Oracle® Fusion Middleware

User's Guide for Oracle B2B

11*g* Release 1 (11.1.1.6.2)

E10229-09

March 2012

Documentation for developers and users that describes how to use Oracle B2B to manage trading partner message exchange by using standard exchange and transport protocols, documents, trading partner agreements, listening channels, and packaging.

ORACLE®

Primary Author: Anirban Ghosh

Contributing Author: Mark Kennedy

Contributor: Oracle B2B development, product management, and quality assurance teams

Contents

Part II   Oracle B2B Process Flow

3   Creating Guideline Files

4   Creating Document Definitions

5   Configuring Trading Partners

6 Creating and Deploying Trading Partner Agreements

Part III   Oracle B2B Administration

7 Importing and Exporting Data

8 Using Document Protocols

9 Managing Deployments

17   Configuring B2B System Parameters

Part IV   Reports and Metrics

18   Creating Reports

19   Using B2B Metrics

Part V   Scripts and Utilities

20   B2B Command-Line Tools

21 Using the Oracle B2B Web Services

22 Scripts for Archiving and Restoring Data

23 Utilities for Enqueuing and Dequeuing

24 Monitoring Instance Message Data With Oracle BAM

25   Programmatically Accessing Instance Message Data

Part VI   Appendixes

A   Handling Large Payloads

B   Synchronous Request/Reply Support

C   Setting B2B Configuration Properties in Fusion Middleware Control

D   Back-End Applications Interface

E   Sequence Message Management

F Setting Up B2B Communication By Using Remote JNDI Queue

G Exception Handling

H Self Service Utility Protocols, Identifications, Security Specifications, and Parameters

Index

Preface

This guide describes how to use Oracle B2B.

Audience

Oracle Fusion Middleware User's Guide for Oracle B2B is intended for businesses that need to extend business processes to trading partners, and want to design, deploy, monitor, and manage business process integrations.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For information about Oracle SOA Suite products, see the following:

- *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*

- *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

- *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite*

- *Oracle Fusion Middleware User's Guide for Technology Adapters*

- *Oracle Fusion Middleware Performance Guide*

For information about the Java API documentation (Javadoc), see the following:

- *Oracle Fusion Middleware B2B Callout Java API Reference*

- *Oracle Fusion Middleware B2B Instance Message Java API Reference*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in This Guide for Release 11.1.1.6.x

For Release 11.1.1.6.x, this guide has been updated in several ways. The following table lists the sections that have been added or changed. If a feature was not available in the first release of 11.1.1.6.x, the last columns denote which documentation release contains the update.

For a list of known issues (release notes), see the "Known Issues for Oracle SOA Products and Oracle AIA Foundation Pack" at
`http://www.oracle.com/technetwork/middleware/docs/soa-aiafp-known` `nissuesindex-364630.html`.

Sections	Changes Made	11.1.1.6.0	11.1.1.6.1
Chapter 5 Configuring Trading Partners			
Section 5.10, "Validating Certificate in Inbound Message Processing"	Section added on partner certificate validation.		X
Chapter 20 B2B Commandline Tools			
Section 20.3, "Purging Data"	Section updated to include a new commandline tool.		X
Appendix C Setting B2B Configuration Properties in Fusion Middleware Control			
Section C.1, "Properties To Set in Fusion Middleware Control"	Section revised add new properties.		X

Part I

Introduction to Oracle B2B

This part contains the following chapters:

1

Introduction to Oracle B2B

This chapter discusses how Oracle B2B, an e-commerce gateway, enables the secure and reliable exchange of business documents between an enterprise and its trading partners. It also covers how Oracle B2B supports business-to-business document standards, security, transports, messaging services, and trading partner management.

With Oracle B2B used as a binding component within an Oracle SOA Suite composite application, end-to-end business processes can be implemented. Oracle B2B also supports Health Level 7, which enables health care systems to communicate with each other.

For more information about Oracle SOA Suite, see *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

This chapter contains the following topics:

- Section 1.1, "Oracle B2B and Business-to-Business E-Commerce"
- Section 1.2, "Protocols Supported in Oracle B2B"
- Section 1.3, "Oracle B2B Metadata"
- Section 1.4, "Security Features of Oracle B2B"
- Section 1.5, "How Does Oracle B2B Fit into a SOA Implementation?"
- Section 1.6, "Sending a Purchase Order: An Example of a SOA Implementation"
- Section 1.7, "Oracle B2B Samples and Cookbooks"
- Section 1.8, "Administering Oracle B2B in the Oracle Fusion Middleware Environment"

1.1 Oracle B2B and Business-to-Business E-Commerce

E-commerce is the buying and selling of products or services over the Internet, including business-to-business (B2B). In B2B e-commerce, an enterprise extends its business processes over the Internet to reach trading partners. B2B e-commerce represents classic business processes, mature business documents, and industry-tempered messaging services. It requires a unified business process platform, end-to-end instance tracking, visibility and auditing, integrated process intelligence, process and service governance, and centralized security.

You can think of an e-commerce transaction between businesses as analogous to a mail or express carrier (shipping) transaction. In both kinds of transactions, the sender must consider the details required for packaging and sending an item, and the receiver's requirements. Table 1–1 provides an example that compares the two kinds of transactions.

Table 1–1   Comparing Traditional and E-Commerce Transactions

	Traditional Shipping Transaction	**E-Commerce Transaction**
What is the item to be shipped, that is, the transaction item?	Cell phone	Electronic document
		Document protocols: Custom, EDI EDIFACT, EDI X12, HL7, OAG, positional flat file, RosettaNet, UCCnet, and more
How is the item packaged?	Box, bubble wrap	Packaging protocols: MIME, SMIME, SOAP, XMLDSig, XMLEncrypt
How is the item sent and received?	Truck, ship, airplane	Transport protocols: HTTP, File, FTP, SFTP (SSH FTP), TCP/IP, SMTP, MLLP, and more
Who is the carrier?	DHL, FedEx, UPS, USPS	Message exchange protocols: RNIF, AS1, AS2, ebMS, and more
What carrier services are required?	Required? ■ Signed receipt ■ Overnight/next day ■ Delivery attempts	Required? ■ Nonrepudiation ■ Time to acknowledge/respond ■ Retry counts

This guide describes how to use Oracle B2B to define the document, the packaging, and the delivery, in addition to configuring trading partners, creating and deploying agreements, and monitoring a deployment.

1.2 Protocols Supported in Oracle B2B

Oracle B2B supports numerous industry-standard e-commerce protocols, as defined for a range of industries, including health care, retail, IT, telecom, electronics, manufacturing, the food industry, and more. Table 1–2 lists the protocols supported in Oracle B2B.

Table 1–2   Protocols Supported in Oracle B2B

Protocol Type	Protocol
Document protocol	■ Custom (user-defined)
	■ EDI EDIFACT, all versions
	■ EDI X12, all versions
	■ HL7, all versions
	■ RosettaNet PIP business documents
	■ OAG
	■ Positional flat file (includes SAP iDoc)
	■ UCCnet
	■ Custom (non-XML)
	■ NCPDP Telecom
	■ EDIEL
Packaging protocol	■ MIME 1.0
	■ S/MIME 2.0, S/MIME 3.0
	■ SOAP
	■ XML digital signature (XMLDSig)
	■ XML encryption (XMLEncrypt)
Transport protocol	■ AQ
	■ Email (SMTP 1.0, IMAP 1.0, POP3)
	■ File
	■ FTP and SFTP (SSH FTP)
	■ HTTP (HTTP 1.0, HTTP 1.1) and HTTPS (HTTPS 1.0, HTTPS 1.1)
	■ JMS
	■ TCP/IP
Message exchange protocol	■ AS1-1.0, AS2-1.1
	■ MLLP-1.0
	■ ebMS-1.0, ebMS-2.0 (ebXML Messaging Service)
	■ RosettaNet-01.10, RosettaNet-V02.00
	■ Generic File-1.0
	■ Generic AQ-1.0
	■ Generic FTP-1.0
	■ Generic SFTP-1.0
	■ Generic JMS-1.0
	■ Generic HTTP-1.0
	■ Generic Email-1.0
	■ Generic TCP

> **About Document Types:**   Using the Custom and positional flat file document protocols, you can use many other document types, including W3CXML Schema (OAGIS, xCBL, UBL, ebXML, and more). Use Oracle Document Editor to create the guideline documents.

1.3 Oracle B2B Metadata

Oracle B2B instance data is stored and managed within the SOAINFRA schema of your database. Oracle B2B metadata for design-time and configuration is stored and managed through Metadata Services (MDS), available in Oracle Fusion Middleware. See *Oracle Fusion Middleware Administrator's Guide* for more information about MDS.

Because documents created in Oracle B2B are stored in the metadata repository, it is possible that the transaction log for the database can become full. If this occurs often, increase the database configuration parameter to allow a larger log file. A larger log file requires more space but reduces the need for applications to retry the operation.

To increase this value, issue the following command:

```
db2 update database config for soainfra using LOGFILESIZ 8192
```

1.4 Security Features of Oracle B2B

Oracle B2B leverages the security features of Oracle Platform Security Services, a comprehensive security platform framework. Oracle Platform Security Service supports:

- Authentication
- Identity assertion and management
- Authorization
- The specification and management of application-specific policies
- Credential and key store management through the Credential Store Framework
- Auditing
- Role administration, and role and credential mappings
- The User and Role API
- Single sign-on solutions
- Security configuration and management
- Cryptography

The default administrator user created during Oracle SOA Suite installation is assigned the administrator role, which has access to all Oracle B2B functionality. The default administrator user can create additional users and assign the following roles:

- Host administrator—This role has access to all Oracle B2B functionality. Only a host trading partner user can have the administrator role for all data.
- Host monitor—This role can access reports and view run-time data for all trading partners.
- Remote administrator—This role has limited access to the Partners page. Users with this role can view and edit only their own design data (channels, documents, and so on); can view only those agreements for which they are a partner; and can access only their own run-time report data.
- Remote monitor—This role can access reports and view run-time data related to its own exchange with the host trading partner.

See Section 5.3, "Adding Trading Partner Users," for how to assign roles.

The partner data you design, deploy, and manage with the Oracle B2B user interface is secured by its centralized storage in the Metadata Service (MDS) repository.

Other security features include:

- Transport protocol-based security for HTTP, FTP, and SMTP exchanges

- Digital envelopes and certificates

- Digital signatures for host and remote trading partners

- Integration with Credential Store Framework for storing all passwords and security credentials

- Secure HTTP (using Secure Socket Layer (SSL))

- Encrypted Key Store password for a host trading partner

> **Note:** Oracle B2B run time does not support the CLIENT-CERT authentication method. Therefore, B2B is not able to post to OAM-SSO protected URLs.

See the following for more information about security:

- *Oracle Fusion Middleware Security Guide*

1.4.1 Payload Obfuscation

Oracle B2B supports payload obfuscation before payloads are stored in the instance repository. The security infrastructure of Oracle Fusion Middleware is used to obfuscate, store, and retrieve the payloads, and ensure that payloads in wire messages, business messages, and application messages are visible to authorized users only. The encryption algorithm is not specifiable. Keys are stored in the Credential Store.

At run time, the payload is obfuscated before it is stored in the instance repository. When this payload is retrieved from the instance store during processing, it is automatically unobfuscated so that B2B engine processes it.

Similarly, in the outbound direction, if payload obfuscation is required, then the payload is obfuscated before it is stored in the instance repository. If exchange-level encryption is specified, then the payload is encrypted using the encryption scheme specified before it is put on the wire.

Payload obfuscation can be configured in Oracle Enterprise Manager Fusion Middleware Control. See *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite* and Appendix C, "Setting B2B Configuration Properties in Fusion Middleware Control," for more information.

When you enable payload obfuscation, consider the following:

- Large payloads, as defined in the **Large Payload Size** parameter on the **Configuration** tab, are not obfuscated because they are stored in a directory (file system) rather than the instance repository. Storing a large payload in the file system is a security risk.

- The obfuscated payload can be accessed in the Oracle B2B interface only by authorized users who have access to the document type. The payload is unobfuscated and displayed in the interface for these authorized users. Other users cannot access the document type at all. The users can be provisioned to access document types. See Section 1.4.2, "Restricting Access to Document Types," for information about document-type provisioning.

Obfuscation is available for payloads that use multibyte characters, and is available for non-Oracle databases.

If you migrate instance stores that contain obfuscated payloads, then you must ensure that you export the Credential Store Framework (CSF) as well, because the CSF has the key to unobfuscate those payloads (the same key is used for obfuscation and unobfuscation). If this is a new store, then no migration is required because the key is created (if not already present) the first time the payload is obfuscated.

A payload that was obfuscated and persisted in Oracle B2B is passed unobfuscated to other SOA components within a composite application, when using the Default or JMS integration types. Users viewing this unobfuscated payload in other SOA components are responsible for ensuring that the payload is obfuscated and persisted securely, and that users are authorized to view the payload.

1.4.2 Restricting Access to Document Types

Oracle B2B supports payload security by restricting access based on document type. The following user permissions for document-type access are available:

- Admin permission for all document types

  With this permission, the user can add, access, edit, and delete all document types. This user also has access to administrative functions such as import, export, and purge.

- Admin permission for specified document types

  With this permission, the user can access, edit, and delete the specified document types for which he has permission. The user is not allowed to access, edit, or delete the restricted document types. The user cannot add new document types or have access to any administrative functions such as import, export, and purge.

- Monitor permission for all document types

  With this permission, the user can access and view (but not edit or delete) all document types.

- Monitor permission for specified document types

  With this permission, the user can access and view (but not edit or delete) the specified document types. The user cannot access and view the restricted document types.

The default administrator user can restrict document-type access to other roles as follows:

- The host administrator can be granted access to all document types, in which case this user can restrict document-type access to other host or remote administrators.

- The host administrator can be granted access only to specified document types, in which case this user cannot restrict document-type access to other host or remote administrators.

- The remote administrator can be granted access to specified document types only, or all document types pertaining to the remote trading partner. In either case, the remote trading partner administrator cannot create document types in the system, or provision users for that particular remote trading partner. Users can only be provisioned by a host trading partner administrator user.

- The host monitor can be granted view-only access to all document types or to specified document types, but cannot restrict document-type access to other users.

- The remote monitor can be granted view-only access to all document types pertaining to the remote trading partner or to specified document types pertaining

to the remote trading partner, but cannot restrict document-type access to other users.

> **Note:** Admin users with access to all **Administration** tab functions lose admin privileges when permission for any or all document types is assigned, and the **Administration** tab is no longer available.

See Task 3, "Add Document Types That the User Has Permission to Access" in Section 5.3, "Adding Trading Partner Users," for how to specify document-type access in the Oracle B2B interface.

When access to specific document types is restricted, consider the following:

- New document definitions for a restricted document type cannot be added.

- No document types can be imported, exported, or purged.

- No document types can be modified on the **Partners** > **Documents** tab, as shown in Figure 1–1.

Figure 1–1 Accessing a Restricted Document Type from the Documents Tab
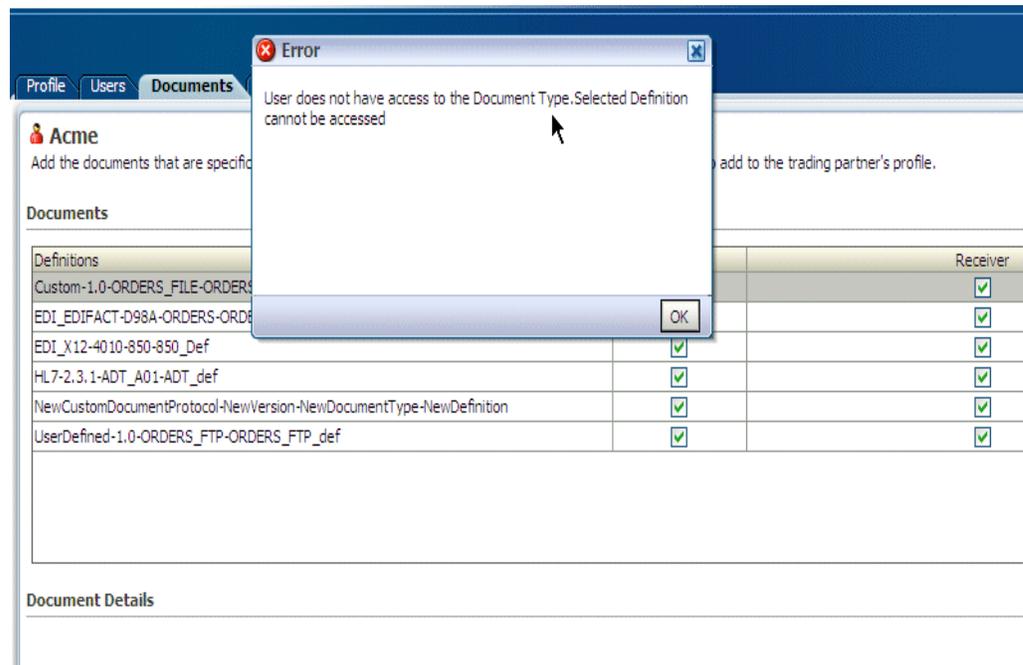
The image shows that the selected document type is restricted for the logged-in user. The error message "User does not have access to the document type. Selected definition cannot be accessed" appears.

****

- The restricted document types are listed, but details cannot be viewed or accessed, on the following tabs:

  – **Administration** > **Document** tab

  – **Reports** tabs

  – **Metrics** tabs

- Agreements that include document definitions for restricted document types cannot be modified or exported.

- In a SOA composite with a B2B binding component, restrictions on document types are *not* in effect. All document types are available to any user in the B2B Configuration Wizard of Oracle JDeveloper.

1.5 How Does Oracle B2B Fit into a SOA Implementation?

As a business-to-business gateway, Oracle B2B is used to extend business processes to trading partners. When Oracle B2B is used in a SOA composite application, you can model an end-to-end business process integration.

Oracle SOA Suite provides a complete set of service infrastructure components for designing, deploying, and managing composite applications. The multiple technology components of a composite application share common capabilities, including a single deployment and management model and tooling, end-to-end security, and unified metadata management. See *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for more information.

In a SOA implementation, Oracle B2B functions as a *binding component*, with network protocols and services that enable message sending and receiving:

- As a *service* (inbound), the SOA composite application receives messages from Oracle B2B

- As a *reference* (outbound), the SOA composite application passes a message to Oracle B2B, which in turn sends the message to partners.

In addition to messages, Oracle B2B can also send attachments and large payloads in a SOA implementation. See Appendix A, "Handling Large Payloads," for information about handling large payloads.

> **Note:** With the integration of Oracle B2B, Mediator, and BPEL components within Oracle SOA Suite, the XML Gateway Internal Delivery channels are not needed in Oracle B2B 11*g* to communicate with Oracle E-Business Suite. This can be achieved by using the Oracle Application Adapter available in Oracle SOA Suite.

1.6 Sending a Purchase Order: An Example of a SOA Implementation

The following example describes how the components of a SOA composite application are used to send a purchase order that originates from Oracle E-Business Suite, as shown in Figure 1–2.

Figure 1–2   An Outbound Purchase Order in a SOA Composite Application
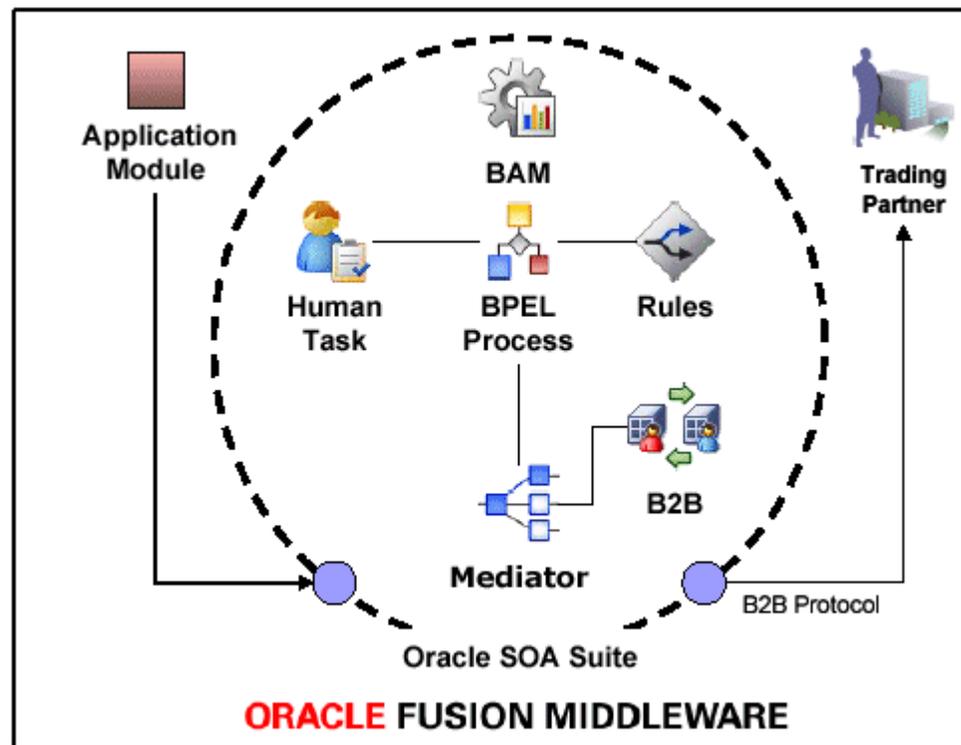
The image shows a graphical representation of Oracle Fusion Middleware. On the left, an application module connects to Oracle SOA Suite, which in turn connects to a trading partner using a B2B (business-to-business) protocol. Oracle SOA Suite consists of Business Activity Monitoring (including human tasks, BPEL processes, and rules), a mediator, and a B2B component.

****

The outbound purchase order (P. O.) is an XML document that participates in an end-to-end business process as follows:

1. An application, for example, Oracle E-Business Suite, initiates the P. O. process. The P. O. document uses the application-generated XML.

2. Oracle Mediator receives the P. O. from Oracle E-Business Suite. The P. O. is translated to canonical XML through XSLT Mapper, and is validated by using the schema obtained when the composite application was validated. Oracle Mediator routes the message to Oracle BPEL Process Manager.

3. Oracle BPEL Process Manager receives the P. O. from Oracle Mediator. Business processes such as human workflow, business rules, and error handling can apply before Oracle BPEL Process Manager sends the P. O. back to Oracle Mediator.

4. Oracle Mediator receives the P. O. from Oracle BPEL Process Manager. The P. O. is transformed from canonical XML to the target XML through XSLT Mapper and then routed to Oracle B2B.

5. Oracle B2B receives the P. O. from Mediator, translates the P. O. to EDI native format, for example, and manages the interaction with the trading partner.

6. Oracle Business Activity Monitoring (BAM) monitors the end-to-end process.

See the following for more information:

- Section 2.5, "Using Oracle B2B in the Oracle JDeveloper Environment," for how to include a B2B binding component in a SOA composite application

- *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for information about Oracle SOA Suite and SOA composite applications

1.7 Oracle B2B Samples and Cookbooks

The Oracle B2B samples guide you through the steps to create guideline files, design Oracle B2B transactions, deploy and monitor trading partner agreements, and create and deploy SOA composite applications. The composite applications include an Oracle B2B binding component and use the document definitions that you create in Oracle B2B.

Samples are available for the following document types:

- 1Sync

- Custom XML

- EDI EDIFACT

- EDI X12

- HL7

- RosettaNet

These end-to-end samples are based on a scenario in which two trading partners, Acme and GlobalChips, participate in a transaction. Acme is the initiator (the buyer, in the case of a purchase order scenario) and GlobalChips is the responder (the seller in a purchase order scenario). In the HL7 sample, Acme (initiator) sends an ADT_A01 admit patient message and receives an ACK_A01 acknowledgment from GlobalChips.

The samples include instructions and sample document definition files for you to create all the documents, agreements, and SOA composites you need to run the samples. The completed SOA composite application is also provided for each sample.

You can download the samples from the Oracle SOA Suite samples.

The Oracle B2B cookbooks provide detailed information on the implementation of ebXML, EDI (EDIFACT and X12), and HL7 in Oracle B2B. The cookbooks include configuration steps, performance best practices, discussions of typical errors, and FAQs. To download the cookbooks, go to

http://www.oracle.com/technetwork/middleware/b2b-integrations

See Section 2.1, "What You Need to Get Started with Oracle B2B," for the components required to use the samples and cookbooks.

1.8 Administering Oracle B2B in the Oracle Fusion Middleware Environment

The following components provide monitoring, configuration, and performance tuning capabilities for Oracle B2B:

- SOA Server—Set memory arguments to optimize B2B performance.

