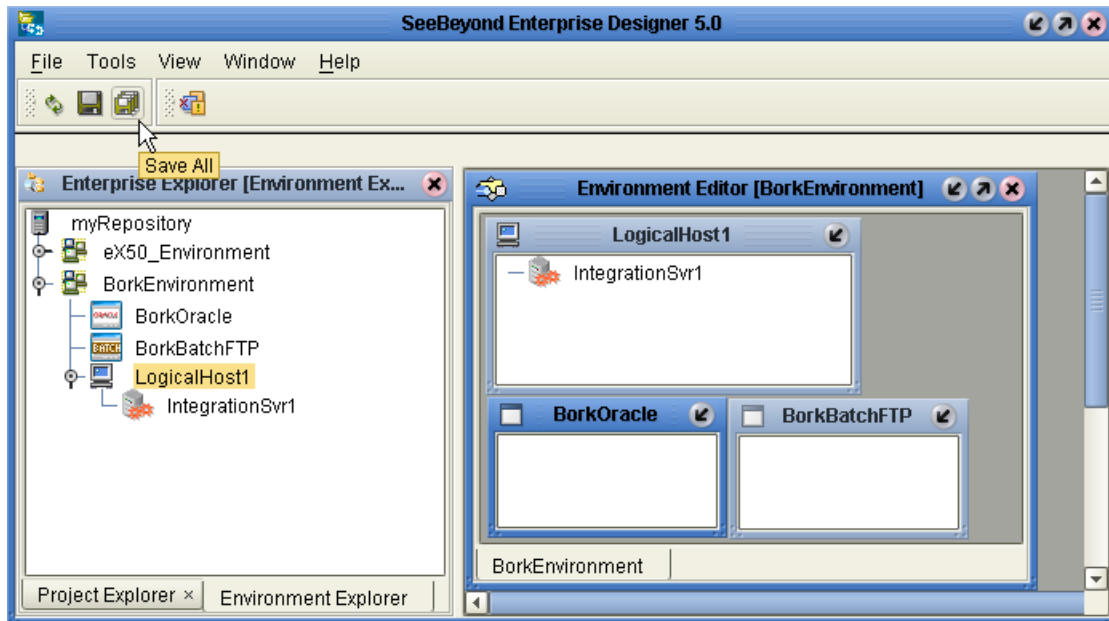
- You will create an environment with servers for external systems, and a logical host with an integration server —see [“Setting Up the Environment”](#) below.
- You will define the custom attribute definitions that will be used by one of your trading partners (Johnson) —see [“Setting Up Attribute Definitions for Protocols” on page 83](#).
- You will create business protocols defining the choreography of message exchange between your company (Bork) and each partner (first Greenfield, then Johnson)—see [“Setting Up the Business Protocols” on page 85](#).
- You will create the B2B Host for Bork—see [“Setting Up the B2B Host” on page 87](#).
- You will create and configure a trading partner profile for each trading partner—see [“Setting Up Trading Partners for BorkHost Using ePM” on page 99](#).

8.1.1. Setting Up the Environment

- 1 In Enterprise Designer, click the **Environment Explorer** tab (at the bottom margin).
- 2 In Environment Explorer: Right-click the repository and, on the popup menu, click **New Environment**. Name the environment: **BorkEnvironment**.
- 3 Right-click BorkEnvironment and add an Oracle external system. Name it **BorkOracle** and designate it Outbound. Configure properties, such as servername, portnumber **1521** datasourcename **local**, databasename, username, and password, appropriately for the Oracle database instance at your site (see [“Creating and Configuring the eXchange Database Instance” on page 29](#)), and click **OK**.
- 4 Right-click BorkEnvironment and add a new BatchFTP external system. Name it BorkBatchFTP. Configure its properties for the **FTP** section (host name, server port, user name, and password) appropriately for using FTP at your site, and click **OK**.
- 5 Right-click BorkEnvironment and add a new logical host. Keep the default name (**LogicalHost1**).
- 6 Right-click LogicalHost1 and add a new integration server. Keep the default name (**IntegrationSvr1**).
- 7 Click **Save All**.

You have completed initial setup of your environment. See Figure 51.

Figure 51 Preliminary Setup of BorkEnvironment



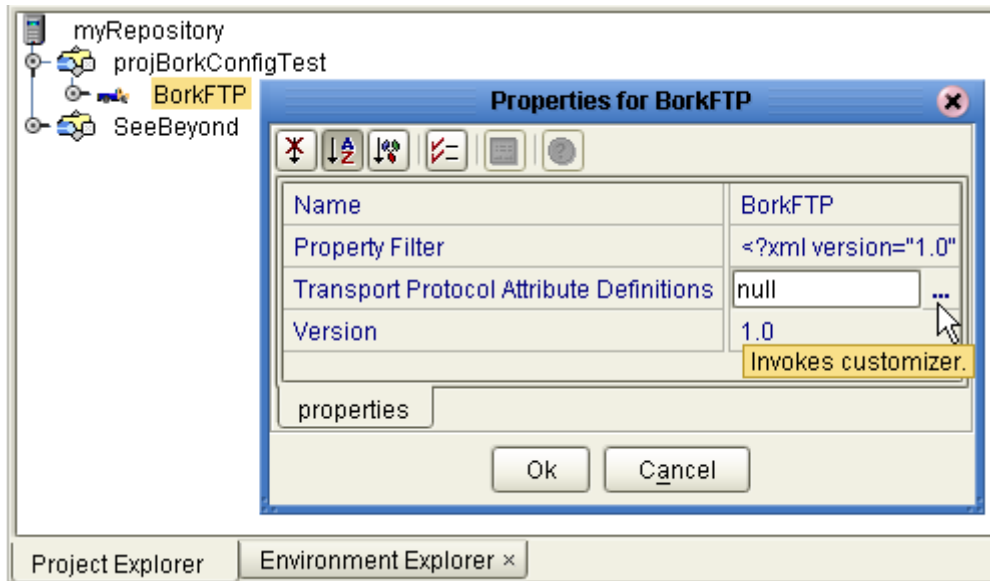
8.1.2. Setting Up Attribute Definitions for Protocols

Note: You need not set up attribute definitions for HTTP (transport protocol) or AS2 (enveloping protocol) that will be used by Greenfield, since they are already supplied under the SeeBeyond eXchange project folder. The following steps set up custom attribute definitions for protocols used by Johnson.

To set up the transport attribute definitions for Johnson

- 1 In Enterprise Designer, click the Project Explorer tab.
- 2 In Project Explorer: Right-click the repository and, on the popup menu, click **New Project**. Name the project: **projBorkConfigTest**.
- 3 In the project tree, right-click projBorkConfigTest, point at **New**, and click **B2B Transport Attributes Definition**. Name the component: **BorkFTP**.
- 4 Right-click BorkFTP and, on the popup menu, click Properties. The Properties dialog box appears. See Figure 52.

Figure 52 Properties of Transport Attribute Definition BorkFTP



- 5 To the far right of the value for Transport Protocol Attribute Definitions, click the ellipsis [...]. The Extended Attribute Definitions dialog box appears.
- 6 Click the **Add** button thirteen times to add thirteen generic rows, and then edit them so that they contain the values shown in Figure 53.

Important: For this scenario, enter all values for **Name** exactly as you see them in Figure 53. The sample files require these exact node names in the OTD you will be generating.

Figure 53 Transport Protocol Attribute Definitions for BorkFTP

Name	Display	Type	Required	Direction	Default
UserName	User name	String	<input checked="" type="checkbox"/>	Both	
Password	Password	String	<input checked="" type="checkbox"/>	Both	
HostName	Host name	String	<input checked="" type="checkbox"/>	Both	
PortNum	Port number	String	<input checked="" type="checkbox"/>	Both	21
Directory	Target directory	String	<input checked="" type="checkbox"/>	Both	
FilePattern	Target file pattern	String	<input checked="" type="checkbox"/>	Both	
PollMilliSeconds	Poll interval (in milliseconds)	String	<input checked="" type="checkbox"/>	Both	
SocksEnabled	Is Socks Enabled?	String	<input type="checkbox"/>	Both	
SocksHostName	Socks host name	String	<input type="checkbox"/>	Both	
SocksUserName	Socks user name	String	<input type="checkbox"/>	Both	
SocksPassword	Socks password	String	<input type="checkbox"/>	Both	
SocksServerPort	Socks server port	String	<input type="checkbox"/>	Both	
ChannelManagerMode	Channel Manager mode	String	<input checked="" type="checkbox"/>	Both	

- 7 Click OK, and then OK again, to close the dialog boxes for attributes and properties.
- 8 Right-click BorkFTP and, on the popup menu, click **Generate OTD**. The **BorkFTPOTD_BorkFTP** OTD is generated in the projBorkConfigTest project.
- 9 Right-click BorkFTP and, on the popup menu, click **Check In**. In the Version Control dialog box, add a comment and then click OK.



To set up the enveloping attribute definitions for Johnson

Note: The icon for enveloping protocols suggests a brown package. (Enveloping protocols are sometimes also called **packaging protocols** or **messaging protocols**.)

- 1 In Project Explorer, right-click projBorkConfigTest > **New > B2B Enveloping Attributes Definition**. Name the component: **BorkCustomMsgProtocol**
A new enveloping attributes definition object is created, as is a new subproject.
- 2 Right-click BorkCustomMsgProtocol and, on the popup menu, click **Properties**. Close the dialog by clicking **OK**. (This step is needed to initialize the properties.)
- 3 Right-click BorkCustomMsgProtocol > **Check In**. In the Version Control dialog box, add a comment and then click OK.

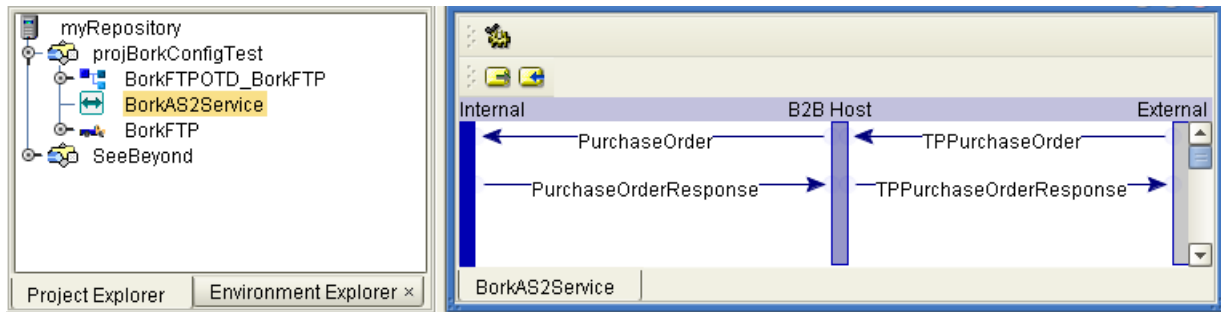
8.1.3. Setting Up the Business Protocols


To set up the AS2 business protocol for Greenfield: BorkAS2Service

- 1 In Project Explorer, right-click projBorkConfigTest > **New > B2B Business Protocol**. Name the protocol: **BorkAS2Service**
The Dialog Designer displays a blank canvas.
- 2 Click the  **message_in** action icon (the second one, with a blue **left**-pointing arrow). Two labeled arrows appear on the canvas, both pointing leftward.
 - ♦ Double-click the arrow on the right (to B2B Host from External). Relabel it (from message_in_0) to: **TPPurchaseOrder**.
 - ♦ Double-click the arrow on the left (to Internal from B2B Host). Relabel it (from message_in_0*) to: **PurchaseOrder**.
- 3 Click the  **from_internal** action icon (the one with a gray **right**-pointing arrow). Two labeled arrows appear on the canvas, both pointing rightward.
 - ♦ Double-click to relabel the arrow on the left (from Internal to B2B Host): **PurchaseOrderResponse**.
 - ♦ Double-click to relabel the arrow on the right (from B2B Host to External): **TPPurchaseOrderResponse**.

Note: Pay careful attention to the sequence: The placement of the leftward arrows above the rightward arrows means that Bork (the Internal system) first accepts POs from trading partners (External) and then responds to them. See Figure 54.

Figure 54 Business Protocol for BorkAS2Service



- 4 Right-click BorkAS2Service > **Properties**. In the Properties dialog box, in the list box for B2B Messaging Protocol, click **AS2**, and then click **OK** to close the dialog box.
- 5 On the Dialog Designer canvas, in the upper left corner, click  **Compile** (the icon suggests a checkmark over a gear).
Project Explorer displays two new service actions below BorkAS2Service.
- 6 Close the Dialog Designer canvas, saving changes.
- 7 Right-click BorkAS2Service > **Check In**. In the Version Control dialog box, add a comment and then click OK.

To set up the custom business protocol for Johnson: BorkMsgService




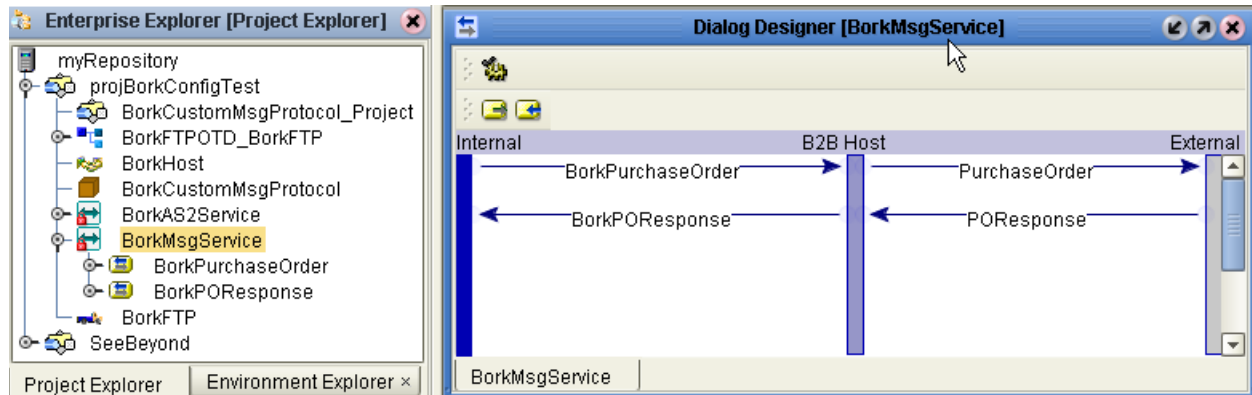
- 1 Right-click projBorkConfigTest > New > **B2B Business Protocol**. Name the protocol: **BorkMsgService**.
- 2 Click the  **from_internal** action icon (the one with a gray **right**-pointing arrow). Two labeled arrows appear on the canvas, both pointing rightward.
 - ♦ Double-click to relabel the arrow on the left (from Internal to B2B Host): **BorkPurchaseOrder**.
 - ♦ Double-click to relabel the arrow on the right (from B2B Host to External): **PurchaseOrder**.
- 3 Click the  **message_in** action icon (the second one, with a blue **left**-pointing arrow). Two labeled arrows appear on the canvas, both pointing leftward.
 - ♦ Double-click the arrow on the right (to B2B Host from External). Relabel it: **POResponse**.
 - ♦ Double-click the arrow on the left (to Internal from B2B Host). Relabel it: **BorkPOResponse**.
- 4 Right-click BorkMsgService > **Properties**. In the Properties dialog box, in the list box for B2B Messaging Protocol, click **BorkCustomMsgProtocol**, and then click **OK** to close the dialog box.
- 5 On the Dialog Designer canvas, in the upper left corner, click  **Compile**.
Project Explorer displays two new service actions below BorkMsgService. See Figure 55.

Figure 55 Project Tree with Both Business Protocols Checked In



- 6 Close the Dialog Designer canvas, saving changes.
- 7 Right-click BorkMsgService > **Check In**. In the Version Control dialog box, add a comment and then click OK.

Your project now has access to the following protocols.

- **Transport protocols:** In addition to the standard transport protocols presupplied by SeeBeyond (such as HTTP, which will be used by Greenfield), your project contains a custom transport attribute definition for Johnson, named BorkFTP.
- **Enveloping protocols** (also called *packaging* or *messaging* protocols): In addition to the standard enveloping protocols presupplied by SeeBeyond (such as AS2, which will be used by Greenfield), your project contains a custom enveloping attribute definition for Johnson, named BorkCustomMsgProtocol.
- **Business protocols** (also called messaging *services*): The BorkAS2Service business protocol, which references the AS2 enveloping protocol, will be used for Greenfield; and the BorkMsgService business protocol, which references the custom enveloping protocol, will be used for Johnson.

You are now ready to use these protocols to set up delivery channels for a B2B Host.

8.1.4. Setting Up the B2B Host

B2B hosts constitute the central nervous system of eXchange. It is in the B2B host that external and internal delivery channels are defined.

- Each external delivery channel combines a configured enveloping protocol with references to two transport protocols (one for each direction, ToPartner and FromPartner). Every trading partner must have at least one external delivery channel, and may have many. In the sample, however, each trading partner uses only one external delivery channel: Greenfield combines AS2 with HTTP, while Johnson combines a custom enveloping protocol with a custom transport protocol.
- Each internal delivery channel (IDC) combines a transport protocol and a direction, either Sender=FromInternal or Receiver=ToInternal. IDCs are optional; in the sample, for instance, the Greenfield trading partner does not use an IDC.

The B2B host functions both as a logical object (that is, an item that can be placed on a connectivity map) and also as a physical server (that is, a host for Channel Managers). In its role as a server, it has a dynamic dependency on the Oracle database instance, and requires preactivation. After preactivation, as a server in an environment, it can be configured with trust stores, keystores, and certificates.

Thus, the following procedures are needed to fully set up a B2B host:

- Create the B2B host and populate it with enveloping and business protocols.
- Create and configure external delivery channels and internal delivery channels.
- Create and activate a connectivity map binding the B2B host to an Oracle external.
- Configure crypto parameters for the B2B host server in the environment.
- Use ePM to validate the successful creation and preactivation of the B2B host.

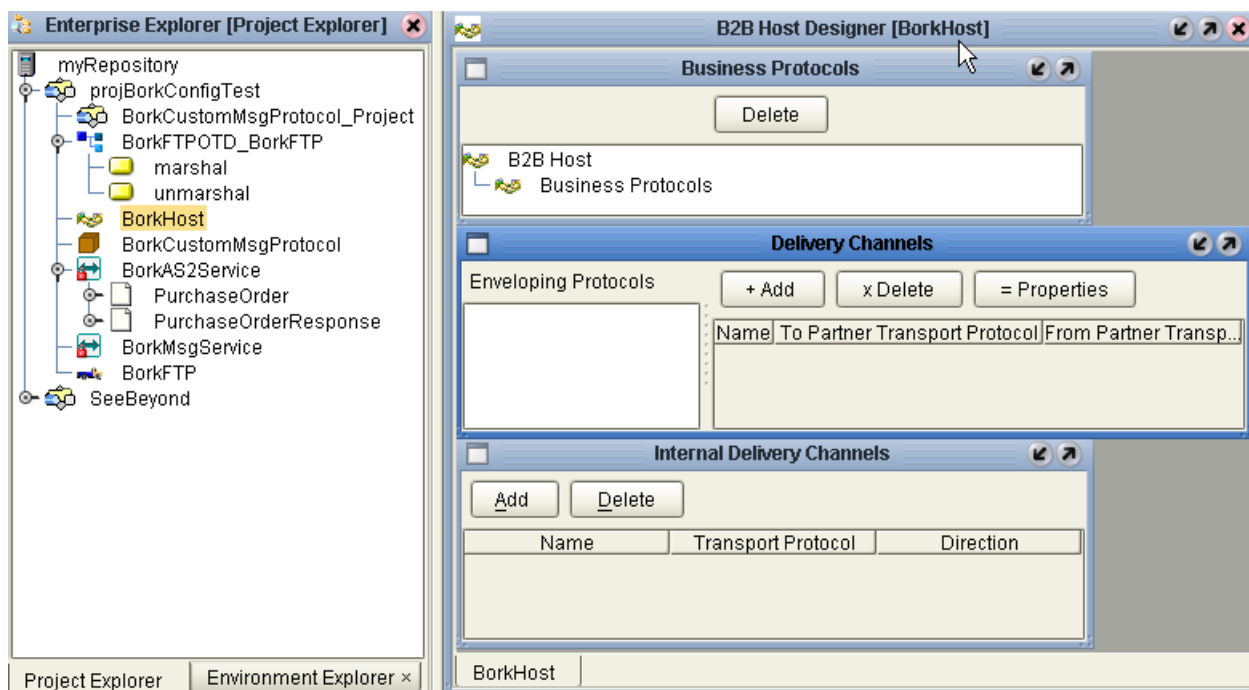
For our B2B host, **BorkHost**, steps for each of these procedures are provided below.

To create the B2B host and populate it with business protocols

- 1 In Project Explorer, right-click projBorkConfigTest > New > **B2B Host**. Name it: **BorkHost**

The Host Designer displays three blank windows, where you will specify business protocols, external delivery channels, and internal delivery channels. See Figure 56.

Figure 56 B2B Host Designer

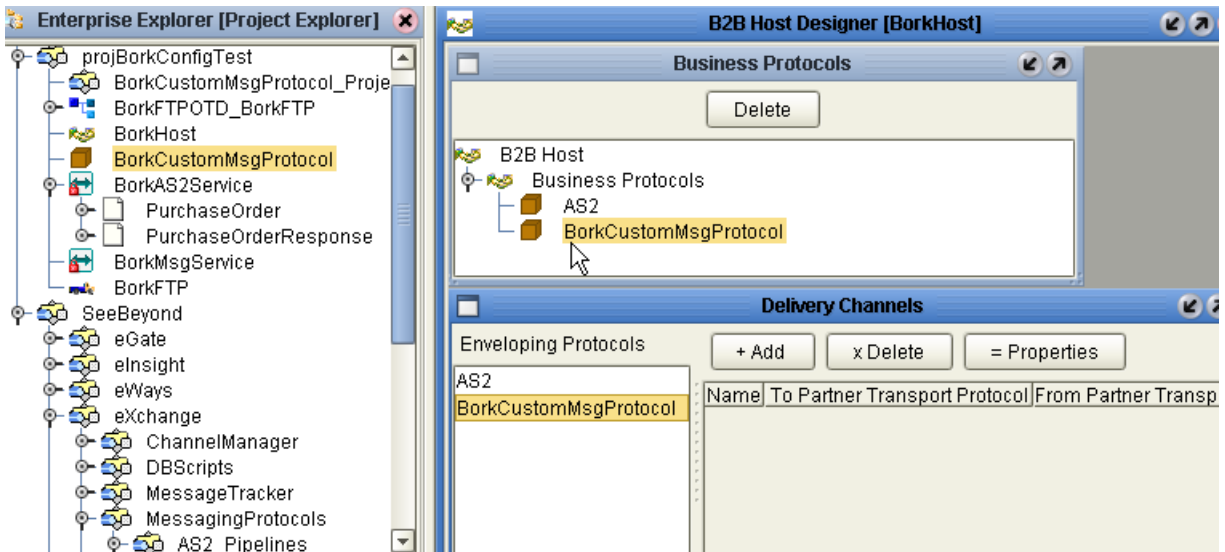


- 2 Because business protocols are organized according to the enveloping protocol they reference, you first need to populate the Business Protocols window with the enveloping protocols your B2B host will be using, as follows:

- A For Greenfield (AS2): In the project tree, expand the SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines folder and drag the **AS2** enveloping protocol to the canvas, into the Business Protocols window, and drop it under **Business Protocols**.
- B For Johnson (custom): In the project tree, under projBorkConfigTest, drag the **BorkCustomMsgProtocol** enveloping protocol into the Business Protocols window and drop it under **Business Protocols**.

Both enveloping protocols also appear on the left side of the Delivery Channels window. See Figure 57.

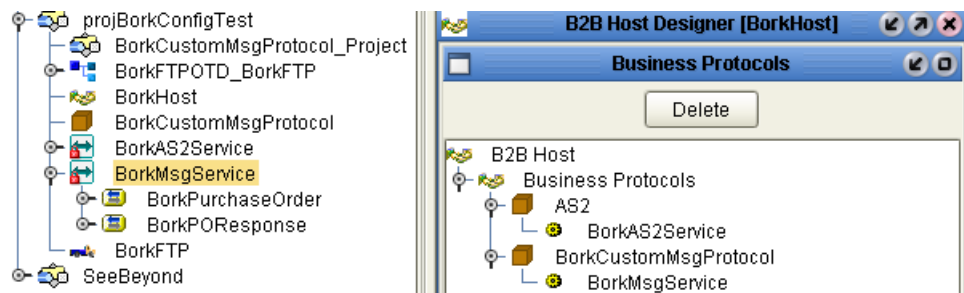
Figure 57 B2B Host with Enveloping Protocols



- 3 Populate each enveloping protocol with appropriate business protocols, as follows:
 - A For Greenfield (AS2): From the project tree, drag the BorkAS2Service business protocol to the canvas and drop it under **AS2**.
 - B For Johnson (custom): From the project tree, drag the BorkMsgService business protocol to the canvas and drop it under **BorkCustomMsgProtocol**.

