

SeeBeyond ICAN Suite

ASC X12 Manager Composite Application User's Guide

Release 5.0



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

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

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 This work is protected as an unpublished work under the copyright laws.

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

Version 20041119081340.

Contents

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

List of Tables	9
----------------	---

Chapter 1

Introduction	10
About X12 Manager	10
About This Document	11
What's in This Document	11
Intended Audience	11
Document Conventions	11
Screenshots	12
Related Documents	12
SeeBeyond Web Site	13
SeeBeyond Documentation Feedback	13

Chapter 2

Overview of X12 Manager	14
About X12	14
What Is X12?	14
What Is a Message Structure?	15
Components of an X12 Envelope	15
Data Elements	16
Segments	16
Loops	16
Delimiters	17
Structure of an X12 Envelope	17
Transaction Set (ST/SE)	19
Functional Group (GS/GE)	19
Interchange Envelope (ISA/IEA)	20
Control Numbers	21
ISA13 (Interchange Control Number)	22
GS06 (Functional Group Control Number)	22
ST02 (Transaction Set Control Number)	22

Backward Compatibility	22
Example of EDI Usage	23
Overview of EDI Payments Processing	23
Types of Information That Is Exchanged Electronically	23
Types of Electronic Payment	24
Transfer of Funds	25
Payment-Related EDI Transactions	25
Acknowledgment Types	25
TA1, Interchange Acknowledgment	25
997, Functional Acknowledgment	26
Application Acknowledgments	26
Key Parts of EDI Processing Logic	26
Structures	27
Validations, Translations, Enveloping, Acknowledgments	27
Trading Partner Agreements	27
Additional Information	28
About eXchange Integrator	28
eXchange and the ICAN Suite	29
eXchange Architectural Overview	29
Process Overview	30
About B2B Protocol Manager Composite Applications	31
Composite Applications and the ICAN Suite	31
About Protocol Manager Composite Applications and eXchange	32
Example: X12 Protocol Manager Composite Application	34

Chapter 3

Installing X12 Manager	35
System Requirements	35
Supported Operating Systems	35
Database Support	36
Database for eXchange Partner Management and Message Tracking	36
Database for Persistence and Monitoring via eInsight Engine	36
Before You Install	36
Installing the Product Files	36
Uploading X12 Manager to the Repository	37
Refreshing Enterprise Designer	39
Database Scripts	42

Chapter 4

Using eXchange Partner Manager	43
Overview of the ePM Parameter Hierarchy	43
Parameters at the Trading Partner Level	44

Trading Partner > Properties	44
Trading Partner > Components: Overview	44
Trading Partner > Components > Delivery Channels	45
Parameters for Delivery Channels: “General” Tab	46
Parameters for Delivery Channels: “ToPartner Transport” Tab	46
Parameters for Delivery Channels: “FromPartner Transport” Tab	46
Parameters for Delivery Channels: “ToPartner Packaging” Tab	47
Parameters for Delivery Channels: “FromPartner Packaging” Tab	48
Trading Partner > Components > Enveloping Channels	49
Parameters for Enveloping Channels: “General” Tab	50
Parameters for Enveloping Bindings: “ToPartner Envelope” Subtab	50
About “cron” expressions	51
Parameters for Enveloping Bindings: “FromPartner DeEnvelope” Tab	52
Parameters at the Trading Partner Profile (TPP) Level	53
Parameters at the Business Service Level	54
Business Service > Business Actions	54
Business Service > Properties	54
Parameters at the Business Action Level	55

Chapter 5

Customizing a Validation Handler BP	58
Overview of the Exercise	58

Chapter 6

Implementation Scenario	61
Overview of the Sample Implementation	61
Initial Setup Steps	62
Installing the Sample Files for X12	62
Importing the Sample Projects	63
Design Steps in Enterprise Designer	64
Setting Up the Sample Environments	65
Viewing and Activating the B2B Host Project	67
Components of the B2B Host Project (X12_Host)	67
Creating and Activating the Project Deployment Profiles	71
Design Steps in ePM	80
Importing the Trading Partners into the Environments	81
Configuring Trading Partner Parameters for the “Berlin” TP	81
TP “Berlin”: Parameters for the Delivery Channel	82
Activating the Trading Partner	90
Configuring Trading Partner Parameters for the “Atlanta” TP	90
TP “Atlanta”: Parameters for the Delivery Channel	91
Activating the Trading Partner	98
Runtime Steps	99
Starting the Logical Hosts	99
Supplying the Input Data	100

Using the Message Tracking Facility	101
-------------------------------------	-----

Glossary

Glossary of Acronyms	106
----------------------	-----

108

OTD Syntax Validation Handlers	109
--------------------------------	-----

Activity Flow	109
---------------	-----

Fault Handling	112
----------------	-----

ValidateException	112
-------------------	-----

UnmarshalException	113
--------------------	-----

GenericException	113
------------------	-----

Other Faults	114
--------------	-----

Variables Referenced by OTD Validation Handler BPs	114
--	-----

The Value of the <code>#{BizRespCorrPath}</code> Variable	114
---	-----

Index	117
-------	-----

List of Figures

Figure 1	X12 Envelope Schematic	18
Figure 2	X12 997 (Functional Acknowledgment) Segment Table	18
Figure 3	Example of a Transaction Set Header (ST)	19
Figure 4	Example of a Transaction Set Trailer (SE)	19
Figure 5	Example of a Functional Group Header (GS)	20
Figure 6	Example of a Functional Group Trailer (GE)	20
Figure 7	Example of an Interchange Header (ISA)	21
Figure 8	Example of an Interchange Trailer (IEA)	21
Figure 9	eXchange and the ICAN Suite	29
Figure 10	eXchange Architecture	31
Figure 11	Composite Applications and the ICAN Suite	32
Figure 12	SeeBeyond-Supplied BP (for Processing Business Acknowledgments in X12)	33
Figure 13	Enterprise Manager ADMIN Page: Uploading the ProductsManifest.xml File	38
Figure 14	Update Center Wizard: Select Modules to Install	40
Figure 15	Update Center Wizard: Progress Bars	41
Figure 16	Update Center Wizard: Restart Enterprise Designer	41
Figure 17	ePM Parameter Hierarchy of TP > TP Profile > Service	43
Figure 18	Trading Partner Profile: "Properties" Tab	53
Figure 19	Accessing the Business Service Configuration Window	54
Figure 20	Business Service Configuration: "Business Actions" Tab	55
Figure 21	Opening the X12 v4021 219 Full Syntax Validation Handler	59
Figure 22	Mappings for the Request/Response Correlation Key	59
Figure 23	Project Explorer Tree, After Importing Sample Projects for X12	64
Figure 24	Environment Constant X12_DataDir for AtlantaEnv	66
Figure 25	Components of B2B Host Project ("X12_Host")	69
Figure 26	Auto-Mapping Services and eWays	70
Figure 27	Deployment Profile for B2B Host, Before Activation	70
Figure 28	Deployment Profile for Error Processing	72
Figure 29	Mapping Variables	72
Figure 30	Deployment Profile for Channel Manager Inbound From Partner	73
Figure 31	Assigning an eXchangeService Connection and Selecting Its Protocol	73
Figure 32	Assigning Components for an Atlanta Request Project (Feeder/Increment)	74

List of Figures

Figure 33	Activation of a Berlin Reply Project	75
Figure 34	Configuring InboundExchangeService Connection to the Validation Handler	76
Figure 35	Deployment Profile DP_X12_AtlantaEnv	78
Figure 36	Initial ePM Screen	81
Figure 37	List of Delivery Channel Bindings for Trading Partner “Berlin”	82
Figure 38	Delivery Channel for “Berlin”: General Properties for BLF_850_Out	83
Figure 39	Properties of Business Service “dlg_850_Out”	88
Figure 40	Business Actions for Business Service “dlg_850_Out”	88
Figure 41	Trading Partner “Atlanta” (in BerlinEnv)	91
Figure 42	Delivery Channel for “Atlanta”: General Properties for BLF_850_Out	92
Figure 43	Message Tracking, on Startup	102
Figure 44	Message Tracking, Showing Initial Search Results	103
Figure 45	Message Tracking, Showing Filtered Results	104
Figure 46	Message Tracking, Showing Package Details and Message Content	105

List of Tables

Table 1	Document Conventions	12
Table 2	Default Delimiters in X12 OTD Libraries	17
Table 3	Key Parts of EDI Processing	26
Table 4	Parameters for XDC Bindings: “General” Tab	46
Table 5	Parameters for XDC Bindings: “ToPartner Packaging” Tab	47
Table 6	Parameters for XDC Bindings: “FromPartner Packaging” Tab	48
Table 7	Parameters for Enveloping Channels: “ToPartner Envelope Binding” Subtab	50
Table 8	Parameters for Enveloping Channels: “FromPartner DeEnvelope Binding” Tab	52
Table 9	Parameters in the (TP->TPP->) “Properties” Tab	53
Table 10	Parameters in the (TP->TPP-> Business Service) “Business Actions” Tab	55
Table 11	List of Delivery Channels for Trading Partner “Berlin”	82
Table 12	ToPartnerTransport Parameters for Trading Partner “Berlin”	83
Table 13	FromPartnerTransport Parameters for Trading Partner “Berlin”	84
Table 14	ToPartnerPackaging Parameters for Trading Partner “Berlin”	84
Table 15	FromPartnerPackaging Parameters for Trading Partner “Berlin”	85
Table 16	ToPartner Enveloping Parameters for Trading Partner “Berlin”	86
Table 17	FromPartner Enveloping Parameters for Trading Partner “Berlin”	87
Table 18	Parameters for Trading Partner “Berlin”, Business Service “dlg_850_Out”	88
Table 19	List of Delivery Channels for Trading Partner “Atlanta”	91
Table 20	ToPartnerTransport Parameters for Trading Partner “Atlanta”	92
Table 21	FromPartnerTransport Parameters for Trading Partner “Atlanta”	93
Table 22	ToPartnerPackaging Parameters for Trading Partner “Atlanta”	93
Table 23	FromPartnerPackaging Parameters for Trading Partner “Atlanta”	94
Table 24	ToPartner Enveloping Parameters for Trading Partner “Atlanta”	95
Table 25	FromPartner Enveloping Parameters for Trading Partner “Atlanta”	96
Table 26	Parameters for Trading Partner “Atlanta”, Business Service “dlg_850_Out”	97
Table 27	Variables Referenced by OTD Validation Handler BPs	114

Introduction

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

What's in This Chapter

- [About X12 Manager](#) on page 10
- [About This Document](#) on page 11
- [Related Documents](#) on page 12
- [SeeBeyond Web Site](#) on page 13
- [SeeBeyond Documentation Feedback](#) on page 13

1.1 About X12 Manager

This user's guide provides instructions and background information for all users of the ASC X12 Manager Composite Application (X12 Manager). This guide is designed for managers, system administrators, and others who use X12 Manager.

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

- Understand the nature of X12 Manager.
- Understand the function of X12 Manager.
- Understand the relationship of X12 Manager to other components of the SeeBeyond Integrated Composite Application Network (ICAN) Suite.
- Learn about the X12 Manager components and how to use them in your environment.

What's New in This Release

ASC X12 Manager Composite Application is a new product at this release.

Version 5.0 is being initially released at the same time as version 5.0.5 of the ICAN suite, which includes version 5.0.5 of eXchange Integrator.

1.2 About This Document

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

1.2.1. What's in This Document

This guide includes the following chapters:

- **Chapter 1, "Introduction"** provides an overview of this document's purpose, contents, writing conventions, and supported documents.
- **Chapter 2, "Overview of X12 Manager"** discusses general features and architecture of X12 Manager.
- **Chapter 3, "Installing X12 Manager"** lists prerequisite that must be met before installation, and provides step-by-step instructions for installing the product files for X12 Manager and setting it up for use.
- **Chapter 4, "Using eXchange Partner Manager"** discusses the X12-specific items that can or must be configured in eXchange Partner Manager (ePM).
- **Chapter 5, "Customizing a Validation Handler BP"** is a short exercise to show how to locate and modify a SeeBeyond-supplied B2B protocol.
- **Chapter 6, "Implementation Scenario"** provides step-by-step procedures for creating implementation scenarios that provide a sample of how X12 Manager can be used to achieve B2B solutions.

The **"Glossary of Acronyms"** explains acronyms and initialisms in this guide, and **Appendix A ("OTD Syntax Validation Handlers")** provides in-depth technical information on B2B protocol processes for handling OTD syntax validation.

1.2.2. Intended Audience

This guide is intended for experienced computer users who have the responsibility of helping to set up and maintain a fully functioning ICAN Suite system. This person must also understand any operating systems on which the ICAN Suite will be installed (Windows, UNIX, and/or HP NonStop Server), and must be thoroughly familiar with Windows-style GUI operations.

1.2.3. Document Conventions

The following conventions are observed throughout this document.

Table 1 Document Conventions

Text	Convention	Example
Names of buttons, files, icons, parameters, variables, methods, menus, and objects	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> .	bootstrap -p <i>password</i>
Hypertext links	Blue text	See " Document Conventions "
Hypertext links for Web addresses (URLs) or email addresses	Blue underlined text	http://www.seebeyond.com docfeedback@seebeyond.com

1.2.4. Screenshots

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

1.3 Related Documents

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

Title	Filename
<i>SeeBeyond ICAN Suite Installation Guide</i>	ICAN_Install_Guide.pdf
<i>eGate Integrator 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>HTTP(S) eWay Intelligent Adapter User's Guide</i>	HTTPS_eWay.pdf
<i>Batch eWay Intelligent Adapter User's Guide</i>	Batch_eWay.pdf
<i>File eWay Intelligent Adapter User's Guide</i>	File_eWay.pdf
<i>eXchange Integrator User's Guide</i>	eXchange_User_Guide.pdf
<i>eXchange Integrator Developer's Guide</i>	eXchange_Developer_Guide.pdf
<i>X12 OTD Library User's Guide</i>	X12_OTD_Library.pdf
Readme for ICAN 5.0.5	Readme.txt

Also see the URLs provided in "[Additional Information](#)" .

1.4 SeeBeyond Web Site

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

<http://www.seebeyond.com>

1.5 SeeBeyond Documentation Feedback

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

docfeedback@seebeyond.com

Overview of X12 Manager

This chapter provides a general overview of the X12 Manager and its place in the ICAN Suite, including system descriptions, general operation, and basic features.

What's in This Chapter

- [About X12](#) on page 14
- [About eXchange Integrator](#) on page 28
- [About B2B Protocol Manager Composite Applications](#) on page 31

For more information about eGate, eXchange, or the X12 OTD Library, see the corresponding user's guides. For more information about X12, see the documentation issued by the standards organization.

2.1 About X12

This section provides the following information:

- An overview of X12, including the structure of an X12 envelope, data elements, and syntax.
- An explanation of how to use the generic message structures provided as an add-on to eGate to help you quickly create the structures you need for X12 transactions.
- An example of how X12 is used in payment processing.

2.1.1. What Is X12?

X12 is an EDI (electronic data interchange) standard, developed for the electronic exchange of machine-readable information between businesses.

The Accredited Standards Committee (ASC) X12 was chartered by the American National Standards Institute (ANSI) in 1979 to develop uniform standards for interindustry electronic interchange of business transactions—electronic data interchange (EDI). The result was the X12 standard.

An organization called the X12 body develops, maintains, interprets, and promotes the proper use of the ASC standard. Data Interchange Standards Association (DISA) publishes the X12 standard and the UN/EDIFACT standard. The X12 body comes together three times a year to develop and maintain EDI standards. Its main objective is to develop standards to facilitate electronic interchange relating to business

transactions such as order placement and processing, shipping and receiving information, invoicing, and payment information.

X12 was originally intended to handle large batches of transactions. However, it has been extended to encompass real-time processing (transactions sent individually as they are ready to send, rather than held for batching).

2.1.2. What Is a Message Structure?

The term *message structure* (also called a transaction set structure) refers to the way in which data elements are organized and related to each other for a particular EDI transaction.

In eGate, a message structure is called an Object Type Definition (OTD). Each message structure (OTD) consists of the following:

- Physical hierarchy

The predefined way in which envelopes, segments, and data elements are organized to describe a particular X12 EDI transaction.

- Delimiters

The specific predefined characters that are used to mark the beginning and end of envelopes, segments, and data elements.

- Properties

The characteristics of a data element, such as the length of each element, default values, and indicators that specify attributes of a data element—for example, whether it is required, optional, or repeating.

The transaction set structure of an invoice that is sent from one trading partner to another defines the header, trailer, segments, and data elements required by invoice transactions. The X12 OTD Library for a specific version includes transaction set structures for each of the transactions available in that version. You can use these structures as provided, or customize them to suit your business needs.

eGate Integrator uses Object Type Definitions based on the X12 message structures to verify that the data in the messages coming in or going out is in the correct format. There is a message structure for each X12 transaction.

The X12 body releases X12 versions on an approximately yearly basis, and the list of transactions provided is different for each version of X12. This book uses version 4010 and version 4021 as examples for illustrating how to install and work with X12 OTDs.

2.2 Components of an X12 Envelope

X12 messages are all ASCII text, with the exception of the BIN segment which is binary.

Each X12 message is made up of a combination of the following elements:

- Data elements

- Segments
- Loops

Elements are separated by delimiters.

More information on each of these is provided below.

2.2.1. Data Elements

The data element is the smallest named unit of information in the X12 standard. Data elements can be broken down into two types. The distinction between the two is strictly a matter of how they are used. The two types are:

- Simple
If a data element occurs in a segment outside the defined boundaries of a composite data structure, it is called a *simple* data element.
- Composite
If a data element occurs as an ordinaly positioned member of a composite data structure, it is called a *composite* data element. A telephone number is a simple example of a composite: It has an area code, which must precede the digits for the local exchange, which must precede the final group of digits.

Each data element has a unique reference number; it also has a name, description, data type, and minimum and maximum length.

2.2.2. Segments

A segment is a logical grouping of data elements. In X12, the same segment can be used for different purposes. This means that a field's meaning can change based on the segment. For example:

- The NM1 segment is for *any* name (patient, provider, organization, doctor)
- The DTP segment is for *any* date (date of birth, discharge date, coverage period)

For more information on the X12 enveloping segments, refer to [“Structure of an X12 Envelope” on page 17](#).

2.2.3. Loops

Loops are sets of repeating ordered segments. In X12 you can locate elements by specifying:

- The transaction set (for example, 270)
- The loop (for example, “loop 1000” or “info. receiver loop”)
- The occurrence of the loop
- The segment (for example, BGN)
- The field number (for example, 01)
- The occurrence of the segment (if it is a repeating segment)

2.2.4. Delimiters

In an X12 message, the various delimiters act as syntax, dividing up the different elements of a message. The delimiters used in the message are defined in the interchange control header, the outermost layer enveloping the message. For this reason, there is flexibility in the delimiters that are used.

No suggested delimiters are recommended as part of the X12 standards, but the industry-specific implementation guides do have recommended delimiters.

The default delimiters used by the SeeBeyond X12 OTD Libraries are the same as those recommended by the industry-specific implementation guides. These delimiters are shown in Table 2.

Table 2 Default Delimiters in X12 OTD Libraries

Type of Delimiter	Default Value
Segment terminator	~ (tilde)
Data element separator	* (asterisk)
Subelement (component) separator	: (colon)
Repetition separator (version 4020 and later)	+ (plus sign)

If you do not specify delimiters, eXchange expects the default delimiters as shown in Table 2.

***Note:** It is important to note that errors could result if the transmitted data itself includes any of the characters that have been defined as delimiters. Specifically, the existence of asterisks within transmitted application data is a known issue in X12, and can cause problems with translation.*

2.3 Structure of an X12 Envelope

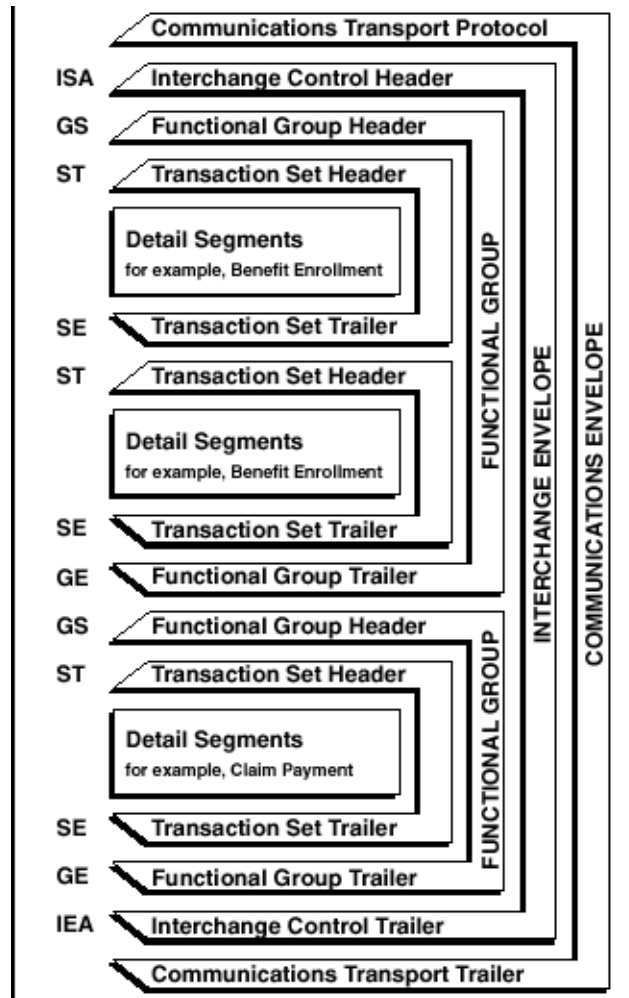
The rules applying to the structure of an X12 envelope are very strict, to ensure the integrity of the data and the efficiency of the information exchange.

The actual X12 message structure has three main levels. From the highest to the lowest they are:

- Interchange Envelope
- Functional Group
- Transaction Set

A schematic of X12 envelopes is shown in Figure 1. Each of these levels is explained in more detail in the following sections.

Figure 1 X12 Envelope Schematic



Note: The above schematic is from Appendix B of an X12 Implementation Guide.

Figure 2 shows the standard segment table for an X12 997 (Functional Acknowledgment) as it appears in the X12 standard and in most industry-specific implementation guides.

Figure 2 X12 997 (Functional Acknowledgment) Segment Table

Table 1 - Header

POS. #	SEG. ID	NAME	REQ. DES.	MAX USE	LOOP REPEAT
010	ST	Transaction Set Header	M	1	
020	AK1	Functional Group Response Header	M	1	
LOOP ID - AK2					999999
030	AK2	Transaction Set Response Header	O	1	
LOOP ID - AK2/AK3					999999
040	AK3	Data Segment Note	O	1	
050	AK4	Data Element Note	O	99	
060	AK5	Transaction Set Response Trailer	M	1	
070	AK9	Functional Group Response Trailer	M	1	
080	SE	Transaction Set Trailer	M	1	

2.3.1. Transaction Set (ST/SE)

Each transaction set (also called a transaction) contains three things:

- A transaction set header
- A transaction set trailer
- A single message, enveloped within the header and footer

The transaction has a three-digit code, a text title, and a two-letter code; for example, **997, Functional Acknowledgment (FA)**.

The transaction is composed of logically related pieces of information, grouped into units called segments. For example, one segment used in the transaction set might convey the address: city, state, postal code, and other geographical information. A transaction set can contain multiple segments. For example, the address segment could be used repeatedly to convey multiple sets of address information.

The X12 standard defines the sequence of segments in the transaction set and also the sequence of elements within each segment. The relationship between segments and elements could be compared to the relationship between records and fields in a database environment.

Figure 3 Example of a Transaction Set Header (ST)

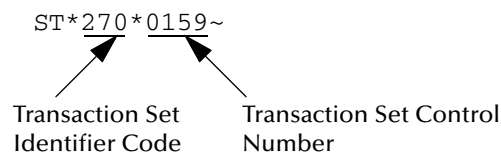
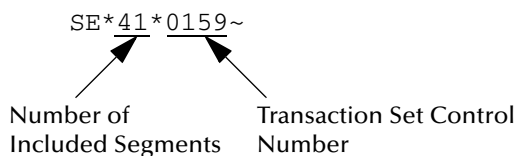


Figure 4 Example of a Transaction Set Trailer (SE)



2.3.2. Functional Group (GS/GE)

A functional group is composed of one or more transaction sets, all of the same type, that can be batched together in one transmission. The functional group is defined by the header and trailer; the Functional Group Header (GS) appears at the beginning, and the Functional Group Trailer (GE) appears at the end. Many transaction sets can be included in the functional group, but all transactions must be of the same type.

Within the functional group, each transaction set is assigned a functional identifier code, which is the first data element of the header segment. The transaction sets that constitute a specific functional group are identified by this functional ID code.

The functional group header (GS) segment contains the following information:

- Functional ID code (the two-letter transaction code; for example, PO for an 850 Purchase Order, HS for a 270 Eligibility, Coverage or Benefit Inquiry) to indicate the type of transaction in the functional group
- Identification of sender and receiver
- Control information (the functional group control numbers in the header and trailer segments must be identical)
- Date and time

The functional group trailer (GE) segment contains the following information:

- Number of transaction sets included
- Group control number (originated and maintained by the sender)

Figure 5 Example of a Functional Group Header (GS)

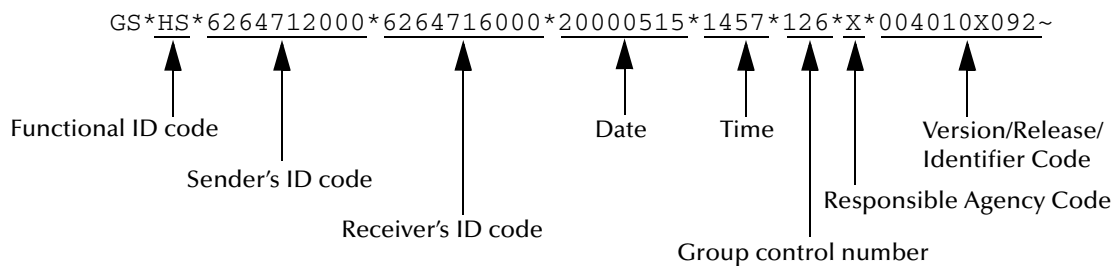
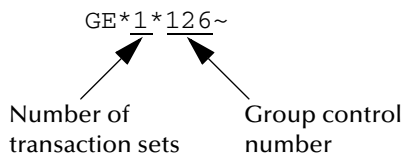


Figure 6 Example of a Functional Group Trailer (GE)



2.3.3. Interchange Envelope (ISA/IEA)

The interchange envelope is the wrapper for all the data to be sent in one transmission. It can contain multiple functional groups. This means that transactions of different types can be included in the interchange envelope, with each type of transaction stored in a separate functional group.

The interchange envelope is defined by the header and trailer; the Interchange Control Header (ISA) appears at the beginning, and the Interchange Control Trailer (IEA) appears at the end.

As well as enveloping one or more functional groups, the interchange header and trailer segments include the following information:

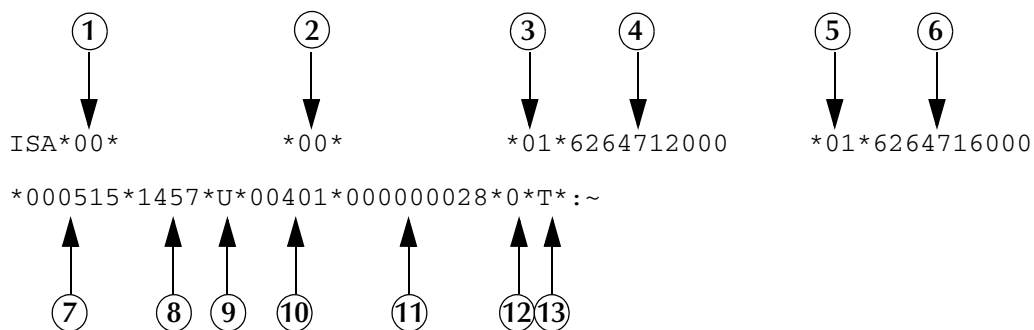
- Data element separators and data segment terminator

- Identification of sender and receiver
- Control information (used to verify that the message was correctly received)
- Authorization and security information, if applicable

The sequence of information that is transmitted is as follows:

- Interchange header
- Optional interchange-related control segments
- Actual message information, grouped by transaction type into functional groups
- Interchange trailer

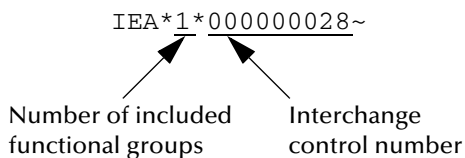
Figure 7 Example of an Interchange Header (ISA)



Interchange Header Segments from Figure 7:

- | | |
|---------------------------------------|---------------------------------------|
| 1 Authorization Information Qualifier | 8 Time |
| 2 Security Information Qualifier | 9 Repetition Separator |
| 3 Interchange ID Qualifier | 10 Interchange Control Version Number |
| 4 Interchange Sender ID | 11 Interchange Control Number |
| 5 Interchange ID Qualifier | 12 Acknowledgment Requested |
| 6 Interchange Receiver ID | 13 Usage Indicator |
| 7 Date | |

Figure 8 Example of an Interchange Trailer (IEA)



2.3.4. Control Numbers

The X12 standard includes a control number for each enveloping layer:

- ISA13—Interchange Control Number
- GS06—Functional Group Control Number
- ST02—Transaction Set Control Number

The control numbers act as identifiers, useful in message identification and tracking. eXchange Integrator includes a flag for each control number, so you can choose not to assign control numbers to outgoing messages and not to store control numbers on incoming messages.

