Information Builders

iWay

iWay Integration Solution for UN/EDIFACT User's Guide

Version 7.0.x and Higher

August 16, 2018

Contents

Preface

This documentation describes how to configure and use the iWay Integration Solution for EDIFACT. It is intended for developers to enable them to parse, transform, validate, store, and integrate UN/EDIFACT electronic data documents into standard XML format.

Note: This Release 7.0.x content is currently being updated to support iWay Release 8.0.x software. In the meantime, it can serve as a reference for your use of iWay Release 8. If you have any questions, please contact *Customer_Success@ibi.com*.

How This Manual Is Organized

This manual includes the following chapters:

	Chapter/Appendix	Contents
1	Introducing the iWay Integration Solution for UN/EDIFACT	Provides an overview of UN/EDIFACT and describes the features that are provided by the iWay Integration Solution for UN/EDIFACT.
2	Deployment Information for Your iWay Integration Solution	Describes the iWay products used with your iWay Integration Solution for UN/EDIFACT and provides a roadmap to full information on those products. Introduces the concept of a channel for the construction of a message flow in iWay Service Manager.
3	Configuring the EDI Activity Log	Describes how to configure the EDI Activity Log using iWay Service Manager.
4	Working With UN/EDIFACT Inbound and Outbound Applications Using iWay Integration Tools (iIT)	Describes how to work with UN/EDIFACT inbound and outbound applications using iWay Integration Tools (iIT).
5	Inbound Processing: UN/EDIFACT to XML	Includes an overview of the iWay business components and processing steps in a basic inbound message flow. The message flow converts a document from EDIFACT format to XML format. Also includes instructions for configuring a basic inbound message flow.

	Chapter/Appendix	Contents
6	Outbound Processing: XML to UN/EDIFACT	Includes an overview of the iWay business components and processing steps in a basic outbound message flow. The message flow converts a document from XML format to EDIFACT format. Also includes instructions for configuring a basic outbound message flow.
A	UN/EDIFACT Supported Versions	Describes the UN/EDIFACT versions supported by the iWay Integration Solution for UN/EDIFACT in the Ebix files supplied with the product.
B	Using iWay Integration Tools to Configure an Ebix for EDIFACT	Describes how to use iWay Integration Tools (iIT) to configure an e-Business Information Exchange (Ebix) file for EDIFACT.
C	EDIFACT Special Register (SREG) Types	Describes the Special Register (SREG) types that are created during EDIFACT to XML transactions and acknowledgment creation.
D	Sample UN/EDIFACT Files	Includes a sample UN/EDIFACT version D97A for APERAK (Application Error and Acknowledgment), INVOIC (Invoice Message), and ORDERS (Purchase Order message).

Documentation Conventions

The following table describes the documentation conventions that are used in this manual.

Convention	Description

`THIS TYPEFACE`
or
`this typeface`
	Denotes syntax that you must enter exactly as shown.

this typeface	Represents a placeholder (or variable), a cross-reference, or an important term. It may also indicate a button, menu item, or dialog box option that you can click or select.
underscore	Indicates a default setting.
Key + Key	Indicates keys that you must press simultaneously.

Convention	Description
{ }	Indicates two or three choices. Type one of them, not the braces.
|	Separates mutually exclusive choices in syntax. Type one of them, not the symbol.
...	Indicates that you can enter a parameter multiple times. Type only the parameter, not the ellipsis (...).
. . .	Indicates that there are (or could be) intervening or additional commands.

Related Publications

Visit our Technical Documentation Library at *http://documentation.informationbuilders.com*. You can also contact the Publications Order Department at (800) 969-4636.

Customer Support

Do you have any questions about this product?

Join the Focal Point community. Focal Point is our online developer center and more than a message board. It is an interactive network of more than 3,000 developers from almost every profession and industry, collaborating on solutions and sharing tips and techniques. Access Focal Point at *http://forums.informationbuilders.com/eve/forums*.

You can also access support services electronically, 24 hours a day, with InfoResponse Online. InfoResponse Online is accessible through our website, *http://www.informationbuilders.com*. It connects you to the tracking system and known-problem database at the Information Builders support center. Registered users can open, update, and view the status of cases in the tracking system and read descriptions of reported software issues. New users can register immediately for this service. The technical support section of *http://www.informationbuilders.com* also provides usage techniques, diagnostic tips, and answers to frequently asked questions.

Call Information Builders Customer Support Services (CSS) at (800) 736-6130 or (212) 736-6130. Customer Support Consultants are available Monday through Friday between 8:00 a.m. and 8:00 p.m. EST to address all your questions. Information Builders consultants can also give you general guidance regarding product capabilities and documentation. Please be ready to provide your six-digit site code number (*xxxx.xx*) when you call.

To learn about the full range of available support services, ask your Information Builders representative about InfoResponse Online, or call (800) 969-INFO.

Help Us to Serve You Better

To help our consultants answer your questions effectively, be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following tables list the environment information our consultants require.

Platform	
Operating System	
OS Version	
JVM Vendor	
JVM Version	

The following table lists the deployment information our consultants require.

Adapter Deployment	For example, iWay Business Services Provider, iWay Service Manager
Container	For example, WebSphere
Version	
Enterprise Information System (EIS) - if any	
EIS Release Level	
EIS Service Pack	
EIS Platform	

The following table lists iWay-related information needed by our consultants.

iWay Adapter	
iWay Release Level	
iWay Patch	

The following table lists additional questions to help us serve you better.

Request/Question	**Error/Problem Details or Information**
Did the problem arise through a service or event?	
Provide usage scenarios or summarize the application that produces the problem.	
When did the problem start?	
Can you reproduce this problem consistently?	
Describe the problem.	
Describe the steps to reproduce the problem.	
Specify the error message(s).	
Any change in the application environment: software configuration, EIS/database configuration, application, and so forth?	
Under what circumstance does the problem *not* occur?	

The following is a list of error or problem files that might be applicable.

- Input documents (XML instance, XML schema, non-XML documents)
- Transformation files
- Error screen shots
- Error output files
- Trace files
- Service Manager package or archive to reproduce problem
- Custom functions and agents in use
- Diagnostic Zip
- Transaction log
- Archive File
- IIA

For information on tracing, see the *iWay Service Manager User's Guide*.

User Feedback

In an effort to produce effective documentation, the Technical Content Management staff welcomes your opinions regarding this document. Please use the Reader Comments form at the end of this document to communicate your feedback to us or to suggest changes that will support improvements to our documentation. You can also contact us through our website, *http://documentation.informationbuilders.com/connections.asp*.

Thank you, in advance, for your comments.

Information Builders Consulting and Training

Interested in training? Information Builders Education Department offers a wide variety of training courses for this and other Information Builders products.

For information on course descriptions, locations, and dates, or to register for classes, visit our website (*http://education.informationbuilders.com*) or call (800) 969-INFO to speak to an Education Representative.

Chapter 1

Introducing the iWay Integration Solution for UN/EDIFACT

The iWay Integration Solution for UN/EDIFACT transforms electronic Data documents into standard XML format, or transforms XML representations into UN/EDIFACT format.

This section provides an overview of UN/EDIFACT and describes the features that are provided by the iWay Integration Solution for UN/EDIFACT.

In this chapter:

A Brief History of EDIFACT

EDIFACT is a set of standards for formatting information that is electronically exchanged between one business and another, or within a business. These standards describe how documents for conducting certain aspects of business-such as purchase orders and invoices-are structured.

By specifying a standardized, computer-readable format for transferring data, EDIFACT (UN/EDIFACT) enables the automation of commercial transactions around the world. It provides a common, uniform language through which computers can communicate for fast and efficient transaction processing.

Early Standardization Efforts

Before the development of standards, many businesses used proprietary systems to exchange trading information such as purchase orders and invoices. However, they recognized the economic need for a faster, less costly way to process information in order to stay competitive in the business world. Business sectors such as transportation, grocery supply, and banking drove the creation of standards for the communication of data.

In 1988, the United Nations chartered UN/EDIFACT (United Nations Electronic Data Interchange for Administration, Commerce, and Transport) to develop a worldwide, internationally approved standard structure for exchanging information among partners. The UN/EDIFACT standards are called United Nations Standard Messages (UNSM). They are comparable to the ANSI ASC X12 transaction sets.

UN/EDIFACT is the standardized data format is widely used by the international business community.

UN/EDIFACT is a collection of 200 messages. UN/EDIFACT standard is used to perform nearly every aspect of business operation such as order placement and processing, shipping and receiving, invoicing and payment, pricing and sales, and inventory. It streamlines the communication of data to and from a broad range of entities, including financial institutions, insurance providers, food and pharmaceutical suppliers, retailers, and automotive manufacturers.

Features of the iWay Integration Solution for UN/EDIFACT

The standards-based iWay Integration Solution for UN/EDIFACT reduces the amount of effort it takes to integrate UN/EDIFACT documents with your internal enterprise applications and third-party trading partners. It includes conversion and validation of documents from UN/EDIFACT to XML format, making it easy to include UN/EDIFACT documents in your XML-based integration projects.

Features of the iWay Integration Solution for UN/EDIFACT include:

- Integration with iWay Service Manager to provide bi-directional conversion of UN/EDIFACT formats and XML between application servers, integration brokers, third party software packages, and messaging services.
- Integration with more than 200 other information assets, including J2EE-based back-office systems; data structures such as DB2, IMS, VSAM, and ADABAS; and front-office systems based on Sybase.
- Integration with leading application servers, integration brokers, and development environments. Supported software platforms include BEA WebLogic, IBM WebSphere, Sun Java Enterprise System, and Oracle Application Server.
- Support for synchronous and asynchronous bi-directional interactions for UN/EDIFACT documents between application servers, integration brokers, third-party software packages, and messaging services.
- Support for UN/EDIFACT transaction sets. For details on the supported transaction sets, see *UN/EDIFACT Supported Versions* on page 143.
- Reusable framework for parsing, transforming, and validating UN/EDIFACT documents without the need to write custom code.

UN/EDIFACT Information Roadmap

The following table lists the location of deployment and user information for products used with the iWay Integration Solution for UN/EDIFACT.

Product	For more information, see...
iWay Service Manager	Chapters 4 and 5 of this guide *iWay Service Manager User's Guide*
iWay Transformer	*iWay Transformer User's Guide*

Chapter 2

Deployment Information for Your iWay Integration Solution

This topic describes the iWay products used with your iWay Integration Solution for UN/EDIFACT and provides a roadmap to full information on those products.

It also introduces the concept of a channel for the construction of a message flow in iWay Service Manager.

In this chapter:

iWay Products and Components

Your iWay integration solution works in conjunction with one or more of the following products and components:

- iWay Service Manager
- iWay Transformer
- iWay Integration Tools Designer
- iWay Activity Facility
- iWay Correlation Facility

iWay Service Manager

iWay Service Manager is the heart of the Universal Integration Framework and is an open transport service bus. Service Manager uses graphical tools to create sophisticated integration services without writing custom integration code by:

- Using metadata from target applications
- Transforming and mapping interfaces
- Managing stateless processes

Its capability to manage complex integration interactions makes it ideally suited to be the foundation of a service-oriented architecture.

For more information, see the *IWay Service Manager User's Guide*.

iWay Transformer

iWay Transformer is a rule-based data transformation tool that converts an input document of one data format to an output document of another data format or structure. The easy-to-use graphical user interface and function tool set facilitate the design of transform projects that are specific to your requirements.

For more information, see the *iWay Transformer User's Guide*.

iWay Integration Tools Designer

iWay Integration Tools (iIT) Designer (previously known as iWay Designer) is a GUI tool that is delivered as a plugin with iIT.

The capability of graphically visualizing a business process is a powerful and necessary component of any e-Business offering. iWay Integration Tools Designer, a Windows-based design-time tool, provides a visual and user-friendly method of creating a business process, also called a process flow. Through a process flow, you control the sequence in which tasks are performed and the destination of the output from each task.

For more information, see the *iWay Integration Tools Designer User's Guide*.

Activity Facility

The Activity Facility maintains a record describing each message that passes through the server. The messages are associated and integrated with the transactions. This makes it possible for an auditor to review them individually or in conjunction with other messages that fall within the scope of the same transaction. The Activity Facility can record:

- Original input messages.
- Each emitted message (XML, APERAK, CONTRL).
- Transaction status.
- Intermediate activities.

For more information on using the Activity Facility, see the *iWay Service Manager User's Guide*.

Correlation Facility

The Correlation Facility (also known as the Correlation Manager) maintains records of anticipated activities occurring in the system. Correlation actions take the correlation from OPEN to CLOSED state, and allow history to be recorded. Agents are provided to implement Correlation Facility interactions within process flows, however, it is possible to use this API to accomplish this same purpose within your own exits.

For more information on using the Correlation Facility, see the *iWay Service Manager User's Guide* and the *iWay Service Manager Programmer's Guide*.

Using a Channel to Construct a Message Flow

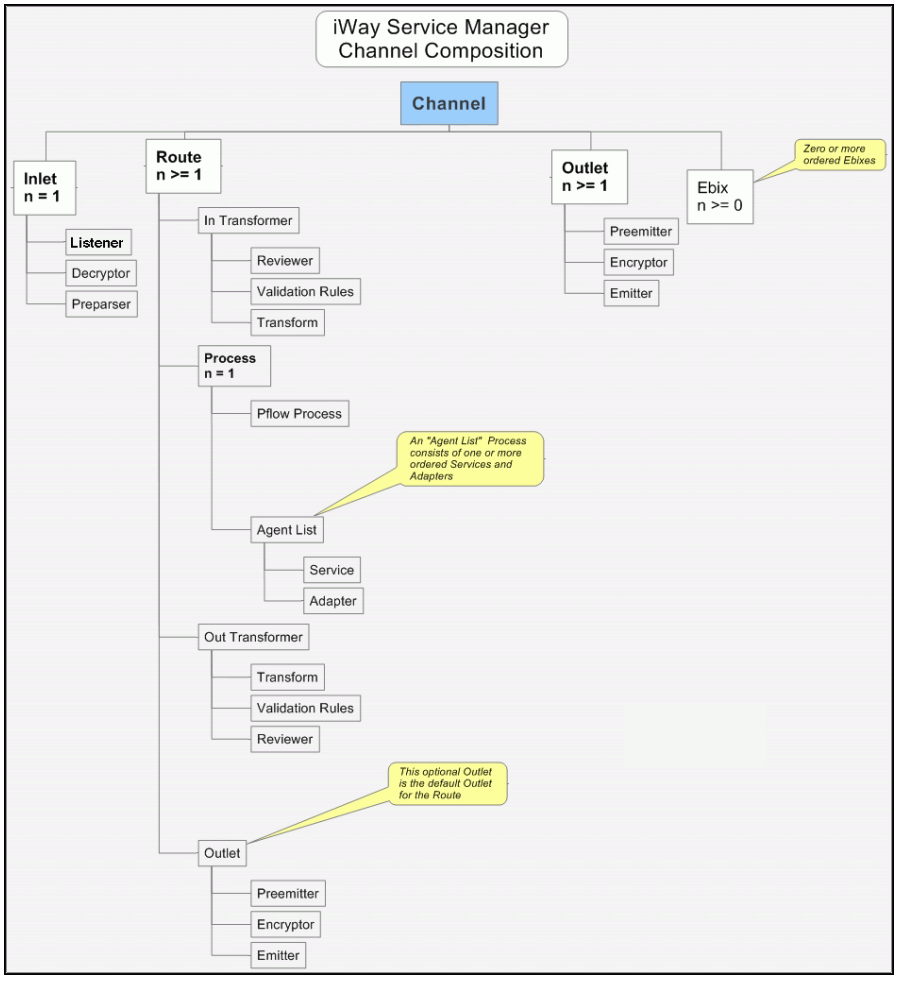
The use of iWay Service Manager is centered on a channel. A channel is a container for all the iWay business components used in an EDIFACT message flow.

At a high level, a channel accepts input data through an inlet, processes the data using a route, and outputs the resulting data through an outlet. Another component in the process is an e-Business Information Exchange (Ebix).

The following diagram shows the channel components available in the construction of a message flow.

In the following diagram, the value *n* underneath a component name indicates how many instances of that component you can have in a channel configuration—zero, one, or more than one. For example, n = 1 for Inlet means that you can have only one inlet on the channel.

Required components are in boldface type.



Components of a Channel

A channel consists of:

- An inlet, which defines how a message enters a channel.
- A route, which defines the path a message takes through a channel.
- Outlets, which define how transformed messages leave a channel.
- An e-Business Information Exchange (Ebix), which is a collection of metadata that defines the structure of data.

iWay Service Manager provides a design-time repository called the Registry, where you assemble and manage the components in a channel.

An inlet can contain:

- A listener (required), which is a protocol handler responsible for picking up an incoming message on a channel.
- A decryptor, which applies a decryption algorithm to an incoming message and verifies the security of the message.
- A preparser, which is a logical process that converts an incoming message into a processable document. The preparsed document then passes through the standard transformation services to reach the designated processing service.

A route can contain:

- An in transformer, which is an exit sequence that applies to a message before processing occurs.
  - A reviewer, which is either the first exit to receive a document after parsing (inbound), or the last exit to receive a document prior to the actual emit operation (outbound). These exits are intended for envelope handling but can be used for any desired purpose.
  - Validation rules, which apply validation using the rules validation engine. Rules are provided when the iWay Integration Solution for UN/EDIFACT is installed.
  - A transform, which is a transformation definition file that contains sets of rules, interpreted and executed by a transformation engine. Transformation is the process by which data is transformed from one structure/format to another.
- A process, which is a stateless, lightweight, short-lived microflow that is executed by iWay Service Manager on a message as it passes through the system. Processes that are published using iIT Designer are available in the Registry and can be bound to channels as routes.
  - A process flow.
  - An agent list.
    - A service, which is an executable Java procedure that handles the business logic of a message.

- An adapter, which refers to a target that represents a specific instance of a connection to a back-end system.

- An out transformer, which is an exit sequence that applies to a message after processing occurs.
  - A transform, which is a transformation definition file that contains sets of rules, interpreted and executed by a transformation engine. Transformation is the process by which data is transformed from one structure/format to another.
  - Validation rules, which apply validation using the rules validation engine. Rules are provided when the iWay Integration Solution for UN/EDIFACT is installed.
  - A reviewer, which is either the first exit to receive a document after parsing (inbound), or the last exit to receive a document prior to the actual emit operation (outbound). These exits are intended for envelope handling but can be used for any desired purpose.
- An outlet (optional), which is responsible for all aspects of preparing a document for emission and then emitting it.
  - A preemitter, which is a logical process that handles a document immediately before transmission. Normally it converts an XML document into non-XML format.
  - An encryptor, which can be called to encrypt an outgoing document.
  - An emitter, which is a transport protocol that sends a document to its recipient.

An outlet can contain:

- A preemitter.
- An encryptor.
- Multiple emitters.

For details on the preceding components, see the *iWay Service Manager User's Guide*.

Components of the iWay Integration Solution for UN/EDIFACT

iWay business components used in the construction of a message flow for EDIFACT transactions include:

- An Ebix (e-Business Information Exchange)
- A preparser
- An acknowledgment service

- A preemitter

Ebix

iWay Software provides various e-Business Information Exchange (Ebix) files used in conjunction with the iWay integration solutions. In iWay Service Manager, the iWay Integration Solution for UN/EDIFACT contains several Ebix files, one for each supported EDIFACT transaction set.

An Ebix file for EDIFACT is named EDIFACT_*transaction_set*.ebx, where *transaction_set* is the transaction set number. For example, the Ebix file for EDIFACT transaction set D01B is named EDIFACT_D01B.ebx.

For details on the supported EDIFACT transaction sets, see *UN/EDIFACT Supported Versions* on page 143.

An Ebix is a collection of metadata that defines the structure of data. The Ebix supplied with the iWay Integration Solution for UN/EDIFACT defines the structure of supported EDIFACT messages.

Each Ebix includes:

- Pre-built data dictionaries. The structure of each EDIFACT document is described by two data dictionaries:
  - Header dictionary, which describes the enveloping structure of the document.
  - Document dictionary, which describes the segments and elements that compose each document.

  The dictionaries from the Ebix are used to transform the structure of a document per EDIFACT regulation.
- Pre-built XML schemas that define the structure and content of XML messages in detail.
- Pre-built EDIFACT to XML transformation templates, and XML to EDIFACT templates, for the supported EDIFACT transaction sets.
- Pre-built rule files for each message. The iWay Integration Solution for UN/EDIFACT uses these rule files to validate inbound and outbound documents.

Preparsers

A preparser is an iWay business component that converts incoming messages into processable documents.

Typically a preparser converts a non-XML document into XML format. The preparser for the iWay Integration Solution for UN/EDIFACT converts an incoming EDIFACT formatted document to XML format.

The EDIFACT Batch Splitter (com.ibi.preparsers.XDEDIFACTBatchSplitter) is provided by iWay Software for the iWay Integration solution for UN/EDIFACT to split EDIFACT input files that contain multiple transactions or envelopes. This preparser should be used with the EDIFACTPreParser.

The EDIFACTPreParser (com.ibi.preparsers.XDEDIFACTPreParser) is provided by iWay Software for the iWay Integration Solution for UN/EDIFACT to parse an EDIFACT input file.

Preemitter

A preemitter is a logical process that handles a document immediately before transmission.

Typically a preemitter is used to convert an XML document to non-XML format. The XML document is created from EDIFACT input data in inbound processing. The iWay Integration Solution for UN/EDIFACT uses a preemitter in outbound processing to convert the XML-formatted EDIFACT document to an EDIFACT formatted document.

The XML structure must be compliant with the schema supplied in the Ebix.

The preemitter for the iWay Integration Solution for UN/EDIFACT is called XDEDIFACTPreEmitter (com.ibi.preemit.XDEDIFACTPreEmitter).

Chapter 3

Configuring the EDI Activity Log

This section describes how to configure the EDI Activity Log using iWay Service Manager.

In this chapter:

EDI Activity Log Overview

The EDI Activity Log is an extension of the Activity Facility in iWay Service Manager. It is used to log events as messages are processed. Logging can occur when:

- a message is acquired.
- a message is emitted.
- an error occurs.
- a component such as an agent or process flow is called.

For more information about the Activity Facility, see the *iWay Service Manager User's Guide*.

Configuring the EDI Activity Log Using iWay Service Manager

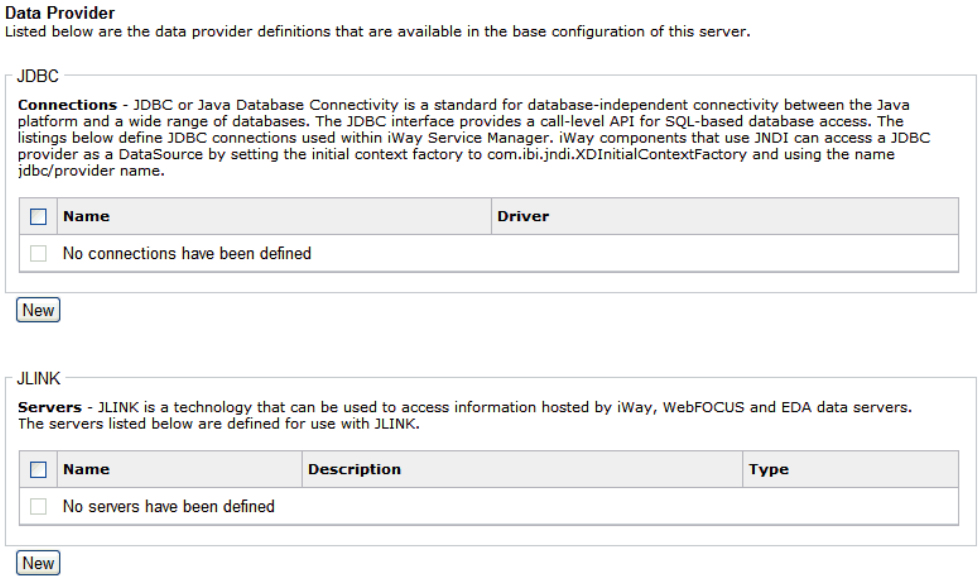
This section describes how to configure the Data Provider and the EDI Activity Log.

Procedure: How to Configure the Data Provider

To configure the Data Provider:

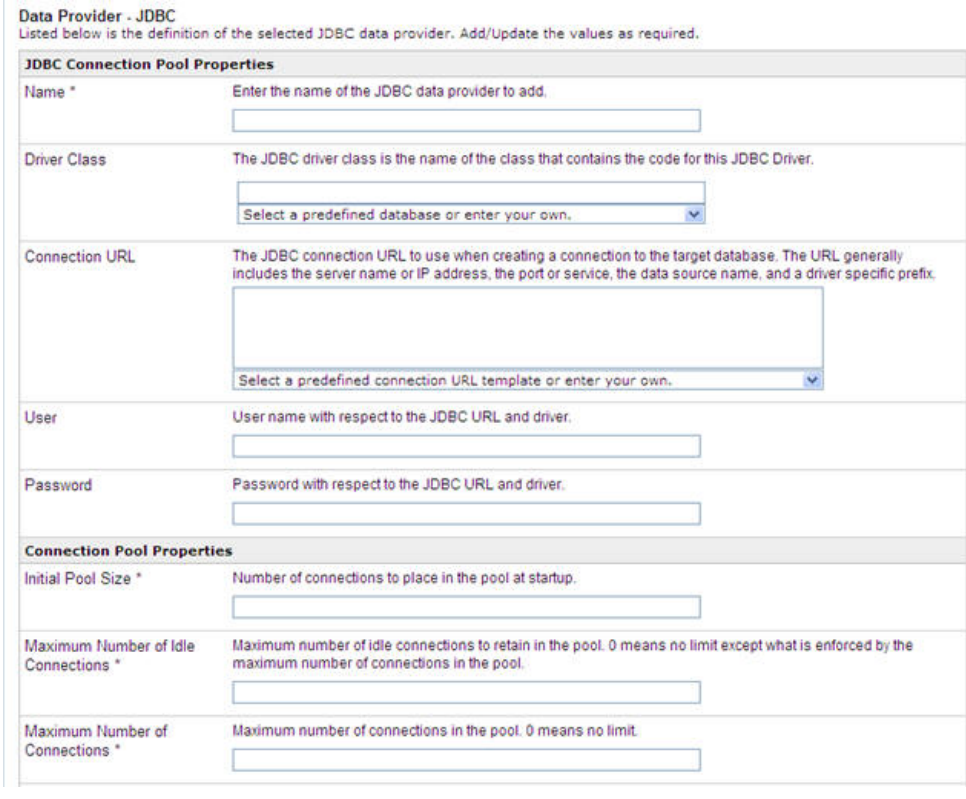
1. In the left console pane of the Server menu, select *Data Provider*.

The Data Provider pane opens, as shown in the following image.



2. Click *New* in the JDBC area to configure a new Data Provider handler.

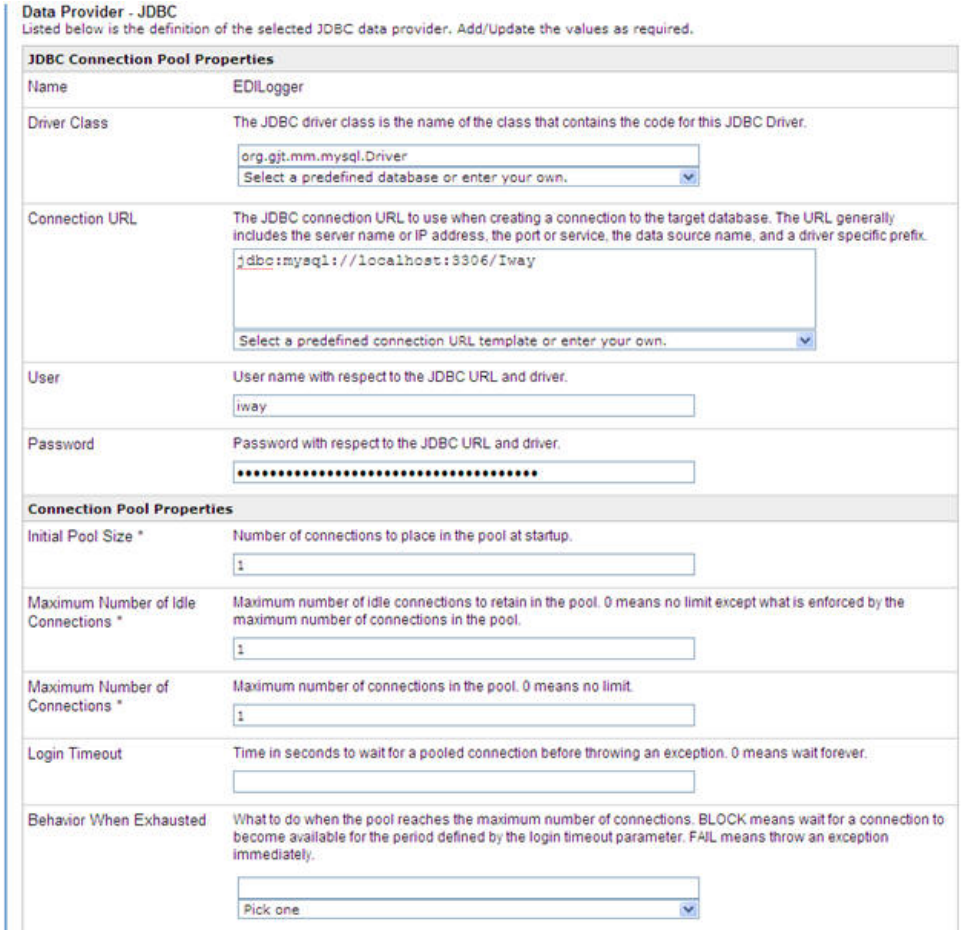
The configuration pane for the Data Provider handler opens, as shown in the following image.



3. In the Name field, type a unique name for the JDBC Data Provider.
4. From the Driver Class drop-down list, click the appropriate database driver. Alternatively, you can manually type the database driver class.
5. From the Connection URL drop-down list, click the database connection URL, or manually type the URL.
6. In the User field, type a valid user name for the JDBC URL and driver.
7. In the Password field, type a valid password for the user name.
8. In the Initial Pool Size field, type the number of connections that are placed in the pool at startup.
9. In the Maximum Number of Idle Connections field, type the maximum number of idle connections that are retained in the pool.
10. In the Maximum Number of Connections field, type the maximum number of connections allowed in the pool.

In the following image, sample values for a new Data Provider have been supplied.

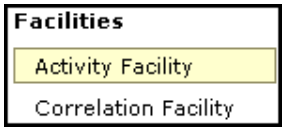
Tip: In the image, the Initial Pool Size, Maximum Number of Idle Connections, and Maximum Number of Connections are set to 1. See your database administrator to determine the optimum connection pool properties for your site.



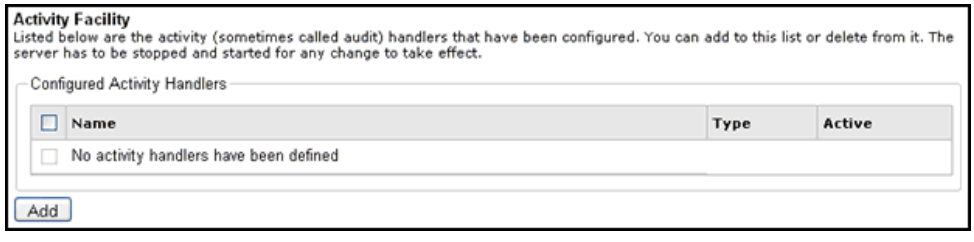
11. Click *Add* to complete the procedure for adding a Data Provider.
    Next, you need to configure the EDI Activity Log, as described in the following procedure.

Procedure: How to Configure the EDI Activity Log

To configure the EDI Activity Log:



1. In the left console pane of the Server menu, select *Activity Facility*.
   The Activity Facility pane opens, as shown in the following image.



   The table that is provided lists the configured Activity Facility handlers. Initially, no handlers are shown.

2. Click *Add* to configure a new Activity Facility handler.

The configuration pane for the Activity Facility handler opens.

3. From the Type drop-down list, click *EDI Activity Logs*.
4. Type a unique name for the EDI Activity Driver and a brief description.
5. From the Active drop-down list, click *true*.
6. In the JNDI Name field, type the name of the Data Provider that you created in *How to Configure the Data Provider* on page 25.
   For example, if the Data Provider name is edilogger, then the JNDI Name is:

   ```
   edilogger
   ```

7. Provide values for the remaining parameters, as defined in the following table.

Parameter Name	Type	Description
Table	String	Table name for the activity log. This must be a valid identifier in the database being used. If the table does not exist at startup, it will be created automatically.
Compression	Drop-down list	Specify whether the messages are to be compressed. Values include: ❏ none (default) ❏ smallest ❏ fastest ❏ standard ❏ Huffman
Start Events	Boolean Drop-down list	If set to *true* (default), the input messages will be recorded in the activity log. This values must be set to *true* for use of the audit reports in the console.
Internal Events	Boolean Drop-down list	If set to *true*, system events are included in the activity log. System events include activities such as parsing and transformations (optional). False is selected by default.
Security Events	Boolean Drop-down list	If set to *true* (default), security events are recorded. This includes digital signature, and so on. However, console activity is not recorded.
Business Error Events	Boolean Drop-down list	If set to *true*, business errors are recorded, such as rules system violations. False is selected by default.

Parameter Name	Type	Description
Emit Events	Boolean Drop-down list	If set to *true* (default), output messages from emitter services will be recorded. This is required for use of the audit log reports in the console.
End Events	Boolean Drop-down list	If set to *true* (default), the end of message processing will be recorded in the activity log. This is required for use of the audit log reports in the console.
Notes Table	String	Table name for the notes table, which contains log annotations. If the table does not exist at startup, it will be created automatically.
MAC Algorithm	String Drop-down list	The Message Authentication Code (MAC) algorithm. None (default) indicates a MAC should not be computed.
MAC Provider	String Drop-down list	The Message Authentication Code (MAC) provider. Not Specified indicates the default provider should be used. The remaining available value is *SunJCE*.
MAC Secret Key	String	The Message Authentication Code (MAC) secret key to use.

8. Click *Update*.

   If necessary, start the database services.

9. Restart iSM to start the EDI Activity Driver and begin logging.

   The EDI Activity Driver inserts records into the configured activity database. The records are designed for fast writing rather than for ease of later analysis. A set of inquiry service agents suitable for use in a process flow is available to assist during the analysis of the log. Users are cautioned that iWay does not guarantee the layout of the record from release to release, and this should be checked against the actual schema.

Database Field	Description
recordkey	Unique record identifier.
recordtype	Type of this record - the event being recorded. ❏ 101 - Message start. ❏ 131 - Entry to event (see subtype codes below). ❏ 132 - Normal exit from event. ❏ 133 - Failed exit from event. ❏ 151 - Ancillary message (usually rules violation). ❏ 181 - Emit. ❏ 191 - Message end.
signature	Encoding of the listener name and protocol.
protocol	Name of the protocol.
address	Address to which an emit is to be issued. The format depends on the protocol.
tstamp	Timestamp of record.
correlid	UNB05
tid	Transaction ID assigned to this message.
msg	Message appropriate to this record type. For example, an input message contains the original message received, if possible. Streaming input does not contain a record.
context	Serialized special registers that were in the context at the time the record was written.
text	Message text for business errors (rules system violations).

Database Field	Description
status	Status code recorded. ❏ 0 - Success ❏ 1 - Success, message end (191 record) ❏ 10 - Rules error
subtype	Event code for event records. ❏ 1 - Preparser ❏ 2 - Parser ❏ 3 - In reviewer ❏ 5 - In validation ❏ 6 - In transform ❏ 7 - Agent or flow ❏ 8 - Out transform ❏ 9 - Out validation ❏ 11 - Preemitter ❏ 1000 - input record written to table before transformation
partner_to	UNB03
partner_from	UNB02
encoding	Encoding of the listener that obtained the document.
mac	Not used in this version.
Driver version	1.0 in 7.0 SM

Chapter 4

Working With UN/EDIFACT Inbound and Outbound Applications Using iWay Integration Tools (iIT)

This chapter describes how to work with UN/EDIFACT inbound and outbound applications using iWay Integration Tools (iIT).

In this chapter:

UN/EDIFACT Inbound and Outbound Application Overview

This chapter provides instructions to create, import, export, and work with UN/EDIFACT inbound and outbound applications using iWay Integration Tools (iIT). In addition, you will learn how to create an iWay Integration Application (iIA) for deployment based on the sample data.

What will the Application do?

The iIAs will be used to transform EDIFACT to XML for inbound processing and XML to EDIFACT for outbound processing.

The inbound application channel creates an XML representation of an EDIFACT inbound message, a functional acknowledgement (APERAK or CONTROL), and an XML-formatted validation report. The documents are routed to designated folders based on the success or failure results of the transformation and EDIFACT validation.

The outbound application channel creates an EDIFACT message from XML and a XML formatted validation report.

EDIFACT Prerequisites

Before you continue, ensure that the following prerequisites are met:

- ❑ You have a working knowledge of iWay Service Manager (iSM) and iWay Integration Tools (iIT).
- ❑ iSM Version 7.0.6 is installed.
- ❑ iWay EDIFACT Adapter is installed.
- ❑ iIT Version 7.0.6 is installed.
- ❑ System and channel Special Registers (SREGs) are updated to match your directory structure, as shown in *How to Extract User Samples for EDIFACT* on page 36.

Extracting EDIFACT User Samples

This section describes how to extract user samples for EDIFACT.

Procedure: How to Extract User Samples for EDIFACT

1. Download the EDIFACT_usr_samples.zip file containing EDIFACT user sample workspace from the following website:

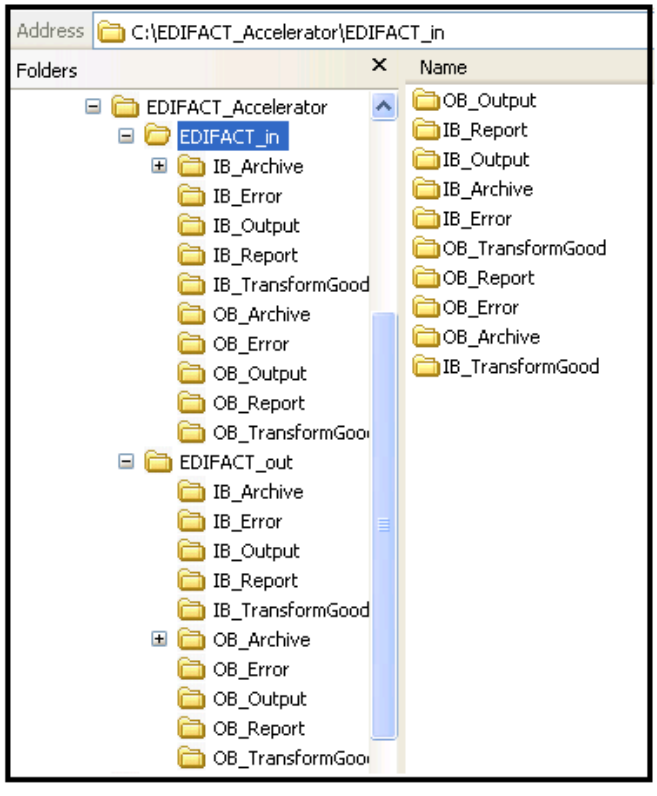
   http://techsupport.informationbuilders.com

   The downloaded EDIFACT_usr_samples.zip contains the following files:

   - ❑ EDIFACT_Accelerator.zip
   - ❑ EDIFACT_usr_samples_iIT_workspace.zip

2. Save the EDIFACT_usr_samples_iIT_workspace.zip file to a folder on your local drive.

3. Save and extract the EDIFACT_Accelerator.zip file to a location where you want to store your data, as shown in the following image.