The business protocols appear in the Business Protocols tree. See Figure 58.

Figure 58 B2B Host with Business Protocols



- 4 Save your changes.

The sample uses only one business protocol for each of the two enveloping protocols (other B2B hosts might use hundreds of business protocols per enveloping protocol), and so you are ready to create and configure external and internal delivery channels.

To create and configure the external delivery channel for Greenfield (AS2+HTTP)

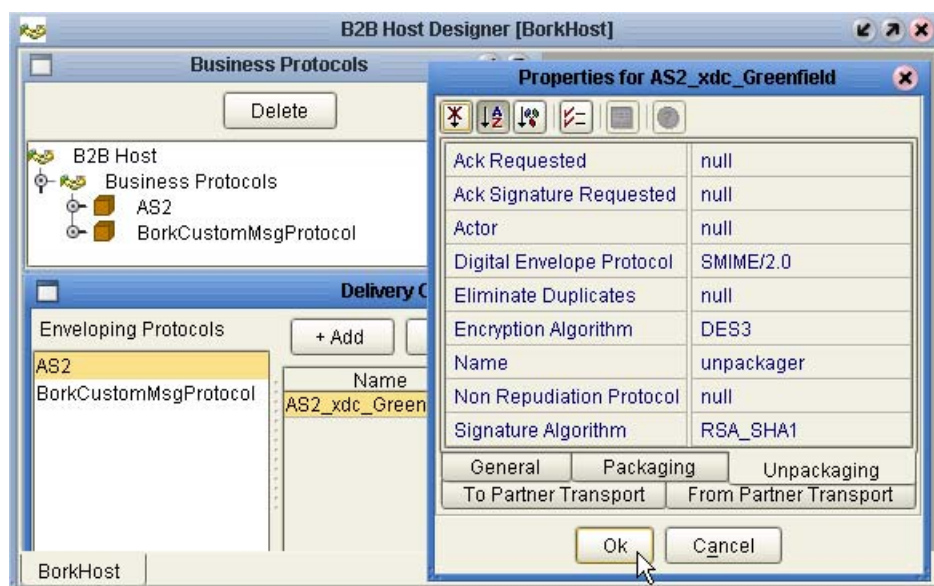
The Greenfield trading partner requires only one simple external delivery channel, which combines the AS2 enveloping protocol with the HTTP transport protocol for both outbound (ToPartner Transport) and inbound (FromPartner Transport).

- 1 If necessary, double-click BorkHost to open the B2B Host Designer editor.
- 2 In the **Delivery Channels** window, in Enveloping Protocols list, click **AS2**. Then, on the right side of the window, click the **Add** button.

A new delivery channel appears, with a default name and no transport protocols.

- 3 Click the new delivery channel, click **Properties**, and set the general properties:
 - ◆ For Name, enter: **AS2_xdc_Greenfield**
 - ◆ For ToPartner Transport Protocol, choose: **HTTP**
 - ◆ For FromPartner Transport Protocol, choose: **HTTP**
 - ◆ For RetryInterval, choose: **PT2M**
- 4 Click the **Packaging** tab and set the packaging (enveloping) properties:
 - ◆ For Digital Envelope Protocol, choose: **SMIME/2.0**
 - ◆ For Encryption Algorithm, choose: **DES3**
 - ◆ For Signature Algorithm, choose: **RSA_SHA1**
- 5 Click the **Unpackaging** tab and set the unpackaging (de-enveloping) properties as you did in the previous step. When you are satisfied all values are correct, click **OK** to close the properties sheet. See Figure 59.

Figure 59 Property Sheet for External Delivery Channel AS2_xdc_Greenfield



To create and configure the internal delivery channel for Greenfield (FILE, fromInternal)

The Greenfield sample uses the presupplied AS2OutboundBP protocol pipeline. Because this pipeline uses a Channel Manager, you need to provide the B2B host with an internal delivery channel for the AS2 transport protocol.

- 1 If necessary, double-click BorkHost to open the B2B Host Designer editor.
- 2 In the **Internal Delivery Channels** window, click the **Add** button.
A new IDC appears, with a default name and no transport protocol or direction.
- 3 Click the new IDC and change its three values to the following:
 - ♦ For Name, enter: **Greenfield_idc_fromInternal**
 - ♦ For Transport Protocol, choose: **FILE**
 - ♦ For Direction, choose: **Receiver**
- 4 Repeat the previous two steps to create an IDC named **Greenfield_idc_toInternal**, also using the **FILE** protocol, but in the **Sender** direction.

You have now finished configuring the Greenfield channels using standard protocols. In the next procedure, you configure the Johnson channel using custom protocols.

To create and configure the delivery channels for Johnson (custom)

The Johnson trading partner requires only one external delivery channel, which combines the custom enveloping protocol BorkCustomMsgProtocol with the custom transport protocol BorkFTP, for both outbound (ToPartner Transport) and inbound (FromPartner Transport).

- 1 If necessary, double-click BorkHost to open the B2B Host Designer editor.
- 2 In **Delivery Channels**, for Enveloping Protocol, click **BorkCustomMsgProtocol**. Then, on the right side of the window, click the **Add** button.
A new delivery channel appears, with a default name and no transport protocols.
- 3 Click the new delivery channel and change its three values to the following:
 - ♦ For Name, enter: **Custom_xdc_Johnson**
 - ♦ For ToPartner Transport Protocol, choose: **BorkFTP**
 - ♦ For FromPartner Transport Protocol, choose: **BorkFTP**
- 4 Click Properties and, in the **Packaging** and **Unpackaging** tabs, set the following:
 - ♦ For Digital Envelope Protocol, choose: **SMIME/2.0**
 - ♦ For Encryption Algorithm, choose: **DES3**
 - ♦ For Signature Algorithm, choose: **RSA_SHA1**
- 5 Save your changes, close all canvases, and check in the **BorkHost** object.

Now that you have finished configuring all delivery channels, BorkHost is now fully configured as a logical object in the project, and can be preactivated to create a server in the environment. If, in the future, changes occur in the project side of the B2B host, reactivation will be required to propagate the changes to the environment.

To create and populate a connectivity map and preactivate BorkHost

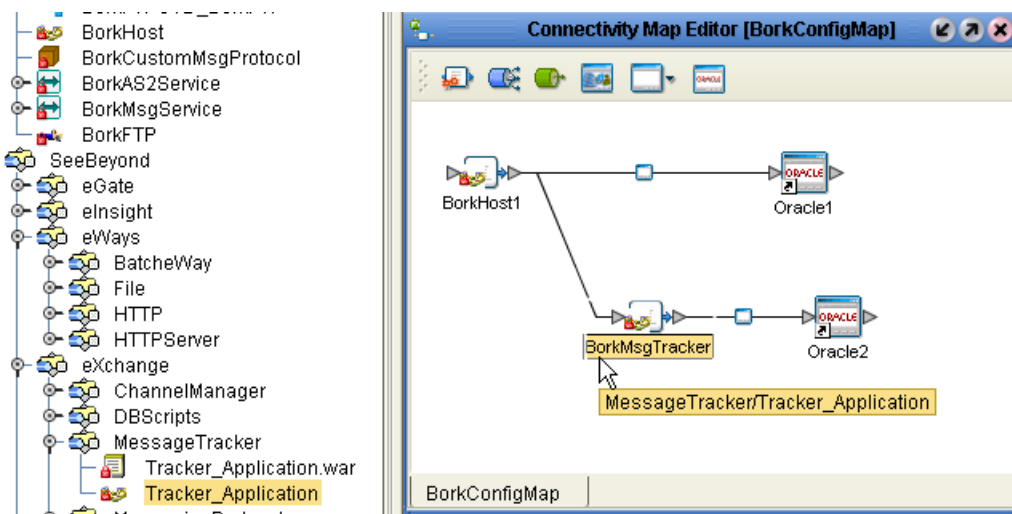
In these steps you will create a map, populate it with a B2B host, a MessageTracker application, and an Oracle external, create and configure connections, and create and activate a deployment profile for the map, thus preactivating the B2B host (BorkHost).

- 1 Right-click projBorkConfigTest > New > **Connectivity Map**. When the connectivity map appears, rename it to: **BorkConfigMap**
- 2 In the Connectivity Map toolbar, click the **External Applications** tool and, from the drop-down list, select the checkbox for **Oracle External Application**.
- 3 From the toolbar, drag two Oracle externals onto the canvas, towards the right side.

Note: The two Oracle externals hold trading partner data and message tracking data. If you wanted to ensure both data sets were served by the same database instance, you could connect both BorkHost and BorkMsgTracker to the same Oracle external.

- 4 From the project tree, drag BorkHost onto the upper left side of the canvas.
- 5 In the project tree, open the **SeeBeyond > eXchange > MessageTracker** folder and drag its **Tracker_Application** protocol pipeline onto the lower center of the canvas. Rename it to **BorkMsgTracker**
- 6 Connect as shown in Figure 60.

Figure 60 Connectivity Map for Preactivating BorkHost

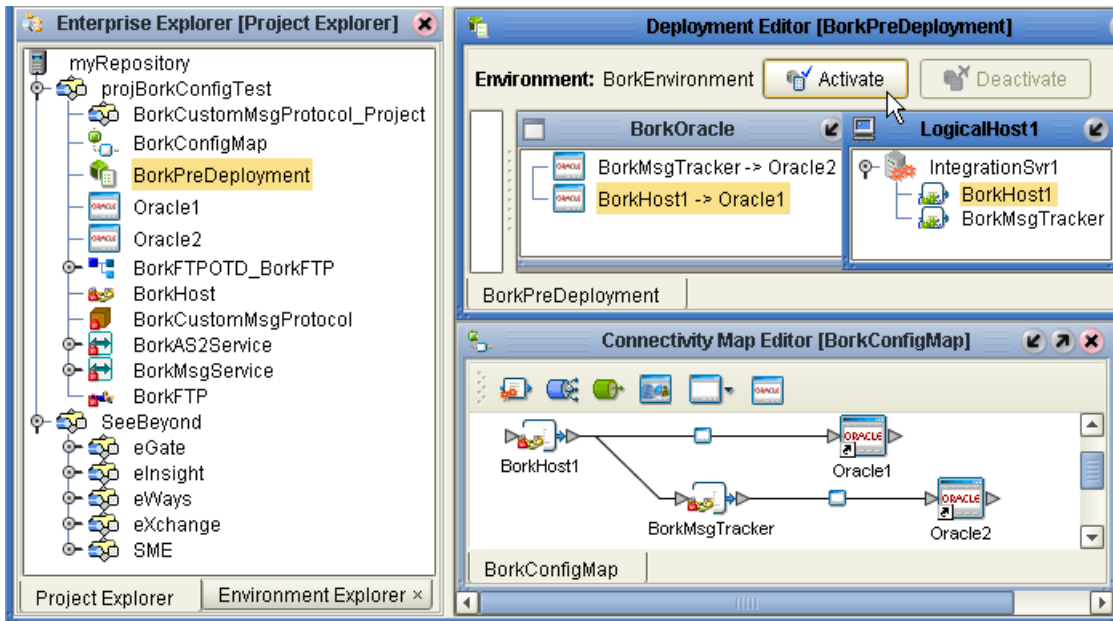


- 7 Configure both eWays, designating them outbound to external Oracle applications. When you are done, save your work.
- 8 Right-click projBorkConfigTest > New > **Deployment Profile** and name it: **BorkPreDeployment**. Point it at **BorkEnvironment**, and then click **OK**.

The Deployment Editor opens a canvas whose left pane shows all components on the map and whose right pane shows all external servers in the environment.

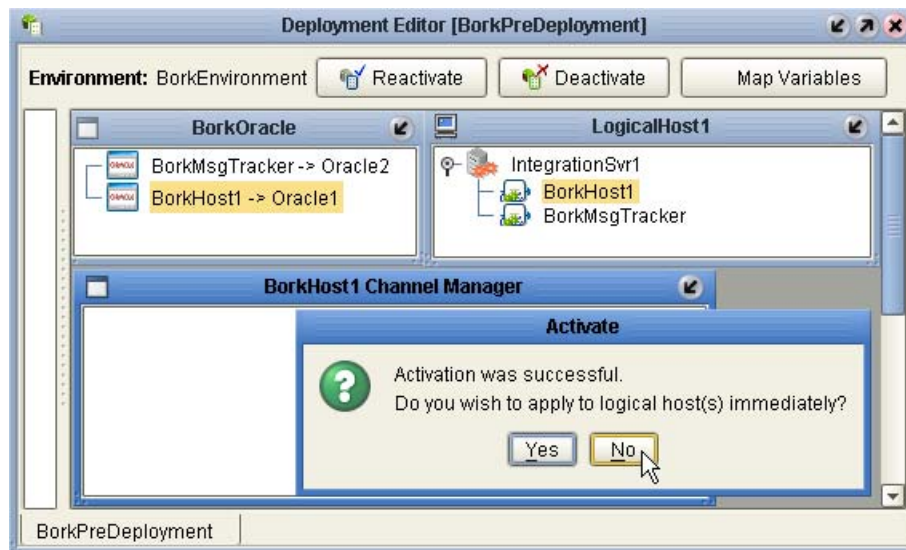
- 9 One by one, drag both Oracle eWays into the BorkOracle server. Then, one by one, drag BorkHost and BorkMsgTracker under the integration server. See Figure 61.

Figure 61 BorkPreDeployment Profile Ready for Activation




- 10 After assigning each of the map's logical components to a server, save your work.
- 11 Click **Activate**. An instance of BorkHost appears in the environment, ready to serve as a Channel Manager container.
- 12 In response to the question about applying to a logical host, click **No**. See Figure 62.

Figure 62 BorkPreDeployment Profile After Activation



Note: If, after successful activation, you make project-side changes to BorkHost, an Oracle eWay, or any other item on the connectivity map, this deployment profile must be reactivated in order to propagate the changes to the environment.

- 13 Save your changes, close all canvases, and then check in **BorkConfigMap** and **BorkPreDeployment**.
- 14 On the main toolbar, click  **Refresh All From Repository**.

You have now finished all project-side work for the BorkHost logical object. However, crypto information for B2B hosts is stored on the physical side, in the environment.

To configure crypto parameters in the environment for the BorkHost server

- 1 Click the **Environment** Explorer tab and, in the environment tree, right-click **BorkHost1 Channel Manager** and open its Properties.

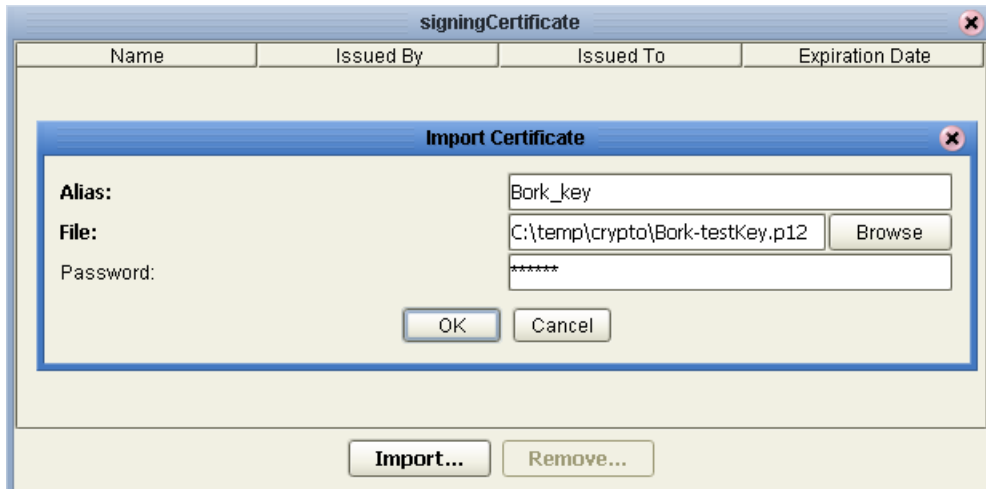
A dialog box appears, allowing you to define up to eight items for each external delivery channel.

Table 9 Certificates and Trust Stores for External Delivery Channels

Item	Protocol Level	Purpose
signingCertificate	enveloping	Outbound: To sign the payload with a public key. Inbound: To check the payload's public signature.
signingTrustStore	enveloping	Points at a truststore for verifying signatures.
encryptionCertificate	enveloping	Outbound: To encrypt the payload (private key). Inbound: To decrypt the payload.
encryptionTrustStore	enveloping	Points at a truststore for verifying crypto keys.
toTransportServerCertificate	transport	(not used)
toTransportClientCertificate	transport	(not used)
fromTransportServerCertificate	transport	(not used)
fromTransportClientCertificate	transport	(not used)

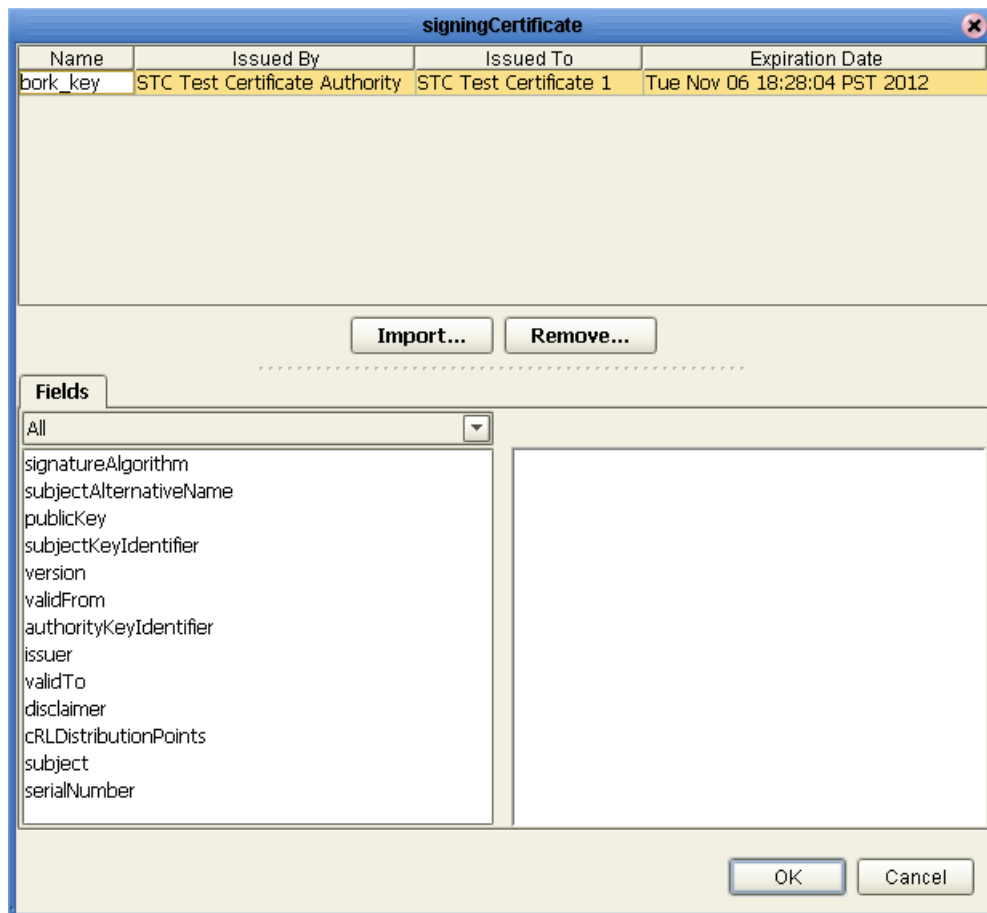
- 2 In the row for **AS2_xdc_Greenfield**, under **signingCertificate**, click the ellipsis [...] and, in the **signingCertificate** dialog, click **Import**. Then, in the **Import Certificate** dialog box (see Figure 63), supply the following values:
 - ♦ For Alias, enter: **bork_key**
 - ♦ For File, click Browse, navigate to the location where you extracted the sample files, and then select: **Bork-testKey.p12**
 - ♦ For Password, enter: **passwd**

Figure 63 Import Certificate Dialog Box for Greenfield (AS2) signingCertificate



3 Click **OK** and verify results. See Figure 64.

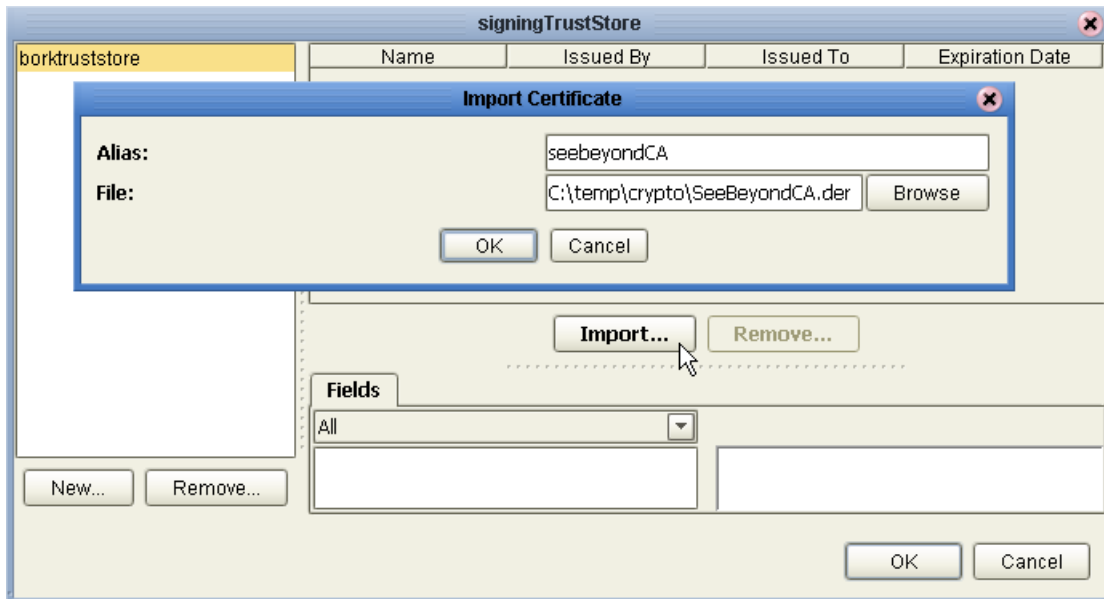
Figure 64 Imported "bork_key" signingCertificate



4 Click **OK** to close the **signingCertificate** dialog box.

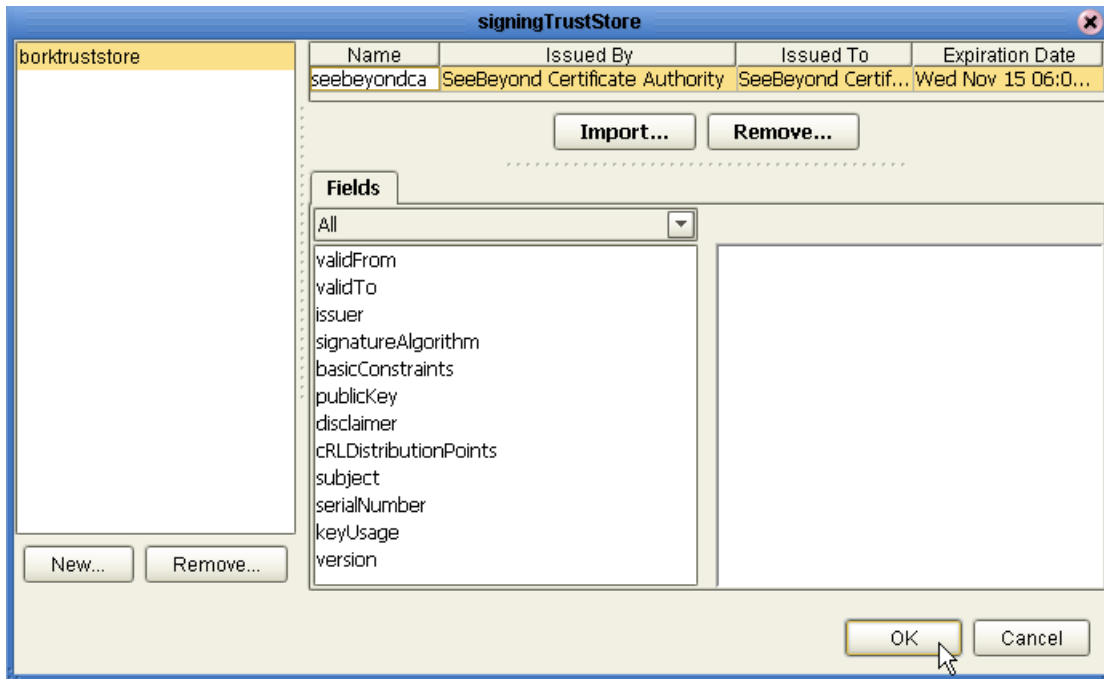
- 5 Still in the row for **AS2_xdc_Greenfield**, under **signingTrustStore**, click the ellipsis and, in the **signingTrustStore** dialog, click **New**.
- 6 In the **New TrustStore** dialog box, enter alias name **borktruststore** and click **OK**.
- 7 In the left list box, click the just-created truststore and click **Import**. Then, in the **Import Certificate** dialog box: for **Alias**, enter **seebeyondCA**; for **File**, browse to the location of the sample files and select **SeeBeyondCA.der**. See Figure 65.