ISA13 (Interchange Control Number)

The ISA13 is assigned by the message sender. It must be unique for each interchange. This is the primary means used by eXchange Integrator to identify an individual interchange.

GS06 (Functional Group Control Number)

The GS06 is assigned by the sender. It must be unique within the Functional Group assigned by the originator for a transaction set.

Note: *The Functional Group control number GS06 in the header must be identical to the same data element in the associated Functional Group trailer, GE02.*

ST02 (Transaction Set Control Number)

The ST02 is assigned by the sender, and is stored in the transaction set header. It must be unique within the Functional Group.

Note: *The control number in ST02 must be identical with the SE02 element in the transaction set trailer, and must be unique within a Functional Group (GS-GE). Once you have defined a value for SE02, eXchange Integrator uses the same value for SE02.*

2.4 Backward Compatibility

Each version of X12 is slightly different. Each new version has some new transactions; in addition, existing transactions might have changed.

New versions of X12 are usually backward compatible; however, this is not a requirement of the X12 rules. You should not expect different versions of X12 to be backward compatible, but you can expect that when you analyze the differences only a few changes are required in the message structures.

Note: *In this context, backward compatible means that software that parses one version might not be able to parse the next version, even if the software ignores any unexpected new segments, data elements at the end of segments, and sub-elements at the end of composite data elements. Not backward compatible means that required segments can disappear entirely, data elements can change format and usage, and required data elements can become optional.*

2.5 Example of EDI Usage

This section provides an overview of the normal processes involved in EDI payment processing.

Note: This section is a general overview of how electronic payments processing is used. Not everything in this section applies to the use of X12 in processing payments.

2.5.1. Overview of EDI Payments Processing

EDI payments processing encompasses both collection and disbursement transactions. The exchange of funds is accomplished by means of credit and debit transfers. It can also include a related bank balance, as well as transaction and account analysis reporting mechanisms.

Most non-monetary EDI trading partner communications are handled either directly between the parties or indirectly through their respective value added networks (VANs). However, the exchange of funds requires a financial intermediary. This is normally the bank or banks that hold deposit accounts of the two parties.

EDI involves the exchange of remittance information along with the order to pay. In the United States this can become complex as two standards are involved in the transaction. The remittance information, which acts as an electronic check stub, can be sent in any of the following ways:

- Directly between trading partners or through their respective EDI VAN mailboxes
- Through the banking system, with the beneficiary's bank sending notice of payment to the beneficiary
- By the originator to the originator's bank as an order to pay, with the originator's bank notifying the beneficiary

The trading partners and the capabilities of their respective banks determine the following:

- The routing of the electronic check stub
- Which of the following the payment is:
 - ♦ a debit authorized by the payor and originated by the beneficiary
 - ♦ a credit transfer originated by the payor

Types of Information That Is Exchanged Electronically

There are several types of information that can be exchanged electronically between bank and customer, including:

- Daily reports of balances and transactions
- Reports of lockbox and EFT (electronic funds transfer) remittances received by the bank
- Authorizations issued to the bank to honor debit transfers

- Monthly customer account analysis statements
- Account reconciliation statements
- Statements of the demand deposit account

The electronic payment mechanism, which is a subset of EDI, involves two separate activities:

- The exchange of payment orders, causing value to transfer from one account to another
- The exchange of related remittance information in standardized machine-processable formats.

Types of Electronic Payment

The electronic payment can be either of the following:

- Credit transfer, initiated by the payor
- Debit transfer, initiated by the payee as authorized by the payor

Regardless of how the credit transfer was initiated, the payor sends a payment order to its bank in the form of an X12 Payment Order/Remittance Advice (transaction set 820).

The bank then adds data in a format prescribed in the United States by the National Automated Clearing House Association (NACHA) and originates the payment through the Automated Clearing House (ACH) system.

A corporate-to-corporate payment performs two functions:

- Transfers actual monetary value
- Transfers notification of payment from payor to payee

When a credit transfer occurs, these two functions are sometimes treated as one, and sometimes treated separately. The two functions can travel in either of these two ways:

- Together through the banking system
- Separately and by different routes

X12 820 is a data format for transporting a payment order from the originator to its bank. This payment order might be either of the following:

- An instruction to the originator's bank to originate a credit transfer
- An instruction to the trading partner to originate a debit transfer against the payor's bank account

Once this decision has been made, the 820 transports the remittance information to the beneficiary. The transfer can either be through the banking system or by a route that is separate from the transport of funds.

Note: *Whenever the 820 remittance information is not transferred with the funds, it can be transmitted directly from the originator to the beneficiary. It can also be transmitted through an intermediary, such as a VAN.*

Transfer of Funds

Before funds can be applied against an open accounts receivable account, the beneficiary must reconcile the two streams—the payment advice from the receiving bank and the remittance information received through a separate channel—that were separated during the transfer. If this reconciliation does not take place and if the amount of funds received differs from the amount indicated in the remittance advice, the beneficiary might have problems balancing the accounts receivable ledger.

The value transfer begins when the originator issues a payment order to the originator's bank. If a credit transfer is specified, the originator's bank charges the originator's bank account and pays the amount to the beneficiary's bank for credit to the beneficiary's account.

If the payment order specifies a debit transfer, the originator is the beneficiary. In this case, the beneficiary's bank originates the value transfer, and the payor's account is debited (charged) for a set amount, which is credited to the originator's (beneficiary's) bank account. The payor must issue approval to its bank to honor the debit transfer, either before the beneficiary presents the debit transfer or at the same time. This debit authorization or approval can take one of four forms:

- Individual item approval
- Blanket approval of all incoming debits with an upper dollar limit
- Blanket approval for a particular trading partner to originate any debit
- Some combination of the above

2.5.2. Payment-Related EDI Transactions

X12 uses an end-to-end method to route the 820 Payment Order/Remittance Advice from the originator company through the banks to the beneficiary. This means that there might be several relay points between the sender and the receiver.

The 820 is wrapped in an ACH banking transaction for the actual funds transfer between the banks.

2.6 Acknowledgment Types

X12 includes two types of acknowledgment, the TA1 Interchange Acknowledgment and the 997 Functional Acknowledgment.

2.6.1. TA1, Interchange Acknowledgment

The TA1 acknowledgment verifies the interchange envelopes only. The TA1 is a single segment and is unique in the sense that this single segment is transmitted without the GS/GE envelope structures. A TA1 acknowledgment can be included in an interchange with other functional groups and transactions.

2.6.2. 997, Functional Acknowledgment

The 997 includes much more information than the TA1; see [Figure 2 on page 18](#). The 997 was designed to allow trading partners to establish a comprehensive control function as part of the business exchange process.

There is a one-to-one correspondence between a 997 and a functional group. Segments within the 997 identify whether the functional group was accepted or rejected. Data elements that are incorrect can also be identified.

Many EDI implementations have incorporated the acknowledgment process into all of their electronic communications. Typically, the 997 is used as a functional acknowledgment to a functional group that was transmitted previously.

The 997 is the acknowledgment transaction recommended by X12.

The acknowledgment of the receipt of a payment order is an important issue. Most corporate originators want to receive at least a Functional Acknowledgment (997) from the beneficiary of the payment. The 997 is created using the data about the identity and address of the originator found in the ISA and/or GS segments.

2.6.3. Application Acknowledgments

Application acknowledgments are responses sent from the destination system back to the originating system, acknowledging that the transaction has been successfully or unsuccessfully completed. The application advice (824) is a generic application acknowledgment that can be used in response to any X12 transaction. However, it has to be set up as a response transaction; only TA1 and 997 transactions are sent out automatically.

Other types of responses from the destination system to the originating system, which may also be considered application acknowledgments, are responses to query transactions—for example, the Eligibility Response (271) which is a response to the Eligibility Inquiry (270).

2.7 Key Parts of EDI Processing Logic

The five key parts of EDI processing logic are listed in Table 3.

Table 3 Key Parts of EDI Processing

Term	Description	Language Analogy	eGate Component
structures	format, segments, loops	syntax rules	OTD elements and fields
validations	data contents “edit” rules	semantic rules	validation methods
translations (also called mappings)	reformatting or conversion	translation	collaborations

Table 3 Key Parts of EDI Processing

Term	Description	Language Analogy	eGate Component
enveloping	header and trailer segments	envelope for a written letter	the special “envelope” OTDs: FunctionalGroupEnv and InterchangeEnv
acks	acknowledgments	return receipt	specific acknowledgment elements in the OTD

eGate uses the structures, validations, translations, enveloping, and acknowledgments listed below to support the X12 standard.

2.7.1. Structures

The X12 OTD Library includes pre-built OTDs for all supported X12 versions. These OTDs can be viewed in the OTD Editor, but cannot be modified.

To customize the OTD structure—for example, to add a segment or loop—you must first generate a SEF file (typically using a third-party tool, such as the EDISIM tool from Foresight Corporation). You then use the SEF OTD Wizard to generate the OTD.

2.7.2. Validations, Translations, Enveloping, Acknowledgments

Within each OTD are Java methods and Java bean nodes for handling validation; and the marshal and unmarshal methods of the two **envelope** OTDs handle enveloping and de-enveloping. No pre-built translations are supplied with the OTD libraries; these can be built in an eGate GUI called the Java Collaboration Editor (JCE).

Note: In eGate, X12 translations are called collaborations.

2.7.3. Trading Partner Agreements

There are three levels of information that guide the final format of a specific transaction. These three levels are:

- The X12 standard
The Accredited Standards Committee publishes a standard structure for each X12 transaction.
- Industry-specific Implementation Guides
Specific industries publish Implementation Guides customized for that industry. Normally, these are provided as recommendations only. However, in certain cases, it is extremely important to follow these guidelines. Specifically, since HIPAA regulations are law, it is important to follow the guidelines for these transactions closely.
- Trading Partner Agreements

It is normal for trading partners to have individual agreements that supplement the standard guides. The specific processing of the transactions in each trading partner's individual system might vary between sites. Because of this, additional documentation that provides information about the differences is helpful to the site's trading partners and simplifies implementation. For example, while a certain code might be valid in an implementation guide, a specific trading partner might not use that code in transactions. It would be important to include that information in a trading partner agreement.

2.8 Additional Information

For more information on the X12 standard, visit the following Web sites:

<http://www.disa.org> and specifically <http://www.x12.org/x12org/index.cfm>

X12 implementation guides can be obtained from Washington Publishing Company:

<http://www.wpc-edi.com>; specifically, <http://www.wpc-edi.com/tg4/tg4home.asp>

Note: This information is correct at the time of going to press; however, SeeBeyond has no control over these sites. If you find the links are no longer correct, use a search engine to search for X12.

2.9 About eXchange Integrator

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

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

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

At run time, all steps in the business process, from initial receipt of the message to final delivery to the trading partner, are tracked in real time and also stored in the eXchange database. The Web-based message/package tracker provides tools for retrieving and filtering tracked message and envelope information. Used in conjunction

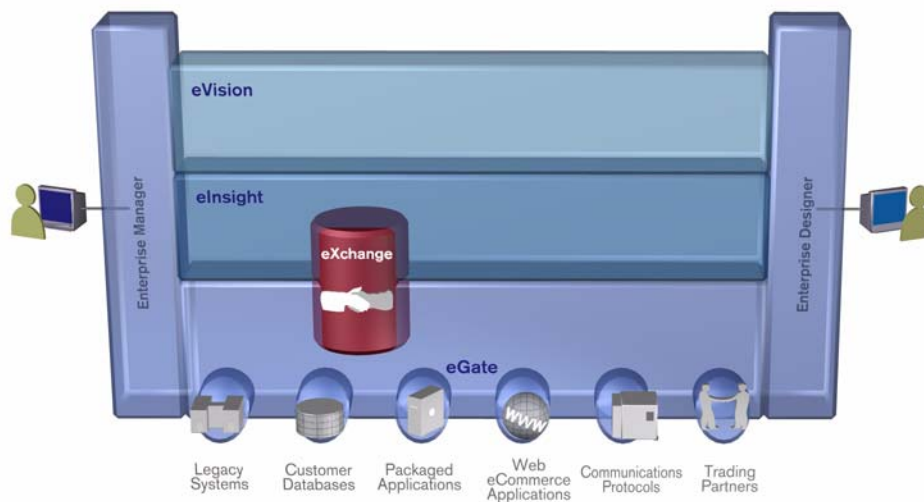
with the other monitoring tools of the ICAN suite, this provides the enterprise with a complete solution for troubleshooting and managing all eBusiness activities.

2.9.1. eXchange and the ICAN Suite

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

eXchange is tightly integrated with the ICAN Suite and runs as a component within the ICAN Suite environment. Figure 9 illustrates how eXchange and other ICAN Suite components work together.

Figure 9 eXchange and the ICAN Suite



2.9.2. eXchange Architectural Overview

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

eXchange uses the following key components:

- **B2B Host Designer** — Using the **Enterprise Designer** GUI framework, eXchange provides an editor for setting up B2B environments, called the *B2B Host Designer*. Each B2B Host provides one or more protocol-specific delivery channels that are exposed to the eXchange database via the Repository. Delivery channels provided by the B2B Host can then be accessed by specific trading partners and reused.

- **B2B Services and Protocols** — eXchange provides two other special editors: The *eXchange Service Designer*, for modeling the choreography of B2B interactions between the internal system and an external trading partner, as mediated by the B2B Host; and the *eXchange Protocol Designer*, for setting up the message flow logic (*B2B protocol processes*) required to address a specific business challenge, using such activity elements as branching activities, timers, and exception handling.

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

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

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

2.9.3. Process Overview

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

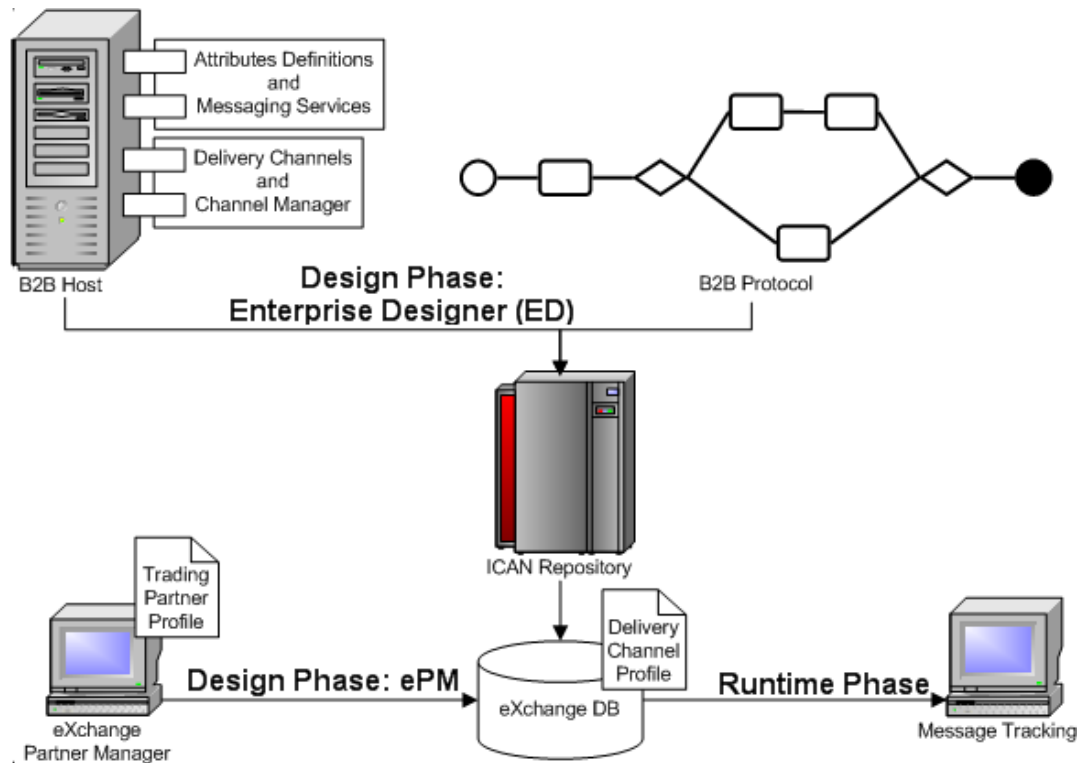
- Design phase within Enterprise Designer
- Design phase within eXchange Partner Manager
- Runtime phase

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

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

These phases are illustrated in Figure 10.

Figure 10 eXchange Architecture

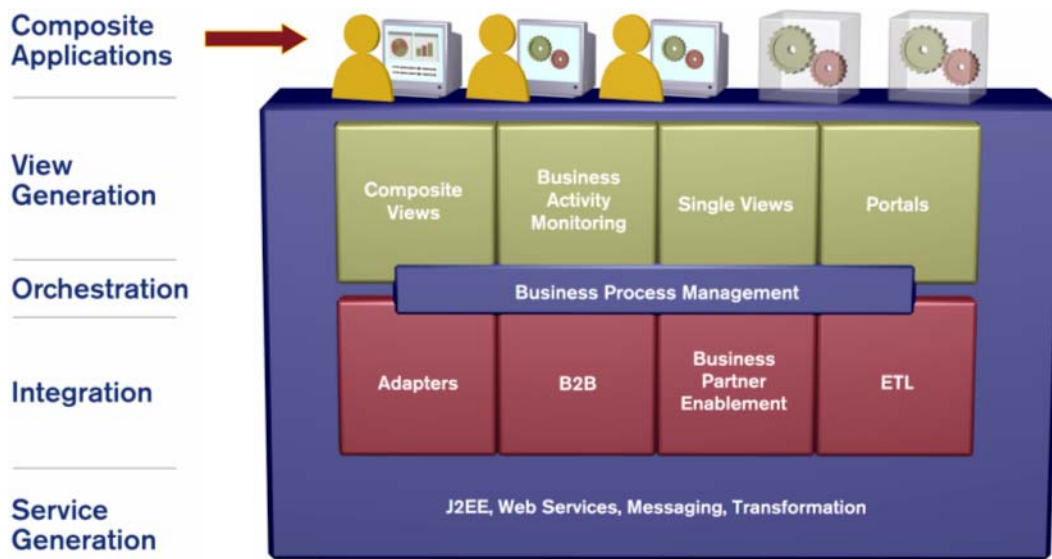


2.10 About B2B Protocol Manager Composite Applications

Composite Applications and the ICAN Suite

The ICAN Suite includes several kinds of *composite applications*—applications that are assembled from components provided by several different products, making use of the suite’s service-generation, integration, orchestration, and view-generation layers. See Figure 11.

Figure 11 Composite Applications and the ICAN Suite

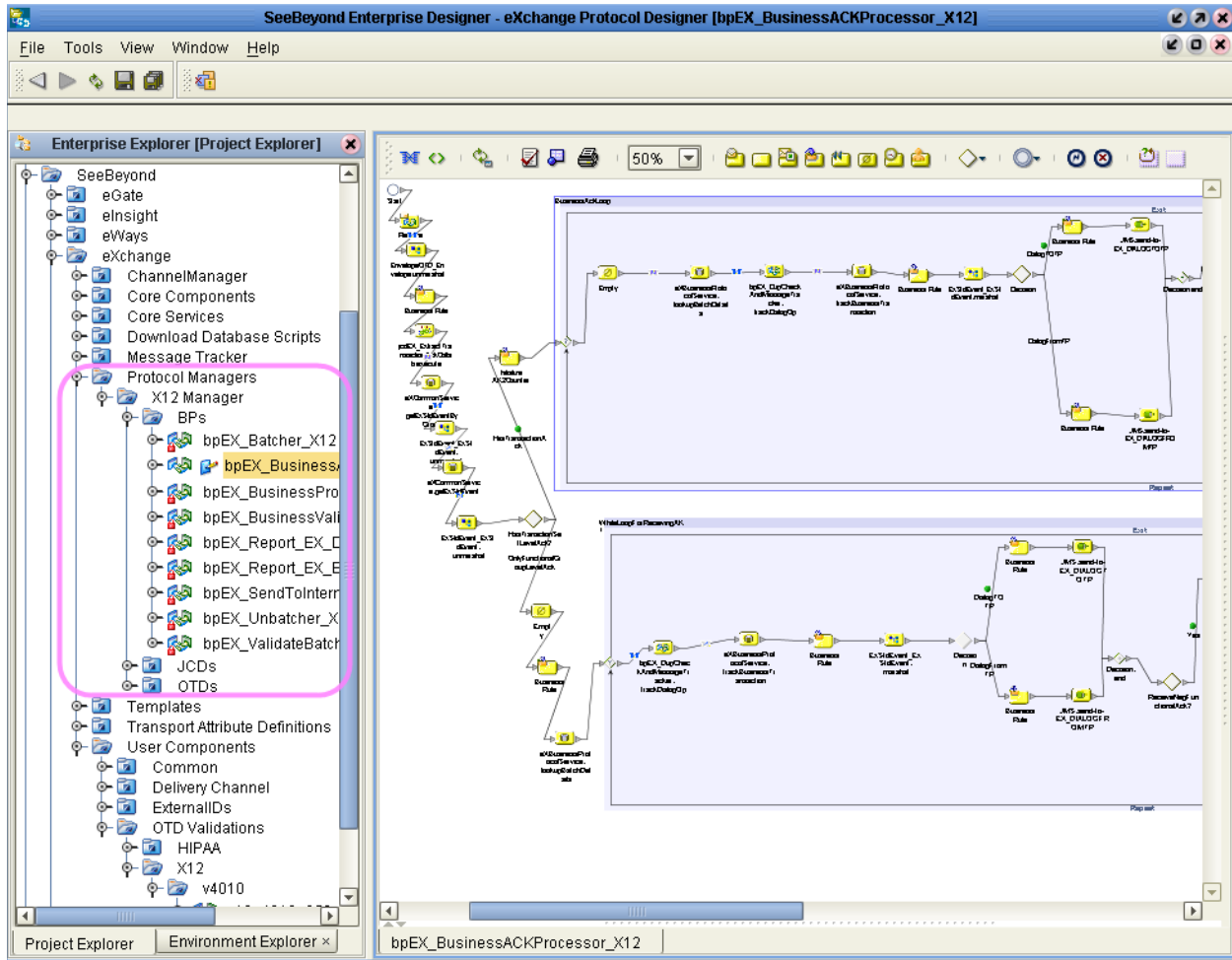


About Protocol Manager Composite Applications and eXchange

eXchange provides an open framework to support standard B2B business protocols and enveloping protocols. Protocol-specific composite applications are designed to dovetail with the eXchange framework while exposing all components they use (OTDs, BPs, validation rules, and so forth), allowing for further expansion or customization as needed, and achieving end-to-end visibility of all business rules and logic.

In Figure 12, the Project Explorer pane lists the SeeBeyond-supplied BPs for X12, and the canvas shows one of the X12 BPs. All business logic is exposed, and each action is customizable.

Figure 12 SeeBeyond-Supplied BP (for Processing Business Acknowledgments in X12)



Each protocol manager has the following characteristics:

- It includes a *messaging service* or *business service*—a choreographed sequence of events incorporating the rules set by the protocol specifications, such as:
 - ♦ Interchange and acknowledgment processing
 - ♦ Business message correlation
 - ♦ Enveloping / de-enveloping
 - ♦ Batching / splitting
 - ♦ Event archiving
- It uses information in the message itself and in the eXchange trading partner profile to prepare the message per the B2B protocol specifications.
- It works with eXchange common services to prepare, archive, and deliver the message. Common services include:
 - ♦ Error handling
 - ♦ Message compression / decompression

- ♦ Message tracking
- ♦ Trading partner profile database lookup
- ♦ PKI security services, such as encryption and signature creation/verification

Example: X12 Protocol Manager Composite Application

In the View layer: The eXchange Protocol Designer provides the ability to extend (customize) the SeeBeyond-supplied X12 protocol implementation, and the eXchange Message Tracker allows searching and viewing of X12 messages.

In the Services Orchestration layer: The eXchange Service Designer is used to choreograph a transaction set; then the X12 protocol implementation prepares and returns the interchange and functional acknowledgments (TA1 and 997; see [“Acknowledgment Types” on page 25](#)) to the trading partner, and also performs message correlation to associate the business response to the request (see [“Application Acknowledgments” on page 26](#)).

In the Integration Services layer: The X12 protocol implementation, in conjunction with the OTD library, validates ISA and GS envelopes from incoming messages (see [Figure 1 on page 18](#)), prepares ISA and GS envelopes for outgoing messages, batches together documents to be delivered as a single transaction (ISA), and records the activity in Message Tracking.

Installing X12 Manager

This chapter provides the prerequisites and steps for installing the ASC X12 Manager Composite Application.

What's in This Chapter

- **System Requirements** on page 35
- **Before You Install** on page 36
- **Installing the Product Files** on page 36
- **Database Scripts** on page 42

3.1 System Requirements

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

3.1.1. Supported Operating Systems

X12 Manager is available on the following operating systems:

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

3.1.2. Database Support

Database for eXchange Partner Management and Message Tracking

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

- Oracle 8.1.7
- Oracle 9.0.1
- Oracle 9.2

Database for Persistence and Monitoring via eInsight Engine

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

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

3.2 Before You Install

Before you begin installing X12 Manager, make sure of all the following:

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

3.3 Installing the Product Files

The steps for installing X12 Manager are the same as for other products in the ICAN Suite. You can find general product installation instructions in the *ICAN Suite Installation Guide*, which is available on the product media and can also be accessed via Enterprise Manager (Documentation tab).

3.3.1. Uploading X12 Manager to the Repository

Before you begin

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

To upload X12 Manager product files to the Repository

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

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

where

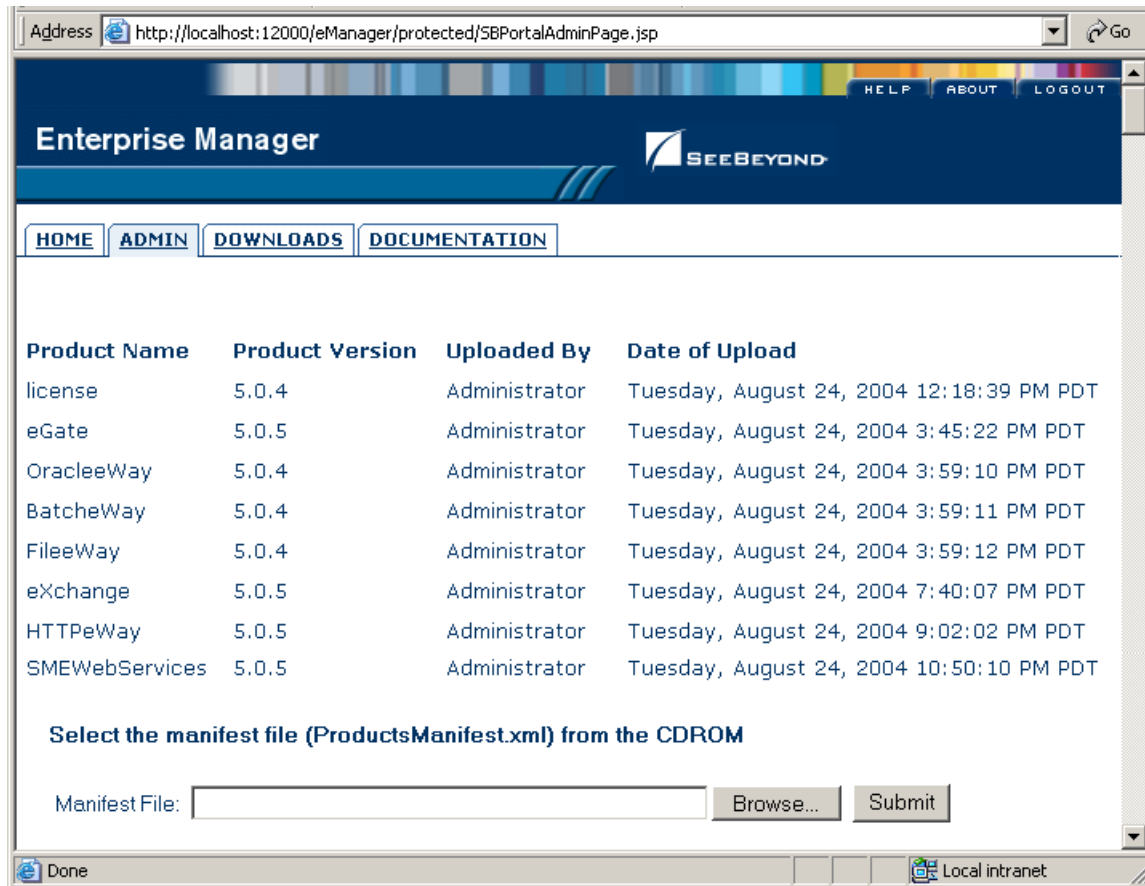
- ♦ *<hostname>* is the name of the machine running the Repository server.
- ♦ *<port>* is the starting port number assigned when the Repository was installed.