  See "Oracle B2B Performance Tuning" in *Oracle Fusion Middleware Performance Guide* for more information.

- Oracle WebLogic Server Administration Console—Administer settings for performance tuning.

    See "Oracle B2B Performance Tuning" in *Oracle Fusion Middleware Performance Guide* for more information.

- Oracle Enterprise Manager Fusion Middleware Control—Set Oracle B2B Server properties to enable Enterprise Manager metrics and monitor the Oracle B2B Infrastructure.

    See the following for more information:

    – Section C.1, "Properties To Set in Fusion Middleware Control"

    – "Configuring Oracle B2B" and "Monitoring Oracle B2B" in *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite*

Within the Oracle B2B interface, use the following for monitoring and configuration:

- **Administration** > **Configuration** tab

    See Chapter 17, "Configuring B2B System Parameters."

- **Administration** > **Reports** tab

    See Chapter 18, "Creating Reports."

- **Administration** > **Metrics** link

    See Chapter 19, "Using B2B Metrics."

2

Getting Started with Oracle B2B

This chapter describes the Web-based interface provided by Oracle B2B for creating B2B transactions. It also discusses the process flow of creating an Oracle B2B transaction and how to use Oracle B2B in the Oracle JDeveloper environment.

This chapter contains the following topics:

- Section 2.1, "What You Need to Get Started with Oracle B2B"
- Section 2.2, "Logging in to Oracle B2B"
- Section 2.3, "Using the Oracle B2B Interface"
- Section 2.4, "Creating a B2B Transaction: An Overview of the Process Flow"
- Section 2.5, "Using Oracle B2B in the Oracle JDeveloper Environment"
- Section 2.6, "What You May Need To Know About Using Oracle B2B"

2.1 What You Need to Get Started with Oracle B2B

In addition to installing Oracle SOA Suite, which includes Oracle B2B, you will need to install:

- Oracle Document Editor
- Oracle JDeveloper

Use the standards-based templates of Oracle Document Editor to create guideline files. Then, using Oracle B2B, you create and deploy the transaction as part of a B2B agreement. To include the B2B transaction in a SOA composite application, use Oracle JDeveloper, as shown in Figure 2–1.

Figure 2–1   Oracle JDeveloper: A SOA Composite Application with a B2B Binding Component
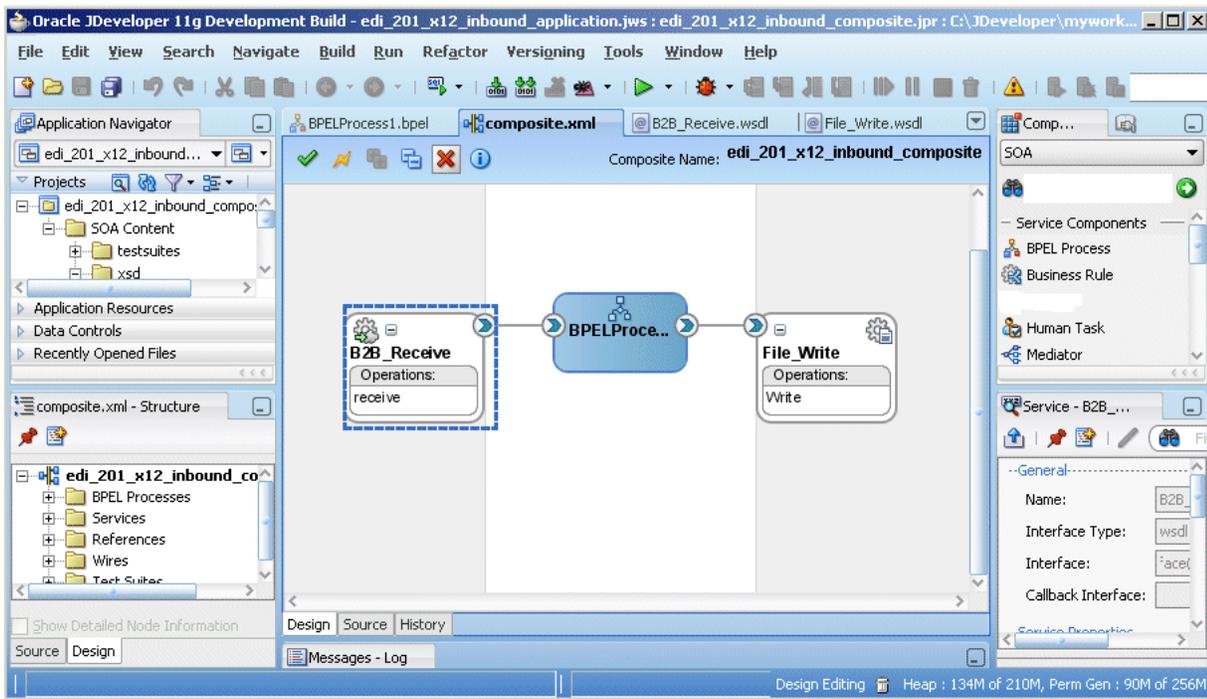
The image shows a composite.xml file in the visual editing window of Oracle JDeveloper. Icons for the B2B_Receive activity, the BPEL process, and the File_Write file adapter activity are shown connected (B2B to BPEL, BPEL to File).

****

See the following for more information:

- *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite* for information on installing Oracle B2B as part of Oracle SOA Suite

- Chapter 3, "Creating Guideline Files"

- Oracle B2B Document Editor **Help** menu

- Oracle JDeveloper **Help** menu

2.2 Logging in to Oracle B2B

These instructions assume that you have installed Oracle SOA Suite, which includes Oracle B2B. See *Oracle Fusion Middleware Installation Guide for Oracle SOA Suite* for more information.

Use a supported Web browser:

- Microsoft Internet Explorer 7.x and 8.x

- Mozilla Firefox 2.x and 3.x

- Apple Safari 4.x

To log in to Oracle B2B:

1. Open a supported Web browser and go to:

   ```
   http://host_name:port_number/b2bconsole
   ```

where:

- *host_name* is the name of the host on which Oracle SOA Suite is installed. (In a cluster environment, the *host_name* can be the front end load balancer)

- *port_number* is the port number used by the Managed Server to listen for regular HTTP (non-SSL) connections. (In a cluster environment, the *port_number* can be the router port.)

  See Section 2.2.1, "Finding Port Information," for more information.

- /b2bconsole (or /b2b) accesses the B2B interface (/b2b is redirected to /b2bconsole).

  See Section 2.2.2, "Accessing Oracle B2B Through Single Sign-On (SSO)," for information on protecting the Oracle B2B user interface page by adding /b2bconsole and /b2b settings to the mod_wl_ohs.conf file in Oracle HTTP Server.

> **Note:** To access Oracle B2B when SAML is enabled or in Windows Native Authentication Environments, use either of the following protected servlet URLs for automatic authentication:
>
> ```
> http://host_name:port_number/b2b/ssologin
> http://host_name:port_number/b2bconsole/ssologin
> ```

---

2. On the log-in page, enter the following:

For This Field...	Do...
Username	Enter the default administrator user name.
Password	Use the administrator password from your Oracle Fusion Middleware 11*g* installation.

3. Click **Login**.

2.2.1 Finding Port Information

You can find port number information in the following ways:

- From Oracle WebLogic Server Administration Console

  1. Log in to the console.

  2. In the Domain Structure pane, shown in Figure 2–2, expand **Environment** and click **Servers**.

Figure 2–2   Domain Structure Nodes in Oracle WebLogic Server Administration Console
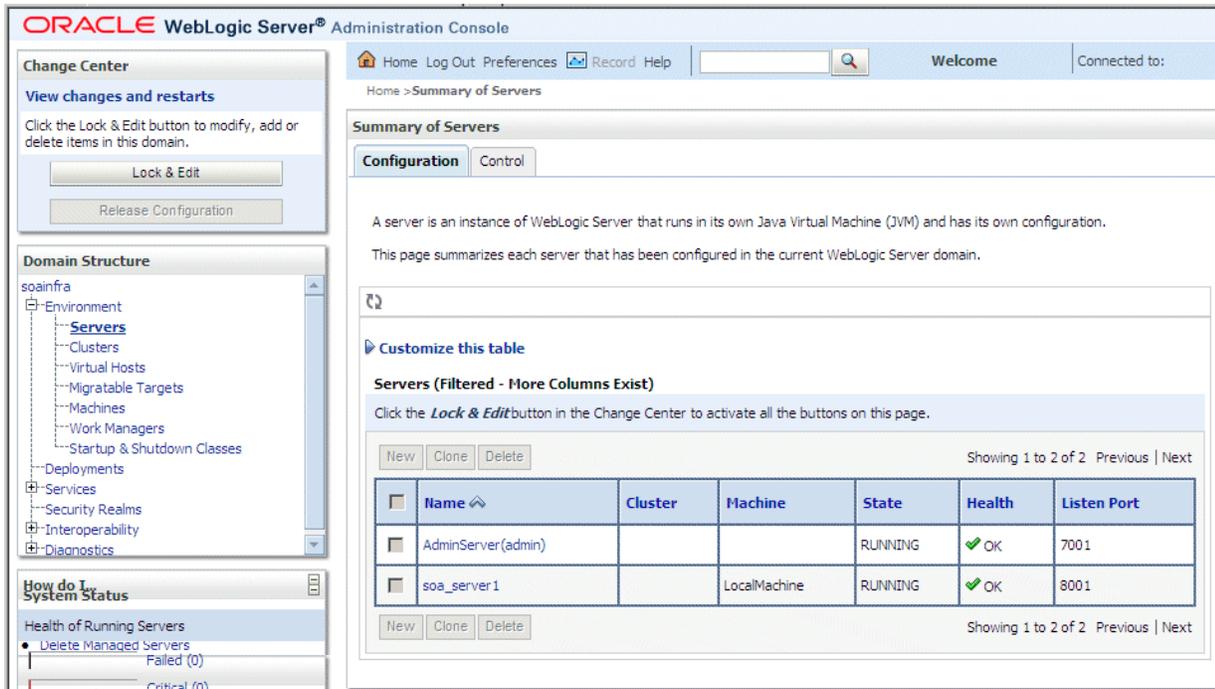
The image shows the Summary of Servers configuration tab, which includes a table showing a summary of each server configured in the current WebLogic Server domain. In the table, each server occupies a single row with the following columns: Name, Cluster, Machine, State, Health, and Listen Port. To the left of each row is a check box. To the left, above and below the table, are the New, Clone, and Delete buttons. To the right, above and below the table, are links to Previous and Next pages of information.

****

   3.   Note the **Listen Port** column for your server.

■   Or from
MW_HOME/user_projects/domains/*your_domain_name*/config/config.
xml

```
<server>
  <name>soa_server1</name>
  <ssl>
    <name>soa_server1</name>
    <listen-port>8002</listen-port>
  </ssl>
  <machine>LocalMachine</machine>
  <listen-port>8001</listen-port>
  <listen-address/>
</server>
```

2.2.2 Accessing Oracle B2B Through Single Sign-On (SSO)

To log in, log out, and relog in to Oracle B2B using SSO in Oracle Identity Management, the /b2bconsole location must be added to the mod_wl_ohs.conf file in Oracle HTTP Server as follows:

```
<Location /b2bconsole>
   SetHandler weblogic-handler
#     PathTrim /weblogic
```

```
      ErrorPage  http:/WEBLOGIC_HOME:WEBLOGIC_PORT/
</Location>
```

This is in addition to the setting required for the /b2b location:

```
<Location /b2b>
   SetHandler weblogic-handler
#     PathTrim /weblogic
   ErrorPage  http:/WEBLOGIC_HOME:WEBLOGIC_PORT/
</Location>
```

2.2.3 Enabling the weblogic User for Logging in to Oracle B2B

For the weblogic user in Oracle Internet Directory (OID) to log in to Oracle B2B as an administrator and search for users, the OID Authenticator must have an Administrators group, and the weblogic user must be a member of that group.

To enable the weblogic user:

1. Create a weblogic user in OID using the LDAP browser. The users.ldif file is imported to OID as follows:

```
dn: cn=weblogic,cn=Users,dc=us,dc=oracle,dc=com
objectclass: inetorgperson
objectclass: organizationalPerson
objectclass: person
objectclass: orcluser
objectclass: orcluserV2
objectclass: top
sn: weblogic
userpassword: welcome1
uid: weblogic
```

2. Create an Administrators group in OID and assign the weblogic user to it. The groups.ldif file is imported to OID as follows:

```
dn: cn=Administrators,cn=Groups,dc=us,dc=oracle,dc=com
objectclass: groupOfUniqueNames
objectclass: orclGroup
objectclass: top
owner: cn=orcladmin,cn=Users,dc=us,dc=oracle,dc=com
uniquemember: cn=weblogic,cn=Users,dc=us,dc=oracle,dc=com
```

2.3 Using the Oracle B2B Interface

B2B activities are grouped as follows:

- Administration

- Partners

- Reports

- Metrics

2.3.1 Administration

Use the tabs of the **Administration** page, shown in Figure 2–3, to manage importing and exporting, document protocols, deployments, types, batching, callouts, purging, listening channels, and B2B configuration.

Figure 2–3   Administration Activities
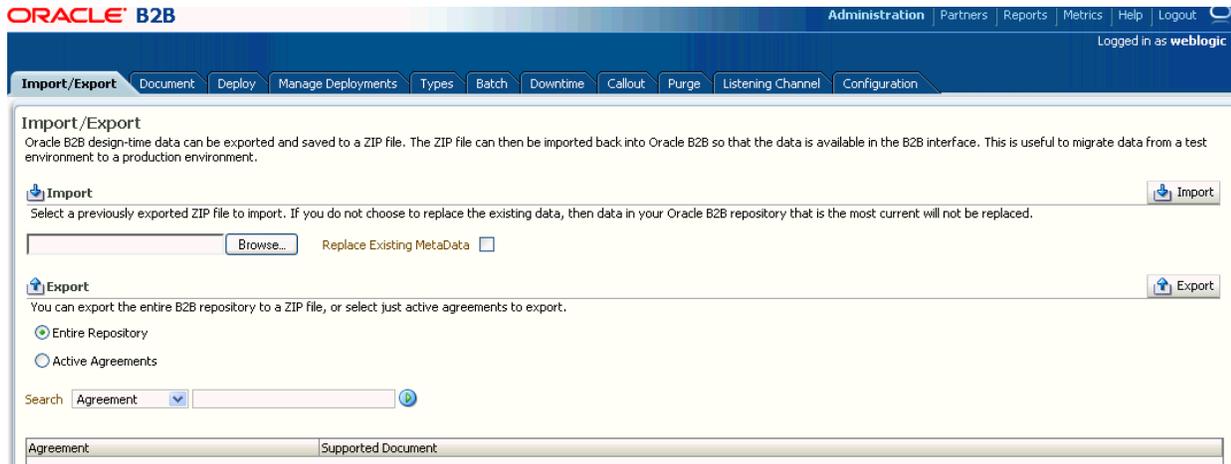
The image shows the following tabs of the Administration link: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Import/Export tab is selected.

****

See Part III, "Oracle B2B Administration" for more information.

2.3.2 Partners

Use the tabs of the **Partners** page, shown in Figure 2–4, to create and update trading partner information, create and update agreement information, add user information, associate documents with trading partners, set up channels, and configure the key store.

Figure 2–4   Partner Activities

The image shows the following tabs of the Partners page: Profile, Users, Documents, and Channels. The Profile tab is selected. The first area of the tab lists the identifiers as a series of type-value pairs. The second area shows the contact information, also as a series of type-value pairs. In both areas, the type field is a drop-down list. There are plus and minus icons above both areas to the right. The third area lists the parameters. The fourth area shows the key store, including the following fields: Password, Confirm Password, and Location. There are Save and Export buttons at the top of the tab on the right side.

****

See Part II, "Oracle B2B Process Flow" for more information.

2.3.3 Reports

Use the tabs of the **Reports** page, shown in Figure 2–5, to create and view reports about the instance (run-time) data.

Figure 2–5   Reports

The image shows the following tabs of the Reports page: Business Message, Wire Message, Application Message, Error, and Conversation. The Business Message tab is selected. The first area of the tab shows the following Search fields: Sender, Receiver, Agreement, Send Time Stamp, Receive Time Stamp, State, and Message Id. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Details, State, Document Type, Agreement, Sender, Receiver, Receive Time Stamp, and Send Time Stamp.

****

See Chapter 18, "Creating Reports," for more information.

2.3.4 Metrics

Use the tabs of the **Metrics** page, shown in Figure 2–6, to see information about deployed agreements, such as lists of the active document types and trading partners, and run-time status, such as error messages and message counts.

Figure 2–6 Metrics
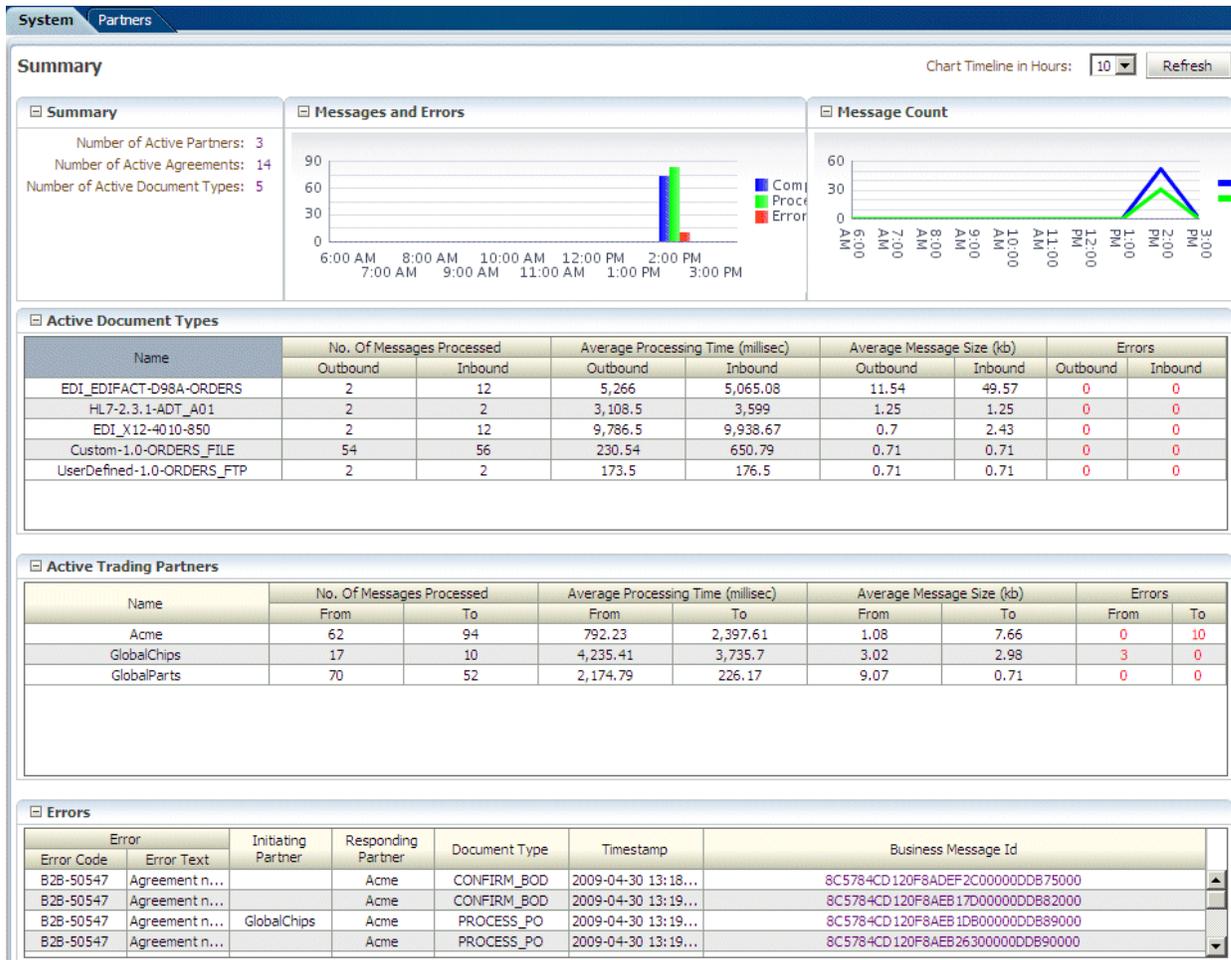
The image shows the following tabs of the Metrics page: System and Partner. The System tab is selected. The first area of the tab shows Summary information, including a high-level Summary, Messages and errors, and the Message count. The second and third areas of the tab show the Active Document Types and the Active Trading Partners, both displayed as tables. The fourth area shows the errors, also displayed as a table. The Refresh button is at the top of the tab on the right.

****

See Chapter 19, "Using B2B Metrics," for more information.

2.4 Creating a B2B Transaction: An Overview of the Process Flow

Figure 2–7 shows the B2B process flow, which starts with creating B2B guideline files in Oracle Document Editor and continues with using the Oracle B2B interface to create document definitions, configure trading partners, and create and deploy agreements.

Figure 2–7   Oracle B2B Process Flow
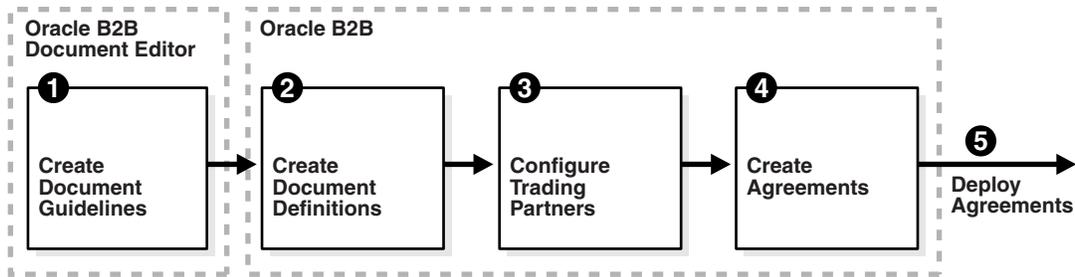
The image shows the steps of the B2B process flow: step 1, create document guidelines; step 2, create document definitions; step 3, configure trading partners; step 4, create agreements; step 5, deploy agreements. You do step 1 using Oracle B2B Document Editor and steps 2 through 5 using the Oracle B2B interface.

****

Step 1: Create guideline files (ECS and optional XSD files) in Oracle Document Editor

Using Oracle B2B Document Editor, shown in Figure 2–8, create transaction documents based on templates for hundreds of industry-standard protocols. The Oracle Document Editor is required only when translation is needed. For XML documents, the editor is not used.

Figure 2–8   Creating a Document in Oracle B2B Document Editor
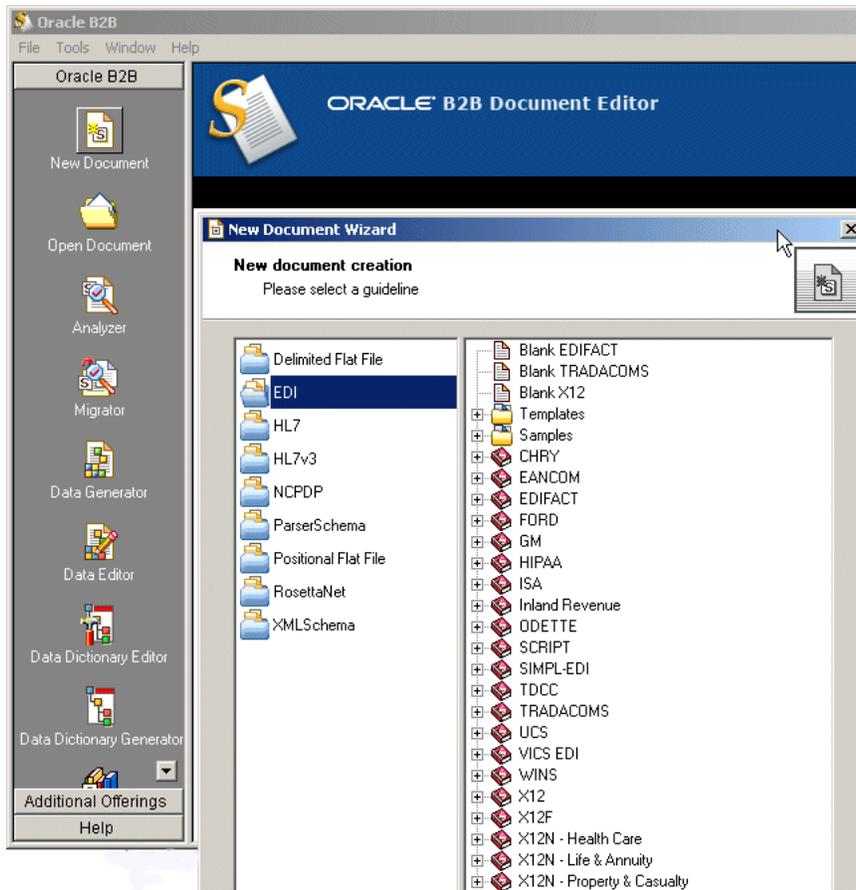
The image shows Oracle B2B Document Editor. The list of new document templates for the EDI protocol (selected) includes Blank EDIFACT, Blank TRADACOMS, Blank X12, Templates, Samples, CHRY, EANCOM, EDIFACT, FORD, GM, HIPAA, ISA, Inland Revenue, ODETTE, SCRIPT, SIMPL-EDI, TDCC, TRADACOMS, UCS, VICS EDI, WINS, X12, X12F, X12N - Health Care, X12N - Life & Annuity, X12N - Property & Casualty.