Figure 65 Import Certificate Dialog Box for signingTrustStore



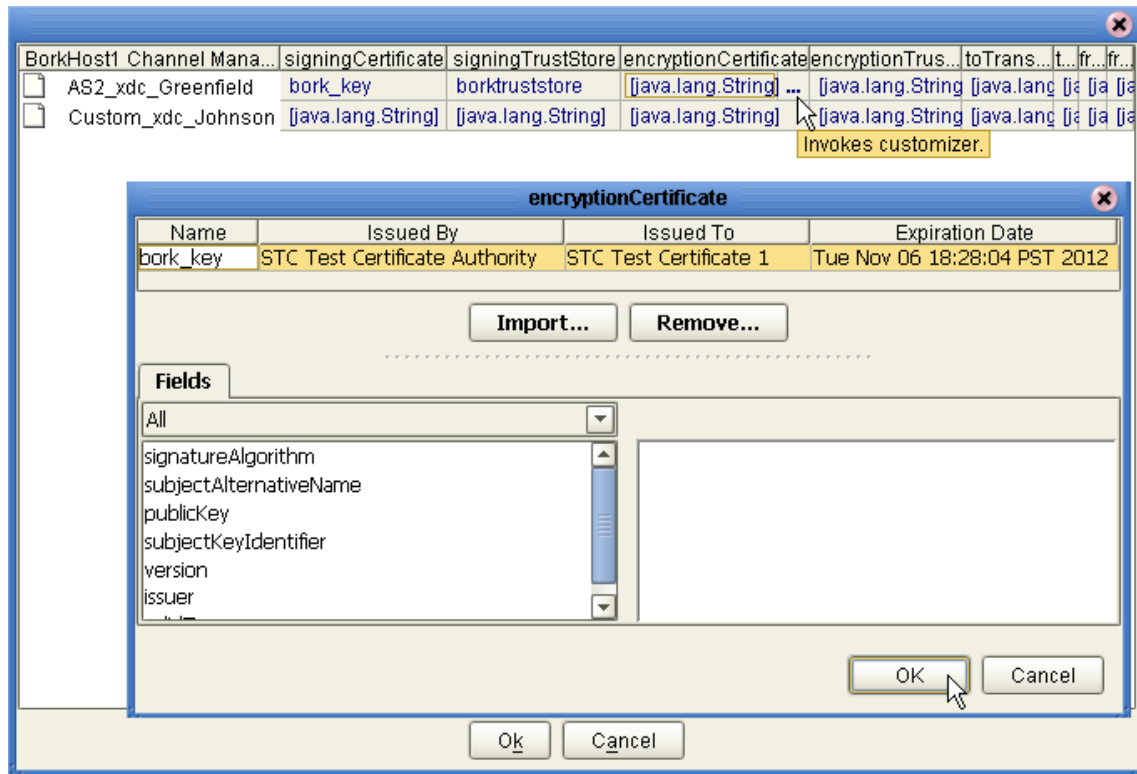
- 8 Click **OK** to import the certificate, and then verify the results. See Figure 66.

Figure 66 Imported “seebeyondca” signingTrustStore



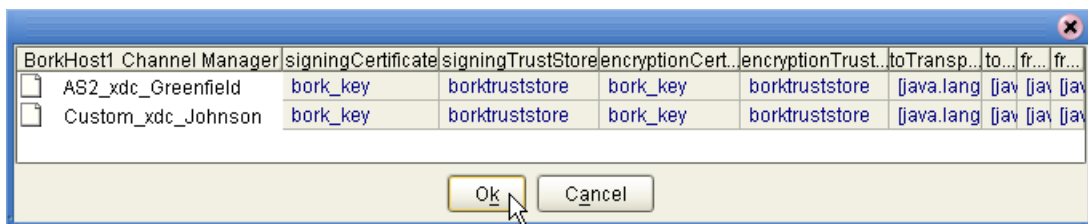
- 9 Click **OK** to close the **signingTrustStore** dialog box.
- 10 Still in the row for **AS2_xdc_Greenfield**, under **encryptionCertificate**, click the ellipsis and, in the **encryptionCertificate** dialog, click the existing certificate (bork_key) and click **OK**. See Figure 67.

Figure 67 Selecting an Existing Certificate



- 11 Still in the row for **AS2_xdc_Greenfield**, for **encryptionTrustStore**, select **borktruststore** (the same truststore you imported in an earlier step).
- 12 For the other delivery channel (**Custom_xdc_Johnson**), supply the same certificates and truststores as you did for Greenfield.

Figure 68 BorkHost Channels Both Configured with Certificates and TrustStores



- 13 After both channels are configured with certificates and truststores, click **OK**.

You have now completed all design work required for BorkHost. Before using it as a B2B host for the business solutions, you should double-check that the host is visible to eXchange Partner Manager (ePM), and then configure your trading partners.

8.1.5. Setting Up Trading Partners for BorkHost Using ePM

In these steps you will start eXchange Partner Manager (ePM) and use the BorkHost external and internal delivery channels to configure communications with two trading partners, Greenfield and Johnson.

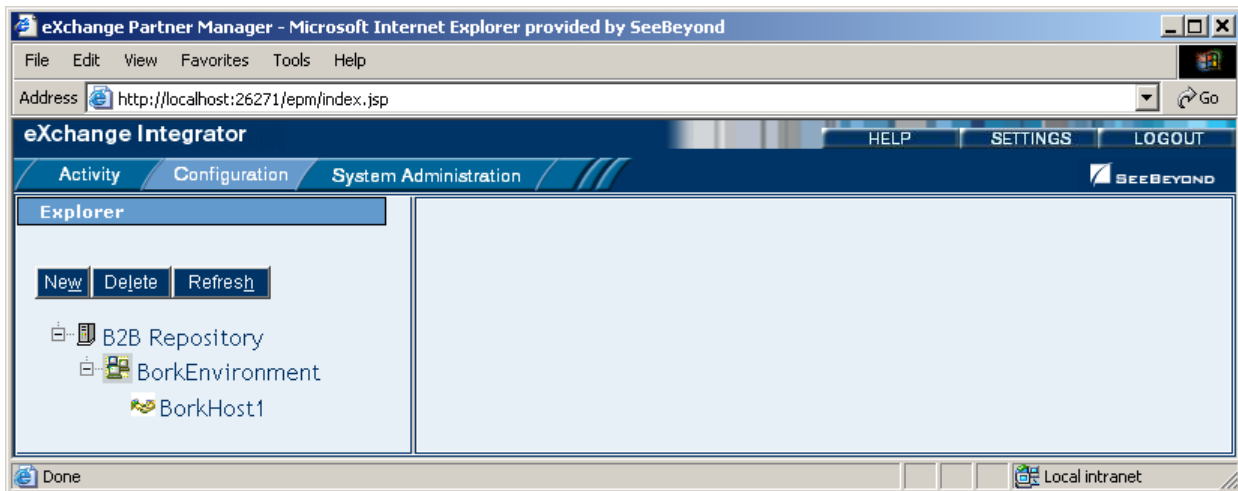
To display currently active environments and their B2B hosts

- 1 If necessary, open a browser window, point it at the hostname and port where your repository is running, and log in to Enterprise Manager.
- 2 To the base URL for Enterprise Manager, append the following string: **/epm** (must be all-lowercase). For example:
 - ♦ <http://BeigeBox:12000/epm>
 - ♦ <http://localhost:22222/epm>

After a brief delay, eXchange Integrator opens to the Configuration tab, displaying a closed tree in the Explorer (left) pane and a blank canvas on the right.

- 3 Fully expand the B2B Repository tree to display environments and their B2B hosts. At a minimum, you should see BorkEnvironment and BorkHost1. See Figure 69.

Figure 69 eXchange Partner Manager Showing BorkEnvironment and BorkHost1



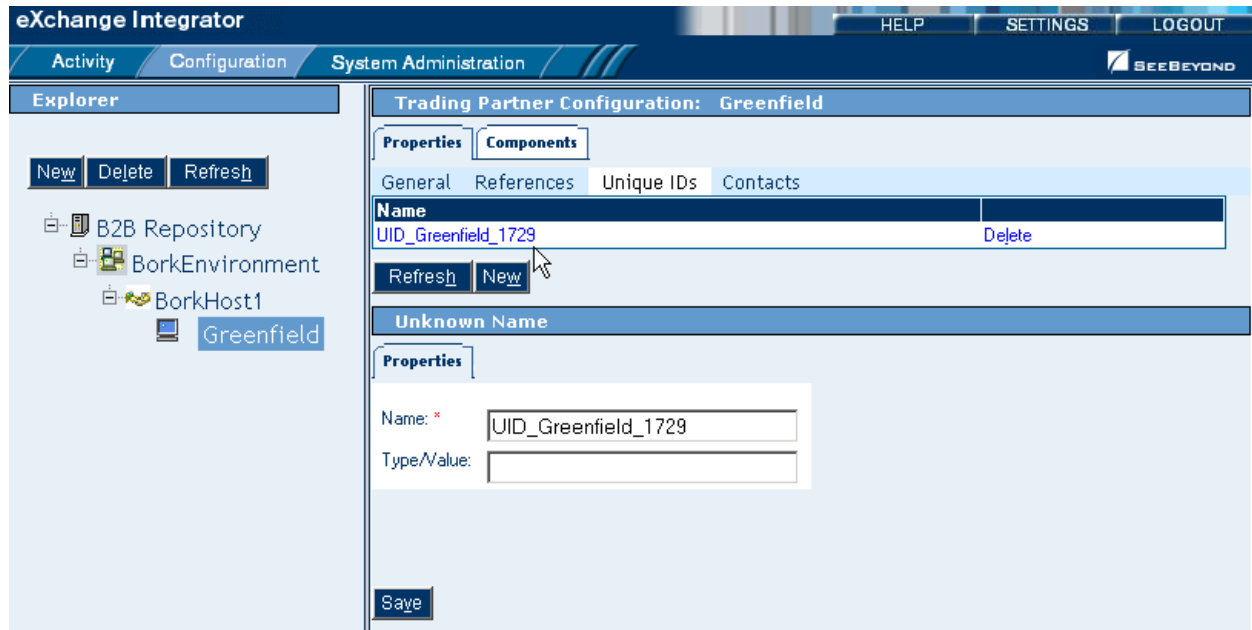
- 4 Click **BorkHost1** to view its properties; for example, the Oracle database parameters match what you specified for the BorkOracle external in the [procedure on page 82](#).

To create new trading partner Greenfield

- 1 In the Explorer tree, click BorkHost1, and then click **New** (top left button).
- 2 For Trading Partner Name, enter **Greenfield** and then click **Save**.
In the Explorer tree, new trading partner Greenfield appears under BorkHost1.
- 3 In the Explorer tree, click Greenfield to display its general properties on the Trading Partner Configuration canvas. Add a description of any kind, and then click **Save**.

- 4 Click **Unique IDs** (the third subtab under Properties), click the **New** button (in the lower center), enter a unique ID of any kind, and then click **Save**. See Figure 70.

Figure 70 New Trading Partner with Unique ID

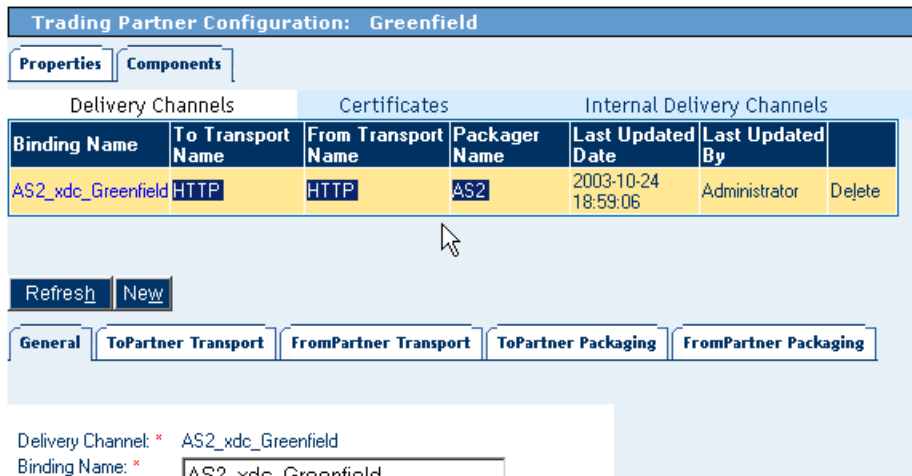


The trading partner is created and named, and needs to be associated with designated values for parameters required by the transport and enveloping protocols specified by the delivery channels. Creating an association between an object and a particular set of values is called a *binding*.

To associate Greenfield with a BorkHost external delivery channel

- 1 For trading partner Greenfield: In the configuration canvas, click **Components**.
You use the Components tab to associate your trading partner with delivery channels and certificates.
- 2 With **Delivery Channels** active (the first subtab), click **New** (lower center button).
- 3 For Delivery Channel, select AS2_xdc_Greenfield. Then click **Continue**.
You have created a new binding (named AS2_xdc_Greenfield unless you change it). The display reflects the fact that you designated the external delivery channel to use HTTP as the transport protocol for both ToPartner and the FromPartner directions, and to use AS2 as the enveloping=packaging protocol. See Figure 71.

Figure 71 General Properties of Binding for AS2_xdc_Greenfield



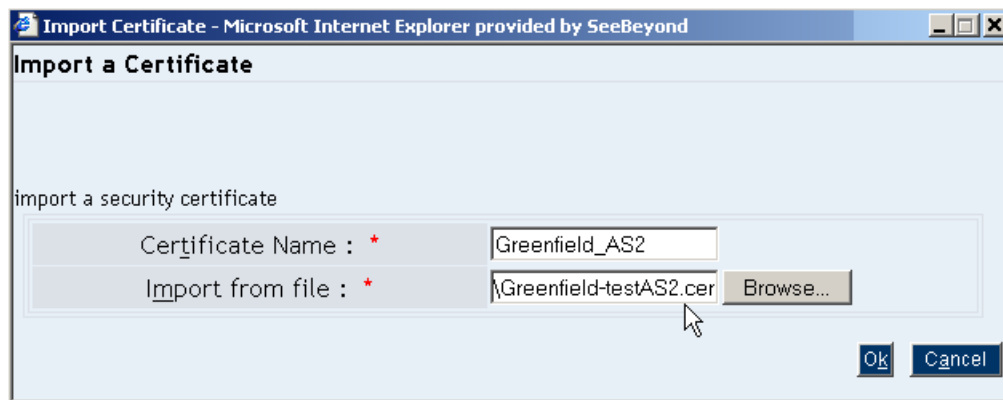
In the next steps, you will use the four subtabs to provide specific parameter values to be used by the transport protocol (HTTP) and enveloping protocols.

- Click the **ToPartner Transport** tab. For All Purpose End Point, enter an appropriate URL and then click **Save**. For example:

```
http://localhost/Inbound/as2.dosend
```

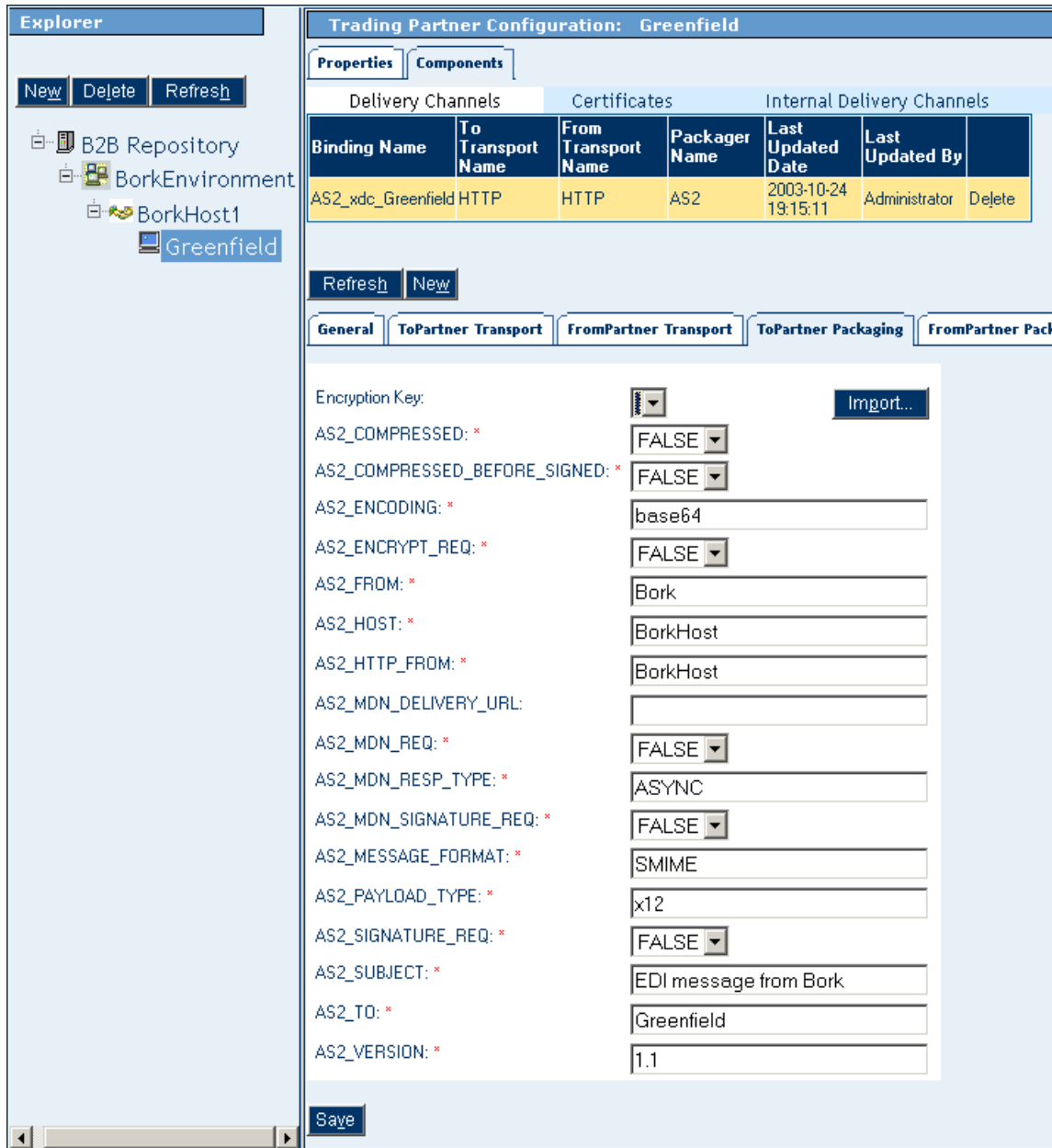
- Repeat the previous step for **FromPartner Transport**. Be sure to click **Save**.
- Click the **ToPartner Packaging** tab and do the following:
 - Click the **Import** button and then, in the **Import Certificate** dialog box, for Certificate Name, enter **greenfield_as2**, and browse to the location of the sample files and select **Greenfield-testAS2.cer**. When done, click **OK**. See Figure 72.

Figure 72 Importing a Certificate for Delivery Channel AS2_xdc_Greenfield



- Supply other values as shown in Figure 73.

Figure 73 Packaging Values for Delivery Channel AS2_xdc_Greenfield



C Click **Save**.

- 7 Click the **FromPartner Packaging** tab and supply the following values:
 - ◆ For Signature Certificate, choose: **greenfield_as2**
 - ◆ For AS2_FROM, enter: **Greenfield**
 - ◆ For AS2_POSITIVE_MDN_DISPOSITION_MESSAGE, keep the default: **positive disposition message**
 - ◆ For AS2_REPORTING_UA, enter: **UA**
 - ◆ For AS2_To, enter: **Bork**

8 Click **Save**.

You have now configured the external delivery channel with specific values for transport and enveloping parameters (including a certificate), and all you have left to do is to configure the internal delivery channels.

To associate Greenfield with BorkHost internal delivery channels

You will supply values for two internal delivery channels: first, the *from*Internal IDC:

- 1** Still within the Components pane, click **Internal Delivery Channels** and then click the **New** button in the lower center of the Trading Partner Configuration pane.
- 2** Choose **Greenfield_idc_fromInternal** for the channel name and binding name, and then click **Continue**.
- 3** Click **Save**. If necessary, make any adjustments and save again.
- 4** Click the **Receiver Transport** tab and supply values for all required parameters:
 - ◆ For ChannelManagerMode, keep: **FILE**
 - ◆ For Directory, enter: **C:\temp**
 - ◆ For FilePattern, enter: **input*.txt**
 - ◆ For PollMilliseconds, enter: **5000**
 - ◆ For HostName, Password, and UserName, enter any values. (These parameters are required for FTP, but are ignored for internal file transfer.)
- 5** Click **Save**.

Next, you create a binding and configure values for the *to*Internal IDC:

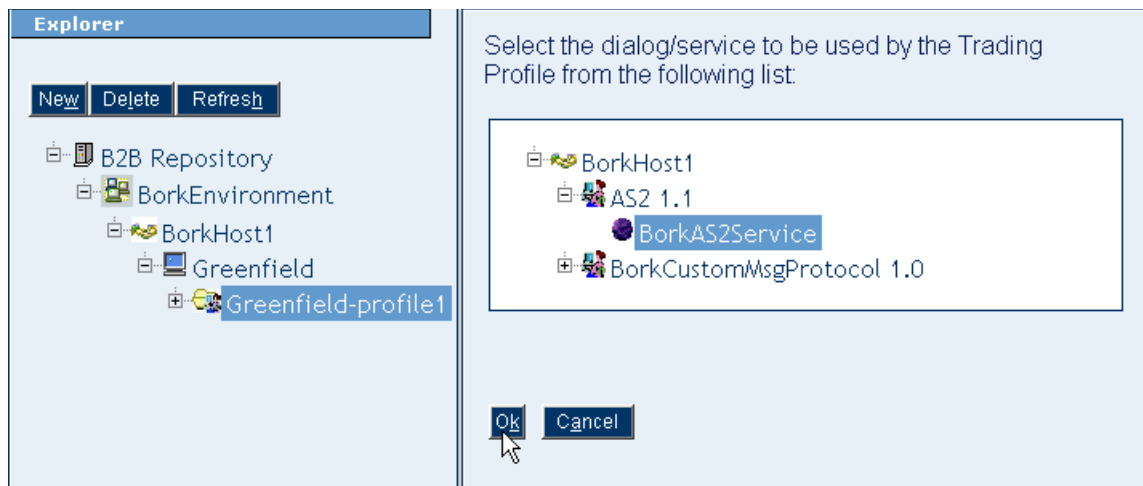
- 6** Click the **New** button (lower center of the Trading Partner Configuration pane).
- 7** Choose **Greenfield_idc_toInternal** for the channel name and binding name, and then click **Continue**.
- 8** Click **Save**. If necessary, make any adjustments and save again.
- 9** Click the **Sender Transport** tab and supply values for all required parameters:
 - ◆ For ChannelManagerMode, keep: **FILE**
 - ◆ For Directory, enter: **C:\temp**
 - ◆ For FilePattern, enter: **output*.txt**
 - ◆ For PollMilliseconds, enter: **5000**
 - ◆ For HostName, Password, and UserName, enter any values. (These parameters are required for FTP, but are ignored for internal file transfer.)
- 10** Click **Save**.

This completes all the configuration setup required by the Greenfield trading partner itself. The only step remaining is to configure a trading partner profile for its business protocols (also known as services).