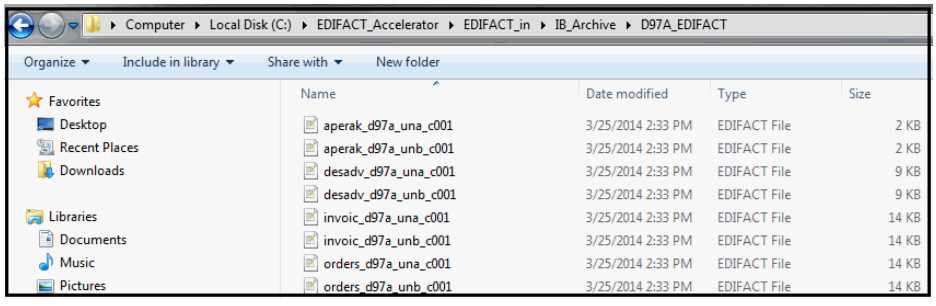
4. The EDIFACT_Accelerator.zip file contains sample input and output data that you can use.

   - Inbound test data is located in the following folder:

     `\EDIFACT_Accelerator\EDIFACT_in\IB_Archive`

     There is a folder called D97A_EDIFACT.

For example:

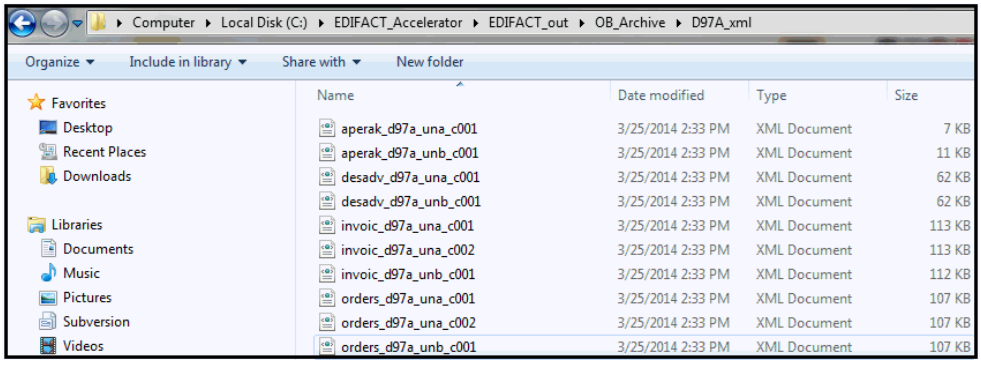


- Outbound test data is located in the following folder:

  `\EDIFACT_Accelerator\EDIFACT_out\OB_Archive`

  There is a folder called D97A_xml.

  For example:



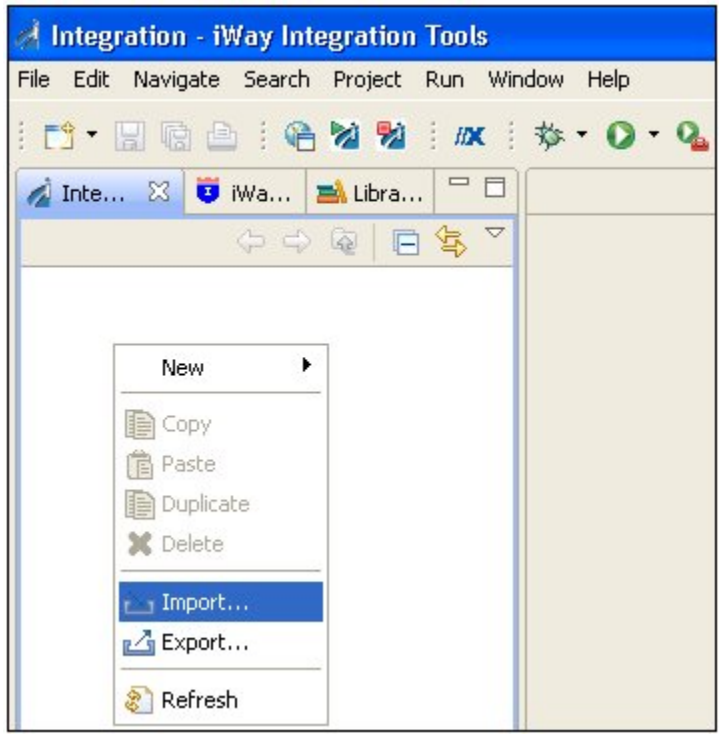
Importing EDIFACT User Samples to iWay Integration Tools as a Workspace

This section describes how to import EDIFACT user samples to iWay Integration Tools (iIT) as a workspace.

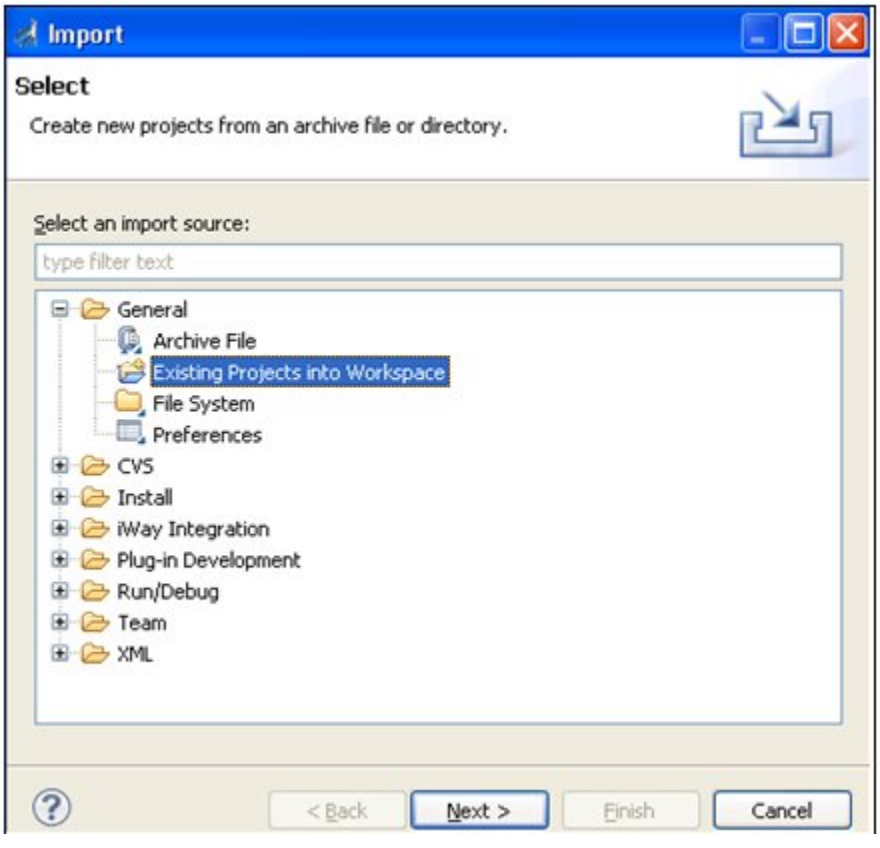
Procedure: How to Import EDIFACT User Samples to iWay Integration Tools as a Workspace

1. Start iWay Integration Tools (iIT).

2. Right-click anywhere inside the Integration Explorer tab and select *Import...* from the context menu, as shown in the following image.

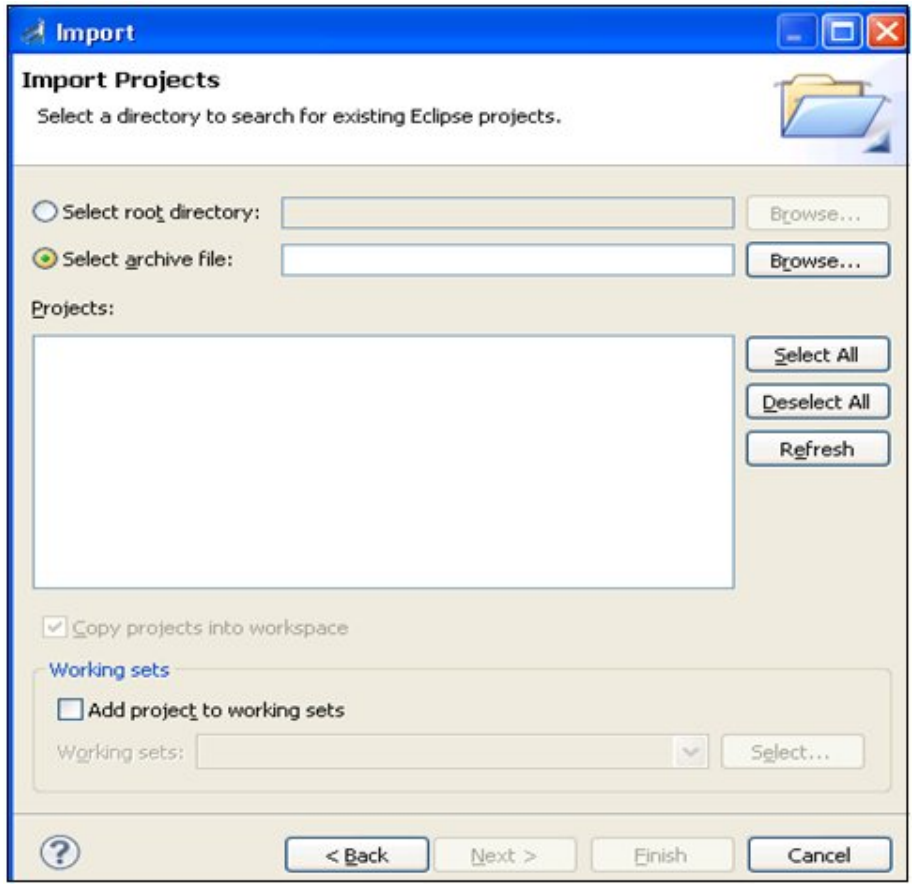


The Import dialog opens, as shown in the following image.



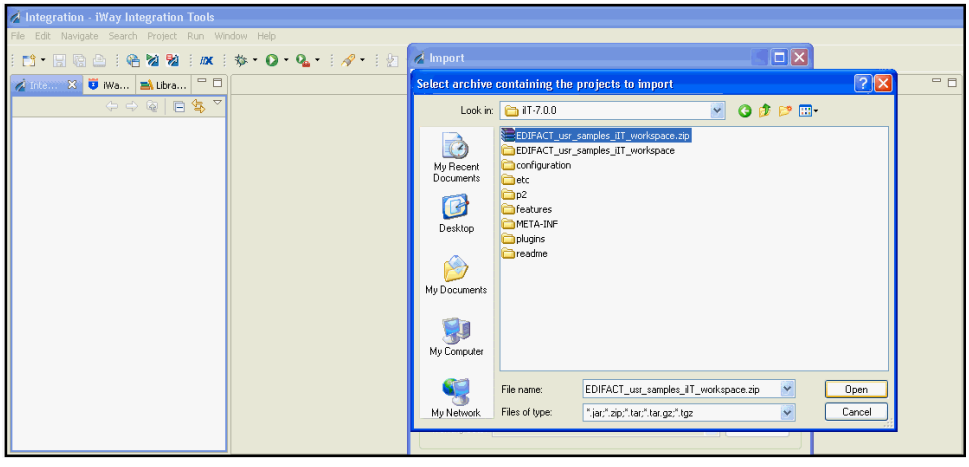
3. Expand the *General* folder, select *Existing Projects into Workspace*, and then click *Next*.

The Import Projects pane opens, as shown in the following image.



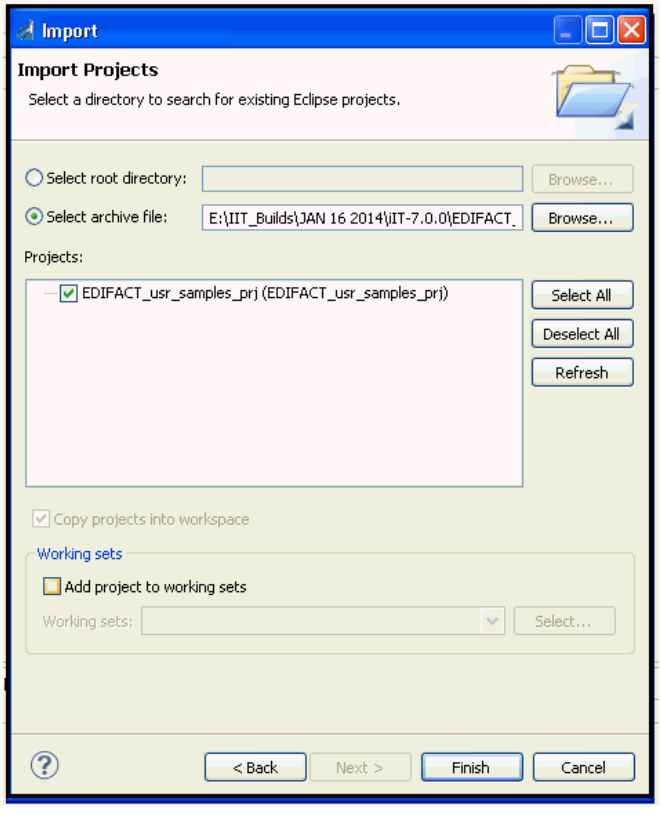
4. Click *Select archive file* and then click *Browse*.

The Select archive containing the projects to import pane opens, as shown in the following image.



5. Select the *EDIFACT_usr_samples_iIT_workspace.zip* file and click *Open*.

You are returned to the Import Projects pane, as shown in the following image.



6. Click *Finish*.

The EDIFACT user samples are loaded into your iIT workspace, as shown in the following image.

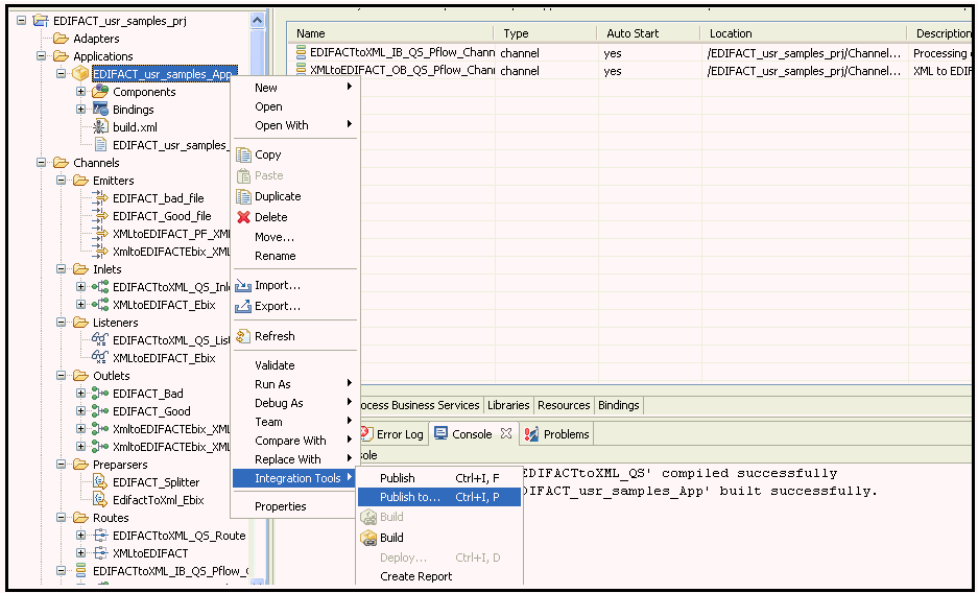
The Integration Explorer tab on the left pane displays a hierarchy of all the imported channel components (for example, Ebixes, listeners, outlets, preparsers, routes, process flows, and so on). The Console tab on the bottom provides a status as each channel component is imported.

Publishing iWay Integration Applications to the iWay Service Manager Registry

This section describes how to publish iWay Integration Applications (iIAs) to the iWay Service Manager (iSM) Registry.

Procedure: How to Publish iWay Integration Applications to the iWay Service Manager Registry

1. In the Integration Explorer tab, right-click *EDIFACT_usr_samples_App*, select *Integration Tools* from the context menu, and then click *Publish to...*, as shown in the following image.



The Publish Resource Wizard dialog opens, as shown in the following image.

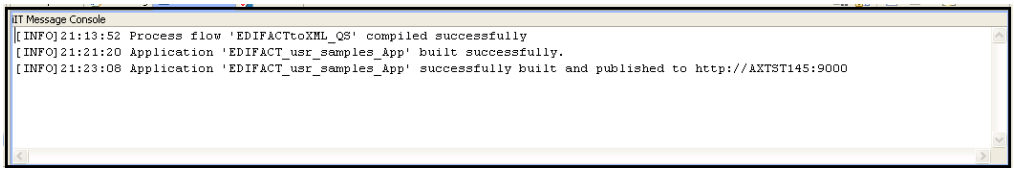
2. In the Server URL field, type the server IP number or computer name and then the port number (default port is 9000). For example:

   `http://111.111.111.000:9000`

   Type the iSM credentials (for example, user name: *iway*, password: *iway*).

3. Click *Finish*.

   The Console tab on the bottom provides a status log that you can use for verification purposes, as shown in the following image.



Deploying iWay Integration Applications to iWay Service Manager

This section describes how to deploy iWay Integration Applications (iIAs) to iWay Service Manager (iSM).

Procedure: How to Deploy iWay Integration Applications to iWay Service Manager

1. Enter the following URL to access the iSM Administration Console:

   `http://[host]:[port]/ism`

   where:

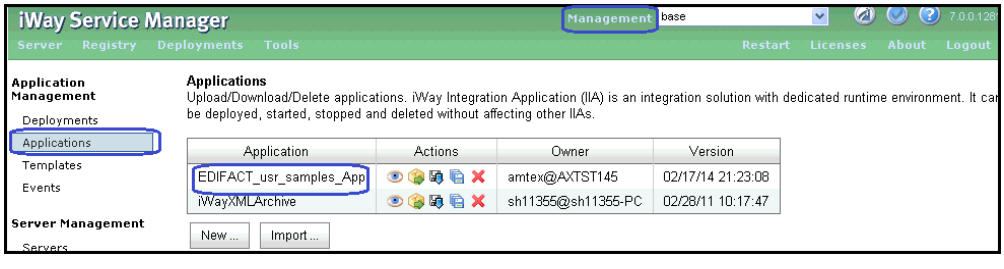
   host

   Is the host machine where iSM is installed. The default value is *localhost*.

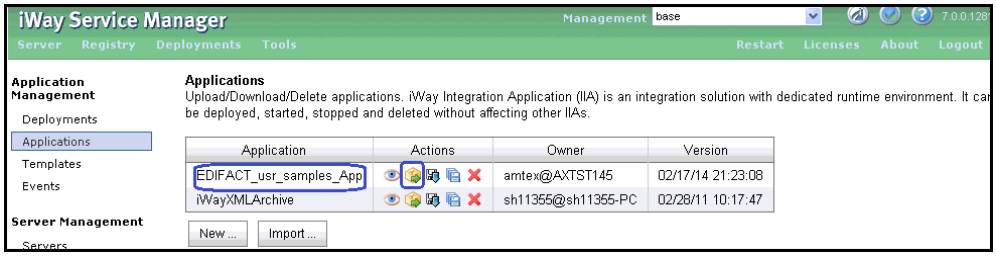
   port

   Is the port where iSM is listening. The default port is *9999*.

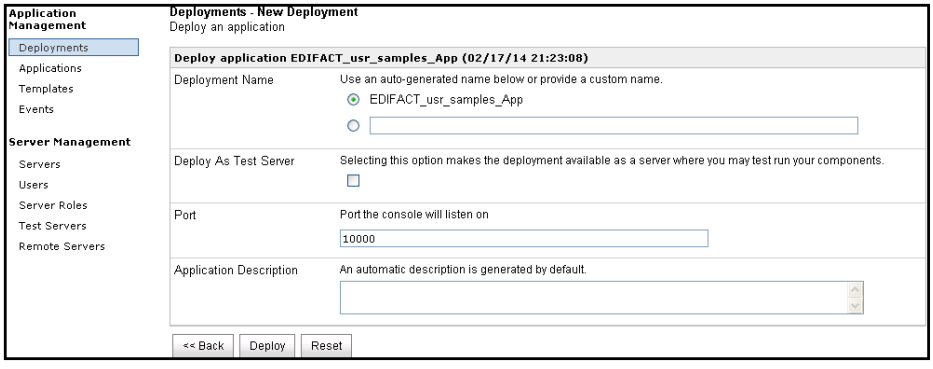
2. After publishing the iWay Integration Application (EDIFACT_usr_samples_App), you can find this iIA under the Management\Applications link in the iSM Administration Console, as shown in the following image.



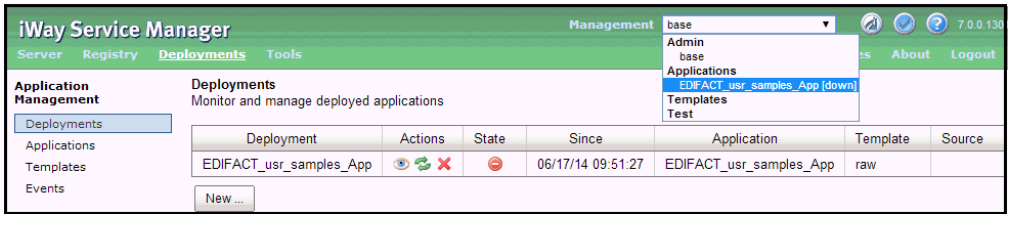
3. Click the *Deploy* icon  next to the application name under the Actions column, as shown in the following image.



The Deployments pane opens, as shown in the following image.



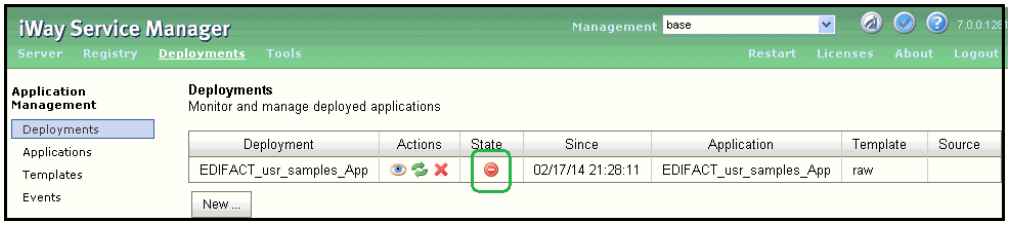
4. Click *Deploy*.
5. From the Management drop-down list, select your deployed application (for example, *EDIFACT_usr_samples_App [down]*), as shown in the following image.



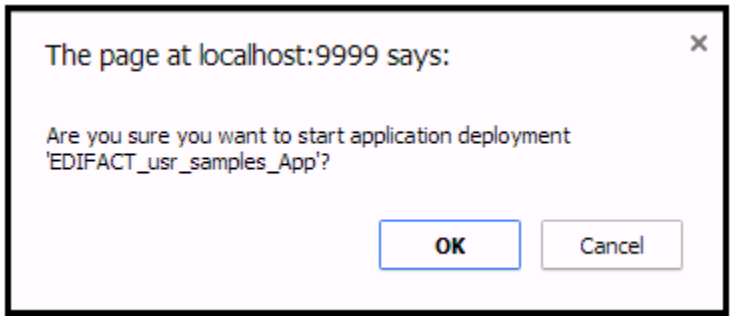
6. Click *Server* in the top menu and then *Register Settings* in the left pane for the *EDIFACT_usr_samples_App [down]* application.
7. Click *Add* to create all required registers (*EDIFACT_Installdir*, *EDIFACT_Input*, *EDIFACT_Output*, and *ValidateEDIFACT*) for the *EDIFACT_usr_samples_App [down]* application.

   For more information, see *Setting Registers in the iWay Service Manager Administration Console* on page 50.

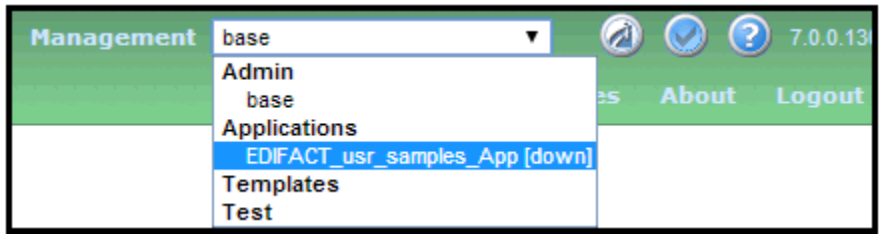
8. In the State column, click the *Deployment State*  icon to start the deployed Application.



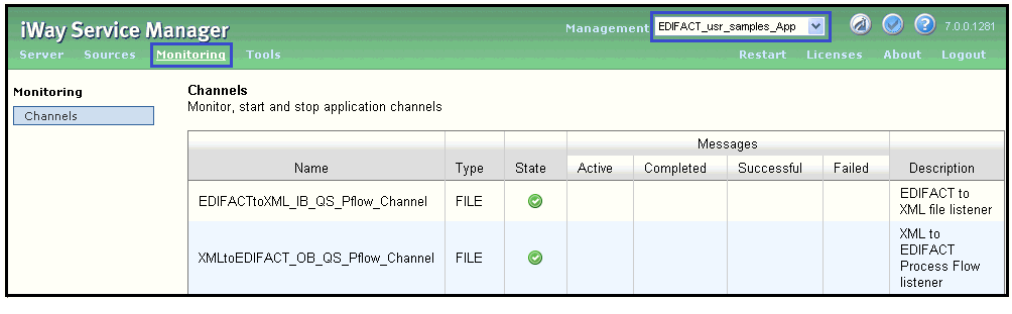
9. When *The page at localhost:9999 says:* window appears, click *OK* to proceed.



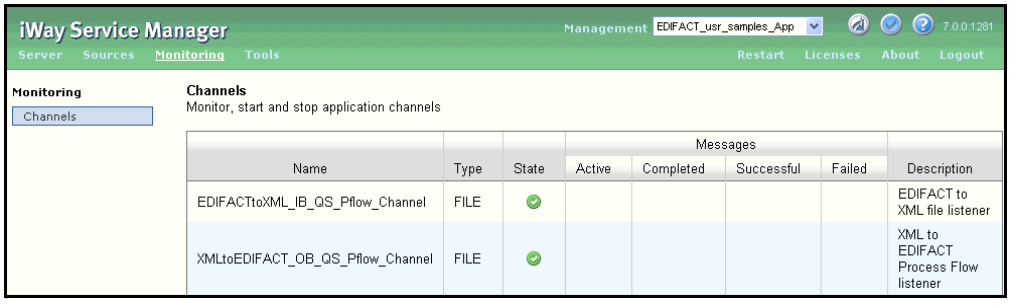
10. Once the application has successfully started, place your input data into the input location that is configured for the application.
11. Select the *EDIFACT_usr_samples_App [down]* application from the Management drop-down list.



12. Click the *Monitoring* link and observe the page. The deployed application channels *EDIFACTtoXML_IB_QS_Pflow_Channel* and *XMLtoEDIFACT_OB_QS_Pflow_Channel* are displayed, as shown in the following image.



The following image shows the inbound and outbound channels that are running in iSM. You can stop either channel and have only one channel running at a time as required.

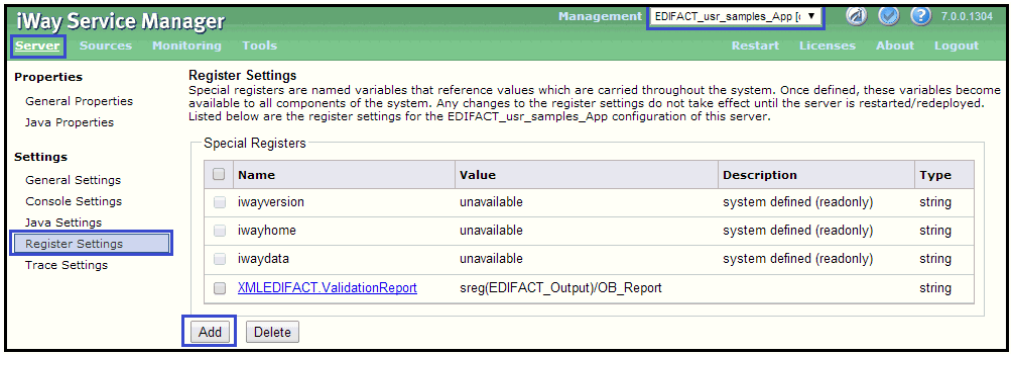


Setting Registers in the iWay Service Manager Administration Console

This section describes how to set Registers in the iWay Service Manager (iSM) Administration Console.

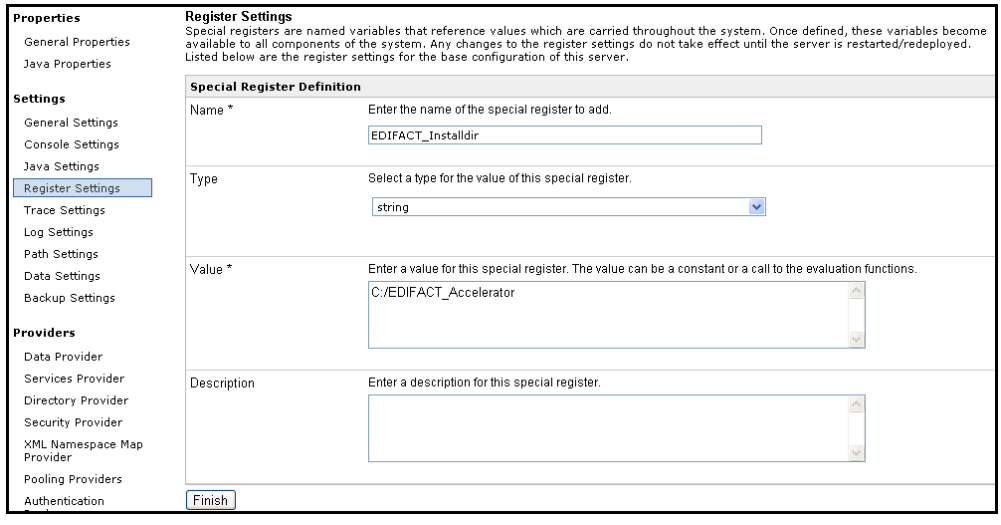
Procedure: How to Set Registers in the iWay Service Manager Administration Console

1. From the iSM Administration Console, select the *EDIFACT_usr_samples_App [down]* application from the Management drop-down list. Click *Server* in the top menu and then *Register Settings* in the left pane.

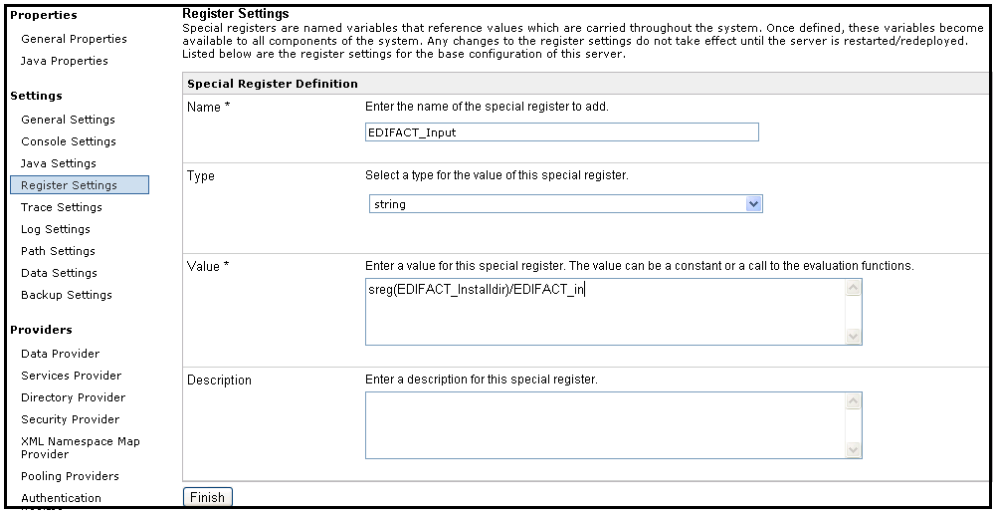


2. Click *Add*.

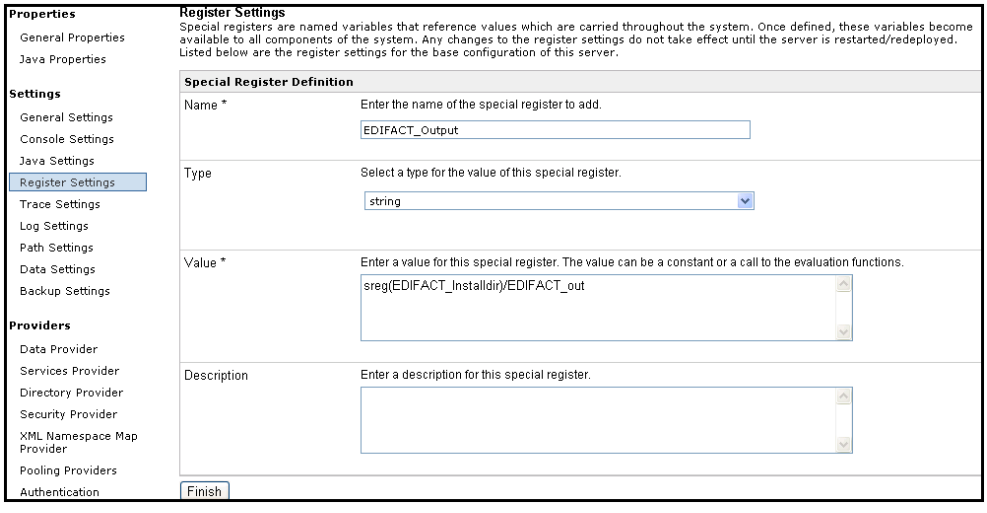
3. Add *EDIFACT_Installdir* and provide the appropriate values in the fields, as shown in the following image. Click *Finish*.



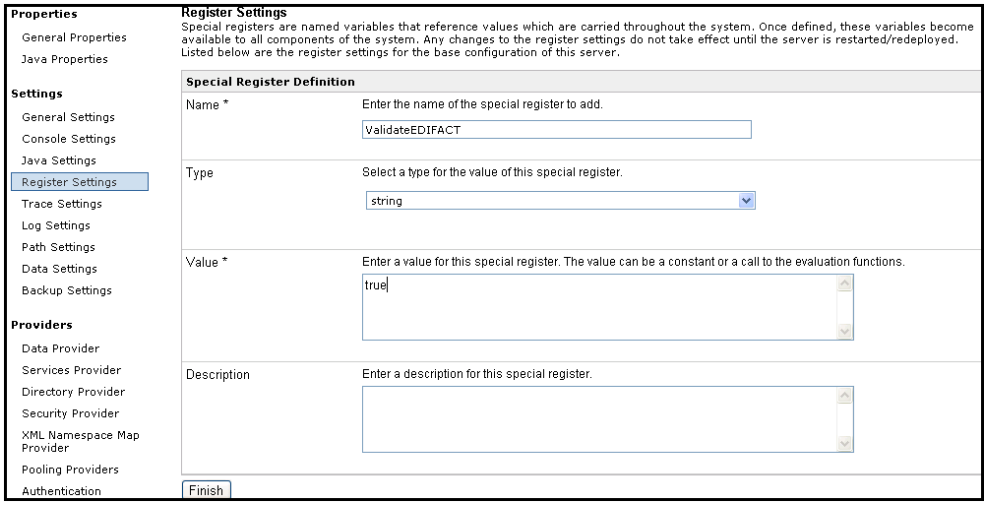
4. Add *EDIFACT_Input* and provide the appropriate values in the fields, as shown in the following image. Click *Finish*.



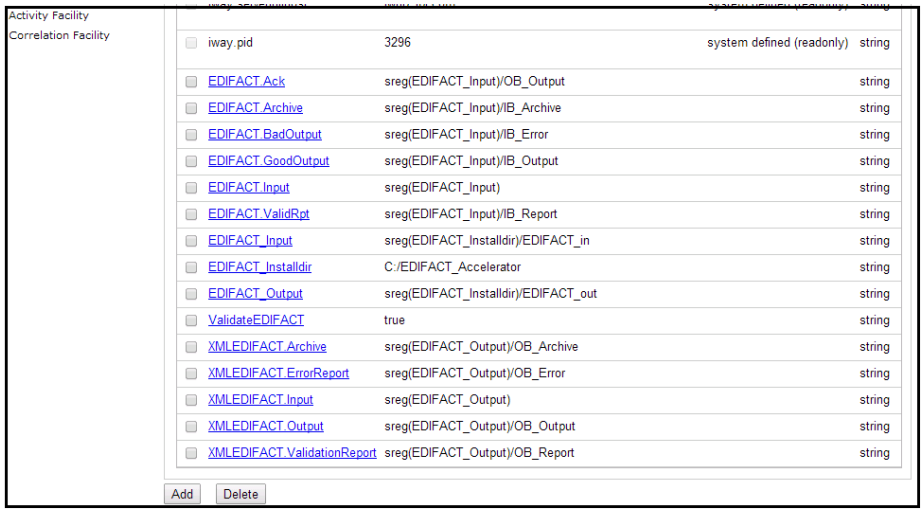
5. Add *EDIFACT_Output* and provide the appropriate values in the fields, as shown in the following image. Click *Finish*.



6. Add *ValidateEDIFACT* and provide the appropriate values in the fields, as shown in the following image. Click *Finish*.



The following image shows the summary of defined Registers.



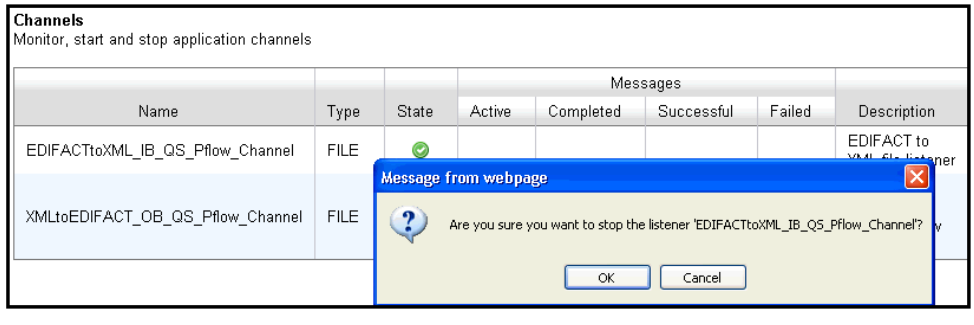
Note: If any changes are made to Registers after an application has started, you must restart that application for these changes to be applied.

Stopping Inbound (EDIFACT to XML) and Outbound (XML to EDIFACT) Processing

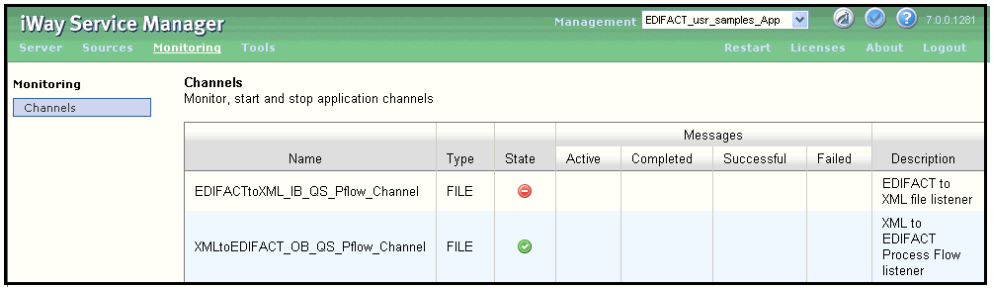
This section describes how to stop inbound (EDIFACT to XML) and outbound (XML to EDIFACT) processing.

Procedure: How to Stop Inbound (EDIFACT to XML) Processing

Click the *State* icon  adjacent to the inbound application channel (EDIFACTtoXML_IB_QS_Pflow_Channel) under Management\Monitoring and click *OK*, as shown in the following image.

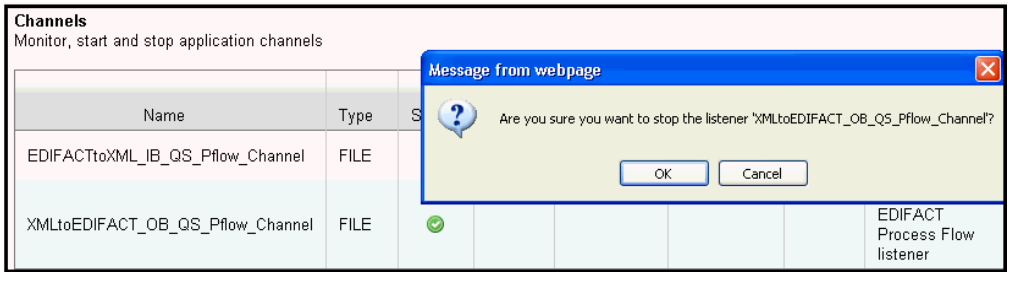


The inbound application channel will be stopped, as shown in the following image.