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

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

- 2 On the Enterprise Manager **SeeBeyond Customer Login** page, enter your username and password.
- 3 When Enterprise Manager responds, click the **ADMIN** tab. See Figure 13.

Figure 13 Enterprise Manager ADMIN Page: Uploading the ProductsManifest.xml File



- 4 In the ADMIN page, under “Select the manifest file [...]”, click **Browse**.
- 5 In the **Choose file** dialog, navigate to the correct directory for the products you want to upload (for example: **CoreProducts** for eXchange, or **Add-ons** for eWays), click the **ProductsManifest.xml** filename, and then click **Open**.
- 6 In the ADMIN page, click **Submit**.
The lower half of the ADMIN page lists the product files you are licensed to upload.
- 7 Stage all needed eWay .sar files now, and then click the **upload now** button.
- 8 If you still need to upload any .sar files (for example, if you have not yet uploaded an OTD library from a Products_CD3, ASCX12, or UNEDIFACT* folder): Submit the appropriate ProductsManifest.xml file, stage the .sar files, and click **upload now**. The sample uses X12 v4010, in ASCX12\ASC_X12_OTD_Lib_v4010.sar, and can also accommodate Products_CD3\HIPAA_2000_Addenda_OTD_Lib.sar if you are licensed to install the corresponding Validation_BP.
- 9 When you are finished uploading all prerequisites, return to the manifest file area, browse to the **CompositeApplications** folder, and submit its ProductsManifest.xml file. Then, in the Products column, find **X12 Manager** and click its **Browse** button.
- 10 In the **Choose file** dialog, click **X12_Manager.sar** and then click **Open**. Then, back in Enterprise Manager, click the **upload now** button.

- 11 For documentation and samples, also upload the corresponding [...]Docs.sar files, such as CompositeApplications\Documentation\X12_ManagerDocs.sar (and/or Add-ons\Documentation\HTTPeWayDocs.sar, and so forth).
- 12 For Validation BPs: Submit a new ProductsManifest.xml file (for example, from the ASCX12) and upload the [...]_Validation_BP.sar file(s) corresponding to the OTD Library .sar file(s) previously uploaded. The sample implementation scenario uses ASCX12\ASC_X12_OTD_Validation_BP_v4010.sar, and can also accommodate Products_CD3\HIPAA_2000_Addenda_OTD_Validation_BP.sar if you have a license to install it.

3.3.2. Refreshing Enterprise Designer

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

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

Before you begin

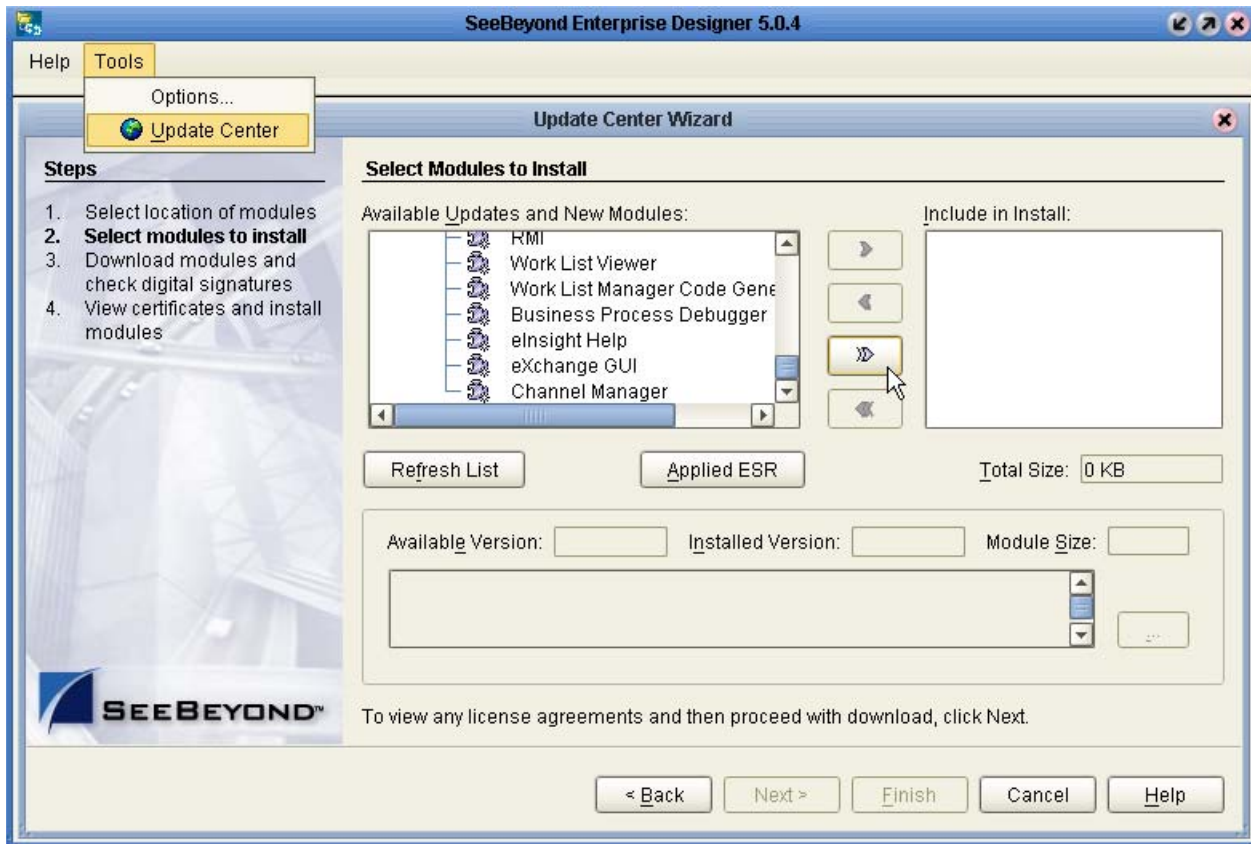
- You must have already downloaded and installed Enterprise Designer.
- A Repository server must be running on the machine where you uploaded the X12 Manager product files.

To refresh an existing installation of Enterprise Designer

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

The Update Center shows a list of components ready for updating. See Figure 14.

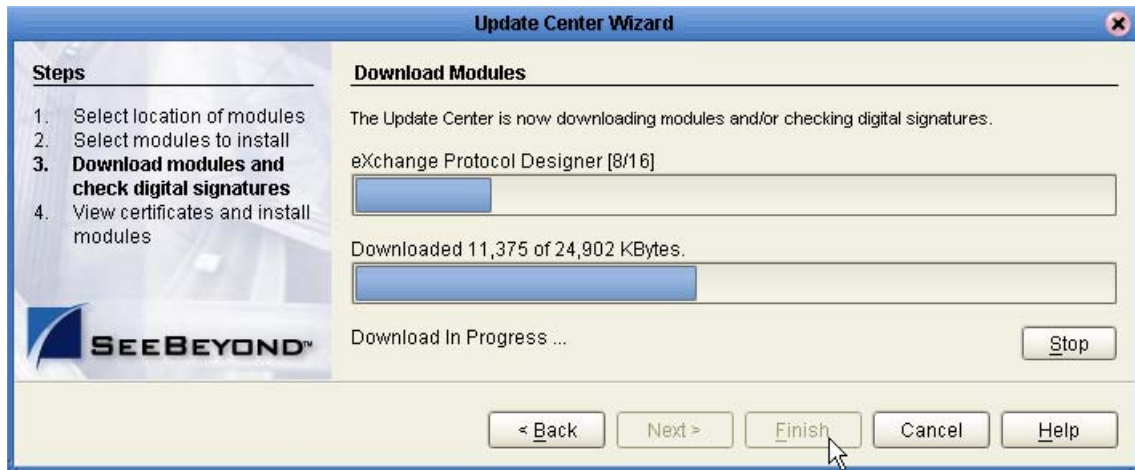
Figure 14 Update Center Wizard: Select Modules to Install



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

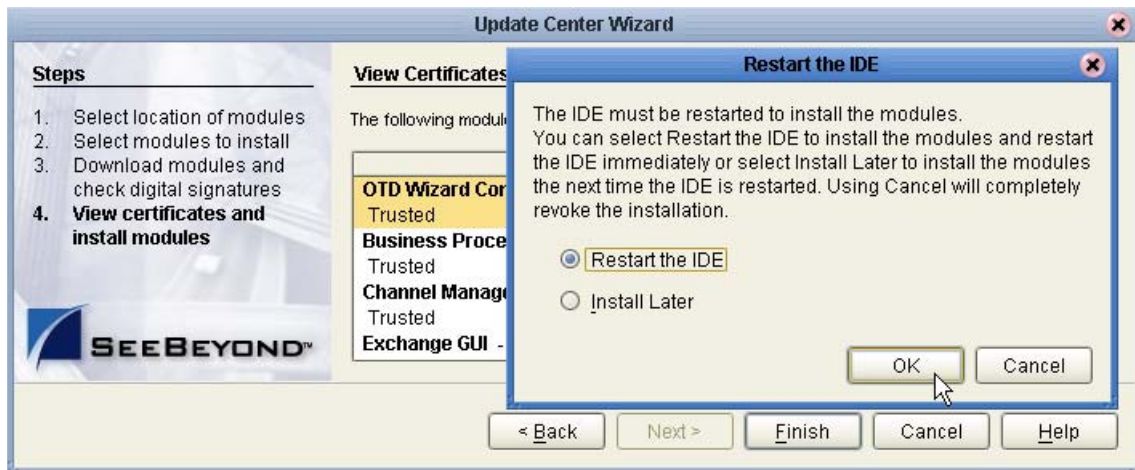
- 3 Click **Add All** (the button with a doubled chevron pointing to the right).
All modules move from the Available/New pane to the **Include in Install** pane.
- 4 Click **Next** and, in the next window, click **Accept** to accept the license agreement.
The wizard shows you the progress of the download. See Figure 15.

Figure 15 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 16.

Figure 16 Update Center Wizard: Restart Enterprise Designer



When Enterprise Designer restarts, the installation of ASC X12 Manager Composite Application is complete, and you can use all eXchange tools provided on the Enterprise Designer framework.

3.4 Database Scripts

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

For convenience, the as-shipped sample implementation allows you to run the sample on a single machine where two trading partners, based in Atlanta ("Company A") and Berlin ("Company B"), are simulated using two different database usernames and two different logicalhost port ranges. (Alternatively, and more realistically, you could create separate databases on separate machines.) Therefore, if you have not previously done so for another sample, create two extra username/password combinations in your eXchange 5.0.5 database. The sample assumes that in your Oracle database, you have already created two new username/password combinations: for Atlanta, the sample is preconfigured with username/password = ex_A/ex_A; for Berlin, the sample is preconfigured with username/password = ex_B/ex_B.

For details on running the **createuser.sql** script to create username/password combinations and running the **createdb.sql** script to populate tablespaces for each user, see the *eXchange User's Guide* section on "Running Database Scripts to Set Up the eXchange Database".

Using eXchange Partner Manager

This chapter lists and explains the X12-specific parameters configurable in eXchange Partner Manager (ePM).

What's in This Chapter

- [Overview of the ePM Parameter Hierarchy](#) on page 43
- [Parameters at the Trading Partner Level](#) on page 44
- [Parameters at the Trading Partner Profile \(TPP\) Level](#) on page 53
- [Parameters at the Business Service Level](#) on page 54
- [Parameters at the Business Action Level](#) on page 55

4.1 Overview of the ePM Parameter Hierarchy

A trading partner's configuration parameters can be set at three different levels:

- You can configure the properties and components of the *trading partner* (TP) itself.
- You can configure the properties of *trading partner profiles* (TPPs) within a TP.
- You can configure the properties and business actions for *services* defined for a TPP. (Business and messaging services are organized according to BAD and/or MAD.)

Figure 17 shows an environment (AtlantaEnv) with one TP (**Berlin**), whose first TPP (**profile_850_Out**) is fully expanded; under the BAD (X12 version 1.0) is its "leaf" terminus—a business service (**dlg_850_Out**) with four actions (850, 997, 855, 997).

Figure 17 ePM Parameter Hierarchy of TP > TP Profile > Service



4.2 Parameters at the Trading Partner Level

Configuration parameters for the Trading Partner itself are organized into two major tabs: **Properties** and **Components**.

4.2.1. Trading Partner > Properties

When a Trading Partner is selected, the **Properties** tab shows four subtabs:

- The **General** subtab allows you to rename the TP itself and, optionally, to provide a description consisting of any length of text.
- The **References** subtab allows you to provide one or more references for this TP. You supply the reference name and, optionally, its type; select a Party Reference type (currently only “simple” is supported); and optionally, supply a schema location for the reference.
- The **Unique IDs** subtab allows you to provide one or more unique IDs to associate with this TP. You supply the name constituting the unique ID and, optionally a text string for the ID’s type or value.
- The **Contacts** subtab allows you to provide one or more contacts to associate with this TP. If used, you must supply the contact’s e-mail, and first/last names, and you can optionally also supply address, telephone/fax, and other information.

Except for Trading Partner Name, all TP properties are optional.

4.2.2. Trading Partner > Components: Overview

When a Trading Partner is selected, clicking the **Components** tab shows four subtabs:

- The **Delivery Channels** subtab allows you to add, modify, and delete bindings for the TP’s external delivery channels (XDCs). The parameters constituting a complete binding for an XDC are discussed in [“Trading Partner > Components > Delivery Channels” on page 45](#).
- The **Certificates** subtab allows you to list certificates for the current environment and, for some protocols (such as AS2 or ebXML), to add, modify, and delete them.
- The **Internal Delivery Channels** subtab allows you to add, modify, and delete bindings for the TP’s internal delivery channels (IDCs). For the X12 protocol, IDCs are usually not used.
- The **Enveloping Channels** subtab allows you to add, modify, and delete bindings for the TP’s enveloping channels. The X12 protocol, unlike AS2 and ebXML, requires enveloping channels. The parameters constituting a complete binding for an enveloping channel are discussed in [“Trading Partner > Components > Enveloping Channels” on page 49](#).

4.2.3. Trading Partner > Components > Delivery Channels

The top of the **Delivery Channels** subtab lists all bindings currently defined for the TP's external delivery channels (XDCs). An XDC "binding" is an association of XDC metadata parameters to a particular set of values.

To add a new binding for a particular XDC

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Delivery Channels** subtab.
- 2 In the Trading Partner (right) pane, click the **New** button.
- 3 From the dropdown list of delivery channels defined for the current host, click the XDC for which you want to add a new binding.
- 4 Optionally, supply a name for the binding. (If you do not supply one, the binding name will be the same as the XDC name.)
- 5 Click the **Continue** button.

Result: In the list of bindings at the top of the pane, a row for the new binding is added to the end; below the list, five new subtabs appear, with **General** preselected. All five subtabs for the new binding have been populated with default parameters.

To delete an existing XDC binding

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Delivery Channels** subtab.
- 2 For the binding you want to delete, click the **Delete** link at the far right.
- 3 In response to the prompt, click the **Delete** button.

Result: The **Object Deleted** message confirms the deletion of the delivery channel.

To view or modify parameters for an existing XDC binding

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Delivery Channels** subtab.
- 2 In the list of bindings, click the name of the binding you want to modify.
Five subtabs appear below the list of bindings. The **General** tab is preselected.
- 3 Within each subtab, modify parameter settings as needed (see "[Parameters for Delivery Channels: "General" Tab" on page 46](#) through "[Parameters for Delivery Channels: "FromPartner Packaging" Tab" on page 48](#)), and then click **Save**.

Tip: In ePM, click the **Save** button any time you view or modify a list of parameters.

Result: This binding of XDC parameter values becomes available as a choice for all subsequent business actions to be configured for this Trading Partner.

Parameters for Delivery Channels: “General” Tab

See Table 4.

Table 4 Parameters for XDC Bindings: “General” Tab

Parameter Name	Default Value	Description / Notes
Delivery Channel	(read-only)	This tells you which XDC was chosen when the binding was initially created.
Binding Name	(dropdown list)	Initially, this shows you the name supplied when the binding was created. Do not use spaces in this name.
Description		
Delivery Channel Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies two standard delivery channel handler BPs: bpEX_DeliveryChannel_File, used in the sample, and bpEX_DeliveryChannel_FTP.)
Delivery Protocol Validation Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies two standard delivery channel handler BPs: bpEX_DeliveryChannel_File, used in the sample, and bpEX_DeliveryChannel_FTP.)
Delivery Message Syntax Validation Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list.
Custom Delivery Protocol Validation Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list.
Delivery Protocol Enveloper Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list.

Parameters for Delivery Channels: “ToPartner Transport” Tab

The names, types, and possible values for parameters in the **ToPartner Transport** tab depend on the XDC definition in the B2B host’s External Delivery Channels window: The transport attributes definition (TAD) that was specified as “To Partner Transport Attributes Definition” for this XDC specifies the attributes that appear in the “ToPartner Transport” tab for all bindings referencing the XDC.

Parameters for Delivery Channels: “FromPartner Transport” Tab

The names, types, and possible values for parameters in the **FromPartner Transport** tab depend on the XDC definition in the B2B host’s External Delivery Channels window: The TAD that was specified as “From Partner Transport Attributes Definition” for this XDC specifies the attributes that appear in the “FromPartner Transport” tab for all bindings referencing the XDC.

Parameters for Delivery Channels: “ToPartner Packaging” Tab

See Table 5. For a given XDC, some **ToPartner Packaging** values are inherited from the values specified in the B2B host, in the XDC’s Properties sheet, “**Packaging**” tab.

Table 5 Parameters for XDC Bindings: “ToPartner Packaging” Tab

Parameter Name	Default Value	Description / Notes
Encryption Key	[None]	(Requires SME. Not used in native X12.)
Check for Duplicates	true	Set this to either true or false ; cannot be [None].
Use Encryption	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Send Acknowledgments	true	Set this to either true or false ; cannot be [None].
Expect Acknowledgments	true	Set this to either true or false ; cannot be [None].
Character Set Encoding		
Sending warnings with ACKs	true	Set this to either true or false ; cannot be [None].
Use Signature	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Use Compression	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Message Encoding		
Batch Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Group Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Transaction Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Batcher Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard batcher handler BP for X12: bpEX_Batcher_X12, used in the sample.)
Batch Encryption Handler	[None]	(Not used in native X12.)
Batch Signing Handler	[None]	(Not used in native X12.)
Batch Compression Handler	[None]	(Not used in native X12.)
Batch Encryption Handler	[None]	(Not used in native X12.)
Batch Signing Handler	[None]	(Not used in native X12.)
Batch Compression Handler	[None]	(Not used in native X12.)

Parameters for Delivery Channels: “FromPartner Packaging” Tab

See Table 6. For a given XDC, some **FromPartner Packaging** values are inherited from the values specified in the B2B host, in the XDC’s Properties sheet, “**Unpackaging**” tab.

Table 6 Parameters for XDC Bindings: “FromPartner Packaging” Tab

Parameter Name	Default Value	Description / Notes
Signature Certificate	[None]	(Requires SME. Not used in native X12.)
Check for Duplicates	true	Set this to either true or false ; cannot be [None].
Use Encryption	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Send Acknowledgments	true	Set this to either true or false ; cannot be [None].
Expect Acknowledgments	false	Set this to either true or false ; cannot be [None].
Character Set Encoding		
Sending warnings with ACKs	true	Set this to either true or false ; cannot be [None].
Use Signature	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Use Compression	false	(Requires SME. Not used in native X12.) Set this to either true or false ; cannot be [None].
Message Encoding		
Batch Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Group Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Transaction Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Batch Splitter Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard splitter handler BP for X12: bpEX_Unbatcher_X12, used in the sample.)
Batch Decryption Handler	[None]	(Not used in native X12.)
Batch Signature Validation Handler	[None]	(Not used in native X12.)
Batch Decompression Handler	[None]	(Not used in native X12.)
Decryption Handler	[None]	(Not used in native X12.)
Signature Validation Handler	[None]	(Not used in native X12.)
Decompression Handler	[None]	(Not used in native X12.)

4.2.4. Trading Partner > Components > Enveloping Channels

The top of the **Enveloping Channels** subtab lists all bindings currently defined for the TP's enveloping protocols. A "binding" is an association of enveloping protocol metadata parameters to a particular set of values.

To delete an existing envelope binding

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Enveloping Channels** subtab.
- 2 For the binding you want to delete, click the **Delete** link at the far right.
- 3 In response to the prompt, click the **Delete** button.

Result: The **Object Deleted** message confirms the deletion of the delivery channel.

To add a new binding for a particular enveloping protocol

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Enveloping Channels** subtab.
- 2 In the Trading Partner (right) pane, click the **New** button.
- 3 From the dropdown list of enveloping attributes definitions (EADs) defined for the current host, click the EAD for which you want to add a new binding.
- 4 Optionally, supply a name for the binding. (If you do not supply one, the binding name will be the same as the enveloping protocol name.)
- 5 Click the **Continue** button.

Result: In the list of bindings at the top of the pane, a row for the new binding is added to the end; below the list, three new subtabs appear, with **General** preselected. All three subtabs for the new binding have been populated with default parameters; to modify them, see the following procedure.

To view or modify parameters for an existing envelope binding

- 1 If you have not already done so: In the Explorer (left) pane, select the trading partner; then, in the Trading Partner (right) pane, click the **Components** tab and the **Enveloping Channels** subtab.
- 2 In the list of bindings, click the name of the envelope binding you want to modify. Three subtabs appear below the list of bindings. The **General** tab is preselected.
- 3 Within each subtab, modify parameter settings as needed (see "[Parameters for Enveloping Channels: "General" Tab](#)" on page 50 through "[Parameters for Enveloping Bindings: "FromPartner DeEnvelope" Tab](#)" on page 52), and then click **Save**.

Tip: In ePM, click the **Save** button any time you view or modify a list of parameters.

Result: This binding of enveloping protocol parameter values becomes available as a choice for all subsequent business actions to be configured for this Trading Partner.

Parameters for Enveloping Channels: “General” Tab

The **General** subtab allows you to rename the binding itself and, optionally, to provide a description consisting of any length of text.

Parameters for Enveloping Bindings: “ToPartner Envelope” Subtab

See Table 7.

Table 7 Parameters for Enveloping Channels: “ToPartner Envelope Binding” Subtab

Parameter Name	Default Value	Description / Notes
Release Quantity	1	Nonnegative integer. Required. Specifies the threshold beyond which a batch send is triggered. Set Release Quantity to a very high value (99999999) if you want to send messages on a schedule, using Release Scheduler String.
Release Scheduler String		A expression specifying when and how often to run. The expression uses cron syntax and consists of six (or optionally seven) arguments, separated by spaces, to specify: second , minute , hour , day-of-month , month , day-of-week (and optionally year). See “About “cron” expressions” on page 51 . When a trigger time occurs, a batch is sent even if its count has not reached Release Quantity.
Time-Out (Minutes)	60	Nonnegative integer. Required. Specifies maximum time to wait for a reply before attempting a re-send.
Max Retry Count	1	Nonnegative integer. Required. Specifies maximum number of times to retry sending before giving up.
Batcher Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard batcher handler BP for X12: bpEX_Batcher_X12, used in the sample.)
Encryption Handler	[None]	(Not used in native X12.)
Signing Handler	[None]	(Not used in native X12.)
Compression Handler	[None]	(Not used in native X12.)
ISA01 Author Info Qual	00	
ISA02 Author Information		
ISA03 Sec Info Qual	00	
ISA04 Security Information		
ISA05 IC Sender ID Qual	01	
ISA06 Interchange Sender ID		
ISA07 IC Rcvr ID Qual	01	

Table 7 Parameters for Enveloping Channels: “ToPartner Envelope Binding” Subtab

Parameter Name	Default Value	Description / Notes
ISA08 Interchange Rcvr ID		
ISA11 IC Control Standard Identifier	U	
ISA12 IC Version Number	00401	
ISA13 IC Control Number	0	
ISA14 Acknowledgment Requested	1	
ISA15 Usage Indicator	T	
ISA16 Comp Elem Sep	:	If you use <i>nondefault</i> delimiters (for example, if you use “!” for segment terminator in v4060), you must ensure that your business rules manually pass the nondefault delimiters into the ExStdEvent/PayloadSection/Envelopes/BusinessProtocol/ location: In other words, pass the ISA into .../Batch/Header , the IEA into .../Batch/Trailer , the GS into .../Group/Header , and the GE into .../Group/Trailer .
Segment Terminator	~	
Element Separator	*	
GS06 Group Control Num	5	

About “cron” expressions

In Table 7, parameter Release Scheduler uses the “cron” syntax to specify a schedule. A cron expression specifies when and how often to run. It consists of six (or optionally seven) arguments, separated by spaces:

- Arg#1: **Second**. An integer in the range 0-59.
- Arg#2: **Minute**. An integer in the range 0-59.
- Arg#3: **Hour**. An integer in the range 0-23.
- Arg#4: **Day-of-month**. An integer in the range 1-31.
- Arg#5: **Month**. Either an integer in the range 1-12, or a case-insensitive three-character string, from: {Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec}.
- Arg#6: **Day-of-week**. Either an integer in the range 1-7, or a case-insensitive three-character string, from: {Sun, Mon, Tue, Wed, Thu, Fri, Sat}.
- Arg#7 (optional): **Year**. Either empty, or an integer in the range 1970-2099.

The following special characters are also allowed:

- * (asterisk): Can be used in any of the seven fields to signify “all values”. For example, “*” as the fourth argument means “every day.”
- , (comma): Can be used in any of the seven fields to separate items in a discrete list. For example, “MON,WED,FRI” in the sixth field means “on Monday, Wednesday, and Friday”.

- - (hyphen): Can be used in any of the seven fields to indicate minimum-maximum of a range. For example, “Nov-Feb” in the fifth field means “in November, December, January, and February”.
- / (slash): Can be used in any of the seven fields to specify increments. For example, “7/20” in the first field means “on seconds 7, 27, and 47”.
- ? (query): Can be used in field 4 or field 6 (but no other field, and not simultaneously) to avoid collision or ambiguity.
- L can be used in field 4 or field 6 or both (but not in combination with lists or ranges) to specify “the last”. For example, “L” in the fourth field means “the last day of the month” and “1L” in the sixth field means “the last Sunday of the month”.

Examples of valid cron expressions

- “0 0 12 * * ?” means “At noon every day”.
- “0 30 21 ? * *” means “At 9:15pm every day”.
- “0 0 2 ? * Mon-Fri” means “Weekdays at 2:00am”.
- “0 10/30 1,22 L * ?” means “At 1:10 and 1:40am and 10:10 and 10:40pm the last day of every month”.

Parameters for Enveloping Bindings: “FromPartner DeEnvelope” Tab

See Table 8.

Table 8 Parameters for Enveloping Channels: “FromPartner DeEnvelope Binding” Tab

Parameter Name	Default Value	Description / Notes
Batch Splitter Handler		
Decryption Handler	[None]	
Signature Validation Handler	[None]	
Decompression Handler	[None]	
ISA01 Author Info Qual	00	
ISA02 Author Information		
ISA03 Sec Info Qual	00	
ISA04 Security Information		
ISA05 IC Sender ID Qual	01	
ISA06 Interchange Sender ID		
ISA07 IC Rcvr ID Qual	01	
ISA08 Interchange Rcvr ID		

Table 8 Parameters for Enveloping Channels: “FromPartner DeEnvelope Binding” Tab

Parameter Name	Default Value	Description / Notes
ISA11 IC Control Standard Identifier	U	
ISA12 IC Version Number	00401	
ISA14 Acknowledgment Requested	1	
ISA15 Usage Indicator	T	

4.3 Parameters at the Trading Partner Profile (TPP) Level

Configuration parameters for each TP Profile are all contained in the **Properties** tab. See Figure 18 and Table 9.