To create a Greenfield trading partner profile and associate it with specific service actions

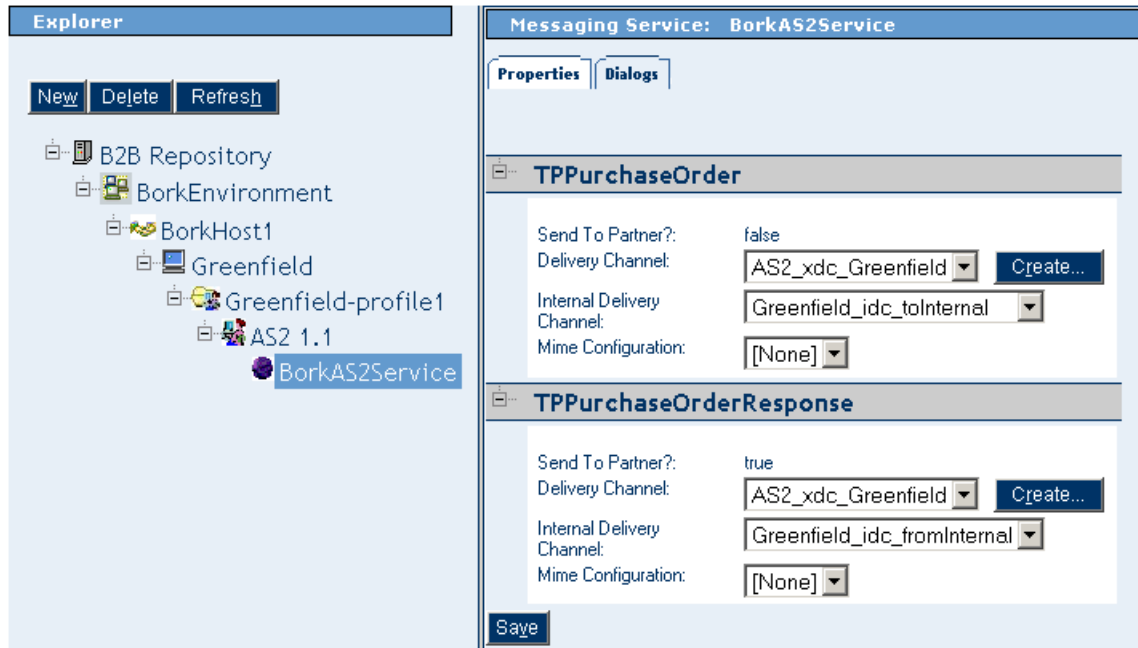
- 1 In the explorer tree, click the **Greenfield** trading partner. Then click the **New** button (at the top left of the Explorer pane). You are creating a new *trading partner profile*.
- 2 For Profile Name, enter: **Greenfield-profile1**. Then click **Save**.
The new profile appears in the tree.
- 3 In the explorer tree, click the newly created **Greenfield-profile1** profile. Then click the **New** button at the top left of the pane. You are creating a new *service binding*.
- 4 In the right pane, expand BorkHost1 > AS2 1.1 to display all available business protocols available to BorkHost that use version 1.1 of the AS2 enveloping protocol. (In this example, there is only one. See Figure 74.)

Figure 74 Associating a Trading Partner Profile With a Specific Business Protocol



- 5 Click **BorkAS2Service** and then click **OK**.
The new service appears in the tree.
- 6 In the right pane, click the **Dialogs** tab, open both service actions, and specify the following values for their parameters as shown in Figure 75:
 - ♦ For the Delivery Channels: Choose **AS2_xdc_Greenfield** for both TPPurchaseOrder and TPPurchaseOrderResponse.
 - ♦ For the Internal Delivery Channels: For the PO, choose *toInternal*; for the response, choose *fromInternal*.
 - ♦ For Mime Configuration: Keep the default, [None].

Figure 75 Parameters for Service Actions TPPurchaseOrder, TPPurchaseOrderResponse



7 Click **Save**.

Now that you have fully configured this trading partner, you are ready to activate it.

8 In the explorer tree, click **Greenfield**.

The right pane displays the General properties of the Greenfield trading partner, with buttons for several actions (Save, Import CPA, Export, Activate, and Copy).

9 At the bottom of the right pane, click the **Activate** button.

10 In the Trading Partner Activation window, click **Activate**.

The right pane tells you that the activation was successful. See Figure 76.

Figure 76 Confirmation Message Upon Successful TP Activation



To create new trading partner Johnson

The following steps for Johnson are greatly condensed, because they largely duplicate what you already did for Greenfield beginning with the [procedure on page 99](#). If you have any problems following them, refer to the corresponding steps for Greenfield.

- 1 In the Explorer tree, click **BorkHost1**, and then click **New** (upper left of left pane). For Trading Partner Name, enter **Johnson** and then click **Save**.

In the Explorer tree, new trading partner Johnson appears just below Greenfield.

- 2 In the Explorer tree, click **Johnson**, add a description of any kind, and then click **Save**. Then, click **Unique IDs** (the third subtab under Properties), click the **New** button (in the lower center), enter a unique ID of any kind, and then click **Save**.

The trading partner is created and named, and needs to be associated with designated values for parameters required by the transport and enveloping protocols specified by the delivery channels.

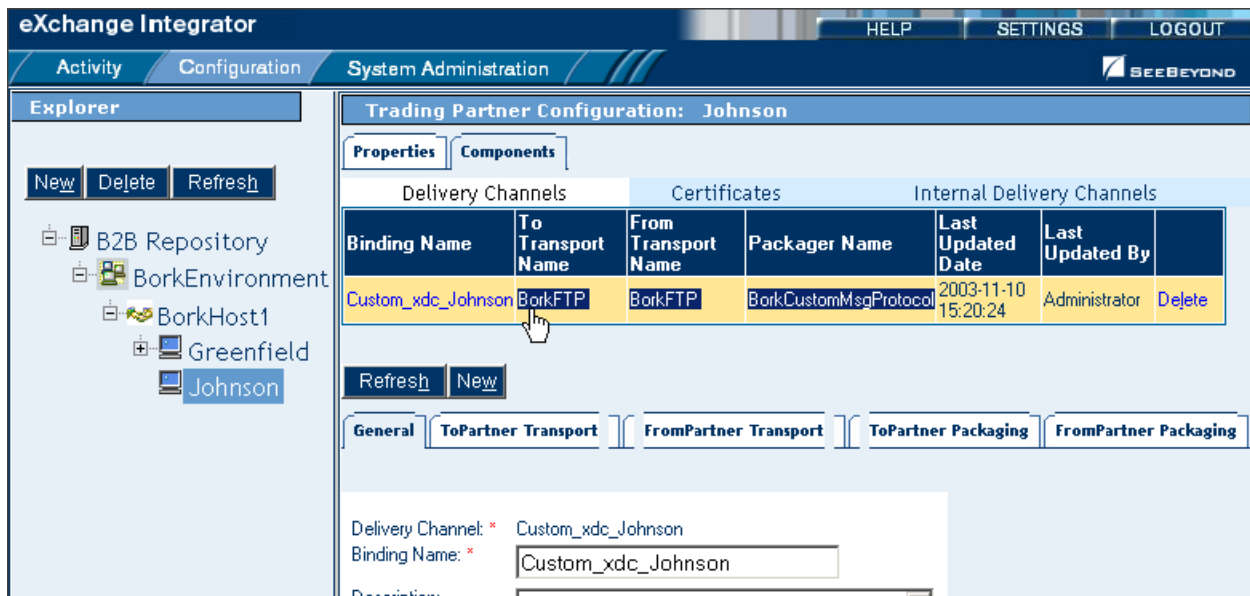
To associate Johnson with a BorkHost external delivery channel

The following steps largely duplicate what you already did for Greenfield beginning with the [procedure on page 100](#).

- 1 For trading partner Johnson: In the configuration canvas, click **Components**.
- 2 With **Delivery Channels** active (the first subtab), click **New** (lower center button).
- 3 For Delivery Channel, select **Custom_xdc_Johnson**. Then click **Continue**.

You have created a new binding, **Custom_xdc_Johnson**. The display reflects the fact that you designated the external delivery channel to use **BorkFTP** (a custom transport protocol) for both ToPartner and the FromPartner directions, and to use **BorkCustomMsgProtocol** as the enveloping=packaging protocol. See Figure 77.

Figure 77 General Properties of Binding for Custom_xdc_Johnson



In the next steps, you will use the four subtabs to provide specific parameter values to be used by the transport protocol (BorkFTP) and enveloping protocols.

- 4 Click the **ToPartner Transport** tab and provide values that will perform an FTP **put** of the file **PO_%d%H%m%s%S.dat** located in a particular directory on a particular machine. Be sure to set ChannelManager Mode to **FTP**. See Figure 78.

Note: Although the **All Purpose End Point** parameter is not used, it requires a nonblank entry, such as `ftp://myServer.Bork.com/toJohnson/PO_%d%H%m%s%S.dat`

Figure 78 Sample ToPartnerTransport Parameters for BorkFTP

The screenshot shows the 'Trading Partner Configuration: Johnson' window. The 'Components' tab is active, showing a table of delivery channels. Below the table, the 'ToPartner Transport' sub-tab is selected, displaying a form with the following parameters:

Binding Name	To Transport Name	From Transport Name	Packager Name	Last Updated Date	Last Updated By	
Custom_xdc_Johnson	BorkFTP	BorkFTP	BorkCustomMsgProtocol	2003-11-10 15:20:24	Administrator	Delete

Buttons: Refresh, New

Sub-tabs: General, **ToPartner Transport**, FromPartner Transport, ToPartner Packaging, FromPartner Packaging

Parameters in the 'ToPartner Transport' sub-tab:

- All Purpose End Point: * ftp://localhost/toJohnson/PO_%d%
- Channel Manager mode: * FTP
- Host name: * localhost
- Is Socks Enabled?: no
- Password: * (myPassword)
- Poll interval (in milliseconds): * 5000
- Port number: * 21
- Socks host name:
- Socks password:
- Socks server port:
- Socks user name:
- Target directory: * C:\Partners\Johnson
- Target file pattern: * PO_%d%H%m%s%S.dat
- User name: * mySelf

Save button

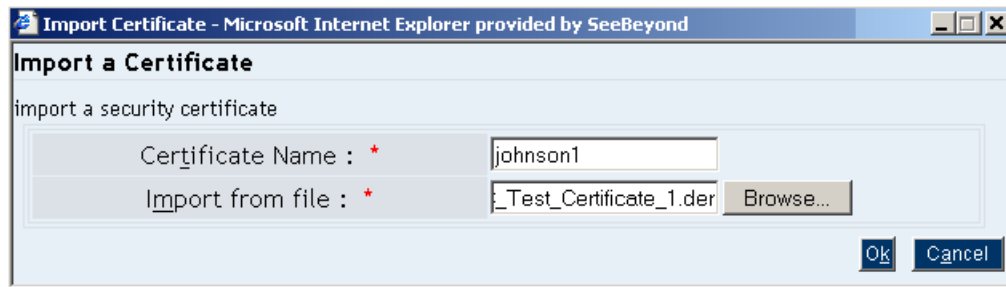
Of necessity, many of these parameters, such as hostname and FTP user/password, depend on conditions at your site.

- 5 When you have supplied all required parameters, click **Save**.
- 6 Click the **FromPartner Transport** and repeat the previous two steps, providing values for an FTP **get** operation of files matching the pattern **POResponse*.dat**

located in a particular directory on a particular machine. Be sure to set ChannelManager Mode to **FTP** and click **Save**.

- 7 Click the **ToPartner Packaging** tab and do the following:
 - A Click the **Import** button and then, in the **Import Certificate** dialog box, for Certificate Name, enter **johnson1**, and browse to the location of the sample files and select **STC_Test_Certificate_1.der**. When done, click **OK**. See Figure 79.

Figure 79 Importing a Certificate for Delivery Channel Custom_xdc_Johnson



- B Click **Save**.

- 8 Click the **FromPartner Packaging** tab, choose **johnson1**, and click **Save**.

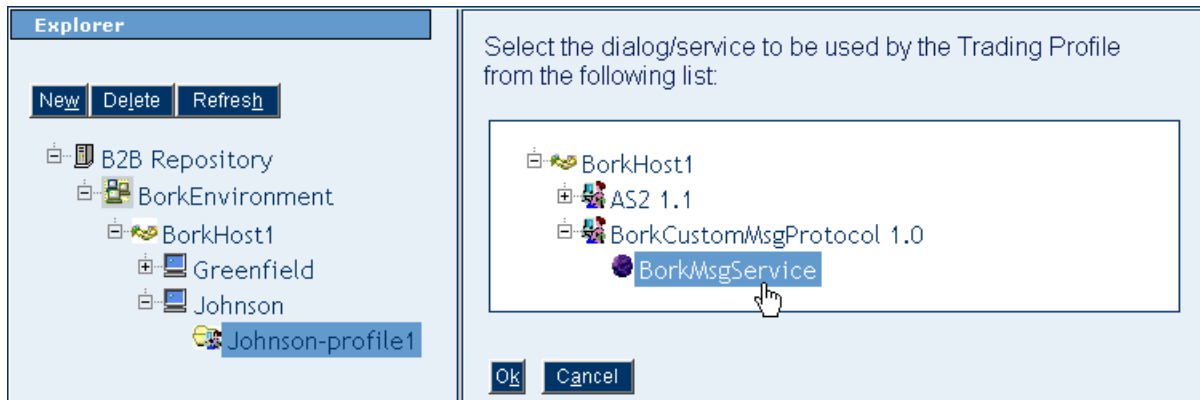
You have now configured the external delivery channel with specific values for transport and enveloping parameters, including a certificate. The only step remaining is to configure a trading partner profile for its business protocols (also known as *services*).

To create a Johnson trading partner profile and associate it with specific service actions

The following steps largely duplicate what you already did for Greenfield beginning with the [procedure on page 104](#).

- 1 In the tree, click **Johnson**, and then click **New** (top left of the Explorer pane). For Profile Name, enter: **Johnson-profile1**. Then click **Save**.
- 2 In the tree, click the newly created **Johnson-profile1** profile. Then click the **New** button at the top left of the pane. You are creating a new *service binding*.
- 3 In the right pane, expand BorkHost1 > BorkCustomMsgProtocol 1.1 to display all business protocols available to BorkHost that use the BorkCustomMsgProtocol enveloping protocol. (In this example, there is only one. See Figure 80.)

Figure 80 Associating a Trading Partner Profile With a Specific Business Protocol

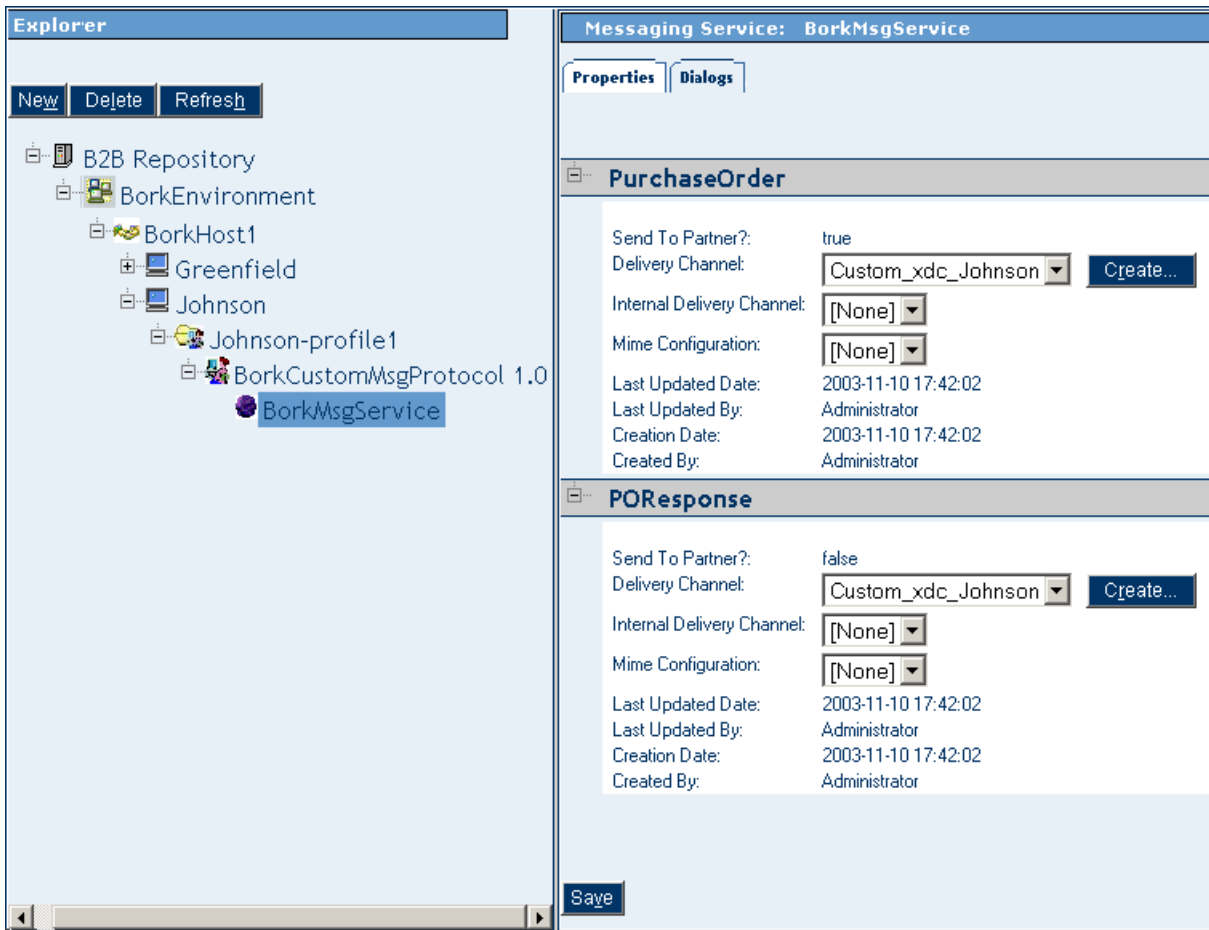


- 4 Click **BorkMsgService** and then click **OK**.

The new service appears in the tree.

- 5 In the right pane, click the **Dialogs** tab, open both service actions (PurchaseOrder and POResponse), and specify the following values for their parameters as shown in Figure 81:
 - ♦ For Delivery Channel: Choose **Custom_xdc_Johnson** for both PurchaseOrder and POResponse.
 - ♦ For Delivery Channel: Keep the default, [None], for both.
 - ♦ For Mime Configuration: Keep the default, [None], for both.

Figure 81 Parameters for Service Actions PurchaseOrder and POResponse



- 6 Click **Save**. Confirmation: **Actions configuration has been successfully saved.**
- Now that you have fully configured this trading partner, you are ready to activate it.
- 7 In the explorer tree, click **Johnson**.
 - 8 At the bottom of the right pane, click the **Activate** button.
 - 9 In the Trading Partner Activation window, click **Activate**.
- Confirmation: **Successful CPA Activation** Trading Profile(s) successfully activated.

8.2 AS2 Design-Time Scenario for Greenfield

In this section

- “Setting Up the Components” on this page.
- “Using the Prebuilt AS2 Pipelines” on page 119.
- “Designing the Outbound Message Flow” on page 124.

- [“Deploying Components to Servers in the Environment” on page 133.](#)

8.2.1. Setting Up the Components

To set up the AS2Greenfield project

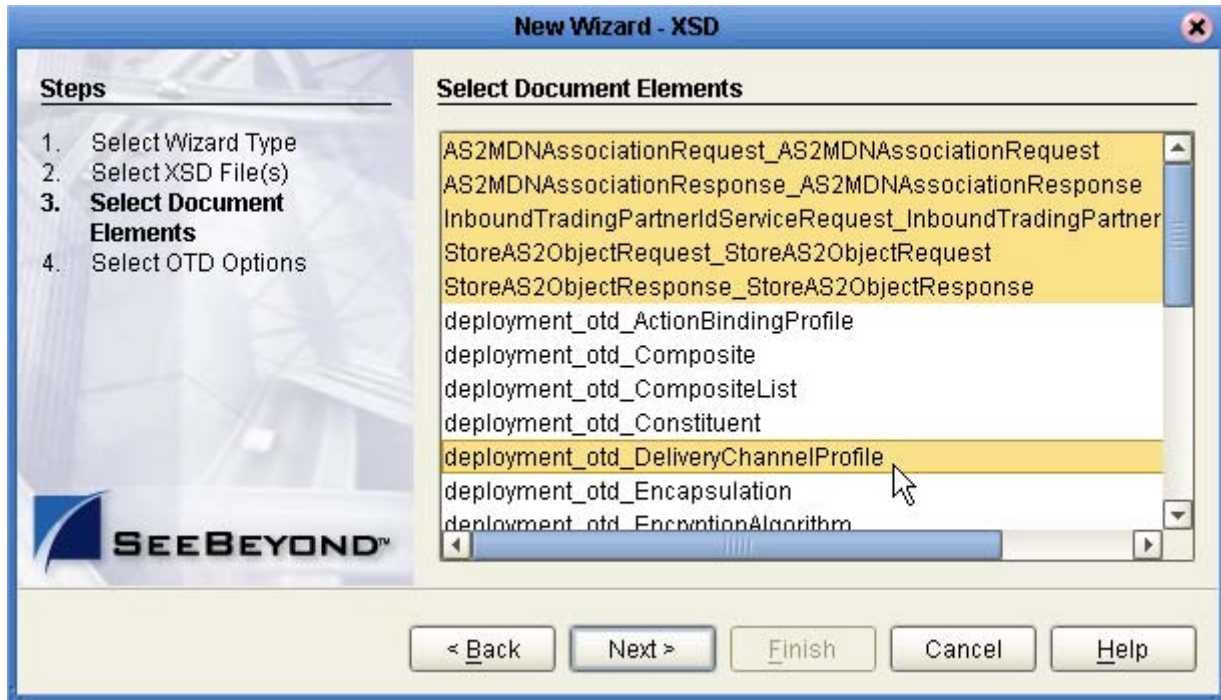
- 1 In Enterprise Designer, click the **Project Explorer** tab (at the bottom left margin).
- 2 In the project tree, right-click the repository and, on the popup menu, click **New Project**. Name the project: **AS2Greenfield**.

To set up the XSD-based OTDs for AS2Greenfield

- 1 In the project tree, right-click AS2Greenfield > **New > Object Type Definition**.
- 2 When asked to specify an OTD Wizard Type, click **XSD**, and then click **Next**.
- 3 When asked to specify XSD Files, navigate to the location where you extracted the sample files, and then, under **XSD_files_for_building_OTDs**, select all six of the following:
 - ♦ AS2MDNAssociationRequest.xsd
 - ♦ AS2MDNAssociationResponse.xsd
 - ♦ deployment_otd.xsd
 - ♦ InboundTradingPartnerIdServiceRequest.xsd
 - ♦ StoreAS2ObjectRequest.xsd
 - ♦ StoreAS2ObjectResponse.xsd
- 4 When asked to select document elements, specify only the following as top nodes:
 - ♦ AS2MDNAssociationRequest_AS2MDNAssociationRequest
 - ♦ AS2MDNAssociationResponse_AS2MDNAssociationResponse
 - ♦ InboundTradingPartnerIdServiceRequest_InboundTradingPartner[...]
 - ♦ StoreAS2ObjectRequest_StoreAS2ObjectRequest
 - ♦ StoreAS2ObjectResponse_StoreAS2ObjectResponse
 - ♦ deployment_otd_**DeliveryChannelProfile**

See Figure 82.

Figure 82 Document Elements (Top Nodes) for the Six XSD-Based OTDs



- 5 When asked to select OTD options, keep the default settings (all unchecked) and click **Finish**.

The project tree displays six new OTDs in the AS2Greenfield project. On the canvas, the OTD editor displays all six OTDs.

In addition to the six XSD-based OTDs, which are used to unmarshal (parse) and marshal (flatten) messages coming from and going to AS2 pipelines, the Greenfield project requires an Oracle database OTD for trading partner lookup.