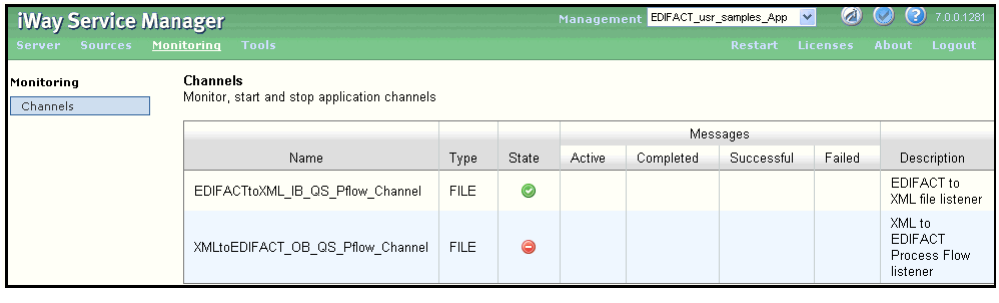


Procedure: How to Stop Outbound (XML to EDIFACT) Processing

Click the *State* icon  adjacent to the outbound application channel (XMLtoEDIFACT_OB_QS_Pflow_Channel) under Management\Monitoring and click *OK*, as shown in the following image.



The outbound application channel will be stopped, as shown in the following image.



Testing the Sample EDIFACT Applications

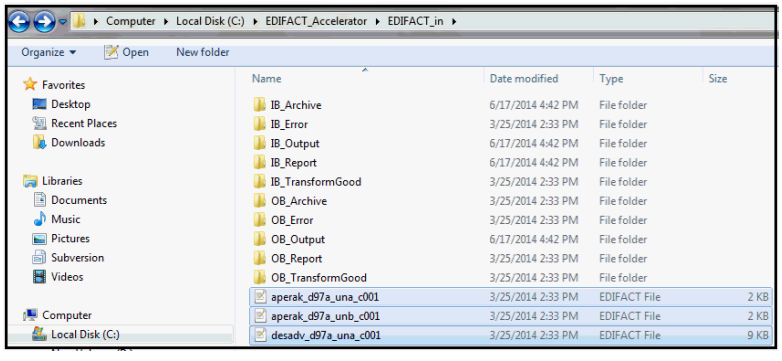
This section describes how to test the sample inbound (EDIFACT to XML) and outbound (XML to EDIFACT) applications.

Procedure: How to Test the Sample Inbound (EDIFACT to XML) Application

1. Copy the input test data to the following directory:

   `EDIFACT_Accelerator\EDIFACT_in`

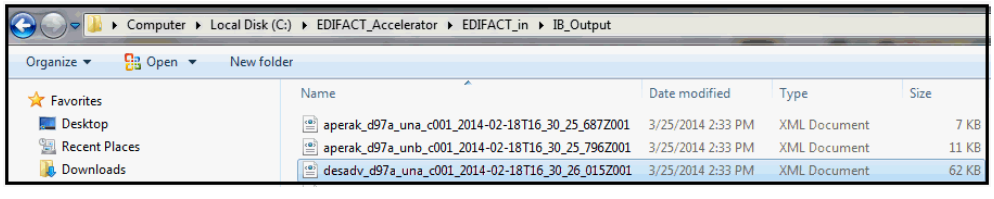
   For example:



2. Observe the transformed XML output in the following directory:

   `EDIFACT_Accelerator\EDIFACT_in\IB_Output`

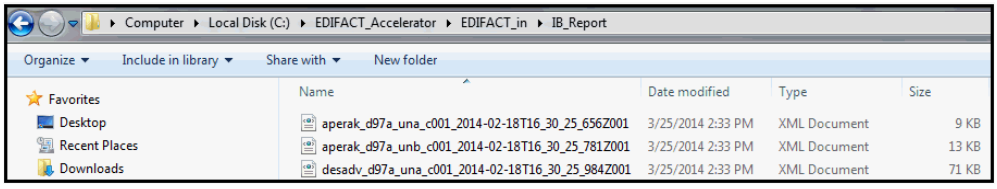
   For example:



3. Observe the Reports in the following directory:

   `EDIFACT_Accelerator\EDIFACT_in\IB_Report`

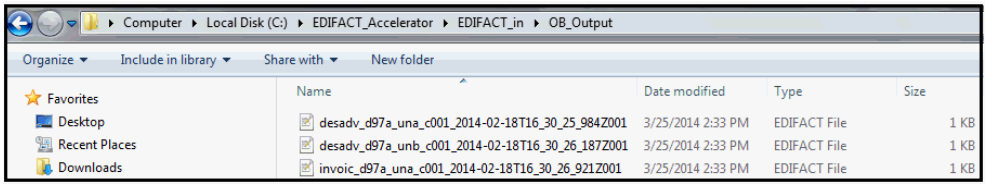
For example:



4. Observe the Acknowledgement in the following directory:

   `EDIFACT_Accelerator\EDIFACT_in\OB_Output`

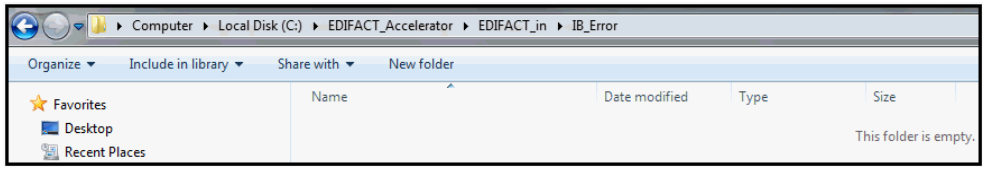
   For example:



5. If any Error occurs in the input test data then observe Error data in the following directory:

   `EDIFACT_Accelerator\EDIFACT_in\IB_Error`

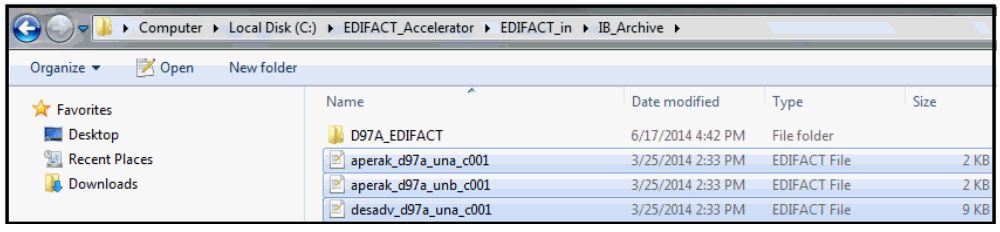
   For example:



6. After processing the input data that you place for transformation, a copy of input data will get stored in the following directory:

   `EDIFACT_Accelerator\EDIFACT_in\IB_Archive`

For example:

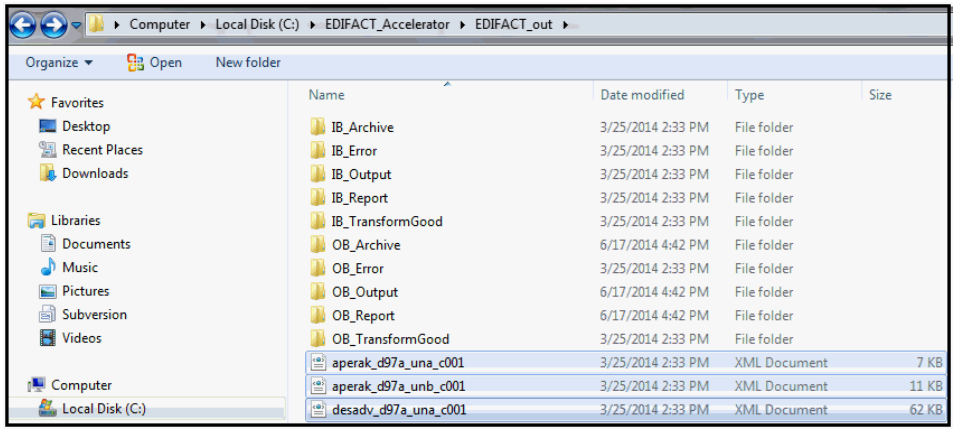


Procedure: How to Test the Sample Outbound (XML to EDIFACT) Application

1. Copy the input test data to the following directory:

   `EDIFACT_Accelerator\EDIFACT_out`

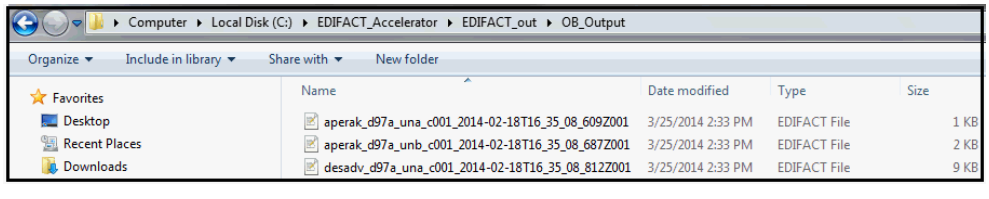
   For example:



2. Observe the transformed XML output in the following directory:

   `EDIFACT_Accelerator\EDIFACT_out\OB_Output`

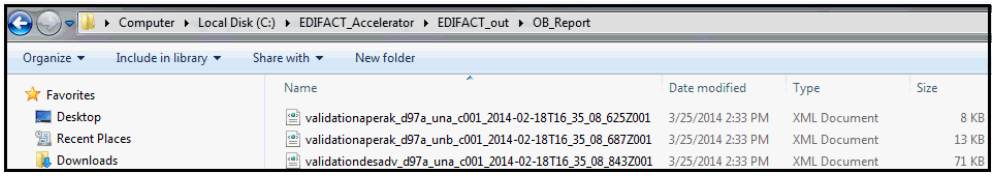
   For example:



3. Observe the Reports in the following directory:

`EDIFACT_Accelerator\EDIFACT_out\OB_Report`

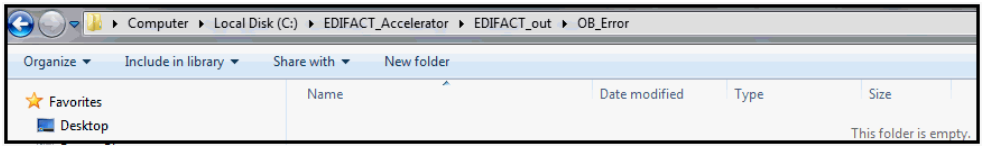
For example:



4. If any Error occurs in the input test data then observe Error data in the following directory:

   `EDIFACT_Accelerator\EDIFACT_out\OB_Error`

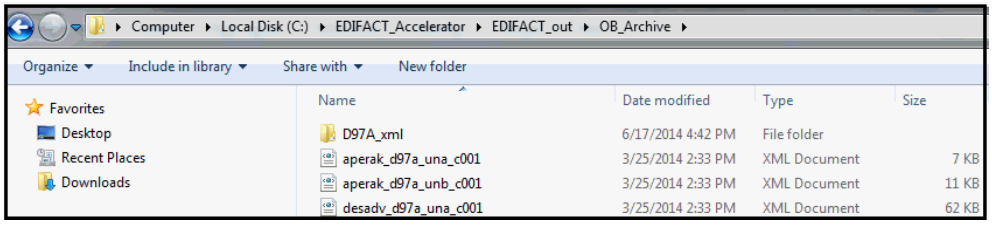
   For example:



5. After processing the input data that you place for transformation, a copy of input data will get stored in the following directory:

   `EDIFACT_Accelerator\EDIFACT_out\OB_Archive`

   For example:



Chapter 5

Inbound Processing: UN/EDIFACT to XML

The iWay Integration Solution for UN/EDIFACT includes iWay Service Manager. iWay Service Manager converts a document from UN/EDIFACT format to XML format, and validates it based on the published implementation guides of EDIFACT.

This chapter provides the information you need to understand and implement a basic inbound message flow.

- The inbound processing overview describes the iWay business components and the processing steps in the basic inbound message flow.
- The sample configuration contains detailed instructions for configuring the basic inbound message flow. This topic guides you through each step of the configuration procedure.

In this chapter:

EDIFACT Inbound Processing Overview

The inbound process converts an EDIFACT formatted document to an XML document.

In a basic message flow, inbound processing consists of the following components and steps. For an illustration of the components available in the construction of a message flow, see *Using a Channel to Construct a Message Flow* on page 19. You will define the components in the configuration instructions in *Sample Configuration for Inbound Processing: EDIFACT to XML* on page 61.

Inlet

- The listener picks up the incoming EDIFACT document.
- The preparser obtains the message type and version from the EDIFACT document, in order to select the appropriate transformation template name. The transformation template converts the original EDIFACT document to an XML representation of that document.

The preparser ensures that the document is converted to a structurally correct EDIFACT XML document. The transformation templates that are provided in the Ebix are used to transform the structure of the document.

The iWay Integration Solution for UN/EDIFACT supports the EDIFACTPreParser (com.ibi.preparsers.XDEDIFACTPreParser) and EDIFACTBatchSplitterPreParser (com.ibi.preparsers.XDEDIFACTBatchSplitter), which are described in *Preparsers* on page 23.

Validation

- The inbound EDIFACT document is validated for structure and content. The published EDIFACT standards and user implementation guides define element types (for example, numeric, alpha, or date) and describe business rules to apply for validation.

  For example, here is a typical date segment in an inbound EDIFACT document:

  ```
  DTM+37:20090214:102
  ```

  The value in DTM01 ("37") is validated against an allowed code list. The value in DTM02 ("20090214") is validated as a properly formatted date.

  In addition, the following business rule is applied: DTM02 is required if DTM01 is present (if there is a qualifier, there must be data).

Route

- After validation, you can apply any additional business logic to the document. You can use a single service or multiple services, passing the output of one service to the input of the next.

  In our basic message flow, the copy service redirects the output document to the destination.

  For details on available services, see the *iWay Service Manager User's Guide*.

- The acknowledgement service creates either a APERAK or CONTRL acknowledgement for the inbound document. The acknowledgement indicates that the document was received and validated for structure.

Outlets

Outlets define how messages leave a channel at the end of a process. In our basic sample channel, two outlets are configured:

- One outputs the XML format of the document. In a real case scenario, it would output the result of your business logic.

- The other outputs the functional acknowledgement. A functional acknowledgement is typically returned to the sender of the document.

Sample Configuration for Inbound Processing: EDIFACT to XML

This topic provides step-by-step instructions on how to configure a basic inbound message flow for the iWay Integration Solution for UN/EDIFACT. The message flow represents the movement and tasks in the conversion of a message from Electronic Data Interchange (EDIFACT) format to XML format and acknowledgement of the message.

Accessing the iWay Service Manager Administration Console

To access the iWay Service Manager Administration Console, you must first ensure that the iWay Service Manager service is running.

For instructions on starting iWay Service Manager, see the *iWay Service Manager User's Guide*.

Procedure: How to Access the iWay Service Manager Administration Console on Windows

1. From the Windows desktop, select *Start*, *All Programs*, *iWay 7.0 Service Manager*, and *Console*.

   or,

   from a browser such as Microsoft Internet Explorer, enter the following URL,

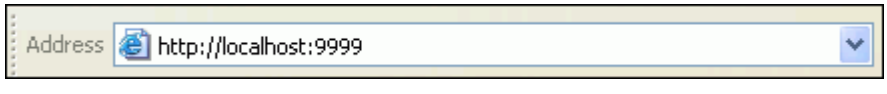
   `http://host:port`

   where:

   host
   Is the host machine on which iWay Service Manager is installed. The default value is localhost.

   port
   Is the port number on which iWay Service Manager is listening. The default value is 9999.

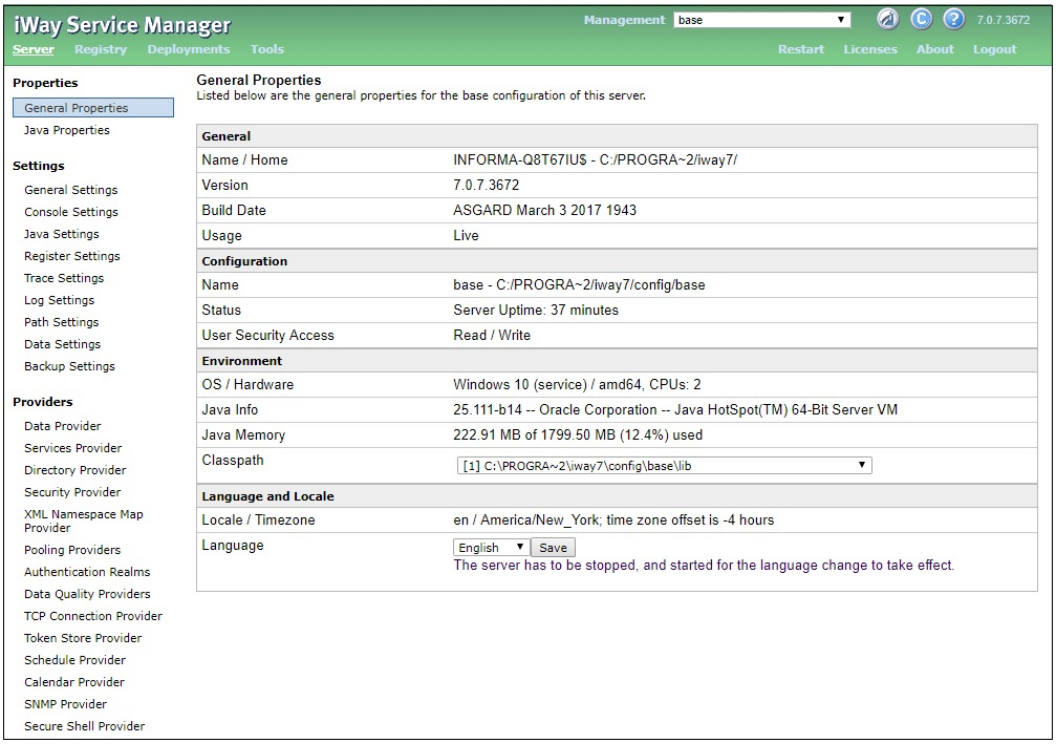
   The following image shows the URL with the default values.

2. When prompted, enter your user name and password, and click OK.

   Note: The default user name and password is *iway*.

The iWay Service Manager Administration Console opens, as shown in the following image.



Adding an Ebix to the Registry

The iWay e-Business Information Exchange (Ebix) framework supplies several Ebix files for the iWay Integration Solution for UN/EDIFACT.

An Ebix file for EDIFACT is named EDIFACT_*transaction_set*.ebx, where *transaction_set* is the transaction set number. For example, the Ebix file for EDIFACT transaction set D01B is named EDIFACT_D01B.ebx.

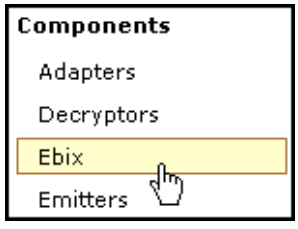
For details on the supported EDIFACT transaction sets, see *UN/EDIFACT Supported Versions* on page 143.

This topic describes how to add an Ebix to the Registry on Windows and UNIX.

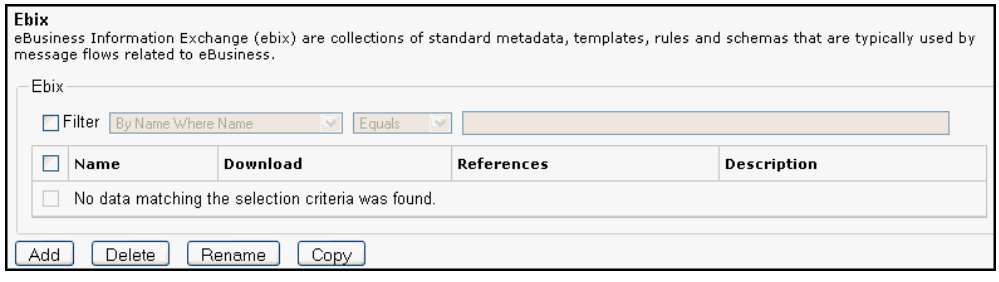
Procedure: How to Add an Ebix to the Registry on Windows

1. To access the Registry, select the *Registry* option in the top pane.

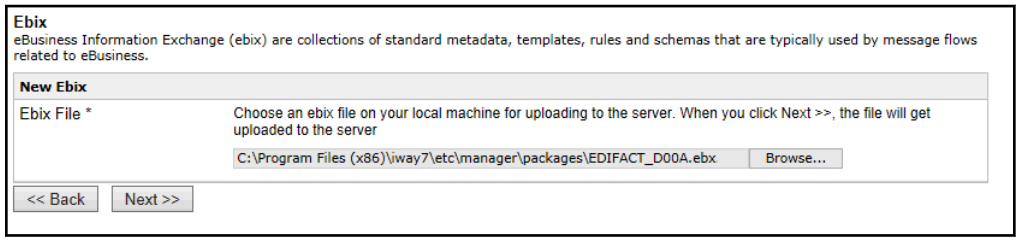
2. Under Components in the left pane of the Registry, select *Ebix*.



The Ebix pane opens, as shown in the following image.



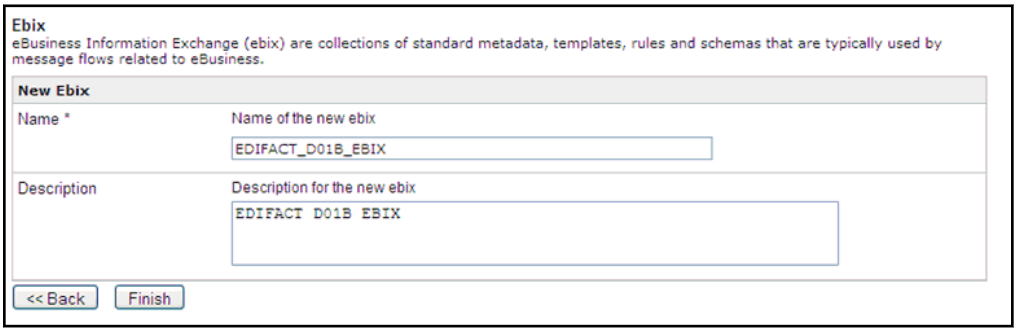
3. Click *Add* to add a new Ebix.
The New Ebix pane opens.



4. Browse to the directory in which the Ebix is located and select the name of the file, for example, *EDIFACT_D01B.ebx*.

5. Once you have selected the Ebix, click *Next*.

You are prompted for the name of the Ebix and an optional description.



6. Enter a name for the Ebix, for example, *EDIFACT_D01B_EBIX*, and an optional description, such as *EDIFACT D01B EBIX*.

   Note: This step must be repeated for each Ebix message set that is added to the Registry.

7. Click *Finish*.

   On the Ebix pane, you will see that the Ebix was successfully added. Later you will associate it with the channel for inbound processing.

Procedure: How to Add an Ebix to the Registry on UNIX

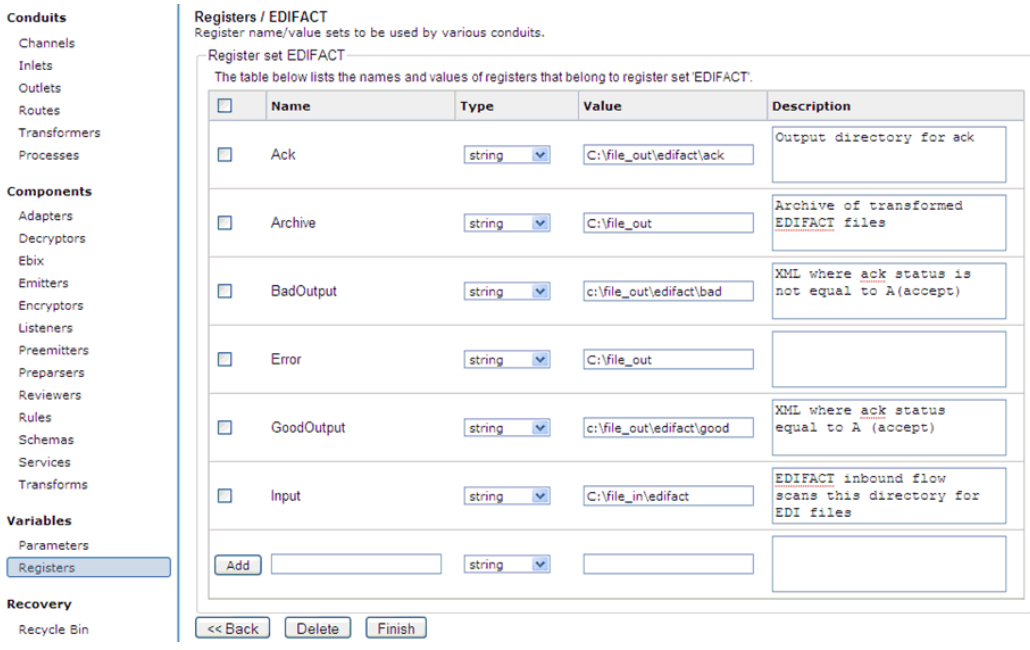
Depending on your system configuration, there are two methods that you can use to add an Ebix to the Registry on UNIX.

- ❏ If you have a web browser on the UNIX machine, follow the instructions for Windows.
- ❏ Use FTP to download the Ebix from the *iWay7/etc/manager/packages* directory to your Windows machine and follow the instructions for Windows.

Adding Special Register Sets

In iWay Service Manager, a special register is a name-value pair that defines a variable that is carried throughout the system. Once defined, this variable is available to all components of the system. Within the EDIFACT components, a Best Practice is to use special registers to define inputs and outputs. When packages containing channels are migrated between systems, the only changes required to deploy in the new location is to modify these special registers and build the channel. Channels may have many locations and this practice will minimize the effort required to migrate. For a complete list of system special registers that are provided, see the *iWay Service Manager Programmer's Guide*. For more information on defining a special register of your own, see the *iWay Service Manager User's Guide*.

The sample inbound channel uses a set of special registers. For example:



Procedure: How to Add a Special Register Set to Your Channel

To add a special register set to your channel:

1. In the left console pane of the Registry menu, select *Channels*.

   The Channels pane opens.

2. In the row for your channel, click *Regs* for the channel you want to modify.

   The Assign register pane opens.

3. Select a register and click *Finish*.
4. Click *Back* to return to the Channels pane.

Defining an Inlet

An inlet defines how a message enters a channel. It typically contains a:

- Listener. A listener is a component that picks up input on a channel from a configured end point.
- Decryptor. A decryptor is a component that applies a decryption algorithm to an incoming message and verifies the security of the message. The configuration example in this topic does not include a decryptor, which is an optional component.

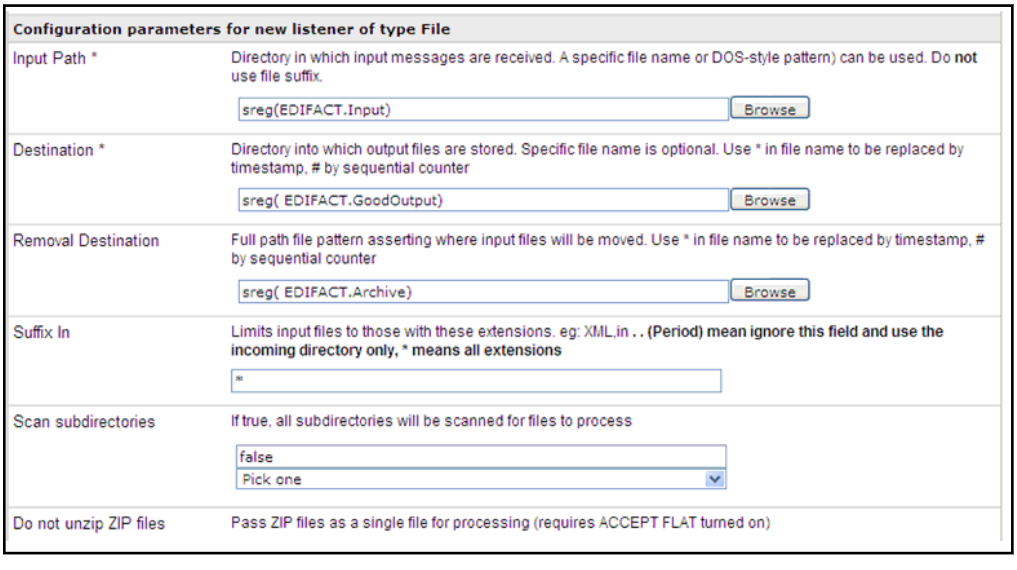
- One or more preparsers. A preparser is a component that converts incoming messages into processable documents. Typically a preparser converts a document into XML format. Other preparsers may perform data decryption or reformatting.

Procedure: How to Add a Listener

1. From the Registry menu options on the left pane, select *Listeners* under Components.
2. On the Listeners pane on the right, click *Add* to add a new listener.
3. For the purpose of this example, we will show the configuration with a File listener. For details on supported protocols, see the *iWay Service Manager Protocol Guide*.

   Select *File* from the Type drop-down list and click *Next*.

   The configuration parameters pane opens.

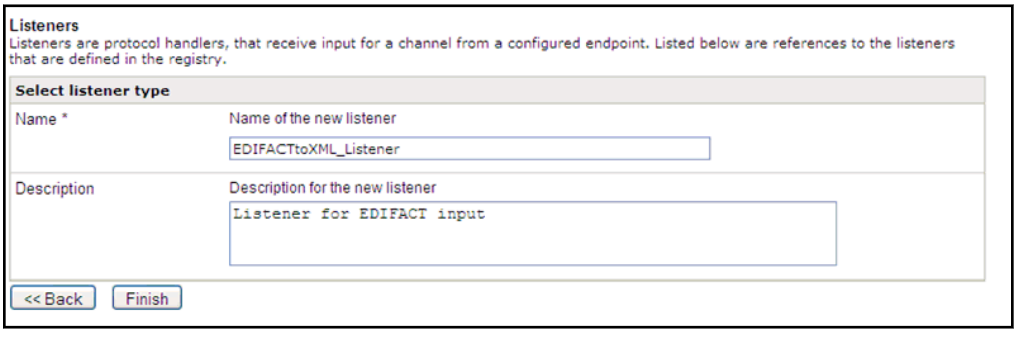


4. Supply configuration parameters for the new File listener as follows. An asterisk indicates that a parameter is required. For parameters not listed in the following table, accept the default value.

Parameter	Value
Input Path *	sreg(EDIFACT.Input) This value is a special register that uses a defined directory in which input messages are received. Make sure that you have created this directory; otherwise, errors will occur during deployment.
Destination *	sreg(EDIFACT.GoodOutput) This value is a special register that uses a defined directory in which output files are stored after transformation. Make sure that you have created this directory; otherwise, errors will occur during deployment.
Removal Destination	sreg(EDIFACT.Archive) This value is a special register that uses a defined directory to which input messages are moved if they fail during transformation. Make sure that you have created this directory; otherwise, errors will occur during deployment. It is recommended to configure a removal destination when you are constructing a basic channel.
Suffix In	* Input files with any file extension are allowed.
Suffix Out	xml The extension for output files is .xml.

5. Click *Next*.

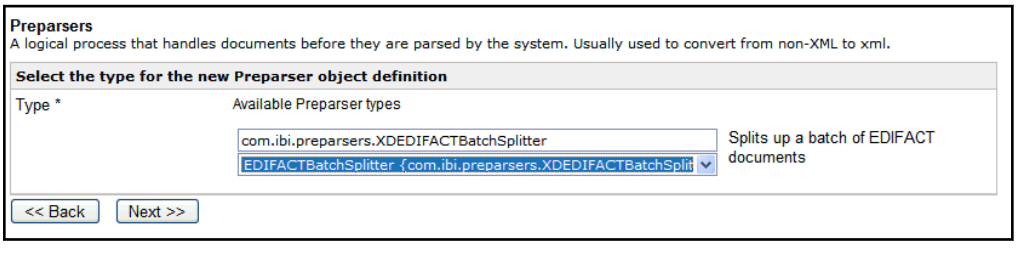
You are prompted for the name of the listener and an optional description.



6. On the Listeners pane, enter the name of the new listener, *EDIFACTtoXML_Listener*, and an optional description. Then click *Finish* to add the listener.
   In a later step, you will associate this listener with the inlet.

Procedure: How to Add a Batch Splitter Preparser

1. From the Registry menu options, select *Preparsers* under Components.
2. On the Preparsers pane, click *Add* to add a new preparser.
   You are prompted for the type of preparser.



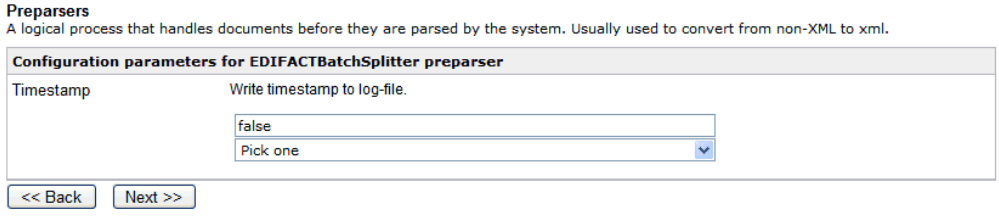
3. Select *EDIFACTBatchSplitter (com.ibi.preparsers.XDEDIFACTBatchSplitter)* from the Type drop-down list.

   The EDIFACTBatchSplitter internally splits EDIFACT input files into multiple individual EDIFACT transactions. The file will then be processed into one XML output when passed to the EDIFACT Preparser. You can also use the EDIFACTBatchSplitter if there is only one EDIFACT transaction.

   For details on the supported EDIFACT transaction sets, see *UN/EDIFACT Supported Versions* on page 143.

4. Click *Next*.

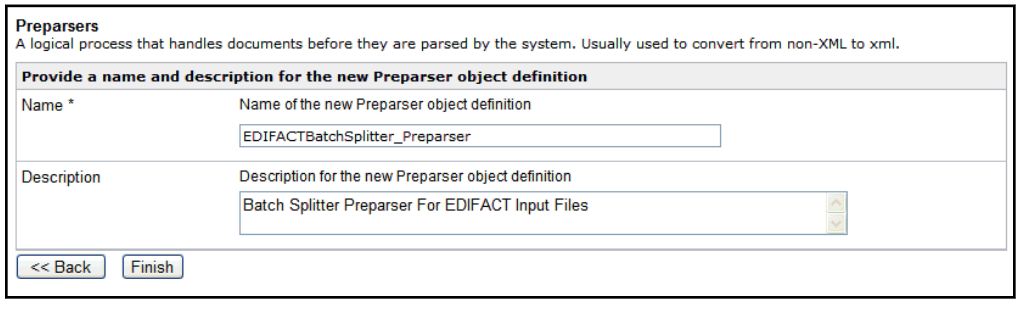
The Preparsers configuration parameters pane opens.



Disabled by default, the Timestamp option writes a timestamp to the log file. When enabled, the log file will display the start and end time of the file transformation and the input file name that is used. This feature is useful in development or debugging environments when processing batches of files. When the transaction log is not in use (for example, in a production mode) then this information is available in the Activity Log.

5. Click *Next*.

   You are prompted for a name and optional description for the new preparser.

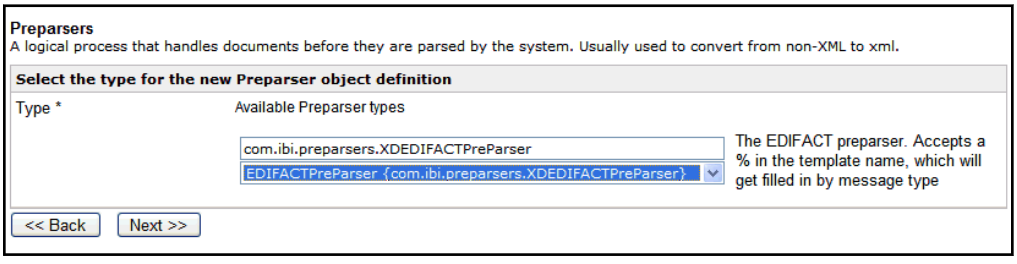


6. Enter a name for the new preparser, for example, *EDIFACTBatchSplitter_Preparser*, and an optional description.
7. Click *Finish* to add the preparser.
   In the next procedure, you will associate this preparser with an inlet.

Procedure: How to Add a Preparser

1. From the Registry menu options, select *Preparsers* under Components.
2. On the Preparsers pane, click *Add* to add a new preparser.

You are prompted for the type of preparser.



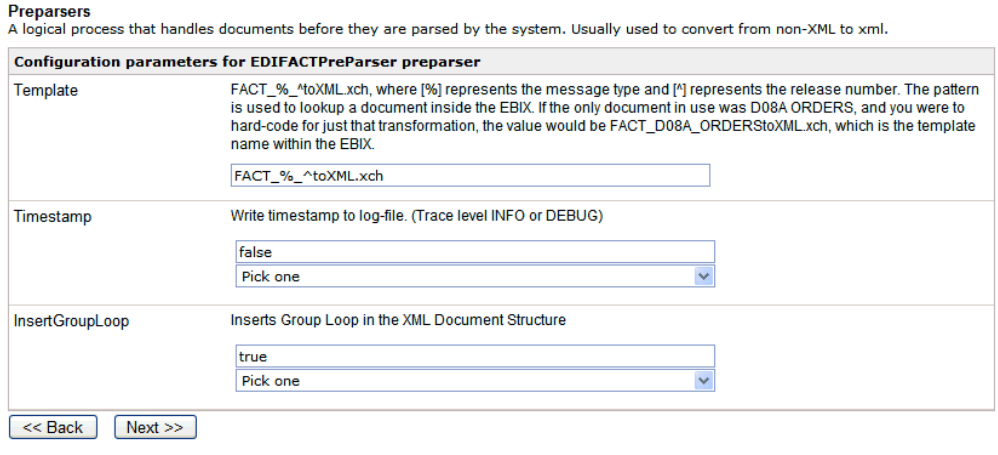
3. Select *EDIFACTPreParser (com.ibi.preparsers.XDEDIFACTPreParser)* from the Type drop-down list.

   The EDIFACTPreparser is used with the EDIFACTBatchSplitter. It processes an EDIFACT input file which has been split by the BatchSplitter into multiple XML output files. One XML output file is produced for each EDIFACT transaction. EDIFACTPreparser should be used when there is only one EDIFACT transaction being processed.

   For details on the supported EDIFACT transaction sets, see *UN/EDIFACT Supported Versions* on page 143.

4. Click *Next*.

   The Preparsers configuration parameters pane opens.



   The Template field is used to locate the template in the Ebix, which is used during the transformation from EDIFACT format to XML format.

Disabled by default, the Timestamp option writes a timestamp to the log file. When enabled, the log file will display the start and end time of the file transformation and the input file name that is used. This feature is useful in development or debugging environments when processing batches of files. When the transaction log is not in use (for example, in a production mode) then this information is available in the Activity Log.

The InsertGroupLoop parameter inserts a group loop node in the generated XML document. By default, this parameter is set to *true*.

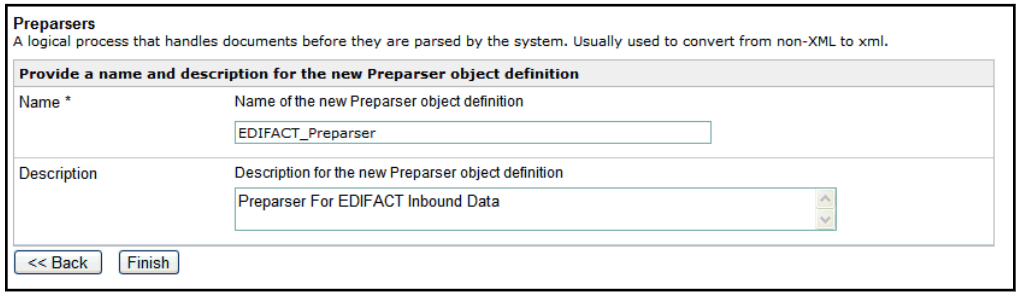
5. In the Template field, enter *FACT_%_^toXML.xch*.