****

For information on Oracle Document Editor, see the following:

- Chapter 3, "Creating Guideline Files"

- The **Help** menu of Oracle Document Editor, shown in Figure 2–9.

Figure 2–9   Oracle Document Editor Help
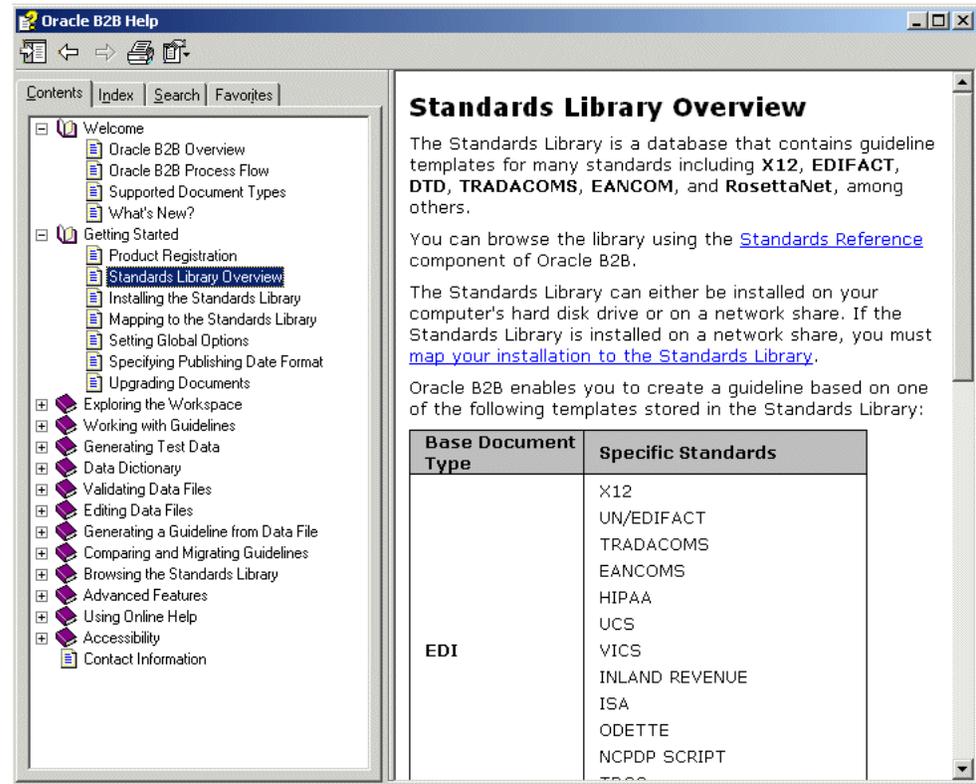
The image shows an example of help information in Oracle B2B Document Editor. The Standards Library Overview topic is shown.

****

Step 2: Create document definitions

Using the **Administration > Document** tab of Oracle B2B, shown in Figure 2–10, select from a list of document protocols, and then provide a document protocol version name, a document type name, and a document definition name. (For a Custom document, rather than selecting from the list of document protocols, you add a custom protocol name to the list in the **Document Protocols** folder.)

Figure 2–10   Creating a Document Definition
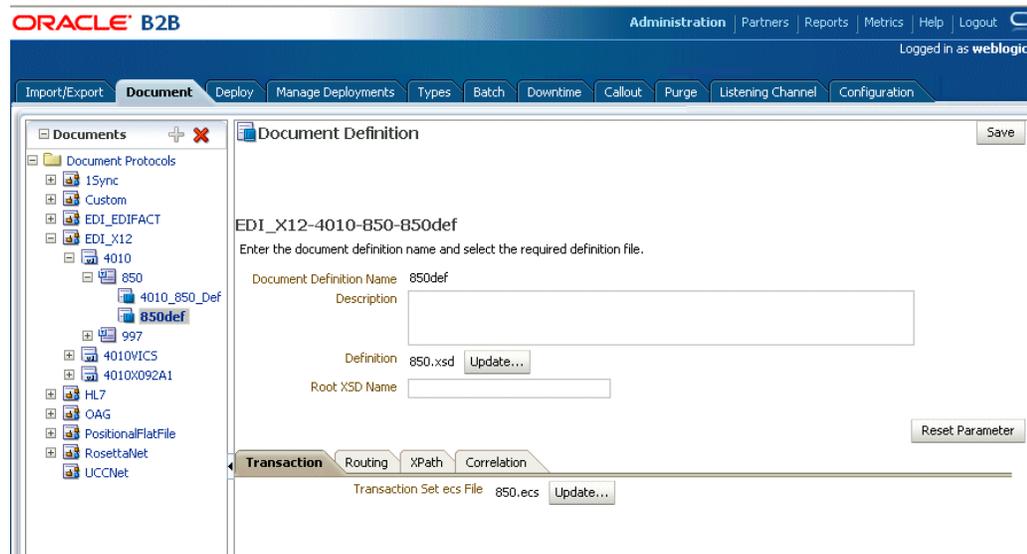
The image shows the Document tab on the Administration page in Oracle B2B. The document protocols—Custom, EDI_EDIFAC, EDI_X12, HL7, OAG, and RosettaNet—are shown. A completed document definition, EDI X12-4010-850-850def, is also shown.

****

After selecting the ECS and optional XSD files you created in Step 1, you have created the document definition.

For more information, see Chapter 4, "Creating Document Definitions."

Step 3: Configure trading partners

Using the tabs of the **Partners** page of Oracle B2B, shown in Figure 2–11, add or update trading partner names, add identifiers and optional contact information, view parameters, add documents and delivery channels, and add key store information.

Figure 2–11   Configuring Trading Partners
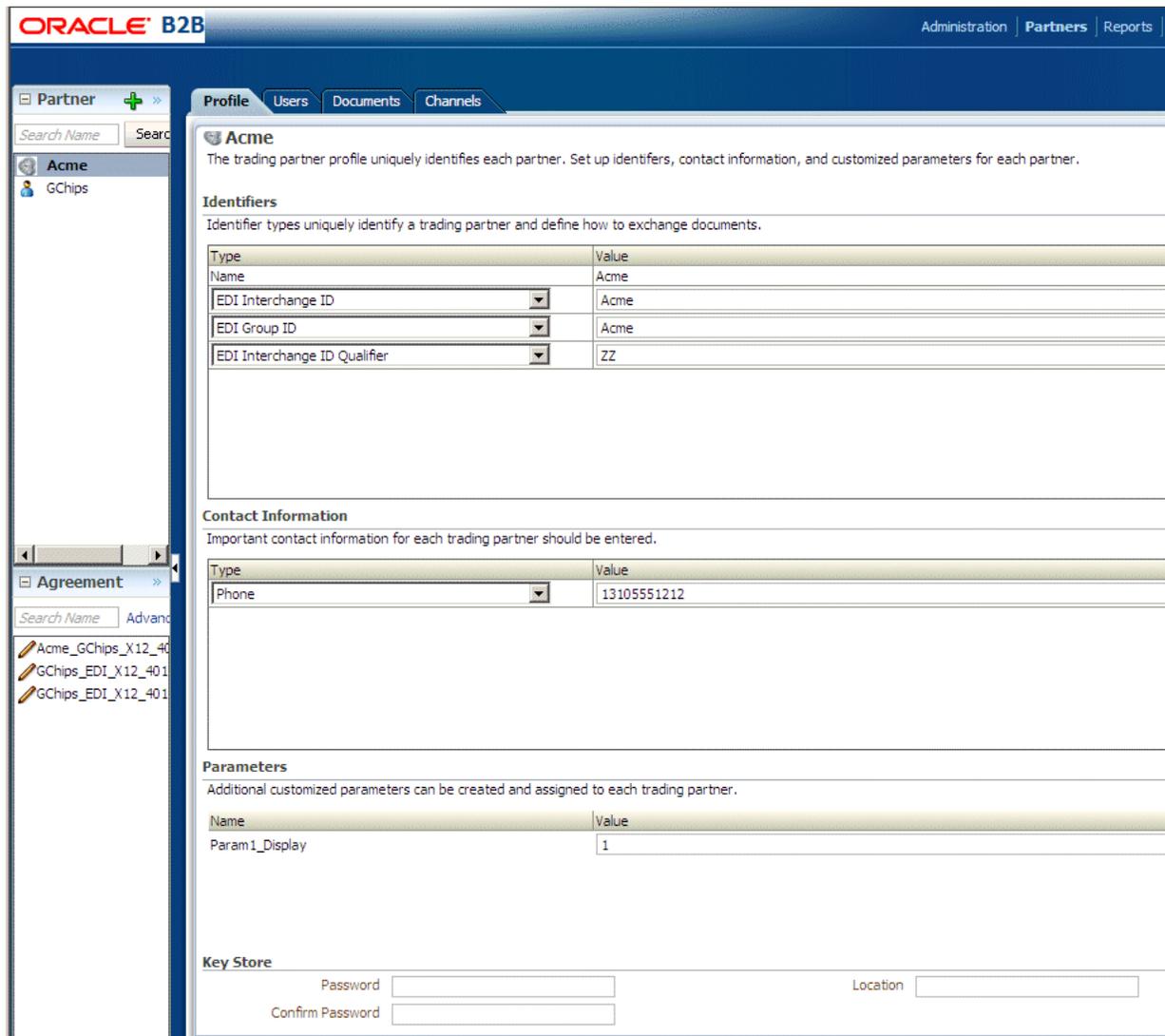
The image shows the Partners tab and trading partner Profile subtab in Oracle B2B. The trading partners—Acme and Gchips—are shown. Identifiers, contacts, and parameters for Acme are also shown, as are fields for the key store location and password.

****

For more information, see Chapter 5, "Configuring Trading Partners."

Step 4: Create agreements

Using the **Partners > Agreement** tab of Oracle B2B, shown in Figure 2–12, create an agreement that specifies the trading partners involved and associates the document definitions, channels, and identifiers with the agreement.

Figure 2–12   Creating a Trading Partner Agreement
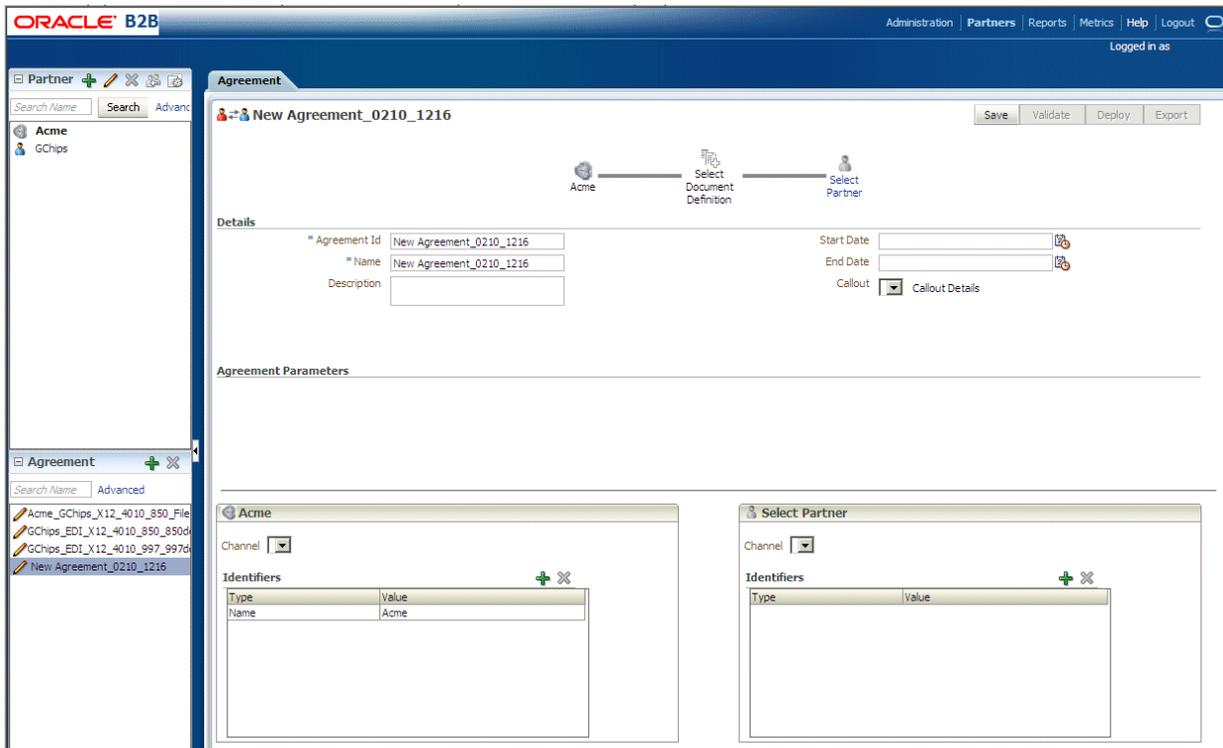
The image shows the Agreement configuration page in Oracle B2B. The first area of the page contains a graphic representation of the agreement. The second area of the tab shows the Agreement Details, including the Agreement Id, Name, Description, Start Date, End Date, and Callout. The third area shows the Agreement Parameters. The bottom of the tab shows two panes displaying the Channel and Identifiers for the trading partners. At the top right of the tab are the Save, Validate, Deploy, and Export buttons.

****

For more information, see Chapter 6, "Creating and Deploying Trading Partner Agreements."

Step 5: Deploy agreements

Using the **Administration > Deploy** tab of Oracle B2B, shown in Figure 2–13, search for and deploy agreements.

Figure 2–13   Searching for and Deploying Agreements

The image shows the following tabs of the Administration page: Document, Deploy,
Manage Deployments, Types, Import/Export, Schedule Batch, Callout, Purge,
Listening Channel, Configuration. The Deploy tab is selected. The first area of the tab
shows the following Search fields: Name, Trading Partner, Identification Value,
Identification Type, Document Protocol Name, Document Protocol Version, Document
Type, and Document Definition. At the top of the area is the Match radio button with
the following options: All or Any. The second area of the tab shows the Agreement
search results in a table with the following columns: Agreement, Initiating Partner,
Responding Partner, Supported Document, Channel (From and To), State, and Last
Deployed. At the top and right of the tab is the Deploy button.

****

2.5 Using Oracle B2B in the Oracle JDeveloper Environment

By using an Oracle B2B binding component in a SOA composite application, you can
create an end-to-end business process, such as sending a purchase order generated by
a back-end application to a trading partner. Binding components establish the
connection between a SOA composite application and the external world.

The B2B Configuration Wizard in Oracle JDeveloper enables you to add B2B binding
components to a SOA composite application as follows:

- B2B is used as a *service* (inbound) to receive messages from trading partners and
  deliver them to SOA composite applications. Oracle B2B is the entry point to the
  SOA composite application.

- B2B is used as a *reference* (outbound) to send messages from the SOA composite
  application to partners.

As you follow the steps in the B2B Configuration Wizard, you are prompted to select a document definition created in Oracle B2B. Or, you can launch Oracle B2B from the wizard to create a document definition. This is the payload, or message, that you are receiving from trading partners or sending to trading partners.

> **Note:** In the B2B Configuration Wizard, if SSL is enabled in the middleware (the B2B Web service), then the SSL port is detected by the wizard and the document definitions are retrieved using the SSL connection.

2.5.1 How To Use B2B Binding Components in a SOA Composite Application

To create a SOA composite application with a B2B binding component, do the following:

- Task 1, "Create a SOA Application and Project"
- Task 2, "Add Service Components"
- Task 3, "Add a B2B Binding Component"

See the following in *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for more information on creating SOA composite applications:

- "Adding Wires"
- "Adding Security"
- "Deploying a SOA Composite Application"
- "Managing and Testing a SOA Composite Application"

Task 1 Create a SOA Application and Project

1. Start Oracle JDeveloper Studio Edition Version 11.1.1.2.0.

2. If Oracle JDeveloper is running for the first time, specify the location for the Java JDK.

3. Create a new SOA composite application, as described in Table 2–1.

Table 2–1   SOA Composite Application Creation

If Oracle JDeveloper...	Then...
Has no applications For example, you are opening Oracle JDeveloper for the first time.	In the Application Navigator in the upper left, click **New Application**.
Has existing applications	From the **File** main menu or the **Application** menu: 1. Select **New** > **Applications**. The New Gallery opens, where you can select different application components to create. 2. In the **Categories** tree, under the **General** node, select **Applications**. In the **Items** pane, select **SOA Application** and click **OK**.

The Create SOA Application wizard starts.

4. On the Name your application page, you can optionally change the name and location for your web project. If this is your first application, from **Application**

Template, select **SOA Application**. Accept the defaults for the package prefix, and click **Next**.

> **Notes:**
>
> - Do *not* create an application name with spaces.
>
> - Do *not* create applications and projects in directory paths that have spaces (for example, `c:\Program Files`).
>
> - In a UNIX operating system, it is highly recommended to enable Unicode support by setting the `LANG` and `LC_All` environment variables to a locale with the UTF-8 character set. This action enables the operating system to process any character in Unicode. SOA technologies are based on Unicode. If the operating system is configured to use non-UTF-8 encoding, SOA components may function in an unexpected way. For example, a non-ASCII file name can make the file inaccessible and cause an error. Oracle does not support problems caused by operating system constraints.
>
>   In a design-time environment, if you are using Oracle JDeveloper, select **Tools** > **Preferences** > **Environment** > **Encoding** > **UTF-8** to enable Unicode support. This setting is also applicable for run-time environments.

5. On the Name your project page, you can optionally change the name and location for your SOA project. By default, Oracle JDeveloper adds the SOA project technology, the `composite.xml` that generates, and the necessary libraries to your model project. Click **Next**.

> **Note:** Composite and component names cannot exceed 500 characters.

A project deployed to the same infrastructure *must* have a unique name across SOA composite applications. The uniqueness of a composite is determined by its project name. For example, do *not* perform the actions described in Table 2–2. During deployment, the second deployed project (composite) overwrites the first deployed project (composite).

Table 2–2   Restrictions on Naming a SOA Project—Caution: Doing This Overwrites the First Deployed Project

Create an Application Named...	With a SOA Project Named...
Application1	Project1
Application2	Project1

The Project SOA Settings page of the Create SOA Application wizard appears.

6. In the Configure SOA Settings page, click **Empty Composite**, and click **Finish**.

7. Select **Save All** from the **File** main menu.

See *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for more information about creating a SOA application and project.

Task 2 Add Service Components

Service components implement the business logic or processing rules of your application.

1. From the Component Palette, select **SOA**.

2. From the **Service Components** list, drag a component into the designer.

   Figure 2–14 shows a BPEL process being added to the designer.

Figure 2–14 Adding BPEL Process to Composite
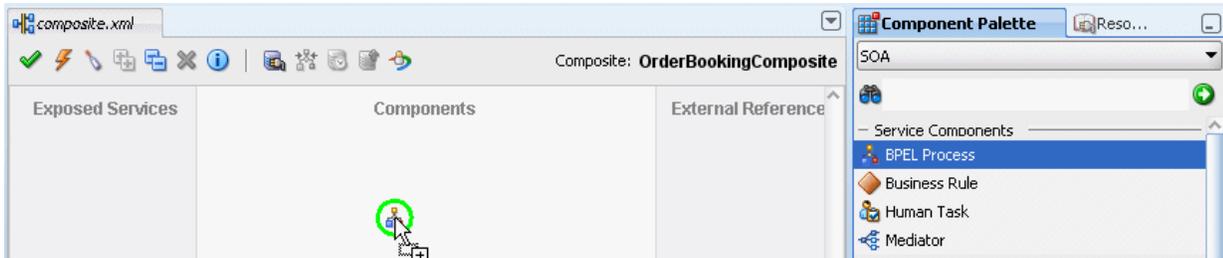
This image shows the Create BPEL Process dialog. The fields of this dialog are described in the table below this image.

****

A specific dialog for the selected service components is displayed. Table 2–3 describes the available editors.

Table 2–3 Starting Service Component Editors

Dragging This Service Component...	Invokes The...
BPEL Process	Create BPEL Process dialog to create a BPEL process that integrates a series of business activities and services into an end-to-end process flow.
Business Rule	Create Business Rules dialog to create a business decision based on rules.
Human Task	Create Human Task dialog to create a workflow that describes the tasks for users or groups to perform as part of an end-to-end business process flow.
Mediator	Create Mediator dialog to define services that perform message and event routing, filtering, and transformations.

3. Configure the settings for a service component. For help with a service component dialog, click **Help** or press **F1**. Click **Finish**.

4. Click **OK**.

5. Select **Save All** from the **File** main menu.

See *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for more information about adding service components.

Task 3 Add a B2B Binding Component

Add a service or a reference binding component.

1. From the Component Palette, select **SOA**.

2. Drag **B2B** to the **Exposed Services** or the **External References** swim lane.

- Select **Exposed Services** for receiving inbound messages.

- Select **External References** for sending outbound messages.

3. On the B2B Configuration Wizard Welcome page, click **Next**.

4. On the Service Name page, provide a name for the B2B service and click **Next**.

5. On the B2B Integration Type page, select an integration type, as described in Table 2–4.

Table 2–4   Selecting an Integration Type

Type	Description
Default	A B2B WSDL is generated for the SOA composite to communicate with Oracle B2B directly.
AQ	An AQ Adapter WSDL and JCA file are generated for the SOA composite to communicate with Oracle B2B through AQ queues.
JMS	A JMS Adapter WSDL and JCA file are generated for the SOA composite to communicate with Oracle B2B through JMS queues.

- If you select **Default**, complete steps 6 through 10.

- If you select **AQ**, complete steps 6 through 9 and 11 through 14

- If you select **JMS**, complete steps 6 through 9 and 15 through 18

6. On the Application Server Connection page, do one of the following:

- From the **AppServer Connection** list, select an application server connection and click **Next**.

- Or, click **New** to create an application server connection. Follow the Create Application Server Connection Wizard.

When the application server connection is established, the following information is displayed: the user name created for the application server connection, the host name for the server instance, and the SOA Server name. The SOA servers configured and running in Weblogic are displayed when you select an application server connection. After you select a SOA server, the SSL or HTTP port is retrieved and the B2B web service URL is generated for retrieving document definitions.

You can also click **Test B2B** to verify that you can connect to your Oracle B2B installation.

7. On the Operation page, select **Send** or **Receive**, as described in Table 2–5.

Table 2–5   Selecting a Send or Receive Operation

Operation	Description
Send	For outbound messages
Receive	For inbound messages

8. On the Document Definition Handling page, select the option on the **Basic** tab or one of the options on the **Advanced** tab, as described in Table 2–6.

Table 2–6   Selecting Document Definition Handling Options

Option	Description
Basic Tab	-
Import Schema from B2B	Imports the schema from Oracle B2B (the same option as on the **Advanced** tab)
Advanced Tab	-
Import Schema from B2B	Imports the schema from Oracle B2B (the same option as on the **Basic** tab)
Refer to Schema in B2B Repository	Selects a metadata service (MDS) connection. Or use this option to create a new one. If you create a new MDS connection, use the MDS Connection Wizard to create a connection. This connection is needed to access the B2B repository. When you select a document definition, a URL is generated to link to the MDS. The selected application server connection refers to a specific B2B instance. The MDS connection used by the specific B2B instance must match the selected MDS connection to avoid inconsistent document definitions. When referring to schema in a B2B Repository, an MDS connection is required only for referring to a schema in a remote MDS, but not if the schema is referred to within the local shared MDS repository. For referring to B2B schemas `b2bException` and `b2bAck` in the local shared MDS, you do not need to select an MDS connection. You can select the MDS option and click **Next** to navigate to the Document Definition page where you can select `b2bException` or `b2bAck`, or other B2B schemas that exist in the local shared MDS.
Browse Resource Schema	Browse for a schema using the SOA Resource Browser. Selecting this option opens the Type Chooser. Select a type and return to the Document Definition Handling page.
Opaque	Handles any type of data (for example, positional flat file) when the content is passed through in base-64 encoding. No schema is specified.
anyType	Handles any type of XML data. No schema is specified. This is valid and available only for the 'Default' integration type and not for AQ and JMS.