Figure 18 Trading Partner Profile: “Properties” Tab

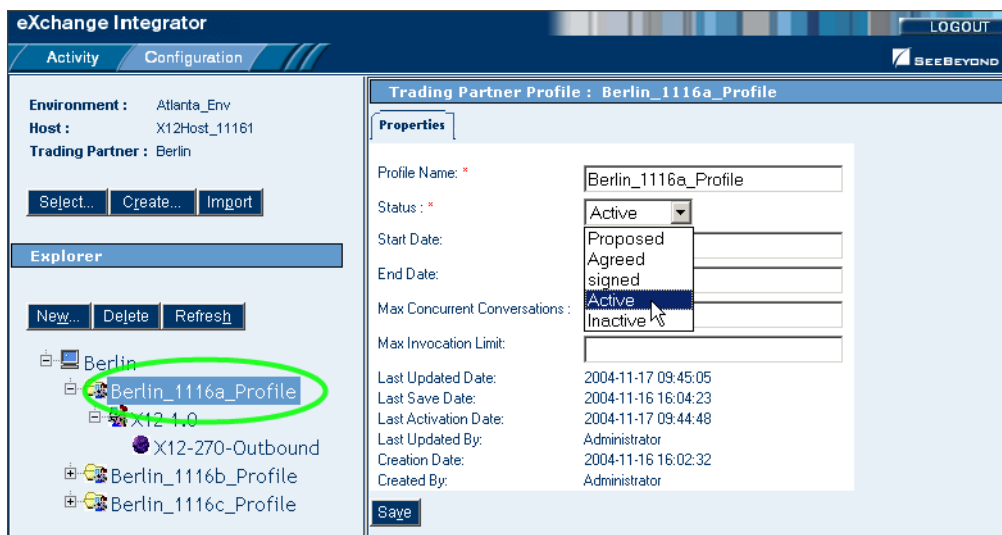


Table 9 Parameters in the (TP->TPP->) “Properties” Tab

Parameter Name	Default Value	Description / Notes
Profile Name		Use this field to rename a TP profile.
Status	Active	(Applies mainly to ebXML.)
Start Date	(no preset value)	(Applies to ebXML.)
End Date	(no preset value)	(Applies to ebXML.)
Max Concurrent Conversations	(no preset value)	Use this field to specify an upper limit to the number of simultaneous business services being processed.
Max Invocation Limit	(no preset value)	(Applies to ebXML.)

4.4 Parameters at the Business Service Level

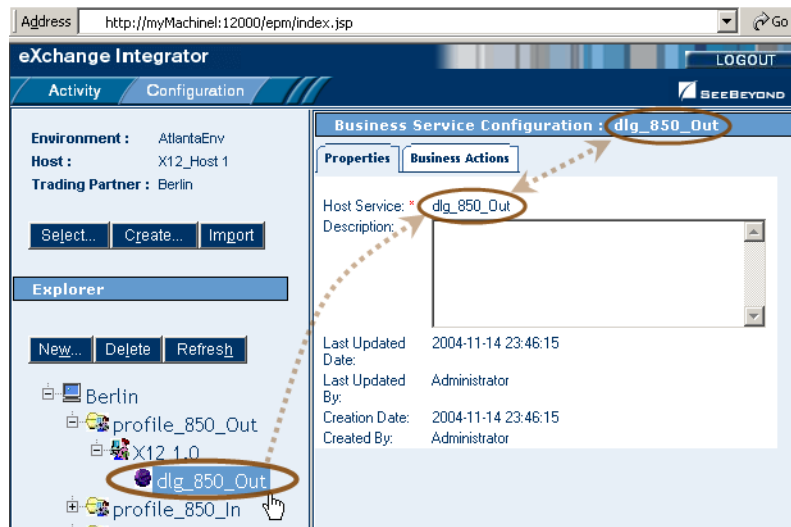
Configuration parameters for a business service (or messaging service) are organized into two major tabs: **Properties** and **Components**.

4.4.1. Business Service > Properties

In the Explorer (left) pane, when you expand the TP > TPP > MAD tree and click a business service (see Figure 19), the right pane displays the Business Service Configuration for the selected service, with the **Properties** tab initially preselected.

In the **Properties** tab, the “Host Service” field shows you the name of the business service you selected (see Figure 19), and the “Description” field allows you to optionally provide a description consisting of any length of text.

Figure 19 Accessing the Business Service Configuration Window



4.4.2. Business Service > Business Actions

In the Business Service Configuration window, click the Business Actions tab to display a list of all business actions defined for the current business service. You can expand one or more of the business actions to view or modify its parameters; see Figure 20 and Table 10.

4.5 Parameters at the Business Action Level

Figure 20 Business Service Configuration: “Business Actions” Tab

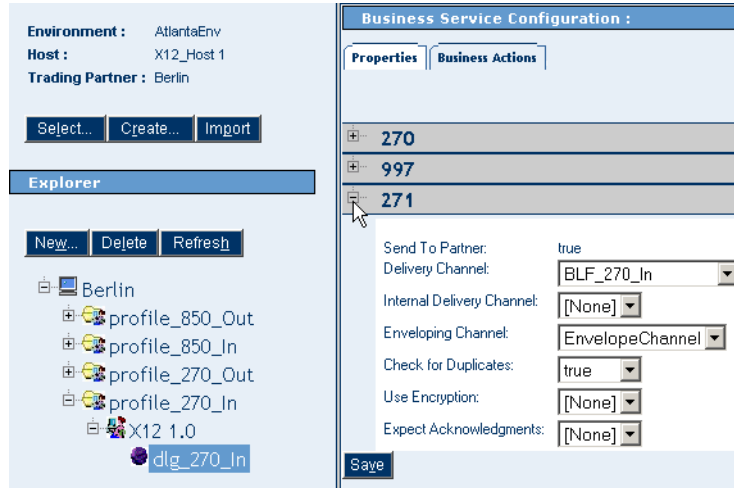


Table 10 Parameters in the (TP->TPP-> Business Service) “Business Actions” Tab

Parameter Name	Default Value	Description / Notes
Send To Partner	(Read only; value depends on action type)	true indicates an <i>outbound</i> (Send ToPartner) action. false indicates an <i>inbound</i> (Receive FromPartner) action.
Delivery Channel	(dropdown)	Choose from a dropdown list of XDC bindings that have been defined for this TP. (See “Trading Partner > Components > Delivery Channels” on page 45.)
Internal Delivery Channel	[None]	If your host uses IDCs, choose from a dropdown list of IDC bindings that have been defined for this TP.
Enveloping Channel	(dropdown)	Choose from a dropdown list of XDC bindings that have been defined for this TP. (See “Trading Partner > Components > Enveloping Channels” on page 49.)
Use Encryption	[None]	(Not used in native X12.)
Expect Acknowledgments	[None]	
Character Set Encoding		(Leave blank for default character set encoding.)
Send warnings with ACKs	[None]	
Use Signature	[None]	(Not used in native X12.)
Use Compression	[None]	(Not used in native X12.)
Batch Tracking	Both	Retain the default setting to assure tracking of outbound and inbound batches.

Table 10 Parameters in the (TP->TPP-> Business Service) “Business Actions” Tab (Continued)

Parameter Name	Default Value	Description / Notes
Group Tracking	Both	Retain the default setting to assure tracking of outbound and inbound groups.
Transaction Tracking	Both	Retain the default setting to assure tracking of outbound and inbound transactions.
Time-Out (Minutes)	10	Set the maximum amount of time to wait for a transaction to complete.
Business Protocol Validation Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard handler BP for X12: bpEX_BusinessValidation_X12, used in the sample.)
Business Message Syntax Validation Handler	[None]	If your B2B host has business message syntax validation handler BPs (as the sample implementation scenario does), choose one from the dropdown list.
Business Transaction Type Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard handler BP for X12: bpEX_BusinessProtocolTransactionType_X12, used in the sample.)
Custom Business Protocol Validation Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list.
Business ACK Generator Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list.
Business ACK Processor Handler	[None]	If your B2B host has a handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard handler BP for X12: bpEX_BusinessACKProcessor_X12.)
Encryption Handler	[None]	(Not used in native X12.)
Signing Handler	[None]	(Not used in native X12.)
Compression Handler	[None]	(Not used in native X12.)
Error Handler	[None]	If your B2B host uses error handler BPs, choose one from the dropdown list. (SeeBeyond supplies a standard error handler BP for JMS: bpEX_JMSErrorHandler, used in the sample.)
Custom External Unique ID Handler	[None]	If your B2B host has a custom handler BP of this type, choose it from the dropdown list. (SeeBeyond supplies a standard handler BP for X12: bpEX_ExternalUniqueID_X12_867).

Table 10 Parameters in the (TP->TPP-> Business Service) “Business Actions” Tab (Continued)

Parameter Name	Default Value	Description / Notes
GS01 Functional ID	(depends on business action)	This must match the value of the “Group Name” attribute of this business action, set in the B2B host’s business service. For example, a Group Name of “HS” is for a 270 action; “HB” for a 271; “PO” for an 850; “PR” for an 855; “FA” for a 997.
GS02 Application Sender Code		
GS03 Application Rcvr Code		
GS04 Date Format	CCYYMMDD	Choose from the dropdown whether to use four-digit or two-digit year format. Examples: <ul style="list-style-type: none"> ▪ 20041201 = December 1, 2004 ▪ 050112 = January 12, 2004
GS05 Time Format	HHMM	Choose from the dropdown whether to specify seconds and degree of accuracy. Examples: <ul style="list-style-type: none"> ▪ 2359 = 11:59PM ▪ 235959 = 11:59:59PM ▪ 23595999 = 11:59:59.99PM
GS07 Resp Agency Code	X	
GS08 Vers/Rel/Indust ID Code	004010X092	
Starting Control Number	0	

Customizing a Validation Handler BP

It would be beyond the scope of this book to discuss all the ways in which you can customize the SeeBeyond-supplied BPs to tailor them to your needs. This chapter supplies a very limited but practical exercise in making a small change to a handler BP.

Because the eXchange open framework exposes all business logic used along the way in every B2B protocol, with a bit of practice you should be able to analyze the functioning of other SeeBeyond-supplied BPs function, and to see how you can tailor them to meet your needs.

For more in-depth information on the activity flow, fault-handling, and variable usage in syntax-validation handler BPs, see [Appendix A](#).

Overview of the Exercise

You will open an X12 syntax-validation BP and locate two activities that involve a correlation key in a Request/Reply pair of transactions. For each of those activities, you will locate the OTD node that currently supplies data to the correlation key, and change it to a different OTD node.

Before You Begin

It is assumed you have already installed eXchange and the X12 Protocol Manager, as well as at least one X12 OTD library and a validation BP corresponding to it. This exercise uses version 4021; if you use a different version, adjust accordingly.

To customize a validation handler BP so as to use a specific correlation key


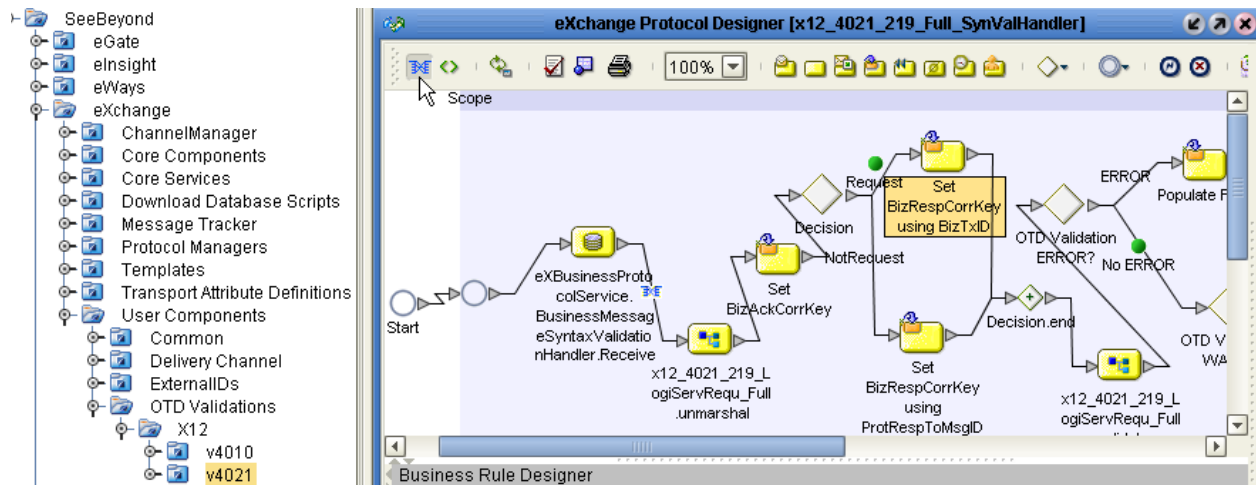
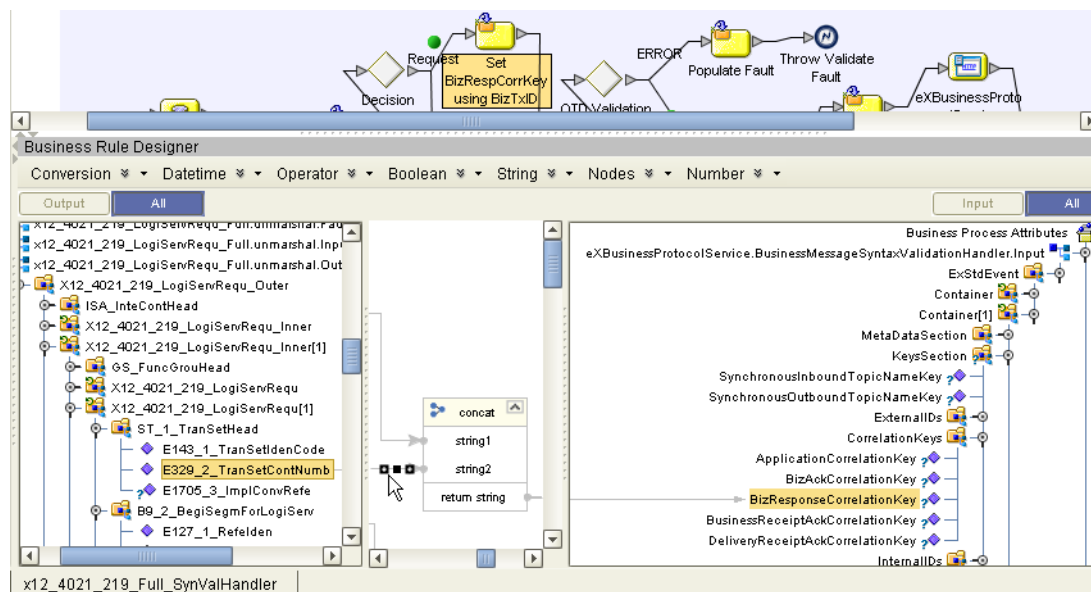
- 1 Start a session of Enterprise Designer, log in, and maximize the window.
- 2 In the project tree, open SeeBeyond > eXchange > User Components > OTD Validations > X12 > **v4021** to display the BPs it contains.
- 3 Check out the following BP and open it: **x12_4021_219_Full_SynValHandler**
- 4 On the canvas toolbar, click  to open the Business Rule Designer (BRD). Then, on the canvas, click the following business rule: **Set BizRespCorrKey using BizTxID**
See Figure 21.

Figure 21 Opening the X12 v4021 219 Full Syntax Validation Handler



Expand the BRD pane and trace the mappings without changing them. See Figure 22.

Figure 22 Mappings for the Request/Response Correlation Key



- ◆ The right side of the mapping receives the end result of a multiple concatenation into a node named **BizResponseCorrelationKey**, which is located under:
 eXBusinessProtocolService.BusinessMessageSyntaxValidationHandler.Input
 ExStdEvent
 Container[1]
 KeysSection
 CorrelationKeys
- ◆ Several nodes on the left side go into this mapping, but you are only interested in the one that feeds into the final (rightmost) **concat** operation: Its name is **E329_2_TransSetContNumb**, and it is the second node located under:

```
X12_4021_219_LogiServRequ_Outer
X12_4021_219_LogiServRequ[1]
  X12_4021_219_Logi_Serv_Requ_Inner[1]
    X12_4021_219_LogiServRequ[1]
      ST_1_1_TransSetHead
```

In Figure 22, the highlighted mapping is the one you will replace.

- 5 Delete the line connecting **E329_2_TransSetContNumb** to **string2** of the concat.
- 6 Replace it with a mapping from a node that is more suitable as a business correlation key, such as **B9A_3_ServRequ/E1644_1_ServRequCode**.
- 7 Validate the BP and save your changes.
- 8 Perform analogous steps to the business rule for the “NotRequest” path in the BP:
Set BizRespCorrKey using ProtRespToMsgID
- 9 Validate the BP and save your changes.

Result: Instead of using the value passed by **E329_2_TransSetContNumb** as a business correlation key, this BP will now use the value passed by the node you chose, such as **E1644_1_ServRequCode**.

Implementation Scenario

This chapter provides a sample scenario showing how eXchange can be used to achieve B2B solutions using the X12 protocol.

What's in This Chapter

The steps for the sample implementation occur in these phases:

Initial Setup Steps	In these steps, you ensure that prerequisites are met, obtain the necessary sample materials, extract sample files (using Enterprise Manager), and import the sample projects (using Enterprise Designer). See section 6.1 starting on page 62 .
Design Steps in Enterprise Designer	In these steps, you use Enterprise Designer to add and configure externals, view components in the B2B host project, and activate the host, creating an eXchange Service. Then you view the components in the feeder and protocol projects, create deployment profiles, and activate them. See section 6.2 starting on page 64 .
Design Steps in ePM	In these steps, you use the eXchange Partner Manager (ePM) facility to create a trading partner for one environment ("A"), view and specify parameter values for it, and activate it; then you do the same for the other ("B") environment. See section 6.3 starting on page 80 .
Runtime Steps	In these steps, you bootstrap two logical hosts, apply the activated deployments, stage sample input data, and use the Message Tracking facility to follow the progress of messages, as well as acknowledgments and errors, tracking messages as they are transported from one end point, received and handled by inbound and outbound B2B protocol processes, sent out, and transported to another end point. See section 6.4 starting on page 99 .

Overview of the Sample Implementation

The sample implementation demonstrates using X12 protocol manager for outbound and inbound message processing: X12 850/855 (and optionally HIPAA 270/271 as well; requires a license for the HIPAA Addenda Validation_BP), with 997 acknowledgments, between two parties: "Atlanta" (company "A") and "Berlin" (company "B").

6.1 Initial Setup Steps

In this section

- [Installing the Sample Files for X12 on page 62](#)
- [Importing the Sample Projects on page 63](#)

6.1.1 Installing the Sample Files for X12

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

To install the sample files

Before you begin: Your repository must already be running, and you must be logged in to Enterprise Manager. If you have already uploaded the documentation for X12 Manager, you can skip steps 1 and 2 and start with step 3.

- 1 In the ADMIN tab, if you have not already done so, browse to the [...] \Documentation \ **ProductsManifest.xml** file and submit it.
- 2 In the ADMIN tab, if you have not previously done so, browse to the **X12_ManagerDocs.sar** file, select it, and click the **upload now** button.
- 3 In the DOCUMENTATION tab, in the lower left pane, under **Composite Applications**, click **X12 Manager**.
- 4 In the pane that appears on the lower right, click **Download Sample**.
- 5 In the File Download dialog box, click **Open**.
- 6 Preserving file paths, extract the files to your **C:\temp\exChange** directory.

Result: The following directories and files are created:

```
C:\temp\exChange\Sample\Common\Projects\  
  eX_Common_Environments.zip  
  eX_Common_Projects.zip  
  eX_Util_EX_FROMINTERNAL_to_File.zip  
  eX_Util_EX_TOINTERNAL_to_File.zip  
  eX_Util_from-File_to-EX_FROMINTERNAL.zip  
  eX_Util_ExStdEvent_Get_Payload.zip  
  eX_Util_4020_EX_FROMINTERNAL.zip  
  
C:\temp\exChange\Sample\Common\Source\  
  InputParameters.xsd  
  FeederParameters.xsd  
  
C:\temp\exChange\Sample\X12\Projects\  
  X12_Host.zip  
  4010_850_855_feeders.zip  
  HIPAA_270_271_feeders.zip  
  
C:\temp\exChange\Sample\X12\TradingPartners\  
  X12_TP_for-AtlantaEnv_to-from_Berlin.xml  
  X12_TP_for-BerlinEnv_to-from_Atlanta.xml  
  X12_TP_for-AtlantaEnv_to-from_Berlin_270_850.xml
```

```
X12_TP_for-BerlinEnv_to-from_Atlanta_270_850.xml
```

```
C:\temp\exChange\Sample\X12\Data\Atlanta
X12_4010_850_incremented.st
X12_4010_855_template.st
X12_dlg_270_In_profile_270_In.xml
X12_dlg_270_Out_feeder_Berlin.~in
X12_dlg_850_In_profile_850_In.xml
X12_dlg_850_Out_feeder_Berlin.~in
X12_HIPAA_270_incremented.st
X12_HIPAA_271_template.st
```

```
C:\temp\exChange\Sample\X12\Data\Berlin
X12_4010_850_incremented.st
X12_4010_855_template.st
X12_dlg_270_In_profile_270_In.xml
X12_dlg_270_Out_feeder_Atlanta.~in
X12_dlg_850_In_profile_850_In.xml
X12_dlg_850_Out_feeder_Atlanta.~in
X12_HIPAA_270_incremented.st
X12_HIPAA_271_template.st
```

Note: If you extract the data files to a different location, modifications will be required, in Enterprise Designer (the value of the “Data Root Directory” project variables) and in eXchange Partner Manager (fields in the ToPartner and FromPartner tabs).

6.1.2 Importing the Sample Projects

To import the sample projects

Before you begin: Your repository must already be running, and you must be logged in to Enterprise Designer. If your repository already has a project at root level whose name is identical to any of the projects you will be importing, you must delete or rename it before you start; and if you import the environments (rather than re-creating them), ensure the repository does not already contain any environment of the same name.

- 1 In Project Explorer, right-click the repository and, on the popup menu, click: **Import**
- 2 In the **Import Manager** dialog, browse to the folder where you installed the sample files (such as C:\temp\exChange\Sample) and do the following.
 - ♦ (required) One by one, open and import these required projects:
 - ♦ [...]Common\Projects\ex_Common_Projects.zip
 - ♦ [...]Common\Projects\ex_Common_Environments.zip
 - ♦ [...]X12\Projects\X12_Host.zip
 - ♦ [...]X12\Projects\4010_850_855_feeders.zip (for 850-855 processing)
 - ♦ (optional) As needed, open and import zero or more of the following projects:
 - ♦ [...]Common\Projects\ex_Util_*.zip (optional; useful for troubleshooting)
 - ♦ [...]X12\Projects\HIPAA_270_271_feeders.zip (for 270-271 processing)
- 3 Close the Import Manager dialog.

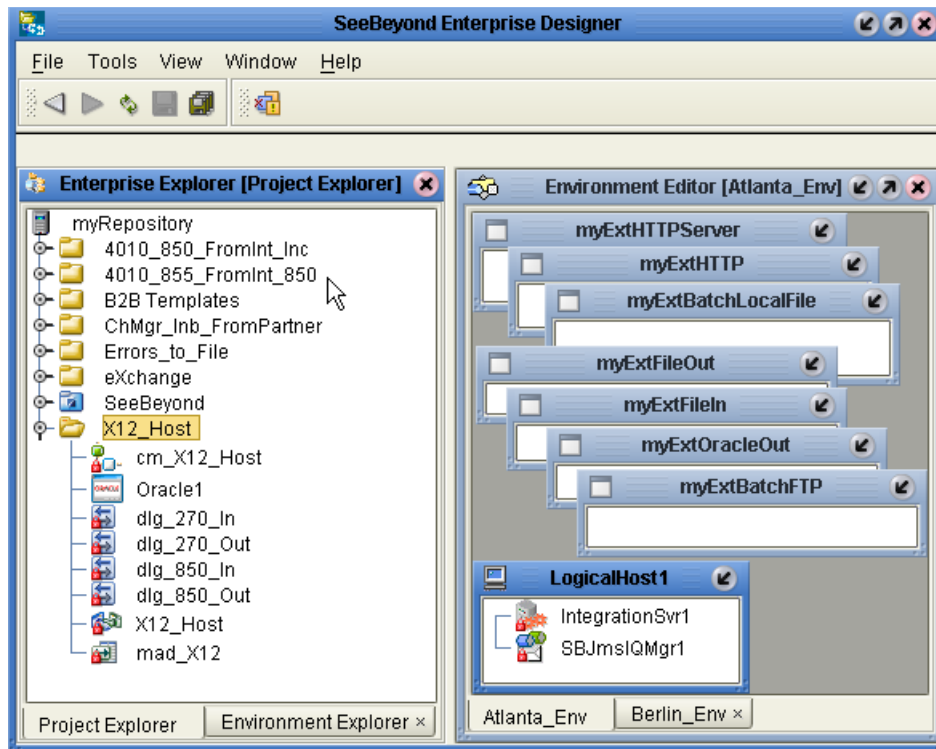
Result: The Project Explorer tree displays the following new projects:

- 4010_855_FromInt_Inc (from 4010_850_855_Feeders.zip)
- 4010_855_FromInt_850 (from 4010_850_855_Feeders.zip)

- **ChMgr_Inb_FromPartner** (from eX_Common_Projects.zip)
- **Errors_to_File** (from eX_Common_Projects.zip)
- **HIPAA_270_FromInt_Inc** (if you imported HIPAA_270_271_Feeders.zip)
- **HIPAA_271_FromInt_270** (if you imported HIPAA_270_271_Feeders.zip)
- **X12_Host** (from X12_Host.zip)

Figure 23 shows the results of importing the projects and environments required for the sample (that is, not including the utility projects or HIPAA 270/271 projects).

Figure 23 Project Explorer Tree, After Importing Sample Projects for X12



6.2 Design Steps in Enterprise Designer

For the X12 sample implementation, design-time steps in Enterprise Designer consist of the following:

- [Setting Up the Sample Environments on page 65](#)
- [Viewing and Activating the B2B Host Project on page 67](#)
- [Creating and Activating the Project Deployment Profiles on page 71](#)

6.2.1 Setting Up the Sample Environments

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

- **Oracle:** You must create a new outbound Oracle external and configure it, even if you imported the sample environment; sample parameters are for reference only.
- **Ports:** To run anything other than a SeeBeyond Integration Server on ports 18000–18009, make adjustments in steps 3 (ports) and/or 5 (type of Integration Server).
- **HTTPS:** To configure your HTTP server or client to use SSL, see the *HTTP(S) eWay Intelligent Adapter User's Guide* for eWay settings and Integration Server Web Server configuration.

If you imported the sample environments, you can skip past the following procedure (unless you need to adjust base ports and/or configure the HTTP externals to use HTTPS) and continue with the Oracle configuration steps [on page 66](#).

To create the sample environment

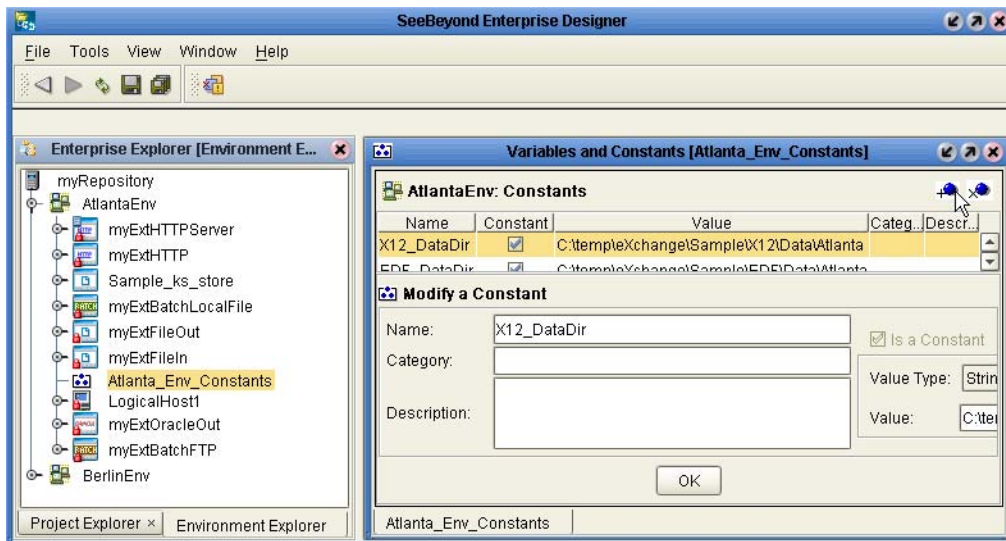
- 1 In Enterprise Designer, near the lower left of the window, click the **Environment Explorer** tab.
- 2 In the Environment Explorer tree, right-click the repository and, on the popup context menu, click **New Environment**
 - ♦ Rename the newly created environment to **AtlantaEnv**.
- 3 Right-click AtlantaEnv and, on the menu, click: **New Logical Host**
 - ♦ Retain the default name: LogicalHost1
- 4 For second and subsequent logical hosts—such as Berlin—change the base port:
 - A Right-click LogicalHost1 and open its properties.
 - B Click **Logical Host Configuration** and change the value **Logical Host Base Port** to a larger multiple of 1000. The sample assumes that for Atlanta, you will retain the default (18000); and for Berlin, you will change the base port to 28000.
 - C Close the properties sheet.
- 5 Right-click LogicalHost11, click **New SeeBeyond Integration Server**, and retain the default name: IntegrationSvr1
- 6 Right-click LogicalHost1 and click: **New SeeBeyond JMS IQ Manager**, and retain the default name: SBJmsIQMgr1
- 7 Right-click AtlantaEnv > **New BatchLocalFile External System**
 - ♦ Name it **myExtBatchLocalFile** and click OK.
- 8 Right-click AtlantaEnv > **New File External System**
 - ♦ Name it **myExtFileIn**, set it to **Inbound File eWay**, and click OK.
- 9 Right-click AtlantaEnv > **New File External System**
 - ♦ Name it **myExtFileOut**, set it to **Outbound File eWay**, and click OK.
- 10 Right-click AtlantaEnv > **New HTTP External System**
 - ♦ Name it **myExtHTTP** and click OK.