   The preparser obtains the message type and version information from the EDIFACT input document. In the parameter, the character "%" represents the message type, and the character "^" represents the version.

   For example, if the message type for the EDIFACT document is INVOIC and the version is D01B, the constructed template name is FACT_INVOIC_D01BtoXML.xch.

6. Click *Next*.

   You are prompted for a name and optional description for the new preparser.



7. Enter a name for the new preparser, for example, *EDIFACT_Preparser*, and an optional description.
8. Click *Finish* to add the preparser.
   In the next procedure, you will associate this preparser with an inlet.

Procedure: How to Define an Inlet

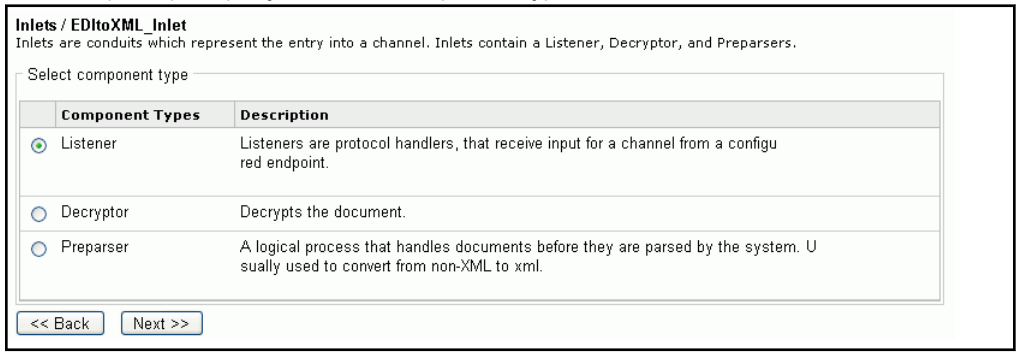
Now that you have added a File listener and splitter preparser to the Registry, you are ready to add and define an inlet. You will associate the previously created listener and preparser with the inlet.

1. From the Registry menu options, select *Inlets* under Conduits.
2. On the Inlet Definitions pane, click *Add* to add an inlet.

3. On the New Inlet Definition pane, enter the name of the new inlet and an optional description, as shown in the following table. Then click *Finish* to add the inlet.

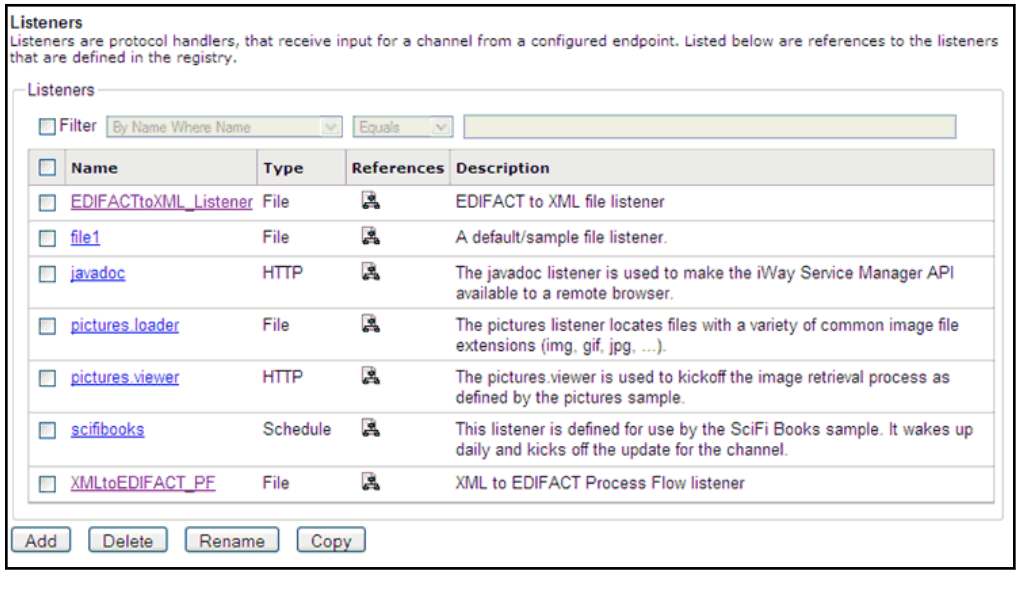
Parameter	Value
Name *	EDIFACTtoXML_Inlet
Description	Inlet for EDIFACT to XML.

4. On the Construct Inlet pane, click *Add* to associate the listener and preparser with the inlet.
The next pane prompts you for the component type.



5. Select *Listener* and click *Next*.

The next pane prompts you to select a listener.



6. Select *EDIFACTtoXML_Listener*, which is the listener you added earlier, and click *Finish*.

   The listener is associated with the inlet. Now you need to associate the preparser created earlier with the inlet.

Defining a Route

For this sample channel configuration, you will define a route that will invoke the EDIFACT to XML validation process flow. The outcome of the validation process flow will place valid XML data in a defined output folder. Invalid EDIFACT data will be routed to an errors folder. A validation report will also be sent to the appropriate folder.

This section describes how to create a validation process flow using iIT Designer and bind it to a sample outbound channel as a route.

Procedure: How to Create a New Project and Start the Process Flow

To create a new project and start the process flow using iIT Designer:

1. From the Windows Start menu select *Programs*, *iWay 7.0 Service Manager*, *tools*, and then *iWay Integration Tools (iIT) Designer*.
2. Connect to the repository from which you want to work, for example, iWay.
3. Right-click the *iWay* node and select *New Project* from the drop-down list.

   The Designer Project Information dialog box opens, prompting you for a project name and optional description.
4. In the Name field, type a project name, for example, *Test*.

   In the Description field, type a brief description (optional) to describe the project.
5. Click *Next*.

   The Designer Project Bindings dialog box opens.
6. To create the project in the iWay Registry, select *iWay Registry* and click *Finish*.
   The choice of project association depends on where you intend to publish (deploy) your process flow. If you are developing a process flow for use as part of a channel, you must publish it to the iWay Registry for subsequent deployment.
7. The Test project node appears under the repository in which it was created (in this example, it appears under iWay).
8. To save the project to the repository, right-click the project node and select *Save* from the context menu.
9. Expand the Test project node to expose the project elements (Processes, Services, Transforms, and so on).

10. Right-click the *Processes* folder and select *New Process* from the drop-down list.

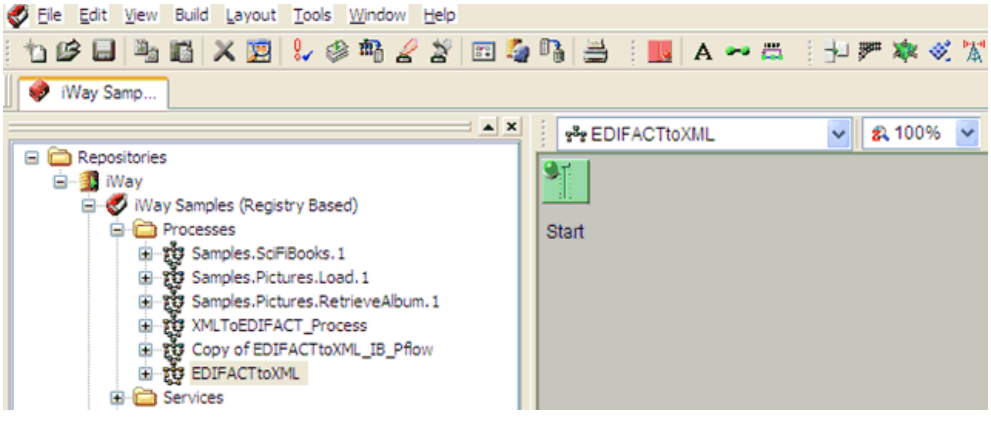
    The iWay Process Configuration dialog box opens.

11. In the Name field, type *EDIFACTToXML* as the process flow name.

    In the Description field, type a brief description (optional).

12. Click *Finish*.

    The new EDIFACTToXML node appears under the Processes folder, and the workspace displays a Start object.



You are ready to build the EDIFACTToXML validation process flow by configuring objects to it and specifying their relationships.

Procedure: How to Configure Objects for the Process Flow

To configure objects for the process flow using iIT Designer:

1. Drag and drop the Service object from the toolbar to the workspace.

   The New Service Object dialog box opens.

2. In the Name field, type *XDEDIFACTAckAgent*, and a brief description (optional) in the Description field.

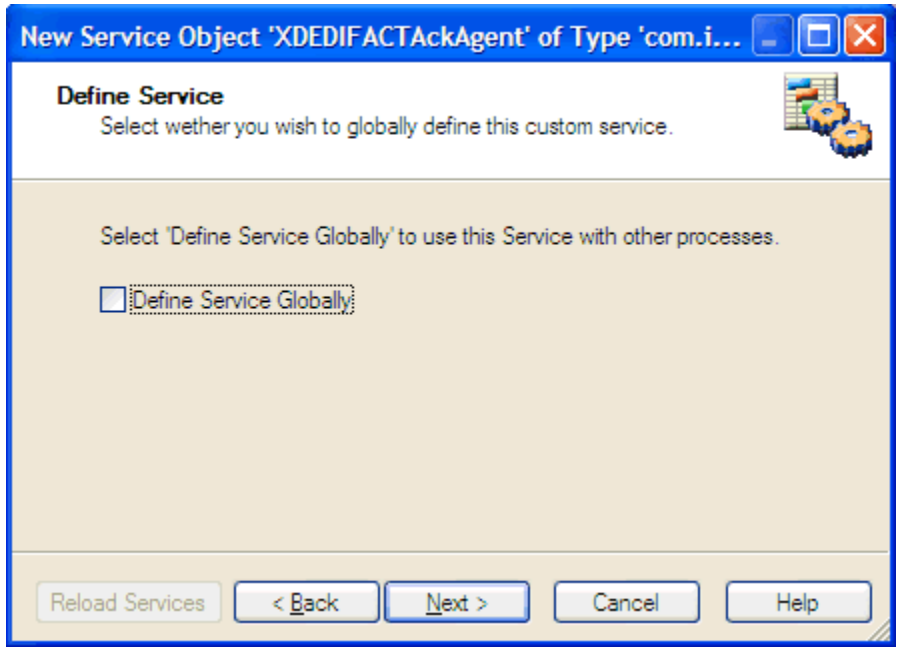
3. Click *Next*.

   The Service Type dialog box opens.

4. Select *Class Name* and enter *com.ibi.agents.XDEDIFACTAckAgent*.

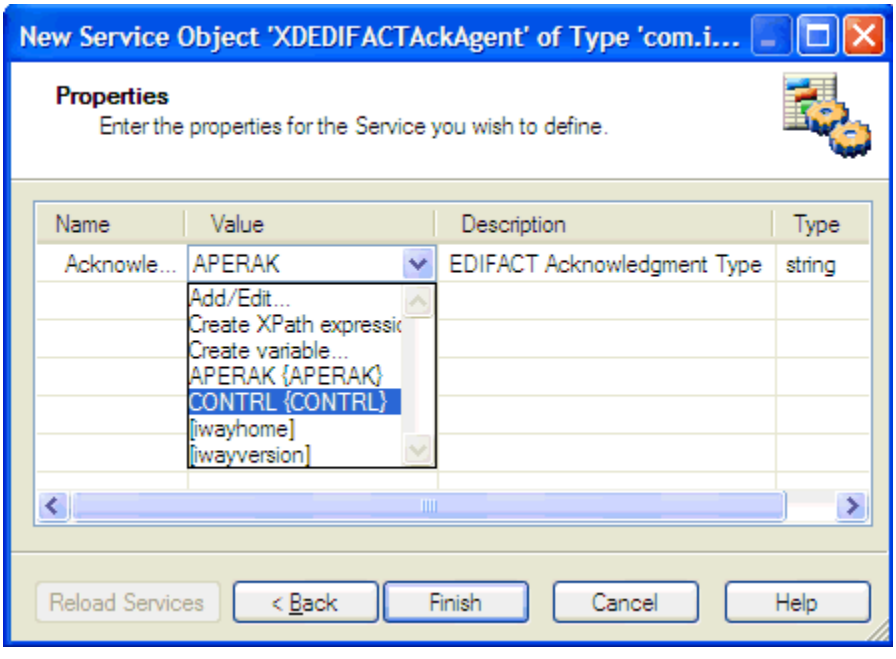
5. Click *Next*.

The Define Service dialog box opens.



6. Click *Next*.

The Properties dialog box opens.



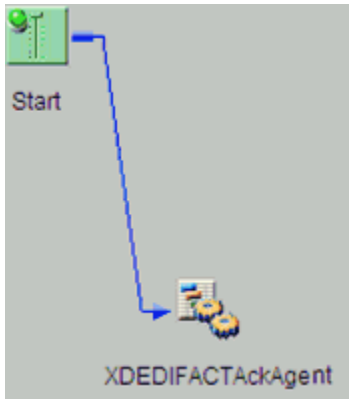
7. For the Acknowledgment type parameter, select *CONTRL* from the drop-down list.
8. Click *Finish*.

   The new Service object (XDEDIFACTAckAgent) appears in the workspace.
9. Select the *Start* object, right-click the *XDEDIFACTAckAgent* object, and select *Relation* from the context menu.

   The Line Configuration dialog box opens.
10. From the Event drop-down list, select *OnCompletion* and click *OK*.

    This option indicates that there are no conditions that affect the path, and that the path between the two objects will always be followed.

A line appears between the objects to indicate that a relationship has been established.



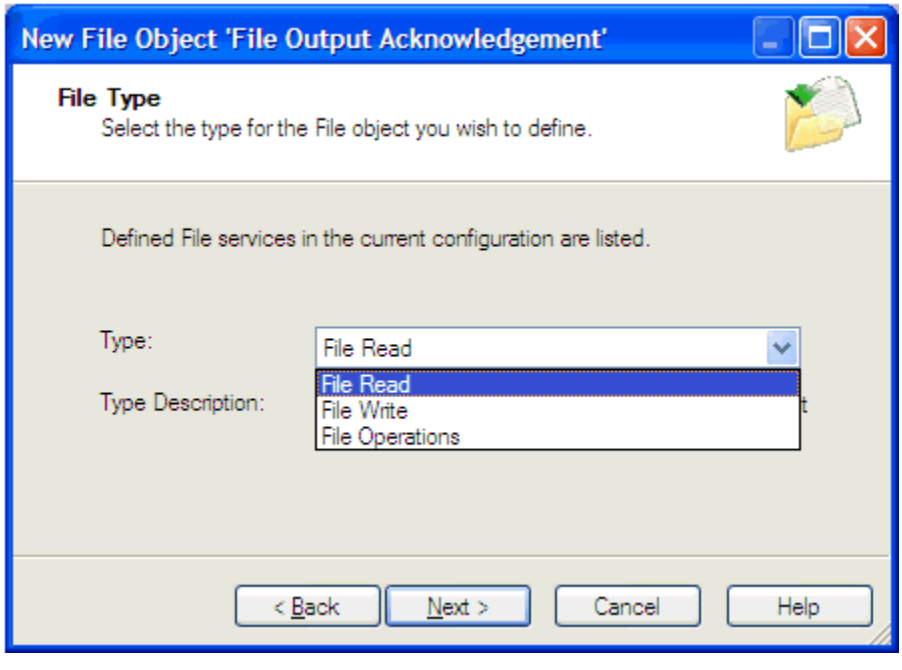
11. Drag and drop the File object from the toolbar to the workspace.

    The New File Object dialog box opens.

12. In the Name field, type *File Output Acknowledgement*, and a brief description (optional) in the Description field.

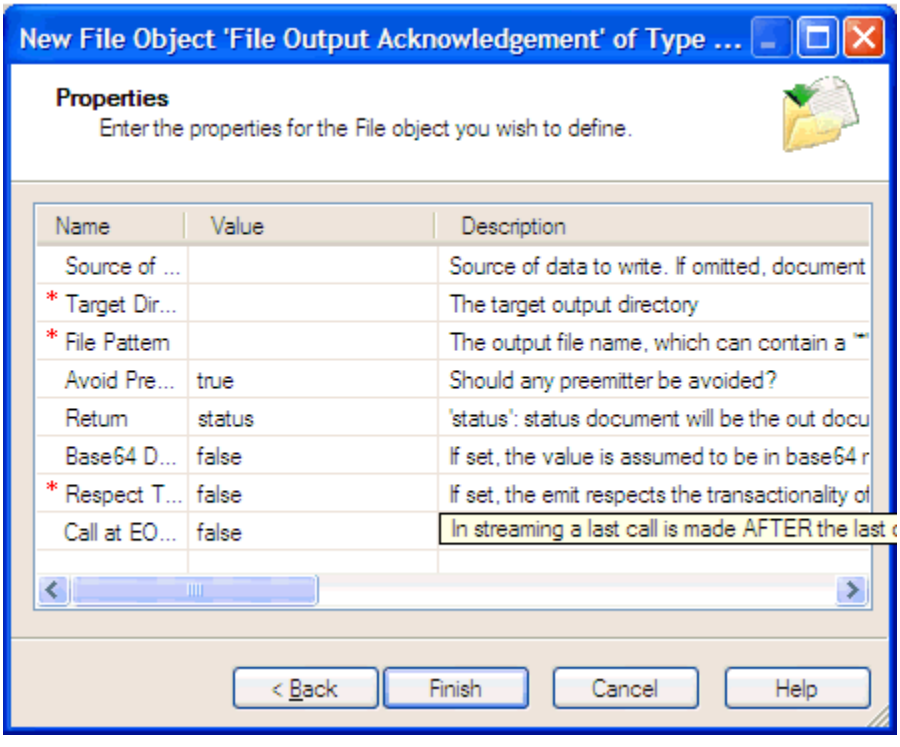
13. Click *Next*.

    The Define File Type dialog box opens.



14. Select *File Write* from the Type drop-down list and click *Next*.

    The Properties dialog box opens.



15. For the Target Directory parameter, enter the following location where the data will be written:

    ```
    Sreg(EDIFACT.Ack)
    ```

16. For the File pattern parameter, enter the following:

    ```
    Sreg(basename)__*.edifact
    ```

17. For the Respect Transactionality parameter, select *true* from the drop-down list.
18. Click *Finish*.

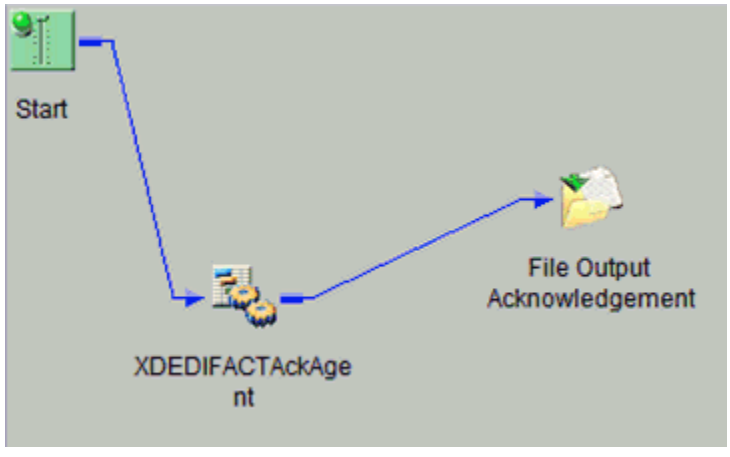
    The new File object, *File Output Acknowledgement*, is added to the workspace area.

19. Select the *XDEDIFACTAckAgent* object, right-click the *File Output Acknowledgement* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

20. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



21. Drag and drop the End object from the toolbar to the workspace.

    The End Name and Description dialog box opens.

22. In the Name field, type *End*, and a brief description (optional) in the Description field.

23. Click *Next*.

    The End Name Schema dialog box opens.

24. Since no schemas are used in this processing path (that is, the process flow will not be exposed as a web service), from the Schema drop-down list, select *None*.

25. Click *Next*.

    The Properties dialog box opens.

26. Click *Finish* to accept the default values and close the dialog box.

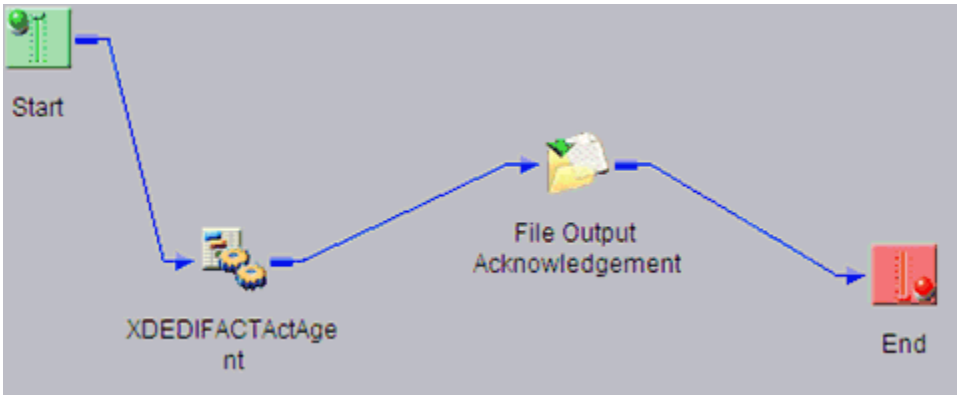
    The new End object appears in the workspace.

27. Select the *File Output Acknowledgement* object, right-click the *End* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

28. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



29. Drag and drop the End object from the toolbar to the workspace.

    The End Name and Description dialog box opens.

30. In the Name field, type *End1*, and a brief description (optional) in the Description field.

31. Click *Next*.

    The End Name Schema dialog box opens.

32. Since no schemas are used in this processing path (that is, the process flow will not be exposed as a web service), from the Schema drop-down list, select *None*.

33. Click *Next*.

    The Properties dialog box opens.

34. Click *Finish* to accept the default values and close the dialog box.

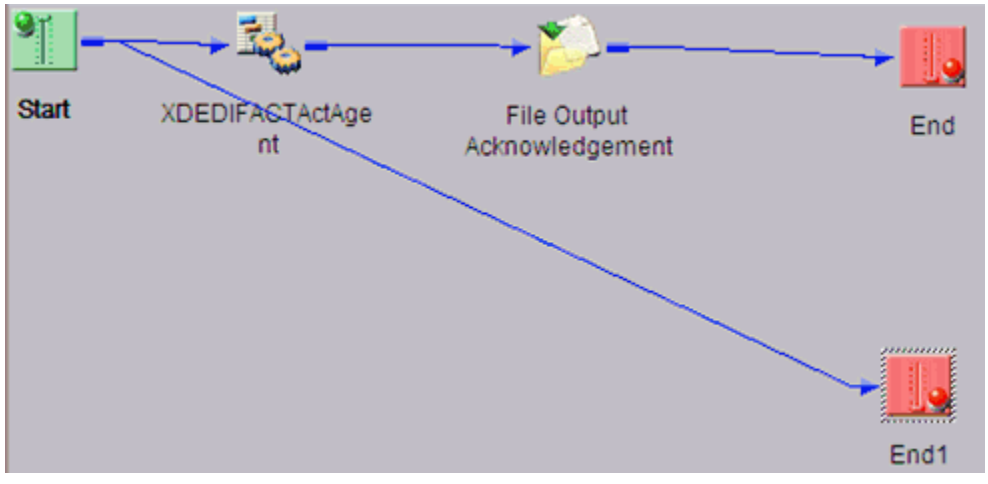
    The new End1 object appears in the workspace.

35. Select the *Start* object, right-click the *End1* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

36. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



37. Drag and drop the Service object from the toolbar to the workspace.

    The New Service Object dialog box opens.

38. In the Name field, type *Validation Report*, and a brief description (optional) in the Description field.

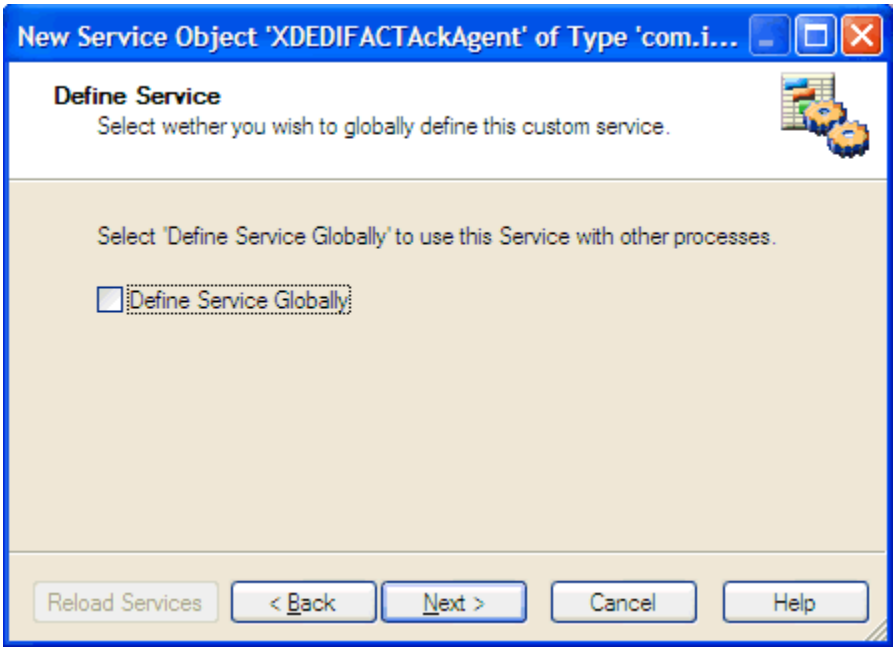
39. Click *Next*.

    The Service Type dialog box opens.

40. Select *Class Name* and enter *com.ibi.agents.XDEDIFACTValidationReportAgent*.

41. Click *Next*.

The Define Service dialog box opens.



42. Click *Next*.

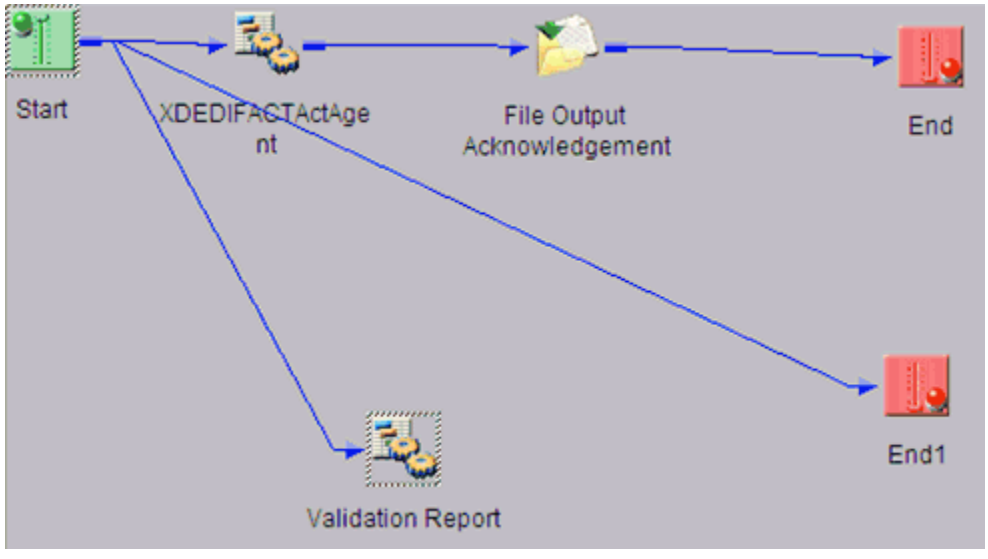
    The Validation Report object appears in the workspace.

43. Select the *Start* object, right-click the *Validation Report* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

44. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



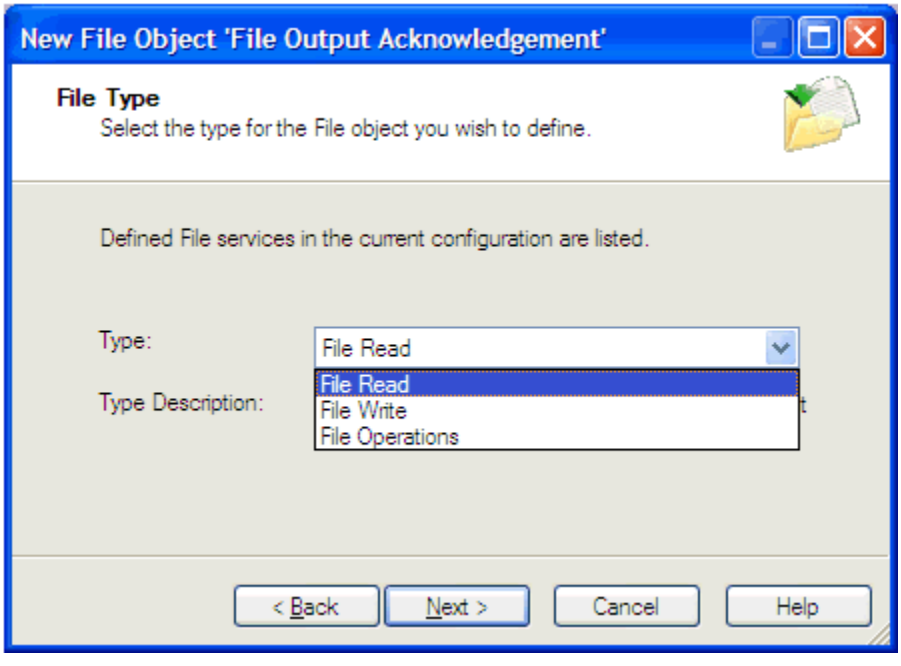
45. Drag and drop the File object from the toolbar to the workspace.

    The New File Object dialog box opens.

46. In the Name field, type *File Output Validation Report*, and a brief description (optional) in the Description field.

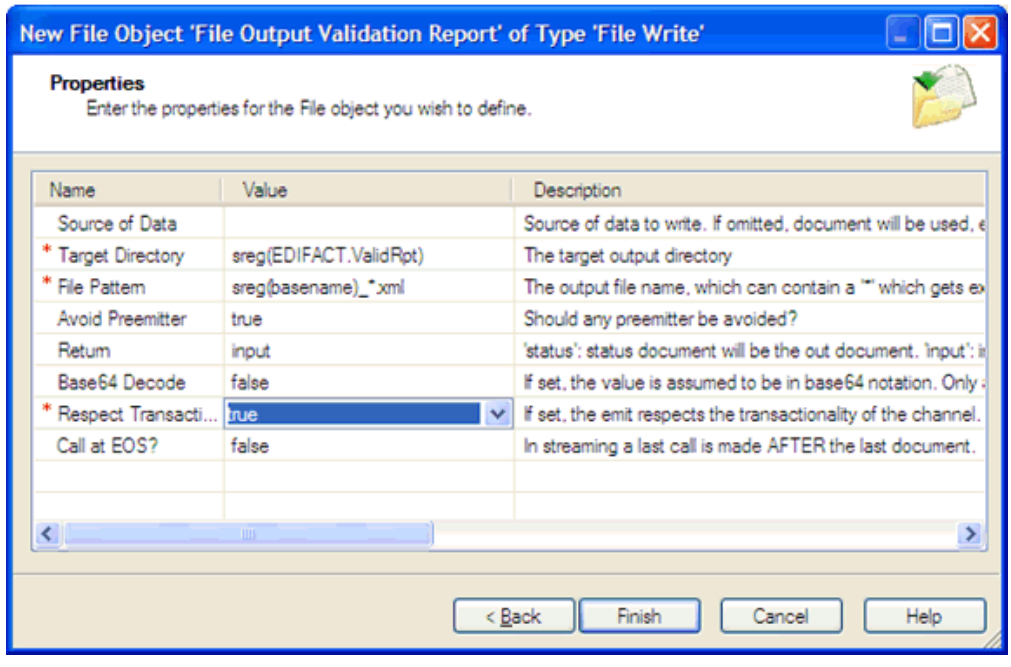
47. Click *Next*.

The Define File Type dialog box opens.



48. Select *File Write* from the Type drop-down list and click *Next*.

The Properties dialog box opens.



49. For the Target Directory parameter, enter the following location where the data will be written:

    ```
    sreg(EDIFACT.ValidRpt)
    ```

50. For the File pattern parameter, enter the following:

    ```
    sreg(basename)_*.xml
    ```

51. For the Respect Transactionality parameter, select *true* from the drop-down list.
52. Click *Finish*.

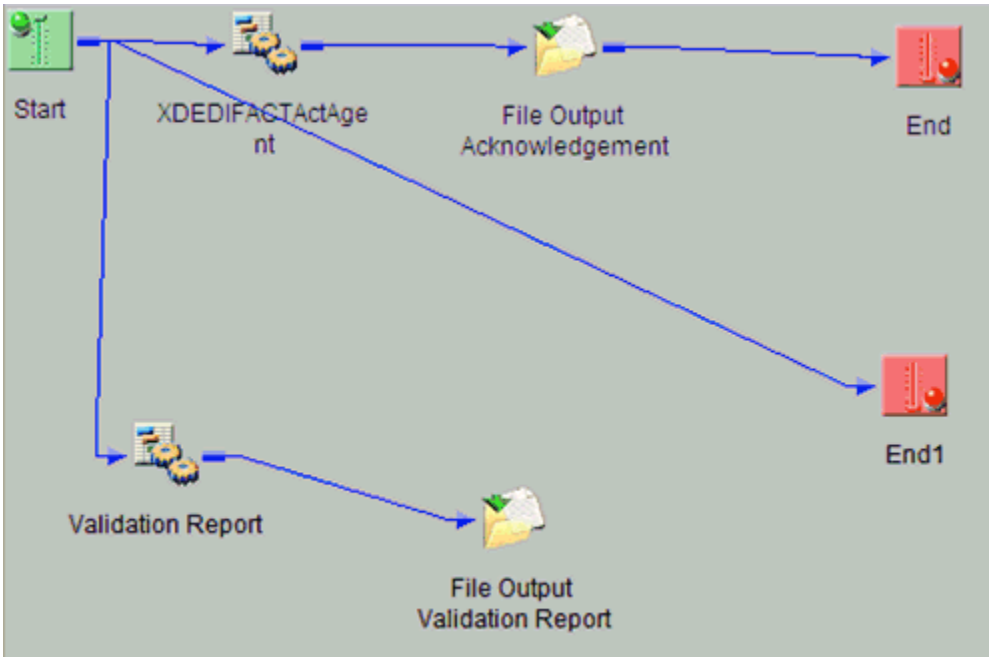
    The new File object, *File Output Validation Report*, is added to the workspace area.

53. Select the *Validation Report* object, right-click the *File Output Validation Report* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

54. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



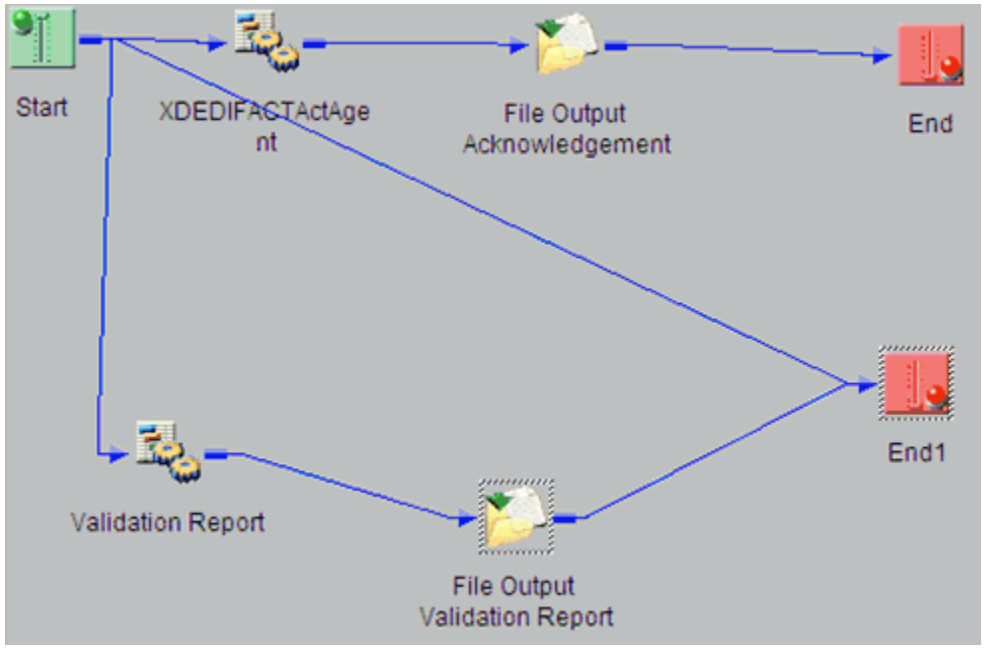
55. Select the *File Output Validation Report* object, right-click the *End1* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

56. From the Event drop-down list, select *OnCompletion* and click *OK*.

    Note: If you do not want to create APERAK or CONTROL, but need to monitor accept or reject rules validation status, a validation report can be substituted for the acknowledgement agent to update the SREG.

A line appears between the objects to indicate that a relationship has been established.



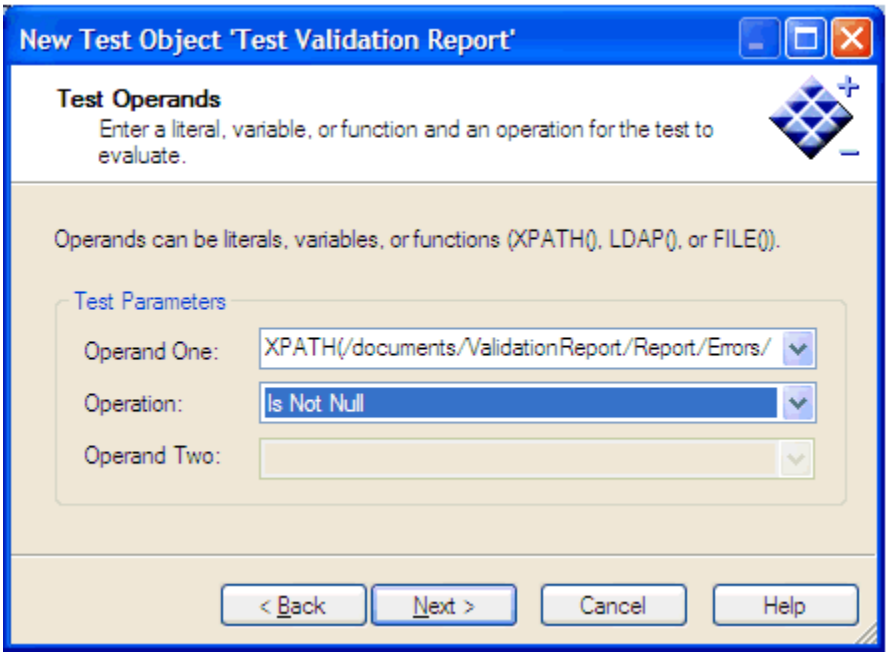
57. Drag and drop the Decision Test object from the toolbar to the workspace.

    The New Test Object dialog box opens.

58. In the Name field, type *Test Validation Report*, and a brief description (optional) in the Description field.

59. Click *Next*.

The Test Operands dialog box opens.