You can also select **Attachment Support** for the **Default** integration type so that the B2B WSDL file includes a message part for the attachment.

9. On the Document Definition page, expand the tree to select a document definition. If you select a document definition with multiple root elements, then select a root element to use and click **OK**.

Table 2–7 describes other options on the page.

Table 2–7   Document Definition Page Options

Option	Description
Search	Enter a definition name. Partial strings are matched if you type the beginning of the definition name. Partial strings with wildcards cannot be used.
Refresh	Retrieves the document definition list from the B2B server. Refresh after a search to see all document definitions.

Table 2–7   (Cont.) Document Definition Page Options

Option	Description
B2B Configuration	Opens a browser to Oracle B2B, using the connection specified on the Application Server Connection page. In Oracle B2B, you can create a document definition, include it in an agreement, and deploy the agreement. Then return to this dialog, click **Refresh**, and select the new document definition.
Use Routing ID	Selects all document definitions that use a document routing ID. See Section 8.10, "Using Document Routing IDs," for more information.

If you selected the

- **Default** integration type, go to Step 10.

- **AQ** integration type, go to Step 11.

- **JMS** integration type, go to Step 15.

10. On the Finish page, click **Finish**.

11. On the Service Connection page, do one of the following:

- From the **Connection** list, select a database connection and click **Next**.

- Or, click **New** to create an application server connection. Follow the Create Application Server Connection Wizard.

Table 2–8 describes the information displayed when the database connection is established.

Table 2–8   Service Connection

Parameter	Description
User Name	The user name created for the database connection.
Driver	The JDBC driver is provided.
Connect String	The JDBC connection string is provided.
JNDI Name	Use the default Java Naming and Directory Interface (JNDI) name or specify a custom name. This connection enables you to configure the adapter during design time and to connect to the database server during run time.
Data Source	Enter the JNDI name that is used to look up the data source in data-sources.xml. If you are using data-sources.xml to get the connection, then this name is required.
XA Data Source	Select this option if the data source name is an XA data source. An XA data source can participate in an XA global transaction that can span multiple resources. In this transaction, the application server acts as the coordinating transaction manager with multiple databases (or other resources such as JMS), each of which is involved in a single transaction. If selected, the adapter becomes part of the XA transaction. Otherwise, it is a local transaction.

12. On the Queue Name page, select a database schema and a queue name.

Only queues for B2B of the type `IP_MESSAGE_TYPE` are available.

13. On the Queue Parameters page, do the following:

- For an enqueue operation, enter a recipient name or a list of recipients separated by commas. If you do not enter a recipient, then the message is sent to all subscribers of the queue. This field can be overridden on a per message basis by setting the RecipientList field in the outbound header. The default value is **b2buser**.

- For a dequeue operation, enter the following:

  – Consumer: The name of the agent subscribing to the queue. This field is required and is limited to 30 characters. The default value is **b2buser**.

  – Message Selector Rule: Optional filtering logic for messages to dequeue based on the message properties or message content (for example, priority < 5 or tab.user_data.amount > 10000). If a rule is supplied, then an agent using the consumer name and the message selector rule are created in the queue. The consumer name must be a new agent name, because the adapter does not change the message selector rule of a previously created agent. No validation is performed on the logic you enter.

  – Dequeue Condition: A Boolean expression similar to the WHERE clause of a SQL query. This expression can include conditions on message properties, user data properties (object payloads only), and PL/SQL or SQL functions. If more than one message satisfies the dequeue condition, then the order of dequeuing is indeterminate, and the sort order of the queue is not honored.

14. Click **Finish**.

15. On the JMS Provider page, select one of the following:

    - Oracle Enterprise Messaging Service (OEMS): **Oracle WebLogic JMS** or **Oracle Advanced Queuing**

      OEMS is built on JMS and the J2EE Connector Architecture (JCA), which enables you to develop and integrate distributed applications in a service-oriented architecture environment. This messaging platform provides service for message persistence and recovery.

    - Third Party: Persists messages in a third-party JMS provider, such as Tibco JMS or IBM WebSphere MQ JMS.

16. If you selected **Oracle WebLogic JMS** or **Oracle Advanced Queuing**, then the Service Connection page appears. On the Service Connection page, do one of the following:

    - From the **AppServer Connection** list, select an exising application server connection and click **Next**.

    - Or, click **New** to create an application server connection. Follow the Create Application Server Connection Wizard.

    If you selected **Third Party**, then the JMS Connection page appears. On the JMS Connection page, provide the JMS Connection JNDI Name and click **Next**.

17. If you selected **Send** in Step 7, then the Produce Operation Parameters page appears. If you selected **Receive** in Step 7, then the Consume Operation Parameters page appears.

    On the Produce Operation Parameters page, provide the following information and click **Next**:

- Destination Name: Enter the JNDI name of the queue or topic to produce the message or click **Browse** to select a name. The value of this field is typically the JNDI name.

- Message Body Type: Select the message body (payload) type.

  - Text Message: Use this option when the payload is a string.

  - Bytes Message: Use this option when the payload is an array of primitive bytes.

- Delivery Mode: Select a message delivery mode (only if WebLogic JMS was selected).

  - Persistent: Use this option for messages that are persisted to a file system or database.

  - Non-Persistent: Use this option for messages that are not persisted and are typically held in process memory only.

- Priority: Select a priority value, with 9 representing the highest priority and 0 representing the lowest priority.

- Time To Live: Enter a value that indicates the life span of the message. If no subscribers consume the message in the given time, then the message is not delivered. There is no limit. A value of 0 indicates that there is no expiration time.

- JNDI Name: Displays the JNDI name based on your selection in the Destination Name field (only if WebLogic JMS or Advanced Queuing is selected).

On the Consume Operation Parameters page, provide the following information and click **Next**:

- Destination Name: Enter the JNDI name of the queue or topic to consume the message or click **Browse** to select a name.

- Message Body Type: Select the message body (payload) type.

  - Text Message: Use this option when the payload is a string.

  - Bytes Message: Use this option when the payload is an array of primitive bytes.

- Message Selector: Specify filtering logic that enables you to receive messages that match certain criteria. Enter an expression between 1 and 255 characters in length. Use SQL92 syntax in this field. The JMS server uses these criteria to filter messages received by this consumer. This works with variables defined in standard JMS headers and user-defined properties. You cannot use variables or elements that are in the payload of the message.

- Use MessageListener: This option is set to false by default if you selected **Oracle Weblogic JMS** on the JMS Provider page. It is not editable. Select true or false if you selected **Oracle Advanced Queueing** on the JMS Provider page (only if **Oracle Advanced Queuing** or **Third Party** is selected).

- Durable Subscriber ID: Enter an ID for receiving messages from a JMS topic. If you do not specify an ID, then you must have an active subscription session to receive messages. If you specify an ID for topics, then you receive messages even if you do not currently have an active subscription session. When a durable subscriber is disconnected from the JMS server, the server stores messages. When the durable subscriber reconnects, the server sends the unexpired messages that accumulated (only if **Third Party** is selected).

- JNDI Name: Displays the JNDI name based on your selection in the **Destination Name** field (only if **WebLogic JMS** or **Advanced Queuing** is selected).

- Enable Streaming: When you enable this feature, the payload is streamed to a database. Use this feature for large payloads. When you enable streaming, a corresponding Boolean property, `StreamPayload`, is appended to the `ActivationSpec` properties defined in the respective `.jca` file. If the StreamPayload property does not exist, then the default value `false` is assumed. The property is applicable when processing `Raw` messages, `XMLType` messages, and `ADT` type messages for which a payload is specified though an `ADT` attribute.

18. Click **Finish**.

See the following in *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for more information on binding components:

- "Adding Service Binding Components"

- "Adding Reference Binding Components"

- "Getting Started with Binding Components"

2.5.2 About Using the JMS Integration Type in the B2B Configuration Wizard

If you select the JMS integration type, then you must use JMS properties prefixed with `jca.jms.JMSProperty`, as in `jca.jms.JMSProperty.property_name`. (In contrast, when you select the Default or AQ integration types in the B2B Configuration Wizard, message properties such as `MSG_ID`, `INREPLYTO_MSG_ID`, `FROM_PARTY`, `TO_PARTY`, `ACTION_NAME`, `MSG_TYPE`, `DOCTYPE_NAME`, and `DOCTYPE_REVISION` can be used in the SOA composite application without any changes.)

Table 2–9 lists the JMS properties to use with the Oracle JCA Adapter for JMS.

Table 2–9   JMS Adapter Properties

Property Name	Property Name When Used with the Oracle JCA Adapter for JMS
MSG_ID	jca.jms.JMSProperty.MSG_ID
INREPLYTO_MSG_ID	jca.jms.JMSProperty.INREPLYTO_MSG_ID
FROM_PARTY	jca.jms.JMSProperty.FROM_PARTY
TO_PARTY	jca.jms.JMSProperty.TO_PARTY
ACTION_NAME	jca.jms.JMSProperty.ACTION_NAME
MSG_TYPE	jca.jms.JMSProperty.MSG_TYPE
DOCTYPE_NAME	jca.jms.JMSProperty.DOCTYPE_NAME
DOCTYPE_REVISION	jca.jms.JMSProperty.DOCTYPE_REVISION
ATTACHMENT	jca.jms.JMSProperty.ATTACHMENT
A2A	jca.jms.JMSProperty.A2A

For JMS message correlation, set the property `A2A=true`. If `MSG_ID` is provided from the back-end application, then `MSG_ID` is set as JMS Correlation ID in the B2B output, otherwise the JMS Message ID is set to JMS Correlation ID in the B2B output.

For example, these sender properties and values are added in the Assign Values dialog of the Mediator component for an outbound composite, as shown in Figure 2–15.

Figure 2–15   Adding JCA JMS Properties in the Assign Values Dialog
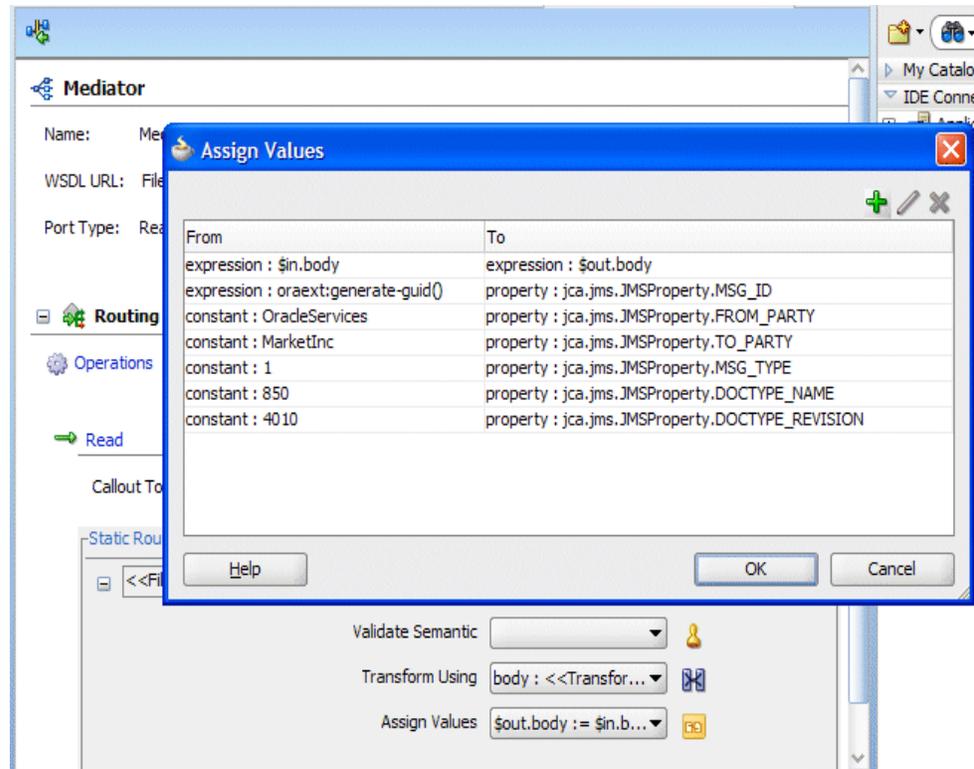
The image shows the following assignments: expression:$in.body is mapped to expression:$out.body; expression:oraext:generate-guid() is the value for property: jca.jms.JMSProperty.MSG_ID; constant: OracleServices is the value for jca.jms.JMSProperty.FROM_PARTY; constant: MarketInc is the value for jca.jms.JMSProperty.TO_PARTY; constant: 1 is the value for property: jca.jms.JMSProperty.MSG_TYPE; constant: 850 is the value for property: jca.jms.JMSProperty.DOCTYPE_NAME; constant: 4010 is the value for the property jca.jms.JMSProperty.DOCTYPE_REVISION.

****

2.5.3 The B2BX12OrderGateway Composite in the Fusion Order Demo

The SOA sample application, Fusion Order Demo, includes a B2B composite, B2BX12OrderGateway. In this demo, a trading partner sends an EDI purchase order request (850) to B2B. The trading partner does this by dropping a file into a directory on which B2B is listening through a listening channel. B2B takes the standard EDI 850 and transforms it into a canonical form, which is a common representation of the data to invoke the Store Front Service—the composite that provides access to the storefront data transaction support to update data for customers, orders, and products.

If the purchase order amount is greater than $2,000 (as determined by a business rule), then the purchase order is routed to a Human Workflow for manual approval. For orders under $2,000, approval is not required. After the order is approved, a purchase order response (855) is generated and sent back to the trading partner.

Figure 2–16 shows the B2BX12OrderGateway composite in Oracle JDeveloper.

Figure 2–16   B2B Composite in the Fusion Order Demo
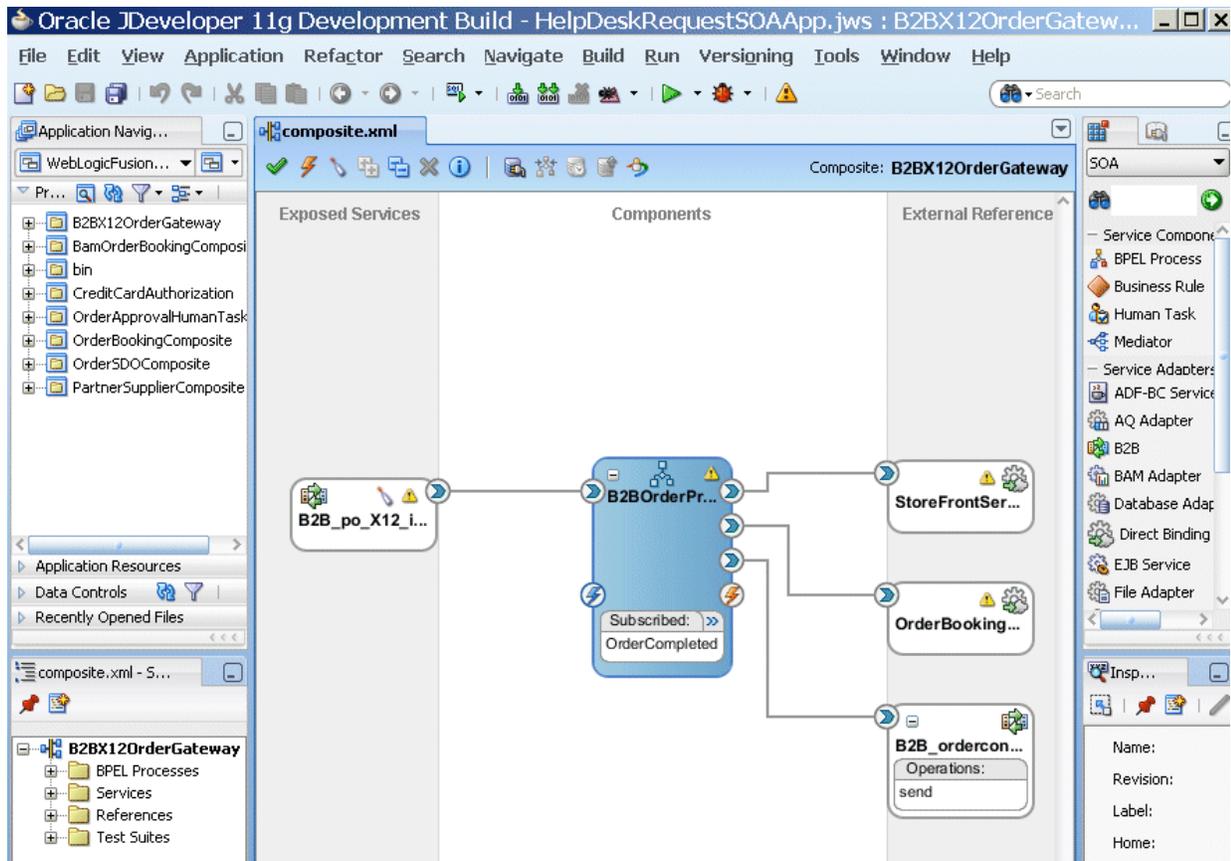
The image shows a B2B service binding component named B2B_po_X12_i..., and BPEL process named B2BOrderPr..., a web service name StoreFrontSer..., a web service named OrderBooking..., and a B2B reference binding component named B2B_ordercon... (send operation).

****

Download the Fusion Order Demo application ZIP file (`FusionOrderDemo_R1PS3.zip`). from:

http://www.oracle.com/technetwork/developer-tools/jdev/index-095536.html

See *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite* for a description of the SOA sample application.

2.6 What You May Need To Know About Using Oracle B2B

This section contains topics to help with troubleshooting.

2.6.1 Enabling Debug Mode at Run Time

Use Oracle Enterprise Manager Fusion Middleware Control to enable logging (**SOA Infrastructure** > **Logs** > **Log Configuration**). See *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite* for more information.

An alternative is to edit the `logging.xml` file at

DOMAIN_HOME`/config/fmwconfig/server/managed_server`

2.6.2 Logging Out: SSO Logout Configuration for Oracle Access Manager

In `web.xml`, the `success_url` parameter of
`oracle.adf.share.security.authentication.AuthenticationServlet`
must contain an SSO logout URL, such as
`../access/oblix/lang/en-us/logout.html`, to ensure that the URL is
accessible and does not result in a 404 error.

See *Oracle Fusion Middleware Security Guide* for information about Oracle Single
Sign-On and Oracle Access Manager.

Part II

Oracle B2B Process Flow

This part describes the Oracle B2B process flow.

This part contains the following chapters:

3

Creating Guideline Files

This chapter describes the process of creating document guideline files by using Oracle B2B Document Editor.

The first step in the Oracle B2B process flow, shown in Figure 3–1, is to create document guideline files.

Figure 3–1   Oracle B2B Process Flow
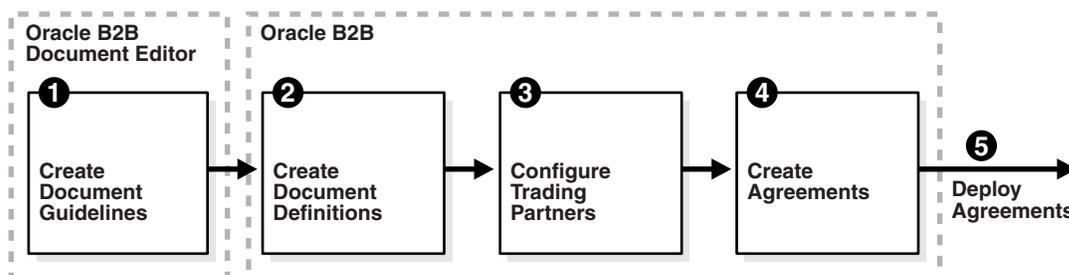
The image shows a step diagram linking the following blocks from left to right: 1. Oracle B2B Document Editor: Create Document Guidelines; 2. Oracle B2B: Create Document Definitions; 3. Oracle B2B: Configure Trading Partners; 4. Oracle B2B: Create Agreements; 5. Deploy Agreements.

****

Oracle B2B Document Editor is a guideline creation and implementation application for defining and managing custom document definitions for Oracle B2B transactions.

This chapter contains the following topics:

- Section 3.1, "Introduction to Oracle B2B Document Editor"

- Section 3.2, "Installing Oracle B2B Document Editor"

- Section 3.3, "Creating Guideline Files: EDIFACT D98 Example"

For complete documentation on the document editor, see the Oracle B2B Document Editor **Help** menu.

3.1 Introduction to Oracle B2B Document Editor

Oracle B2B Document Editor is a guideline creation and implementation application for business-to-business (B2B) electronic commerce (e-commerce). Use the document editor to simplify developing, migrating, testing, distributing, and printing your electronic business (e-business) guideline documents. You can create new guideline

documents or use the document editor's comprehensive library of standards as templates.

Using an existing standard as a template, you can create new guidelines by changing the attributes of underlying segments, elements, and codes. You can also create a guideline file from a data file.

Figure 3–2 shows the types of available document guidelines: delimited flat file, EDI, HL7 2.x, HL7v3, NCPDP, ParserSchema, positional flat file (which includes SAP iDocs), RosettaNet, and XMLSchema.

Figure 3–2 Document Guidelines Available in Oracle B2B Document Editor
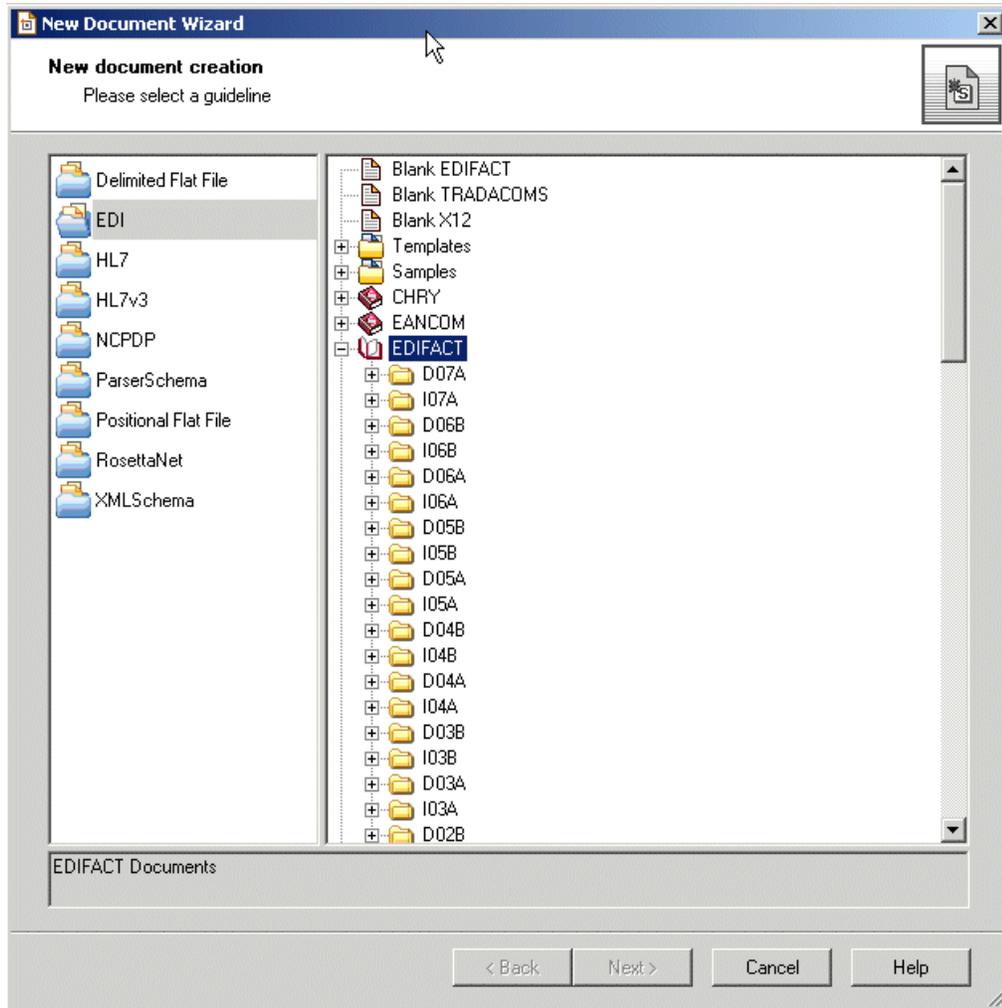
The image shows the following available document guidelines: Delimited flat file, EDI, HL7, HL7v3, NCPDP, ParserSchema, Positional flat file, RoesettaNet, and XMLSchema. The EDIFACT note is expanded and various EDIFACT guideline files (such as D07A) are listed.