- 11 Right-click myExtHTTP > **Properties** to initialize its properties, and then click OK.
- 12 Right-click AtlantaEnv > **New HTTP Server External System**
 - ♦ Name it **myExtHTTPServer** and click OK.

Tip: To configure for HTTPS, see the HTTP(S) eWay Intelligent Adapter User's Guide.

- 13 Right-click AtlantaEnv > **New Oracle External System**
 - ♦ Name it **myExtOracleOut**, designate it **Outbound Oracle eWay**, and click OK.
- 14 Right-click AtlantaEnv > **New Constant**
 - ♦ In the **Create a Constant** dialog, name it **X12_DataDir**, give it the value **C:\temp\exChange\Sample\X12\Data\Atlanta** (if you installed sample data files other than in C:\temp\exChange\Sample\Data, substitute appropriately), and click **OK** (see Figure 24); then close the **Variables and Constants** canvas.

Figure 24 Environment Constant X12_DataDir for AtlantaEnv



- 15 On the main toolbar, click **Save All**.
- 16 For Berlin: Repeat all previous steps, substituting "**BerlinEnv**" for "AtlantaEnv"; also, in step 4, be sure to change the base port to 28000; and in step 14, use "...Sample\Data\X12\Berlin".


Result: The environments, named **AtlantaEnv** and **BerlinEnv**, have all but two of the externals needed by the projects; see the right-hand portion of [Figure 23 on page 64](#). Steps for configuring outbound Oracle external are provided in the following procedure. The final externals will be created by activating the project containing the B2B host and then providing keystore information.

To create and configure the Oracle external

Before you begin: Your eXchange 5.0.5 database must be accessible, and you must you know its SID and username/passwords.

Note: This sample assumes that in your Oracle database, you have already created two new username/password combinations: for Atlanta, the sample is preconfigured

with `username/password = ex_A/ex_A`; for Berlin, the sample is preconfigured with `username/password = ex_B/ex_B`. (For details on running the `createuser.sql` script to create `username/password` combinations and running the `createdb.sql` script to populate tablespaces for each user, see the eXchange User's Guide section on "Running Database Scripts to Set Up the eXchange Database".) For convenience, this setup allows you to run the sample on a single machine where two trading partners ("A" and "B") are simulated using two different database usernames and two different logicalhost port ranges; alternatively (and more realistically), you could create separate databases on separate machines.

- 1 In the Environment Explorer tree, open **AtlantaEnv**, right-click **myExtOracleOut**, check it out if necessary, and configure properties appropriately. For example:
 - ♦ **DatabaseName:** *eXchange* (change this to the SID for your eXchange 5.0.5 database)
 - ♦ **DataSourceName:** **local**
 - ♦ **Password:** (replace this with the password for your eXchange 5.0.5 database user "A")
 - ♦ **PortNumber:** **1521** (change this only if your Oracle administrator changed the default)
 - ♦ **ServerName:** *thatMachine* (change this to the hostname of the Oracle server machine)
 - ♦ **User:** *ex_A* (change this to username "A" for your eXchange 5.0.5 database)
- 2 When all properties have been configured correctly for your site, click OK.
- 3 For Berlin: Repeat all previous steps, but instead of "Atlanta", substitute "Berlin" and instead of "A", substitute "B".
- 4 Collapse all environment trees, click AtlantaEnv, click  **Save All**, and close all canvases.

Result: The environments are now well-configured; they contain all externals needed for activation of the B2B Host. You can export the environments at this point if you like; you should not export an environment after it contains an eXchange service.

The final externals—eXchange services for the Atlanta and Berlin environments—will be created by activating the project that contains the B2B host.

6.2.2 Viewing and Activating the B2B Host Project

In the Project Explorer tree, open the B2B host project (named **X12_Host**) to display its components. Below is a summary of the B2B host's contents. Activating this project will create an eXchange service that acts as a channel manager and provides a connection to the message tracking application and the eXchange database.

Components of the B2B Host Project (X12_Host)

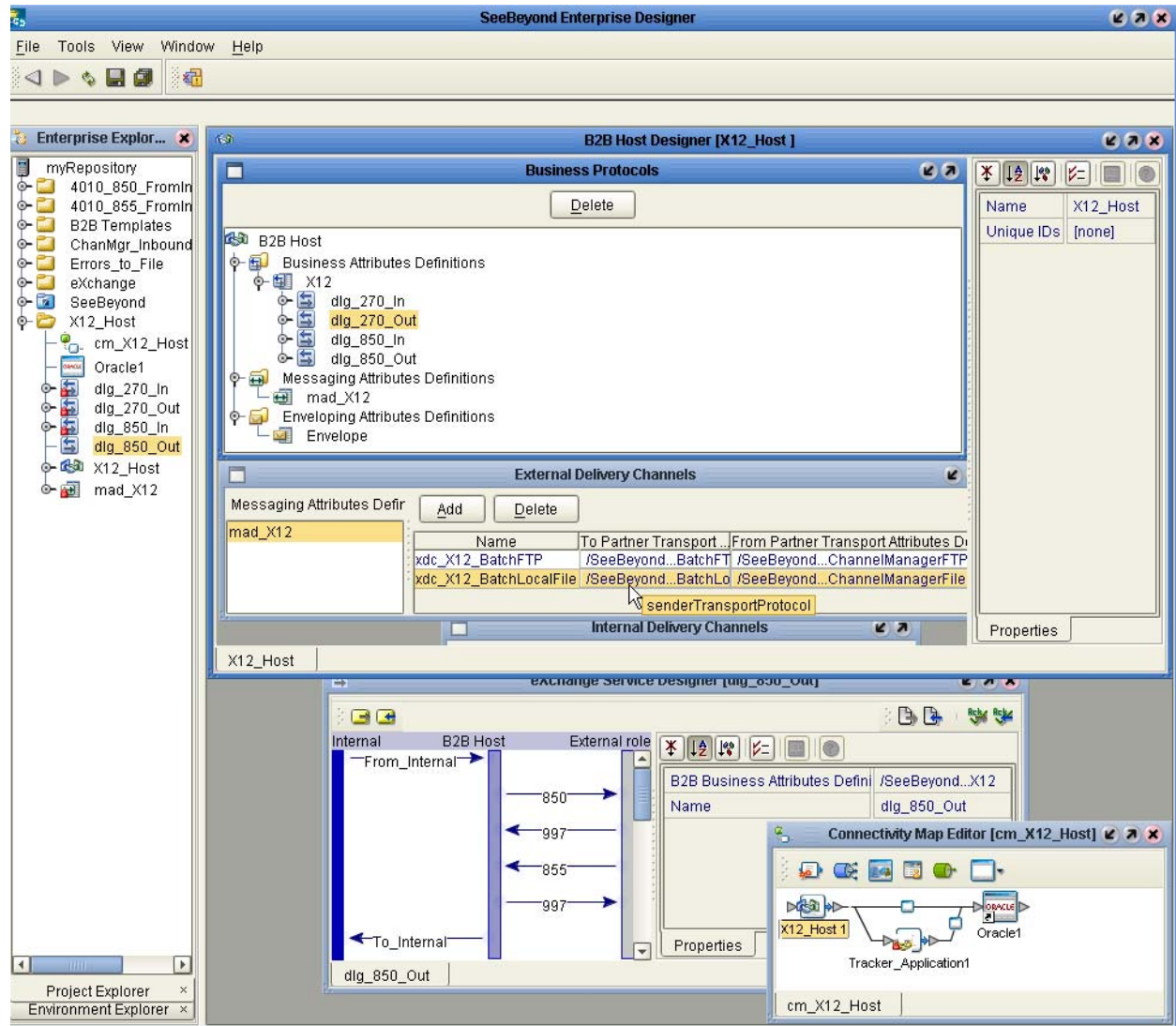
- **dlg_<action>_Out** and **dlg_<action>_In** are the host project's services, also called "business dialogs":
 - ♦ The **dlg_<action>_Out** dialog has six actions (see [Figure 25 on page 69](#)): It starts with an outbound internal-to-host action, then continues with an outbound-to-TP action (such as an 850 or a 270) and a 997 acknowledgment,

followed by an inbound-from-TP reply (such as an 855 or a 271) and a 997 acknowledgment; and it ends with an inbound host-to-internal action.

- ♦ The **dlg_<action>_In** dialog also has six actions: It starts with an inbound-from-TP action and a 997 acknowledgment, then continues with an inbound host-to-internal action, followed by an outbound internal-to-host action; and it ends an outbound-to-TP reply action and a 997 acknowledgment.
- **mad_X12** is the host project's message attributes definition (MAD).
- **X12_Host** is the name of the B2B host itself, as well as the project name:
 - ♦ Its Business Protocols window references the services (business dialogs) named **dlg_<action>_Out** and **dlg_<action>_In**. To find them, open the Business Attributes Definitions (BADs) folder and open the BAD named "X12".
 - ♦ This window also references the messaging attributes definition (MAD), **mad_X12**. To find it, open the Messaging Attributes Definitions folder.
 - ♦ Its External Delivery Channels window defines two delivery channels for the X12Messaging MAD. The sample implementation uses only the channel named **xdc_X12_BatchLocalFile**, which references the BatchLocalFile transport attributes definition (TAD) as its *sender* (to partner) transport protocol, and a ChannelManagerFile TAD as its *receiver* (from partner) transport protocol.
- **cm_X12_Host** is the map whose activation will create the "eXchange Service" external:
 - ♦ Its only input is an instance of X12_Host, with two connections going out (towards the right).
 - ♦ Its only output is an instance of Oracle, with two connections coming in (from the left).
 - ♦ Connecting to both is an instance of a SeeBeyond-supplied tracking application.

See Figure 25.

Figure 25 Components of B2B Host Project (“X12_Host”)



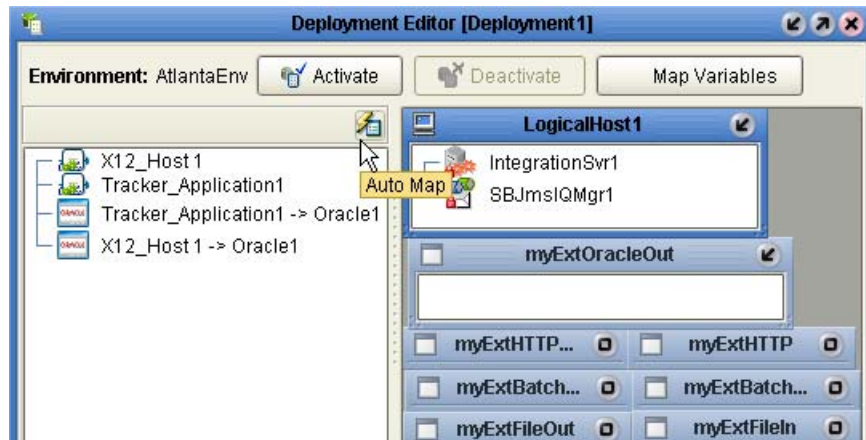
To activate the B2B host, creating the eXchange service

Before you begin: Your environment must contain a well-configured Oracle external (see the preceding procedure), and your Oracle database services must be running.

- 1 Right-click the **X12_Host** project and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Keep the default name (**Deployment1**), point it at **AtlantaEnv**, and click OK. (The sample assumes Deployment1 points at AtlantaEnv and Deployment2 points at BerlinEnv. If you vary from these assumptions, you will need to make corresponding changes in ePM.)

The Deployment Editor opens. Its left pane has two services and two Oracle eWays.

Figure 26 Auto-Mapping Services and eWays

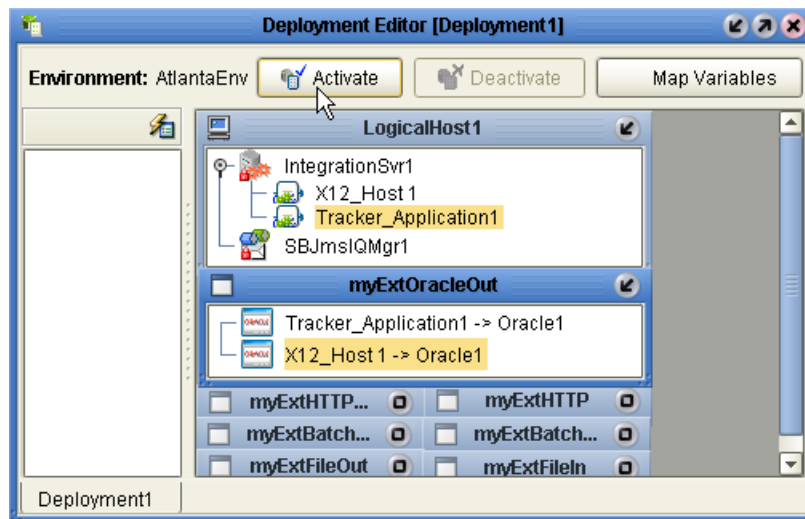


- 3 Click the **Auto Map** button; see Figure 26.

The services are assigned to the Integration Server, and the eWays to the Oracle external; see Figure 27.

Tip: *If the Oracle external refuses to accept either or both eWays, it may be an indication that it was misdefined as inbound. Delete myExtOracleOut and re-create it as outbound. Then: Click **Save All**, followed by **Refresh All from Repository**.*

Figure 27 Deployment Profile for B2B Host, Before Activation



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


Result: A new external is created, named **X12_Host1 eXchange Service**.

Tip: *If activation is unsuccessful, it may be an indication that the eXchange 5.0.5 database instance (normally named “eXchange”) is inaccessible. Ensure it is running and that the myExtOracleOut properties match its hostname, SID, username, and password. If necessary, see [“To create and configure the Oracle external” on page 66](#).*

- 6 For Berlin: Open Environment Explorer and click **BerlinEnv**. Then return to Project Explorer and repeat all previous steps, but with the following substitutions: make sure the deployment profile is named **Deployment2**, and make sure it references **BerlinEnv**.

Result: Another new external is created, also named **X12_Host1 eXchange Service**.

Tip: *If either or both of the eXchange Service appear misplaced in the environment tree, it reflects either an uncollapsed tree or a mis-selected environment upon activation. This display error will correct itself when the Repository is refreshed.*

Result: The environments now have all the externals they need. Save all of your work, close all canvases, and click  **Refresh All from Repository**.

6.2.3. Creating and Activating the Project Deployment Profiles

Before you begin: Both environments (AtlantaEnv and BerlinEnv) must already contain eXchange services; if necessary, see the [procedure on page 69](#).

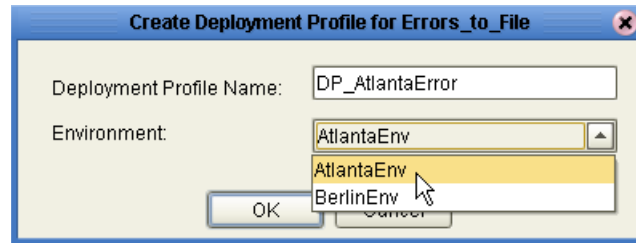
After you have created and activated deployment profiles for the B2B host project (X12_Host), you now create/activate deployment profiles for other projects:

- Project **Errors_to_File** requires two deployment profiles. The first DP points at AtlantaEnv; the second, at BerlinEnv. Also, because its File eWay uses a project variable (for its Directory property), you must map the **Directory** variable to the environment constant **DataDir** before activating the DPs.
- Project **ChanMgr_Inbound_FromPartner** also requires two deployment profiles (one referencing AtlantaEnv, the other referencing BerlinEnv).
- You can use either or both sets of “feeder” projects; the 4010_850/855 projects use OTD library and validator BP from X12 v4010, whereas the 270/271 projects use the OTD library and validator BP from HIPAA 2000 Addenda (a separately licensed product). In each case, the DP for the project named *<request>_FromInt_Inc* must reference AtlantaEnv, and the DP for the project named *<reply>_FromInt_<request>* must reference BerlinEnv. Both of the *<request>_FromInt_Inc* projects have File eWays that use a project variable (for its Directory property), you must map the **DataDir** variable to the environment constant **DataDir** before activating the DPs.

To activate the **Errors_to_File** project

- 1 In the Project Explorer tree, right-click **Errors_to_File** and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Rename it to **DP_AtlantaError**, point it at **AtlantaEnv**, and click OK. See Figure 28.

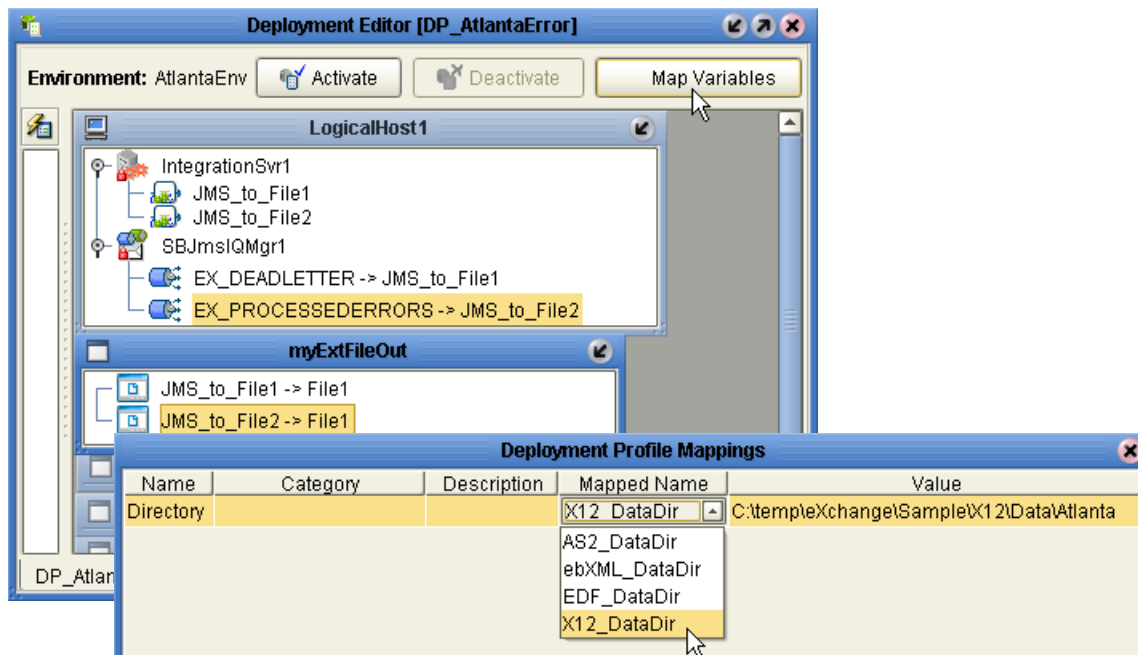
Figure 28 Deployment Profile for Error Processing



The Deployment Editor’s left pane displays two JCD services, two topics, and two outbound File eWays.

- 3 For clarity, minimize all externals except LogicalHost1 and myExtFileOut.
- 4 Click the **Auto Map** button. This assigns all components to their servers:
 - ♦ The JCD services are assigned to **IntegrationSvr1**
 - ♦ The topics are assigned to **SbJmsIQMgr1**
 - ♦ The outbound File eWays are assigned to **myExtFileOut**
- 5 Click **Map Variables**. When the Deployment Profile Mappings dialog box appears: In the row for the variable named **Directory**, click in the **Mapped Name** column and, from the drop-down list, choose **X12_DataDir**. Then click **OK**. See Figure 29.

Figure 29 Mapping Variables



- 6 When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers and the variables are mapped—click **Save All**, and then click **Activate**. In response to the dialog box, click **No**; that is, do *not* apply to Logical Host(s).

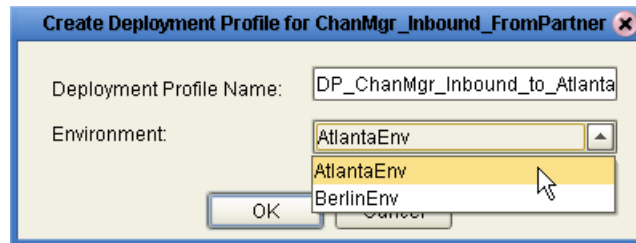
Tip: *If you receive an error message regarding a JNDI name prefix—`CodeGenException: Failed to obtain JNDI name prefix(ERROR_GET_JNDINAMEPREFIX)`—ensure your variables have been correctly defined and are correctly mapped.*

- 7 For Berlin: Repeat all previous steps, but with the following substitutions: name the deployment profile **DP_BerlinError**, point it at **BerlinEnv**, and when you map the variables, confirm that the value for **X12_DataDir** is a path to a **...Berlin** folder.

To activate the **ChanMgr_Inbound_FromPartner** project

- 1 In the Project Explorer tree, right-click **ChanMgr_Inbound_FromPartner** and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Rename it to **DP_ChanMgr_Inbound_to_Atlanta**, point it at **AtlantaEnv**, and click OK. See Figure 30.

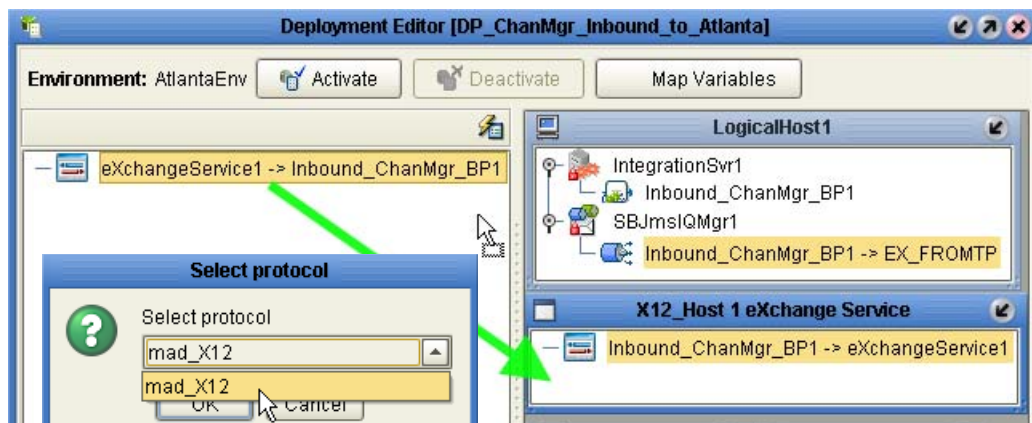
Figure 30 Deployment Profile for Channel Manager Inbound From Partner



The Deployment Editor's left pane displays a BP service, a topic, and two eXchangeService connections.

- 3 Minimize all externals except **LogicalHost1** and **X12_Host1** eXchange Service.
- 4 Click the **Auto Map** button. This assigns all components to their servers:
 - ♦ The BP service is assigned to **IntegrationSvr1**
 - ♦ The topic is assigned to **SbJmsIQMgr1**
 - ♦ One eXchangeService connection is assigned to **X12_Host eXchange Service**
- 5 Drag the other eXchangeService connection into **X12_Host eXchange Service** and, in the Select Protocol dialog, choose **mad_X12** (the only choice). See Figure 31.

Figure 31 Assigning an eXchangeService Connection and Selecting Its Protocol



- When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers and the variables are mapped—click **Save All**, and then click **Activate**. In response to the dialog box, click **No**; that is, do not apply to Logical Host(s).

Tip: *If you receive an error message regarding a JNDI name prefix—CodeGenException: Failed to obtain JNDI name prefix(ERROR_GET_JNDINAMEPREFIX)—ensure your variables have been correctly defined and are correctly mapped.*

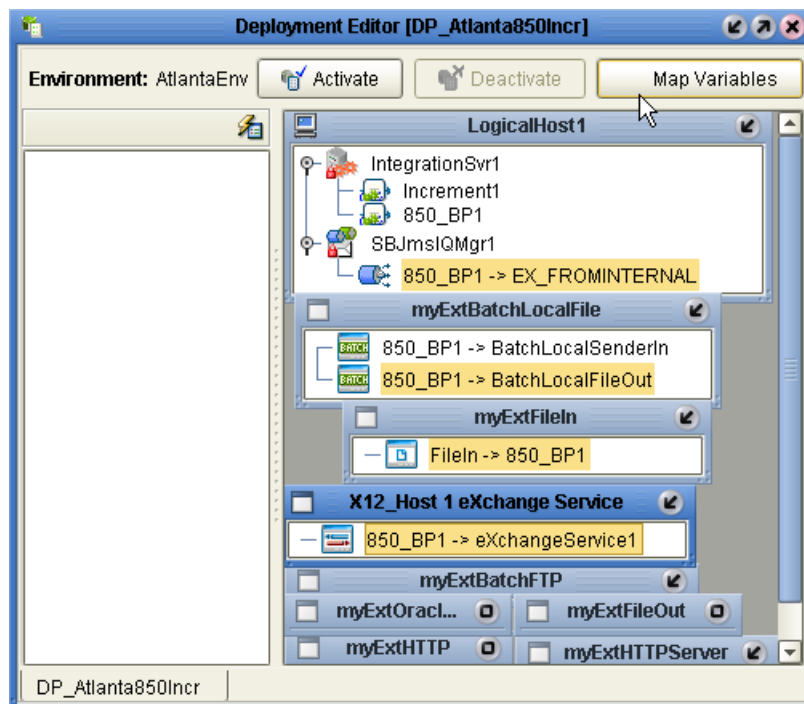
- For Berlin: Repeat all previous steps, but with the following substitutions: name the deployment profile **DP_ChanMgr_Inbound_to_Berlin**, and point it at **BerlinEnv**.

To activate the `<request>_FromInt_Inc` projects (AtlantaEnv only)

- In the Project Explorer tree, right-click **4010_850_FromInt_Inc** and, on the popup context menu, point at **New** and click **Deployment Profile**
- Rename it to **DP_Atlanta850Incr**, point it at **AtlantaEnv**, and click OK.
- Click the **Auto Map** button. This assigns all components to their servers:
 - The two BP services are assigned to **IntegrationSvr1**
 - The topic is assigned to **SbJmsIQMgr1**
 - The two outbound BatchLocalFile eWays are assigned to **myExtBatchLocalFile**
 - The inbound File eWay is assigned to **myExtFileIn**
 - The eXchangeService connection is assigned to **X12_Host eXchange Service**

See Figure 32.

Figure 32 Assigning Components for an Atlanta Request Project (Feeder/Increment)

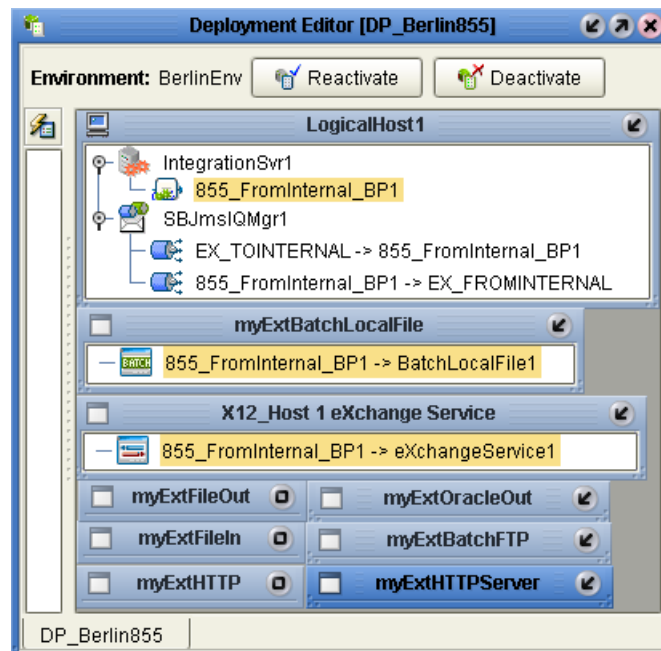


- 4 Click **Map Variables**. When the Deployment Profile Mappings dialog box appears: In the row for variable **DataDir**, click in the **Mapped Name** column and, from the drop-down list, choose **X12_DataDir**. Then click **OK**.
- 5 When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers and the variables are mapped—click **Save All**, and then click **Activate**. In response to the dialog box, click **No**.
- 6 If you are also using 270/271 processing: Open the project **270_FromInt_Inc** and repeat the same steps (except, in step 2, use the name **DP_Atlanta270Incr**).