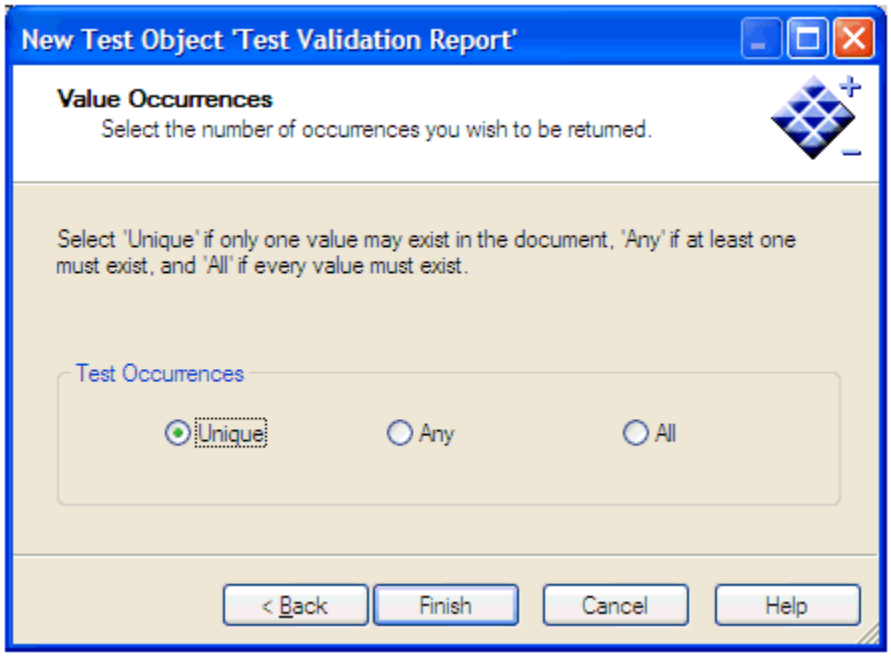


60. In the Operand One field, enter the following:

    ```
    XPATH(/documents/ValidationReport/Report/Errors/error)
    ```

61. From the Operation drop-down list, select *Is Not Null*.
62. Click *Next*.

The Value Occurrences dialog box opens.

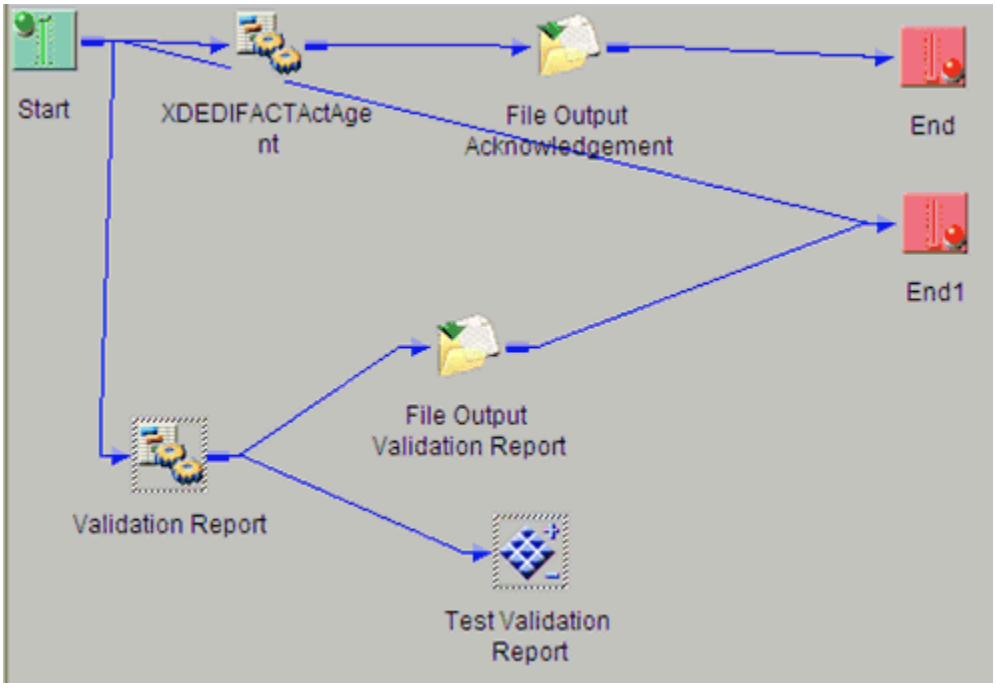


63. Ensure that *Unique* is selected from the available options.
64. Click *Finish*.

    The new Decision Test object appears in the workspace.
65. Select the *Validation Report* object, right-click the *Test Validation Report* object, and select *Relation* from the context menu.

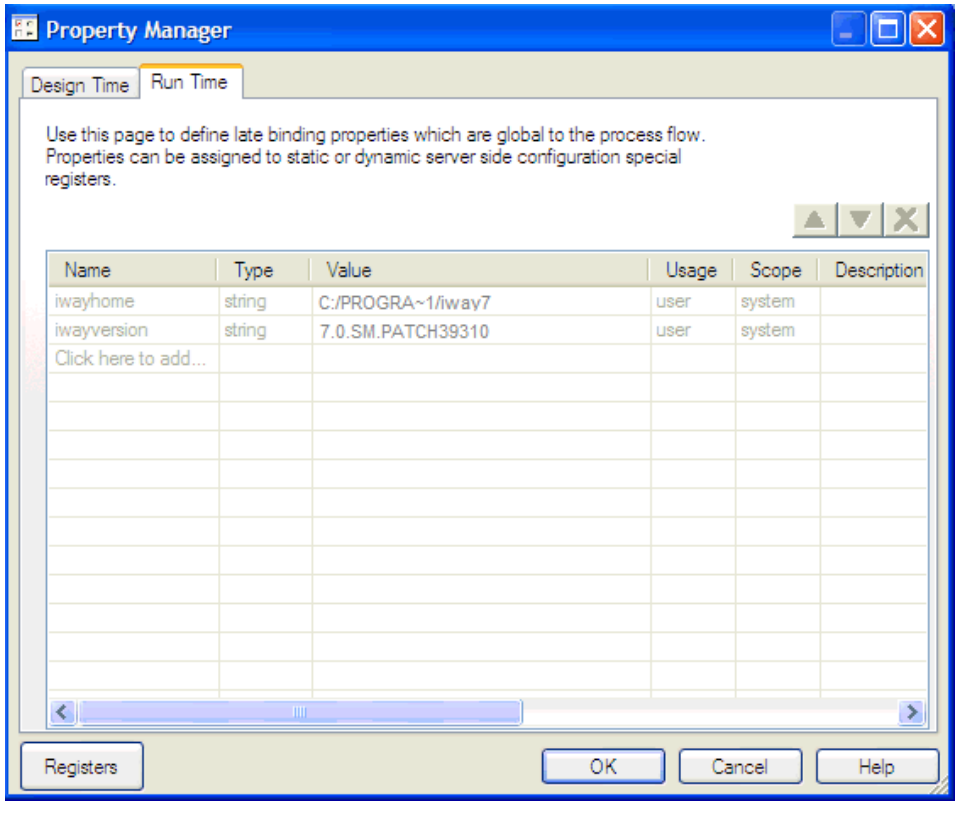
    The Line Configuration dialog box opens.
66. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



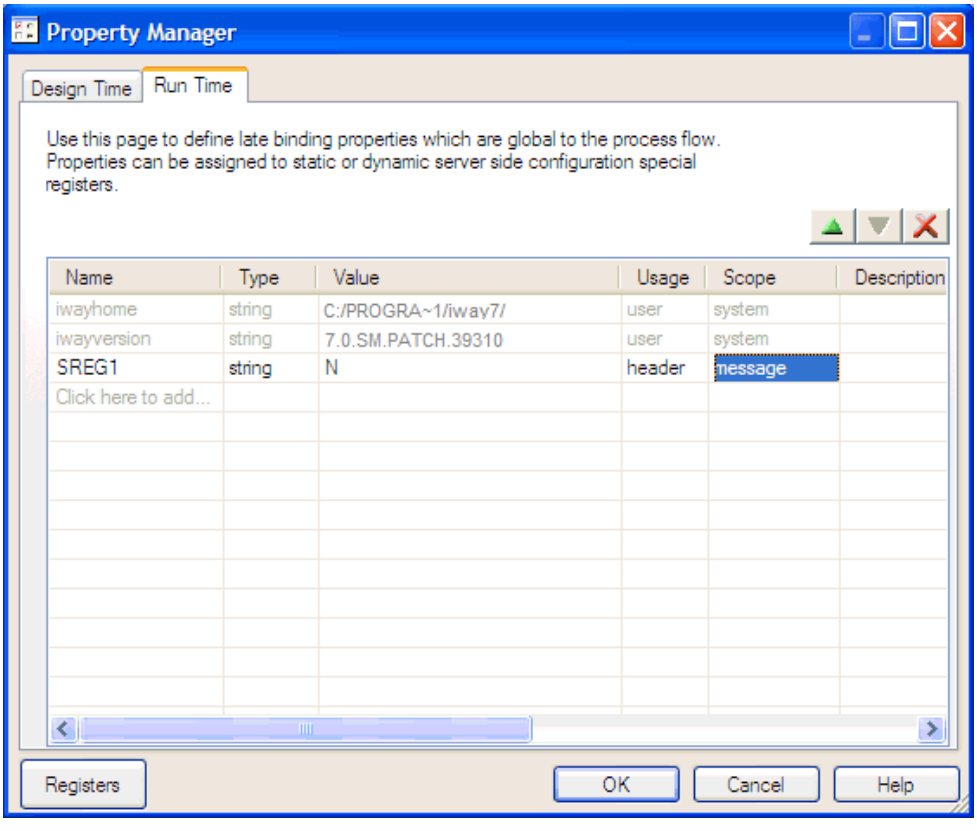
67. From the Tools menu, select *Property Manager*.

The Property Manager dialog box opens.



68. Click the *Run Time* tab.
69. In the Name column, click on an empty row.

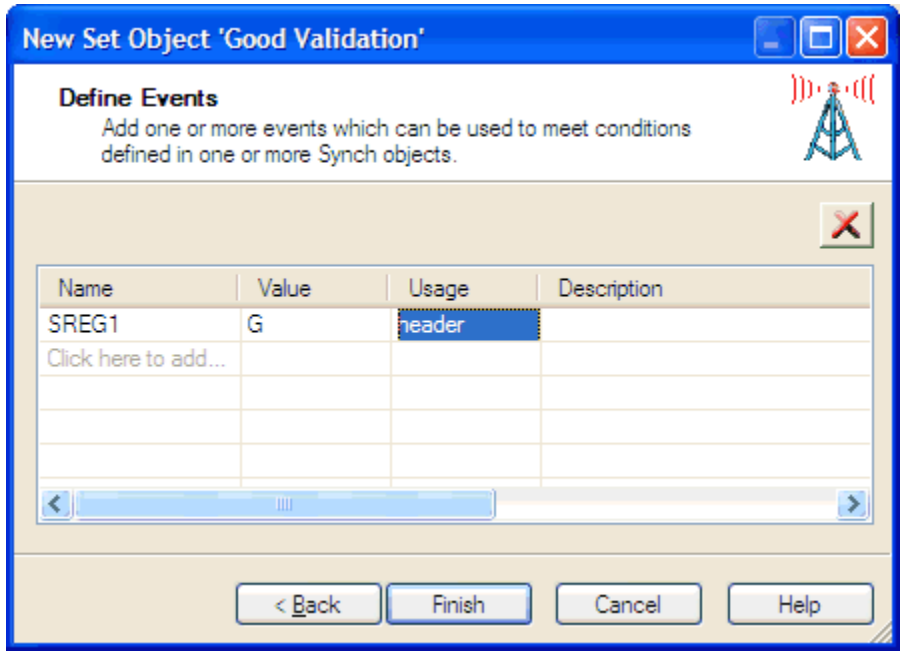
SREG1 appears in the Name column.



70. In the Value column, enter the value *N*.
71. In the Usage column, select *Header* from the drop-down list.
72. In the Scope column, select *Message* from the drop-down list.
73. Click *OK*.
74. Drag and drop the Set object from the toolbar to the workspace.

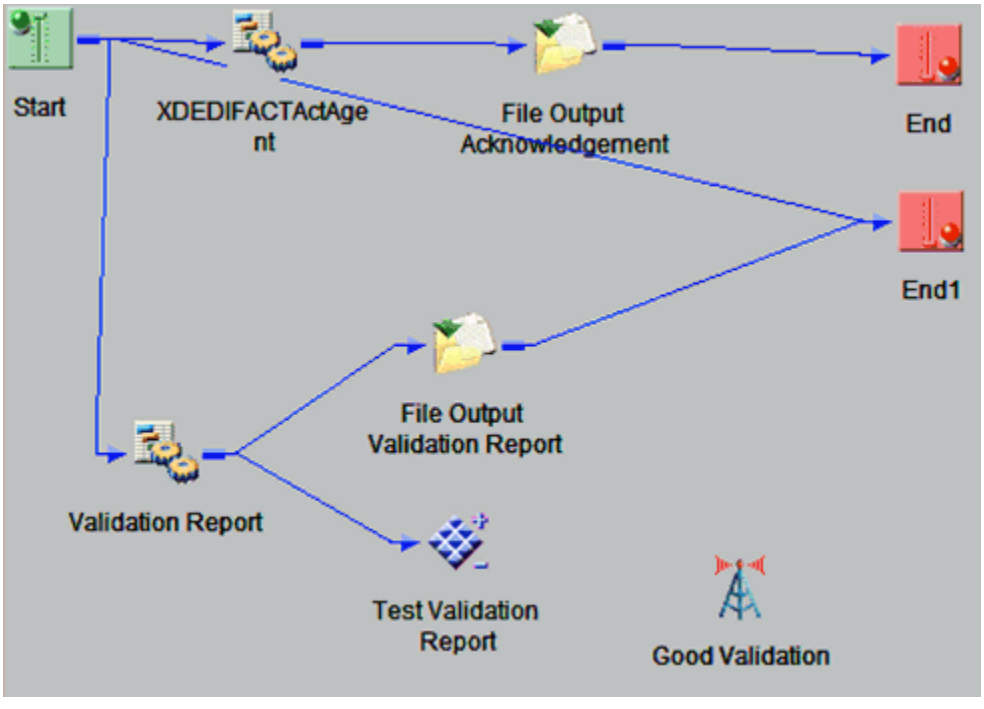
    The Set Name and Description dialog box opens.
75. In the Name field, type *Good Validation*, and a brief description (optional) in the Description field.
76. Click *Next*.

The Define Events dialog box opens.



77. Click the first row in the Name column and enter *SREG1*.
78. In the first row of the Value column, type *G*.
79. Select *header* from the drop-down list in the Usage column.
80. Click *Finish*.

The new Set object (Good Validation) appears in the workspace.



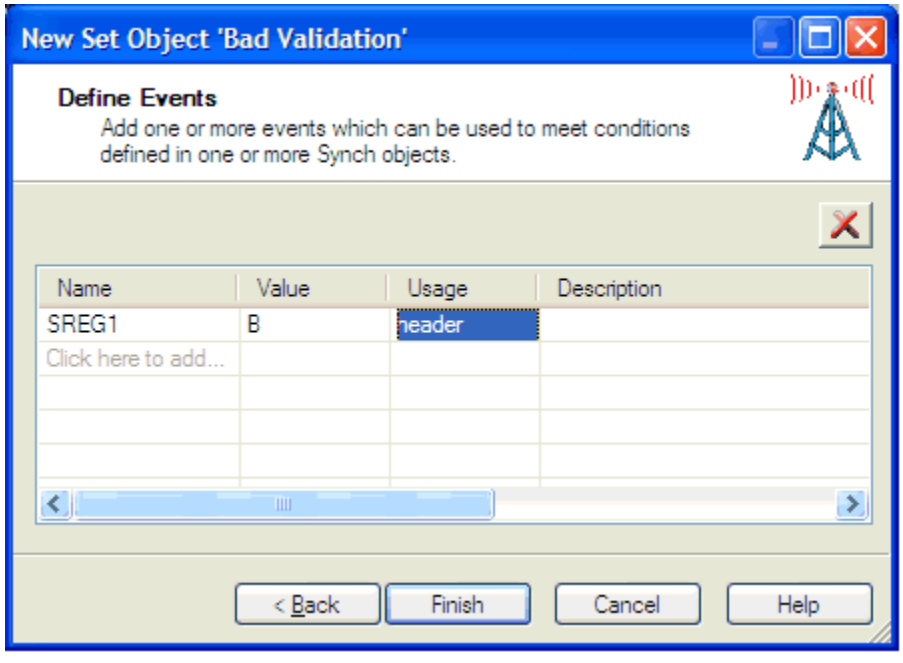
81. Drag and drop the Set object from the toolbar to the workspace.

    The Set Name and Description dialog box opens.

82. In the Name field, type *Bad Validation*, and a brief description (optional) in the Description field.

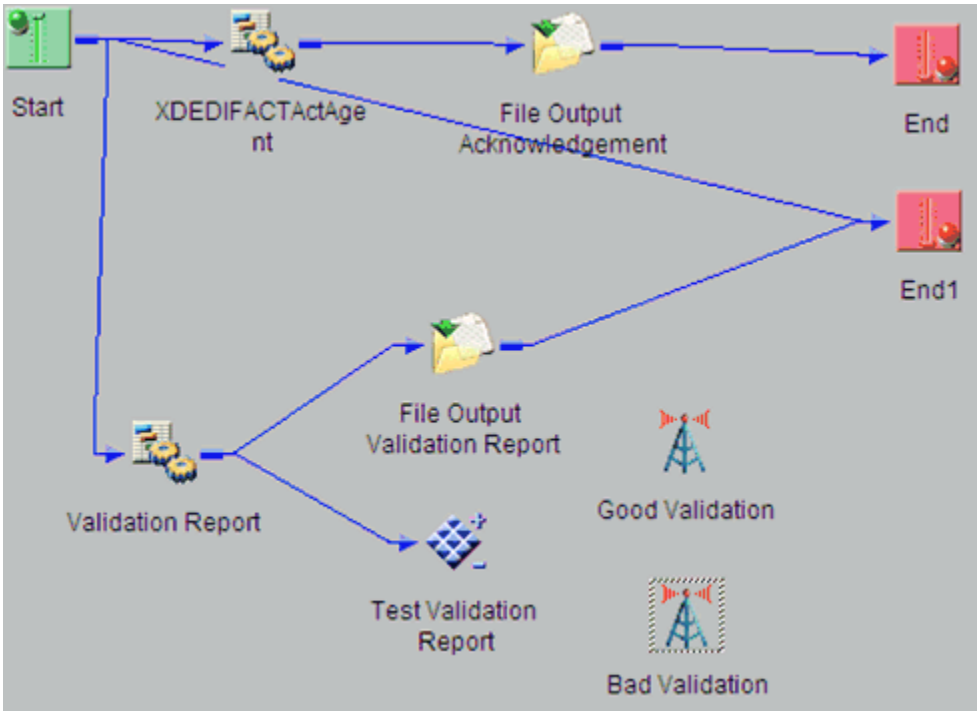
83. Click *Next*.

The Define Events dialog box opens.



84. Click the first row in the Name column and enter *SREG1*.
85. In the first row of the Value column, type *B*.
86. Select *header* from the drop-down list in the Usage column.
87. Click *Finish*.

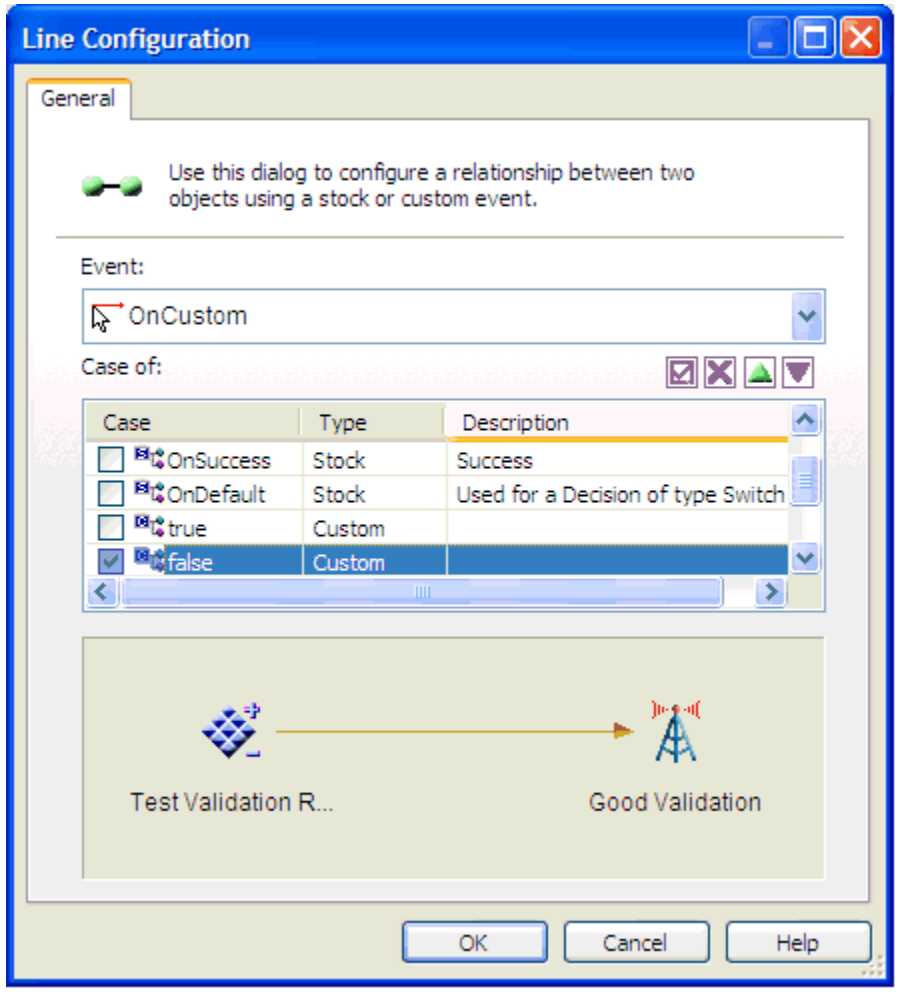
The new Set object (Bad Validation) appears in the workspace.



88. Select the *Test Validation Report* object, right-click the *Good Validation Set* object, and select *Relation* from the context menu.

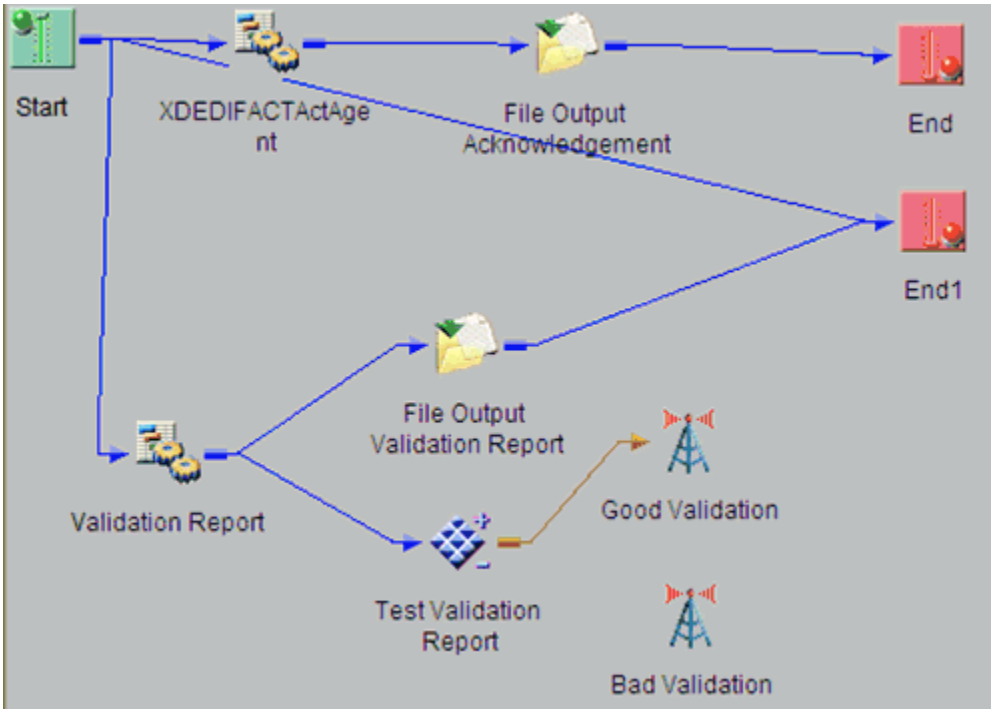
    The Line Configuration dialog box opens.

89. From the Event drop-down list, select *OnCustom*.



90. In the Case of section, select *false*.
91. Click *OK*.

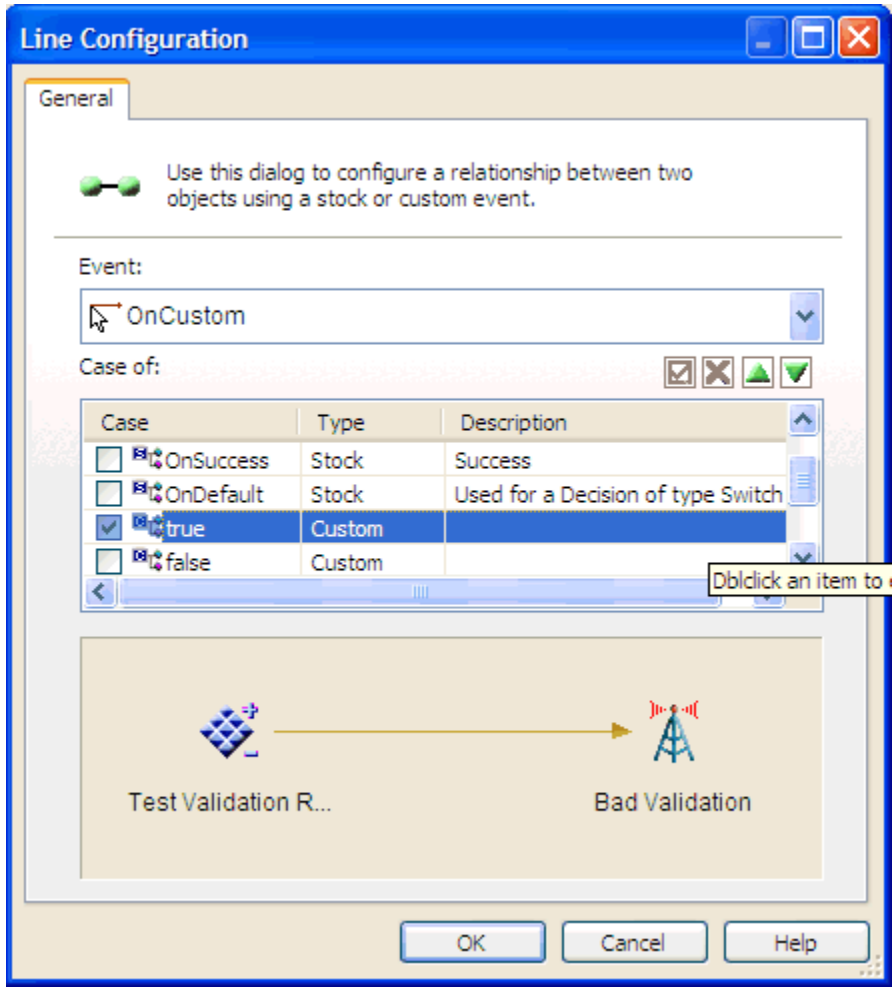
A line appears between the objects to indicate that a relationship has been established.



92. Select the *Test Validation Report* object, right-click the *Bad Validation* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

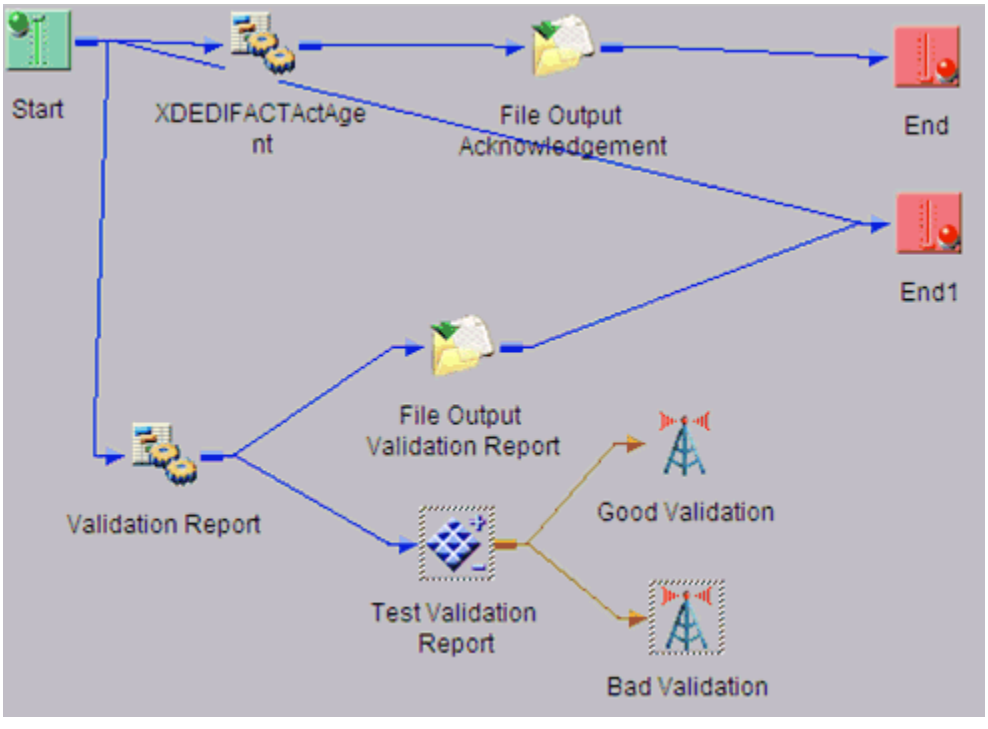
93. From the Event drop-down list, select *OnCustom*.



94. In the Case of section, select *true*.

95. Click *OK*.

A line appears between the objects to indicate that a relationship has been established.

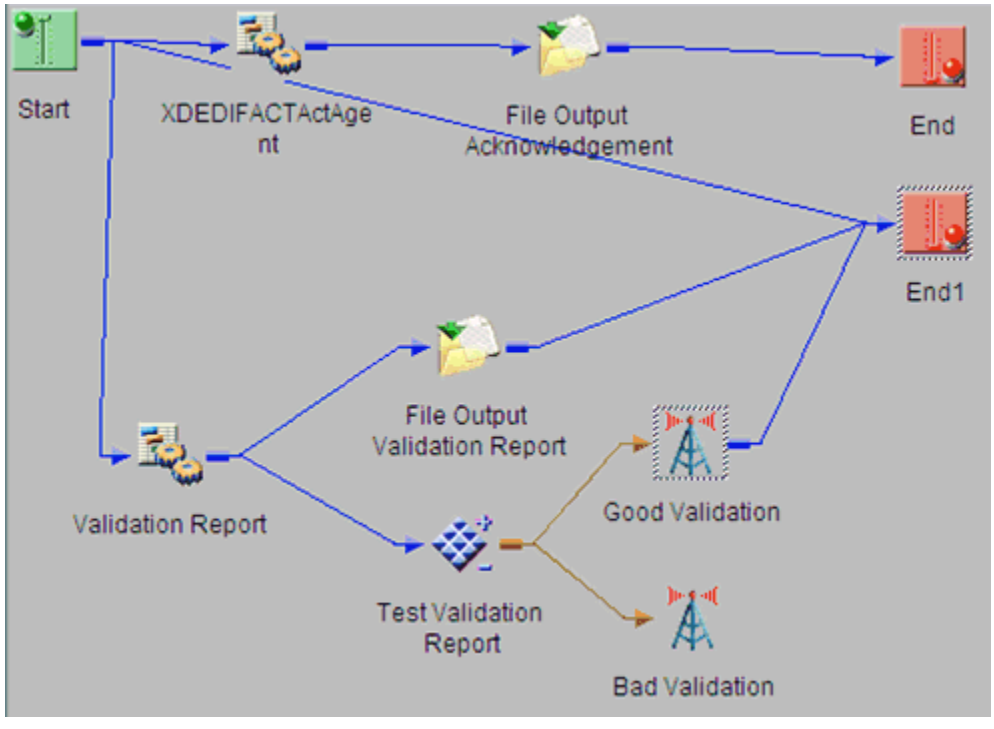


96. Select the *Good Validation* set object, right click the *End1* object, and select *Build Relation* from the context menu.

    The Line Configuration dialog box opens.

97. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.

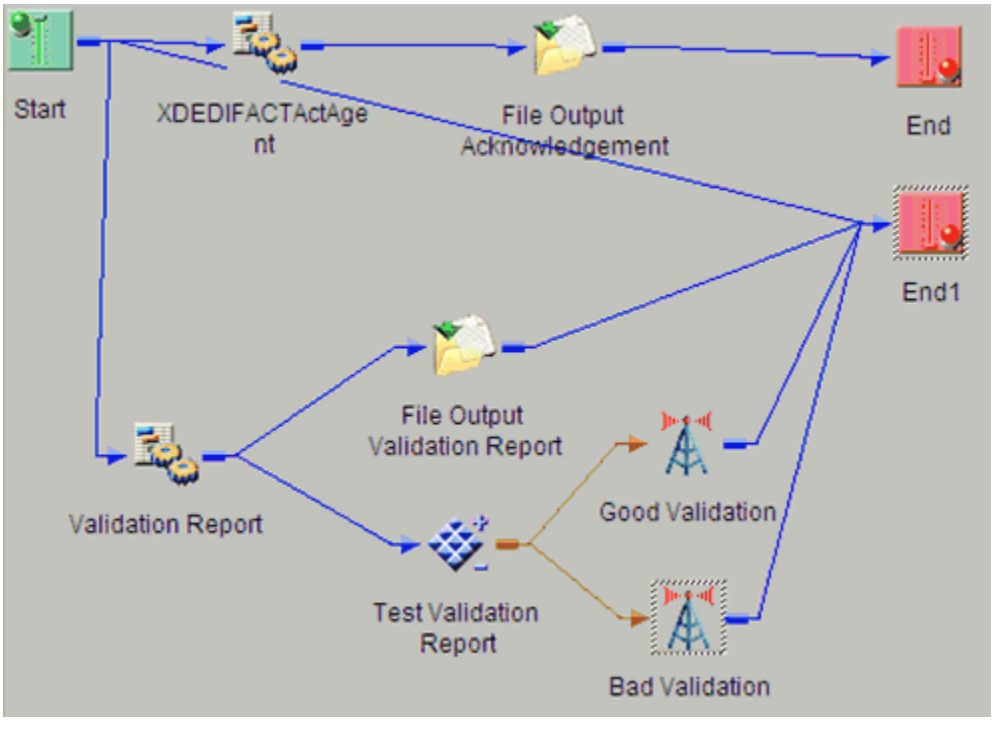


98. Select the *Bad Validation* set object, right click the *End1* object, and select *Build Relation* from the context menu.

    The Line Configuration dialog box opens.

99. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



The process flow is now complete.

100. To save the process flow, right-click the *EDIFACTtoXML* node in the left pane and select *Save* from the context menu.

Now you need to validate the process flow and publish it to the Registry of the iWay Service Manager Administration Console for use in the route of a channel for outbound processing.

Validating a process flow ensures that its structure is correct. Publishing a process flow makes it available in the Registry for use in a channel configuration. For instructions on validating and publishing the process flow, see the *iWay Integration Tools Designer User's Guide*.

101. Close iIT Designer.

Your next step is to add a new route to the Registry using the iWay Service Manager Administration Console and associate the process flow with it.

Procedure: How to Define a Route and Associate the Process Flow With It

To define a route and associate the process flow with It:

1. From the Registry menu options in the iWay Service Manager Administration Console, click *Routes*.
2. On the Route Definitions pane, click *Add* to add a route.
3. On the New Route Definition pane, enter a name for the route and an optional description, as shown in the following table.

Parameter	Value
Name *	EDIFACTToXML_Route
Description	This route will invoke the EDIFACT to XML process. The outcome of the process will place valid EDIFACT XML data in your *Good File* folder. Invalid EDIFACT XML data will be routed to your *Bad File* folder.

4. Click *Finish*.
5. On the Construct Route pane, click *Add*.

   You are prompted for the type of component to associate with the route.
6. Select *Process* and click *Next*.
7. The next pane prompts you to select a process. Select the process flow you created earlier with iIT Designer, *EDIFACTToXML*, and click *Finish*.

   The route, with its associated process flow, has been successfully defined.

Defining the Outlets

An outlet defines how a message leaves a channel. An emitter is a transport protocol that sends a document to its recipient. In the sample configuration, we will use a File emitter. For details on supported protocols, see the *iWay Service Manager Protocol Guide*.

For the channel in this example, you will add two emitters to the Registry. Then you will define two outlets, associating one emitter with each outlet.

When you associate the outlets with the channel in later steps, you will apply a condition to each one to dynamically direct the flow of the output document based on its content.

In the example, you will add an emitter for the transformed EDIFACT input data. Two emitters will be defined. The purpose is to separate EDIFACT data that passes validation and those that failed. You will set a condition statement to ensure that the data is emitted to their corresponding folders. The condition settings will be as follows:

Good Output Emitter:

```
Cond(_sreg(Sreg1),EQ, 'G') and _isXML() and _Cond(_root(),ne,'documents')
```

This condition is checking the special register SREG1 for a value of 'G' (this was set in the process flow). It is also checking to see if the data is formatted in XML. The last statement is allowing only XML documents where the root tag is not equal to documents. An XML with a root tag of documents is the validation report. In this Emitter we are interested in the transformed XML document only.

Bad Output Emitter:

```
Cond(_sreg(Sreg1),EQ, 'B') and _isXML() and _Cond(_root(),ne,'documents')
```

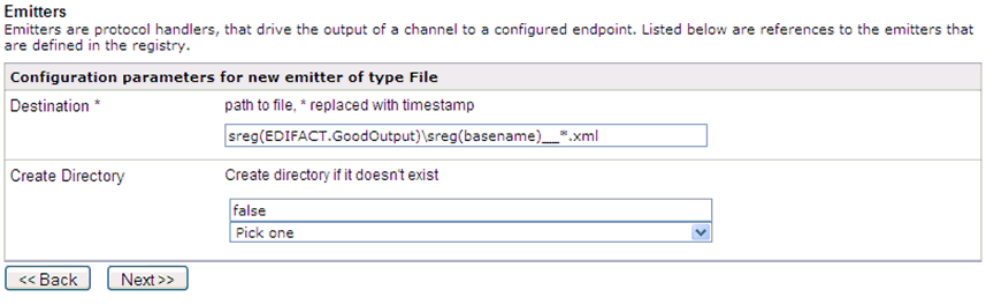
This condition is checking the special register SREG1 for a value of 'B' (this was set in the process flow). It is also checking to see if the data is formatted in XML. The last statement is allowing only XML documents where the root tag is not equal to documents. An XML with a root tag of documents is the validation report. In this Emitter we are interested in the transformed XML document only.

Procedure: How to Add an Emitter for Good EDIFACT Output

This procedure describes how to configure an emitter for transformed EDIFACT input data that passes validation.

1. On the Emitters pane, click *Add* to add an emitter.
   The next pane prompts you for the emitter type.
2. For this example, select *File* from the drop-down list and click *Next*.

   The configuration parameters pane opens.



3. Supply configuration parameters for the File emitter as follows, then click *Next*.

Parameter	Value
Destination *	sreg(EDIFACT.GoodOutput)\sreg(basename)__*.xml This value is the directory where the transformed XML will be placed for any EDIFACT input data that has passed validation. sreg(EDIFACT.GoodOutput) is a special register value that uses a defined directory in which output files are stored after transformation. For more information, see *Adding Special Register Sets* on page 64. Make sure that you have created this directory; otherwise, errors will occur during deployment. On output, an asterisk (*) in the destination file name is replaced by a date and time stamp. For details on the special register (SREG) used in the preceding file name, see the *iWay Service Manager User's Guide*.
Create Directory	false

4. On the Emitters pane, enter the name of the new emitter and an optional description, as shown in the following table. Then click *Finish* to add the emitter.

Parameter	Value
Name *	Good_Output_Emitter
Description	Emitter for EDIFACT XML output that passes Report Validation.

Procedure: How to Define an Outlet for Good EDIFACT Output

1. From the Registry menu options, select *Outlets*.
2. On the Outlet Definitions pane, click *Add* to add an outlet.
3. On the New Outlet Definition pane, enter the name of the new outlet and an optional description, as shown in the following table. Then click *Finish* to add the outlet.

Parameter	Value
Name *	Good_Output_Outlet
Description	Good File Outlet for EDIFACT

4. On the Construct Outlet pane, click *Add* to associate the created emitter with its outlet. The next pane prompts you for the component type.
5. Select *Emitter* and click *Next*.