****

In addition to using the RosettaNet document guide lines in the document editor, you can also download standard DTD files from the RosettaNet Web site.

After creating a custom guideline file, use the Oracle B2B interface to include the documents in the document definition, as shown in Figure 3–3. See Section 4.2, "Creating Document Definitions," for more information about this step.

Figure 3–3   Importing XSD and ECS File Created in Oracle B2B Document Editor
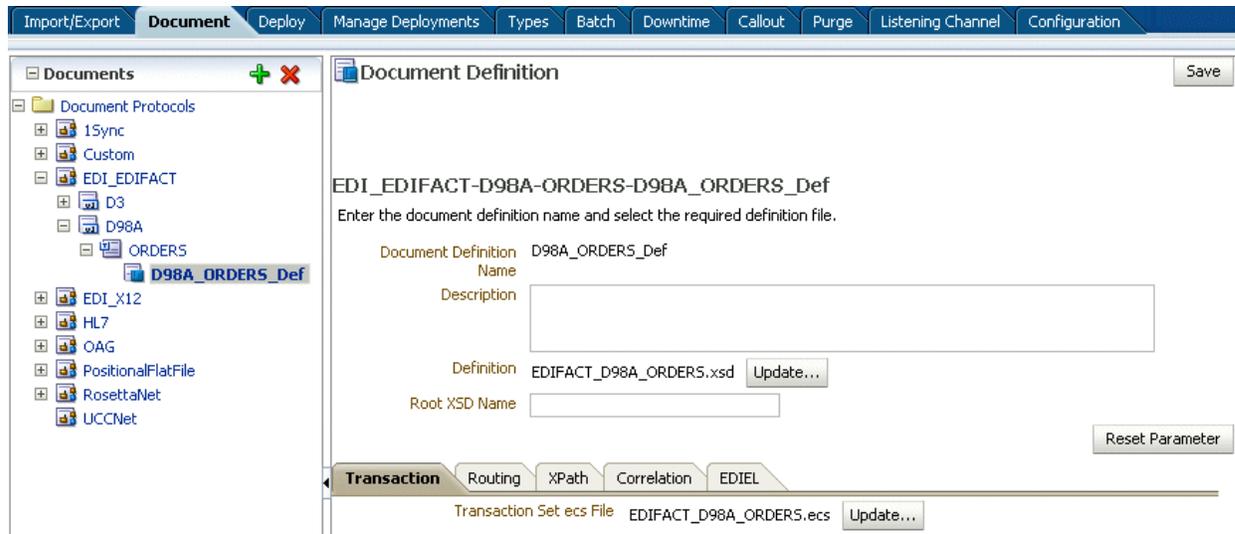
The image shows the Oracle B2B Document Editor displaying the following tabs: Document, Deploy, Management Deployments, Types, Import/Export, Schedule Batch, Manage Batch, and Callout. The Document tab is selected. The left pane of the editor shows an expanded hierarchy of Document Protocols with ORDERS_def selected. The right pane of the editor show the ORDERS_def details including the Document Definition Name, Document Definition Description, and Definition fields. Below the fields is the Update button.

****

In Figure 3–3, `EDIFACT_D98A_ORDERS.xsd` and `EDIFACT_D98A_ORDERS.ecs` are imported to create the document definition. The ECS file is required in B2B for translating and validating documents. The XSD is optional in B2B; however, it provides an easy reference to the document schema when modeling a SOA composite for sending and receiving the document.

> **Note:**   The MapBuilder component of Oracle Document Editor is not certified or supported.

3.2 Installing Oracle B2B Document Editor

Oracle B2B Document Editor can be obtained by product CD or downloaded from the Oracle Technology Network Software Downloads page:

`http://www.oracle.com/technetwork/middleware/downloads/fmw-11-download-092893.html#depends`

To install the Oracle Document Editor, extract the downloaded files onto your disk drive, and run `Install.exe`. Several installation scripts guide you through the process of installing each of the Oracle Document Editor components.

Oracle B2B Document Editor runs on Microsoft Windows only (Win 2000, WinXP, Vista[1] 32-bit and 64-bit, and Windows Server 2003), and requires the Microsoft .NET

framework (installed automatically from the CD) for full support of W3C XML Schema guidelines.

Complete usage instructions are available from the Oracle B2B Document Editor **Help** menu by searching on *installation* and displaying the **Preparation** topic. A list of new features in this release of the document editor is also provided.

3.3 Creating Guideline Files: EDIFACT D98 Example

The following example describes how to create the guideline files—the ECS and XSD files— required to send an EDIFACT D98A purchase order, and how to generate and validate test data files based on the D98A–ORDERS guideline.

To create the EDIFACT transaction documents for this scenario, do the following:

- Task 1, "Create the ECS File"
- Task 2, "Create the XSD File"
- Task 3, "Generate Data Using the ECS File"
- Task 4, "Analyze the Data"

Task 1 Create the ECS File

Using an existing EDIFACT guideline (standard) as a template, create a purchase order guideline file called **orders.ecs**.

1. Open Oracle B2B Document Editor.

2. Click **New Document** and then **EDI**, as shown in Figure 3–4.

Figure 3–4   Creating a New Document in Oracle Document Editor
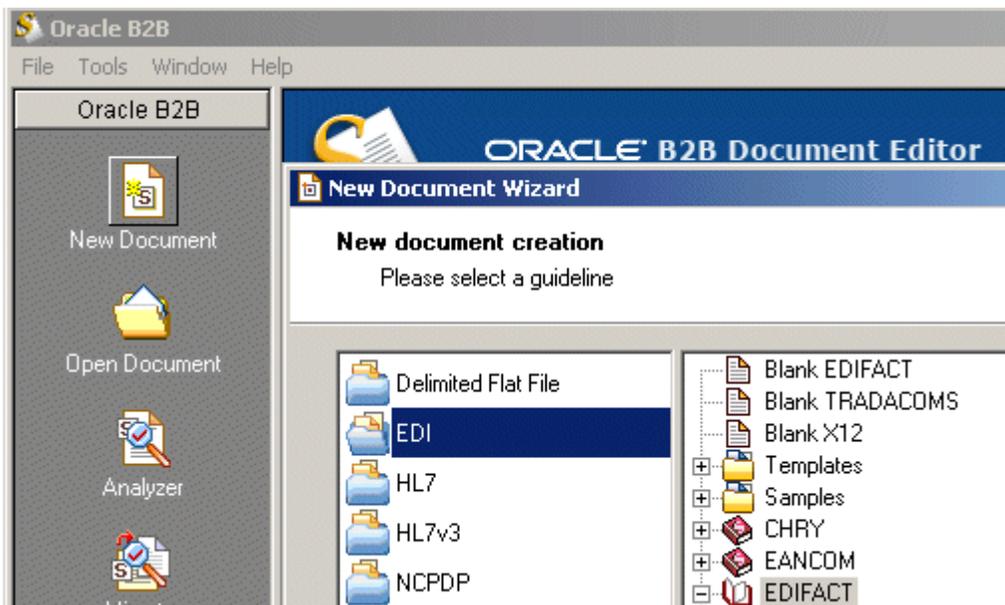
The image shows the New Document Wizard. The left pane of the wizard shows a series of folders with EDI selected. The right pane of the wizard shows a hierarchy of guidelines with EDIFACT selected.

---

[1] When using Microsoft Vista, do not install Oracle Document Editor in the program folder, for which admin privilege is needed.

****

3. Expand **EDIFACT** and **D98A**.

4. Select **ORDERS - Purchase order message**, as shown in Figure 3–5, and click **Next**.

Figure 3–5   Creating a New Document: Selecting a Guideline

The image shows the New Document Wizard. The left pane of the wizard shows a series of folders with EDI selected. The right pane of the wizard shows a hierarchy of guidelines with ORDERS - Purchase order message selected.

****

5. Ensure that **Insert Envelope Segments** is *not* selected, as shown in Figure 3–6, and click **Finish**.

Figure 3–6   Creating a New Document: Envelope Segments
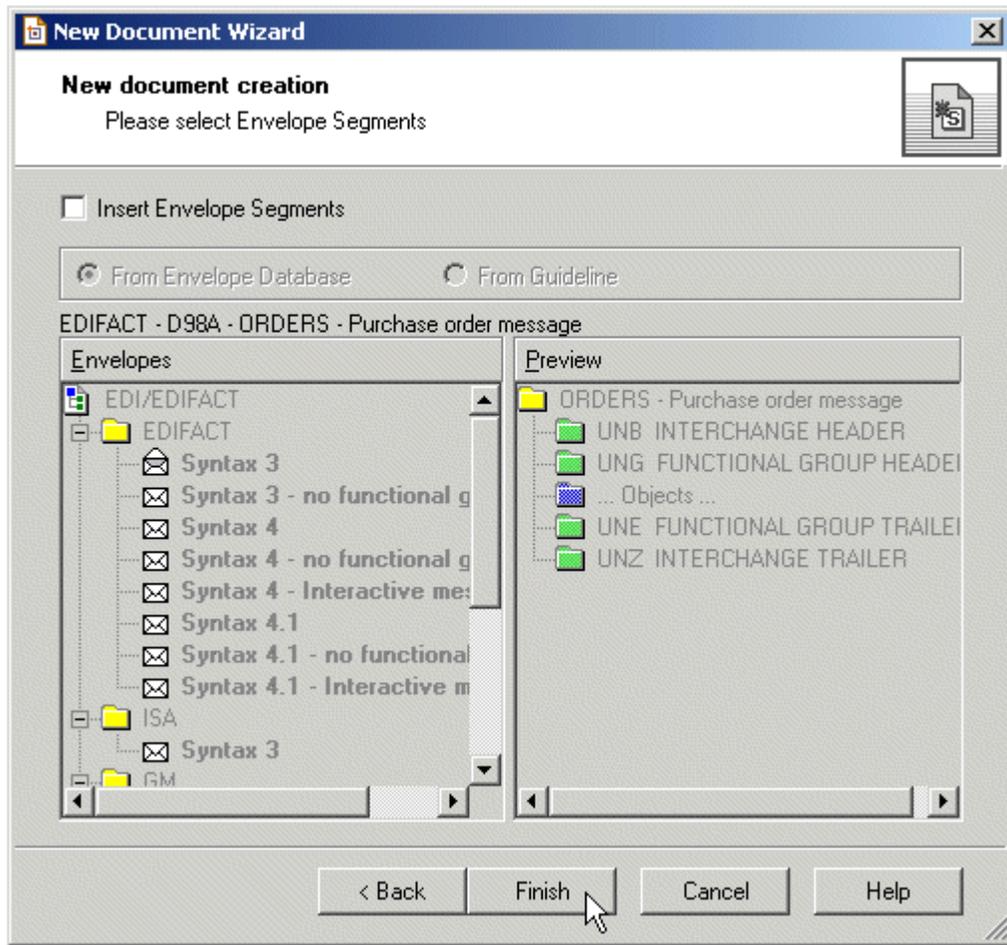
The image shows the New Document Wizard. The first area of the wizard shows the Insert Envelope Segments check box. The left pane of the wizard shows a hierarchy of envelopes with EDIFACT and ISA expanded. The right pane of the wizard shows a preview hierarchy with ORDERS - Purchase order message expanded. Below the panes are the Back, Finish, Cancel, and Help buttons.

****

Oracle B2B Document Editor is preseeded with all versions of the interchange (envelope). Oracle B2B handles the envelop based on the settings.

To override the default preseeded envelope information, create a separate ECS file with only the required envelope information (without the transaction details) and then upload it using the envelope override option available in the document version.

6. (Optional) Edit the segment-level details.

No edits are needed for this scenario, as shown in Figure 3–7.

Figure 3–7   Creating a New Document: Segment-Level Details
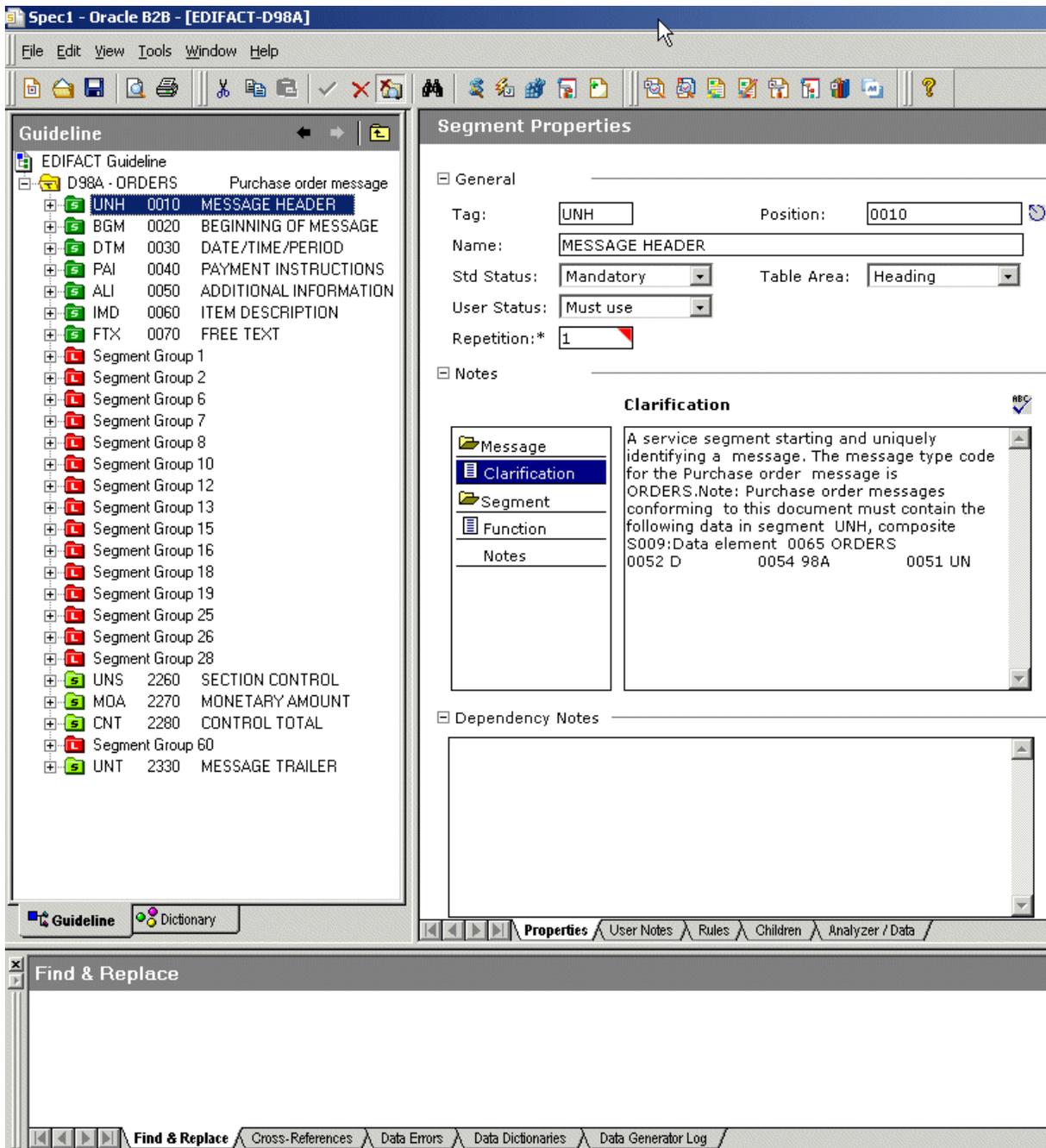
The image shows the Oracle B2B Document Editor. The left pane is labeled Guideline and shows an expanded hierarchy of guidelines with the UNH 0010 MESSAGE HEADER selected. The right pane shows the Segment Properties and includes General information, Notes, and Dependency Notes. The General information includes the following fields: Tag, Position, Name, Std Status, Table Area, User Status, and Repetitor. The Find & Replace pane shows at the bottom of the screen.

****

7.   From **File**, select **Save**.

8.   Accept the default directory and enter `orders.ecs` for the file name.

By default, the ECS file is saved to `My Documents\Oracle\Oracle B2B\ Guidelines`.

Task 2 Create the XSD File

Using the guideline file in its internal format (the ECS file), create an XML schema definition file (the XSD format) to use with Oracle B2B.

1. From **File**, select **Open**.

2. Select `orders.ecs` and click **Open**.

3. From **File**, select **Export**.

4. In the Export Wizard, select **Oracle B2B 2.0** from the list of export types, as shown in Figure 3–8, and click **Next**.

    Use the **Oracle B2B 2.0** export type to provide a namespace of your choice, as in `urn:oracle:b2b:EDIFACT/D98A/ORDERS` for this example. (Use the **Oracle B2B** option to have a fixed namespace provided for you.)

Figure 3–8 Selecting the Oracle B2B 2.0 Export Type

The image shows the welcome screen of the Export Wizard dialog box. The first area shows a list of export options with Oracle B2B 2.0 selected. The second area shows a description of the selected export option. The final area at the bottom shows the Back, Next, Cancel, and Help buttons. In this situation, the Back button is unavailable.

****

5. In the Export Destination dialog, do the following and click **Next**.

    ■ Accept the default directory

    ■ Select **Save guideline before exporting**

- Select **Show advanced options**

The XSD file is saved with the ECS file in `My Documents\Oracle\Oracle B2B\Guidelines`.

6. In the XSD Namespace Options dialog, do the following and click **Next**.

    - Select **Custom namespace**

    - Provide a namespace, in this example,
      `urn:oracle:b2b:EDIFACT/D98A/ORDERS`

7. In the Templates Configuration dialog, click **Next**.

    No edits to the elements in the template are needed in this scenario.

8. In the Conversion Options dialog, shown in Figure 3–9, do the following and click **Next**.

Figure 3–9   Selecting Conversion Options
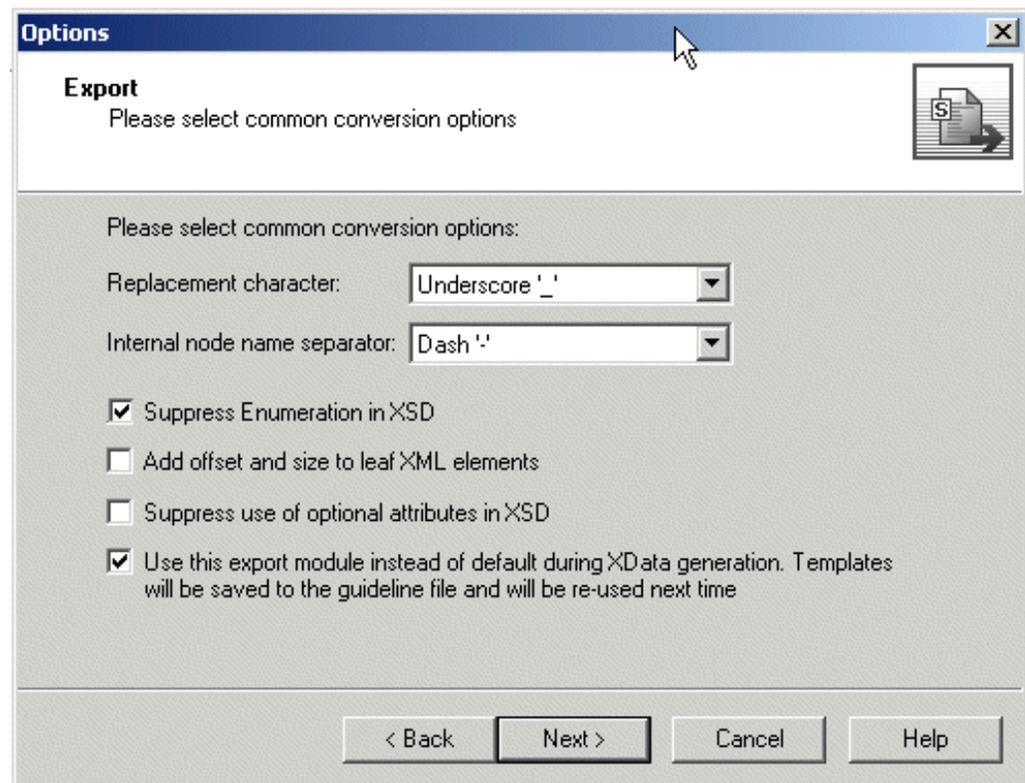
The image shows the Options screen of the Export Wizard dialog box. The first area shows the Replacement character drop-down list and Internal node name separator drop-down list. The first area also shows the following four check boxes: Suppress Enumeration in XSD, Add offset and size to leaf XML elements, Suppress use of optional attributes in XSD, and Use this export module instead of default during XData generation (selected). The final area at the bottom show the Back, Next, Cancel, and Help buttons.

****

- Check the **Suppress Enumeration in XSD** option. This is recommended because code lists are in the ECS file. Suppressing enumeration reduces the XSD size considerably.

■ Check the **Use this export module instead of default during XData generation** option.

9. In the Document Conversion Options dialog, accept the default, **Allow to use SegmentCount** macro, and click **Next**.

The SegmentCount macro counts the number of segments. The data type of the XSD element is changed from numeric to string to enable the count.

10. Ensure that the **Launch Oracle B2B** option is not select (it is not needed in this scenario) and click **Next**.

If you want to start Oracle B2B, enter the URL for your B2B interface (`http://host_name:port/b2b`).

11. In the Macro Nodes dialog, click **Next**.

No macros are needed for any of the nodes in this scenario.

If you see the message "Some characters were replaced in XSD names because they are not allowed," click **OK**.

12. Click **Finish**.

The `orders.xsd` file is created in Oracle B2B 2.0 format.

Task 3 Generate Data Using the ECS File

Using the Data Generator, create a test data file based on the guideline.

1. Click **Data Generator**.

2. Select **New Test Case** and click **Next**.

3. Click **Generate** and click **Next**.

This step generates new data using the specified data dictionaries.

4. Select **From a guideline file**, select **ORDERS.ecs**, and click **Next**.

5. Select **Select Envelope Segments from the Standards Database** and click **Next**.

6. Select the **Syntax 3** envelope segment, as shown in Figure 3–10, and click **Next**.

Figure 3–10   Selecting Envelope Segments from the Database
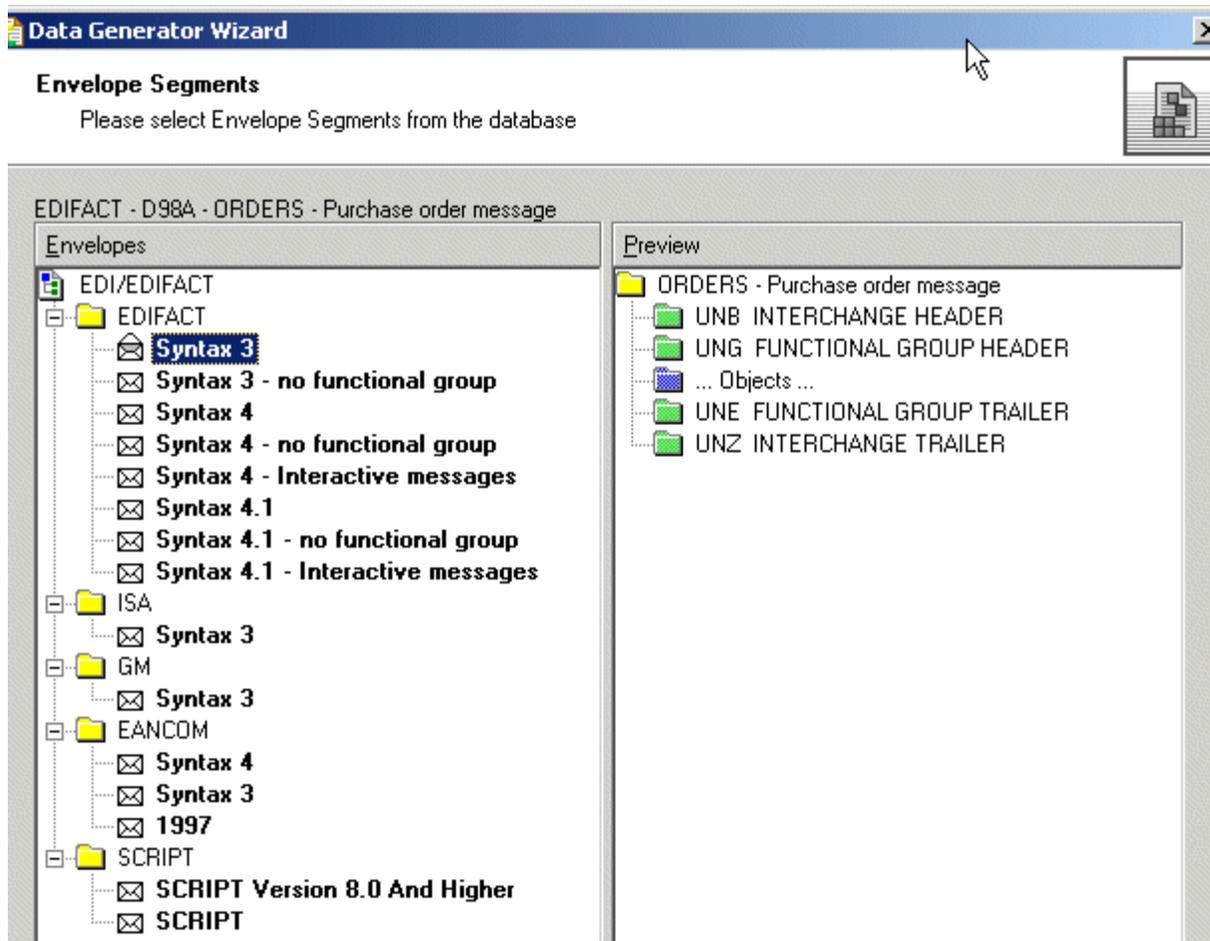
The image shows a section of the Envelope Segments screen of the Data Generator Wizard. The left pane shows Envelopes, an expanded hierarchy with EDI/EDIFACT > EDIFACT > Syntax 3 selected. The right pane shows Preview, an expanded hierarchy for ORDERS - Purchase order message.