To activate the `<reply>_FromInternal_<request>` project (BerlinEnv only)

- 1 In the Project Explorer tree, right-click **4010_855_FromInt_850** and, on the popup context menu, point at **New** and click **Deployment Profile**
- 2 Rename it to **DP_Berlin855**, point it at **BerlinEnv**, and click **OK**.
- 3 Click the **Auto Map** button. This assigns all components to their servers:
 - ♦ The BP service is assigned to **IntegrationSvr1**
 - ♦ The two topics are assigned to **SbJmsIQMgr1**
 - ♦ The outbound BatchLocalFile eWay is assigned to **myExtBatchLocalFile**
 - ♦ The eXchangeService connection is assigned to **X12_Host eXchange Service**
- 4 Click **Map Variables** and map variable **DataDir** to **X12_DataDir**. Then click **OK**.
- 5 When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers and the variables are mapped—click **Save All**, and then click **Activate**. In response to the dialog box, click **No**. The result is shown in Figure 33.

Figure 33 Activation of a Berlin Reply Project



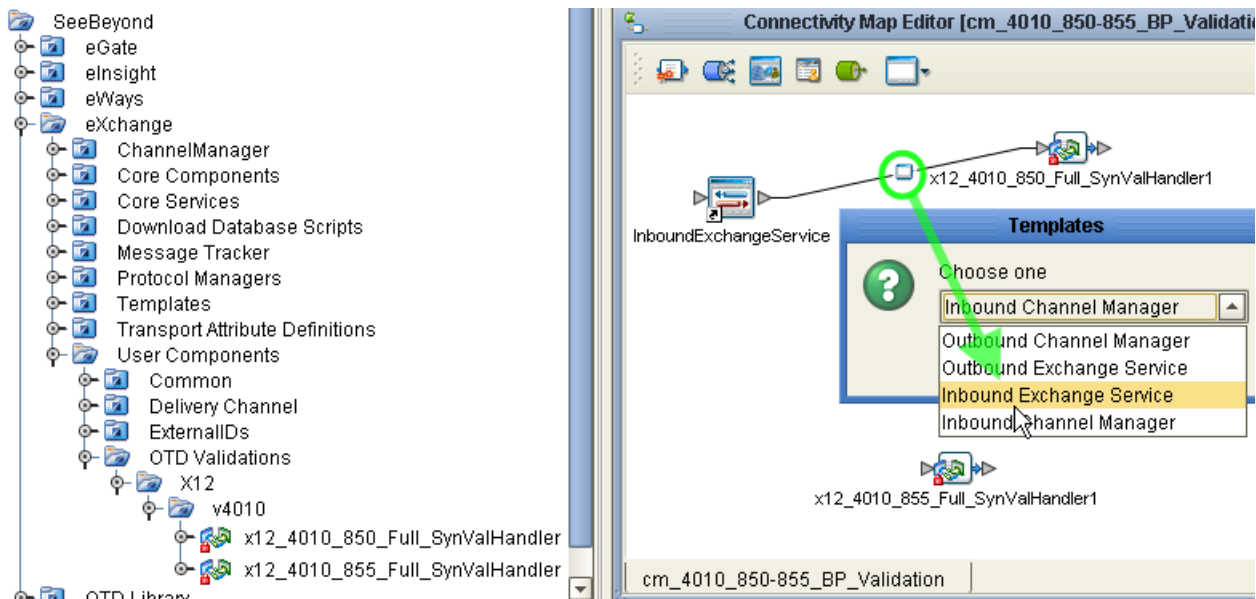
- 6 If you are also using 270/271 processing: Open the project **271_FromInt_270** and repeat the same steps (except, in step 2, use the name **DP_Berlin271**).

To create and activate the main (BP_Validation) project for 850/855


Tip: For a preview of the final result before activation, see **Figure 35 on page 78**.

- 1 In the Project Explorer tree, open the eXchange folder, right-click its **Deployment** folder and, on the popup context menu, point at **New** and click **Connectivity Map**
- 2 Rename the new connectivity map to: **cm_4010_850-855_BP_Validation**
- 3 Open the following folder so that you can see its contents: SeeBeyond > eXchange > User Components > OTD Validations > X12 > **v4010**. In the folder:
 - A Click **x12_4010_850_Full_SynValHandler**, and then drag it to the upper right of the connectivity map.
 - B Click **x12_4010_855_Full_SynValHandler**, and then drag it to the lower right of the connectivity map.
- 4 Back in the eXchange > **Deployment** folder: Click **InboundExchangeService** and then drag it to the middle left of the connectivity map.
- 5 In the connectivity map:
 - A Drag the arrow pointing out of **InboundExchangeService** to the arrow pointing into **x12_4010_850_Full_SynValHandler1** (the BP in the canvas's upper right)
 - B When the connection is formed: Double-click the connection box and, in the Templates dialog, click **Inbound Exchange Service**. See Figure 34.

Figure 34 Configuring InboundExchangeService Connection to the Validation Handler

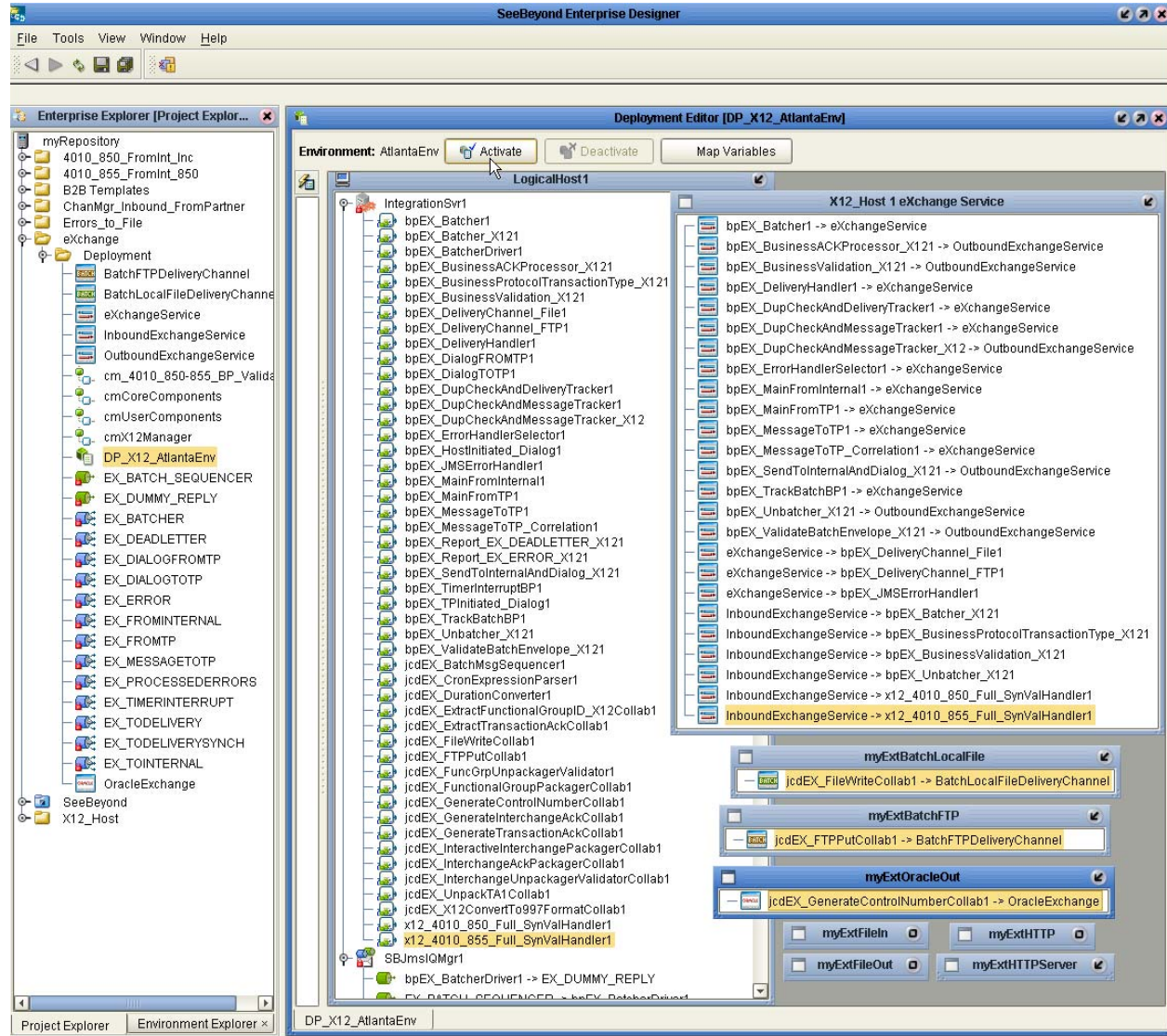


- C Click OK. Then click OK again to close the Properties dialog.
- D Drag the arrow pointing out of **InboundExchangeService** to the arrow pointing into **x12_4010_855_Full_SynValHandler1** (the BP in the canvas's lower right)

- E As before, when the connection is formed, double-click the connection box and, in the Templates dialog, click **Inbound Exchange Service** and click OK. Then click OK again to close the Properties dialog.
- 6 In the project tree, right-click the **Deployment** folder and, on the popup context menu, point at **New** and click **Deployment Profile**
- 7 Rename it to **DP_X12_AtlantaEnv**, point it at **AtlantaEnv**, and click OK. (If you use a different name for the DP, you will need to make corresponding changes in ePM.)
The Deployment Editor's left pane displays: A long list of services, topics, and queues; a long list of connections for one inbound and four outbound eXchange services; and outbound eWays for BatchLocalFile, BatchFTP, and Oracle.
- 8 Click the  **Auto Map** button. This assigns all components to their servers:
- ♦ The services are assigned to **IntegrationSvr1**
 - ♦ The topics and queues are assigned to **SbJmsIQMgr1**
 - ♦ The eXchange Service connectors are assigned to **X12_Host1_eXchange Service**
 - ♦ The outbound BatchLocalFile eWay is assigned to **myExtBatchLocalFile**
 - ♦ The outbound BatchFTP eWay is assigned to **myExtBatchFTP**
 - ♦ The outbound Oracle eWay is assigned to **myExtOracle**

Result: See Figure 35.

Figure 35 Deployment Profile DP_X12_AtlantaEnv



9 When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers—click **Save All**, and then click **Activate**. In response to the dialog box, click **No**; that is, do *not* apply to Logical Host(s).

Tip: Successful activation will take some time.

10 For Berlin, repeat steps 6 through 9, but with two substitutions: Name the deployment profile DP_X12_BerlinEnv, and point it at BerlinEnv.



11 Click **Save All**, close all canvases, and click **Refresh All from Repository**.

Final result: All projects are activated, and will run when the logical hosts are started.

To create and activate the main (BP_Validation) project for 270/271 (optional)



Tip: For a preview of the final result before activation, see [Figure 35 on page 78](#).

- 1 In the Project Explorer tree, open the eXchange folder, right-click its **Deployment** folder and, on the popup context menu, point at **New** and click **Connectivity Map**
- 2 Rename the new connectivity map to: **cm_270-271_BP_Validation**
- 3 Open the following folder so that you can see its contents: SeeBeyond > eXchange > User Components > OTD Validations > HIPAA > **2000_Addenda**. In the folder:
 - A Click **addenda_hipaa_270_Full_SynValHandler**, and then drag it to the upper right of the connectivity map.
 - B Click **addenda_hipaa_271_Full_SynValHandler**, and then drag it to the lower right of the connectivity map.
- 4 Back in the eXchange > **Deployment** folder: click **X12_Host1 eXchange Service** and then drag it to the middle left of the connectivity map.
- 5 In the connectivity map:
 - A Connect the arrow pointing out of **InboundExchangeService** to the arrow pointing into **addenda_hipaa_270_Full_SynValHandler**
 - B When the connection is formed: Double-click the connection box and, in the Templates dialog, click **Inbound Exchange Service** and click OK. Then click OK again to close the Properties dialog.
 - C Drag the arrow pointing out of **InboundExchangeService** to the arrow pointing into **addenda_hipaa_271_Full_SynValHandler**
 - D When the connection is formed: Double-click the connection box and, in the Templates dialog, click **Inbound Exchange Service** and click OK. Then click OK again to close the Properties dialog.
- 6 In the project tree, right-click the **Deployment** folder and, on the popup context menu, point at **New** and click **Deployment Profile**
- 7 Rename it to **DP_270-271_AtlantaEnv**, point it at **AtlantaEnv**, and click OK.

The Deployment Editor's left pane displays: A long list of services, topics, and queues; a long list of connections for one inbound and four outbound eXchange services; and outbound eWays for BatchLocalFile, BatchFTP, and Oracle.
- 8 Click the  **Auto Map** button. This assigns all components to their servers:
 - ♦ The services are assigned to **IntegrationSvr1**
 - ♦ The topics and queues are assigned to **SbJmsIQMgr1**
 - ♦ The eXchange Service connectors are assigned to **X12_Host1_eXchange Service**
 - ♦ The outbound BatchLocalFile eWay is assigned to **myExtBatchLocalFile**
 - ♦ The outbound BatchFTP eWay is assigned to **myExtBatchFTP**
 - ♦ The outbound Oracle eWay is assigned to **myExtOracle**
- 9 When the profile is complete—that is, when all runnable components in the project are associated with corresponding external servers—click  **Save All**, and then

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

Tip: *Successful activation will take some time.*

- 10 For Berlin, repeat steps 6 through 9, but with two substitutions: Name the deployment profile DP_270-271_ **BerlinEnv**, and point it at **BerlinEnv**.
- 11 Click  **Save All**, close all canvases, and click  **Refresh All from Repository**.

Final result: All projects are activated, and will run when the logical hosts are started.

6.3 Design Steps in ePM

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

- ♦ [Importing the Trading Partners into the Environments on page 81](#)
- ♦ [Configuring Trading Partner Parameters for the “Berlin” TP on page 81](#)
- ♦ [Activating the Trading Partner on page 90](#)
- ♦ [Configuring Trading Partner Parameters for the “Atlanta” TP on page 90](#)
- ♦ [Activating the Trading Partner on page 98](#)

Before you begin: Your repository and your eXchange 5.0 Oracle database must be running and accessible. Enterprise Designer does *not* need to be running, and you do *not* need to have any logical hosts running.

To start eXchange Partner Manager (ePM)

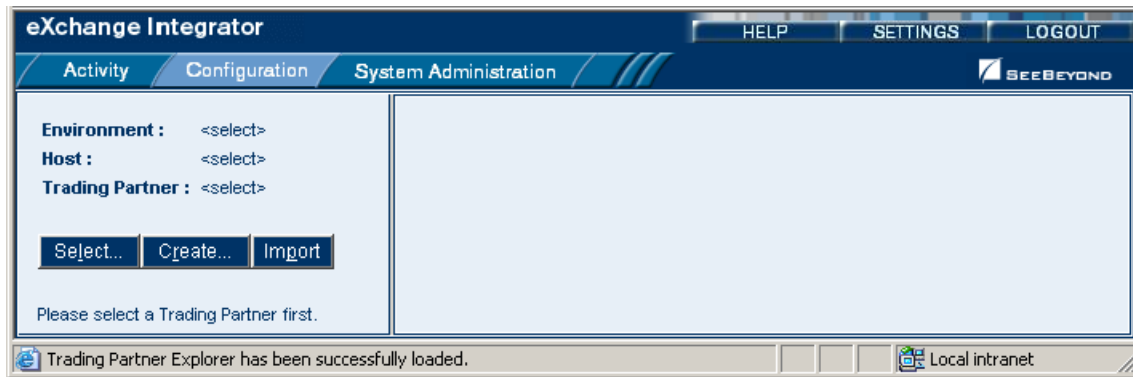
- 1 Start a *new* browser session (that is, do *not* clone a new window of an existing session) pointing it at a repository URL, with **epm** appended. For example:
 - ♦ If your repository were running local on port 12000, the URL would be:
`http://localhost:12000/epm`
 - ♦ For a repository running on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

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

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

Result: The status bar (along the lower margin of the window) confirms that Trading Partner Explorer has loaded successfully, and the initial ePM screen appears, with no environment, host, or trading partner. See Figure 36.

Figure 36 Initial ePM Screen



6.3.1 Importing the Trading Partners into the Environments

To import the “Atlanta” trading partner into the Berlin environment

- 1 From the initial ePM screen, in the upper left side, click **Import**.
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **BerlinEnv** and click **X12_Host1**.
- 3 For Name, enter **Atlanta**. Browse to C:\temp\exchange\Sample\TradingPartners and open **X12_TP_for-BerlinEnv_to-from_Atlanta.xml**.
- 4 Click the **Import** button and wait for the profile to be modified and imported.

Result: In the explorer tree, under **BerlinEnv**, new trading partner **Atlanta** appears.

To import the “Berlin” trading partner into the Atlanta environment

- 1 In the upper left side of the ePM screen, click **Import**.
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **AtlantaEnv** and click **X12_Host1**.
- 3 For Name, enter **Berlin**. Browse to C:\temp\exchange\Sample\TradingPartners and open **X12_TP_for-AtlantaEnv_to-from_Berlin.xml**.
- 4 Click the **Import** button and wait for the profile to be modified and imported.

Result: In the explorer tree, under **AtlantaEnv**, new trading partner **Berlin** appears.

6.3.2 Configuring Trading Partner Parameters for the “Berlin” TP

When you imported trading partner “Berlin” into the Atlanta environment, parameter settings were valuated in part based on parameters stored in the export file, and in part based on the name of the trading partner. In this section, you will view, set, and/or update the following parameters for the delivery channel for trading partner “Berlin”.

TP “Berlin”: Parameters for the Delivery Channel

Purpose: To view the parameters governing Atlanta’s message exchange with Berlin.

Keep in mind: You have imported a trading partner that was configured for the Atlanta environment, and so you will take the viewpoint of the Atlanta B2B host: “ToPartner” means “to Berlin”; “FromPartner” means “from Berlin”.

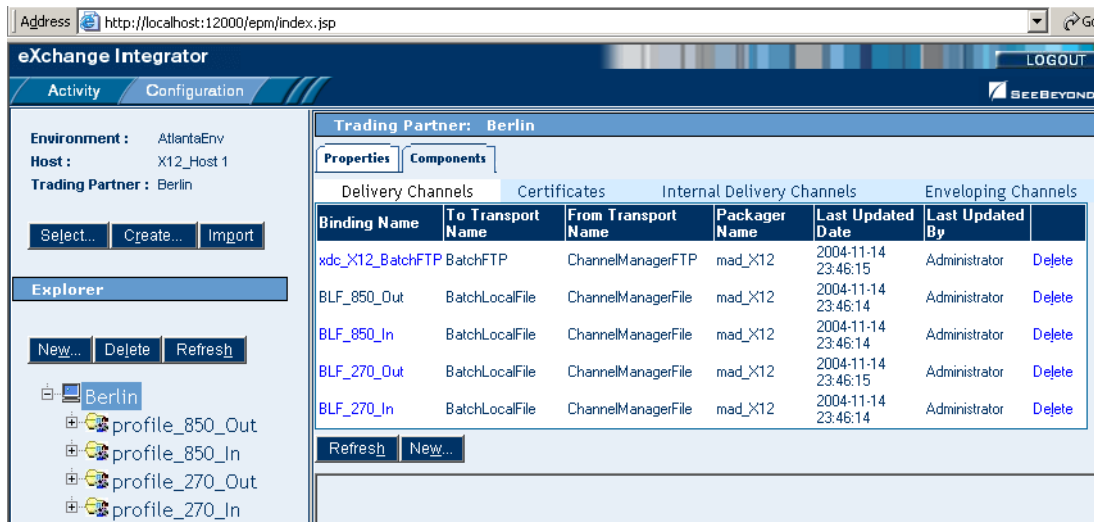
To view or reconfigure delivery channel parameters for trading partner “Berlin”

- 1 In the explorer (lower left) side of the ePM screen, click **Berlin** to display the trading partner’s general properties.
- 2 Click the **Components** tab to display the trading partner’s delivery channel parameters. See Table 11 and Figure 37.

Table 11 List of Delivery Channels for Trading Partner “Berlin”

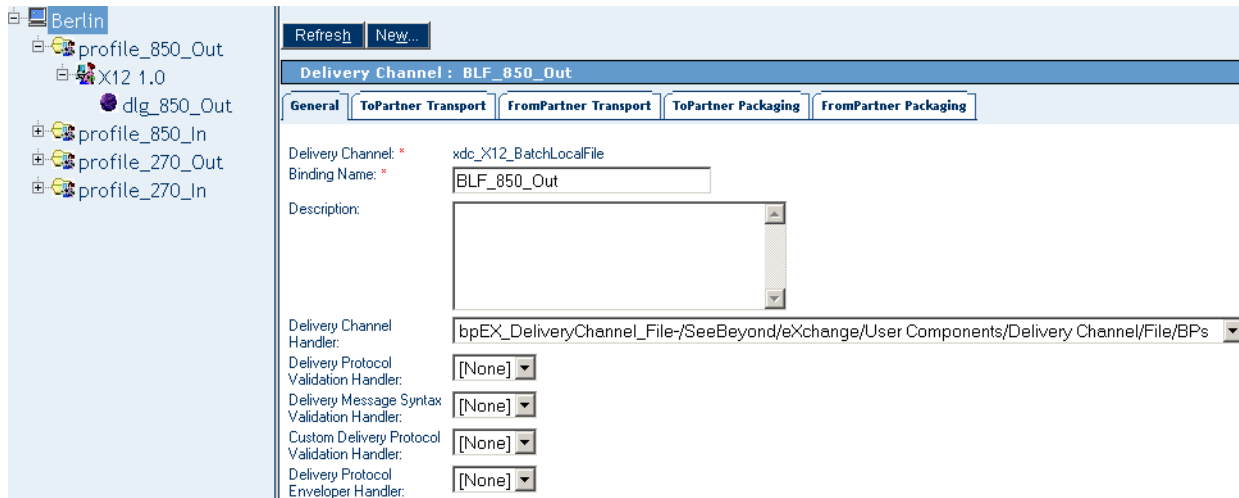
Binding Name	ToPartner Transport Name	FromPartner Transport Name	Packager Name	Comments
xdc_X12_BatchFTP	BatchFTP	ChannelManagerFTP	mad_X12	(not used)
BLF_850_Out	BatchLocalFile	ChannelManagerFile	mad_X12	
BLF_850_In	BatchLocalFile	ChannelManagerFile	mad_X12	
BLF_270_Out	BatchLocalFile	ChannelManagerFile	mad_X12	(optional)
BLF_270_In	BatchLocalFile	ChannelManagerFile	mad_X12	(optional)

Figure 37 List of Delivery Channel Bindings for Trading Partner “Berlin”



- 3 Click one of the binding names, such as **BLF_850_Out**, to display its general properties. See Figure 38.

Figure 38 Delivery Channel for “Berlin”: General Properties for BLF_850_Out



- 4 Click the **ToPartnerTransport** tab to display its values, especially to verify the target directory, which must match the value of the DataDir environment constant for AtlantaEnv.

The values provided with the sample are shown in Table 12. If you need to make any changes, be sure to click **Save** after you finish.

Table 12 ToPartnerTransport Parameters for Trading Partner “Berlin”

All Purpose End Point	Atlanta To Berlin
Use Synchronous Channel	false
Track for Auditing	false
Append	[None]
TargetDirectoryName	C:\temp\eXchange\Sample\X12\Data\Atlanta
TargetDirectoryNamesPattern	[None]
TargetFileName	Berlin_<action>_<direction>_ISA_%d_%H%m%s_%S.isa (Example1: For the binding named BLF_850_Out , the filename is Berlin_850_Out_ISA_%d_%H%m%s_%S.isa (Example2: For the binding named BLF_270_In , the filename is Berlin_270_In_ISA_%d_%H%m%s_%S.isa)
TargetFileNamesPattern	true
MaxSequenceNumber	
StartingSequenceNumber	
PreDirectoryName	
PreDirectoryNamesPattern	[None]
PreFileName	
PreFileNamesPattern	[None]
PreTransferCommand	
PostDirectoryName	
PostDirectoryNamesPattern	[None]

Table 12 ToPartnerTransport Parameters for Trading Partner “Berlin” (Continued)

PostFileName	
PostFileNamesPattern	[None]
PostTransferCommand	
ResumeReadingEnabled	[None]
ActionOnMalformedCommand	
IncludeOrderRecordInErrorRecord	[None]
IncludePayloadInErrorRecord	[None]
PublishStatusRecordOnError	[None]
PublishStatusRecordOnSuccess	[None]

- 5 Click the **FromPartnerTransport** tab to display its values, especially to verify the target directory, which must match the value of the DataDir environment constant for BerlinEnv.

The values provided with the sample are shown in Table 13. If you need to make any changes, be sure to click **Save** after you finish.

Table 13 FromPartnerTransport Parameters for Trading Partner “Berlin”

All Purpose End Point	Berlin To Atlanta
Use Synchronous Channel	false
Track for Auditing	false
ChannelManagerMode	FILE
FilePattern	Atlanta_<action>_<opposite-direction>_ISA_*.isa (Example: For BLF_850_Out , this is Atlanta_850_In_ISA_*.isa)
Directory	C:\temp\Exchange\Sample\X12\Data\Berlin
PollMilliSeconds	10000

- 6 Click the **ToPartnerPackaging** tab to display its values. The values provided with the sample are shown in Table 14. If you need to make any changes, be sure to click **Save** after you finish..

Table 14 ToPartnerPackaging Parameters for Trading Partner “Berlin”

Encryption Key	[None]
Check for Duplicates	true
Use Encryption	false
Send Acknowledgments	true
Expect Acknowledgments	true
Character Set Encoding	
Send warnings with ACKs	true
Use Signature	false
Use Compression	false

Table 14 ToPartnerPackaging Parameters for Trading Partner “Berlin” (Continued)

Message Encoding	
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Batcher Handler	bpEX_Batcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Batch Encryption Handler	[None]
Batch Signing Handler	[None]
Batch Compression Handler	[None]
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]

- Click the **FromPartnerPackaging** tab to display its values; the values provided with the sample are shown in Table 15. If you need to make any changes, be sure to click **Save** after you finish..

Table 15 FromPartnerPackaging Parameters for Trading Partner “Berlin”

Signature Certificate	[None]
Check for Duplicates	true
Use Encryption	false
Send Acknowledgments	true
Expect Acknowledgments	false
Character Set Encoding	
Send warnings with ACKs	true
Use Signature	false
Use Compression	false
Message Encoding	
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Batch Splitter Handler	bpEX_Unbatcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Batch Decryption Handler	[None]
Batch Signature Validation Handler	[None]
Batch Decompression Handler	[None]
Decryption Handler	[None]
Signature Validation Handler	[None]
Decompression Handler	[None]

- 8 Browse other bindings for delivery channel `xdc_X12_BatchLocalFile`, such as `BLF_850_In` (and, if you use them, `BLF_270_Out` and `BLF_270_In`). If you need to make any changes, be sure to click **Save** after you finish.

Result: For trading partner “Berlin”, the parameters for all bindings for external delivery channel `xdc_X12_BatchLocalFile` are now set correctly, and you are ready to view and/or configure the parameters for the Enveloping Channel.

To view or reconfigure enveloping channel parameters for trading partner “Berlin”

- 1 In the explorer (lower left) side of the ePM screen, click **Berlin** to display the trading partner’s general properties.
- 2 In the canvas, click the **Components** tab; click the Enveloping Channels subtab to display the trading partner’s single enveloping channel, and then click its binding name (`EnvelopeChannel`) to display general parameters for this envelope channel.
- 3 Click the **ToPartner Envelope Binding** tab to display its values. The values provided with the sample are shown in Table 16. If you need to make any changes, be sure to click **Save** after you finish.

Table 16 ToPartner Enveloping Parameters for Trading Partner “Berlin”

Release Quantity	1
Release Scheduler String	50 * * ? * *
Time-Out (Minutes)	10
Max Retry Count	1
Batcher Handler	bpEX_Batcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]
ISA01 AUTHOR INFO QUAL	00
ISA02 AUTHOR INFORMATION	
ISA03 SEC INFO QUAL	00
ISA04 SECURITY INFORMATION	
ISA05 IC SENDER ID QUAL	01
ISA06 INTERCHANGE SENDER ID	SIDAtlanta
ISA07 IC RCVR ID QUAL	01
ISA08 INTERCHANGE RCVR ID	RID:Berlin
ISA11 IC CONTROL STANDARD ID	U
ISA12 IC VERSION NUMBER	00401
ISA13 IC CONTROL NUMBER	0
ISA14 ACKNOWLEDGMENT REQUESTED	1
ISA15 USAGE INDICATOR	T
ISA16 COMP ELE SEP	:

Table 16 ToPartner Enveloping Parameters for Trading Partner “Berlin” (Continued)

SEGMENT TERMINATOR	~
ELEMENT SEPARATOR	*
GS06 GROUP CONTROL NUM	0

- 4 Click the **FromPartner DeEnvelope Binding** tab to display its values. The values provided with the sample are shown in Table 17. If you need to make any changes, be sure to click **Save** after you finish.