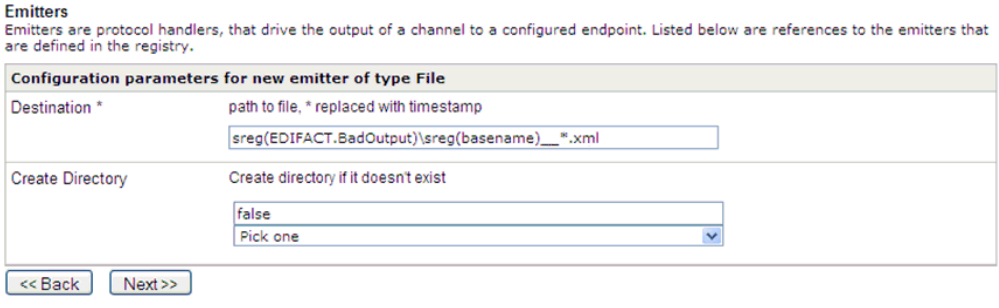
   The next pane prompts you to select an emitter.
6. Select *Good_Output_Emitter*, which is the emitter you added earlier, and click *Finish*.

Procedure: How to Add an Emitter for Bad EDIFACT Output

This procedure describes how to configure an emitter for transformed EDIFACT input data that fails validation.

1. On the Emitters pane, click *Add* to add an emitter.
   The next pane prompts you for the emitter type.
2. For this example, select *File* from the drop-down list and click *Next*.

   The configuration parameters pane opens.



3. Supply configuration parameters for the File emitter as follows, then click *Next*.

Parameter	Value
Destination *	sreg(EDIFACT.BadOutput)\sreg(basename)__*.xml This value is the directory where the transformed XML will be placed for any EDIFACT input data that has failed validation. sreg(EDIFACT.BadOutput) is a special register value that uses a defined directory in which output files are stored after transformation. For more information, see *Adding Special Register Sets* on page 64. Make sure that you have created this directory; otherwise, errors will occur during deployment. On output, an asterisk (*) in the destination file name is replaced by a date and time stamp. For details on the special register (SREG) used in the preceding file name, see the *iWay Service Manager User's Guide*.
Create Directory	false

4. On the Emitters pane, enter the name of the new emitter and an optional description, as shown in the following table. Then click *Finish* to add the emitter.

Parameter	Value
Name *	Bad_Output_Emitter
Description	Emitter for EDIFACT XML output that failed Report Validation.

Procedure: How to Define an Outlet for Bad EDIFACT Output

1. From the Registry menu options, select *Outlets*.
2. On the Outlet Definitions pane, click *Add* to add an outlet.
3. On the New Outlet Definition pane, enter the name of the new outlet and an optional description, as shown in the following table. Then click *Finish* to add the outlet.

Parameter	Value
Name *	Bad_Output_Outlet
Description	Bad File Outlet for EDIFACT

4. On the Construct Outlet pane, click *Add* to associate the created emitter with its outlet. The next pane prompts you for the component type.
5. Select *Emitter* and click *Next*.

   The next pane prompts you to select an emitter.
6. Select *Bad_Output_Emitter*, which is the emitter you added earlier, and click *Finish*.

Defining a Channel

This section describes how to define a channel.

Procedure: How to Define a Channel

1. From the Registry menu options, select *Channels* under Conduits.
2. On the Channel Definitions pane, click *Add* to add a channel.
3. On the New Channel Definition pane, enter the name of the new channel and an optional description, as shown in the following table. Then click *Finish* to add the channel.

Parameter	Value
Name *	EDIFACTtoXML_Channel
Description	Channel for EDIFACT to XML inbound processing.

4. On the Construct Channel pane, click *Add* to associate the inlet, route, and outlets defined previously with the channel.

   You are prompted to associate components with the channel.
5. Select *Inlet* and click *Next*.
   The next pane prompts you to select an inlet.
6. Select *EDIFACTtoXML*, which is the inlet you defined earlier, and click *Finish*.

   The inlet is added to the channel. Now you need to associate the route defined earlier with the channel.
7. On the Construct Channel pane, click *Add*.

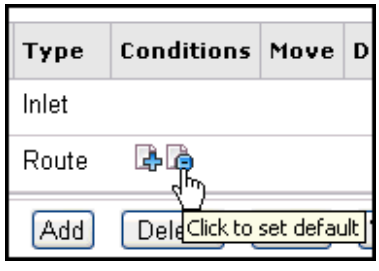
   The next pane prompts you for the component type.

8. Select *Route* and click *Next*.

   On the next pane, you are prompted to select a route.

9. Select *EDIFACTtoXML_Route*, which is the route created earlier, and click *Finish*.

10. On the Construct Channel pane, click the *minus sign (-)*under Conditions next to the name of the route to set it as the default.



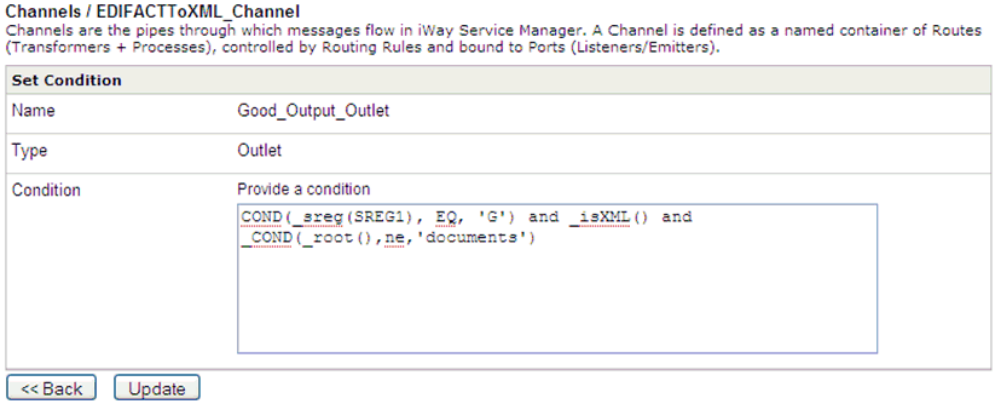
11. On the Construct Channel pane, click *Add* to add the outlets.

12. On the next pane, select *Outlet* and click *Next*.

13. Select the outlet you defined earlier, *Good_Output_Outlet*, and click *Finish*.

14. To set a condition for the Good_Output_Outlet, on the Construct Channel pane, click the *plus sign (+)* under Conditions for the Good_Output_Outlet.

    The Set Condition pane opens.



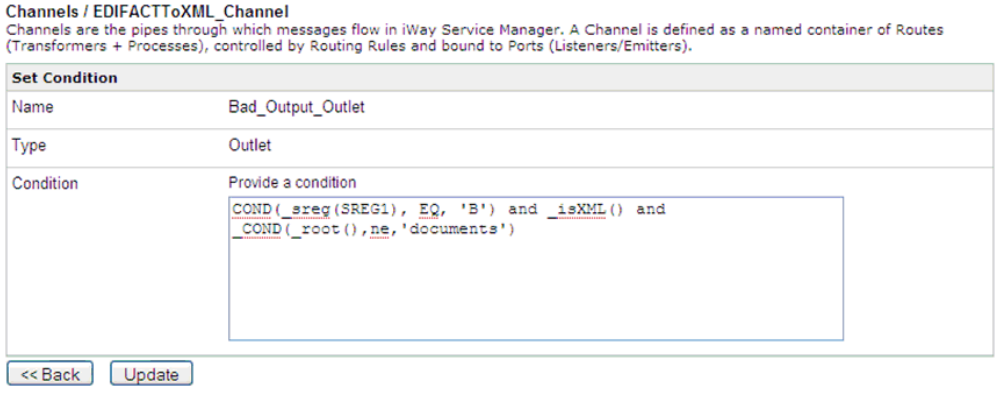
15. In the Condition input field, enter the following:

```
COND(_sreg(SREG1), EQ, 'G') and _isXML() and
_COND(_root(),ne,'documents')
```

16. Click *Update*.

17. On the Construct Channel pane, click *Add* to add the outlets.
18. On the next pane, select *Outlet* and click *Next*.
19. Select the outlet you defined earlier, *Bad_Output_Outlet*, and click *Finish*.
20. To set a condition for the Bad_Output_Outlet, on the Construct Channel pane, click the plus sign(+) under conditions for the Bad_Output_Outlet.

    The Set Condition pane opens.



21. In the Condition input field, enter the following:

```
COND(_sreg(SREG1), EQ, 'B') and _isXML() and
_COND(_root(),ne,'documents')
```

22. Click *Update*.

Procedure: How to Add the Ebix to the Channel

1. From the Registry menu options, select *Channels*.

   The Channel Definitions pane opens.

2. Click the link in the Ebix column for the EDIFACTtoXML_Channel.
3. On the next pane, which prompts you to add Ebix components, click *Add* to add the Ebix to the channel.
4. On the next pane, select *EDIFACT_D01B_Ebix*, which is the name of the Ebix you added previously, and click *Finish*.

   Now that you have associated all the components with the channel, you are ready to build it.

Procedure: How to Add the Register to the Channel

1. From the registry menu options, select *Channels*.

   The Channel Definitions pane opens.

2. Click the link in the *Regs* column for the EDIFACTtoXML_Channel.
3. On the next pane, which prompts you to add register sets, click *Add*.
4. On the next pane, select *EDIFACT*, which is the register set you created previously.

   For more information on creating register sets, see *Adding Special Register Sets* on page 64.

   Now that you have associated all the components with the channel you are ready to build it.

Procedure: How to Build the Channel

1. From the Registry menu options on the left pane, select *Channels* under Conduits.
2. On the Channel Definitions pane, select the channel defined previously, *EDIFACTtoXML_Channel*, and click *Build*.

   The results of the build are displayed on the right pane.

3. Review the results of your build and then click *Back*.

   If an error or errors are displayed in the Message column, take the appropriate action as instructed.

Procedure: How to Deploy the Channel

Deployment is the mechanism by which a channel moves from being stored in the Registry to becoming active in iWay Service Manager. For more information on deployment, see the *iWay Service Manager User's Guide*.

1. Select the *Deployments* option in the top pane.
2. On the Channel Management pane, click *Deploy*.
3. On the Available Channels pane, select the channel you defined previously, *EDIFACTtoXML_Channel*, and click *Deploy*.

   The Channel Management pane reopens.

4. Select *EDIFACTtoXML_Channel* and click *Start*.

   The red X under Status changes to a green check mark to indicate that the channel has been started. If an error or errors are displayed, take the appropriate action as instructed.

Reusing Your Channel Configuration

Using the Archive Manager feature of iWay Service Manager, you can archive your channel configuration with its associated components and import them into another Registry. They will then be available from that Registry for modification or reuse.

For details on this feature, see the *iWay Service Manager User's Guide*.

Chapter 6

Outbound Processing: XML to UN/EDIFACT

The iWay Integration Solution for UN/EDIFACT includes iWay Service Manager. iWay Service Manager validates an XML document based on EDIFACT published implementation guides and converts it to a document in UN/EDIFACT format.

This chapter provides the information you need to understand and implement a basic outbound message flow.

- The outbound processing overview describes the iWay business components and the processing steps in the basic outbound message flow.
- The sample configuration contains detailed instructions for configuring the basic outbound message flow. This topic guides you through each step of the configuration procedure.

In this chapter:

EDIFACT Outbound Processing Overview

The standard outbound process converts an XML document to an EDIFACT-formatted document.

The input document that is sent to the channel may not be in XML format. It can be any input document that first will be processed by the channel and transformed to an EDIFACT document.

In a basic message flow, outbound processing consists of the following components and steps. For an illustration of the components available in the construction of a message flow, see *Using a Channel to Construct a Message Flow* on page 19. You will define the components in the configuration instructions in *Sample Configuration for Outbound Processing: XML to EDIFACT* on page 114.

Inlet

- The listener picks up the input document.

Route / Process Flow

- A process flow guides the XML-formatted EDIFACT document through the next stages of the process.

  Rules processing runs against the XML-formatted EDIFACT document to validate its structure and content. The published EDIFACT standards and user implementation guides define element types (for example, numeric, alpha, or date) and describe business rules to apply for validation.

  The *XMLToEDIFACTTransformationAgent* obtains the message type and version from the XML-formatted EDIFACT document. The appropriate transformation template is applied from the Ebix. The transformation converts the XML-formatted EDIFACT document to EDIFACT format.

  The *XDEDIFACTValidationReportAgent* creates a report (an XML document) containing the XML-formatted EDIFACT document and resulting EDIFACT formatted data, as well as the validation status.

  If the EDIFACT document did not contain any errors during the rules processing stage, it is emitted and continues to its next destination. The validation report is always emitted. In the sample process flow that is described later in this chapter, *good* validation reports are written with a file name prefix of *validation*. All other validation reports are written with a file name prefix of *error*. Information in the *error* validation reports can be routed accordingly for repair and reprocessing.

Outlet

- The EDIFACT document is passed to the next step in the integration process.

Sample Configuration for Outbound Processing: XML to EDIFACT

This topic provides step-by-step instructions for configuring a basic outbound message flow for the iWay Integration Solution for UN/EDIFACT. The message flow represents the movement and tasks in the conversion of a message from XML to EDIFACT.

If you plan to modify the message flow presented here and would like more information on the supported iWay business components that you can use in channel construction, see the *iWay Service Manager User's Guide*.

Accessing the iWay Service Manager Administration Console

For instructions, see *Accessing the iWay Service Manager Administration Console*.

Adding an Ebix to the Registry

The iWay e-Business Information Exchange (Ebix) framework supplies several Ebix files for the iWay Integration Solution for UN/EDIFACT.

An Ebix file for EDIFACT is named EDIFACT_*transaction_set*.ebx, where *transaction_set* is the transaction set number. For example, the Ebix file for EDIFACT transaction set D01B is named EDIFACT_D01B.ebx.

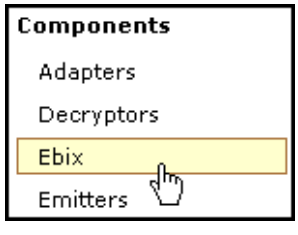
For details on the supported EDIFACT transaction sets, see *UN/EDIFACT Supported Versions* on page 143.

This topic describes how to add an Ebix to the Registry on Windows and UNIX.

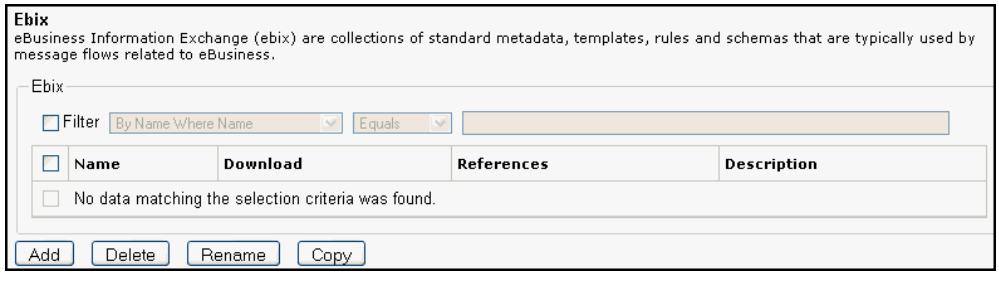
Tip: If you already added an Ebix to the Registry as described in *Adding an Ebix to the Registry* on page 62, you do not need to add it again for outbound processing. You can go directly to *Defining an Inlet* on page 118.

Procedure: How to Add an Ebix to the Registry on Windows

1. To access the Registry, click the *Registry* option in the top pane.
2. Under Components in the left pane of the Registry, select *Ebix*.

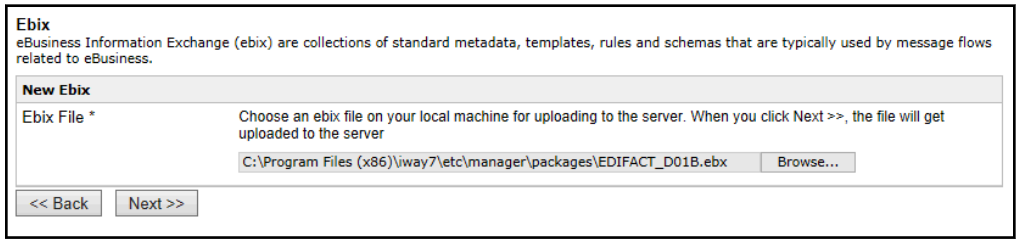


The Ebix pane opens, as shown in the following image.



3. Click *Add* to add a new Ebix.

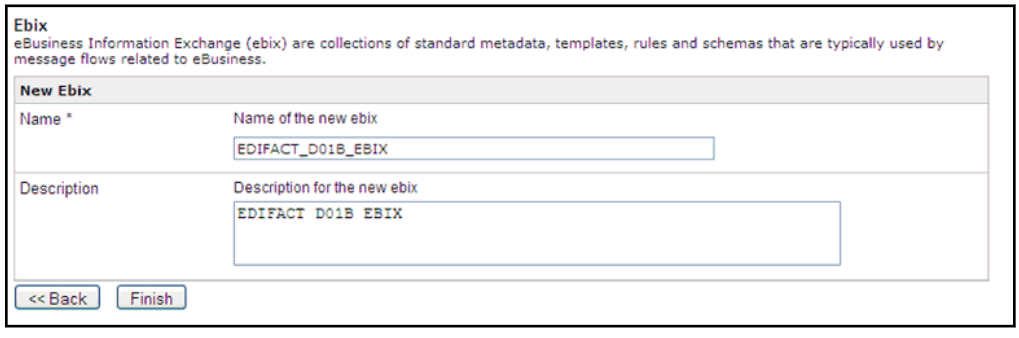
The New Ebix pane opens.



4. Browse to the directory in which the Ebix is located and select the name of the file, for example, *EDIFACT_D01B.ebx*.

5. Once you have selected the Ebix, click *Next*.

   You are prompted for the name of the Ebix and an optional description.



6. Enter a name for the Ebix, for example, *EDIFACT_D01B_EBIX*, and an optional description, such as *EDIFACT D01B EBIX*.

   Note: This step must be repeated for each Ebix message set that is added to the Registry.

7. Click *Finish*.

   On the Ebix pane, you will see that the Ebix was successfully added. Later you will associate it with the channel for inbound processing.

Procedure: How to Add an Ebix to the Registry on UNIX

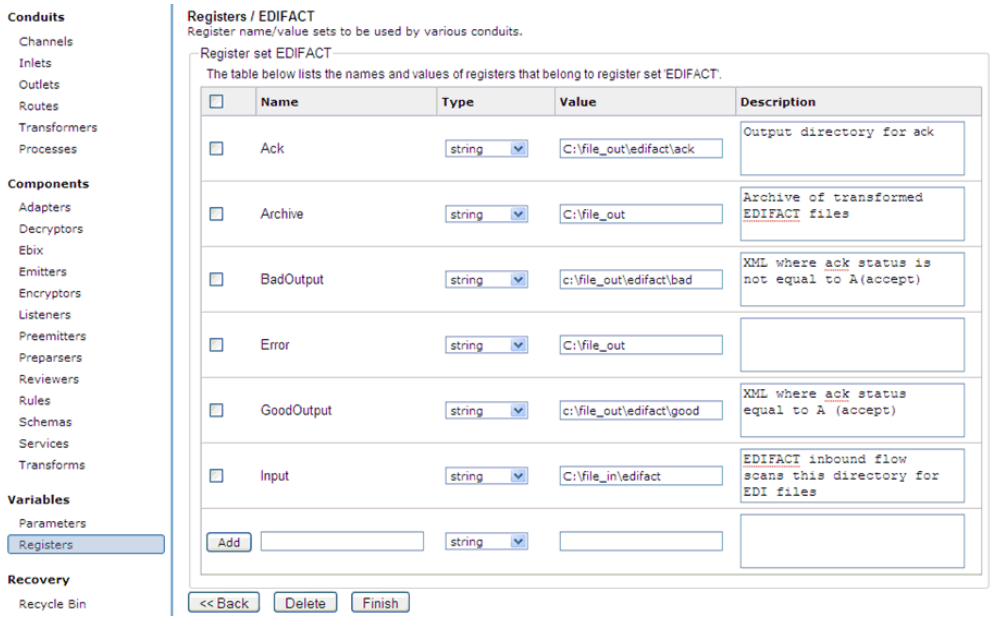
Depending on your system configuration, there are two methods that you can use to add an Ebix to the Registry on UNIX.

- ❑ If you have a web browser on the UNIX machine, follow the instructions for Windows.
- ❑ Use FTP to download the Ebix from the *iWay7/etc/manager/packages* directory to your Windows machine and follow the instructions for Windows.

Adding Special Register Sets

In iWay Service Manager, a special register is a name-value pair that defines a variable that is carried throughout the system. Once defined, this variable is available to all components of the system. Within the EDIFACT components, a Best Practice is to use special registers to define inputs and outputs. When packages containing channels are migrated between systems, the only changes required to deploy in the new location is to modify these special registers and build the channel. Channels may have many locations and this practice will minimize the effort required to migrate. For a complete list of system special registers that are provided, see the *iWay Service Manager Programmer's Guide*. For more information on defining a special register of your own, see the *iWay Service Manager User's Guide*.

The sample outbound channel uses a set of special registers. For example:



Procedure: How to Add a Special Register Set to Your Channel

To add a special register set to your channel:

1. In the left console pane of the Registry menu, select *Channels*.

   The Channels pane opens.

2. In the row for your channel, click *Regs* for the channel you want to modify.

   The Assign register pane opens.

3. Select a register and click *Finish*.
4. Click *Back* to return to the Channels pane.

Defining an Inlet

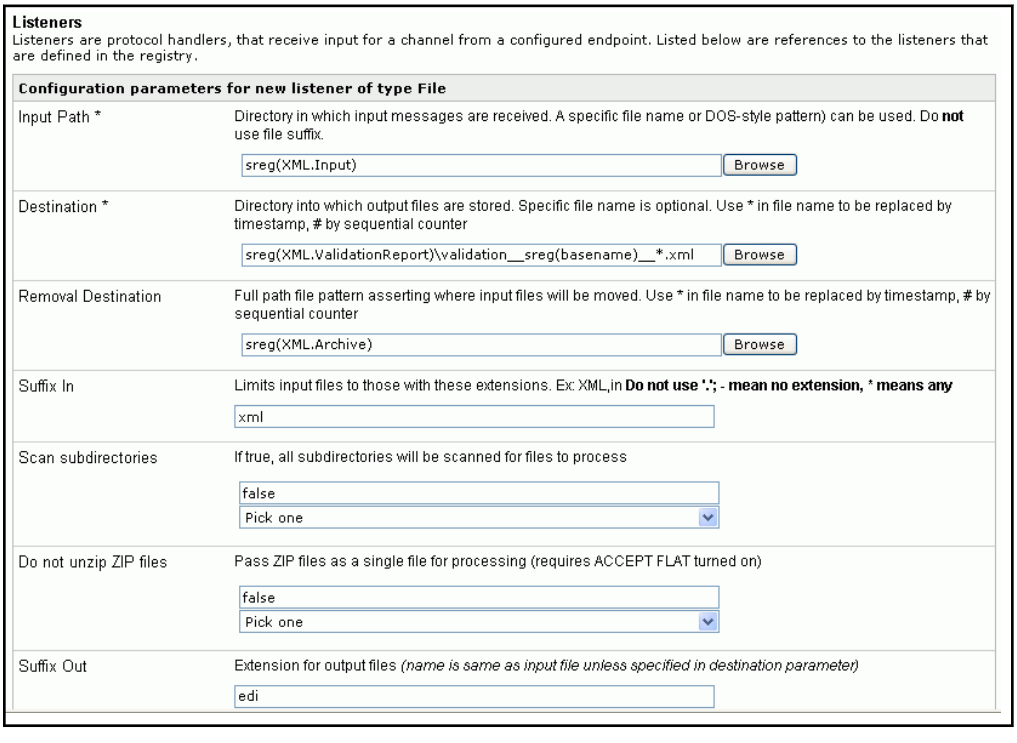
You will add a listener to the Registry, then associate that listener with a new inlet.

Procedure: How to Add a Listener

1. From the Registry menu options, select *Listeners*.
2. On the Listeners pane, click *Add* to add a new listener.
3. For the purpose of this example, we will show the configuration with a File listener. For details on supported protocols, see the *iWay Service Manager Protocol Guide*.

   Select *File* from the Type drop-down list and click *Next*.

   The configuration parameters pane opens.



4. Supply configuration parameters for the new File listener as follows. An asterisk indicates that a parameter is required. For parameters not listed in the following table, accept the default value.

Parameter	Value
Input Path *	sreg(XML.Input) This value is a special register that uses a defined directory in which input messages are received. Make sure that you have created this directory; otherwise, errors will occur during deployment.
Destination *	sreg(XML.ValidationReport)\validation__sreg(basename)__*.xml This value is a special register that uses a defined directory in which output messages are received. **Note:** The double underscore characters are used in the destination to escape the underscore. Make sure that you have created this directory; otherwise, errors will occur during deployment.
Removal Destination	sreg(XML.Archive) This value is a special register that uses a defined directory to which output messages are moved if they fail during transformation. Make sure that you have created this directory; otherwise, errors will occur during deployment. It is recommended to configure a removal destination when you are constructing a basic channel.
Suffix In	xml Input files with the extension .xml are allowed.
Suffix Out	edifact In this example, the extension for output files is .edifact.

5. Click *Next*.
6. On the Listeners pane, enter the name of the new listener and a brief description, as shown in the following table.

Parameter	Value
Name *	XmlToEDIFACT_Ebix
Description	XML to EDIFACT file listener

7. Click *Finish* to add the listener.

Procedure: How to Define an Inlet

1. From the Registry menu options, select *Inlets*.
2. On the Inlet Definitions pane, click *Add* to add an inlet.
3. On the New Inlet Definition pane, enter the name of the new inlet and an optional description, as shown in the following table.

Parameter	Value
Name *	XmlToEDIFACT_Ebix
Description	The file inlet contains a file listener for XML to EDIFACT processing.

4. Click *Finish* to add the inlet.
5. On the Construct Inlet pane, click *Add* to associate the listener with the inlet. The next pane prompts you for the component type.
6. Select *Listener* and click *Next*.

   The next pane prompts you to select a listener.
7. Select *XmlToEDIFACT_Ebix*, which is the listener you added earlier for outbound processing, and click *Finish*.

   The listener is added to the inlet.

Defining a Route

For this sample channel configuration, you will define a route that will invoke the XML to EDIFACT validation process flow. The outcome of the validation process flow will place valid EDIFACT data in a defined output folder. Invalid EDIFACT data will be routed to an errors folder. A validation report will also be sent to the appropriate folder.

This section describes how to create a validation process flow using iIT Designer and bind it to a sample outbound channel as a route.

Procedure: How to Create a New Project and Start the Process Flow

To create a new project and start the process flow using iIT Designer:

1. From the Windows Start menu select *Programs*, *iWay 7.0 Service Manager*, *tools*, and then *iWay Integration Tools (iIT) Designer*.
2. Connect to the repository from which you want to work, for example, iWay.
3. Right-click the *iWay* node and select *New Project* from the drop-down list.

   The Designer Project Information dialog box opens, prompting you for a project name and optional description.
4. In the Name field, type a project name, for example, *Test*.

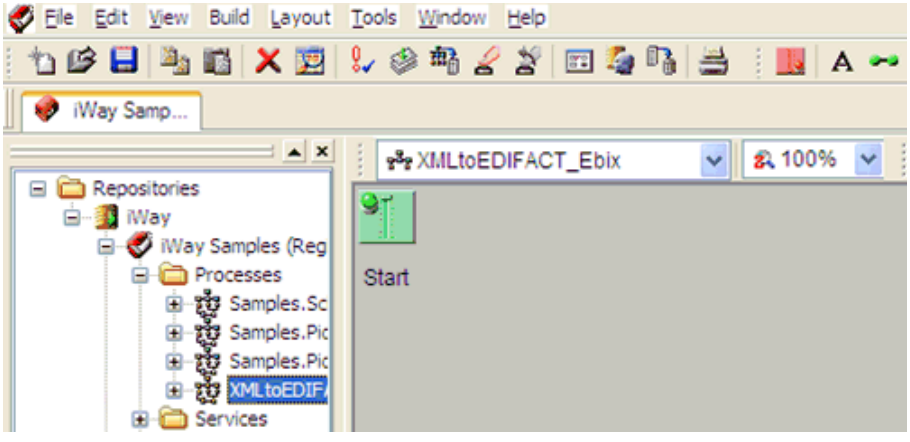
   In the Description field, type a brief description (optional) to describe the project.
5. Click *Next*.

   The Designer Project Bindings dialog box opens.
6. To create the project in the iWay Registry, select *iWay Registry* and click *Finish*.
   The choice of project association depends on where you intend to publish (deploy) your process flow. If you are developing a process flow for use as part of a channel, you must publish it to the iWay Registry for subsequent deployment.
7. The Test project node appears under the repository in which it was created (in this example, it appears under iWay).
8. To save the project to the repository, right-click the project node and select *Save* from the context menu.
9. Expand the Test project node to expose the project elements (Processes, Services, Transforms, and so on).
10. Right-click the *Processes* folder and select *New Process* from the drop-down list.

    The iWay Process Configuration dialog box opens.
11. In the Name field, type *XMLToEDIFACT_Ebix* as the process flow name.

    In the Description field, type a brief description (optional).
12. Click *Finish*.

The new XMLToEDIFACT_Ebix node appears under the Processes folder, and the workspace displays a Start object.



You are ready to build the XMLToEDIFACT_Ebix validation process flow by configuring objects to it and specifying their relationships.

Procedure: How to Configure Objects for the Process Flow

To configure objects for the process flow using iIT Designer:

1. Drag and drop the Service object from the toolbar to the workspace.

   The New Service Object dialog box opens.

2. In the Name field, type *XMLToEDIFACTTransformAgent*, and a brief description (optional) in the Description field.

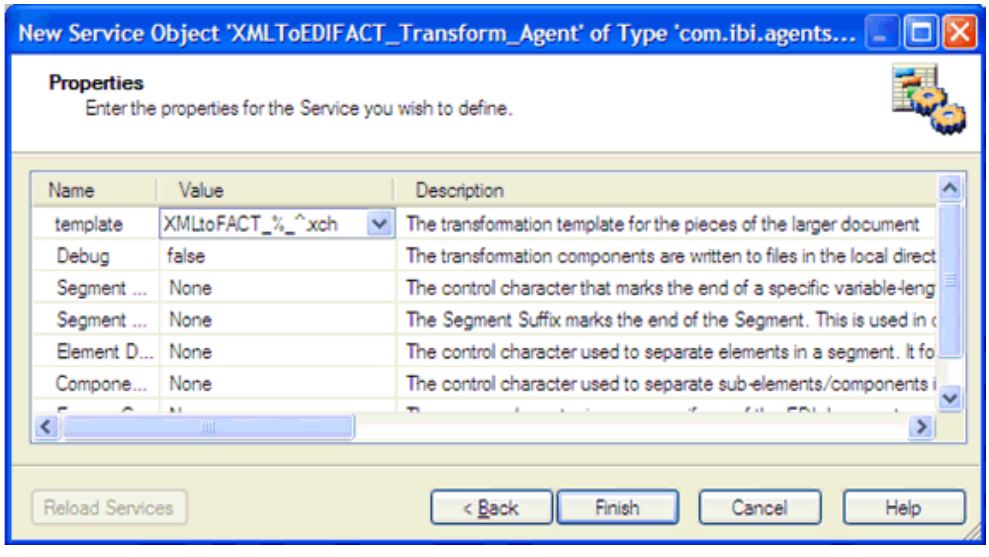
3. Click *Next*.

   The Service Type dialog box opens.

4. Select *Class Name* and enter *com.ibi.agents.XMLToEDIFACTTransformAgent*.

5. Click *Next*.

The Properties dialog box opens.



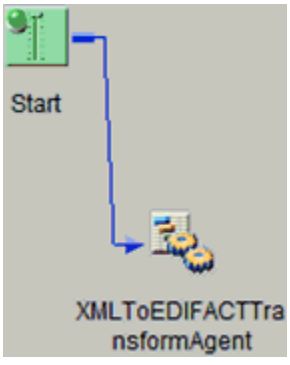
6. For the template parameter, enter the name of the transformation template, for example, *XMLtoFACT_%_^.xch*.
7. For the debug parameter, select *false* from the drop-down list.
8. Click *Finish*.

   The new Service object (XMLtoEDIFACTTransformAgent) appears in the workspace.
9. Select the *Start* object, right-click the *XMLtoEDIFACTTransformAgent* object, and select *Relation* from the context menu.

   The Line Configuration dialog box opens.
10. From the Event drop-down list, select *OnCompletion* and click *OK*.

    This option indicates that there are no conditions that affect the path, and that the path between the two objects will always be followed.

A line appears between the objects to indicate that a relationship has been established.



11. Drag and drop the File object from the toolbar to the workspace.

    The New File Object dialog box opens.

12. In the Name field, type *Write To Error Dir*, and a brief description (optional) in the Description field.

13. Click *Next*.

    The File Type dialog box opens.

14. From the Type drop-down list, select *File Write*.

15. Click *Next*.

    The Properties dialog box opens.

Name	Value	Description	Type
Source of Data		Source of data to write. If omitted, docu...	string
* Target Directory	sreg(XML.Error)	The target output directory	string
* File Pattern	error__sreg(basename)__*.xml	The output file name, which can contain ...	string
Avoid Preemitter	true	Should any preemitter be avoided?	boolean
Return	input	'status': status document will be the out d...	string
Base64 Decode	false	If set, the value is assumed to be in base...	boolean
* Respect Transactionality	false	If set, the emit respects the transactionali...	boolean
Call at EOS?	false	In streaming a last call is made AFTER th...	boolean

16. For the Target Directory parameter, enter a location where error data will be written, for example, *sreg(XML.Error)*.

17. For the File Pattern parameter, enter *error__sreg(basename)__*.xml*.

18. For the Return parameter, select *input* from the drop-down list.

19. Click *Finish*.

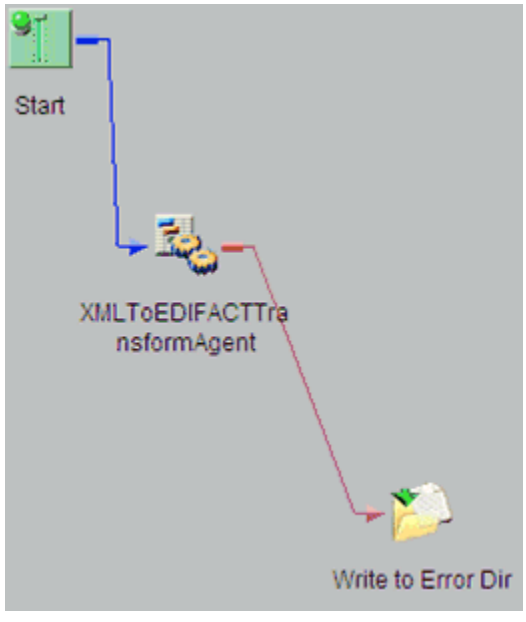
    The new File object (Write To Error Dir) appears in the workspace.

20. Select the *XMLtoEDIFACTTransformAgent* object, right-click the *Write To Error Dir* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

21. From the Event drop-down list, select *OnFailure* and click *OK*.

    A line appears between the objects to indicate that a relationship has been established.



22. Drag and drop the End object from the toolbar to the workspace.

    The End Name and Description dialog box opens.

23. In the Name field, type *End_Fail*, and a brief description (optional) in the Description field.
24. Click *Next*.

    The End Name Schema dialog box opens.

25. Since no schemas are used in this processing path (that is, the process flow will not be exposed as a web service), from the Schema drop-down list, select *None*.
26. Click *Next*.

    The Properties dialog box opens.

27. Click *Finish* to accept the default values and close the dialog box.

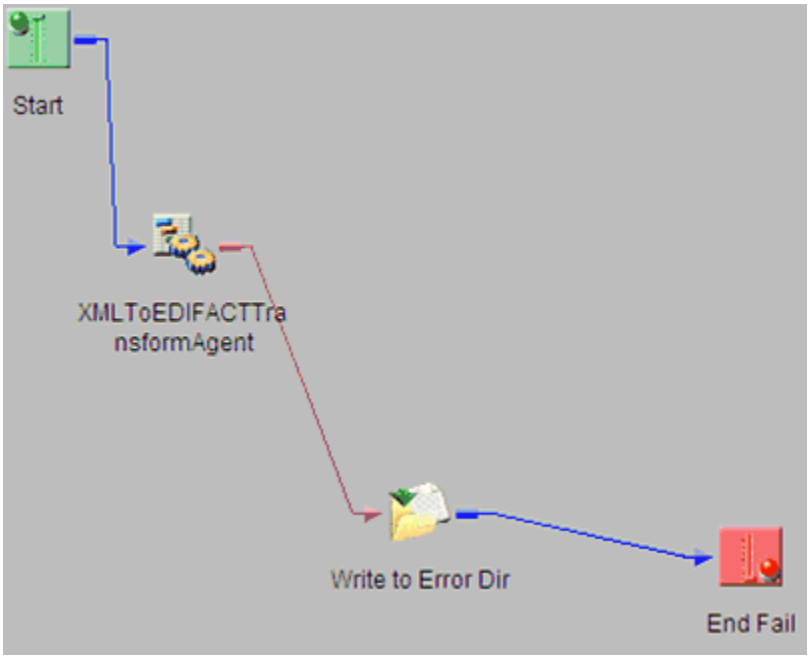
    The new End_Fail object appears in the workspace.

28. Select the *Write To Error Dir* object, right-click the *End_Fail* object, and select *Relation* from the drop-down list.

    The Line Configuration dialog box opens.

29. From the Event drop-down list, select *OnCompletion* and click *OK*.

    A line appears between the objects to indicate that a relationship has been established.



30. Drag and drop the Service object from the toolbar to the workspace.

    The New Service Object dialog box opens.

31. In the Name field, type *XDEDIFACTValidationReportAgent*, and a brief description (optional) in the Description field.

32. Click *Next*.

    The Service Type dialog box opens.

33. Select *Class Name* and enter *com.ibi.agents.XDEDIFACTValidationReportAgent*.

34. Click *Next*.

    The Properties dialog box opens.

35. Configure the available parameters according to your requirements.

36. Click *Finish*.

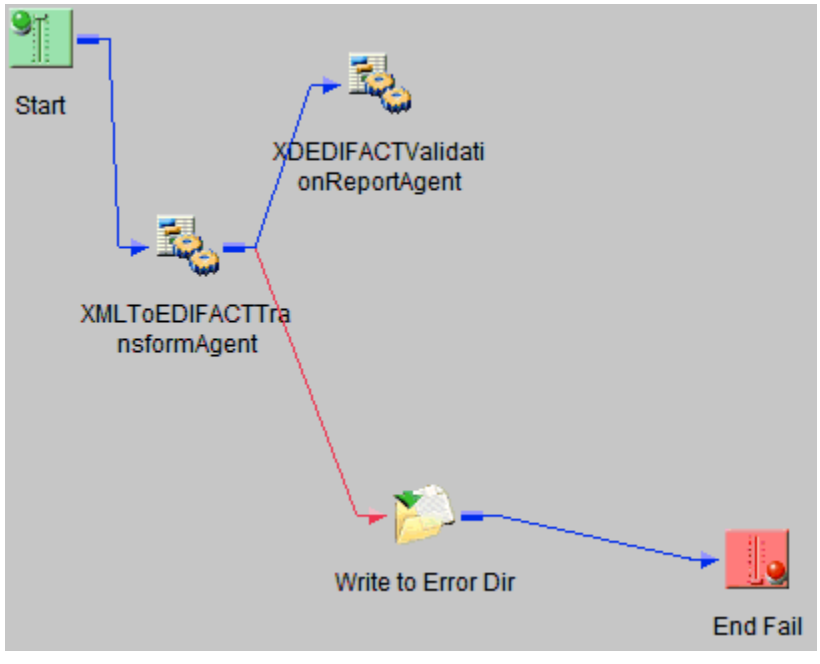
The new Service object (XDEDIFACTValidationReportAgent) appears in the workspace.

37. Select the *XMLtoEDIFACTTransformAgent* object, right-click the *XDEDIFACTValidationReportAgent* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.