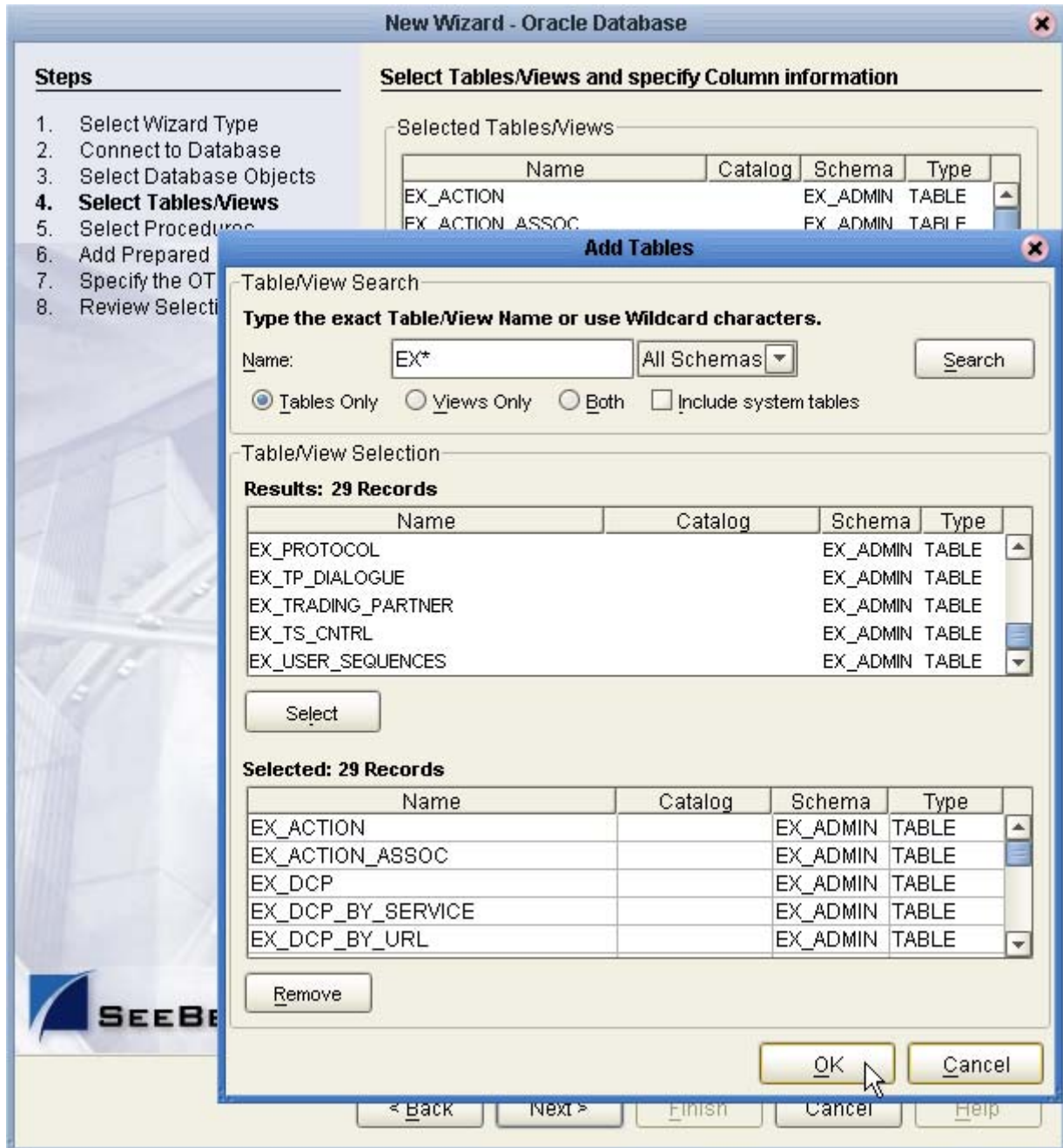
To set up the Oracle database OTD for Greenfield

Before you begin: You must have a connection to a running eXchange Oracle database instance—if necessary, see [“Creating and Configuring the eXchange Database Instance” on page 29](#)—and you must know its connection parameters (hostname, port ID, SID, user name, and password)

- 1 In the project tree, right-click AS2Greenfield > **New > Object Type Definition**.
- 2 When asked to specify an OTD Wizard Type, click **Oracle**, and then click **Next**.
- 3 When asked to specify database connection information, supply the correct values for the Oracle database instance you are using. The database parameters must match what you specified for the BorkOracle external in the [procedure on page 82](#).
If you cannot progress, double-check that your Oracle database instance is running.
- 4 When asked to specify database objects, select the **Tables/Views** checkbox (only).
- 5 When asked to select Tables/Views, click **Add**. Then, in the **Add Tables** dialog box:

- A For Name, enter **EX***; then with the **Tables Only** option selected, click **Search**.
- B In the middle pane (Table/View Selection), highlight all 29 records and click **Select**. See Figure 83.

Figure 83 The 29 EX* Table Records in the Oracle Database Instance for eXchange



- C Click **OK** to return to the OTD Wizard.
- 6 In the OTD Wizard, click **Next** and, for the OTD name, enter: **eXchangeDB**
- 7 Click **Next** and then click **Finish**. When the new eXchangeDB OTD appears, save your work and close the canvas.

The seven OTDs you just created will be used in the Java Collaboration Definition (JCDs) you will create, as shown in Table 10.

Table 10 How the Greenfield OTDs Are Used by Its Java Collaboration Definitions

This OTD is used by this Java Collaboration Definition:
InboundTradingPartnerIdServiceRequest_[...]	Provides input to InboundTradingPartnerIdService
deployment_otd_DeliveryChannelProfile	Receives output of InboundTradingPartnerIdService
StoreAS2ObjectRequest_[...]	Provides input to StoreAS2Object
StoreAS2ObjectResponse_[...]	Receives output of StoreAS2Object
AS2MDNAssociationRequest_[...]	Provides input to AS2MDNAssociation
AS2MDNAssociationResponse_[...]	Receives output of AS2MDNAssociation
eXchangeDB	Referenced by InboundTradingPartnerIdService Referenced by StoreAS2Object Referenced by AS2MDNAssociation

For each of the three Java Collaboration Definitions, you will follow these procedures:

- Use the JCD Wizard to create and name the JCD, stipulate that it constitutes a new web service operation named **execute**, and then specify the OTDs it requires.
- Import the **.jar** files needed by the JCD, taking care to add them in the correct order. All **.jar** files needed by AS2Greenfield are supplied with the eXchange product, under **SeeBeyond > eXchange > MessagingProtocols > AS2Pipelines > Collabs: stc.com.eXchange*.jar**
- Import the Java code that will be used in the scenario, and then validate the JCD (to ensure that you find and correct any errors before proceeding further). The three **.java** files for the three JCDs are supplied along with the other sample files, in the **Java_files_for_Collab_Defs** folder.

To define JCD “InboundTradingPartnerIdService” (web service operation: *execute*)

- 1 In the project tree, right-click AS2Greenfield > **New** > **Java Collaboration Definition**. Name it **InboundTradingPartnerIdService**. For Web Service Type, be sure you select the **New** option button before clicking Next. See Figure 84.

Figure 84 Creating a Java Collaboration Definition with New Web Service Operation



- 2 When asked to enter an operation name, type in the following: **execute**
- 3 When asked to select an input message (of type web service message), open the AS2Greenfield project and select: **InboundTradingPartnerIdServiceRequest_ [...]**
- 4 When asked to select an output message (of type web service message), open the AS2Greenfield project and select: **deployment_otd_DeliveryChannelProfile**
- 5 When asked to select OTDs, open the AS2Greenfield project, select **eXchangeDB**, click **Add** (only once, creating an instance named **eXchangeDB_1**), and click **Finish**.

The AS2Greenfield tree displays the new JCD (**InboundTradingPartnerIdService**), and the canvas opens the Java Collaboration editor. See Figure 85.

Before the JCD can be used, however, it requires four **.jar** files.

To import .jar files from AS2_Pipelines for JCD “InboundTradingPartnerIdService”


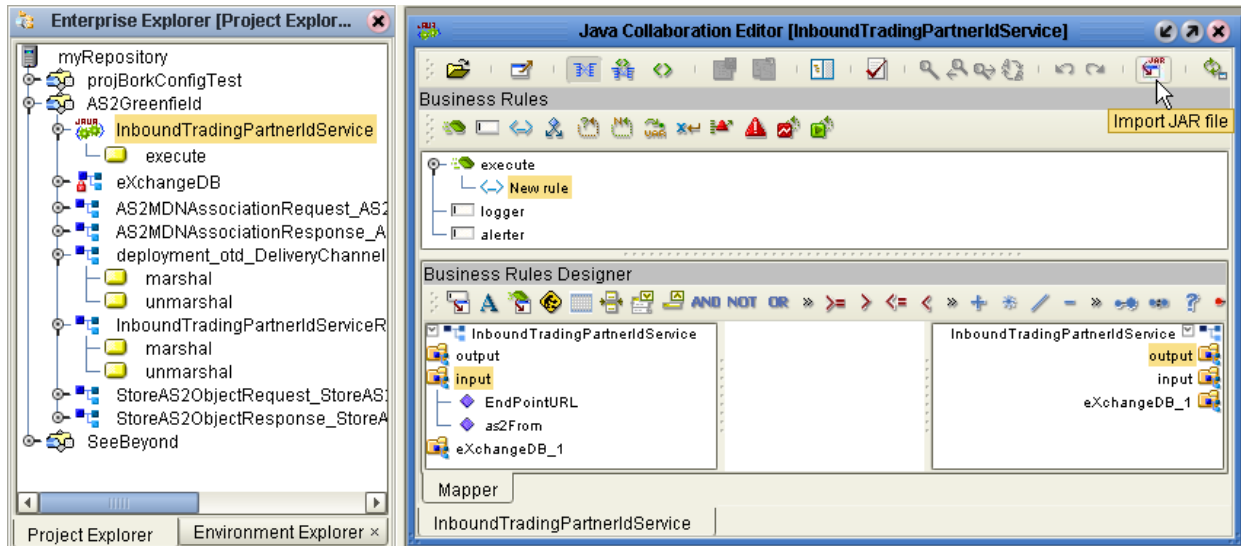
- 1 In the Java Collaboration Editor, on the right of the tool palette, click the  **Import JAR file** tool. See Figure 85.

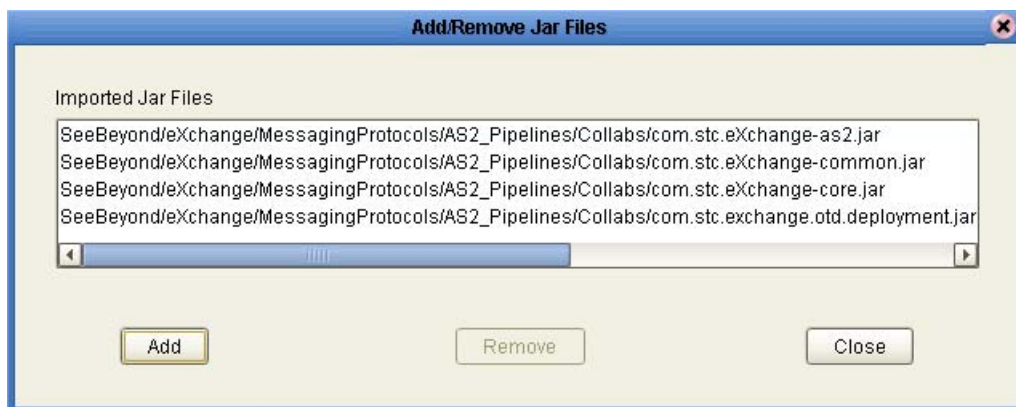
Figure 85 Importing .jar Files Into a Java Collaboration Definition



- 2 In the Add/Remove Jar Files dialog box, click **Add**.
- 3 Open SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines > Collabs and, one by one, import the following .jar files *in the order listed below*:
 - A com.stc.eXchange-common.jar
 - B com.stc.eXchange-core.jar
 - C com.stc.eXchange-as2.jar
 - D com.stc.exchange.otd.deployment.jar



See Figure 86.

Figure 86 Specifying Names and Order of .jar Files to Be Imported



- 4 When these four files appear in the above order, click **Close**.

To import the .java code for JCD “InboundTradingPartnerIdService”


- 1 On the upper left of the JCE tool palette, click the  **Import a local file** tool
- 2 Navigate to the location where you extracted the sample files, and then, under **Java_files_for_Collab_Defs**, select: **InboundTradingPartnerIdService.java**
- 3 When the editor redisplay the Java Collaboration Definition, save your changes, and then click the  **Validate** tool to verify that the code has no syntax errors.

You will need to repeat similar procedures for the other two JCDs (“StoreAS2Object” and “AS2MDNAssociation”).



To define JCD “StoreAS2Object” (web service operation: *execute*)

- 1 Right-click AS2Greenfield > **New** > **Java Collaboration Definition** and name it **StoreAS2Object**. For Web Service Type, be sure you select the **New** option button before clicking Next.
- 2 For operation name, enter: **execute**
- 3 For input, open AS2Greenfield and select: **StoreAS2ObjectRequest_...**
- 4 For output, open AS2Greenfield and select: **StoreAS2ObjectResponse_...**
- 5 For select OTDs, open the AS2Greenfield project, select **eXchangeDB**, click **Add** (only once, creating an instance named eXchangeDB_1), and click **Finish**.

To import .jar files from AS2_Pipelines for JCD “StoreAS2Object”

- 1 In the JCE, on the right of the tool palette, click the  **Import JAR file** tool.
- 2 In the Add/Remove Jar Files dialog box, click **Add**.
- 3 Open SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines > Collabs and, one by one, import the following .jar files *in the order listed below*:
 - A com.stc.eXchange-common.jar
 - B com.stc.eXchange-core.jar
 - C com.stc.eXchange-as2.jar
- 4 When these three files appear in the above order, click **Close**.


To import the .java code for JCD “StoreAS2Object”

- 1 On the upper left of the JCE tool palette, click the  **Import a local file** tool
- 2 Navigate to the location where you extracted the sample files, and then, under **Java_files_for_Collab_Defs**, select: **StoreAS2Object.java**
- 3 When the editor redisplay the Java Collaboration Definition, save your changes, and then click the  **Validate** tool to verify that the code has no syntax errors.


To define JCD “AS2MDNAssociation” (web service operation: *execute*)

- 1 Right-click AS2Greenfield > **New** > **Java Collaboration Definition** and name it **AS2MDNAssociation**. For Web Service Type, be sure you select the **New** option button before clicking Next.
- 2 For operation name, enter: **execute**
- 3 For input, open AS2Greenfield and select: **AS2MDNAssociationRequest_ [...]**
- 4 For output, open AS2Greenfield and select: **AS2MDNAssociationResponse_ [...]**
- 5 For select OTDs, open the AS2Greenfield project, select **eXchangeDB**, click **Add** (only once, creating an instance named eXchangeDB_1), and click **Finish**.

To import .jar files from AS2_Pipelines for JCD “AS2MDNAssociation”

- 1 In the JCE, on the right of the tool palette, click the  **Import JAR file** tool.
- 2 In the Add/Remove Jar Files dialog box, click **Add**.
- 3 Open SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines > Collabs and, one by one, import the following .jar files *in the order listed below*:
 - A com.stc.eXchange-common.jar
 - B com.stc.eXchange-core.jar
 - C com.stc.eXchange-as2.jar
- 4 When these three files appear in the above order, click **Close**.

To import the .java code for JCD “StoreAS2Object”

- 1 On the upper left of the JCE tool palette, click the  **Import a local file** tool
- 2 Navigate to the location where you extracted the sample files, and then, under **Java_files_for_Collab_Defs**, select: **AS2MDNAssociation.java**
- 3 When the editor redisplay the Java Collaboration Definition, save your changes, and then click the **Validate** tool to verify that the code has no syntax errors.
- 4 Close all canvases.

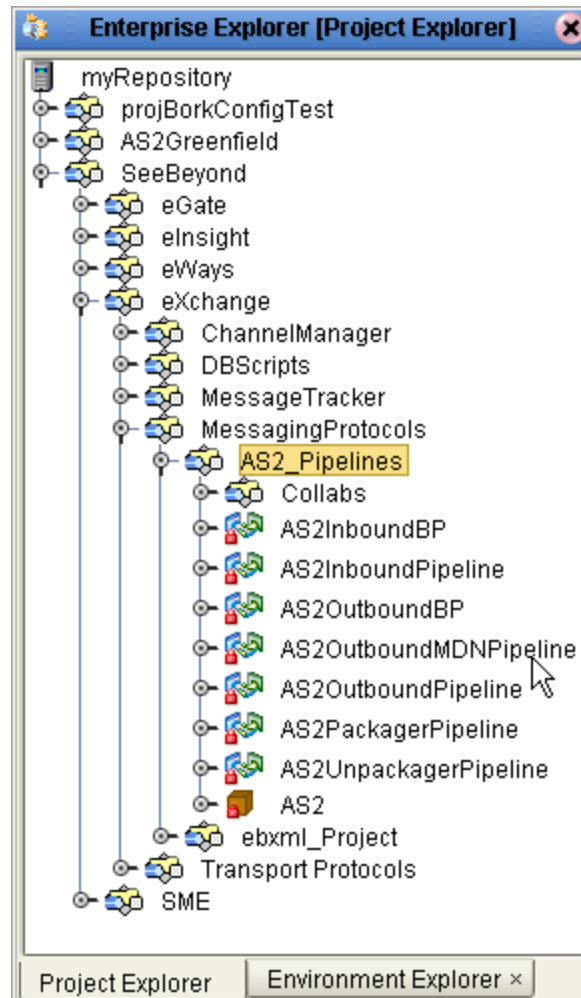
Now that you have created, populated, and validated the three JCDs that will be used by the AS2Greenfield project, you can use their web service operations to modify the prebuilt AS2 protocol pipelines.

8.2.2. Using the Prebuilt AS2 Pipelines

To modify the prebuilt AS2OutboundMDNPipeline

- 1 In the project tree, under the **SeeBeyond** project, successively open eXchange > MessagingProtocols > **AS2_Pipelines**. See Figure 87.

Figure 87 Locating the Prebuilt AS2 Pipelines

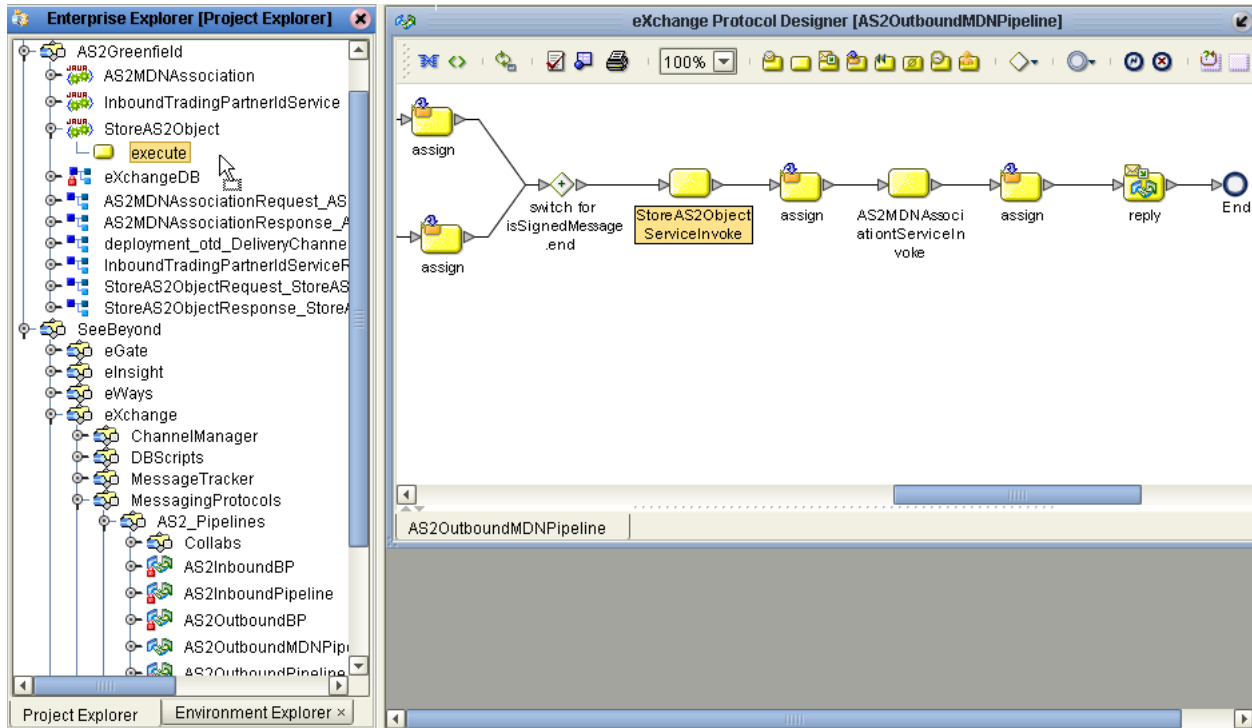


- 2 Open the **Collabs** folder to display all of the Java Collaboration Definitions (JCDs) and other objects associated with the AS2 Pipelines. Then, one by one, right-click each of the fourteen JCDs and, on the popup context menu, click **Check Out**.

After all JCDs (AS2HTTPHeader through VerifyMessageSignature) are checked out, you are ready to check out and use the pipelines themselves.

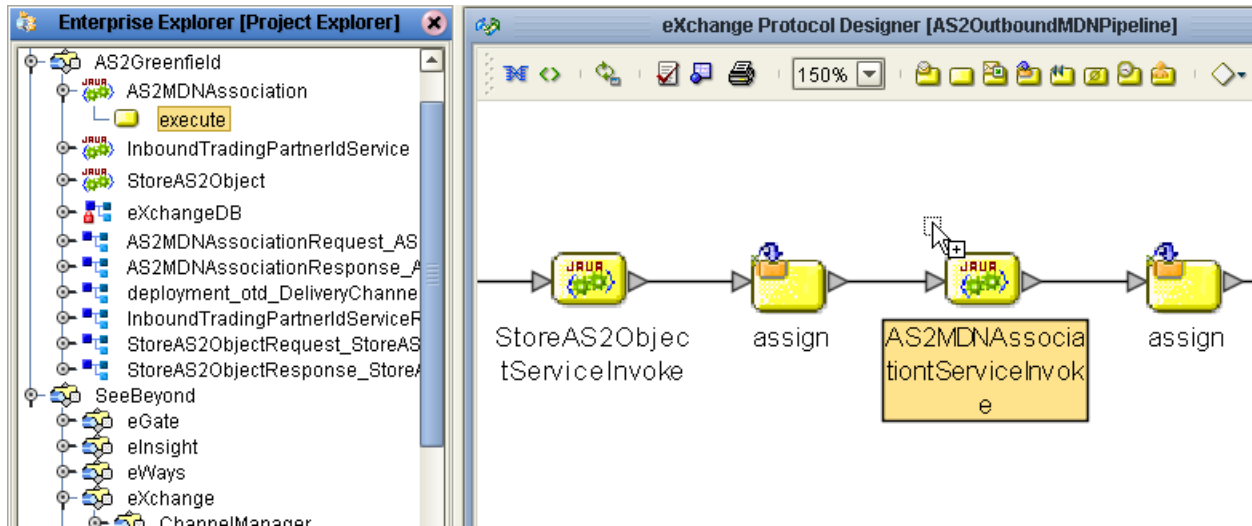
- 3 Under AS2_Pipelines, right-click **AS2OutboundMDNPipeline** and, on the popup context menu, click **Check Out**.
- 4 Double-click **AS2OutboundMDNPipeline** to open the business protocol pipeline in the Protocol Designer. Towards the right of the pipeline, five activities before End, locate the **StoreAS2ObjectServiceInvoke** activity. See Figure 88.

Figure 88 Modifying the Operation of the StoreAS2ObjectServiceInvoke Activity



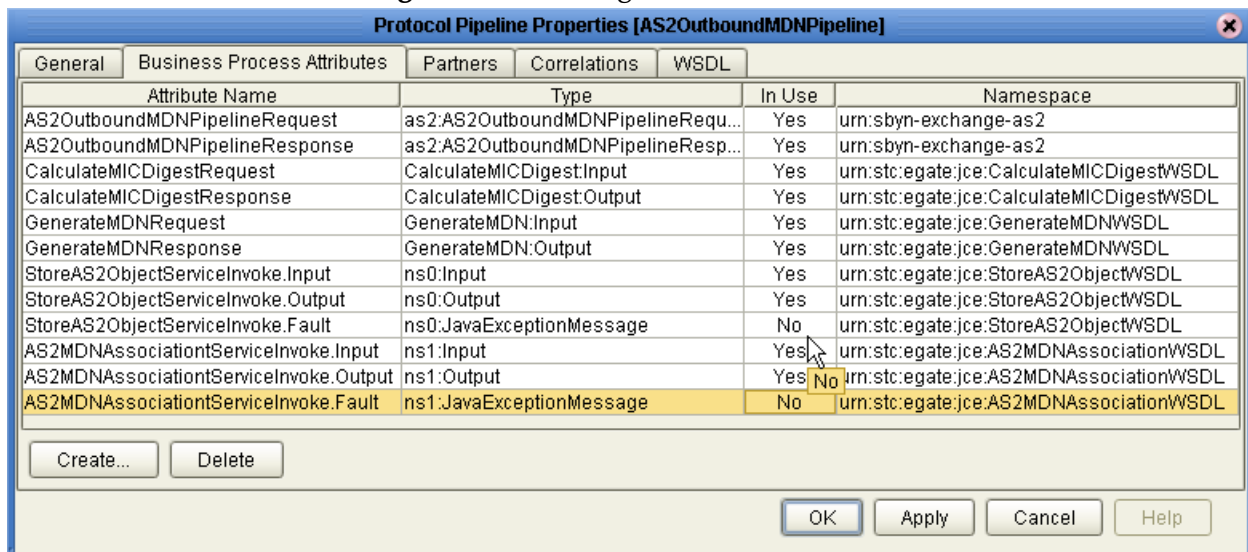
- 5 In the project tree, under the **AS2Greenfield** project, open **StoreAS2Object** and drag its **execute** operation onto the **StoreAS2ObjectServiceInvoke** activity.
The icon for the activity changes from plain yellow to a green gear labeled **JAVA**, indicating the drag-and-drop operation was successful.