Table 17 FromPartner Enveloping Parameters for Trading Partner “Berlin”

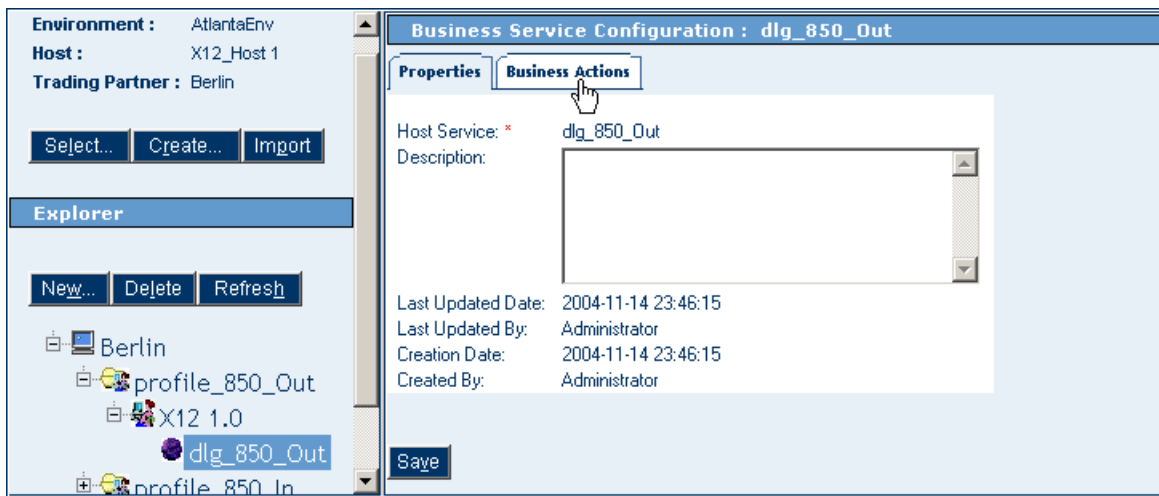
Batcher Splitter Handler	bpEX_Unbatcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Decryption Handler	[None]
Signature Validation Handler	[None]
Deompression Handler	[None]
ISA01 AUTHOR INFO QUAL	00
ISA02 AUTHOR INFORMATION	
ISA03 SEC INFO QUAL	00
ISA04 SECURITY INFORMATION	
ISA05 IC SENDER ID QUAL	01
ISA06 INTERCHANGE SENDER ID	SID:Berlin
ISA07 IC RCVR ID QUAL	01
ISA08 INTERCHANGE RCVR ID	RIDAtlanta
ISA11 IC CONTROL STANDARD ID	U
ISA12 IC VERSION NUMBER	00401
ISA13 IC CONTROL NUMBER	0
ISA14 ACKNOWLEDGMENT REQUESTED	1
ISA15 USAGE INDICATOR	T

Result: For trading partner “Berlin”, the parameters for delivery and enveloping channels are now set correctly, and you are ready to view and/or reconfigure each of the actions in each of the business services.

To configure trading partner “Berlin”, business service “dlg_850_Out” (required steps)

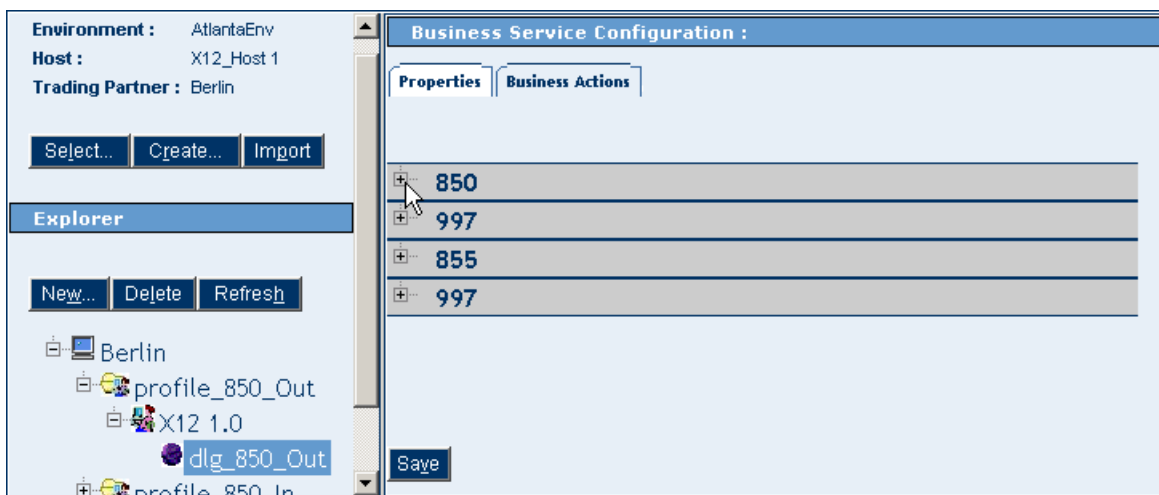
- 1 In the explorer (lower left) side of the ePM screen, open trading partner **Berlin** and successively open **profile_850_Out** and **X12 1.0** and click **dlg_850_Out** to display its business service configuration. See Figure 39.

Figure 39 Properties of Business Service “dlg_850_Out”



- 2 Click the **Business Actions** tab to list the business actions for the selected business service dialog (**dlg_850_Out**). See Figure 40.

Figure 40 Business Actions for Business Service “dlg_850_Out”



- 3 Expand the **850** action to display its values. The values provided with the sample are shown in Table 18. For the **Business Message Syntax Validation Handler**, you *must make the change indicated*. Be sure to click **Save** after you finish.

Table 18 Parameters for Trading Partner “Berlin”, Business Service “dlg_850_Out”

Send To Partner	true
Delivery Channel	BLF_850_Out
Internal Delivery Channel	[None]
Enveloping Channel	EnvelopeChannel
Check for Duplicates	true
Use Encryption	[None]

Table 18 Parameters for Trading Partner “Berlin”, Business Service “dlg_850_Out”

Expect Acknowledgments	[None]
Character Set Encoding	
Send warnings with ACKs	true
Use Signature	[None]
Use Compression	[None]
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Time-Out (Minutes)	10
Business Protocol Validation Handler	bpEX_BusinessValidation_X12 (in SeeBeyond / eX.../ Protocol Managers / X12 Manager /BPs)
Business Message Syntax Validation Handler	<i>Important:</i> You must change this value from [None] to a syntax-validation handler BP appropriate for this action. Example1 (for 850): x12_4010_850_Full_SynValHandler (in SB / eX / User Components / OTD Validat... / X12 / v4010) Example2 (for 270): addenda_hipaa_270_Full_SynValHandler (in SB/eX/User Components/OTD Validat.../HIPAA/2000_A...)
Custom Business Protocol Validation Handler	[None]
Business ACK Generator Handler	[None]
Business ACK Processor Handler	[None]
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]
Error Handler	bpEX_JMSErrorHandler (in SB / eX / User Components / Common / Error / BPs)
Custom External Unique ID Handler	[None]
Custom Business Translation Handler	[None]
GS01 FUNCTIONAL ID CODE	PO
GS02 APPLICATION SENDER CODE	GS:Atlanta
GS03 APPLICATION RCVR CODE	GR:Berlin
GS04 DATE FORMAT	CCYYMMDD
GS05 TIME FORMAT	HHMM
GS07 RESP AGENCY CODE	X
GS08 VERS/REL/INDUST ID CODE	004010X092
Starting Control Number	0

- 4 Repeat step 3 for each of the other actions, making sure you supply an appropriate Business Message Syntax Validation Handler for the action (as well as any optional changes you may want to make) and clicking **Save** when you are finished.

Tip: *Each 997 action uses the same Business Message Syntax Validation Handler as the action immediately preceding it.*

Result: For trading partner “Berlin”, all parameters are now set correctly, and the TP is ready to be activated.

6.3.3 Activating the Trading Partner

To activate the “Berlin” trading partner

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

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

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

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

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

6.3.4 Configuring Trading Partner Parameters for the “Atlanta” TP

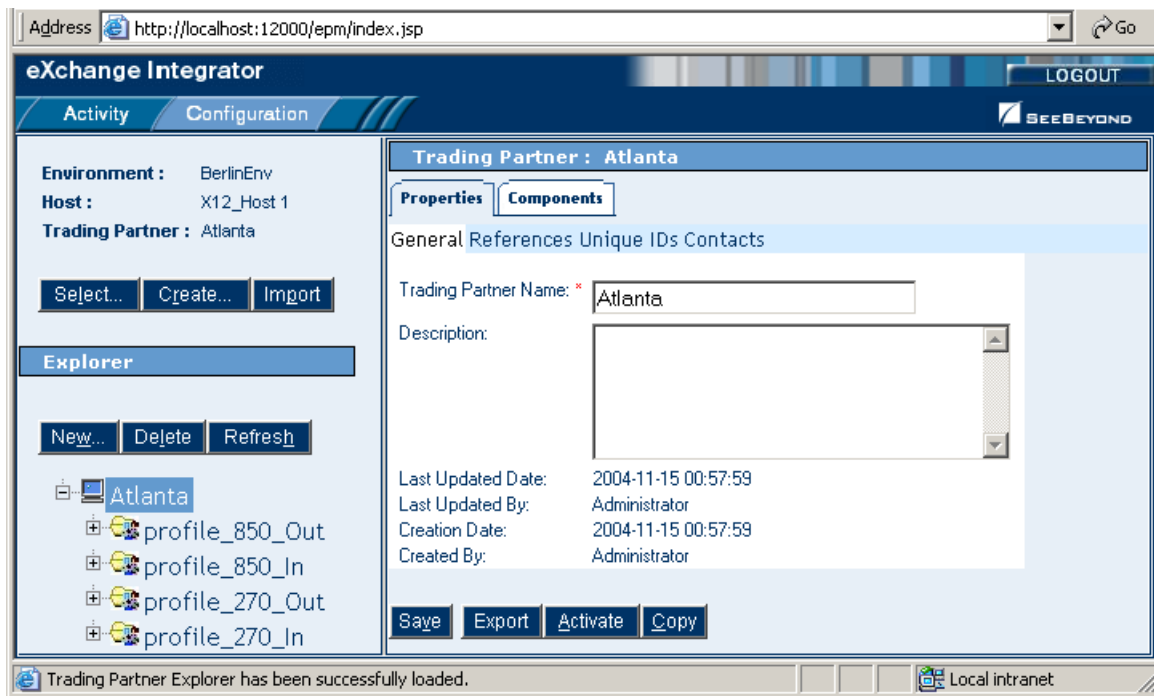
When you imported trading partner “Atlanta” into the Berlin environment, parameter settings were valuated in part based on parameters stored in the export file, and in part based on the name of the trading partner. In this section, you will view, set, and/or update the following parameters for the delivery channel for trading partner “Atlanta”.

To find the “Atlanta” trading partner (in BerlinEnv)

- 1 In the upper left side of the ePM screen, click **Select**.
A new window opens, prompting you to select a B2B host and trading partner.
- 2 Open the B2B Repository and **BerlinEnv** and click **X12_Host1**.
- 3 Click **Search**, and then click **OK**.

Result: In the explorer tree, under BerlinEnv, trading partner **Atlanta** reappears. See Figure 41.

Figure 41 Trading Partner “Atlanta” (in BerlinEnv)



TP “Atlanta”: Parameters for the Delivery Channel

Purpose: To view the parameters governing Berlin’s message exchange with Atlanta.

Keep in mind: You have imported a trading partner that was configured for the Berlin environment, and so you will take the viewpoint of the Berlin B2B host: “ToPartner” means “to Atlanta”; “FromPartner” means “from Atlanta”.

Purpose: To view the parameters governing Atlanta’s message exchange with Berlin.

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

- 1 In the explorer (lower left) side of the ePM screen, click **Atlanta** to display the trading partner’s general properties.
- 2 Click the **Components** tab to display the trading partner’s delivery channel parameters. Table 19.

Table 19 List of Delivery Channels for Trading Partner “Atlanta”

Binding Name	ToPartner Transport Name	FromPartner Transport Name	Packager Name	Comments
xdc_X12_BatchFTP	BatchFTP	ChannelManagerFTP	mad_X12	(not used)
BLF_850_Out	BatchLocalFile	ChannelManagerFile	mad_X12	
BLF_850_In	BatchLocalFile	ChannelManagerFile	mad_X12	
BLF_270_Out	BatchLocalFile	ChannelManagerFile	mad_X12	(optional)
BLF_270_In	BatchLocalFile	ChannelManagerFile	mad_X12	(optional)

- Click one of the binding names, such as **BLF_850_Out**, to display its general properties. See Figure 42.

Figure 42 Delivery Channel for “Atlanta”: General Properties for BLF_850_Out

- Click the **ToPartnerTransport** tab to display its values, especially to verify the target directory, which must match the value of the DataDir environment constant for BerlinEnv.

The values provided with the sample are shown in Table 20. If you need to make any changes, be sure to click **Save** after you finish.

Table 20 ToPartnerTransport Parameters for Trading Partner “Atlanta”

All Purpose End Point	BerlinTo Atlanta
Use Synchronous Channel	false
Track for Auditing	false
Append	[None]
TargetDirectoryName	C:\temp\eXchange\Sample\X12\Data\Berlin
TargetDirectoryNamesPattern	[None]
TargetFileName	Atlanta_<action>_<direction>_ISA_%d_%H%m%s_%S.isa (Example1: For the binding named BLF_850_Out , the filename is Atlanta_850_Out_ISA_%d_%H%m%s_%S.isa (Example2: For the binding named BLF_270_In , the filename is Atlanta_270_In_ISA_%d_%H%m%s_%S.isa)
TargetFileNamesPattern	true
MaxSequenceNumber	
StartingSequenceNumber	
PreDirectoryName	
PreDirectoryNamesPattern	[None]
PreFileName	
PreFileNamesPattern	[None]
PreTransferCommand	

Table 20 ToPartnerTransport Parameters for Trading Partner “Atlanta” (Continued)

PostDirectoryName	
PostDirectoryNameIsPattern	[None]
PostFileName	
PostFileNameIsPattern	[None]
PostTransferCommand	
ResumeReadingEnabled	[None]
ActionOnMalformedCommand	
IncludeOrderRecordInErrorRecord	[None]
IncludePayloadInErrorRecord	[None]
PublishStatusRecordOnError	[None]
PublishStatusRecordOnSuccess	[None]

- Click the **FromPartnerTransport** tab to display its values, especially to verify the target directory, which must match the value of the DataDir environment constant for BerlinEnv.

The values provided with the sample are shown in Table 21. If you need to make any changes, be sure to click **Save** after you finish.

Table 21 FromPartnerTransport Parameters for Trading Partner “Atlanta”

All Purpose End Point	Atlanta To Berlin
Use Synchronous Channel	false
Track for Auditing	false
ChannelManagerMode	FILE
FilePattern	Berlin_<action>_<opposite-direction>_ISA_*.isa (Example: For BLF_850_Out , this is Berlin_850_In_ISA_*.isa)
Directory	C:\temp\Exchange\Sample\X12\Data\Atlanta
PollMilliseconds	10000

- Click the **ToPartnerPackaging** tab to display its values. The values provided with the sample are shown in Table 22. If you need to make any changes, be sure to click **Save** after you finish..

Table 22 ToPartnerPackaging Parameters for Trading Partner “Atlanta”

Encryption Key	[None]
Check for Duplicates	true
Use Encryption	false
Send Acknowledgments	true
Expect Acknowledgments	true
Character Set Encoding	
Send warnings with ACKs	true

Table 22 ToPartnerPackaging Parameters for Trading Partner “Atlanta” (Continued)

Use Signature	false
Use Compression	false
Message Encoding	
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Batcher Handler	bpEX_Batcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Batch Encryption Handler	[None]
Batch Signing Handler	[None]
Batch Compression Handler	[None]
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]

- Click the **FromPartnerPackaging** tab to display its values; the values provided with the sample are shown in Table 23. If you need to make any changes, be sure to click **Save** after you finish..

Table 23 FromPartnerPackaging Parameters for Trading Partner “Atlanta”

Signature Certificate	[None]
Check for Duplicates	true
Use Encryption	false
Send Acknowledgments	true
Expect Acknowledgments	false
Character Set Encoding	
Send warnings with ACKs	true
Use Signature	false
Use Compression	false
Message Encoding	
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Batch Splitter Handler	bpEX_Unbatcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Batch Decryption Handler	[None]
Batch Signature Validation Handler	[None]
Batch Decompression Handler	[None]
Decryption Handler	[None]

Table 23 FromPartnerPackaging Parameters for Trading Partner “Atlanta” (Continued)

Signature Validation Handler	[None]
Decompression Handler	[None]

- Browse other bindings for delivery channel `xdc_X12_BatchLocalFile`, such as `BLF_850_In` (and, if you use them, `BLF_270_Out` and `BLF_270_In`). If you need to make any changes, be sure to click **Save** after you finish.

Result: For trading partner “Atlanta”, the parameters for all bindings for external delivery channel `xdc_X12_BatchLocalFile` are now set correctly, and you are ready to view and/or reconfigure the parameters for the Enveloping Channel.

To view or reconfigure enveloping channel parameters for trading partner “Atlanta”

- In the explorer (lower left) side of the ePM screen, click **Atlanta** to display the trading partner’s general properties.
- In the canvas, click the **Components** tab; click the Enveloping Channels subtab to display the trading partner’s single enveloping channel, and then click its binding name (`EnvelopeChannel`) to display general parameters for this envelope channel.
- Click the **ToPartner Envelope Binding** tab to display its values. The values provided with the sample are shown in Table 24. If you need to make any changes, be sure to click **Save** after you finish.

Table 24 ToPartner Enveloping Parameters for Trading Partner “Atlanta”

Release Quantity	1
Release Scheduler String	50 * * ? * *
Time-Out (Minutes)	10
Max Retry Count	1
Batcher Handler	bpEX_Batcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]
ISA01 AUTHOR INFO QUAL	00
ISA02 AUTHOR INFORMATION	
ISA03 SEC INFO QUAL	00
ISA04 SECURITY INFORMATION	
ISA05 IC SENDER ID QUAL	01
ISA06 INTERCHANGE SENDER ID	SID:Berlin
ISA07 IC RCVR ID QUAL	01
ISA08 INTERCHANGE RCVR ID	RIDAtlanta
ISA11 IC CONTROL STANDARD ID	U
ISA12 IC VERSION NUMBER	00401
ISA13 IC CONTROL NUMBER	0

Table 24 ToPartner Enveloping Parameters for Trading Partner “Atlanta” (Continued)

ISA14 ACKNOWLEDGMENT REQUESTED	1
ISA15 USAGE INDICATOR	T
ISA16 COMP ELE SEP	:
SEGMENT TERMINATOR	~
ELEMENT SEPARATOR	*
GS06 GROUP CONTROL NUM	0

- 4 Click the **FromPartner DeEnvelope Binding** tab to display its values. The values provided with the sample are shown in Table 17. If you need to make any changes, be sure to click **Save** after you finish.

Table 25 FromPartner Enveloping Parameters for Trading Partner “Atlanta”

Batcher Splitter Handler	bpEX_Unbatcher_X12 (in SB > eX > Protocol Managers > X12 Manager > BPs)
Decryption Handler	[None]
Signature Validation Handler	[None]
Deompression Handler	[None]
ISA01 AUTHOR INFO QUAL	00
ISA02 AUTHOR INFORMATION	
ISA03 SEC INFO QUAL	00
ISA04 SECURITY INFORMATION	
ISA05 IC SENDER ID QUAL	01
ISA06 INTERCHANGE SENDER ID	SIDAtlanta
ISA07 IC RCVR ID QUAL	01
ISA08 INTERCHANGE RCVR ID	RID:Berlin
ISA11 IC CONTROL STANDARD ID	U
ISA12 IC VERSION NUMBER	00401
ISA13 IC CONTROL NUMBER	0
ISA14 ACKNOWLEDGMENT REQUESTED	1
ISA15 USAGE INDICATOR	T

Result: For trading partner “Atlanta”, the parameters for delivery and enveloping channels are now set correctly, and you are ready to view and/or reconfigure each of the actions in each of the business services.

To configure trading partner “Atlanta”, business service “dlg_850_Out” (required steps)

- 1 In the explorer (lower left) side of the ePM screen, open trading partner **Atlanta** and successively open **profile_850_Out** and **X12 1.0** and click **dlg_850_Out** to display its business service configuration.
- 2 Click the **Business Actions** tab to list the business actions for the selected business service dialog (**dlg_850_Out**).

- Expand the **850** action to display its values. The values provided with the sample are shown in Table 26. For the **Business Message Syntax Validation Handler**, you *must make the change indicated*. Be sure to click **Save** after you finish.

Table 26 Parameters for Trading Partner “Atlanta”, Business Service “dlg_850_Out”

Send To Partner	true
Delivery Channel	BLF_<action>_<direction> (Example1: For the service dlg_850_Out, this is BLF_850_Out) (Example2: For the service dlg_270_In, this is BLF_270_In)
Internal Delivery Channel	[None]
Enveloping Channel	EnvelopeChannel
Check for Duplicates	true
Use Encryption	[None]
Expect Acknowledgments	[None]
Character Set Encoding	
Send warnings with ACKs	true
Use Signature	[None]
Use Compression	[None]
Batch Tracking	Both
Group Tracking	Both
Transaction Tracking	Both
Time-Out (Minutes)	10
Business Protocol Validation Handler	bpEX_BusinessValidation_X12 (in SeeBeyond / eX.../ Protocol Managers / X12 Manager / BPs)
Business Message Syntax Validation Handler	<i>Important:</i> You must change this value from [None] to a syntax-validation handler BP appropriate for this action. Example1 (for 850): x12_4010_850_Full_SynValHandler (in SB / eX / User Components / OTD Validat... / X12 / v4010) Example2 (for 270): addenda_hipaa_270_Full_SynValHandler (in SB/eX/User Components/OTD Validat.../HIPAA/2000_A...)
Custom Business Protocol Validation Handler	[None]
Business ACK Generator Handler	[None]
Business ACK Processor Handler	[None]
Encryption Handler	[None]
Signing Handler	[None]
Compression Handler	[None]
Error Handler	bpEX_JMSErrorHandler (in SB / eX / User Components / Common / Error / BPs)
Custom External Unique ID Handler	[None]
Custom Business Translation Handler	[None]

Table 26 Parameters for Trading Partner “Atlanta”, Business Service “dlg_850_Out”

GS01 FUNCTIONAL ID CODE	PO
GS02 APPLICATION SENDER CODE	GS:Atlanta
GS03 APPLICATION RCVR CODE	GR:Berlin
GS04 DATE FORMAT	CCYYMMDD
GS05 TIME FORMAT	HHMM
GS07 RESP AGENCY CODE	X
GS08 VERS/REL/INDUST ID CODE	004010X092
Starting Control Number	0

- Repeat step 3 for each of the other actions, making sure you supply an appropriate Business Message Syntax Validation Handler for the action (as well as any optional changes you may want to make) and clicking **Save** when you are finished.

Tip: Each 997 action uses the same Business Message Syntax Validation Handler as the action immediately preceding it.

- Repeat steps 3 and 4 for each of the other business services, such as **dlg_850_In** (as well as **dlg_270_Out** and **dlg_270_In** if you use them), making sure you supply an appropriate Business Message Syntax Validation Handler for each action in each business service. Be sure to click **Save** after finishing each change.

Result: For trading partner “Atlanta”, all parameters are now set correctly, and the TP is ready to be activated.

6.3.5 Activating the Trading Partner

To activate the “Atlanta” trading partner

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

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

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

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

Result: Both trading partners are entirely complete and ready to be run.

6.4 Runtime Steps

In this section

For the X12 sample implementation, runtime steps consist of the following:

- [Starting the Logical Hosts on page 99](#)
- [Supplying the Input Data on page 100](#)
- [Using the Message Tracking Facility on page 101](#)

6.4.1 Starting the Logical Hosts

These steps assume you have already installed two or more logical hosts. The directory for the logical host that will run Atlanta is called *<AtlantaLH>*, and the directory for the logical host that will run Berlin is called *<BerlinLH>*.

To bootstrap the logical host

- 1 Open a command prompt and change directories to the location of your logical host's bootstrap executables. For example:

```
cd <AtlantaLH>\bootstrap\bin
```

- 2 Start the bootstrap script using appropriate parameters. For example:

```
bootstrap -r http://myBox:12345/myRepository -i myId -p myPassword  
-e AtlantaEnv -l LogicalHost1 -f
```

- For the **-r** (repository) parameter), supply the correct URL with repository name.
- For the **-i** and **-p** (ID and password) parameters, supply the appropriate values.
- For **-e** (environment) parameter, use: **AtlantaEnv**
- For **-l** (logical host name) parameters, use: **LogicalHost1**
- You need not supply the **-f** (force) flag if all configuration changes have already been applied, or if this is the first time that bootstrap has been run.

Result: After a time, the logical host starts running, and all activated projects that reference AtlantaEnv are automatically applied to it.

- 3 Repeat steps 1 and 2 on a different logical host, referencing the same repository but pointing it at the **BerlinEnv** environment. For example:

```
cd <BerlinLH>\bootstrap\bin  
bootstrap -r -f[...] -e BerlinEnv -l LogicalHost1
```

Optional: To apply environment changes when a logical host is running

Purpose: If changes are made to parameters in an environment component while a logical host is running—for example, adding or modifying keystores for an eXchange service—use these steps to apply the changes without having to shutdown and re-bootstrap the logical host. This is the equivalent of shutting down the logical host and re-bootstrapping it using the **-f** flag.

- 1 In Enterprise Designer, in Environment Explorer, open the environment where the changes have occurred.

- 2 Right-click the logical host that is running and, on the popup menu, click **Apply**
- 3 Repeat the previous step as needed for other logical hosts in the same environment.

Result: In the back end, a “mini-shutdown/mini-rebootstrap” occurs, and the changes are applied to the running logical host.

6.4.2. Supplying the Input Data

Before you begin: Your logical hosts must both be running and the eXchange 5.0.5 Oracle database must be accessible.

The scenarios are set up so that the project variable **Data Root Directory** points at C:\temp\exChange\Sample\Data\X12_Atlanta\ for the Atlanta host, but points at C:\temp\exChange\Sample\Data\X12_Berlin\ for the Berlin host. Therefore:

- The Atlanta project ...
 - ♦ Reads files of this form:
... \Sample\X12\Data\Atlanta\X12_outbound_feeder_Berlin_profile*.in

(The .in file is an XML file that is hardcoded to point at a directory named C:\temp\exChange\Sample\Data\X12\Atlanta and file named X12_270_incremented.st; if you are using a different directory, make appropriate changes.)
 - ♦ Writes output resulting from successful processing to files of this form:
... \Sample\X12\Data\Berlin\X12_Atlanta_<timestamp>.dat
 - ♦ Writes error messages to files of this form:
... \Sample\X12\Data\Atlanta\ErrorTopic_%d.dat
 - ♦ Writes error payloads to files of this form:
... \Sample\X12\Data\Atlanta\ErrorTopic_payload_%d.dat
- The Berlin project ...
 - ♦ Reads files of this form:
... \Sample\X12\Data\Berlin\X12_271*.st
 - ♦ Writes output resulting from successful processing to files of this form:
... \Sample\X12\Data\Atlanta\X12_Berlin_<timestamp>.dat
 - ♦ Writes error messages to files of this form:
... \Sample\X12\Data\Berlin\ErrorTopic_%d.dat
 - ♦ Writes error payloads to files of this form:
... \Sample\X12\Data\Berlin\ErrorTopic_payload_%d.dat

To feed input data to the Atlanta project

- 1 In C:\temp\exChange\Sample\X12\Data\Atlanta\, rename file X12_270_Outbound_feeder_Berlin_profile.~in to X12_270_Outbound_feeder_Berlin_profile.in
- 2 Watch the file extension change from .in to .~in as the file is picked up.

To feed input data to the Berlin project

- 1 In C:\temp\exchange\Sample\X12\Data\Berlin\, rename files X12_271*.* to X12_271*.st
- 2 Watch the file extensions change from .st to .* as the files are picked up.

6.4.3. Using the Message Tracking Facility

Before you begin: Your logical hosts must both be running and the eXchange 5.0.5 Oracle database must be accessible.