38. From the Event drop-down list, select *OnCompletion* and click *OK*.

    A line appears between the objects to indicate that a relationship has been established.



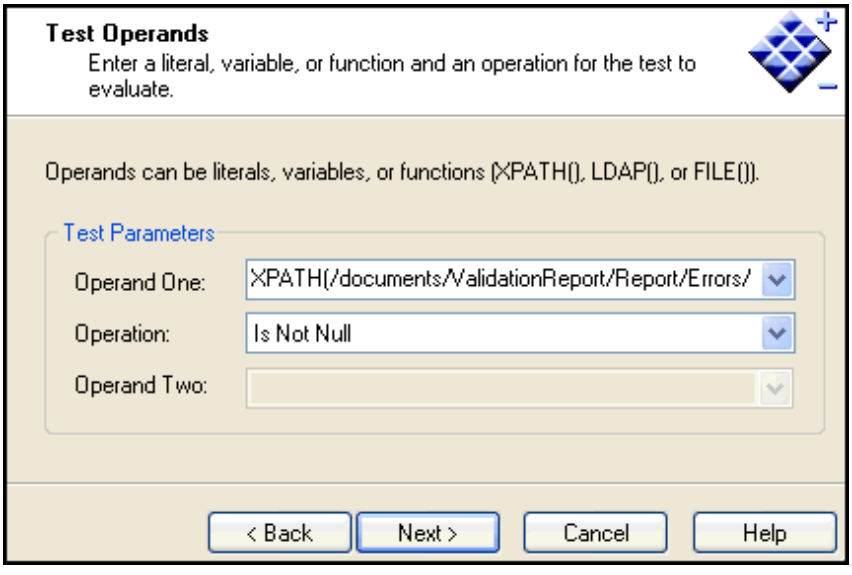
39. Drag and drop the Decision Test object from the toolbar to the workspace.

    The New Test Object dialog box opens.

40. In the Name field, type *Decision Test*, and a brief description (optional) in the Description field.

41. Click *Next*.

The Test Operands dialog box opens.

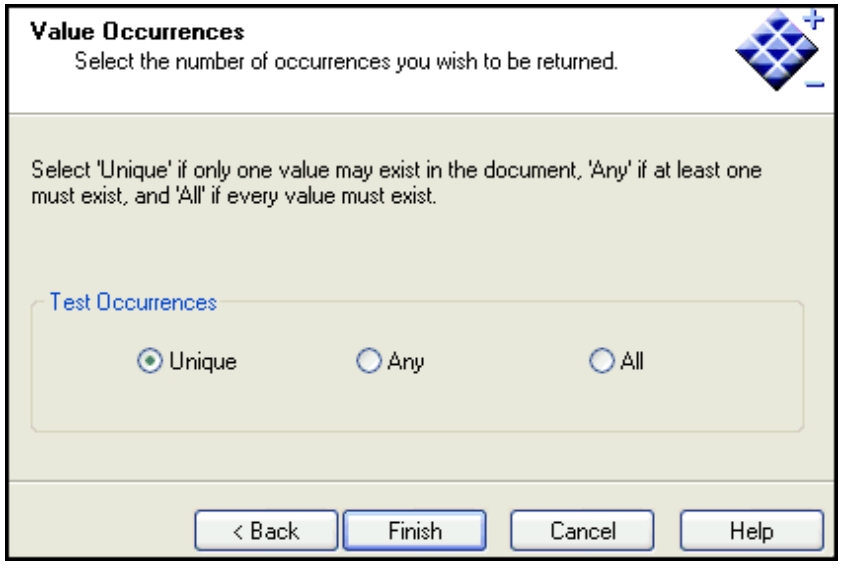


42. In the Operand One field, enter the following:

    ```
    XPATH(/documents/ValidationReport/Report/Errors/error)
    ```

43. From the Operation drop-down list, select *Is Not Null*.
44. Click *Next*.

The Value Occurrences dialog box opens.

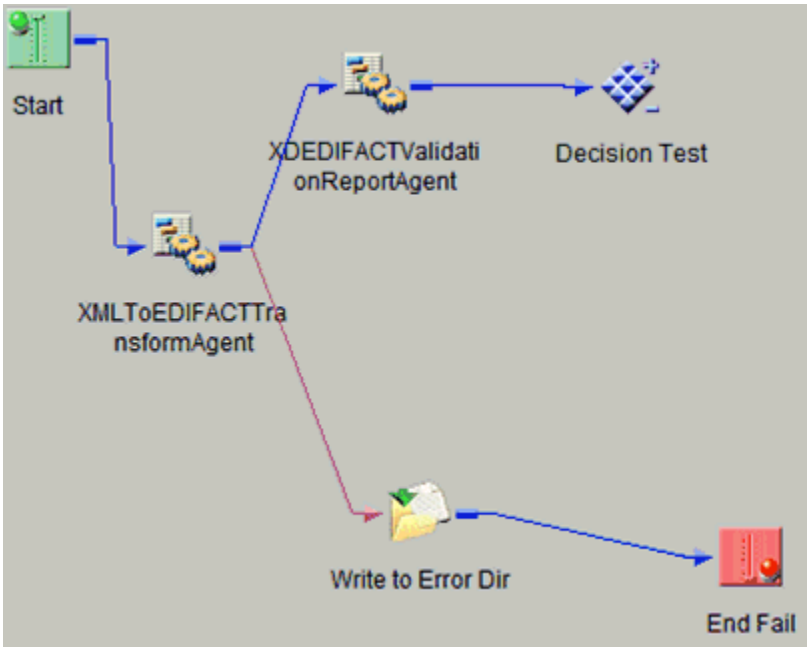


45. Ensure that *Unique* is selected from the available options.
46. Click *Finish*.

    The new Decision Test object appears in the workspace.
47. Select the *XDEDIFACTValidationReportAgent* object, right-click the *Decision Test* object, and select *Relation* from the context menu.

    The Line Configuration dialog box opens.
48. From the Event drop-down list, select *OnCompletion* and click *OK*.

A line appears between the objects to indicate that a relationship has been established.



49. Drag and drop the File object from the toolbar to the workspace.

    The New File Object dialog box opens.

50. In the Name field, type *Write Good File*, and a brief description (optional) in the Description field.

51. Click *Next*.

    The File Type dialog box opens.

52. From the Type drop-down list, select *File Write*.

53. Click *Next*.

    The Properties dialog box opens.

Name	Value	Description	Type
Source of ...	XPATH(/documents/output)	Source of data to write. If omitted, document will be used, else specify ...	string
* Target Dir...	sreg(XML.Output)	The target output directory	string
* File Pattern	sreg(basename)__*.edifact	The output file name, which can contain a '*' which gets expanded to ...	string
Avoid Pre...	true	Should any preemitter be avoided?	boolean
Return	input	'status': status document will be the out document. 'input': in document...	string
Base64 D...	false	If set, the value is assumed to be in base64 notation. Only applicable is...	boolean
* Respect T...	false	If set, the emit respects the transactionality of the channel. If not set, t...	boolean
Call at EO...	false	In streaming a last call is made AFTER the last document. Does this S...	boolean

54. For the Source of Data parameter, enter the following:

    ```
    XPATH(/documents/output)
    ```

55. For the Target Directory parameter, enter the following location where valid data will be written:

    ```
    sreg(XML.Output)
    ```

56. For the File Pattern parameter, enter the following:

    ```
    sreg(basename)__*.edifact
    ```

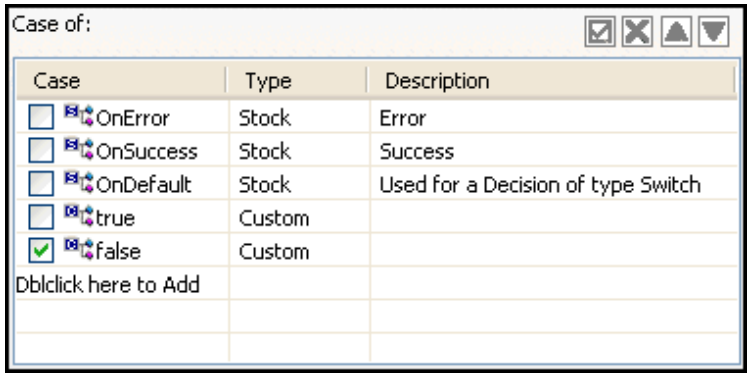
57. For the Return parameter, select *input* from the drop-down list.
58. Click *Finish*.

    The new File object (Write Good File) appears in the workspace.

59. Select the *Decision Test* object, right-click the *Write Good File* object, and select *Relation* from the context menu.

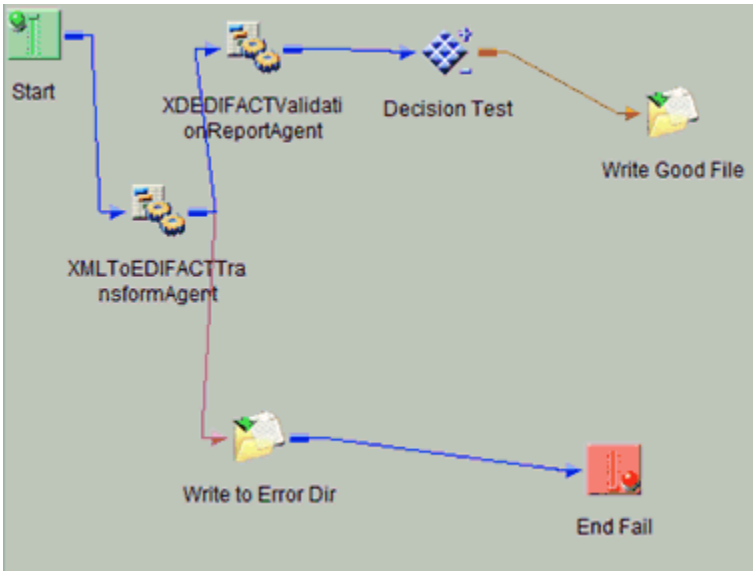
    The Line Configuration dialog box opens.

60. From the Event drop-down list, select *OnCustom*.



61. In the Case of section, select *false*.
62. Click *OK*.

A line appears between the objects to indicate that a relationship has been established.



63. Drag and drop the End object from the toolbar to the workspace.

    The End Name and Description dialog box opens.

64. In the Name field, type *End_Success*, and a brief description (optional) in the Description field.

65. Click *Next*.

    The End Name Schema dialog box opens.

66. Since no schemas are used in this processing path (that is, the process flow will not be exposed as a web service), from the Schema drop-down list, select *None*.

67. Click *Next*.

    The Properties dialog box opens.

68. Click *Finish* to accept the default values and close the dialog box.

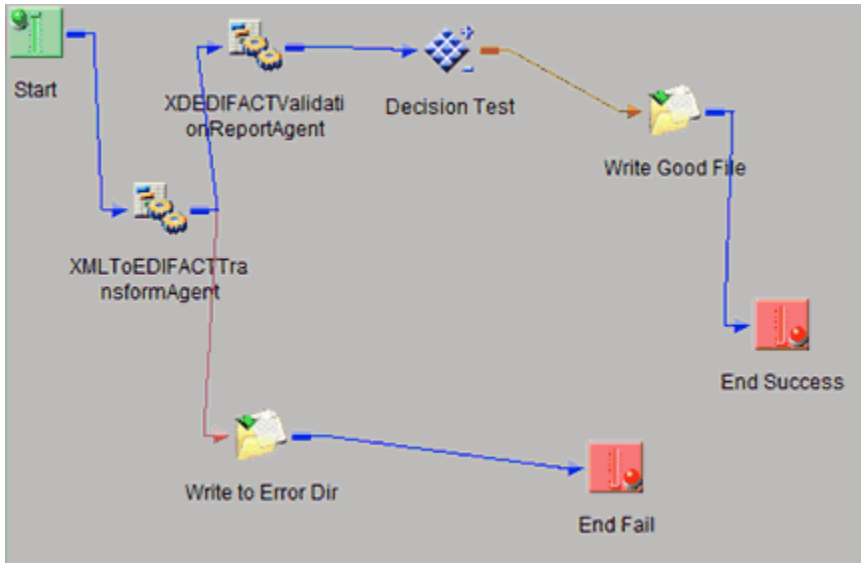
    The new End_Success object appears in the workspace.

69. Select the *Write Good File* object, right-click the *End_Success* object, and select *Relation* from the drop-down list.

    The Line Configuration dialog box opens.

70. From the Event drop-down list, select *OnCompletion* and click *OK*.

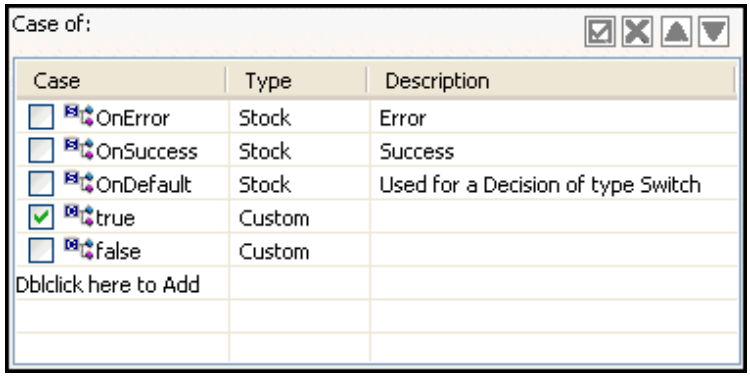
A line appears between the objects to indicate that a relationship has been established.



71. Select the *Decision Test* object, right-click the *End_Success* object, and select *Relation* from the context menu.

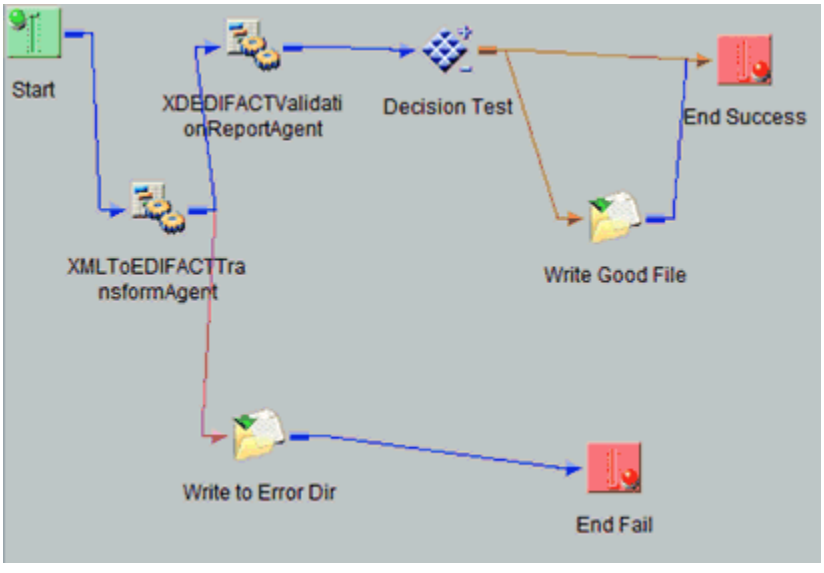
    The Line Configuration dialog box opens.

72. From the Event drop-down list, select *OnCustom*.



73. In the Case of section, select *true*.
74. Click *OK*.

A line appears between the objects to indicate that a relationship has been established.



The process flow is now complete.

75. To save the process flow, right-click the *XMLToEDIFACT_Ebix* node in the left pane and select *Save* from the context menu.

    Now you need to validate the process flow and publish it to the Registry of the iWay Service Manager Administration Console for use in the route of a channel for outbound processing.

    Validating a process flow ensures that its structure is correct. Publishing a process flow makes it available in the Registry for use in a channel configuration. For instructions on validating and publishing the process flow, see the *iWay Integration Tools Designer User's Guide*.

76. Close iIT Designer.

    Your next step is to add a new route to the Registry using the iWay Service Manager Administration Console and associate the process flow with it.

Procedure: How to Define a Route and Associate the Process Flow With It

To define a route and associate the process flow with It:

1. From the Registry menu options in the iWay Service Manager Administration Console, click *Routes*.

2. On the Route Definitions pane, click *Add* to add a route.
3. On the New Route Definition pane, enter a name for the route and an optional description, as shown in the following table.

Parameter	Value
Name *	XMLToEDIFACT
Description	This route will invoke the XML to EDIFACT validation process. The outcome of the validation process will place valid EDIFACT data in your valid outbound folder. Invalid EDIFACT will be routed to an errors folder. A validation report will also be sent to the appropriate folder.

4. Click *Finish*.
5. On the Construct Route pane, click *Add*.

   You are prompted for the type of component to associate with the route.
6. Select *Process* and click *Next*.
7. The next pane prompts you to select a process. Select the process flow you created earlier with iIT Designer, *XMLToEDIFACT_Ebix*, and click *Finish*.

   The route, with its associated process flow, has been successfully defined.

Defining an Outlet

For the iWay Integration Solution for UN/EDIFACT, you will add an emitter to the Registry, then associate it with a new outlet.

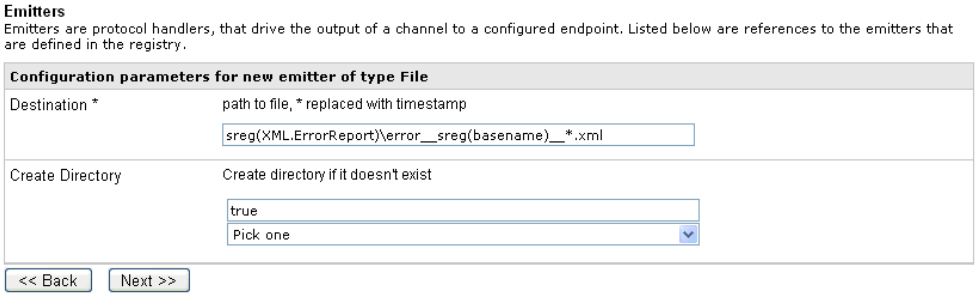
Procedure: How to Add an Emitter for an Error Validation Report

To add an emitter that will emit an error validation report and error file due to the XML to EDIFACT validation process:

1. From the Registry menu options, select *Emitters*.
2. On the Emitters pane, click *Add* to add an emitter.

   The next pane prompts you for the emitter type.
3. Select *File* from the drop-down list and click *Next*.

The File Emitter configuration parameters pane opens.



4. In the Destination field, enter the following:

   ```
   sreg(XML.ErrorReport)\error__sreg(basename)__*.xml
   ```

5. From the Create Directory drop-down list, select *true*.
6. Click *Next*.
7. On the Emitters pane, enter the name of the new emitter and an optional description, as shown in the following table:

Parameter	Value
Name *	XmltoEDIFACTEbix_XML_error
Description	XmltoEDIFACTEbix_XML

8. Click *Finish* to add the emitter.

Procedure: How to Add an Emitter for a Valid Validation Report

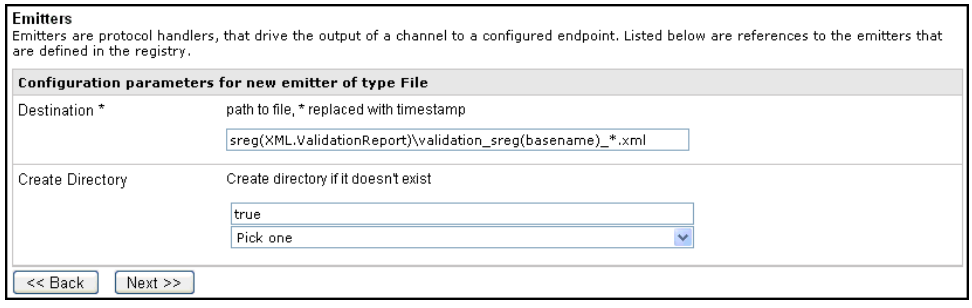
To add an emitter that will emit a valid validation report due to the XML to EDIFACT validation process:

1. From the Registry menu options, select *Emitters*.
2. On the Emitters pane, click *Add* to add an emitter.

   The next pane prompts you for the emitter type.

3. Select *File* from the drop-down list and click *Next*.

The File Emitter configuration parameters pane opens.



4. In the Destination field, enter the following:

   ```
   sreg(XML.ValidationReport)\validation_sreg(basename)_*.xml
   ```

5. From the Create Directory drop-down list, select *true*.
6. Click *Next*.
7. On the Emitters pane, enter the name of the new emitter and an optional description, as shown in the following table:

Parameter	Value
Name *	XmltoEDIFACTEbix_XML_validation
Description	XmltoEDIFACTEbix_XML

8. Click *Finish* to add the emitter.

Procedure: How to Define the Outlets

Now that you have added two emitters to the Registry, you are ready to define the required outlets. Each emitter will be associated with a corresponding outlet.

1. From the Registry menu options, select *Outlets*.
2. On the Outlet Definitions pane, click *Add* to add the first outlet.
3. On the New Outlet Definition pane, enter the name of the first new outlet and an optional description, as shown in the following table.

Parameter	Value
Name *	XmltoEDIFACTEbix_XML_error

Parameter	Value
Description	Outlet which will contain error validation report and error file due to the XML to EDIFACT validation process.

4. Click *Finish* to add the outlet.
5. On the Construct Outlet pane, click *Add* to associate the emitter with the outlet.

   The next pane prompts you for the component type.
6. Select *Emitter* and click *Next*.

   The next pane prompts you to select an emitter.
7. Select *XmltoEDIFACTEbix_XML_error*, which is the first emitter you added earlier, and click *Finish*.
8. On the Outlet Definitions pane, click *Add* to add the second outlet.
9. On the New Outlet Definition pane, enter the name of the second outlet and an optional description, as shown in the following table.

Parameter	Value
Name *	XmltoEDIFACTEbix_XML_validation
Description	Outlet which will contain valid validation report produced by the validation process.

10. Click *Finish* to add the outlet.
11. On the Construct Outlet pane, click *Add* to associate the emitter with the outlet.

    The next pane prompts you for the component type.
12. Select *Emitter* and click *Next*.

    The next pane prompts you to select an emitter.
13. Select *XmltoEDIFACTEbix_XML_validation*, which is the second emitter you added earlier.
14. Click *Finish*.

Defining a Channel

Now that you have defined the required components for the outbound channel, you are ready to add the channel to the Registry and associate the conduits with it. At this time you will also add the route to the channel.

Procedure: How to Define a Channel

To define a channel:

1. From the Registry menu options, select *Channels*.
2. On the Channel Definitions pane, click *Add* to add a channel.
3. On the New Channel Definition pane, enter the name of the new channel (for example, *XmlToEDIFACT_Ebix*) and an optional description. Then click *Finish* to add the channel.
4. On the Construct Channel pane, click *Add* to associate the inlet, route, and outlets with the channel.

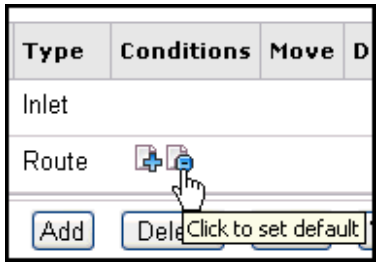
   You are prompted to associate components with the channel.
5. Select *Inlet* and click *Next*.
   The next pane prompts you to select an inlet.
6. Select *XmlToEDIFACT_Ebix*, which you defined earlier, and click *Finish*.

   The inlet is associated with the channel. Now you need to associate a route with the channel and set it to the default.
7. On the Construct Channel pane, click *Add*.

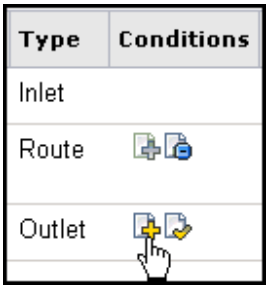
   The next pane prompts you for the component type.
8. Select *Route* and click *Next*.

   On the next pane, you are prompted to select a route.
9. Select *XmltoEDIFACTEbix*, which you defined earlier, and click *Finish*.

   The Construct Channel pane reopens.
10. Click the *minus sign* (-) under Conditions to set this route as the default.



11. On the Construct Channel pane, click *Add* to add the next component.
12. When prompted for the component type, select *Outlet* and click *Next*.
13. Select the two outlets you defined earlier, *XmltoEDIFACTEbix_XML_error* and *XmltoEDIFACTEbix_XML_validation*.
14. Click *Finish*.
15. To set a condition for the outlets, on the Construct Channel pane, click the *plus sign* (+) under Conditions for the specific outlet.



The Set Condition pane opens.

16. In the Condition input field, enter the appropriate conditional expression, and click *Update*.

The following table lists the expression that must be entered for each outlet.

Outlet	Expression
XmltoEDIFACTEbix_XML_validation	`_isxml() and sreg(iwaf.validationSuccess) = true`
XmltoEDIFACTEbix_XML_error	`_isxml() and sreg(iwaf.validationSuccess) != true`

For details on supported conditions, see the topic on using functions in the *iWay Service Manager User's Guide*.

Procedure: How to Add the Ebix to the Channel

1. From the Registry menu options, select *Channels*.

   The Channel Definitions pane opens.

2. Click the link in the Ebix column for the XmlToEDIFACT_Ebix channel.
3. On the next pane, which prompts you to add Ebix components, click *Add* to add the Ebix to the channel.

4. On the next pane, select *EDIFACT_D01B_EBIX*, which is the name of the Ebix you added previously, and click *Finish*.

Procedure: How to Build the Channel

1. From the Registry menu options, select *Channels*.
2. On the Channel Definitions pane, select the channel for outbound processing defined previously, *XmlToEDIFACT_Ebix*, and click *Build*.

   The results of the build are displayed on the right pane.
3. Review the results of your build and then click *Back*.

   If an error or errors are displayed in the Message column, take the appropriate action as instructed.

Procedure: How to Deploy the Channel

Deployment is the mechanism by which a channel moves from being stored in the Registry to becoming active in iWay Service Manager. For more information on deployment, see the *iWay Service Manager User's Guide*.

1. Select the *Deployments* option.
2. On the Channel Management pane, click *Deploy*.
3. On the Available Channels pane, select the channel you defined previously, *XmlToEDIFACT_Ebix*, and click *Deploy*.

   The Channel Management pane reopens.
4. Select *XmlToEDIFACT_Ebix* and click *Start*.

   The red X under Status changes to a green check mark to indicate that the channel has been started. If an error or errors are displayed, take the appropriate action as instructed.

Reusing Your Channel Configuration

Using the Archive Manager feature of iWay Service Manager, you can archive your channel configuration with its associated components and import them into another Registry. They will then be available from that Registry for modification or reuse.

For details on this feature, see the *iWay Service Manager User's Guide*.

Appendix A

UN/EDIFACT Supported Versions

This topic describes the UN/EDIFACT versions supported by the iWay Integration Solution for UN/EDIFACT in the Ebix files supplied with the product. All UN/EDIFACT message sets for the supported versions are included in the packaged Ebix files.

In this appendix:

UN/EDIFACT Version Support

The iWay Integration Solution for UN/EDIFACT supports all documents that belong to the versions listed in the following table:

❑ D93A	❑ D00A	❑ D06A
❑ D94A	❑ D00B	❑ D06B
❑ D94B	❑ D01A	❑ D07A
❑ D95A	❑ D01B	❑ D07B
❑ D95B	❑ D01C	❑ D08A
❑ D96A	❑ D02A	❑ D08B
❑ D96B	❑ D02B	❑ D09A
❑ D97A	❑ D03A	❑ D09B
❑ D97B	❑ D03B	❑ D10A
❑ D98A	❑ D04A	❑ D10B
❑ D98B	❑ D04B	❑ D11A
❑ D99A	❑ D05A	❑ D11B
❑ D99B	❑ D05B	❑ D12A
		❑ D12B
		❑ UNCONTRL

All UN/EDIFACT message sets are delivered within each supported version.

Document envelopes may be sent or received as either EDIFACT Syntax Version 3 or 4. APERAK and CONTRL messages are created with EDIFACT Syntax Version 3 or 4, depending on the options that are chosen during the acknowledgment service configuration.

Appendix B

Using iWay Integration Tools to Configure an Ebix for EDIFACT

This section describes how to use iWay Integration Tools (iIT) to configure an e-Business Information Exchange (Ebix) file for EDIFACT.

In this appendix:

Using iIT to Configure an Ebix File for EDIFACT Overview

You can use iWay Integration Tools (iIT) to import, edit, export, and work with e-Business Information Exchange (Ebix) files for EDIFACT. The topics in this appendix describe how to:

- Import an EDIFACT D97A INVOIC Ebix into iIT.
- Add a qualifier at the 01 [4439 Payment Conditions, Coded] element under the C534 loop, in the PAI loop level, to the EDIFACT Ebix.
- Export the edited Ebix to a physical location.

The edited Ebix can be returned and then tested with the appropriate EDIFACT D97A INVOIC message.

Using iIT to Configure an Ebix File for EDIFACT Prerequisites

This section provides a list of prerequisites for using iWay Integration Tools (iIT) to configure an Ebix for EDIFACT:

- Have a working knowledge of iIT and EDIFACT.
- Ensure the iWay EDIFACT adapter is installed.
- Ensure iIT Version 7.0.6 is installed.

Downloading and Extracting an Ebix

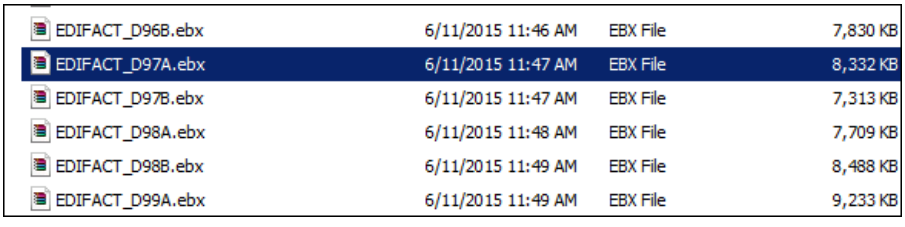
This section describes how to download and extract an Ebix.

Procedure: How to Download and Extract an Ebix

To download and extract an Ebix:

1. Download the *EDIFACT_ebixes.zip* file from *http://techsupport.informationbuilders.com*.
2. Unzip the downloaded *EDIFACT_ebixes.zip* file and save *EDIFACT_D97A.ebx* into any physical location on your local drive.

   For example, this Ebix contains the EDIFACT_D97A document.



   Note: Ensure all folders used for the extracted EDIFACT_ebixes.zip file do not have any blank spaces in the folder name.

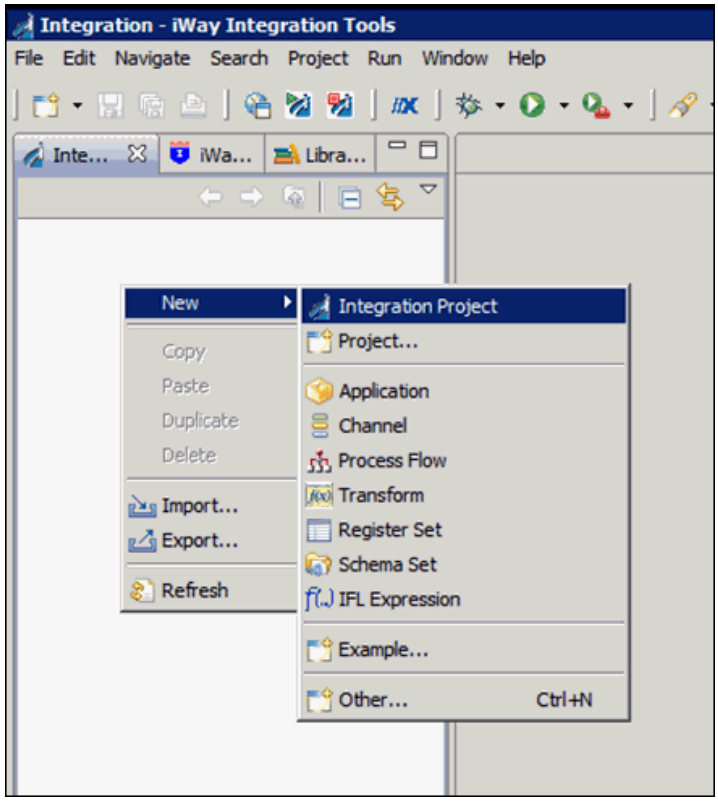
Working With iWay Integration Tools (iIT)

This section describes how to import, edit, and export an Ebix using iWay Integration Tools (iIT).

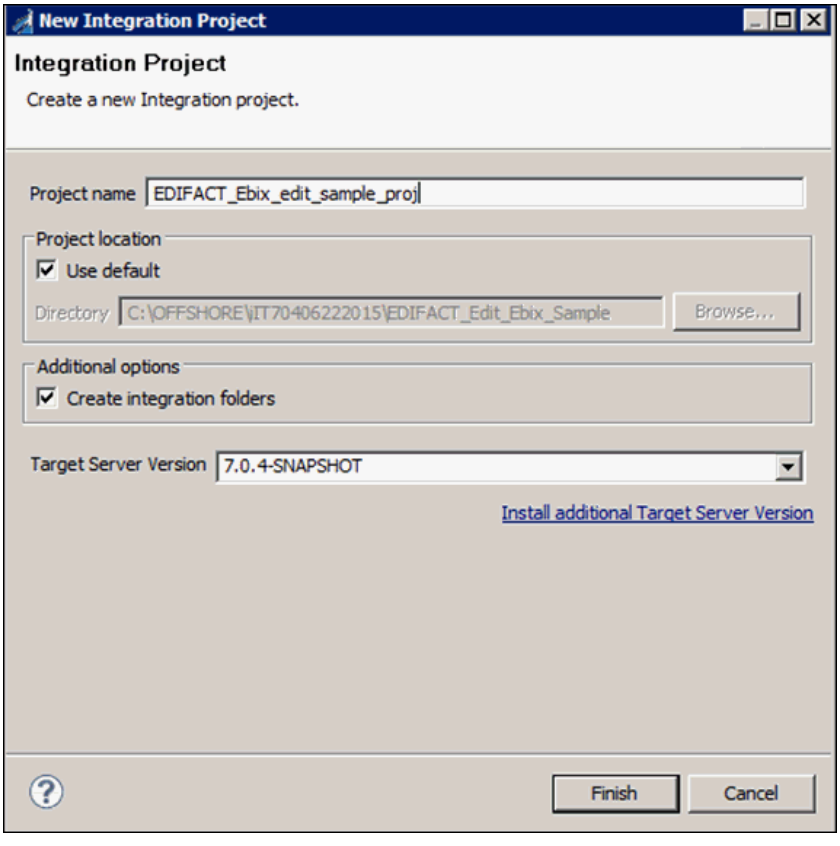
Procedure: How to Import an Ebix

1. Start iWay Integration Tools (iIT).

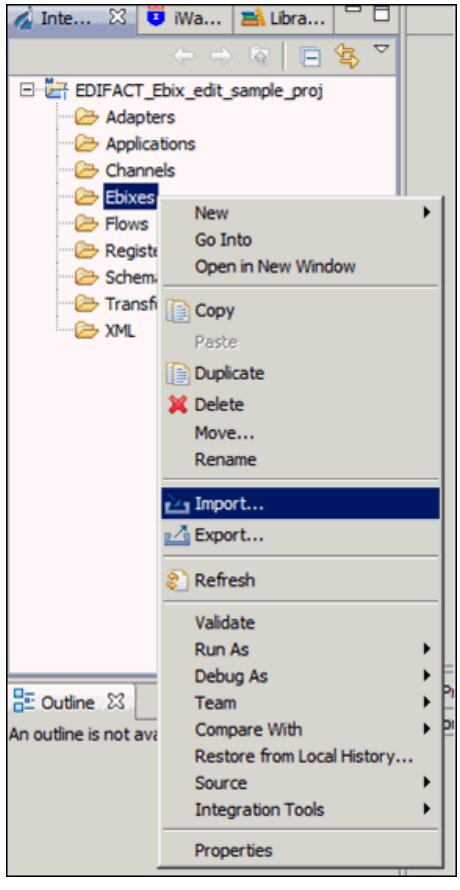
2. Right-click the Integration Explorer pane, click *New*, and then select *Integration Project* from the context menu, as shown in the following image.



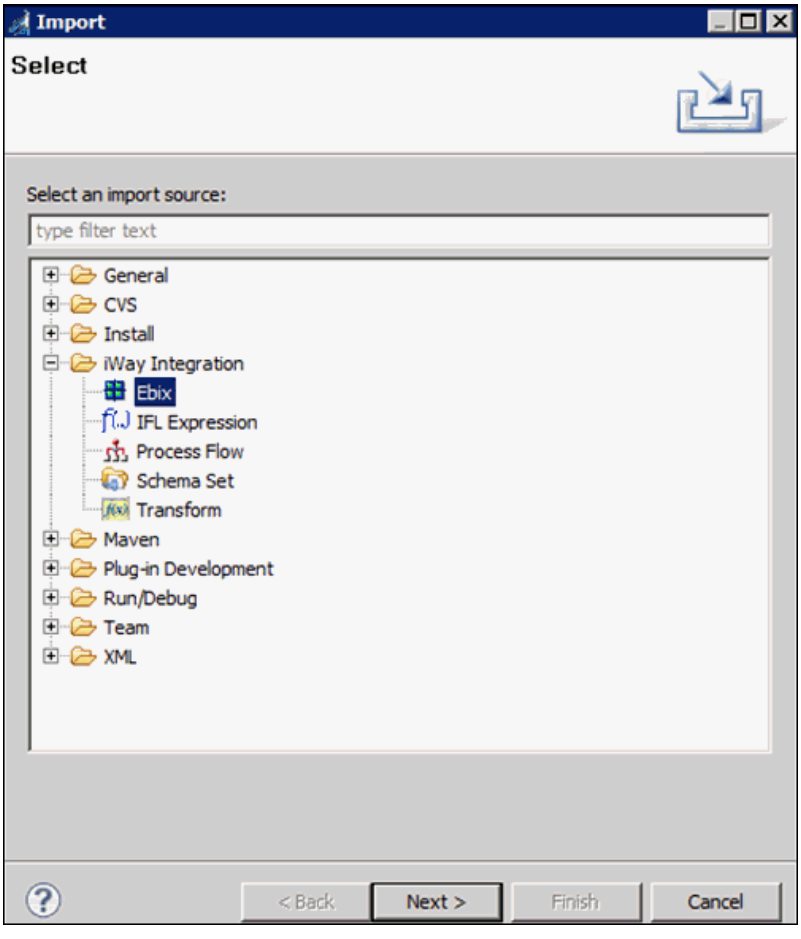
3. Enter a new Integration Project name, for example, *EDIFACT_Ebix_edit_sample_proj*, in the Project name field, and then click *Finish*, as shown in the following image.



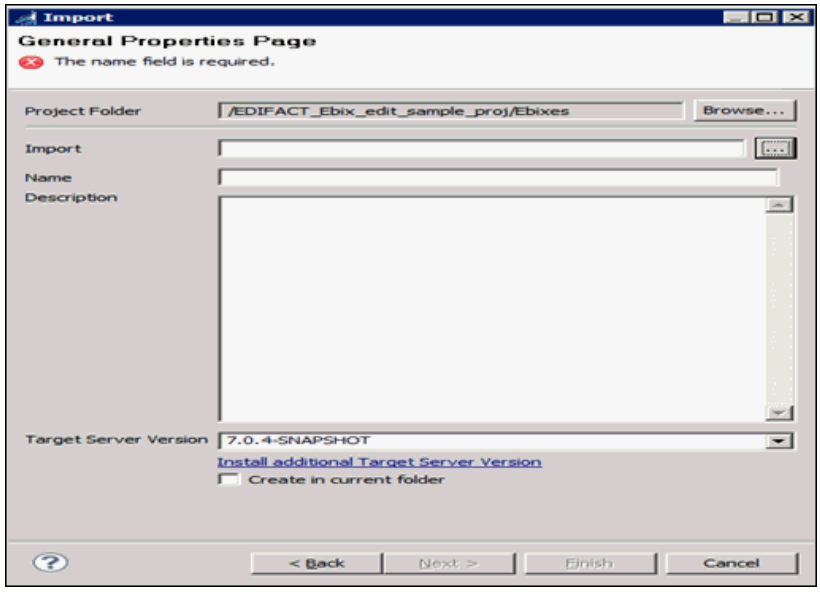
4. Right-click the Integration Explorer pane and select *Import* from the context menu, as shown in the following image.