Figure 89 Successful Drag-and-Drop



- 6 In the project tree, open **AS2MDNAssociation** and drag its **execute** operation onto the **AS2MDNAssociationServiceInvoke** activity. See Figure 89.
- 7 Save your changes and, on the tool palette, click the **Validate** tool.
A warning tells you that two attributes (namely, the **...Fault** attributes), are defined but not used. This is not an error, and you need not take any action.
- 8 To optionally delete these unused attributes: In the project tree, right-click the pipeline and, on the popup context menu, click **Open Property Sheet** to display the Protocol Pipeline Properties. In the **Business Process Attributes** tab, delete any line whose **In Use** value is **No**. See Figure 90. When done, click OK and re-validate.

Figure 90 Deleting Unused Attributes



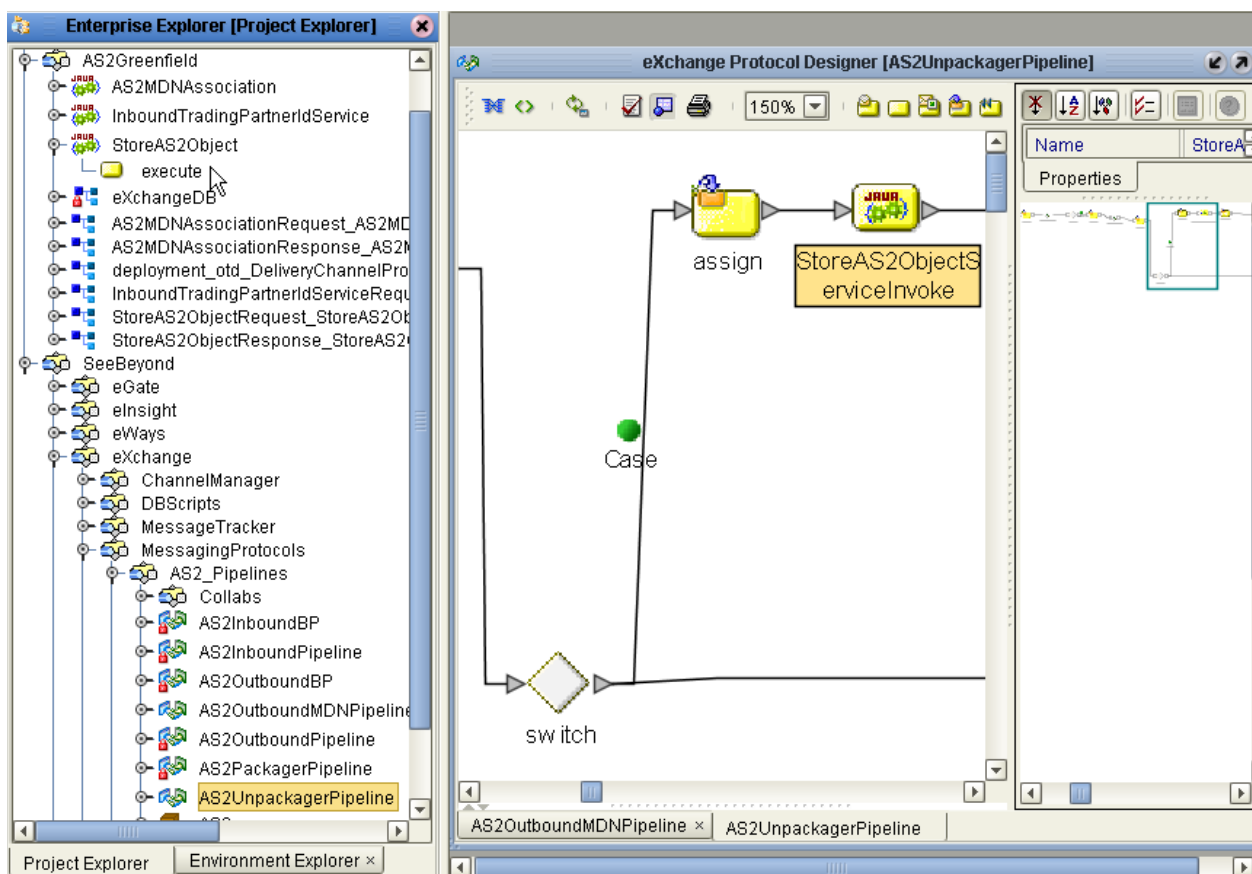
You will need to follow corresponding procedures for the other protocol pipelines: AS2UnpackagerPipeline, AS2PackagerPipeline, and AS2InboundBP.

To modify the prebuilt AS2UnpackagerPipeline

- 1 In the project tree, under SeeBeyond > eXchange > MessagingProtocols > **AS2_Pipelines**, right-click **AS2UnpackagerPipeline** and check it out.
- 2 Open **AS2UnpackagerPipeline** in Protocol Designer. Scroll a third of the way through, to the right of the dip, to find the **StoreAS2ObjectServiceInvoke** activity. (The Zoom Out or Property Sheet tools can help navigate through long pipelines).
- 3 In the project tree, under the **AS2Greenfield** project, open StoreAS2Object and drag its **execute** operation onto the **StoreAS2ObjectServiceInvoke** activity.

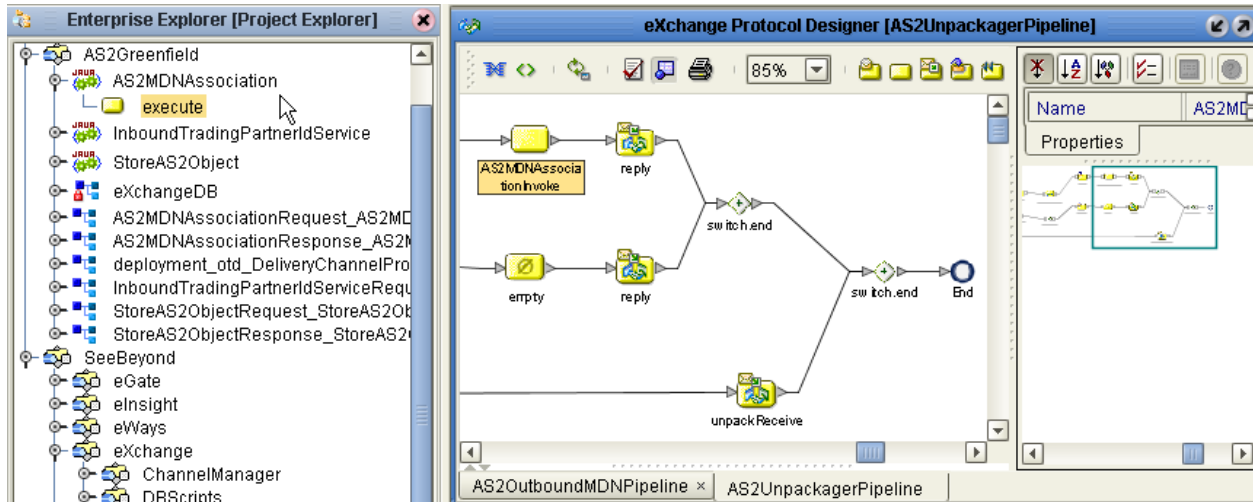
The icon for the activity changes from plain yellow to a green gear labeled JAVA, indicating the drag-and-drop operation was successful. See Figure 91.

Figure 91 Modifying AS2UnpackagerPipeline Activity “StoreAS2ObjectServiceInvoke”



- 4 On the canvas, locate the **AS2MDNAssociationInvoke** activity (topmost branch, fourth from the End). In the project tree, open AS2MDNAssociation and drag its **execute** operation onto the **AS2MDNAssociationInvoke** activity. See Figure 92.

Figure 92 Modifying the AS2MDNAssociationInvoke Activity in AS2UnpackagerPipeline

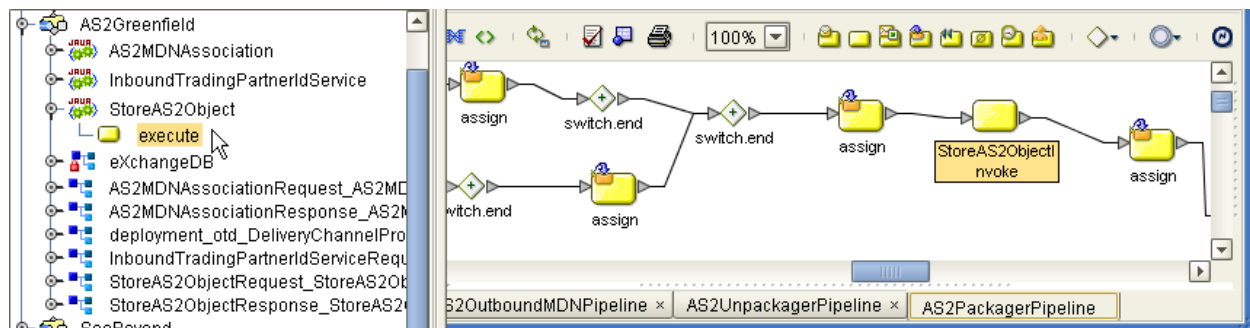


- 5 Save your changes, validate **AS2UnpackagerPipeline**, and, optionally, open its property sheet to delete two unused**Fault** attributes.

To modify the prebuilt AS2PackagerPipeline

- 1 In the project tree, under SeeBeyond > eXchange > MessagingProtocols > **AS2_Pipelines**, right-click **AS2PackagerPipeline**, check it out, and open it.
- 2 Scroll about two-thirds of the way through the pipeline, after both main branches join and before they split again, to find the **StoreAS2ObjectInvoke** activity.
- 3 In the project tree, under the **AS2Greenfield** project, open **StoreAS2Object** and drag its **execute** operation onto the **StoreAS2ObjectInvoke** activity. See Figure 93.

Figure 93 Modifying the StoreAS2ObjectInvoke Activity in AS2PackagerPipeline

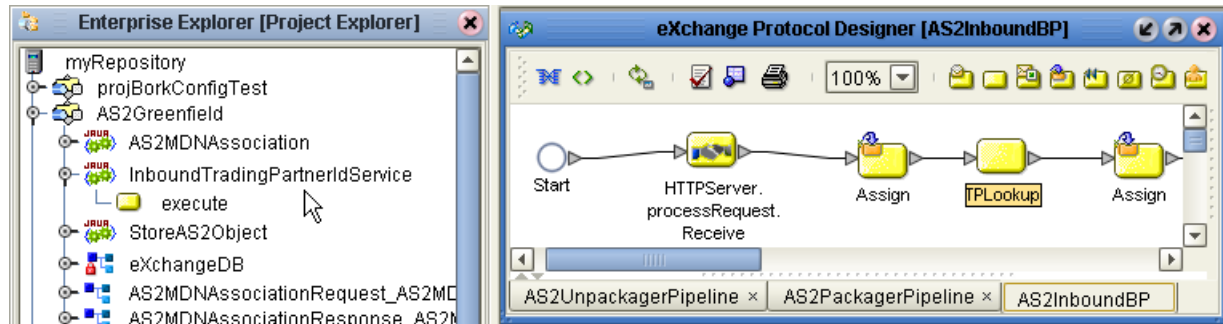


- 4 Save your changes, validate **AS2PackagerPipeline**, and, optionally, open its property sheet to delete the unused**Fault** attribute.

To modify the prebuilt AS2InboundBP pipeline

- 1 In the project tree, under SeeBeyond > eXchange > MessagingProtocols > **AS2_Pipelines**, right-click **AS2InboundBP**, check it out, and open it.
- 2 Locate the **TPLookup** activity (near Start, third from the left).
- 3 In the project tree, open **InboundTradingPartnerIdService** and drag its **execute** operation onto the **TPLookup** activity. See Figure 94.

Figure 94 Modifying the TPLookup Activity in AS2InboundBP



- 4 Save your changes, validate **AS2InboundBP**, and, optionally, open its property sheet to delete the unused**Fault** attribute.
- 5 Close all canvases and check in each of the four protocol pipelines you modified.

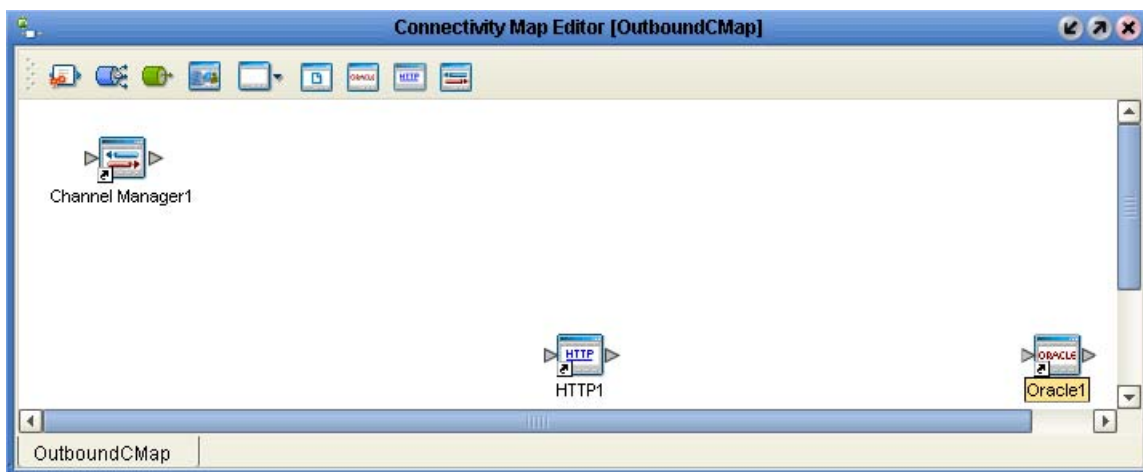
Now that you have saved, validated, and checked in your modifications to the four prebuilt protocol pipelines (AS2OutboundMDNPipeline, AS2UnpackagerPipeline, AS2PackagerPipeline, and AS2InboundBP), you can create services on a connectivity map and populate them with protocol pipelines and Java Collaboration Definitions.

8.2.3. Designing the Outbound Message Flow

To set up the outbound connectivity map and externals for Greenfield

- 1 In the project tree, right-click AS2Greenfield > **New > Connectivity Map**. Name it: **OutboundCMap**.
- 2 On the tool palette, click **External Applications** and add tools for Channel Manager, Oracle, and HTTP.
- 3 Drag a Channel Manager instance to the upper left of the connectivity map, an HTTP external to the lower center, and an Oracle external to the lower right, keeping the default names. See Figure 95.

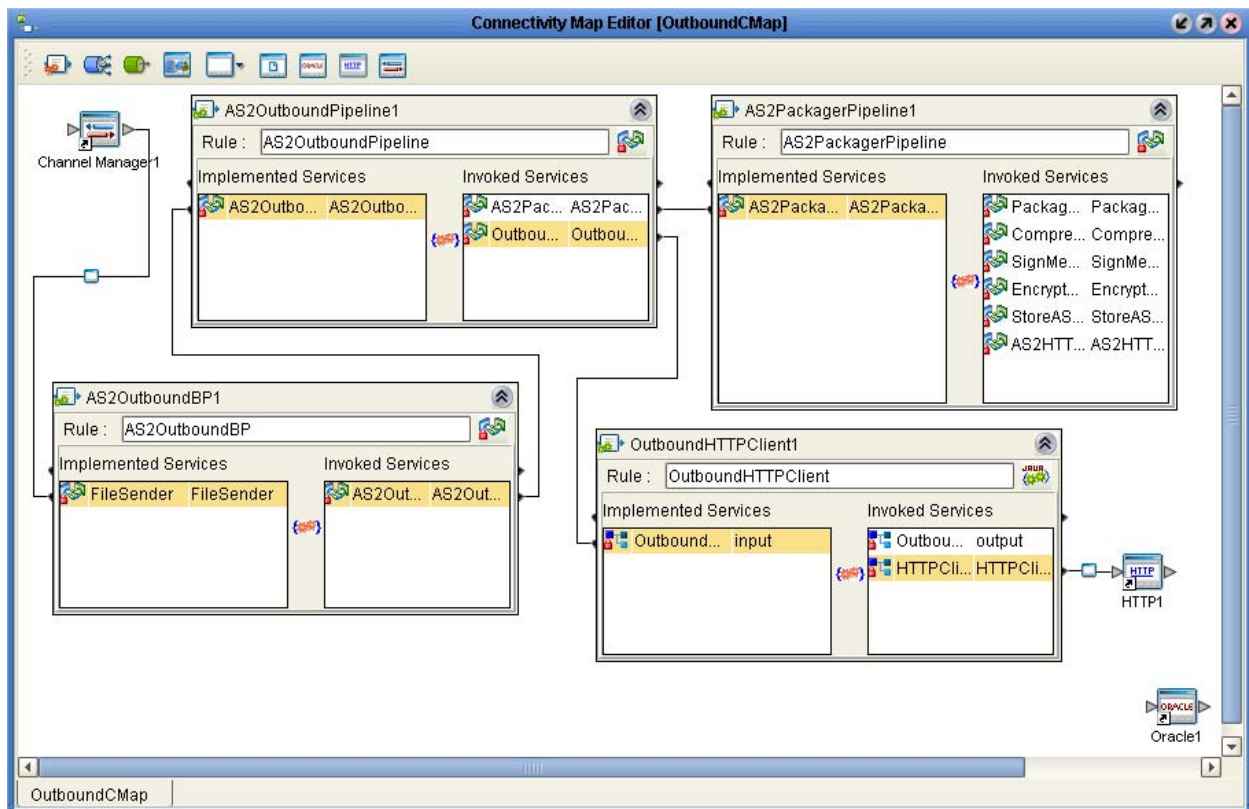
Figure 95 Outbound Connectivity Map With Three Externals



To connect OutboundCMap externals to protocol pipelines and JCDs

- 1 In the tree, under SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines, drag **AS2OutboundBP** onto the map, just below ChannelManager1.
- 2 Drag **AS2OutboundPipeline** onto the map, a bit to the right of ChannelManager1.
- 3 Drag **AS2PackagerPipeline** onto the map, far right of AS2OutboundPipeline.
- 4 Still under AS2_Pipelines, open **Collabs.** (If you did not check out all fourteen JCDs in step 2 of the [procedure on page 119](#), do so now.) Drag **OutboundHTTPClient** onto the map, below and to the right of AS2OutboundPipeline.
- 5 Double-click each pipeline to expand it, and then form links as seen in Figure 96:
 - ♦ For AS2OutboundBP: under Implemented Services, drag FileSender to the ChannelManager1 external. Double-click the eWay (small blue box) on the link, choose **Inbound Channel Manager**, keep default settings, and click **OK**.
 - ♦ For AS2OutboundBP: under Invoked Services, drag AS2OutboundPipeline to AS2OutboundPipeline1's Implemented Service, AS2OutboundPipeline.
 - ♦ For AS2OutboundPipeline1: under Invoked Services, drag AS2PackagerPipeline to AS2PackagerPipeline1's Implemented Service.
 - ♦ For OutboundHTTPClient1: under Invoked Services, drag HTTPClient_1 to the HTTP1 external. Double-click the eWay on the link; configure settings only if needed (for example, if SSL is used, or if a proxy is in effect), and click **OK**.

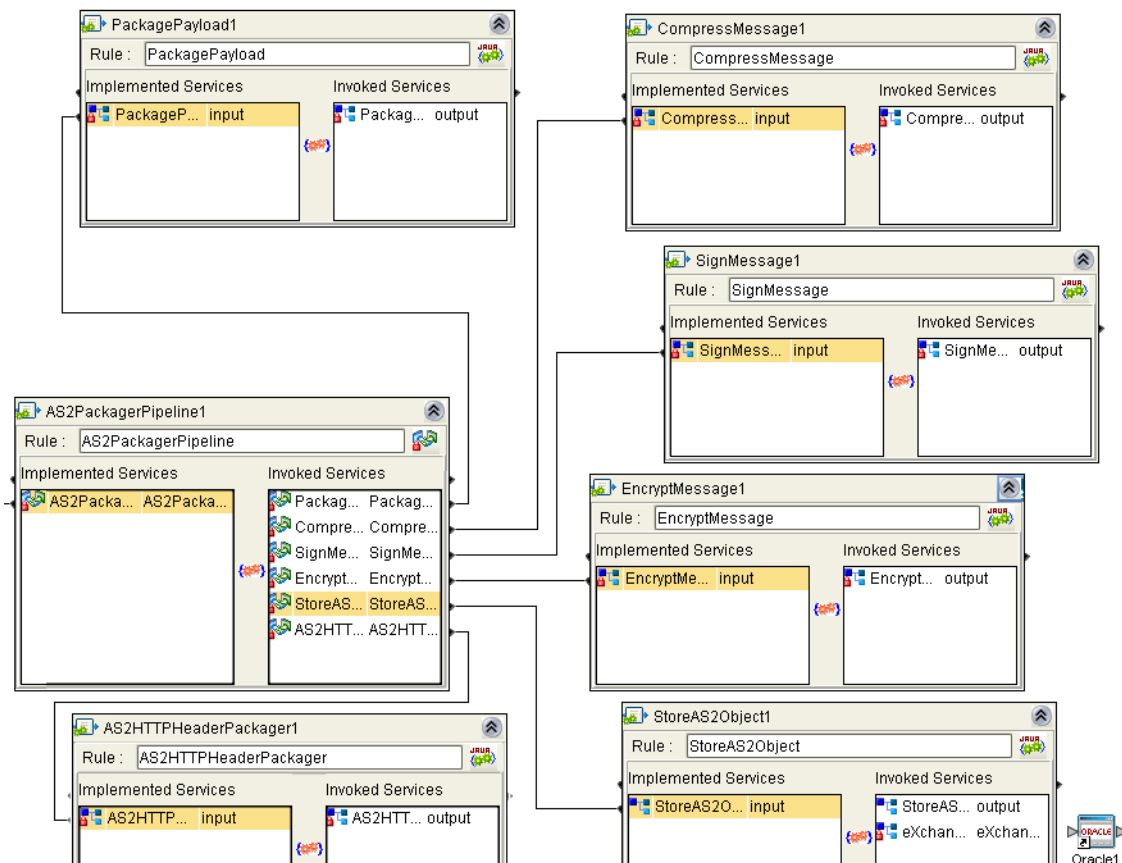
Figure 96 OutboundCMap with Four Protocol Pipelines Partly Linked



To add Java Collaborations to OutboundCMap

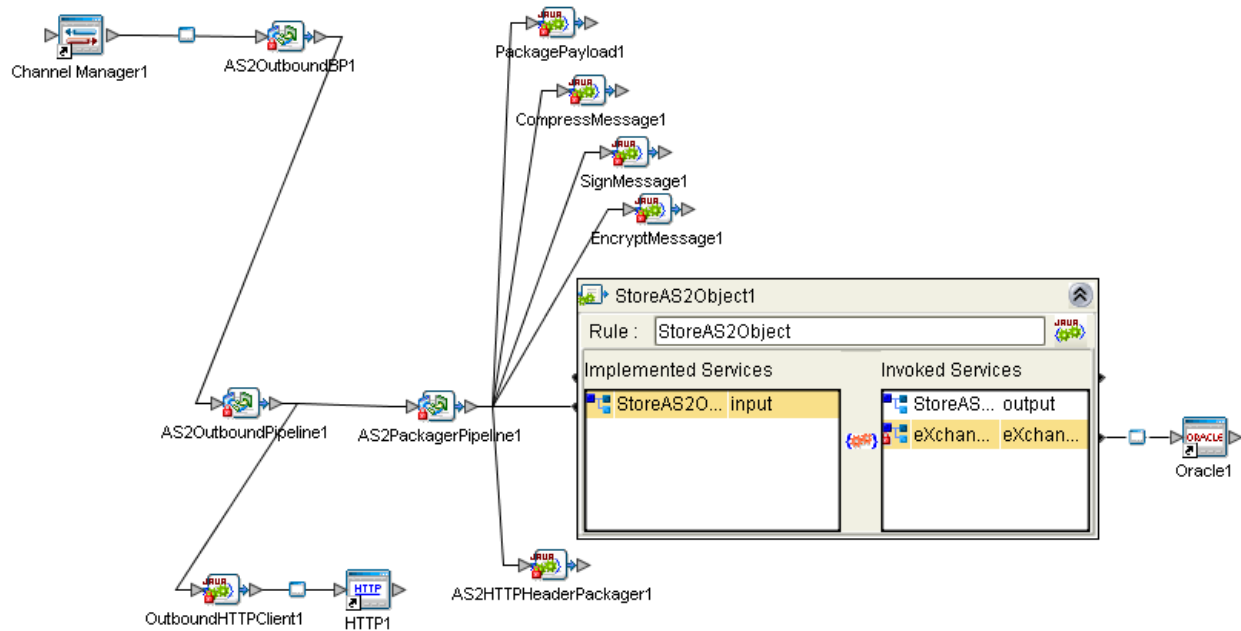
- 1 In the tree, under eXchange > MessagingProtocols > AS2_Pipelines > Collabs, drag the following five JCDs onto the map just to the right of AS2PackagerPipeline:
 - ◆ PackagePayload
 - ◆ CompressMessage
 - ◆ SignMessage
 - ◆ EncryptMessage
 - ◆ AS2HTTPHeaderPackage
- 2 From AS2Greenfield, drag **StoreAS2Object** onto the map, below EncryptMessage and above AS2HTTPHeaderPackage.
- 3 Create links from AS2PackagerPipeline1's Invoked Services to these six JCDs as shown in Figure 97:
 - ◆ From PackagePayload to PackagePayload (input for PackagePayload1)
 - ◆ From CompressMessage to CompressMessage (input for CompressMessage1)
 - ◆ From SignMessage to SignMessage (input for SignMessage1)
 - ◆ From EncryptMessage to EncryptMessage (input for EncryptMessage1)
 - ◆ From StoreAS2Object to StoreAS2Object (input for StoreAS2Object1)
 - ◆ From AS2HTTPHeaderPackage to AS2HTTPHeaderPackage (input for AS2HTTPHeaderPackage1)

Figure 97 OutboundCMap with AS2PackagerPipeline Linked to Six JCDs



- 4 Collapse AS2PackagerPipeline1 and all of the JCDs except StoreAS2Object1.
- 5 Connect StoreAS2Object1 to the Oracle external. Double-click the eWay; choose **Outbound Oracle eWay**, keep default JDBC Connector settings, and click **OK**. Rearrange so that the map looks like Figure 98.