****

7. Select **Use directly from the Standards Database** and click **Next**.

    The envelope segments are not incorporated in the guideline file.

8. Select **Mandatory + Percentage of optional data** and move the slider to indicate the percentage.

9. Select **User Option** and click **Next**.

10. Select **Any size** and click **Next**.

11. Select **Do not reset** and click **Next**.

12. Set the repeat count options, depending on how many messages you want generated.

13. Select any data dictionaries you want to use.

14. Accept the default delimiters and click **Next**.

15. Click **Output Data file name**, enter `C:\D98A_ORDERS.dat` and click **Next**.

The DAT file opens, as shown in Figure 3–11.

Figure 3–11   The DAT File
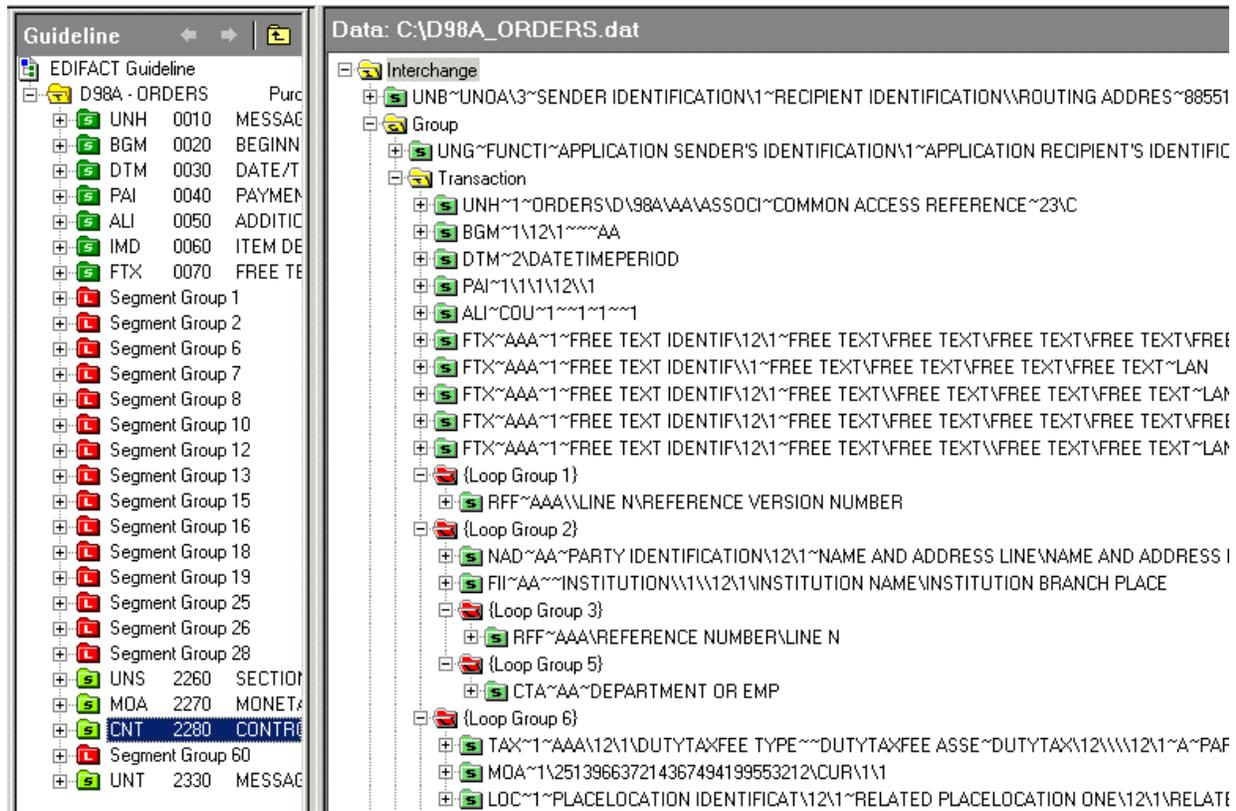
The image shows a screen with left and right panes. The left pane is labeled Guideline, and shows an expanded hierarchy with CNT 2280 selected. The right pane is labeled Data, and shows the expanded hierarchy of the D98A_ORDERS.dat file.

****

16. Save and close the file.

Task 4   Analyze the Data

Using the Analyzer, validate the data file against the `orders.ecs` guideline file, and test the data file against the standard to check for required segments or elements that may be missing.

1. Click **Analyzer**, shown in Figure 3–12.

Figure 3–12   The Analyzer Icon
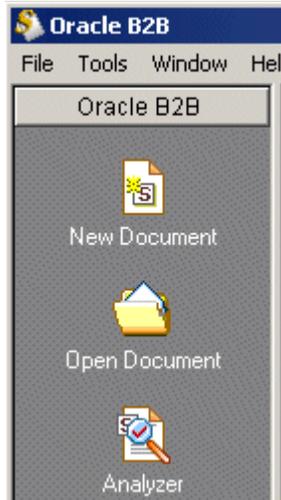
The image shows a section of the Oracle B2B window with the New Document, Open Document, and Analyzer icons visible.

****

2.   Browse for `D98A_orders.dat` and click **Next.**

3.   Ensure that **Show Advanced Options** is selected, as shown in Figure 3–13, and click **Next**.

Figure 3–13   Selecting Show Advanced Options
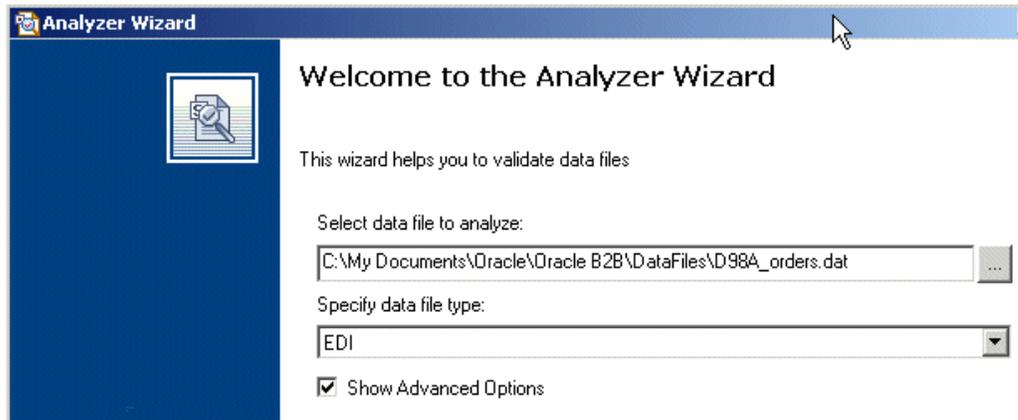
The image shows a section of the welcome screen of the Analyzer Wizard. The screen shows the Select data file to analyze field, the Specify data file type drop-down list, and the Show Advanced Options check box.

****

4.   In the Clean Up Data File dialog, click **Next**.

      No preprocessing is needed in this scenario.

5.   In the Data Structure dialog, click **Next**.

      The entire document is validated by default.

6. Select the guideline file (ECS file) against which to check the data, as shown in Figure 3–14. Do the following and click **Next**.

- Select **From a guideline file**.

- Select **orders.ecs**.

Figure 3–14   Selecting a Guideline (ECS) File

The image shows a section of the Guideline selection screen of the Analyzer Wizard. The first area shows two radio controls: From the database and From a guidelines file (selected). The second area shows the Look in field with the Guidelines folder selected. The final area shows a list with the orders.ecs and Spec1.ecs files.

****

7. Select **Select Envelope Segments from the Standards Database** and click **Next**.

   The selected guideline file (ECS file) does not contain envelope segments.

8. Select the **Syntax 3** envelope segment, as shown in Figure 3–15, and click **Next**.

Figure 3–15   Selecting the Envelope Segment
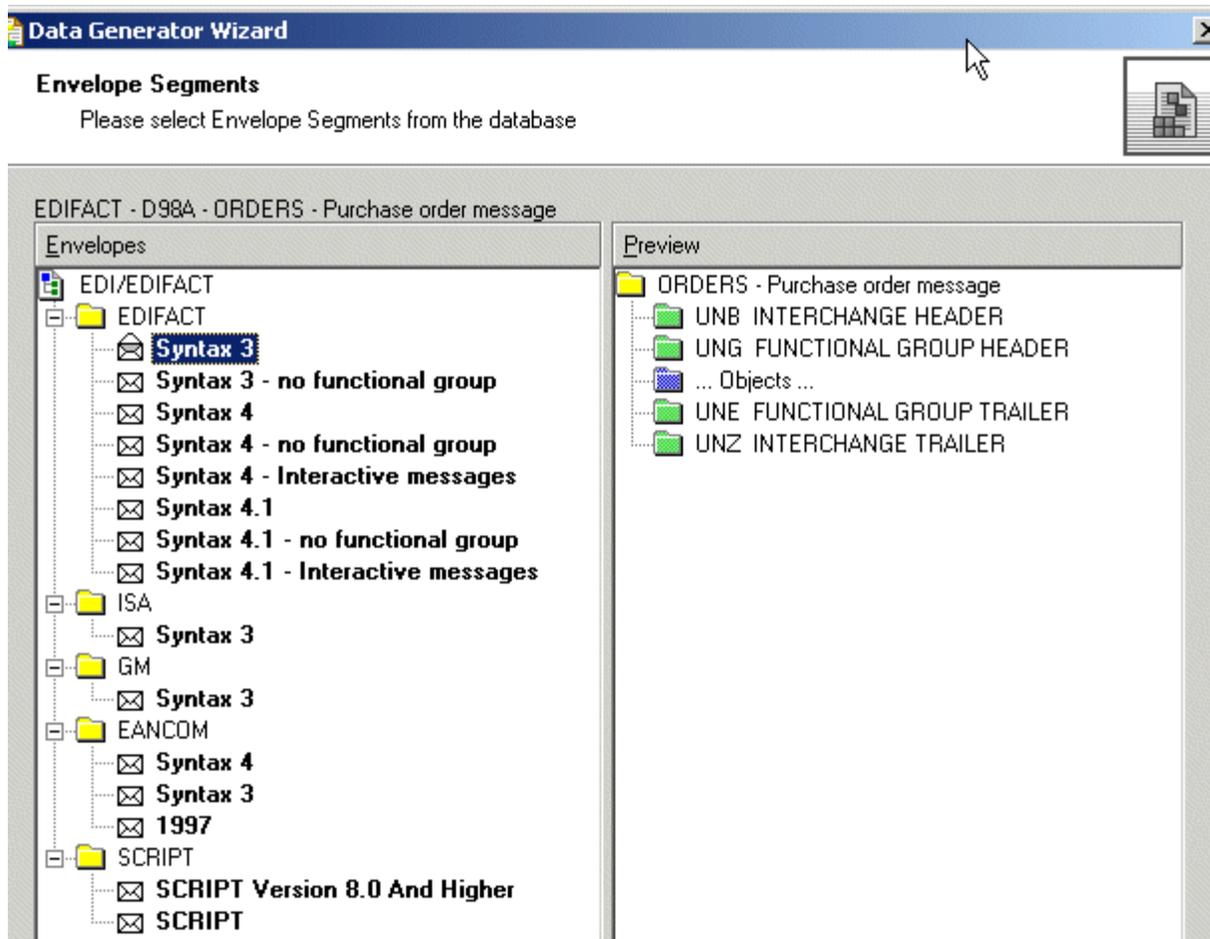
The image shows a section of the Envelope Segments screen of the Data Generator Wizard. The left pane shows Envelopes, an expanded hierarchy with EDI/EDIFACT > EDIFACT > Syntax 3 selected. The right pane shows Preview, an expanded hierarchy for ORDERS - Purchase order message.

****

9. Select **Use directly from the Standards Database** and click **Next**.

    The envelope segments are not incorporated in the guideline file.

10. In the Analyzer Mode and Outputs dialog, accept the default settings, set **Generate XData (XML)** to **Always** and click **Next**.

    The results, including any error messages, are displayed, as shown in Figure 3–16.

Figure 3–16   Results from the Analyzer
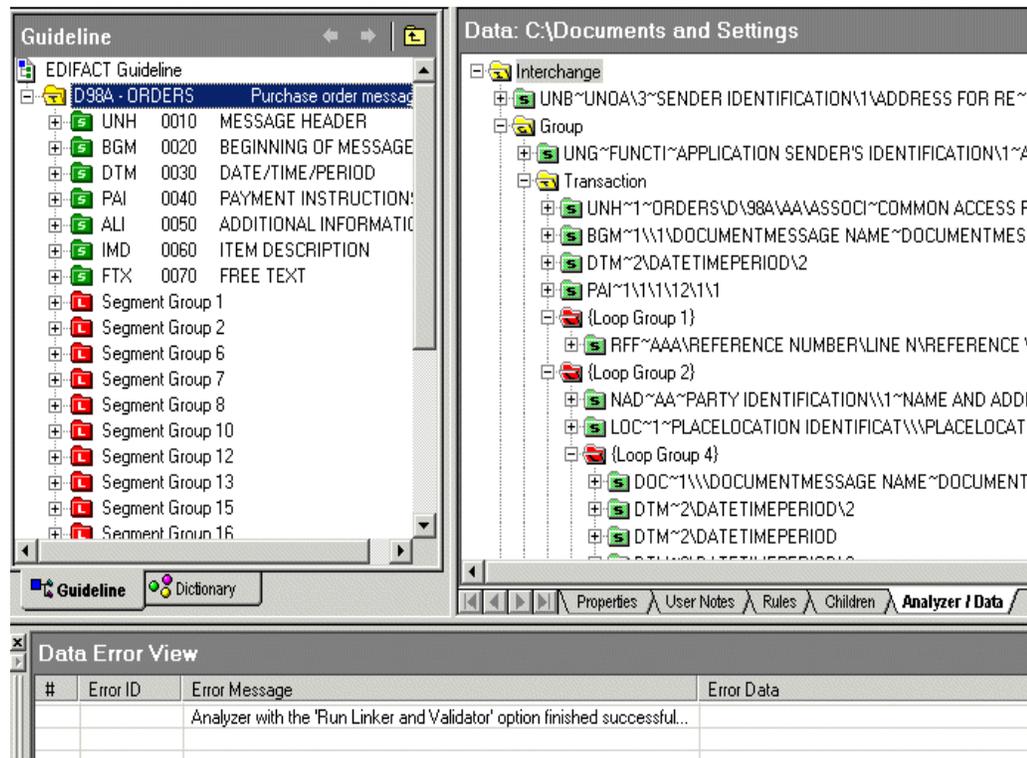
The image shows the D98A_orders.dat file structure. The left pane is labeled Guideline and shows an expanded hierarchy with EDIFACT Guideline > D98A - ORDERS selected. The right pane is labeled Data and shows an expanded hierarchy for Interchange. The bottom pane is labeled Data Error View and shows the following error message: "Analyzer with the "Run Linker and Validator" option finished successfully..."

****

To view the data in XML format, click the **XML** icon in the upper right corner, as shown in Figure 3–17.

Figure 3–17   Viewing the Data in XML Format
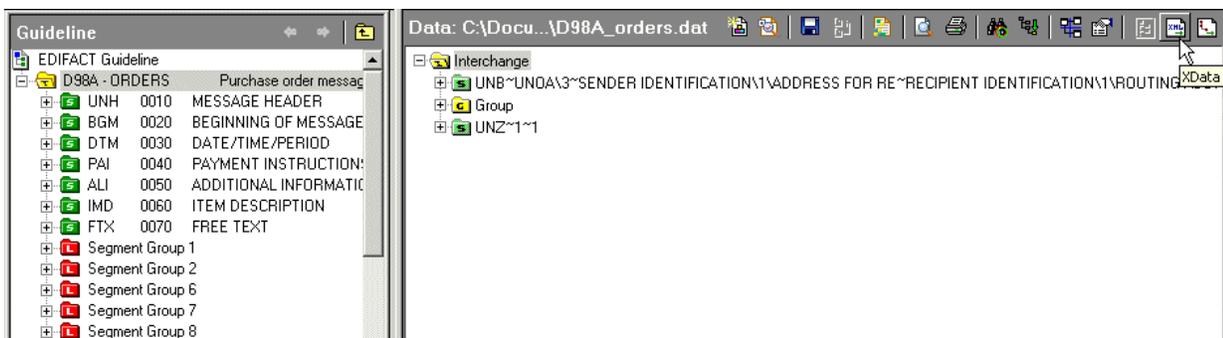
The image shows the D98A_orders.dat file structure. The left pane is labeled Guideline and shows an expanded hierarchy with EDIFACT Guideline > D98A - ORDERS selected. The right pane is labeled Data and shows an expanded hierarchy for Interchange.

****

Use the **View as XML** (shown in Figure 3–18) and **View as HTML** options to view the data. Click the **Save Data As** icon to export the XML report as an XML file.

Figure 3–18 Analyzer Data
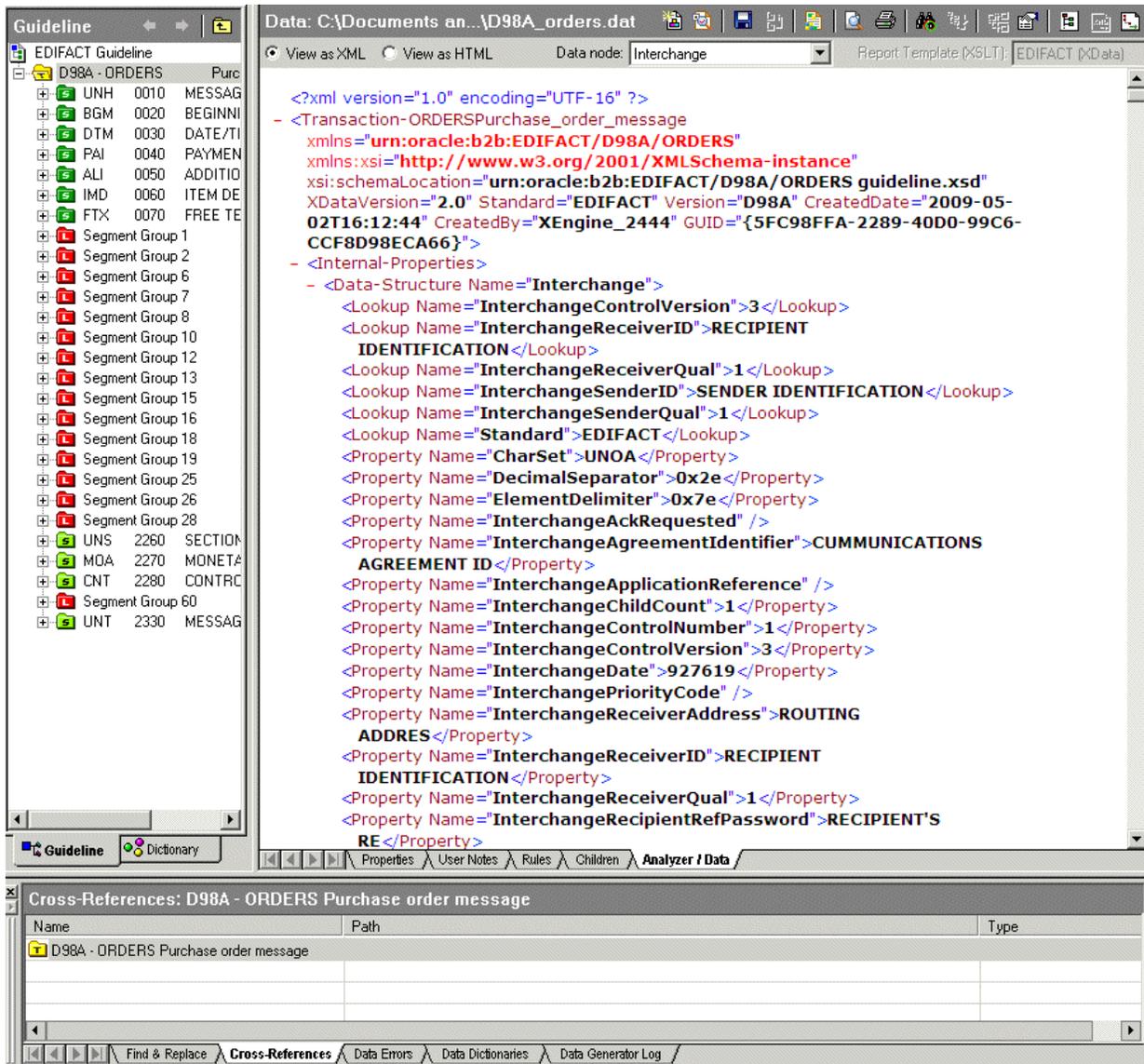
The image shows the D98A_orders.dat file as XML. The left pane is labeled Guideline and shows an expanded hierarchy with EDIFACT Guideline > D98A - ORDERS selected. The right pane displays the following XML:

<?xml version="1.0" encoding="UTF-16" ?>

<Transaction-ORDERSPurchase_order_message

xmlns="Urn:oracle:b2b:EDIFACT/D98A/ORDERS"

xsi:schemaLocation="urn:orcle:b2b:EDIFACT?D98A/ORDERS guideline.xsd"

XDataVersion="2.0" Standard="EDIFACT" Version="D98A"
CreatedDate="2009-05-02T16:12:44" CreatedBy="XEngine_2444"
GUID="{5FC98FFA-2289-40D0-99C6-CCF8D98ECA66}">

<Internal-Properties>

 <Data-Structure Name="Interchange">

<Lookup Name="InterchangeControlVersion">3</Lookup><Lookup
Name="InterchangeReceiverID">RECIPIENT IDENTIFICATION</Lookup><Lookup
Name="InterchangeReceiverQual">1</Lookup><Lookup
Name="InterchangeSenderID">SENDER IDENTIFICATION</Lookup><Lookup
Name="InterchangeSenderQual">1</Lookup><Lookup
Name="Standard">EDIFACT</Lookup><Property
Name="CharSet">UNOA</Property><Property
Name="DecimalSeparator">0x2e</Property><Property
Name="ElementDelimiter">0x7e</Property><Property
Name="InterchangeAckRequested"></Property><Property
Name="InterchangeAgreementIdentifier">CUMMUNICATIONS AGREEMENT
ID</Property><Property
Name="InterchangeApplicationReference"></Property><Property
Name="InterchangeChildCount">1</Property><Property
Name="InterchangeControlNumber">1</Property><Property
Name="InterchangeControlVersion">3</Property><Property
Name="InterchangeDate">927619</Property>

and so on.

The bottom pane is labeled Cross References and lists D98A - ORDERS Purchase order
message.

****

4

Creating Document Definitions

This chapter describes the process of creating and deleting document definitions. It also covers the concept of document protocols.

The second step in the Oracle B2B process flow, shown in Figure 4–1, is to create document definitions.