To access Message Tracking

- 1 Open a *new* browser session. (In other words, do *not* clone a new window from an existing session).
- 2 Point your browser at the URL for message and package tracking. This takes the following form:

http://<loghostname>:<port>/<AppName>/msgTrack/EnterPkgTrack.do

where:

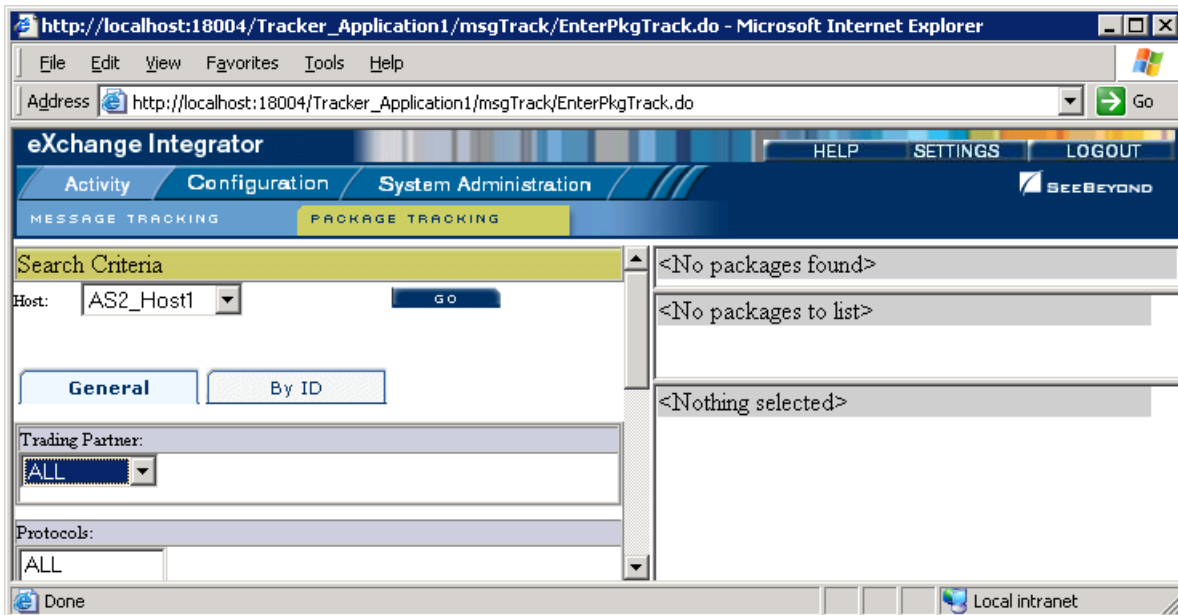
- ♦ <loghostname> is the hostname of the physical machine where you ran bootstrap.exe and pointed it at a particular environment and logical host.
- ♦ <port> is the port number assigned in that environment and logical host (in the configuration for Integration Server .> ... > Web server).
- ♦ <TrackerAppName> is the name of the Message Tracking application as it appears in the connectivity map. If there is only one such instance, the default name is **Tracker_Application1**

Examples:

- To access message tracking for Atlanta (IS ports 18000–18009, web port=18004), for a logical host bootstrapped on a physical machine whose hostname is “Dixie”:
http://Dixie:**18004**/Tracker_Application1/msgTrack/EnterPkgTrack.do
- To access message tracking for Berlin (IS ports 28000–28009, web port=28004), if the logical host was bootstrapped locally:
http://localhost:**28004**/Tracker_Application1/msgTrack/EnterPkgTrack.do
- To access message tracking for an Integration Server that was customized to use a special SSL-only web port=21444, if the logical host was bootstrapped on a physical machine whose IP address is 10.18.5.5:
http://10.18.5.5:**21444**/Tracker_Application1/msgTrack/EnterPkgTrack.do

Result: See Figure 43.

Figure 43 Message Tracking, on Startup



To search by B2B host, trading partner, and protocol

- 1 Under Search Criteria, use the Host drop-down list to choose the B2B host whose messages you want to examine (in this case, **X12_Host1**), and click **GO**.
- 2 Under Trading Partner, either click **ALL** or choose a particular trading partner (such as **Berlin**) from the drop-down list.
- 3 Under Protocols, either click **ALL** or choose **X12** from the drop-down list.
- 4 At the lower left of the window, click **SEARCH**.

Result: The canvas (right side), under Search Results, displays a page containing the Package IDs of the latest ten tracked messages fitting the criteria you specified. Navigation links (Previous, Next, and Go to Page) allow you to see other pages of ten results each. See Figure 44.

Figure 44 Message Tracking, Showing Initial Search Results

The screenshot shows the Message Tracking application interface. The top bar has 'MESSAGE TRACKING' and 'PACKAGE TRACKING' tabs. Below the tabs, there are sections for 'Search Criteria' and 'Search Results'.

Search Criteria:

- Host: AS2_Host1
- Trading Partner: Berlin
- Protocols: AS2 (selected)
- Filters: Error Type: ALL, Direction: Both, Date Filter: Process date

Search Results:

Package ID	Trading Partner	Protocol	Package Conversation Type ID	Direction	Error Data	Process Date	Response Req	Ack Date
<20040605004911969@Berlin_Host>	Berlin	AS2		Inbound	No	06/05/2004 00:48:35	No	
<20040605004907723@Berlin_Host>	Berlin	AS2		Inbound	No	06/05/2004 00:48:33	No	
<20040605004822359@Atlanta_Host>	Berlin	AS2		Outbound	No	06/05/2004 00:48:24	Yes	06/05/2004 00:48:36
<20040605004822437@Atlanta_Host>	Berlin	AS2		Outbound	No	06/05/2004 00:48:23	Yes	06/05/2004 00:48:33
<20040605004326232@Berlin_Host>	Berlin	AS2		Inbound	No	06/05/2004 00:43:09	No	
<20040605004119906@Atlanta_Host>	Berlin	AS2		Outbound	No	06/05/2004 00:41:35	Yes	06/05/2004 00:43:20
<20040604191043487@Berlin_Host>	Berlin	AS2		Inbound	No	06/04/2004 19:09:35	No	
<20040604190931328@Atlanta_Host>	Berlin	AS2		Outbound	No	06/04/2004 19:09:31	Yes	
<20040604183039911@Berlin_Host>	Berlin	AS2		Inbound	No	06/04/2004 18:29:32	No	
<20040604182911140@Atlanta_Host>	Berlin	AS2		Outbound	No	06/04/2004 18:29:17	Yes	06/04/2004 18:29:35

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

Purpose: After performing a search, or after setting up a search using the previous procedure, you can specify one or more further criteria.

- Near the bottom of the left pane, under **Filters**, specify one or more of the following:
 - For **Error Type**: If you do not choose ALL, you can restrict your search either to display error messages only, or to display non-error messages only.
 - For **Direction**: If you do not choose ALL, you can restrict your search either to display inbound messages only, or to display outbound messages only.
 - For **Date**: You can choose to include only those messages whose *processing* date lies within a range you specify, or only those messages whose *acknowledgment* date lies within the range.
- At the lower left of the window, click **SEARCH**.

Result: The canvas displays a page containing the Package IDs of the latest ten tracked messages fitting the criteria you specified. See Figure 45.

Figure 45 Message Tracking, Showing Filtered Results

The screenshot shows a web-based interface for message tracking. On the left, the 'Search Criteria' section includes a 'Host' dropdown set to 'AS2_Host1' and a 'GO' button. Below this are tabs for 'General' and 'By ID'. The 'Trading Partner' is set to 'Berlin'. Under 'Protocols', a list includes 'ALL', 'ebxml', 'AS2' (which is selected), and 'EmptyMAD'. The 'Filters' section has 'Error Type' set to 'Error', 'Direction' to 'Inbound', 'Date Filter' to 'Acknowledgement date', and 'From and To' set to 'Process date' and 'Acknowledgement date'. 'SEARCH' and 'CLEAR' buttons are at the bottom of the criteria section.

The 'Search Results' section on the right shows a table with the following data:

Package ID	Trading Partner	Protocol Type	Conversation ID	Direction	Error Data	Process Date	Response Req	Ack Date
<20040603155546140@Berlin_Host>	Berlin	AS2		Inbound	Yes	06/03/2004 15:56:52	No	06/03/2004 15:57:13
<20040603155546125@Berlin_Host>	Berlin	AS2		Inbound	Yes	06/03/2004 15:56:53	No	06/03/2004 15:57:13
<20040603161205390@Berlin_Host>	Berlin	AS2		Inbound	Yes	06/03/2004 16:12:07	No	06/03/2004 16:12:08
<20040603181209546@Berlin_Host>	Berlin	AS2		Inbound	Yes	06/03/2004 18:12:11	No	06/03/2004 18:12:11

Below the table, there is a message '<Nothing selected>'.

To obtain details of a specified package

Purpose: On a package-by-package basis, you can examine the text of either the original message or the acknowledgment.

- 1 After obtaining results from a search using any of the procedures mentioned earlier, click the package ID for any of the returned results.
- 2 In the “Details for package <package-ID>” pane, click **Open** to see the contents (possibly encrypted) of either the original message or the acknowledgment.

Result: See Figure 46. You can use cut, copy, and paste on any text in the window.

Figure 46 Message Tracking, Showing Package Details and Message Content

The screenshot shows the 'Search Results' page with a table of message tracking data. The first row is highlighted, and its package ID is circled in green. A green arrow points from this package ID to the 'Open' button in the message details view, which is also circled in green. The message content is displayed in a separate window titled 'Message View - Microsoft Internet Explorer, provided by SeeBeyond'. The content includes transaction details and an EDI message body.

Search Results

Print << Previous Page 1 of 1 Next >> Go to Page: 1 GO

Package ID	Trading Partner	Protocol	Package Type	Conversation ID	Direction	Error Data	Process Date	Response Req	Ack Date
<20040611165132643@Berlin_Host>	Berlin_hluxp	AS2			Outbound No		06/11/2004 17:01:31	Yes	

Step 1: Select Package ID.

Details for package: <20040611165132643@Berlin_Host>

Message Attributes Errors Dialog

Following are the messages available to the selected item:

Original message: **Step 2: Click Open.**

Result is displayed in a new window.

Message View - Microsoft Internet Explorer, provided by SeeBeyond

File Edit View Favorites Tools Help

Message for Transaction ID:
<20040611165132643@Berlin_Host>

Content-Type: multipart/signed; protocol="application/pkcs7-sig boundary="-----_Part_0_255376287.1086998019233"

-----_Part_0_255376287.1086998019233

Content-Type: Application/EDI-X12

ISA*00*TEST060804*00*TEST060804*ZZ*Atlanta *ZZ*Berlin
*961007*2013*U*00200*000000001*0*T*::~GS*PO*S1S1S1S1S1S1S1S1S1R1R
Number~PER*AA*Hans Gutten*CP*1.322.323.4444*****rgg4egv4t4~TA
aaa*1000000*AS*90.00*BD*AK*234235v3534q6f3534v4353453vq3q32***
aaa*1000000*AS*90.00*BD*AK*234235v3534q6f3534v4353453vq3q32***
aaa*1000000*AS*90.00*BD*AK*234235v3534q6f3534v4353453vq3q32***
aaa*1000000*AS*90.00*BD*AK*234235v3534q6f3534v4353453vq3q32***

----- Part 0 255376287.1086998019233

Glossary of Acronyms

AS2

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

BAD

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

B2B

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

DCP

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

EAD

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

ebXML

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

ePM

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

FTP

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

HTTP

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

ICAN

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

IDC

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

MAD

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

MIME

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

OTD

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

SME

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

S/MIME

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

SMTP

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

TAD

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

TP, TPP

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

URL

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

XDC

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

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

XML

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

OTD Syntax Validation Handlers

This appendix provides background and conceptual information on OTD syntax validation handler BPs. This information in this appendix complements the hands-on exercise provided in [Chapter 5](#).

What's in This Appendix

- [Activity Flow](#) on page 109
- [Fault Handling](#) on page 112
- [Variables Referenced by OTD Validation Handler BPs](#) on page 114

A.1 Activity Flow

The activity flow of an OTD syntax validation handler business process consists of the following steps: (1) Receive input; (2) Copy **ExStdEvent**; (3) Concatenate the payload's headers/data/trailer into a string; (4) Unmarshal the concatenated string; (5) Populate **BizAckCorrKey**; (6) Set **BizRespCorrKey**; (7) Perform validate; (8) Check results.

These steps are discussed in detail below.

- 1 The BP receives the inbound message data from its invoker.
- 2 The BP copies **ExStdEvent** from inbound to outbound: In other words, copying the entire content of the **ExStdEvent** portion of the inbound data into the **ExStdEvent** portion of the outbound data for the handler.
- 3 The BP concatenates the following headers, data, and trailers, in order, from the **PayloadSection** part of the inbound data's **ExStdEvent**, and then copying this concatenated string into the contents part of input of **unmarshal**:
 - A Envelopes/BusinessProtocol/Batch/**Header**
 - B Envelopes/BusinessProtocol/Group/**Header**
 - C Envelopes/BusinessProtocol/Group/**Header**
 - D **RawData**
 - E Envelopes/BusinessProtocol/Group/**Trailer**
 - F Envelopes/BusinessProtocol/Batch/**Trailer**

- 4 The BP unmarshals the input string constructed in step 3.
If the unmarshaling process throws an `UnmarshalException` or `GenericException`, the exceptions are handled by fault handlers. See [“Fault Handling” on page 112](#).
- 5 The BP populates **BizAckCorrKey** in the following four sub-steps:
 - ♦ For the `ExStdEvent` part of the Handler's outbound data: Populate the `KeysSection/CorrelationKeys/`**BizAckCorrelationKey** with a string formed by concatenating by the following components, in order:
 - A `PayloadSection/KeysSection/InternalIDs/ExTradingPartnerGUID`
(this comes from the `ExStdEvent` part of inbound data).
 - B |
(the pipe character)
 - C `PayloadSection/MetaDataSection/Event/`**BusinessProtocolName**
(this comes from the `ExStdEvent` part of inbound data)
 - D |
(the “pipe” character)
 - E `${OuterNodeName}/${InnerNodeName}[1]/${TransNodeName}[1]/`
`${TSHeaderNodeName}/${BusinessTransactionIdentifier}`
(this comes from the unmarshaled OTD).
 - F |
(the “pipe” character)
 - G `${OuterNodeName}/${InnerNodeName}[1]/${FGHeaderNodeName}/`
`${FGCtrlNumNodeName}`
(this comes from the unmarshaled OTD)
 - H |
(the “pipe” character)
 - I `${OuterNodeName}/${InnerNodeName}[1]/${TransNodeName}[1]/`
`${TSHeaderNodeName}/${TransactionExternalID}`
(this comes from the unmarshaled OTD)
 - ♦ Assign `${OuterNodeName}/${InnerNodeName}[1]/${TransNodeName}[1]/`
`${TSHeaderNodeName}/${BusinessTransactionIdentifier}` (from the unmarshaled OTD) to `MetaDataSection/Event/`**BusinessTransactionIdentifier** (of the `ExStdEvent` part of Handler's outbound data)
 - ♦ Assign `${OuterNodeName}/${InnerNodeName}[1]/${TransNodeName}[1]/`
`${TSHeaderNodeName}/${TransactionExternalID}` (from the unmarshaled OTD) to `KeysSection/ExternalIDs/`**TransactionExternalID** (of the `ExStdEvent` part of Handler's outbound data)
 - ♦ Assign unmarshaled OTD to the OTD part of the validate input.
- 6 At this point, are two possibilities for setting **BizRespCorrKey**, depending on whether the inbound data is a request or a response. This is determined by the `MetaDataSection/Event/BusinessTransactionType` (of the `ExStdEvent` part of inbound data).

- ◆ In the case of a *request* (that is, the BusinessTransactionType is 'Request'), then the BP sets BizRespCorrKey using BizTxID by populating KeysSection/CorrelationKeys/**BizResponseCorrelationKey** (of the ExStdEvent part of the Handler's outbound data) with a string formed by concatenating the following components, in order:
 - A KeysSection/InternalIDs/**ExTradingPartnerGUID**
(from the ExStdEvent part of inbound data).
 - B |
(the “pipe” character)
 - C MetaDataSection/Event/**BusinessProtocolName**
(from the ExStdEvent part of inbound data)
 - D |
(the “pipe” character)
 - E MetaDataSection/Event/**BusinessTransactionIdentifier**
(from the ExStdEvent part of inbound data)
 - F |
(the “pipe” character)
 - G \${OuterNodeName}/\${InnerNodeName}[1]/\${TransNodeName}[1]/
\${BizRespCorrPath}
(from the unmarshaled OTD)
 - ◆ In the case of a *response* or *reply* (that is, the BusinessTransactionType is *not* 'Request'), then the BP sets BizRespCorrKey using BizTxID by populating the KeysSection/CorrelationKeys/**BizResponseCorrelationKey** (of the ExStdEvent part of the Handler's outbound data) with a string string formed by concatenating the following components, in order:
 - A KeysSection/InternalIDs/**ExTradingPartnerGUID**
(from the ExStdEvent part of inbound data)
 - B |
(the “pipe” character)
 - C MetaDataSection/Event/**BusinessProtocolName**
(from the ExStdEvent part of inbound data)
 - D |
(the “pipe” character)
 - E MetaDataSection/Event/**ProtocolRespondToMessageID**
(from the ExStdEvent part of inbound data)
 - F |
(the “pipe” character)
 - G \${OuterNodeName}/\${InnerNodeName}[1]/\${TransNodeName}[1]/
\${BizRespCorrPath}
(from the unmarshaled OTD)
- 7 The BP performs validate operation.

- 8 The BP checks the output of the validate operation and acts based on its severity. There are three cases: error, warning, or no problem.
 - ◆ In the case of an *error* (in other words, when the contents of the output includes the string `<Severity>ERROR</Severity>`): The BP copies the validate output's contents into the message part of the validate fault, and throws this populated fault. The validate fault is then handled by the fault handler. See [“ValidateException” on page 112](#).
 - ◆ In the case of a *warning only* (that is, the output does not contain the string `<Severity>ERROR</Severity>` but is found to contain the string `<Severity>WARN</Severity>`), the BP populates the following fields in the ExStdEvent part of Handler's output before the Handler returns the populated ExStdEvent part to its invoker:
 - ◆ It assigns contents (from the validate output) to ErrorSection/ExException/ExceptionDetails/OTDParsingErrors/**ParsingErrorsXML**
 - ◆ It assigns `'${OtdName}'` to ErrorSection/ExException/ExceptionDetails/OTDParsingErrors/**OTDIdentifier**
 - ◆ It assigns 'BusinessMessageSyntaxValidation results' to ErrorSection/ExException/ExceptionDetails/**ErrorDescription**
 - ◆ It assigns 'OtdErrors' to ErrorSection/ExException/ExceptionDetails/**ErrorCode**
 - ◆ It assigns 'WARN' to ErrorSection/ExException/ExceptionDetails/**ErrorSeverity**
 - ◆ In the case of *no problem* (that is, the output does not contain either string `<Severity>ERROR</Severity>` or `<Severity>WARN</Severity>`): The BP returns to its invoker. Some fields of the ExStdEvent part have been populated already, as noted in previous steps.

A.2 Fault Handling

This section takes a closer look at the Fault Handling activity flow mentioned in steps 4 and 8 in the previous section.

In this section

- [ValidateException](#) on page 112
- [UnmarshalException](#) on page 113
- [GenericException](#) on page 113
- [Other Faults](#) on page 114

A.2.1 ValidateException

If a `ValidateException` (validate fault) is thrown during the validate operation, then the following additional assignments are performed to populate certain fields in the

ExStdEvent part of the handler's output before the handler returns the populated ExStdEvent part to its invoker:

- The message part of the validate fault is assigned to ErrorSection/ExException/ExceptionDetails/OTDParsingErrors/**ParsingErrorsXML**
- The variable **#{OtdName}** is assigned to ErrorSection/ExException/ExceptionDetails/OTDParsingErrors/**OTDIdentifier**
- The string literal '**BusinessMessageSyntaxValidation results**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorDescription**
- The string literal '**OtdErrors**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorCode**
- The string literal '**ERROR**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorSeverity**

A.2.2 UnmarshalException

If an UnmarshalException is thrown during the unmarshal operation, then the following additional assignments are performed to populate certain fields in the ExStdEvent part of the handler's output before the handler returns the populated ExStdEvent part to its invoker:

- The message part of the unmarshal fault is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorDescription**
- The string literal '**com.stc.otd.runtime.UnmarshalException**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorCode**
- The string literal '**ERROR**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorSeverity**

A.2.3 GenericException

If a GenericException is thrown during the unmarshal operation, then the following additional assignments are performed to populate certain fields in the ExStdEvent part of the handler's output before the handler throws the populated output as a fault:

- The message part of the unmarshal fault is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorDescription**
- The string literal '**Unknown Exception from otd unmarshal**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorCode**
- The string literal '**ERROR**' is assigned to ErrorSection/ExException/ExceptionDetails/**ErrorSeverity**

After the assignments are made, the BP throws the populated output as a fault. It is expected that a GenericException fault will be handled by the handler's invoker.

A.2.4 Other Faults

If another fault is thrown but not caught as a `ValidateException`, `UnmarshalException`, or `GenericException`, then the following additional assignments are performed to populate certain fields in the `ExStdEvent` part of the Handler's output before the Handler throws the populated output as a fault.

- The string literal '**Unknown Error(s) Occurred in the OTD message syntax validation handler**' is assigned to `ErrorSection/ExException/ExceptionDetails/ErrorDescription`
- The string literal '**Unknown Errors**' is assigned to `ErrorSection/ExException/ExceptionDetails/ErrorCode`
- The string literal '**ERROR**' is assigned to `ErrorSection/ExException/ExceptionDetails/ErrorSeverity`

A.3 Variables Referenced by OTD Validation Handler BPs

The following variables are used. (This example uses an 850 of X12 version 4010.)

Table 27 Variables Referenced by OTD Validation Handler BPs

Variable Name	Assignment (using X12 v4010 850 as an example)
<code>\${OtdName}</code>	x12_4010_850_PurcOrde_Full
<code>\${HandlerName}</code>	x12_4010_850_Full_SynValHandler
<code>\${OuterNodeName}</code>	X12_4010_850_PurcOrde_Outer
<code>\${InnerNodeName}</code>	X12_4010_850_PurcOrde_Inner
<code>\${TransNodeName}</code>	X12_4010_850_PurcOrde
<code>\${TSHeaderNodeName}</code>	ST_1_TransSetHead
<code>\${BusinessTransactionIdentifier}</code>	E143_1_TransSetIdenCode
<code>\${TransactionExternalID}</code>	E329_2_TransSetContNumb
<code>\${FGHeaderNodeName}</code>	GS_FuncGrouHead
<code>\${FGCtrlNumNodeName}</code>	E28_6_GrouContNumb
<code>\${BizRespCorrPath}</code>	BEG_2_BegiSegmForPurcOrde/E324_3_PurcOrdeNumb
<code>\${BizRespCorrPath}</code>	(see below)

A.3.1 The Value of the `${BizRespCorrPath}` Variable

The value of the `${BizRespCorrPath}` variable is obtained by looking up mapping tables with entries of *key=value* where *key* is the OTD name and *value* is the lookup value.

If lookup does not retrieve a value from mapping tables, then a default value is supplied based on the protocol is X12 or EDIFACT, as described below.

Default Value for $\${BizRespCorrPath}$ in X12 v4010

Mapping table entries:

- x12_4010_270_EligCoveOrBeneInqu_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4010_271_EligCoveOrBeneInfo_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4010_850_PurcOrde_Full=BEG_2_BegiSegmForPurcOrde/
E324_3_PurcOrdeNumb
- x12_4010_855_PurcOrdeAckn_Full=BAK_2_BegiSegmForPurcOrdeAckn/
E324_3_PurcOrdeNumb
- x12_4010_997_FuncAckn_Full=ST_1_TransetHead/E329_2_TransetContNumb

Default value: ST_1_TransetHead/E329_2_TransetContNumb

Default Value for $\${BizRespCorrPath}$ in X12 v4030

Mapping table entries:

- x12_4030_270_EligCoveOrBeneInqu_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4030_271_EligCoveOrBeneInfo_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4030_850_PurcOrde_Full=BEG_2_BegiSegmForPurcOrde/
E324_3_PurcOrdeNumb
- x12_4030_855_PurcOrdeAckn_Full=BAK_2_BegiSegmForPurcOrdeAckn/
E324_3_PurcOrdeNumb
- x12_4030_997_FuncAckn_Full=ST_1_TransetHead/E329_2_TransetContNumb

Default value: ST_1_TransetHead/E329_2_TransetContNumb

Default Value for $\${BizRespCorrPath}$ in X12 v4061

Mapping table entries:

- x12_4061_270_EligCoveOrBeneInqu_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4061_271_EligCoveOrBeneInfo_Full=BHT_2_BegiOfHierTran/
E127_3_RefeIden
- x12_4061_850_PurcOrde_Full=BEG_2_BegiSegmForPurcOrde/
E324_3_PurcOrdeNumb
- x12_4061_855_PurcOrdeAckn_Full=BAK_2_BegiSegmForPurcOrdeAckn/
E324_3_PurcOrdeNumb

- `x12_4061_997_FuncAckn_Full=ST_1_TransetHead/E329_2_TransetContNumb`

Default value: `ST_1_TransetHead/E329_2_TransetContNumb`

Default Value for `#{BizRespCorrPath}` in Other X12 Versions

Default value: `ST_1_TransetHead/E329_2_TransetContNumb`

Default Value for `#{BizRespCorrPath}` in HIPAA Addenda

Mapping table entries:

- `x12_004010X092A1_00_hipaa_270_EligCoveOrBeneInqu_Full=BHT_msk1_2_BegiOfHierTran/E127_3_RefIden`
- `x12_004010X092A1_00_hipaa_271_EligCoveOrBeneInfo_Full=BHT_msk1_2_BegiOfHierTran/E127_3_RefIden`

Default value: `ST_1_TransetHead/E329_2_TransetContNumb`

Default Value for `#{BizRespCorrPath}` in HIPAA Standard

Mapping table entries:

- `x12_004010X092_00_hipaa_270_EligCoveOrBeneInqu_Full=BHT_msk1_2_BegiOfHierTran/E127_3_RefIden`
- `x12_004010X092_00_hipaa_271_EligCoveOrBeneInfo_Full=BHT_msk1_2_BegiOfHierTran/E127_3_RefIden`

Default value: `ST_1_TransetHead/E329_2_TransetContNumb`

Default Value for `#{BizRespCorrPath}` in UN/EDIFACT v3 and v4

Default value: `BGM_2_BegiOfMess/C106_2_DocuIden/E1004_1_DocuIden`

Default Value for `#{BizRespCorrPath}` in EANCOM v3 and v4

Default value: `BGM_2_BegiOfMess/C106_2_DocuIden/E1004_1_DocuIden`

Index

A

Accredited Standards Committee 14
 acknowledgments
 application 26
 as part of EDI logic 26
 functional acknowledgment (997) 26
 interchange acknowledgment (TA1) 25
 receipt of payment order 26
 types of 25
 American National Standards Institute 14
 ANSI 14
 application acknowledgments 26
 ASC 14

B

backward compatibility 22

C

CodeGenException (JNDI name prefix) 73, 74
 configuring
 HTTPS 65
 Oracle eWay 65, 67
 ports 65, 66
 control numbers 21
 functional group control number (GS06) 22
 interchange control number (ISA13) 22
 transaction set control number (ST02) 22
 conventions, document 11

D

data element separator 17
 data elements 16
 Data Interchange Standards Association 14
 database for eInsight engine
 supported levels 36
 database for eXchange
 supported levels 36
 Delimiters 15
 delimiters 15, 17
 data element separator 17
 repetition separator 17

 segment terminator 17
 subelement (component) separator 17
 DISA 14
 document conventions 11

E

EDI 14
 payment processing overview 23
 usage example 23
 EDISIM 27
 eInsight engine
 supported databases 36
 enveloping
 as part of EDI logic 26
 error "Failed to obtain JNDI name prefix" 73, 74
 example of EDI usage 23
 eXchange database
 system requirements 36
 eXchange support
 for Oracle databases 36
 for platforms 35

F

Foresight Corporation 27
 functional acknowledgments (997) 26
 functional group 19
 functional group control number (GS06) 22

G

GS06 (functional group control number) 22

H

HTTPS (HTTP on SSL), configuring 65

I

IC (interchange envelope) 20
 implementation 26
 interchange acknowledgment (TA1) 25
 interchange control number (ISA13) 22
 interchange envelope 20
 ISA13 (interchange control number) 22

J

JNDI name prefix (error message) 73, 74

L

loops 16

O

operating systems supported by eXchange 35
Oracle database for eXchange
 supported levels 36
Oracle eWay, configuring 65, 67
overview
 of EDI payments processing 23
 of X12 14–28

P

payment-related EDI transactions 25
platforms supported by eXchange 35
ports, configuring 65, 66

R

repetition separator 17
response transactions 26

S

sample implementation 61–105
Screenshots 12
SEF file 27
SEF OTD Wizard 27
segment terminator 17
segments 16
SSL (Secure Sockets Layer), configuring 65
ST02 (transaction set control number) 22
structure of an X12 envelope 17
structures 27
 as part of EDI logic 26
subelement (component) separator 17
supporting documents 12
syntax
 control numbers 21
 delimiters 17

T

TA1 (interchange acknowledgment) 25
trading partner agreements 27
transaction set control number (ST02) 22
translations
 as part of EDI logic 26

U

UN/EDIFACT standard 14

V

validations
 as part of EDI logic 26

W

what is a message structure? 15

X

X12
 acknowledgment types 25
 additional information (Web sites) 28
 data elements 16
 end-to-end example 25
 envelope structure 17
 functional group 19
 interchange envelope 20
 loops 16
 segments 16
 what is it? 14
X12 body 14, 15
X12 overview 14–28
X12 sample implementation 61–105