Figure 98 Fully Linked OutboundCMap



8.2.4. Designing the Inbound Message Flow

To set up the inbound connectivity map and externals for Greenfield

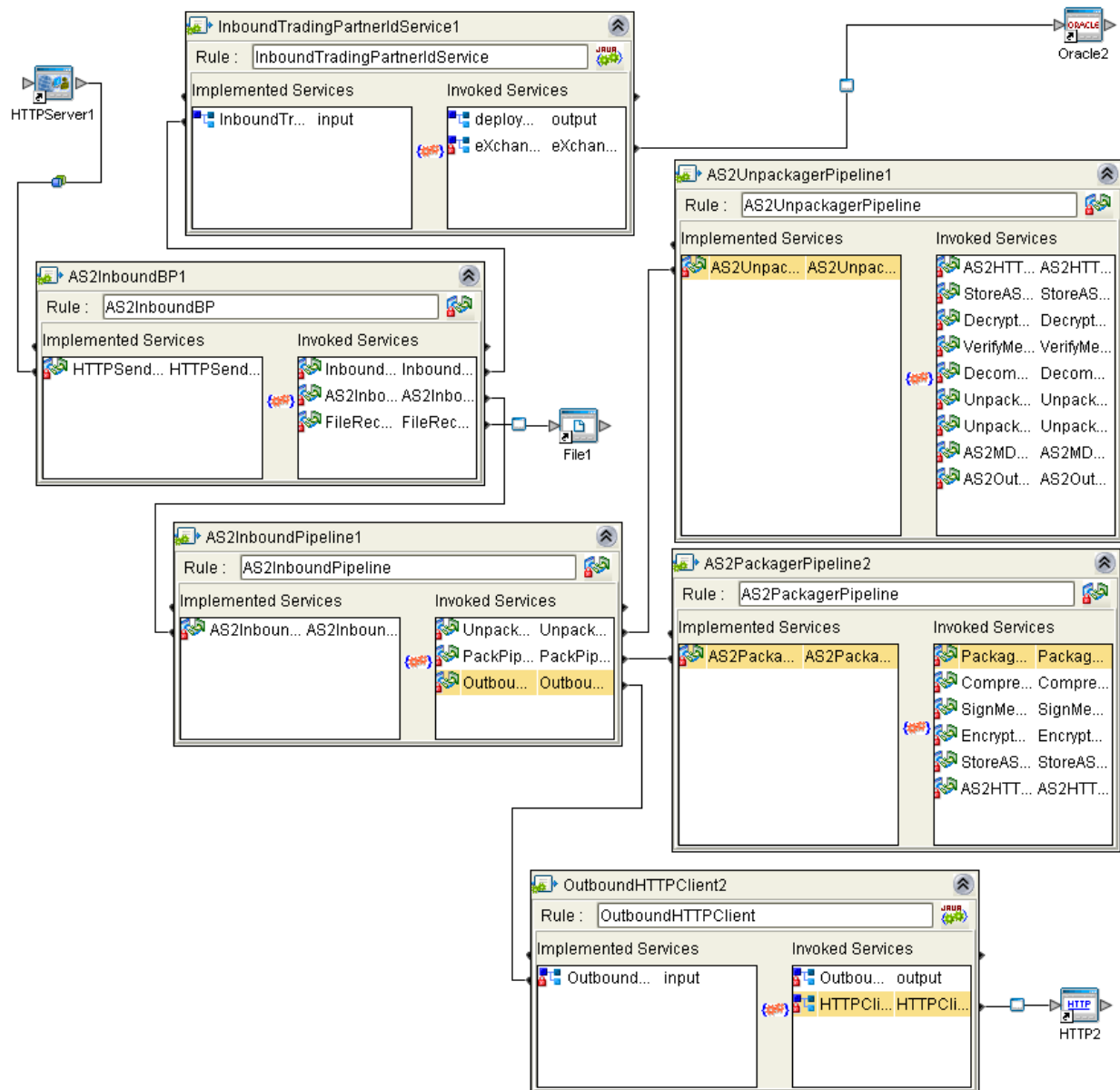
- 1 In the project tree, right-click AS2Greenfield > **New > Connectivity Map**. Name it: **InboundCMap**.
- 2 On the tool palette, click **External Applications** and add tools for Oracle, HTTP, HTTPServer, and File.
- 3 Drag an HTTPServer instance to the far upper left of the connectivity map, a File external to the far lower left, an Oracle external to the far upper right, and an HTTP to the far lower right. Keep the default names.

To add protocol pipelines and Java Collaboration Definitions to InboundCMap

- 1 In the tree, under SeeBeyond > eXchange > MessagingProtocols > AS2_Pipelines, drag **AS2InboundBP** onto the map, below HTTPServer.
- 2 Drag **AS2InboundPipeline** onto the map, to the mid center left.
- 3 Drag **AS2UnpackagerPipeline** and **AS2UnpackagerPipeline** onto the map, to the middle right and mid lower right.

- 4 Still under AS2_Pipelines, open **Collabs** and drag **OutboundHTTPClient** onto the map, to the lower center.
- 5 From the AS2Greenfield project, drag **InboundTradingPartnerIdService** onto the map, to the upper left center.
- 6 Double-click each pipeline and JCD to expand it. Form links as seen in Figure 99:
 - ♦ For AS2InboundBP1: Under Implemented Services, drag **HTTPSender** to the **HTTPServer1** external. Double-click the eWay (small blue box) on the link, edit the value for **servlet-url** from *servlet_name_here* to **Inbound**, and then click **OK**.
 - ♦ For AS2InboundBP1: Under Invoked Services, drag **FileReceiver** to the **File1** external. Double-click the eWay on the link, choose **Outbound File eWay**, edit values for Directory and Output file name to make them appropriate for your setup, change the default **Multiple records per file** value to **False**, and click **OK**.
 - ♦ For AS2InboundBP1: Drag Invoked Service **AS2InboundPipelineClient** down to AS2InboundPipeline1's Implemented Service, **AS2InboundPipelineClient**.
 - ♦ For AS2InboundBP1: Drag Invoked Service **InboundTradingPartnerIdService** up to InboundTradingPartnerIdService1's Implemented Service.
 - ♦ For InboundTradingPartnerIdService1: Drag Invoked Service **eXchangeDB_1** over to the **Oracle2** external. Double-click the eWay; choose **Outbound Oracle eWay**, keep default JDBC Connector settings, and click **OK**.
 - ♦ For AS2InboundPipeline1: Drag Invoked Service **UnpackPipeline** up to AS2UnpackagerPipeline1's Implemented Service.
 - ♦ For AS2InboundPipeline1: Drag Invoked Service **PackPipeline** over to AS2PackagerPipeline1's Implemented Service.
 - ♦ For AS2InboundPipeline1: Drag Invoked Service **OutboundHTTPClient** over to OutboundHTTPClient1's Implemented Service.
 - ♦ For OutboundHTTPClient2: Drag Invoked Service **HTTPClient_1** over to the **HTTP1** external. Double-click the eWay on the link; configure settings only if needed (for example, if SSL is used, or if a proxy is in effect), and click **OK**.

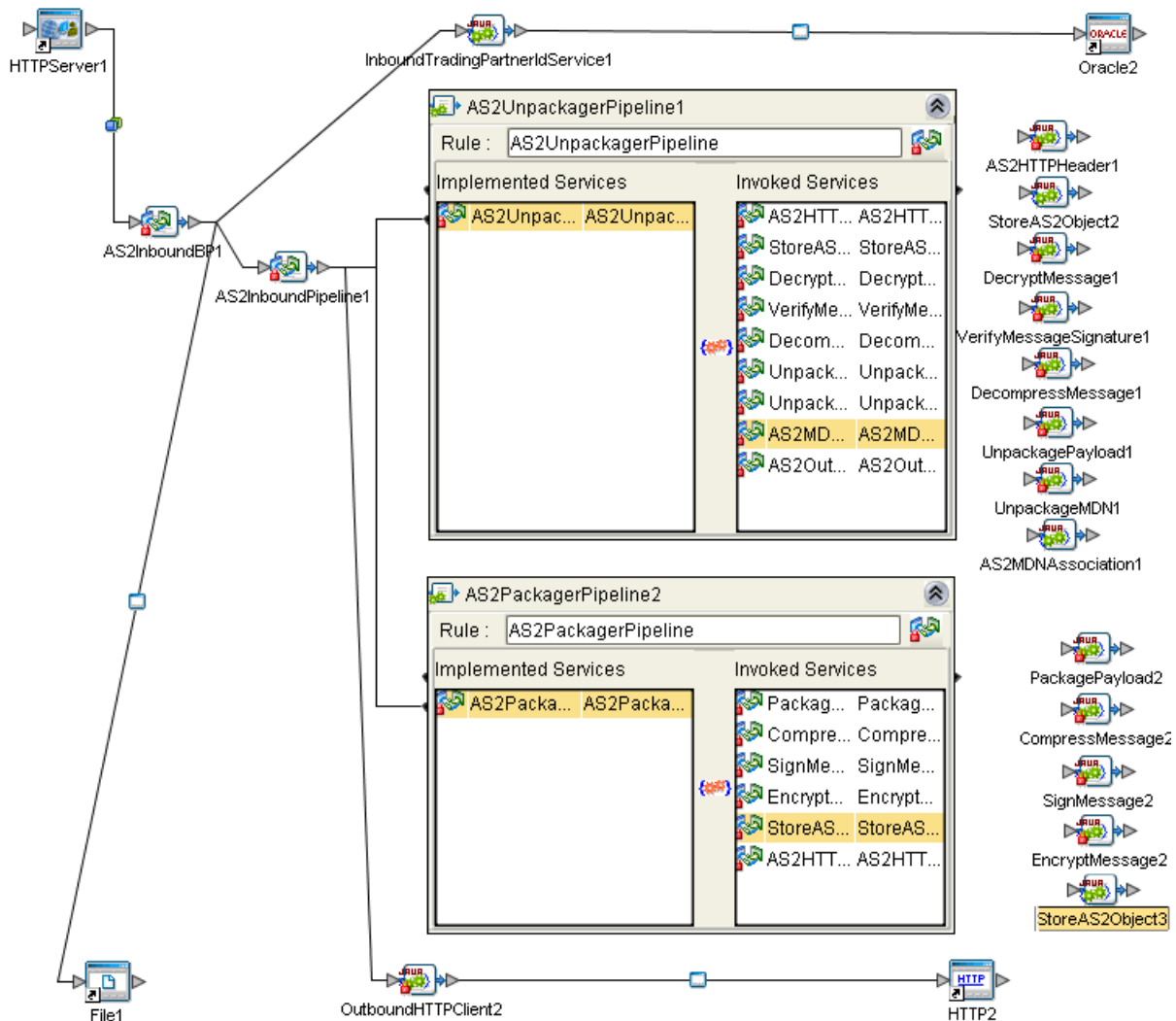
Figure 99 InboundCMap with Externals Linked to Protocol Pipelines and JCDs



- 7 Collapse the four leftmost components (that is, InboundTradingPartnerIdService1, AS2InboundBP1, AS2UnpackagerPipeline1, and OutboundHTTPClient2). The two pipelines that remain open (AS2UnpackagerPipeline1 and AS2PackagerPipeline1) still require further connections to JCDs. Rearrange as needed for clarity.

- 8 Locate the following JCDs and drag them onto the map as shown in Figure 100.
 - ◆ From SeeBeyond > eXchange > ... AS2_Pipelines > Collabs: **AS2HTTPHeader**
 - ◆ From project AS2Greenfield: **StoreAS2Object**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **DecryptMessage**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **VerifyMessageSignature**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **DecompressMessage**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **UnpackagePayload**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **UnpackageMDN**
 - ◆ From project AS2Greenfield: **AS2Greenfield**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **PackagePayload**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **CompressMessage**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **SignMessage**
 - ◆ From SeeBeyond > eXchange > ...Pipelines > Collabs: **EncryptMessage**
 - ◆ From project AS2Greenfield: **StoreAS2Object**

Figure 100 InboundCMap Showing Pipelines and JCDs, Not Yet Linked



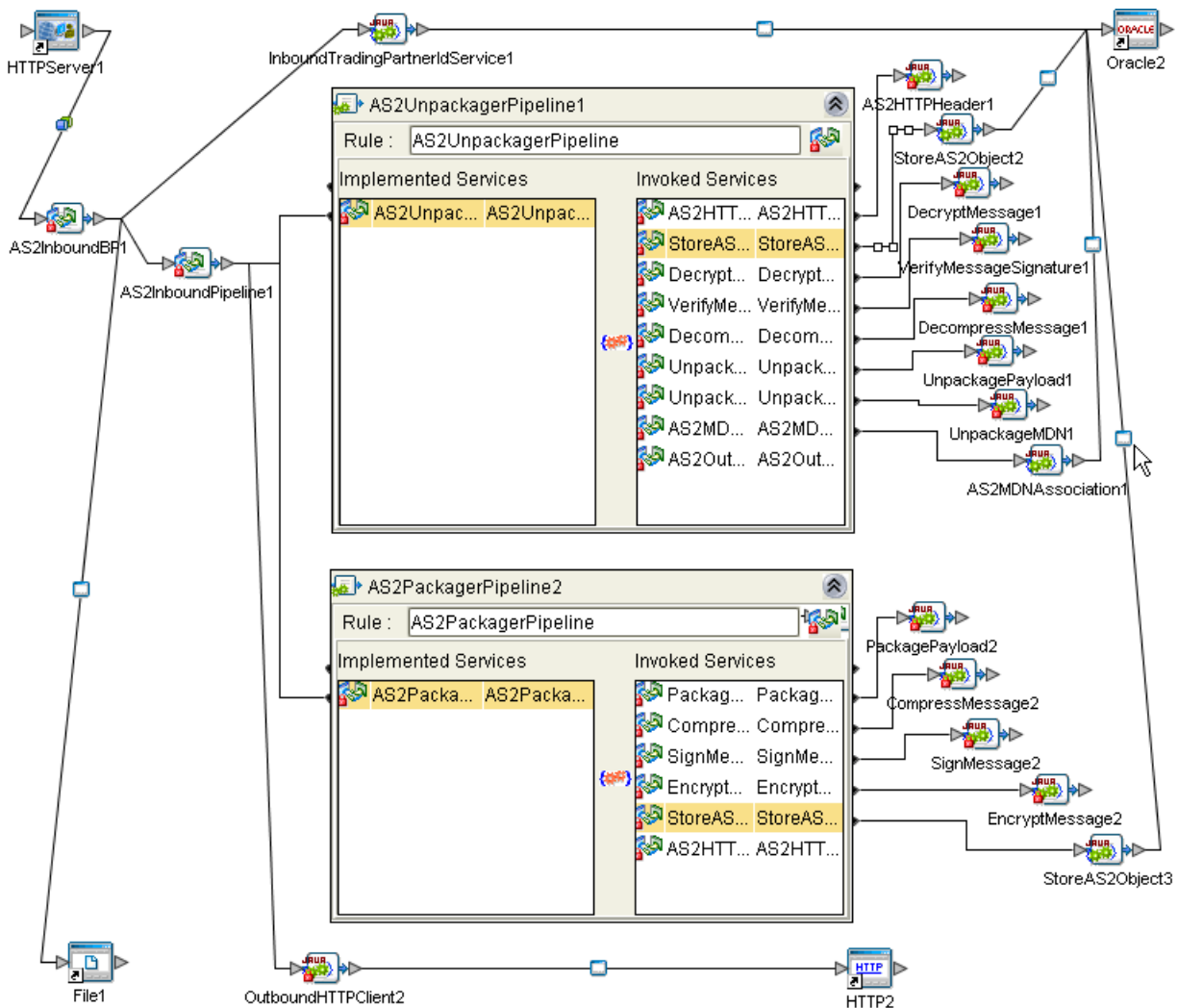
- 9 For each of ten JCDs from SeeBeyond > eXchange > ... > Collabs, do the following:
Expand the component; link its Implemented Service to the corresponding Invoked

Service on the right side of AS2UnpackagerPipeline1 or AS2PackagerPipeline1; and then recollapse the component.

- 10 For each the three JCDs from AS2Greenfield (that is, for the two StoreAS2Objects and for AS2MDNAssociation1), do the following: Expand the component; link its Implemented Service to the corresponding Invoked Service on the right side of AS2UnpackagerPipeline1 or AS2PackagerPipeline1; recollapse the component; and then double-click the Oracle eWay to make it outbound, with default JDBC Connector settings.

The map now looks like Figure 101.

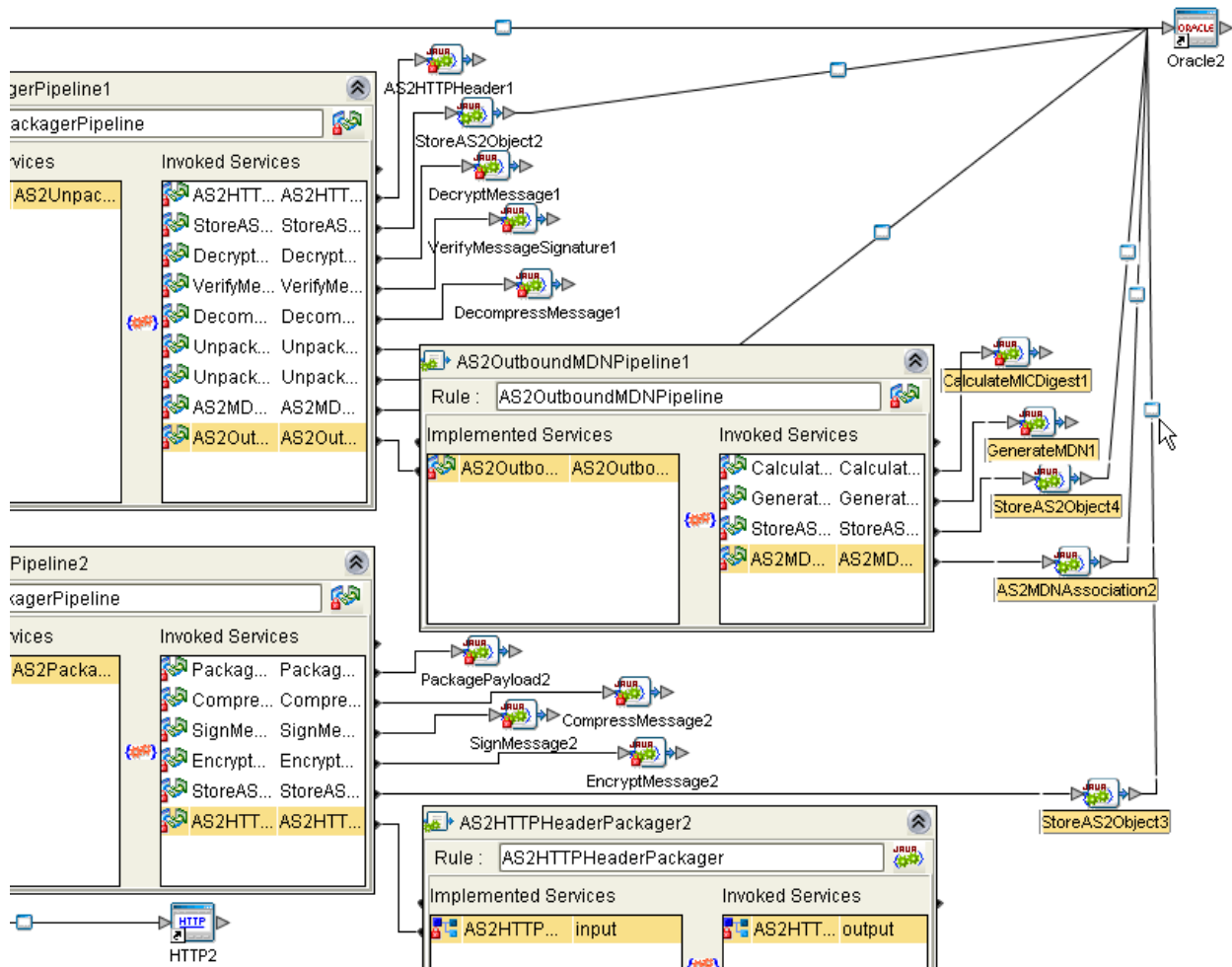
Figure 101 InboundCMap Showing Links To and From JCDs



- 11 Drag the following six components onto the map:
 - From SeeBeyond > ... AS2Pipelines > Collabs: Drag **AS2HTTPHeaderPackager** below and to the right of AS2PackagerPipeline1.
 - From SeeBeyond > ... AS2Pipelines: Drag **AS2OutboundMDNPipeline** below and to the right of AS2UnpackagerPipeline1.

- From SeeBeyond > ... AS2Pipelines > Collabs: Drag **CalculateMICDigest** above and to the right of AS2OutboundMDNPipeline1 (the preceding bullet item).
 - From SeeBeyond > ... AS2Pipelines > Collabs: Drag **GenerateMDN** to the right of AS2OutboundMDNPipeline1.
 - From the project (AS2Greenfield): Drag **StoreAS2Object** to the right of AS2OutboundMDNPipeline1.
 - From the project (AS2Greenfield): Drag **AS2MDNAssociation** below and to the right of AS2OutboundMDNPipeline1.
- 12 For both of the JCDs from SeeBeyond > eXchange > ... > Collabs, do the following: Expand the component; link its Implemented Service to the corresponding Invoked Service of AS2OutboundMDNPipeline1; and then recollapse the component.
 - 13 For both of the JCDs from AS2Greenfield, do the following: Expand the component; link its Implemented Service to the corresponding Invoked Service on the right side of AS2OutboundMDNPipeline1; recollapse the component; and then double-click the Oracle eWay to make it outbound, with default JDBC Connector settings.
 - 14 When you are done (see Figure 102), save your changes and close all canvases.

Figure 102 Final Components and Links for InboundCMap



After all logical components have been placed, linked, and configured, they need to be deployed to servers that have been set up in an environment.

8.2.5. Deploying Components to Servers in the Environment

This section assumes you have already set up BorkEnvironment with no more than the minimum set components, as follows:

- One outbound Oracle external, named **BorkOracle**
- One BatchFTP external, named **BorkBatchFTP**
- One Channel Manager bound to BorkHost, named **BorkHost1 ChannelManager**
- One logical host, named **LogicalHost1**, containing
 - ◆ One integration server, named **IntegrationSvr1**

It is also assumed that your logical host and integration server use default values for all parameters (such as port numbers in the 1800x range); if they do not, make appropriate adjustments in the following material to match your customized settings.

The Greenfield components require three additional servers in the environment: HTTPServer, HTTP, and outbound File.

To create additional external servers in BorkEnvironment

- 1 In Enterprise Designer, click the **Environment Explorer** tab (at the bottom margin).
- 2 Right-click BorkEnvironment and, on the popup menu, click **New HTTP Server External System**. Name it: **BorkHTTPServer**
- 3 Right-click BorkEnvironment and, on the popup menu, click **New HTTP External System**. Name it: **BorkHTTP**
- 4 If appropriate for your setup, customize the properties (such as SSL) of BorkHTTP.
- 5 Right-click BorkEnvironment and, on the popup menu, click **New File External System**. Name it: **BorkFileOut**, and specify it as outbound.
- 6 Save your changes and close all canvases.

To create a new deployment profile for the Greenfield scenario

- 1 In Enterprise Designer, click the **Project Explorer** tab (at the bottom margin).
- 2 Right-click AS2Greenfield > New > **Deployment Profile**. Name it **AS2Bork** and be sure that it references **BorkEnvironment** before you click **OK**.