Figure 4–1   Oracle B2B Process Flow
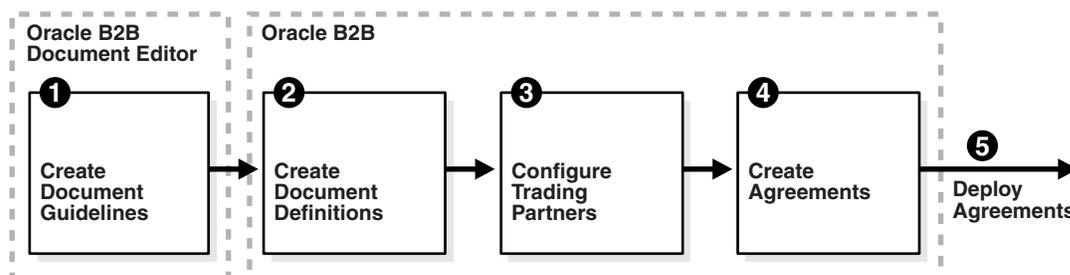
The image shows a step diagram linking the following blocks from left to right: 1. Oracle B2B Document Editor: Create Document Guidelines; 2. Oracle B2B: Create Document Definitions; 3. Oracle B2B: Configure Trading Partners; 4. Oracle B2B: Create Agreements; 5. Deploy Agreements.

****

A document definition specifies the document protocol—the document protocol version and document type—that is used to validate the message. The document definition can be an ECS file, in the case of EDI and HL7 messages, or an XSD/DTD, in the case of XML messages.

The same document definition is used by both the host and remote trading partner in a transaction. It must adhere to the standards for document protocols, protocol versions, and document types. This is straightfoward when you use Oracle B2B Document Editor to create the document guideline files (Step 1 in Figure 4–1) and then the Oracle B2B interface to import those files when creating the document definition (Step 2 in Figure 4–1).

This chapter contains the following topics:

- Section 4.1, "Introduction to Document Protocols"
- Section 4.2, "Creating Document Definitions"
- Section 4.3, "Deleting a Document Definition"

For more information on document protocols, see Chapter 8, "Using Document Protocols."

4.1 Introduction to Document Protocols

Figure 4–2 shows the document protocols supported in Oracle B2B. Using the Custom protocol and the many guideline documents in Oracle B2B Document Editor, you can define most protocols. When you add a new document protocol, it is always a Custom document.

Figure 4–2 Oracle B2B Document Protocols
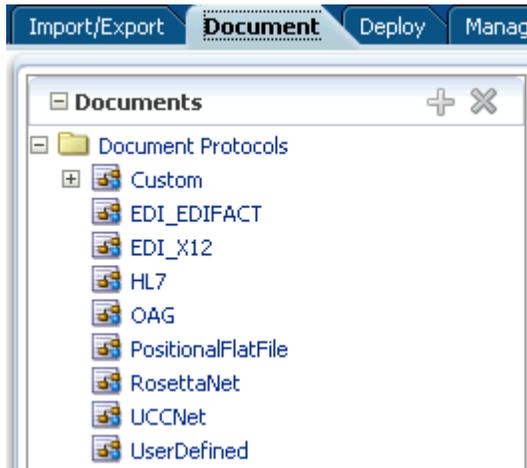
The image shows the following Oracle B2B tabs: Import/Export, Document, Deploy, and Manage Deployments. The Document tab is selected and shows the following document protocols: Custom, EDI_EDIFACT, EDI_X12, HL7, OAG, PositionalFlatFile, RosettaNet, UCCNet, and UserDefined.

****

As part of the document definition, you provide the document guideline files, which are typically created in Oracle B2B Document Editor. (For Custom documents, you cannot use Oracle B2B Document Editor.) If validation is enabled, then, at run time, the payload must conform to the document definition file type you use.

4.1.1 The Document Hierarchy

You can think of a document protocol as a hierarchy, as shown in Figure 4–3.

Figure 4–3 Document Hierarchy
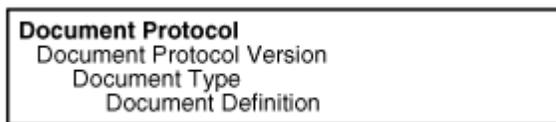
The image shows the document hierarchy displaying the following items (indented): Document Protocol, Document Protocol Version, Document Type, and Document Definition.

****

A document protocol can consist of multiple document protocol versions. A document protocol version can consist of multiple document types. A document type can consist of multiple document definitions. Typically, you start with one document definition and customize it for different trading partners.

Figure 4–4 shows a document protocol hierarchy as it applies to EDI X12.

Figure 4–4   EDI X12 Document Hierarchy
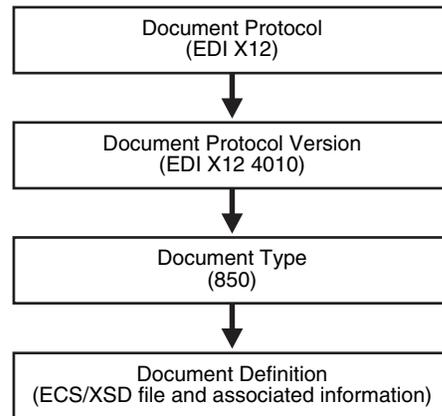
The image shows the document protocol hierarchy. The top level consists of the document protocol (EDI X12 in this example). The second level consists of the document protocol revision (EDI X12 4010 in this example). The third level consists of the document type (850 in this example). The fourth level (bottom of the hierarchy) consists of the document definition (the XSD file and associated information in this example).

****

In the Oracle B2B interface, as you create a document definition, the document protocol hierarchy is reflected in the definition:

DocumentProtocol—Version—DocumentType—DocumentDefinitionName

Example 4–1 shows the hierarchy reflected in the definition for an EDI EDIFACT document.

Example 4–1   Document Definition Name for an EDI EDIFACT Document

Document protocol: EDI_EDIFACT

Document protocol version: D98A

Document type: ORDERS

Document definition: ORDERS_def

The resulting document definition is

EDI_EDIFACT-D98A-ORDERS-ORDERS-def

Example 4–2 shows examples of document definitions for a Health Care 7 admit/visit notification and an X12 version 4010 purchase order, respectively.

Example 4–2   Document Definition Names for HL7 and X12 Documents

HL7-2.3.1-ACK_A01-ACK_A01_Doc_Def

EDI_X12-4010-850-850def

4.1.2 Document Protocols with Acknowledgments

For any message flow that involves an acknowledgment, Oracle B2B sends an acknowledgment only once. Resubmission does not generate another acknowledgment if the message has already been acknowledged. If further information about the message state is needed, then the trading partner must be notified by some other means (for example, e-mail).

4.2 Creating Document Definitions

After using Oracle B2B Document Editor to create the transaction set files, use the Oracle B2B interface to create the document definition and import the transaction set files.

Note: The document version, document type, and document definition are not editable after they are created. You must delete the specific document element (version, type, or definition) and create a new one. Updating the document elements after creation can lead to metadata inconsistency, metadata validation issues, and run-time errors.

To create a document definition:

1. Click the **Administration** tab.

2. Click the **Document** tab.

3. Select one of the document protocols, as shown in Figure 4–5.

Figure 4–5   Selecting a Document Protocol
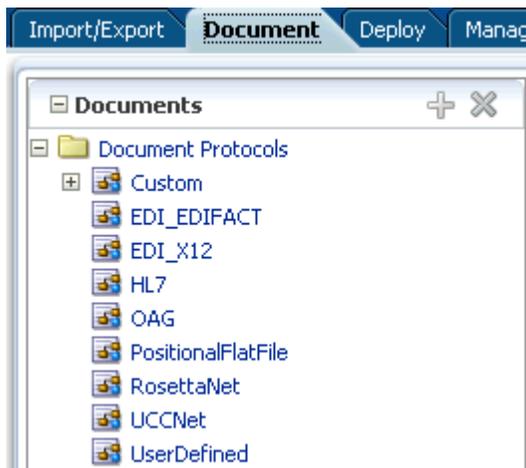
The image shows the following Oracle B2B tabs: Document, Deploy, and Manage Deployments. The Document tab is selected and shows the following document protocols: Custom, EDI_EDIFACT, EDI_X12, HL7, OAG, PositionalFlatFile, RosettaNet, UCCNet, and UserDefined.

****

> **Note:**
>
> To create a new Custom document with a name that you provide, click the **Document Protocols** folder and click **Add**.
>
>
>
> Then enter a protocol name, for example, MyXML_Document. Do not use an existing document protocol name.
>
>
>
> The image shows NewCustomDocumentProtocol in the Document Protocols list and provides a field for you to enter a name, MyXML_Document in this example, and optional description.
>
> **
> ***********************************
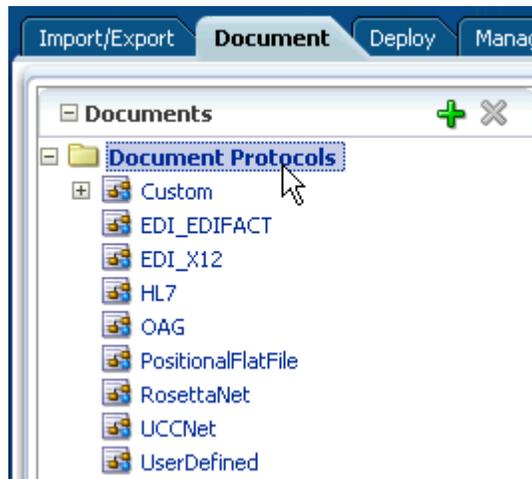
4. Click **New Version**.

Figure 4–6 shows the **New Version** button.

Figure 4–6   Creating a Document Version
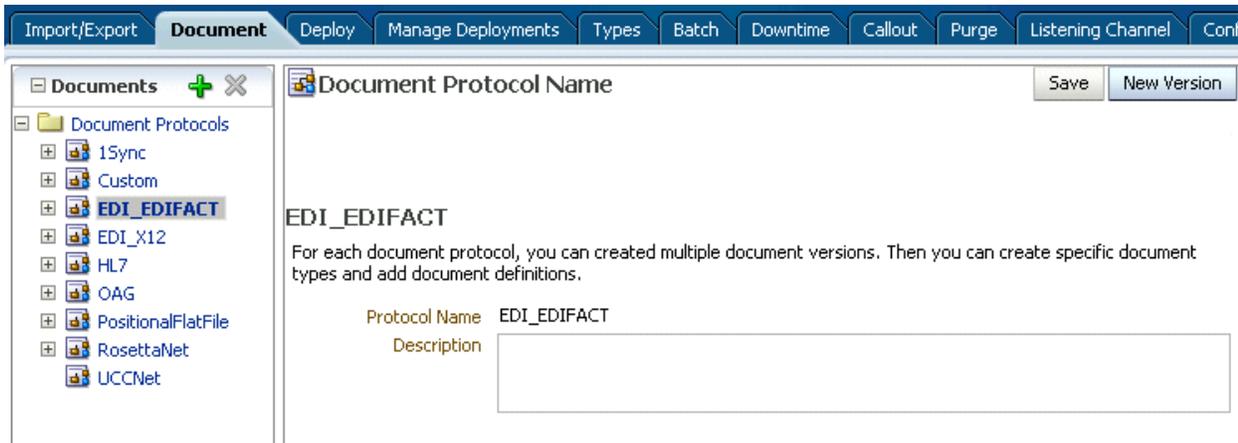
The image shows a section of the Oracle B2B screen displaying the Document, Deploy, Manage Deployments, Types, Import/Export, and Schedule Batch tabs. The Document tab is selected. The left pane is labeled Documents and shows an expanded hierarchy of Document Protocols with ED_EDIFACT selected. The right pane is labeled Document Protocol Name and shows the Protocol Name and Description fields for the EDI_EDIFACT document. The right pane also shows the Save and New Version buttons.

****

5.   Enter a version name, provide document version parameters as applicable, and click **Save**.

The version is used for document identification and can be case sensitive and use a fixed syntax, depending on the protocol.

Figure 4–7 shows the document protocol version page for an EDI EDIFACT D98A document.

Figure 4–7   Entering Document Protocol Parameter Information
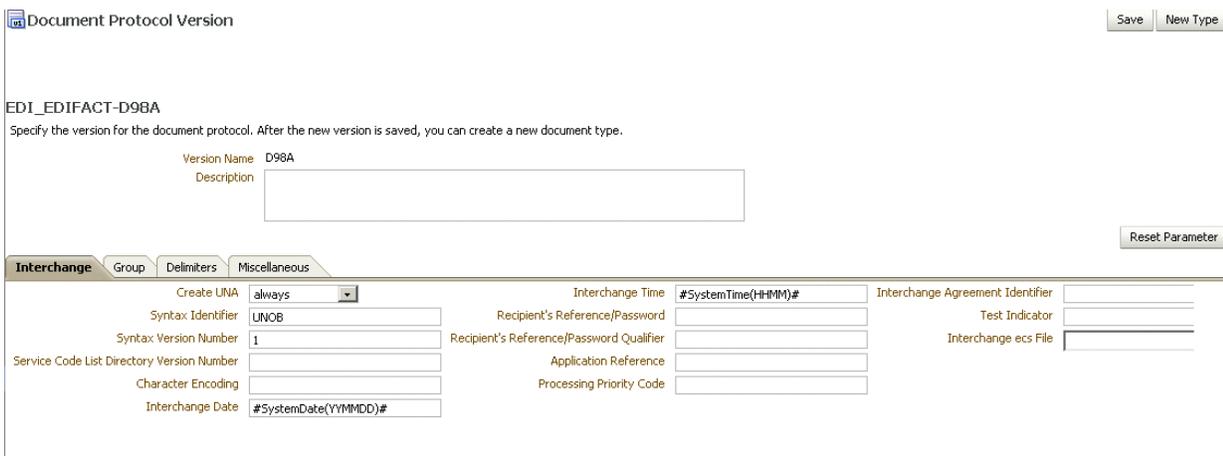
The image shows a section of the Oracle B2B screen displaying the Document tab. The left pane is labeled Documents and shows an expanded hierarchy of Document Protocols with ED_EDIFACT > NewVersion selected. The right pane is labeled Document Protocol Version and shows the Version Name and Description fields for

the EDI_EDIFACT - NewVersion document and the Save and New Version buttons. The bottom of the right pane also shows the Interchange, Group, and Delimiters tabs, with the Interchange tab selected.

****

For parameter descriptions, see the following:

- Table 8–4, " Document Version Parameters for an EDI EDIFACT Document"

- Table 8–8, " Document Version Parameters for an EDI X12 Document"

- Table 8–11, " Document Version Parameters for an HL7 Document"

6. With the new version name selected, click **New Type**.

7. Enter a document type name, provide document type parameters as applicable, and click **Save**.

   Figure 4–8 shows the document type parameters page for an EDI EDIFACT D98A document.

Figure 4–8   Entering Document Type Parameter Information

The image shows the EDI_EDIFACT-D98A-NewDocumentType section of the Document tab in Oracle B2B. The first area of the screen shows the Document Type Name and Document Type Description fields. The second area of the screen shows the Transaction tab and includes the following fields: Functional Group Identifier Code, Controlling Agency, Transaction Association Assigned Code, and Common Access Reference.

****

For parameter descriptions, see the following:

- Table 8–1, " Document Type Parameters for a Custom Document"

- Table 8–5, " Document Type Parameters for an EDI EDIFACT Document".

- Table 8–9, " Document Type Parameters for an EDI X12 Document"

- Table 8–12, " Document Type Parameters for an HL7 Document"

- Table 8–18, " Document Type Parameters for a RosettaNet Document"

8. With the new document type name selected, click **New Definition**.

9. Enter a document definition name and do the following:

- Browse for an optional definition (XSD) file for any of the document protocols.

- Browse for the required transaction set ECS file for the following protocols: EDI EDIFACT, EDI X12, HL7, and positional flat file.

- Provide document definition parameters as applicable.

  Figure 4–9 shows the document definition parameters page for an EDI EDIFACT D98A document.

Figure 4–9   Entering Document Definition Parameter Information
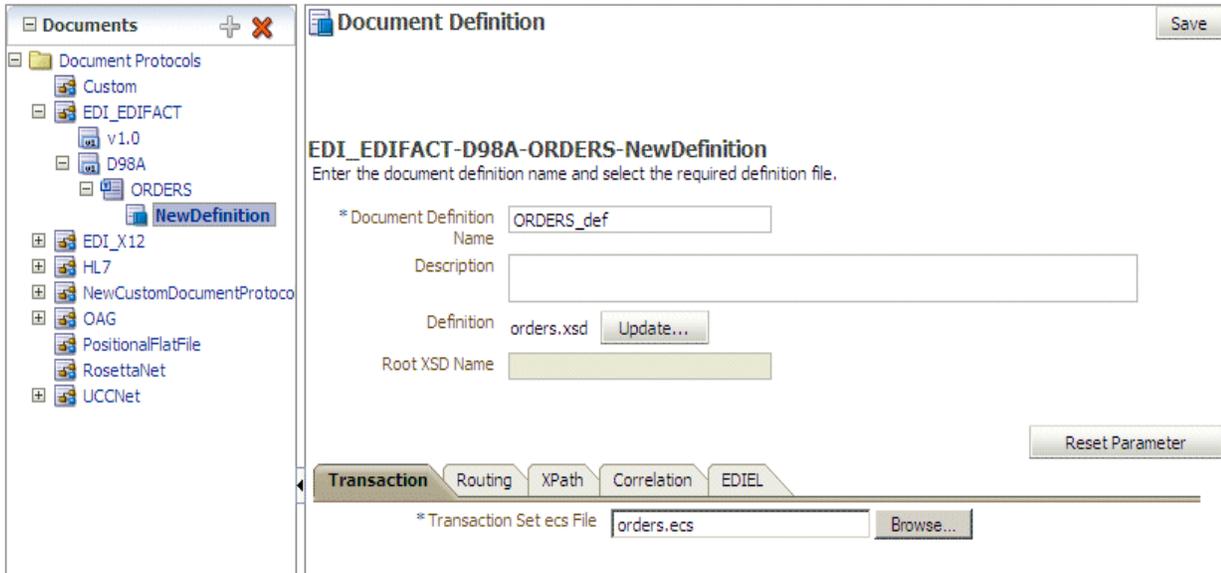
The image shows a section of the Oracle B2B screen displaying the Document tab. The left pane is labeled Documents and shows an expanded hierarchy of Document Protocols with ED_EDIFACT > D98A > ORDER3 > ORDERS_def selected. The right pane shows the ORDERS_def file and includes the following fields: Document Definition Name, Document Definition Description, and Definition. The bottom of the right pane also shows the Transaction and Routing tabs, with the Transaction tab selected. The Transaction tab shows the Transaction Set ecs File field.

****

For parameter descriptions, see the following:

- Table 8–2, " Document Definition Parameters for a Custom Document"

- Table 8–6, " Document Definition Parameters for an EDI EDIFACT Document"

- Table 8–10, " Document Definition Parameters for an EDI X12 Document"

- Table 8–13, " Document Definition Parameters for an HL7 Document"

- Table 8–15, " Document Definition Parameters for an OAG Document"

- Table 8–16, " Document Definition Parameters for a Positional Flat File"

- Table 8–19, " Document Definition Parameters for a RosettaNet Document"

- Table 8–21, " Document Definition Parameters for a UCCnet Document"

10. Click **Save**.

4.3 Deleting a Document Definition

To delete a document definition, first delete all agreements that use that document definition and then remove the supported document definitions from the host and all remote trading partners that reference the definition.

5

Configuring Trading Partners

This chapter describes how to configure trading partners by creating a trading partner profile (providing values for identifiers, contact information, trading partner parameters, and Key Store information); adding trading partner users; adding document definitions and assigning sender and receiver roles, and configuring channel details, including security.

The third step in the Oracle B2B process flow, shown in Figure 5–1, is to configure the trading partners.

Figure 5–1 Oracle B2B Process Flow
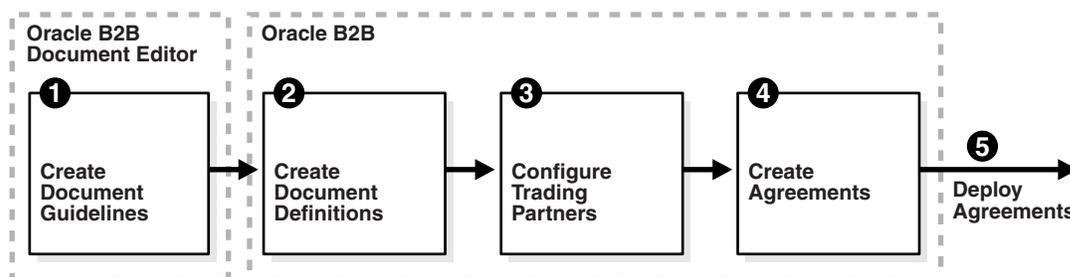
The image shows a step diagram linking the following blocks from left to right: 1. Oracle B2B Document Editor: Create Document Guidelines; 2. Oracle B2B: Create Document Definitions; 3. Oracle B2B: Configure Trading Partners; 4. Oracle B2B: Create Agreements; 5. Deploy Agreements.

****

This chapter contains the following topics:

- Section 5.1, "Introduction to Trading Partners"

- Section 5.2, "Creating Trading Partner Profiles"

- Section 5.3, "Adding Trading Partner Users"

- Section 5.4, "Adding Document Definitions"

- Section 5.5, "Configuring Channels"

- Section 5.6, "Using the Auto Create Agreement Feature"

- Section 5.7, "Using Identifiers for Trading Partner Lookup"

- Section 5.8, "Scheduling Trading Partner Downtime"

- Section 5.9, "Broadcasting Messages to Multiple Trading Partners"

- Section 5.10, "Validating Certificate in Inbound Message Processing"

5.1 Introduction to Trading Partners

In Oracle B2B, a transaction involves two trading partners, the host trading partner and a remote trading partner. The host trading partner is typically the organization where Oracle B2B is installed. The remote trading partner is the organization with whom the host trading partner conducts an e-business transaction. A trading partner can have host (back-end) applications, databases, or customers to involve in the transaction. Either the initiator of a transaction or the responder can be the host or the remote trading partner.

The host trading partner organization configures all the trading partners, host and remote. By using the trading partner users created for each remote trading partner by the host trading partner, remote partners can access their own data in Oracle B2B. Figure 5–2 shows the steps to configure a trading partner.

Figure 5–2   Configuring Trading Partners
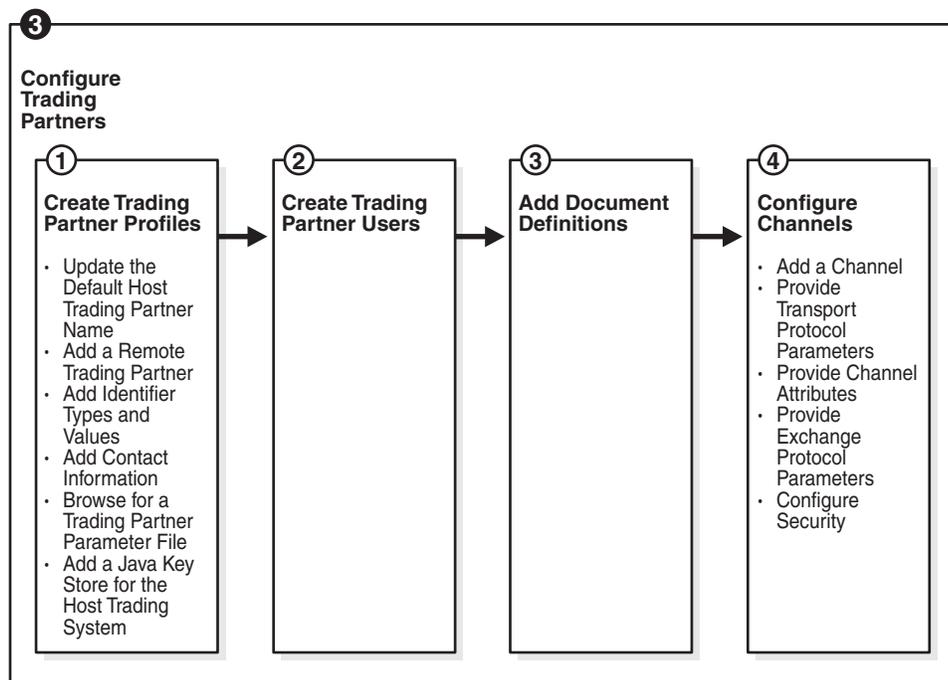
The image shows a step diagram linking the following blocks from left to right: 1. Create Trading Partner Profiles; 2. Create Trading Partner Users; 3. Add Document Definitions; 4. Configure Channels.

****

5.2 Creating Trading Partner Profiles

Oracle B2B supplies a default host trading partner name, **MyCompany**, which you update to reflect your enterprise. After you create one or more remote trading partners, use the cloning feature to create new trading partners that participate in similar transactions. Cloning copies the source trading partner's document definitions and delivery channels (except MLLP channels), but does not copy identifiers, contacts, and users. Renaming the delivery channel in the newly created trading partner is recommended.