5. In the Import wizard, expand *iWay Integration*, select *Ebix*, and then click *Next*, as shown in the following image.

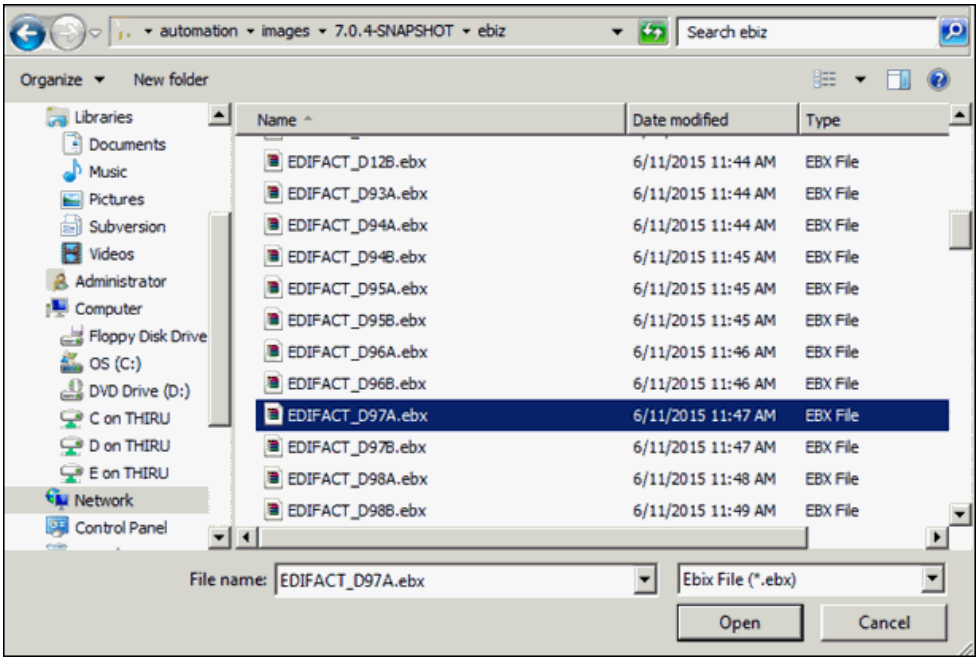


6. Click the *ellipsis* (...) button, as shown in the following image.

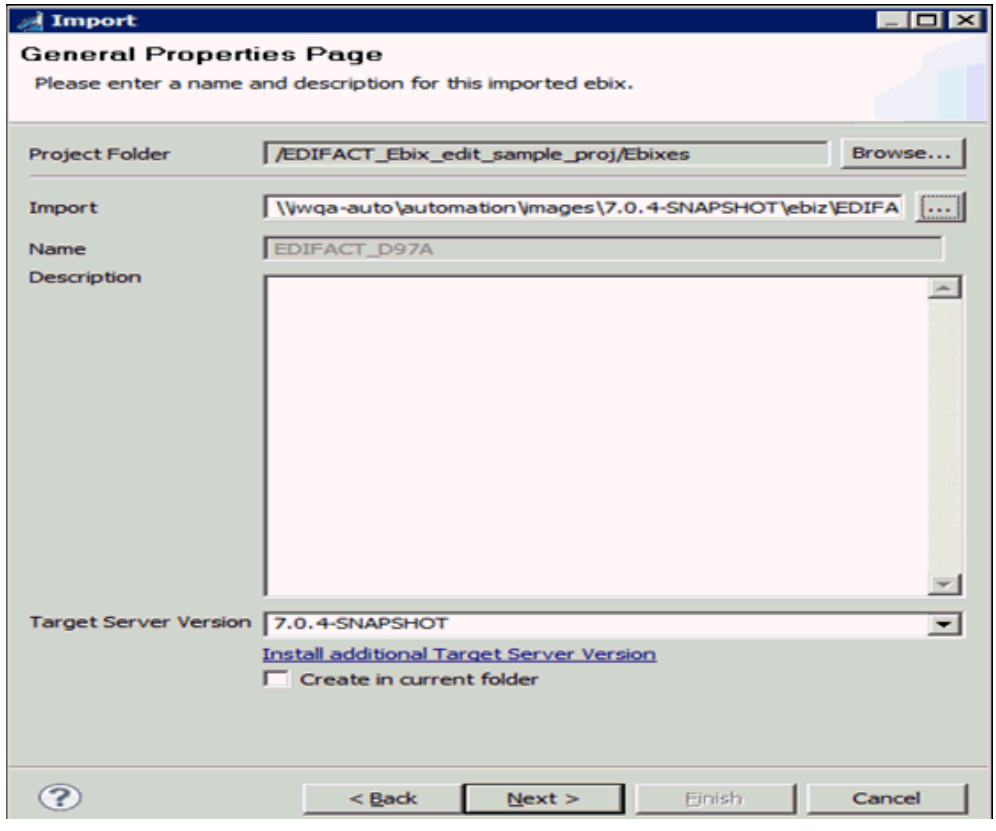


The Open dialog is displayed.

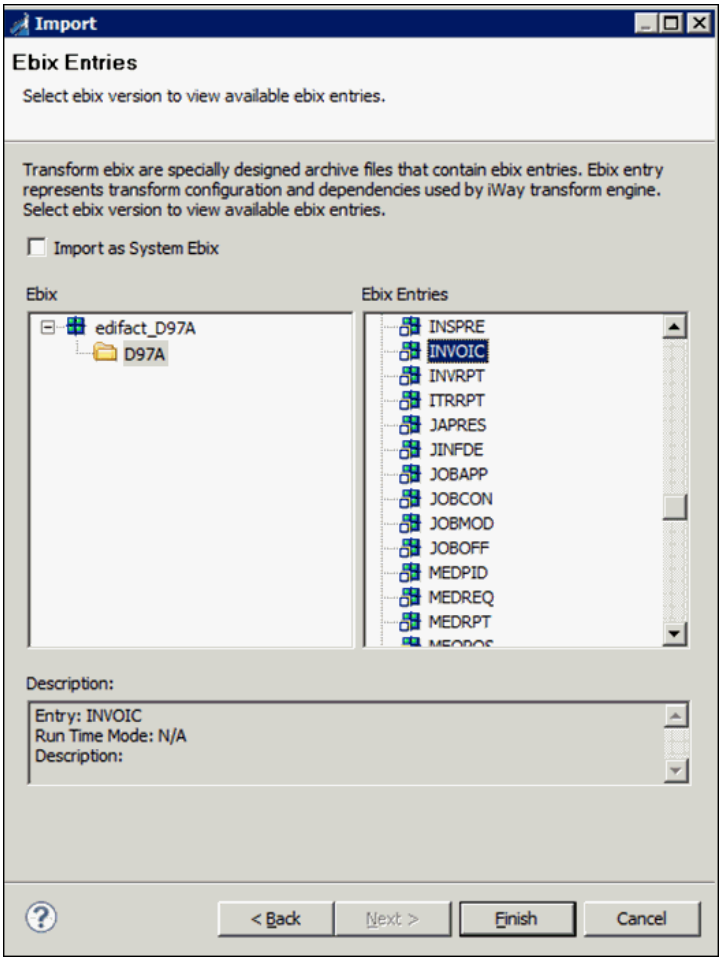
7. Select the downloaded *EDIFACT_D97A.ebx* file from the physical drive location and then click *Open*, as shown in the following image.



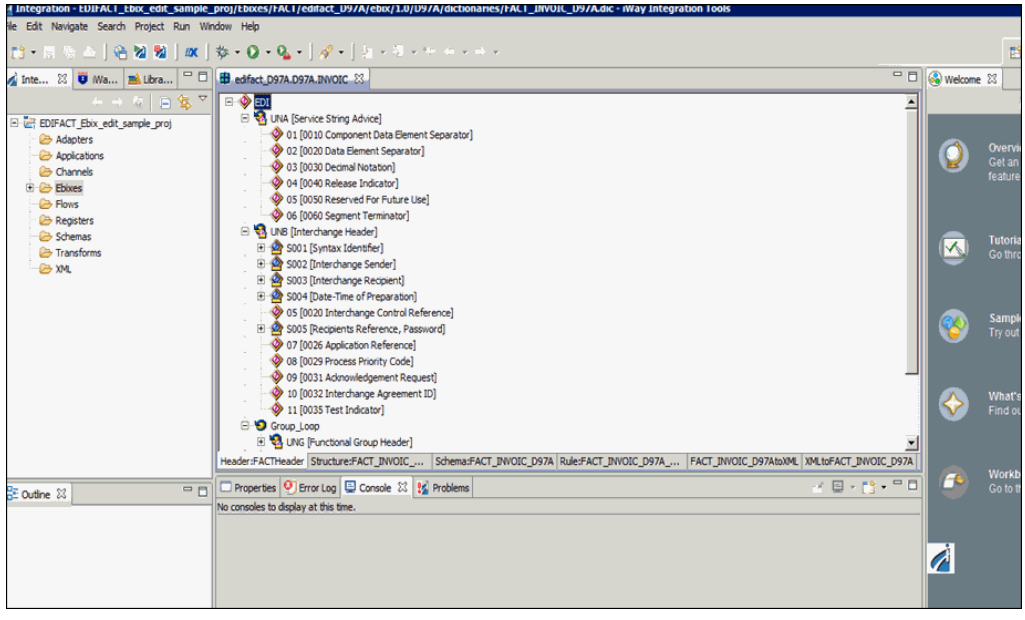
8. Click *Next*, as shown in the following image.



9. Expand *edifact_D97A* in the Ebix pane, click the *D97A* folder, select *INVOIC* in the Ebix Entries pane, and then click *Finish*, as shown in the following image.

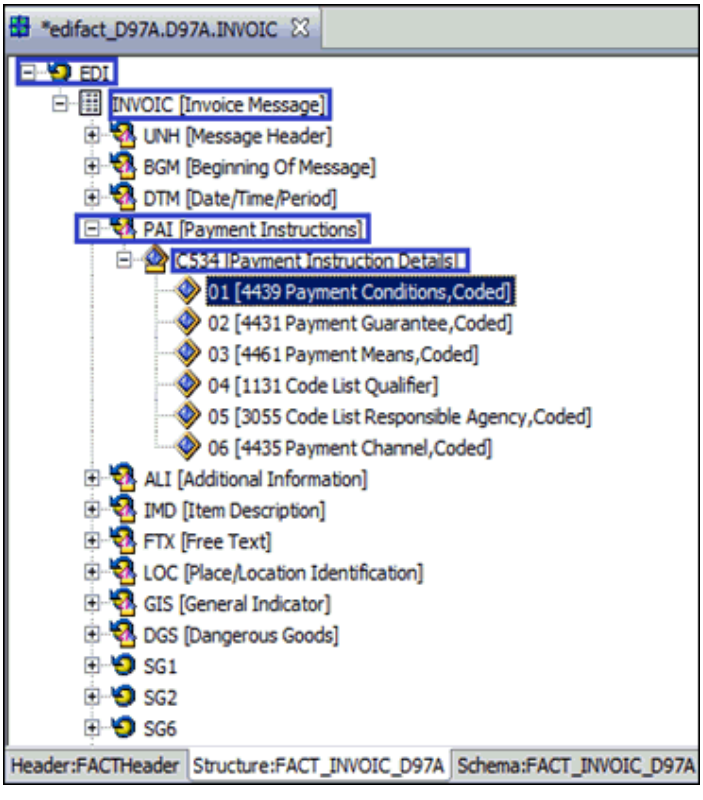


Your iIT interface should now resemble the following image:

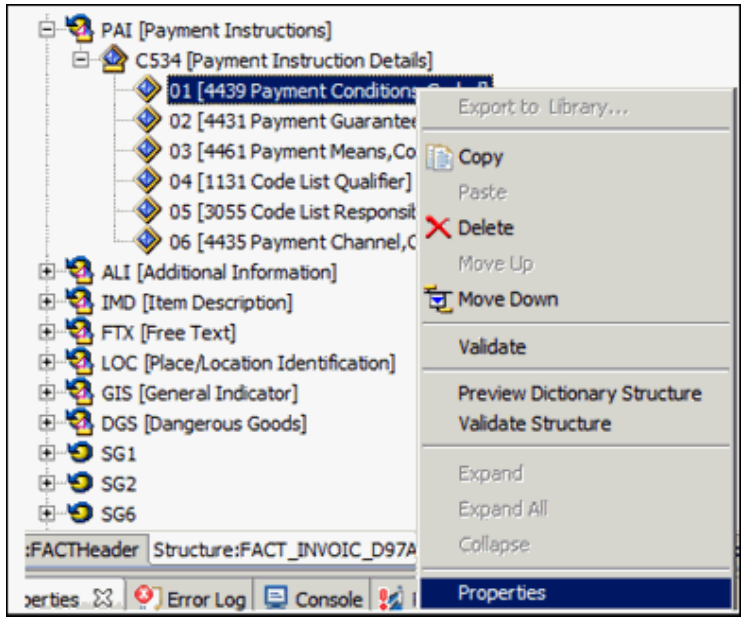


Procedure: How to Edit an Ebix

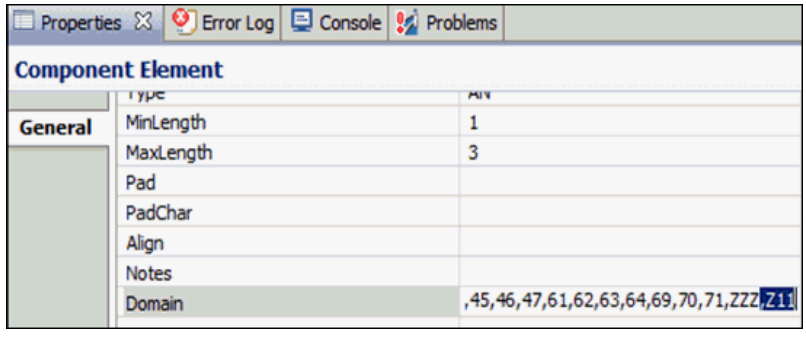
1. Click the *Structure:FACT_INVOIC_D97A* tab and navigate to the *01 [4439 Payment Conditions, Coded]* element by expanding *EDI*, *INVOIC [Invoice Message]*, *PAI [Payment Instructions]*, and then *C534 [Payment Instruction Details]*, as shown in the following image.



2. Right-click the *01 [4439 Payment Conditions, Coded]* composite element and then click *Properties* from context menu, as shown in the following image.

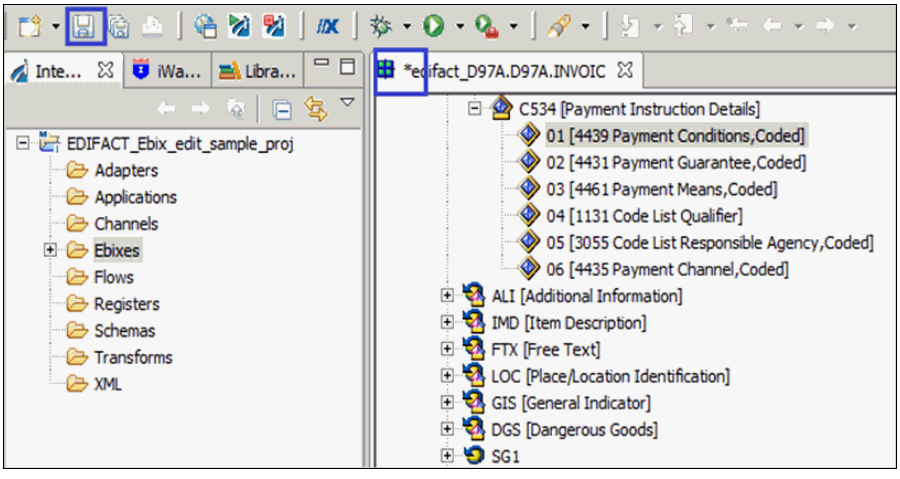


3. Scroll down to view the Domain value, and add *Z11* into the Domain value field in the properties window.

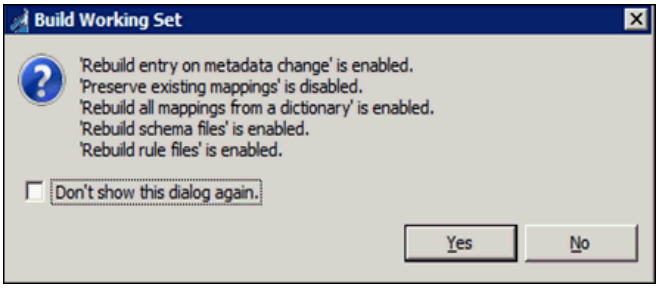


4. Save your edited Ebix by clicking the *Save* icon, which is located near the File menu. If you are using a Windows platform, you can also use the shortcut key CTRL+S to save your work.

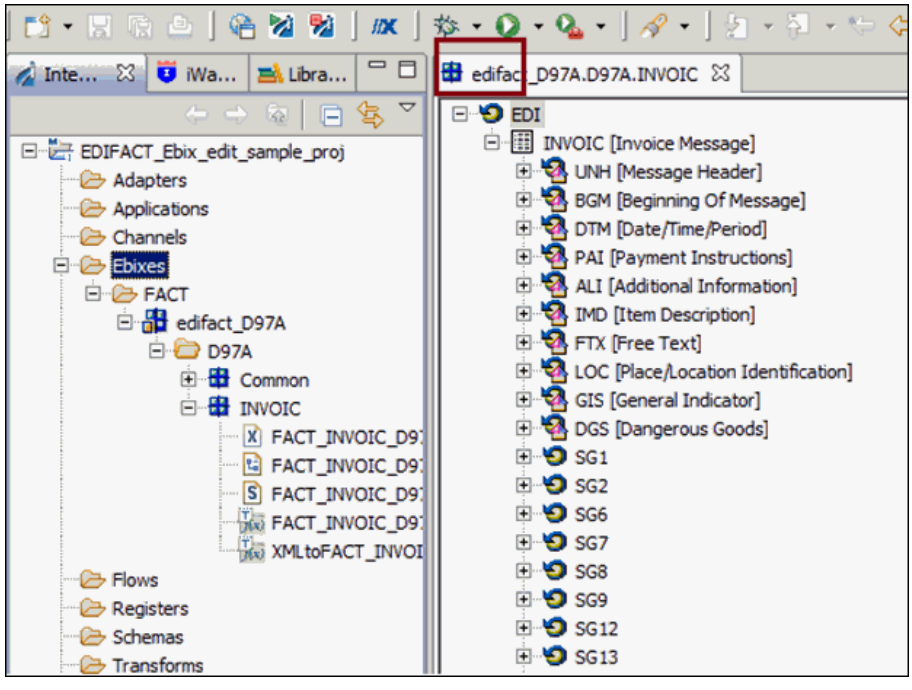
An asterisk (*) character appears next to the file name until you have saved the edited changes, as shown in the following image.



5. Click on *Yes* to confirm your changes.

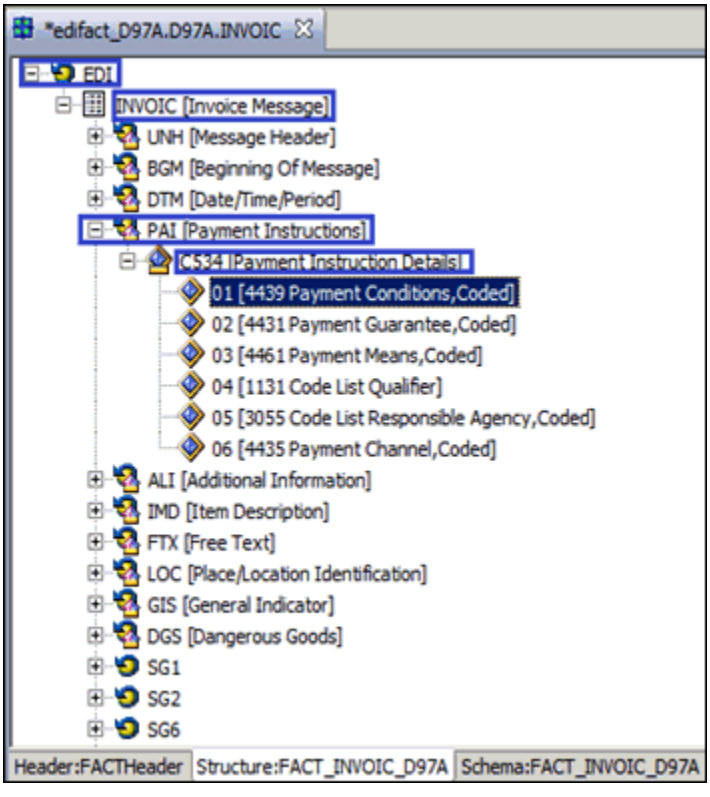


Your iIT interface should now resemble the following image:



Note: The asterisk (*) character will disappear once the edited Ebix has been saved successfully.

6. Click the *Structure:FACT_INVOIC_D97A* tab and navigate to the *01 [4439 Payment Conditions, Coded]* element by expanding *EDI*, *INVOIC [Invoice Message]*, *PAI [Payment Instructions]*, and then *C534 [Payment Instruction Details]*, as shown in the following image.

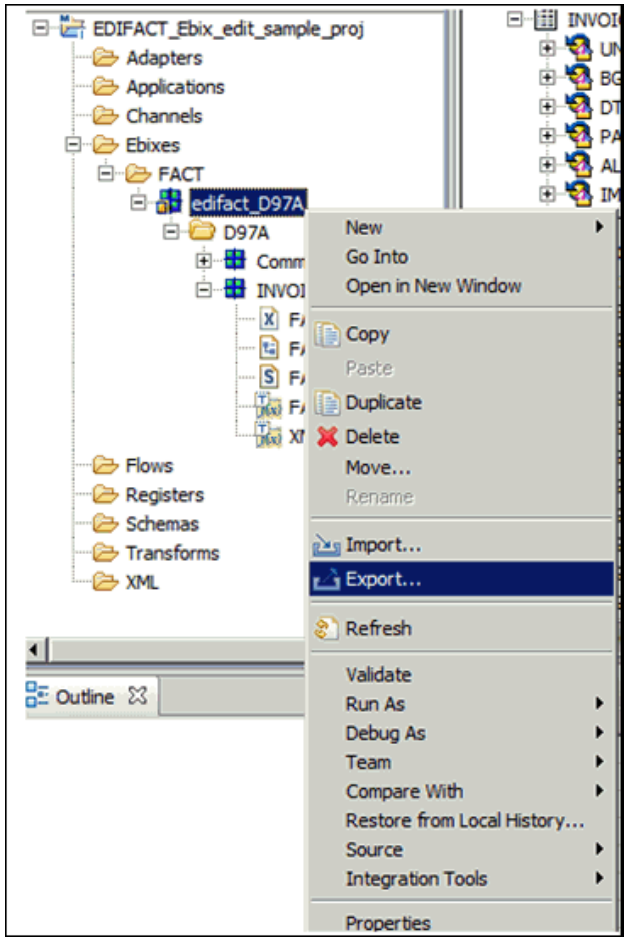


7. Repeat steps 2 - 4 in *How to Edit an Ebix* on page 156.

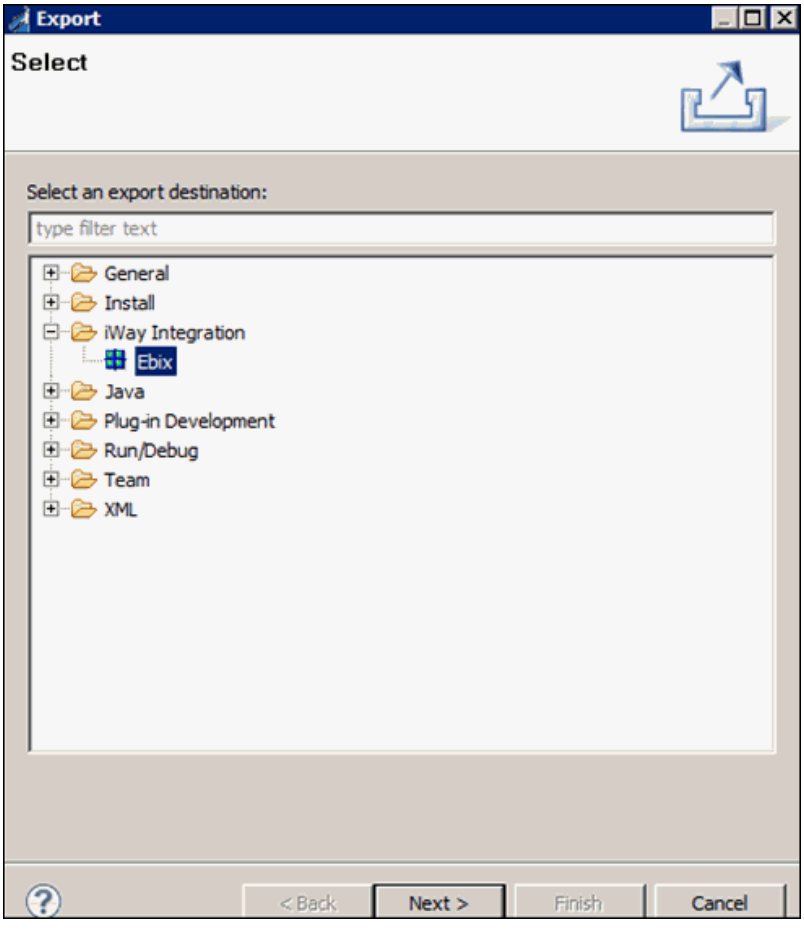
Procedure: How to Export an Ebix

To export an Ebix:

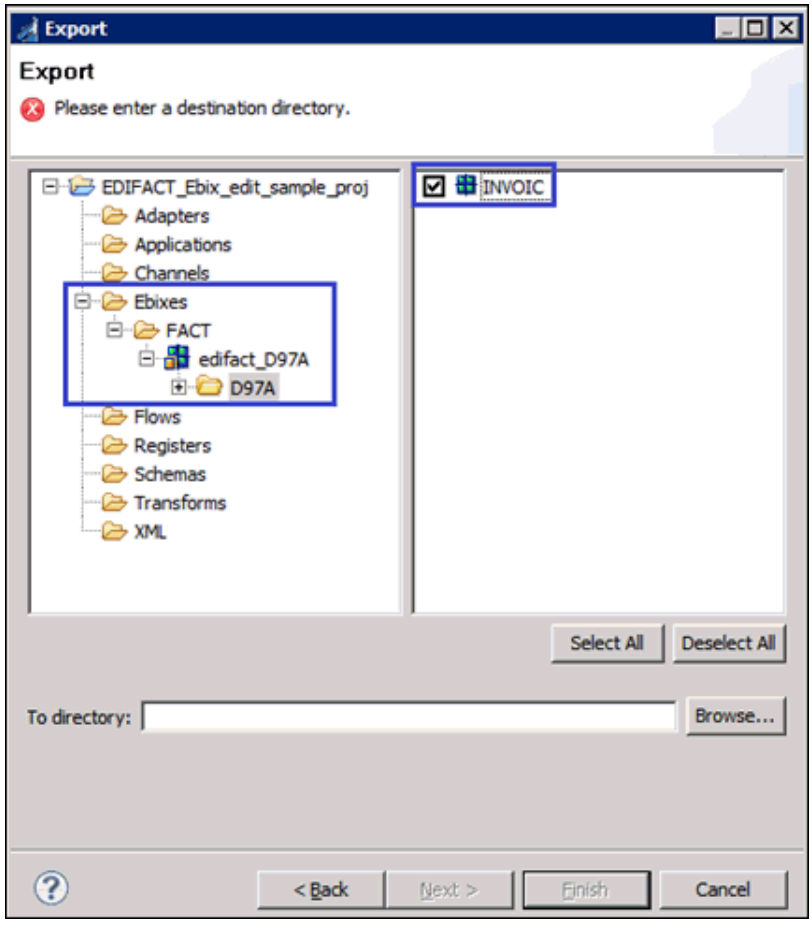
1. Right-click the *edifact_D97A* Ebix from the Integration Explorer window and then select the *Export* option from the context menu, as shown in the following image.



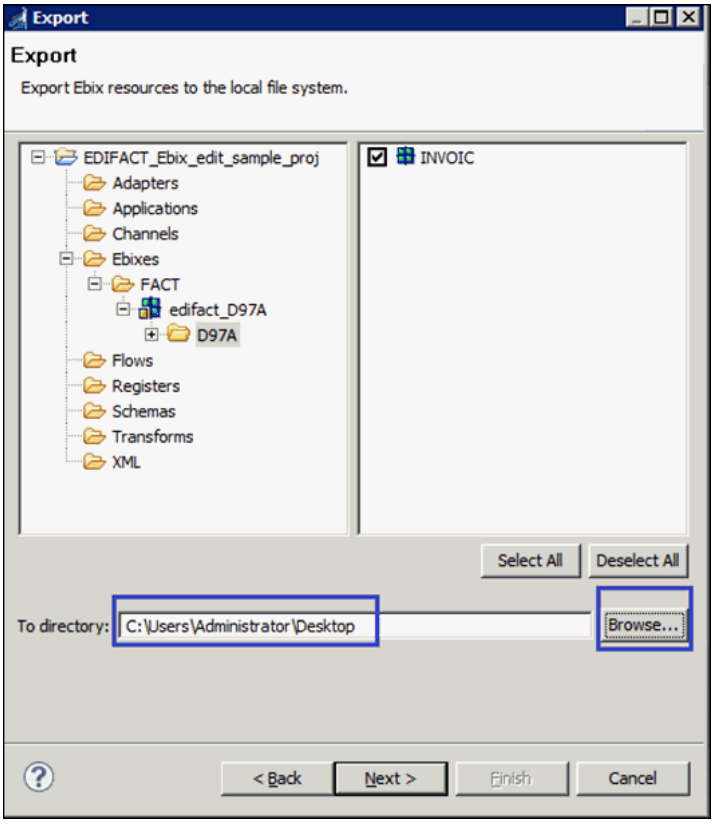
2. Expand the *iWay Integration* folder, select *Ebix*, and then click *Next*, as shown in the following image.



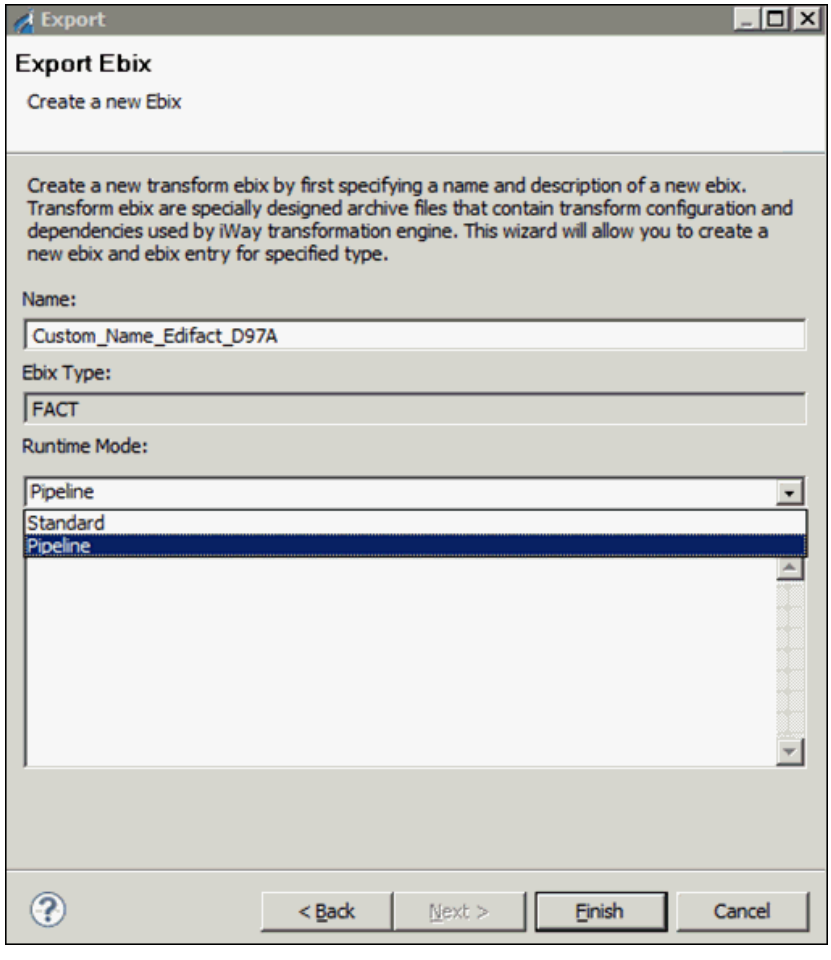
3. Expand *EDIFACT_Ebix_edit_sample_proj*, *Ebixes*, *FACT*, *edifact_D97A*, select the *D97A* folder in the left pane, and then select the *INVOIC* check box from the right pane, as shown in the following image.



4. Click *Browse* and choose a folder location to store the Ebix, and then click *Next*, as shown in the following image.



5. Provide a valid name for the Ebix in the Name field, select *Pipeline* from the Runtime Mode drop-down list, add a description (optional), and then click *Finish*, as shown in the following image.



Your exported Ebix is now available in the specified location.

Appendix C

EDIFACT Special Register (SREG) Types

This section describes the Special Register (SREG) types that are created during EDIFACT to XML transactions and acknowledgment creation.

In this appendix:

Special Register (SREG) Types

New Special Registers (SREGs) are available for EDIFACT preparsers and EDIFACT preemitters.

```
<variable type="USR" name="edi.transactionID" otype="0">ORDERS</variable>
<variable type="USR" name="edi.type" otype="0">EDIFACT</variable>
<variable type="USR" name="edi.version" otype="0">D00B</variable>
```

These may be used to route your data by placing them in your process flow. SREG(edi.ackstatus) is available from the XDEDIFACTAckAgent. This SREG will contain the acknowledgment status from APRAK or CNTROL that corresponds to each XML output file. This value can be used to route error data (for example, a failed EDIFACT document) from standard processing.

- SYS (System) - These SREGs exist until you restart iWay Service Manager.
- USR/DOC - These SREGs exist throughout the life of the document.
- CFG - These SREGs are configuration related.

```
CORRELATION ID

4. <variable name="correlid" type="USR">000001000</variable>
5. <variable name="doclocation" type="SYS">config</variable>

END OF STREAM FLAG

6. <variable name="eos" type="USR">1</variable>
7. <variable name="extension" type="DOC">EDIFACT</variable>
8. <variable name="filename" type="DOC">stephan_orders_bad.edifact</
variable>

FROM PARTY

9. <variable name="fromparty" type="USR">NOTP</variable>
```

GROUP CONTROL NUMBER

```
10. <variable name="ung_groupctlnumber" type="USR">1000</variable>
```

NUMBER OF TRANSACTIONS

```
11. <variable name="ge_numtransactions" type="USR">1</variable>
12. <variable name="ibse-port" type="CFG">9000</variable>
```

INTERCHANGE CONTROL NUMBER

```
13. <variable name="unb_interchangectlnum" type="USR">000001000</variable>
```

SPLIT COUNT

```
26. <variable name="splitcount" type="USR">1</variable>
27. <variable name="tid" type="DOC">EDI_XML-FILE-W.EDI_XML.
1_20080605152319600Z</variable>
```

TRANSACTION ID

```
28. <variable name=”edi.transactionID” type=”USR”>ORDERS</variable>
```

VERSION

```
29. <variable name=”edi.version” type=”USR”>D00B</variable>
```

Appendix D

Sample UN/EDIFACT Files

This appendix includes a sample UN/EDIFACT version D97A for APERAK (Application Error and Acknowledgment), INVOIC (Invoice Message), and ORDERS (Purchase Order Message).

For more information on obtaining UN/EDIFACT sample files for testing purposes, see *Extracting EDIFACT User Samples* on page 36.

In this appendix:

APERAK (Application Error and Acknowledgment)

The following is a sample EDIFACT version D97A APERAK (Application Error and Acknowledgment).

```
UNB+UNOA:1+RECIPIENT:12:ROUTING ADDR+SENDER PHONE:12:REVERSE ROUT
+090723:1010+1+RECEIVER PSWD:AA+APPL
REF+A+1+COMM AGREE ID+1
UNH+1+APERAK:D:97A:UN
BGM+23:::PRPAID+PRPAID1:D:97A+27+AP
DTM+137:20090723101039:204
ERC+36+6
FTX+AAO+++Maximum Usage Count exceeded for UNT
UNT+6+1
UNZ+1+1
```

Version 3 CONTRL

The following is a sample EDIFACT version 3 CONTRL.

```
UNB+UNOC:3+RECIPIENT:12+SENDER PHONE:12+090819:1435+1++++++1
UNH+GESMES1+CONTRL:D:3:UN+1F
UCI+1+SENDER PHONE+RECIPIENT+4
UNT+3+GESMES1
UNZ+1+1
```

INVOIC (Invoice Message)

The following is a sample EDIFACT version D97A EANCOM INVOIC (Invoice Message).

```
UNB+UNOA:2+MYCHAINCM:ZZ+925485MX00:8+080909:1357+807
UNH+000012102+INVOIC:D:97A:UN:EAN008
BGM+380+ABCD1234+9
DTM+137:19980110:102
RFF+ON:0123456789123
NAD+BY+1111111::31B
NAD+PE+2002450:31B
NAD+SU+3333333:31B
LIN+1
PIA+5+0835201031:IB
QTY+47:2
PRI+AAA:8.4
PRI+AAB:10.5::SRP
ALC+A
PCD+3:20
LIN+2
PIA+5+0835208338:IB
QTY+47:1
PRI+AAA:24
PRI+AAB:30::SRP
ALC+A
PCD+3:20
UNS+S
MOA+86:40.8
CNT+2:2
CNT+1:3
UNT+26+000012102
UNZ+1+807
```

ORDERS (Purchase Order Message)

The following is a sample EDIFACT version D97A ORDERS (Purchase Order Message).

```
UNB+UNOA:2+925485MX00:8+MYCHAINCM:ZZ+080909:1357+807
UNH+1+ORDERS:D:97A:UN
BGM+220::9+8900140287+9
DTM+137:20080909:102
DTM+43E:20080914:102
DTM+10:20080909:102
FTX+SPH+++==============================
FTX+SPH+++SI HAY ENTREGA A MULTIPLES
FTX+SPH+++DESTINOS EN LA MISMA FECHA,
FTX+SPH+++PRIMERO EMBARQUE AL DESTINO
FTX+SPH+++MAS LEJANO Y LUEGO AL MAS
FTX+SPH+++CERCANO.
FTX+SPH+++==============================
RFF+SD:46
RFF+ZZZ:63
RFF+PD:APER3884
NAD+BY+7507003106959::9++WAL-MART DC 7471
NAD+SF+++YOUR WAREHOUSE MEXICO SA CV
RFF+IA:330470000
NAD+FR+7507003100025::9++BODEGA
NAD+SN+7507003138844::9++CHALCO 2000     3884
CUX+2:MXN:9
PAT+1++21:3:D:60
TDT+20++30+31+:::VENDOR ROUTE
TOD+6+PO
LIN+1++3412245410013:EN
PIA+1+004663000:IN+11700005:VN
IMD+C+35+TROPIC::92
IMD+C+98+50ML::92
QTY+21:18
MOA+203:1553.64
PRI+AAB:86.3133::EUP::EA
PAC+6+3
UNS+S
MOA+86:1553.64
CNT+2:1
UNT+36+1
UNZ+1+807
```

Feedback

Customer success is our top priority. Connect with us today!

Information Builders Technical Content Management team is comprised of many talented individuals who work together to design and deliver quality technical documentation products. Your feedback supports our ongoing efforts!

You can also preview new innovations to get an early look at new content products and services. Your participation helps us create great experiences for every customer.

To send us feedback or make a connection, contact Sarah Buccellato, Technical Editor, Technical Content Management at *Sarah_Buccellato@ibi.com.*

To request permission to repurpose copyrighted material, please contact Frances Gambino, Vice President, Technical Content Management at *Frances_Gambino@ibi.com.*

Information Builders

iWay

iWay Integration Solution for UN/EDIFACT User's Guide

Version 7.0.x and Higher

DN3502265.0418

Information Builders, Inc.
Two Penn Plaza
New York, NY 10121-2898