The Deployment Editor opens, displaying all 46 unassigned logical components on the left and seven empty servers on the right.

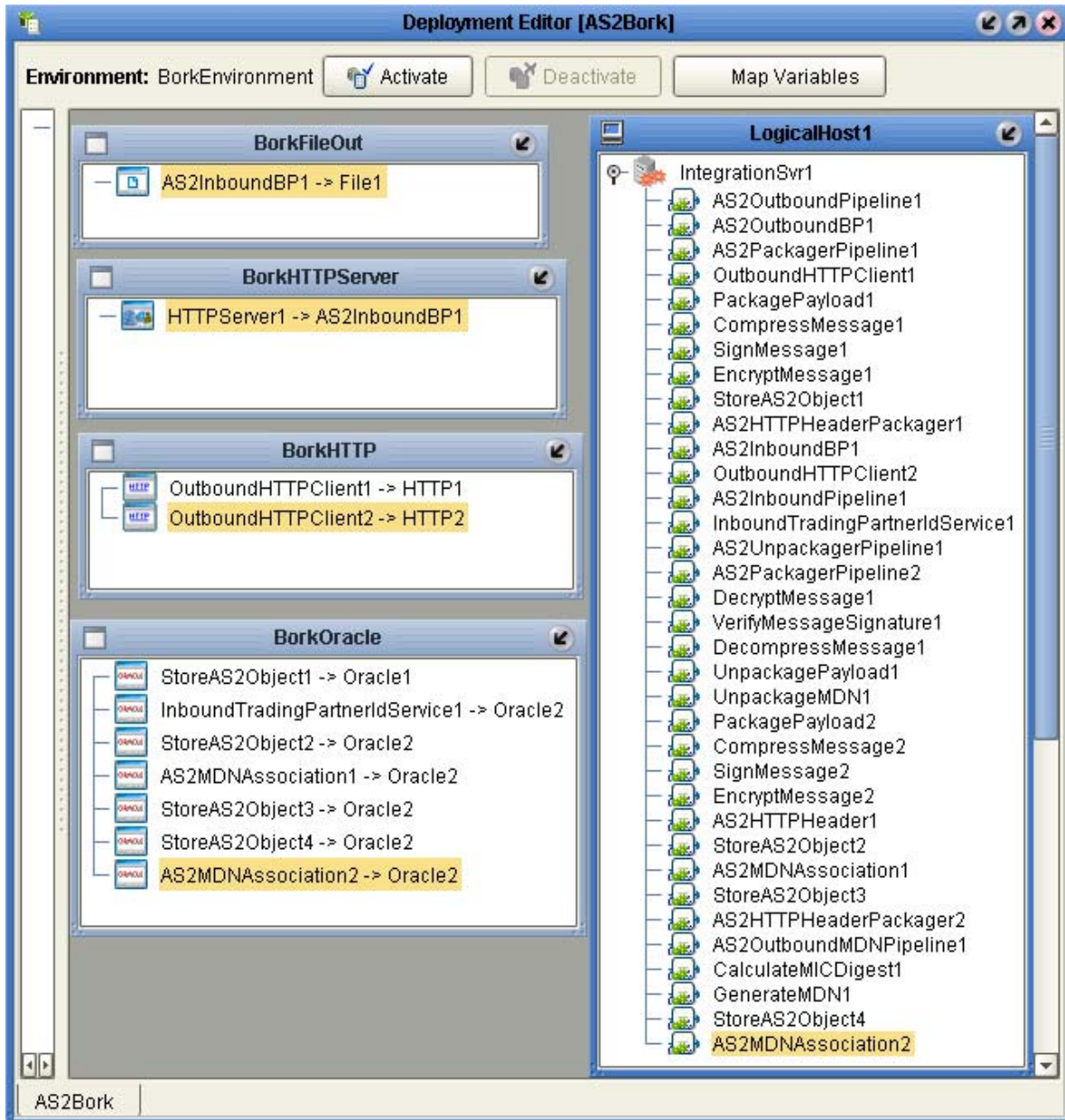
- 3 Minimize the servers BorkHost1 ChannelManager and BorkBatchFtp, since they are not used by the Greenfield scenario.

You will assign the 46 components to the five servers used by the Greenfield scenario as follows: one component apiece to BorkFileOut and BorkHTTPServer, two to BorkHTTP, seven to BorkOracle, and 35 (all of them pipelines and JCDs) to IntegrationSvr1.

To assign Greenfield components to servers

- 1 Drag the following outbound File eWay into BorkFileOut:
 - ♦ AS2InboundBP1 -> File
- 2 Drag the following HTTPServer eWay into BorkHTTPServer:
 - ♦ HTTPServer1 -> AS2InboundBP1
- 3 Drag the following two HTTP eWays into BorkHTTP:
 - ♦ OutboundHTTPClient1 -> HTTP1
 - ♦ OutboundHTTPClient2 -> HTTP2
- 4 Drag the following seven outbound Oracle eWays into BorkOracle:
 - ♦ StoreAS2Object1 -> Oracle1
 - ♦ InboundTradingPartnerIdService -> Oracle1
 - ♦ StoreAS2Object2 -> Oracle2
 - ♦ AS2MDNAssociation1 -> Oracle2
 - ♦ StoreAS2Object3 -> Oracle2
 - ♦ StoreAS2Object4 -> Oracle2
 - ♦ AS2MDNAssociation2 -> Oracle2
- 5 Drag the remaining 35 services (Java Collaborations and protocol pipelines) into IntegrationSvr1, under LogicalHost1, as shown in Figure 103.

Figure 103 The 46 Greenfield Components Assigned to External Servers



6 Save your changes, and then click **Activate**.

Method Palette

A.1 Operators

Figure 104 Method Palette: Operator tab

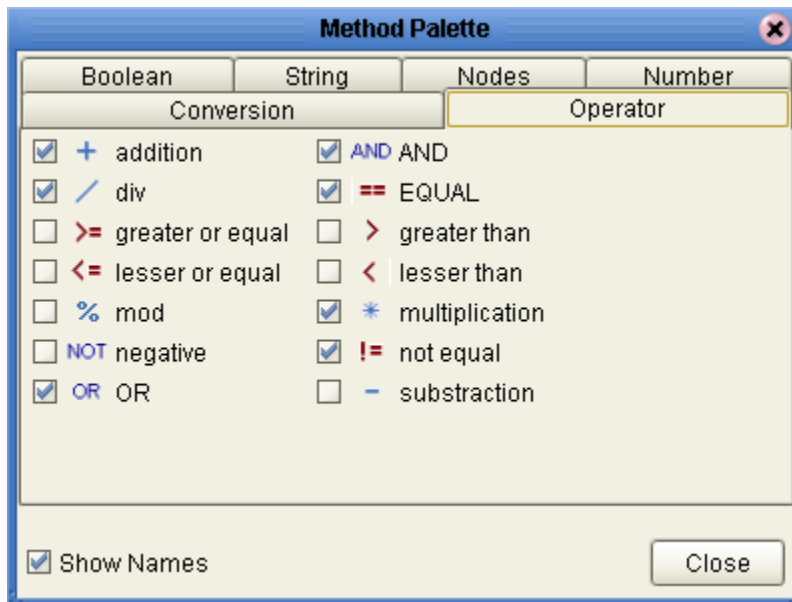


Table 11 Operator Methods

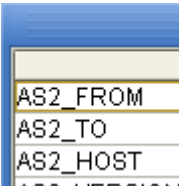
Method Box	Name	Description/Usage
	addition	Adds the value of <i>number1</i> to the value of <i>number2</i> , returns the sum.

Table 11 Operator Methods (Continued)

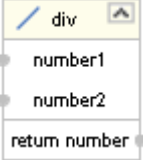
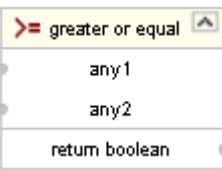
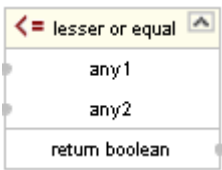
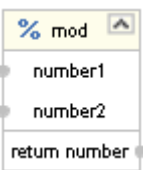
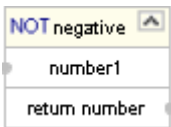
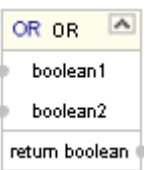
Method Box	Name	Description/Usage
 <p>The method box for the division operator shows a blue slash icon followed by the text 'div'. Below this are two input fields labeled 'number1' and 'number2', and a return field labeled 'return number'.</p>	division	Divides the value of <i>number1</i> by the value of <i>number2</i> , returns the quotient.
 <p>The method box for the greater_or_equal operator shows a red greater-than-or-equal icon followed by the text 'greater or equal'. Below this are two input fields labeled 'any1' and 'any2', and a return field labeled 'return boolean'.</p>	greater_or_equal	Returns Boolean true if <i>number1</i> is greater than or equal to <i>number2</i> ; otherwise, returns Boolean false.
 <p>The method box for the lesser_or_equal operator shows a red less-than-or-equal icon followed by the text 'lesser or equal'. Below this are two input fields labeled 'any1' and 'any2', and a return field labeled 'return boolean'.</p>	lesser_or_equal	Returns Boolean true if <i>number1</i> is less than or equal to <i>number2</i> ; otherwise, returns Boolean false.
 <p>The method box for the mod operator shows a blue percent icon followed by the text 'mod'. Below this are two input fields labeled 'number1' and 'number2', and a return field labeled 'return number'.</p>	mod	
 <p>The method box for the negative operator shows the text 'NOT negative' in blue. Below this is one input field labeled 'number1' and a return field labeled 'return number'.</p>	negative	Converts the input number to negative. Result is a negative number having the same absolute value as the input number.
 <p>The method box for the or operator shows the text 'OR OR' in blue. Below this are two input fields labeled 'boolean1' and 'boolean2', and a return field labeled 'return boolean'.</p>	or	Returns Boolean false if both <i>boolean1</i> and <i>boolean2</i> are false; otherwise, returns Boolean true.

Table 11 Operator Methods (Continued)

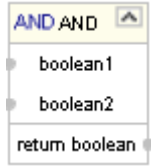
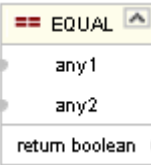
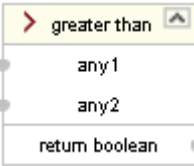
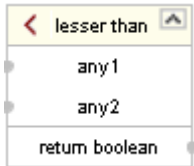
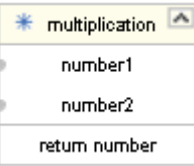
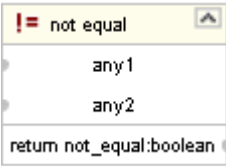
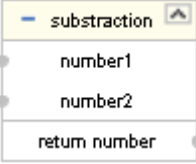
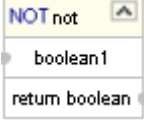
Method Box	Name	Description/Usage
	and	Returns Boolean true if both <i>boolean1</i> and <i>boolean2</i> are true; otherwise, returns Boolean false.
	equal	Returns Boolean true if <i>number1</i> is equal to <i>number2</i> ; otherwise, returns Boolean false.
	greater_than	Returns Boolean true if <i>number1</i> is greater than <i>number2</i> ; otherwise, returns Boolean false.
	less_than	Returns Boolean true if <i>number1</i> is less than <i>number2</i> ; otherwise, returns Boolean false.
	multiplication	Multiplies the value of <i>number1</i> by the value of <i>number2</i> , returns the product.
	not_equal	Returns Boolean true if <i>number1</i> is not equal to <i>number2</i> ; otherwise, returns Boolean false.

Table 11 Operator Methods (Continued)

Method Box	Name	Description/Usage
	subtraction	Subtracts the numerical value of <i>number2</i> from the numerical value of <i>number1</i> , returns the difference.
	not	Returns the inverse of <i>boolean1</i> .

2.2 String

Figure 105 Method Palette: String tab

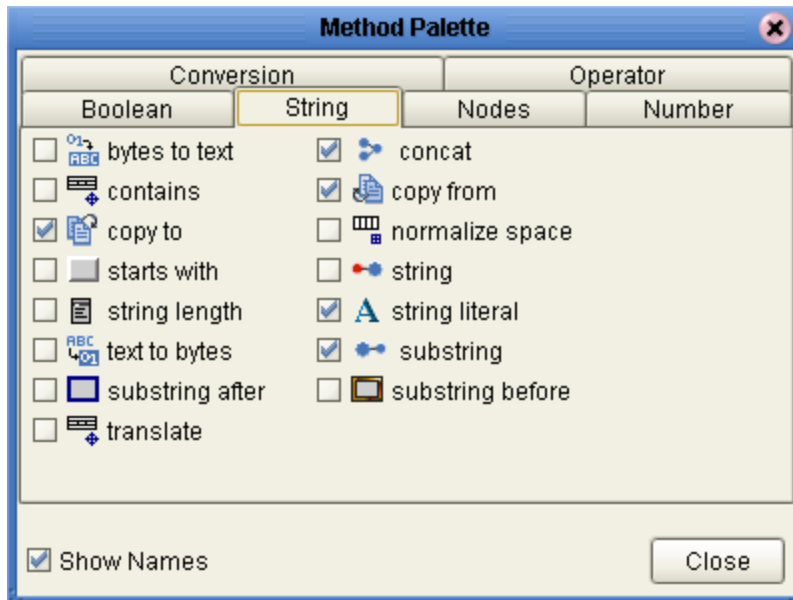


Table 12 String Methods

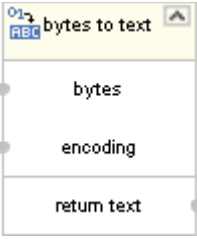

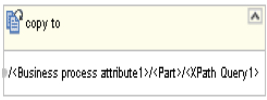
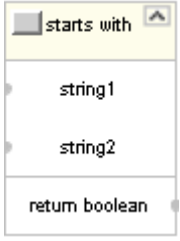
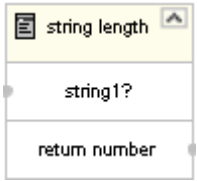
Symbol	Name	Description
	bytes to text	Decodes bytes into text using the specified encoding. If no encoding is specified, the platform's default encoding is used.
	contains	Returns true if the second string is contained within the first string, otherwise it returns false
	copy to	Allows you to type in the xpath expression for the destination of a copy operation. This is useful for entering xpath predicates. Note: This is for advanced users who are familiar with xpath and BPEL syntax.
	starts with	Returns true if the first string starts with the second string, otherwise it returns false
	string length	Returns the number of characters in a string

Table 12 String Methods (Continued)

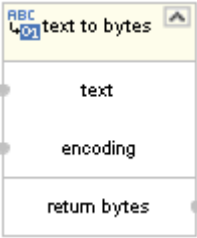
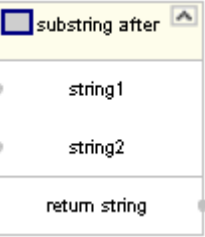

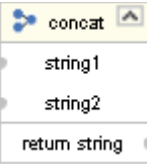
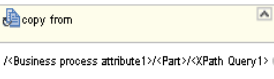
Symbol	Name	Description
	text to bytes	Encodes the input text into a sequence of bytes using the specified encoding. If no encoding is specified, the platform's default encoding is used
	substring after	Returns the part of the string in the string argument that occurs after the substring in the substring argument
	translate	Performs a character by character replacement. It looks in the value argument for characters contained in string1, and replaces each character for the one in the same position in the string2
	concat	Returns the concatenation of all its arguments
	copy from	Allows you to type in xpath expression for the source of a copy operation. This is useful for entering xpath predicates. Note: This is for advanced users who are familiar with xpath and BPEL syntax

Table 12 String Methods (Continued)

Symbol	Name	Description
	normalize space	Removes leading and trailing spaces from a string
	string	Converts the value argument to a string
	string literal	A sequence of characters of fixed length and content
	substring	Returns a part of the string in the string argument
	substring before	Returns the part of the string in the string argument that occurs before the substring in the substr argument

2.3 Number

Figure 106 Method Palette: Number tab

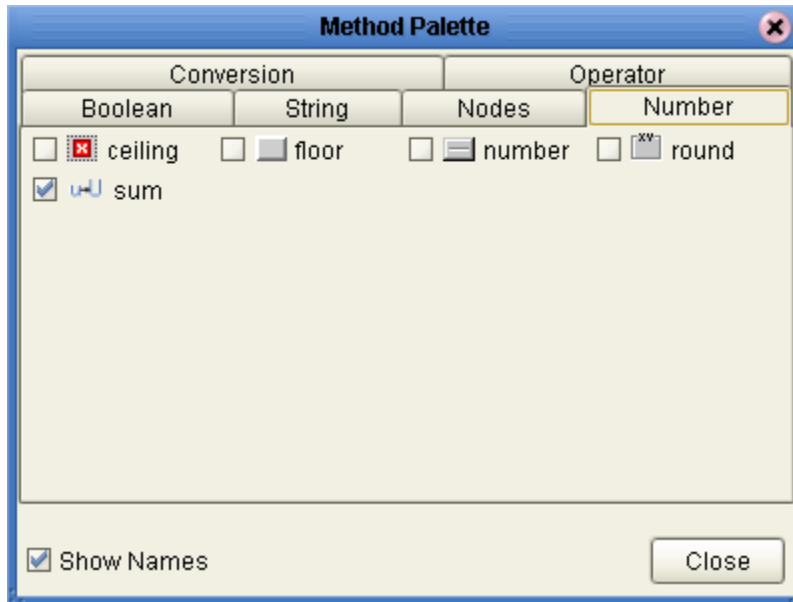


Table 13 Number Methods

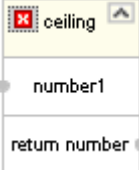
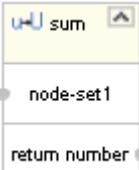

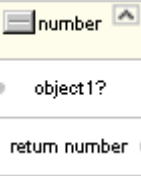
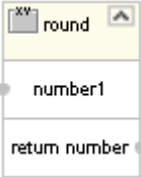
Symbol	Name	Function
	ceiling	Returns the smallest integer that is not less than the number argument
	sum	Returns the total value of a set of numeric values in a node-set

Table 13 Number Methods (Continued)

Symbol	Name	Function
 <p>The icon shows a yellow header with the text 'floor' and a small upward arrow. Below the header is a white box containing the parameter 'number1' and the return value 'return number'.</p>	<p>floor</p>	<p>Returns the largest integer that is not greater than the number argument</p>
 <p>The icon shows a yellow header with the text 'number' and a small upward arrow. Below the header is a white box containing the parameter 'object1?' and the return value 'return number'.</p>	<p>number</p>	<p>Converts the value argument to a number</p>
 <p>The icon shows a yellow header with the text 'round' and a small upward arrow. Below the header is a white box containing the parameter 'number1' and the return value 'return number'.</p>	<p>round</p>	<p>Rounds the number argument to the nearest integer</p>

2.4 Boolean

Figure 107 Method Palette: Boolean tab

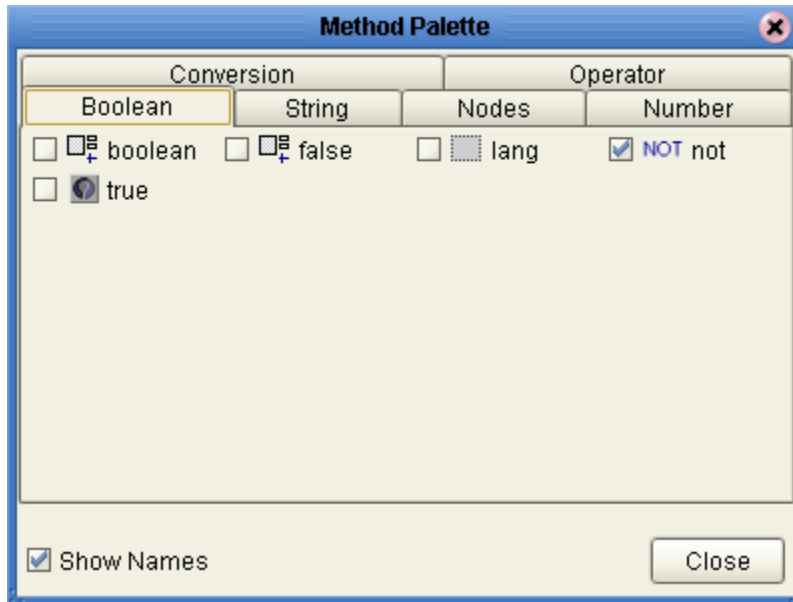

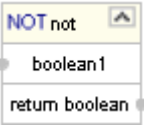


Table 14 Boolean Methods

Symbol	Name	Function
	boolean	Converts the value argument to Boolean and returns true or false
	true	Returns true
	false	Returns false

Table 14 Boolean Methods (Continued)

Symbol	Name	Function
	lang	Returns true if the language argument matches the language of the xsl:lang element, otherwise it returns false
	not	Returns true if the condition argument is false, and false if the condition argument is true

2.5 Nodes

Figure 108 Method Palette: Nodes tab

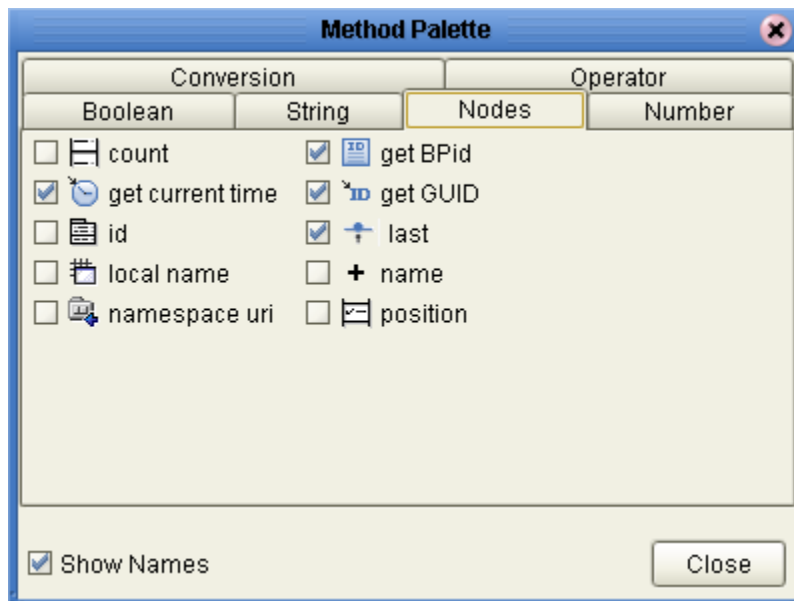


Table 15 Nodes Methods

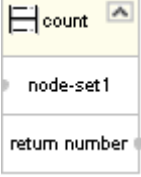
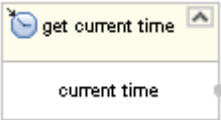
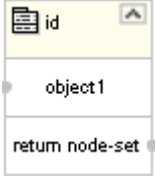
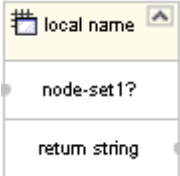
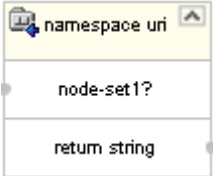


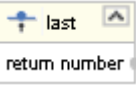
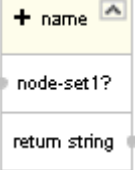
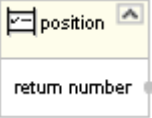
Symbol	Name	Function
	count	Returns the number of nodes in a node-set
	getCurrentTime	Gets the current time in ISO 8601 format (e.g. 2003-08-15T02:03:49.92Z).
	id	Selects elements by their unique ID
	local name	Returns the local part of a node. A node usually consists of a prefix, a colon, followed by the local name
	namespace uri	Returns the namespace URI of a specified node
	getBPId	Gets the B2B protocol pipeline instance ID.

Table 15 Nodes Methods (Continued)

Symbol	Name	Function
 <p>The symbol shows a method palette for 'get GUID'. It has a yellow header with a blue 'ID' icon, a plus sign, and an up arrow. Below the header, the text 'GUID' is displayed.</p>	getGUID	Gets a randomly generated globally unique ID.
 <p>The symbol shows a method palette for 'last'. It has a yellow header with a blue plus sign and an up arrow. Below the header, the text 'return number' is displayed.</p>	last	Returns the position number of the last node in the processed node list
 <p>The symbol shows a method palette for 'name'. It has a yellow header with a blue plus sign and an up arrow. Below the header, the text 'node-set 1?' and 'return string' are displayed.</p>	name	Returns the name of a node
 <p>The symbol shows a method palette for 'position'. It has a yellow header with a blue icon of a document with a checkmark and an up arrow. Below the header, the text 'return number' is displayed.</p>	position	Returns the position in the node list of the node that is currently being processed