After you create and configure a trading partner, the information is saved as a trading partner profile in Oracle Metadata Repository. Partner data can be exported to a ZIP file by using the **Export** button on the **Profile** tab.

To create a trading partner profile, do the following:

- Task 1, "Update the Default Host Trading Partner Name"

- Task 2, "Add a Remote Trading Partner"

- Task 3, "Add Identifier Types and Values"

- Task 4, "Add Contact Information"

- Task 5, "Add a Trading Partner Parameter and Value"

- Task 6, "Provide Key Store Information for the Host Trading Partner"

Task 1 Update the Default Host Trading Partner Name
Do this the first time you set up Oracle B2B.

1. Click the **Partners** link.

2. Click **MyCompany**.

3. Click **Edit**, as shown in Figure 5–3.

Figure 5–3 Editing the Host Trading Partner Profile
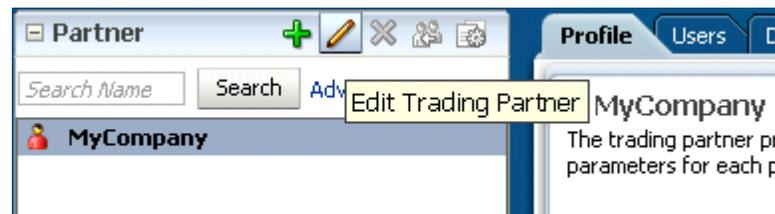
The image shows a section of the Partner screen for MyCompany. The cursor is hovering over the Edit icon.

****

4. Provide the host trading partner name and optional icon file, and click **OK**.

   The optional icon file must be a 16 x 16-pixel PNG file.

   The host trading partner name appears in the **Partner** list.

Task 2 Add a Remote Trading Partner
Do this for each remote trading partner.

1. Click the **Partners** link.

2. Click **Add**, as shown in Figure 5–4.

Figure 5–4   Adding a Remote Trading Partner
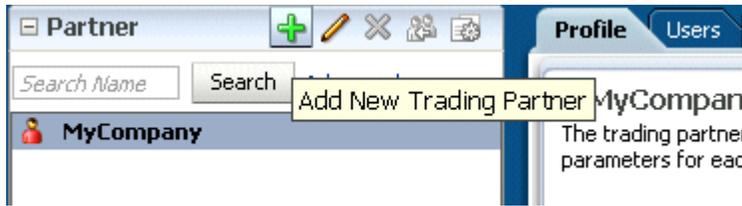
The image shows a section of the Partner screen with MyCompany selected in the list. The cursor is hovering over the Add New Trading Partner icon.

****

3.   Provide a partner name and click **OK**.

The remote trading partner name appears in the **Partner** list.

4.   (Optional) Click **Edit** to add a 16 x16-pixel PNG file as an icon for the remote trading partner, as shown in Figure 5–5, and click **OK**.

Figure 5–5   Editing a Remote Trading Partner Profile
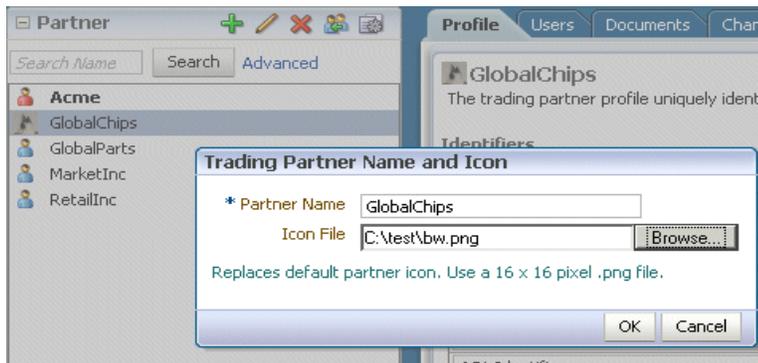
The image shows the Trading Partner Name and Icon dialog. Fields for the partner name and an icon file are displayed. A Browse button is used to browse for an icon file.

****

A variation on this task is to use the clone feature. If you have already created a trading partner that is similar to a trading partner you want to create, click the **Clone** icon, as shown in Figure 5–6, and provide the trading partner information that is not cloned: identifiers, contacts, and users.

Figure 5–6   Cloning a Remote Trading Partner
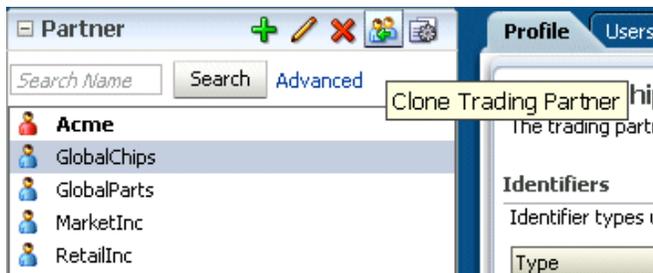
The image shows a section of the Partner screen with GlobalChips selected in the list. The cursor is hovering over the Clone Trading Partner icon.

****

> **Note:** Use the **Delete** icon to delete a remote trading partner. However, you cannot delete a remote trading partner that is part of a deployed trading partner agreement. You must first delete the agreement.

Task 3 Add Identifier Types and Values

Identifier types enable Oracle B2B to identify a trading partner at run time. In general, the identification process is to identify the partner, then the document, and then the partner-document pair identifies the agreement. Oracle B2B provides each trading partner with a default identifier type, **Name**, whose value is the name of the trading partner.

Add identifier types and values for both the host and remote trading partners. See Chapter 10, "Creating Types," for how to create the types that you add here.

1. Click the **Partners** link.

2. Click the **Profile** tab.

3. Select a trading partner.

4. In the **Identifiers** area, click **Add**, as shown in Figure 5–7.

Figure 5–7   Adding Identifier Types
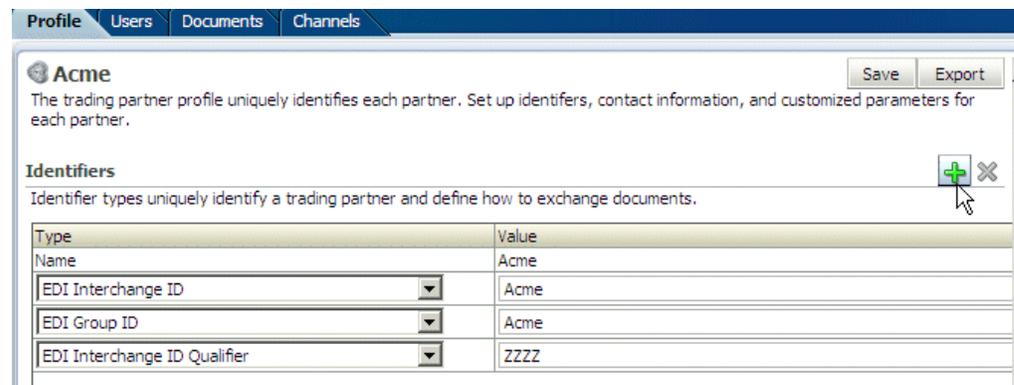
The image shows the following Oracle B2B tabs: Profile, Users, Documents, and Channels. The Profile tab is selected and lists the identifiers as a series of type-value pairs. There are plus and minus icons above and to the right of the identifiers. There are Save and Export buttons at the top of the tab on the right side.

****

5. From the **Type** list, select an identifier type.

   See Table 10–1, " Identifier Types Defined in Oracle B2B" for descriptions of the identifier types.

6. Provide a value.

7. Repeat Steps 4 through 6 as needed.

8. Click **Save**.

Task 4 Add Contact Information

To add optional contact information for a trading partner, use the preseeded types—Contact Name, Email, Fax, Phone. Or, you can create a contact type on the **Administration** > **Types** page. See Section 10.2, "Creating Custom Contact Information Types," for more information.

1. Click the **Partners** link.

2. Click the **Profile** tab.

3. In the **Contact Information** area, click **Add**.

4. Select from the list under **Type** and enter a value, as shown in Figure 5–8.

Figure 5–8   Adding Contact Information
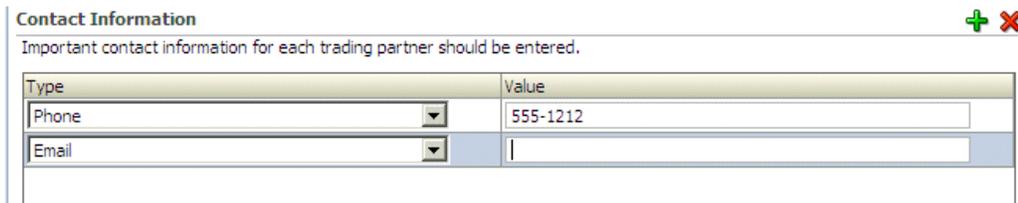
The image shows a section of the Profile tab displaying the Contact Information. The information appears as a series of type-value pairs. There are plus and minus icons above and to the right of the contact information.

****

5. Click **Save**.

Task 5 Add a Trading Partner Parameter and Value

Before adding an optional trading partner parameter and value for a trading partner, you must create the parameter on the **Administration** > **Types** page. (If you have not already created a parameter, the **Add** icon does not appear.) See Chapter 10, "Creating Types," for more information.

1. Click the **Partners** link.

2. Click the **Profile** tab.

3. In the **Parameters** area, click **Add**, as shown in Figure 5–9.

Figure 5–9   Adding Trading Partner Parameters and Values
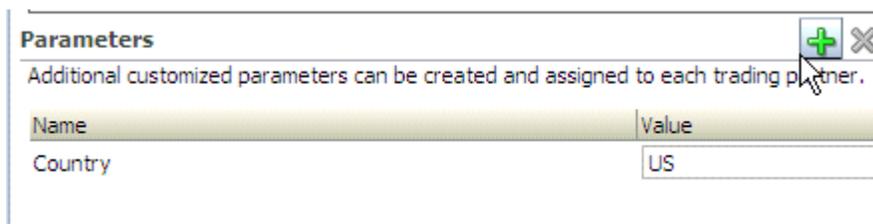
The image shows a section of the Profile tab displaying the Parameters. The information appears as a series of name-value pairs. There are plus and minus icons above and to the right of the parameters.

****

4. Select a parameter, as shown in Figure 5–10, and click **OK**.

Figure 5–10   Selecting Trading Partner Parameters
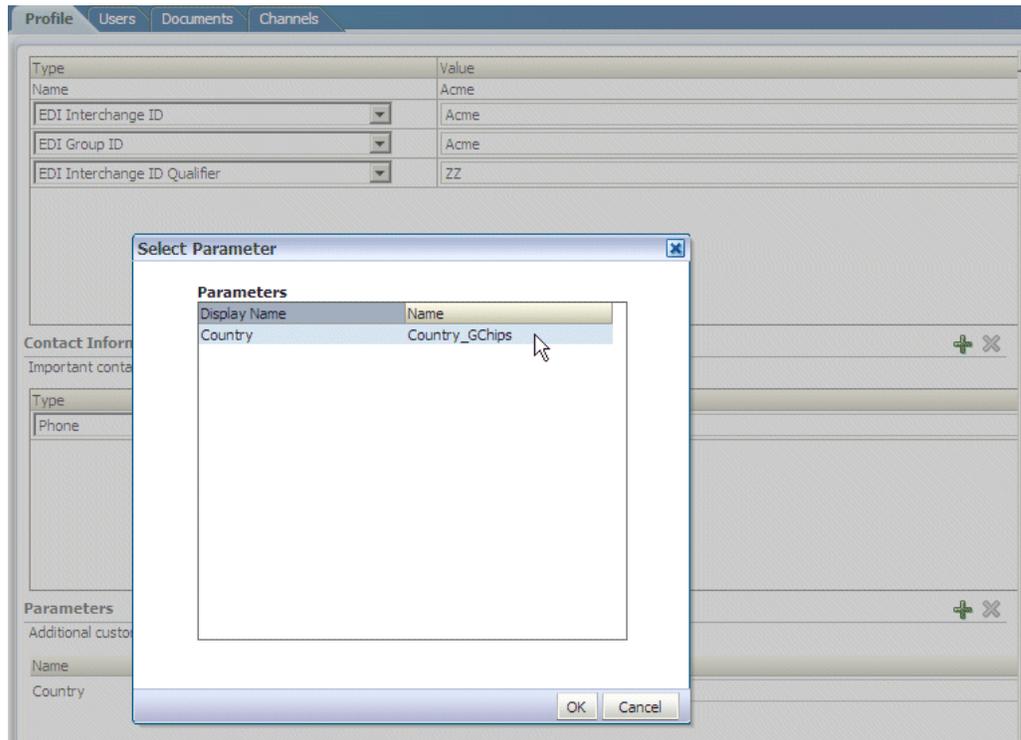
The image shows the Select Parameter dialog box displaying the list of parameters. Each parameter shows the Display Name and Name. The bottom right of the dialog box shows the OK and Cancel buttons.

****

5. Click **Save**.

You can also update values for a specific trading partner on this page.

Task 6   Provide Key Store Information for the Host Trading Partner

Add an optional Key Store password and location for host trading partner security. If a digital signature, encryption, or SSL are enabled, you must specify a Key Store location. See Task 5, "Configure Security" for where you specify digital signatures and encryption, and Table 5–6 for descriptions of security parameters.

You can choose any Key Store for Oracle B2B. If you are using SSL, using the same Key Store for both B2B and Oracle WebLogic Server SSL configuration is recommended to avoid SSL-related problems when exchanging messages with trading partners.

1. Click the **Partners** link.

2. Click the **Profile** tab.

3. Select the host trading partner.

4. In the **Key Store** section, provide a password and location, as shown in Figure 5–11.

Figure 5–11 Entering Key Store Information

The image shows a section of the Profile tab displaying the Key Store information. The section shows the following fields: Password, Confirm Password, and Location.

****

5. Click **Save**.

Note: If you re-enter a Key Store password that you previously entered incorrectly (which produces errors trying to connect to the Key Store), then you must restart the server after entering the correct password.

5.3 Adding Trading Partner Users

The host trading partner administrator (the default login username-password combination) can add additional host and remote trading partner users. These users can log in to Oracle B2B and access their own trading partner data only.

The following roles are available:

- Administrator role—Provides access to all Oracle B2B functionality

- Monitor role—Provides access to reporting and metrics functionality only (use of the **Reports** and **Metrics** links)

Users with the administrator role can access all B2B functions for their trading partner data only. No data for other trading partners is displayed. Users with the monitor role can access report functionality for their trading partner data only. No other links and no data for other trading partners are displayed. Oracle B2B also supports restricting access based on document type. See Section 1.4.2, "Restricting Access to Document Types," for more information.

To add users, do the following:

- Task 1, "Create a New User in the Identity Store"

- Task 2, "Add a User in the Oracle B2B Interface"

- Task 3, "Add Document Types That the User Has Permission to Access"

Task 1 Create a New User in the Identity Store

A user must exist in the Identity Store before you can provision the user in Oracle B2B. Although there are many tools that you can use to create users, one way is to use the **Security Realms** function in Oracle WebLogic Server Administration Console, as shown in Figure 5–12.

Figure 5–12   Oracle WebLogic Server Administration Console—Security Realms
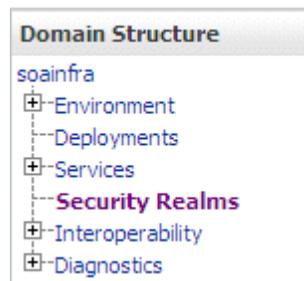
The image shows the Domain Structure pane, displaying the hierarchy of information related to the soainfra domain including the following: Environment, Deployments, Services, Security Realms, Interoperability, and Diagnostics.

****

> **Note:**   It is recommended that you use an LDAP-based policy store because the default file-based policy store is not supported if Oracle B2B is running on a managed server.

Then, within the **myrealm** settings, the **Users and Groups** tab displays a table of all users in your realm. Click **New**, and then add a user and user password on the page shown in Figure 5–13.

Figure 5–13   Oracle WebLogic Server Administration Console—Adding a New User

The image shows the Create a New User pane. The User Properties area shows the following fields: Name, Description, Provider, Password, and Confirm Password. Above and below the User Properties area are the OK and Cancel buttons.

****

Task 2 Add a User in the Oracle B2B Interface

The default administrator can add users. Host administrators and remote administrators can add users (remote administrators for their own data only) if they have been granted that permission by the default administrator.

1. Click the **Partners** link.

2. Click the **Users** tab.

3. Select a trading partner.

4. Click **Add**.

5. Provide the user name created in Task 1 and click **Search**.

Enter the user name exactly is it was created.

6. Select the **Monitor** or **Administrator** role, shown in Figure 5–14, and click **OK**.

Figure 5–14   Assigning the Monitor or Administrator Role

The image shows the Users tab and Register User dialog box. The first area of the dialog box shows the Username field with a Search button to the right. The second area of the dialog box shows the user Details and includes the following fields: Name, Phone, Email, and Role (either Monitor or Administrator). The final area of the dialog box shows the OK and Cancel buttons.

****

Task 3   Add Document Types That the User Has Permission to Access

The default administrator can add document types for a user. Host administrators and remote administrators can add document types for a user (remote administrators for their own data only) if they have been granted that permission by the default administrator. If no document types are added here, then the user has access to *all* document types.

1. In the **Supported Document Types** area, shown near the bottom of Figure 5–15, click **Add**.

Figure 5–15   Adding Document Types

The image shows the Users tab, where multiple users can be added for the trading partner Acme. A user named b2badmin1 with the role Administrator is displayed. In the Supported Document Types area, no document type names are listed.

****

2. Select a document type and click **Add**, as shown in Figure 5–16.

Figure 5–16   Selecting a Document Type
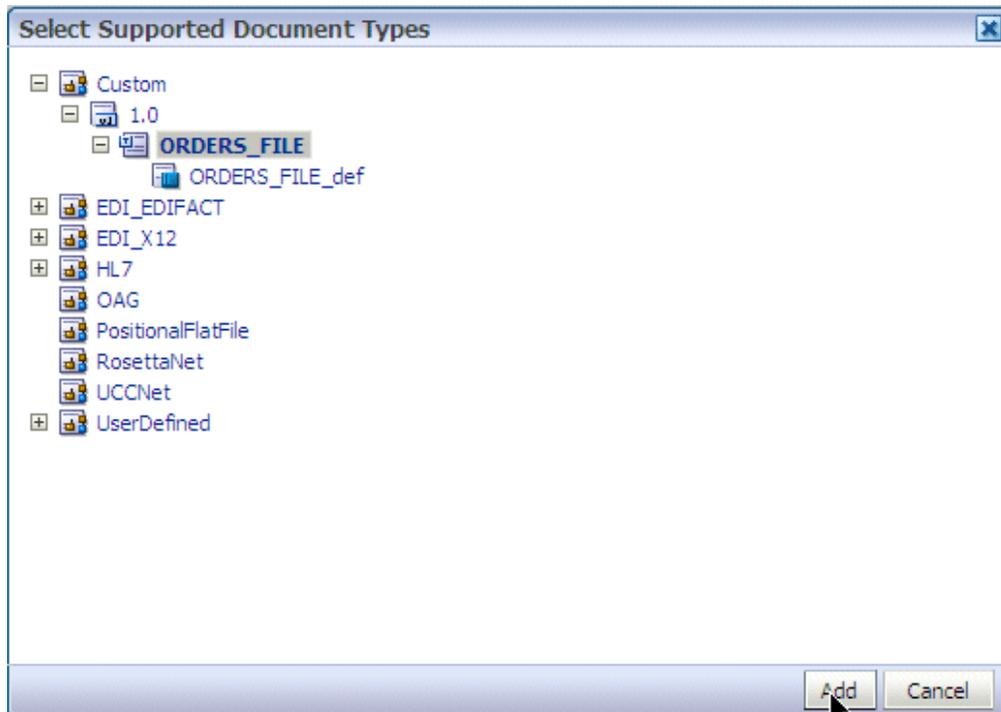
The image shows the Select Supported Document Types dialog, The following document types are displayed in a tree structure: Custom > 1.0 > ORDERS_FILE > ORDERS_FILE_def; EDI_EDIFACT; EDI_X12; HL7; OAG; PositionalFlatFile; RosettaNet; UCCNet; UserDefined.

****

3. Repeat Steps 1 and 2 as needed.

The document types that the user has permission to access are displayed in the **Document Type Names** column.

The document types in the **Document Type Names** column can also be deleted. If all types in the list are deleted, then the user has access to all document types.

5.4 Adding Document Definitions

The Oracle B2B host administrator creates all document definitions, which are automatically assigned to the host trading partner. The host administrator can assign any document definition to a remote trading partner. For both the host and remote trading partners, the sender and receiver for each document must be identified.

For information on updating a document definition after it has been added, see Section 8.9, "Changing Document Details."

> **Note:** Document definitions that are automatically associated with the host trading partner must be deleted from the host trading partner profile (and also from the remote trading partner profile) before you can delete a document definition (from **Administration** > **Document**).

Consider the scenario in which Acme (buyer) sends a purchase order to GlobalChips. As part of this transaction, Acme also receives an acknowledgment that GlobalChips (seller) received the purchase order. Therefore, this EDIFACT transaction uses two document definitions, one for the purchase order and one for the functional acknowledgment. GlobalChips receives the purchase order and also sends the acknowledgment.

For information on creating a document definition—required before you can add it to the trading partner profile—see Chapter 4, "Creating Document Definitions."

To add document definitions, do the following:

- Task 1, "Add Document Definitions"

Task 1 Add Document Definitions

Add document definitions to both host and remote trading partner profiles. You can also change document type parameters and document version parameters for the remote trading partner on this page. See Chapter 8, "Using Document Protocols," for more information.

1. Click the **Partners** link.

2. Click the **Documents** tab.

3. Select a trading partner.

4. Click **Add**.

5. Expand the nodes, select a document definition as shown in Figure 5–17, and click **Add**.

Figure 5–17   Selecting a Document Definition
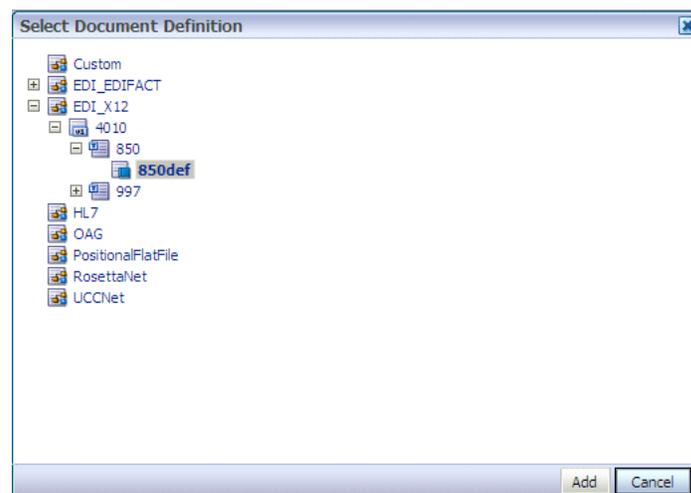
The image shows the Select Document Definition dialog. EDI_X12 is expanded as follows: 4010 > 850 > 850def, which is selected. The Add and Cancel buttons show at the bottom of the pane.

****

6. For each document listed, identify if the selected partner is the sender or receiver or both, as shown in Figure 5–18.

Figure 5–18   Selecting the Sender and Receiver
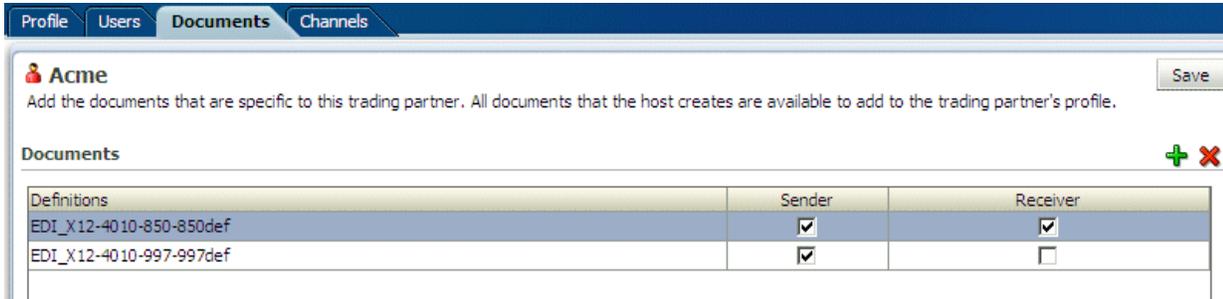
The image shows the following tabs: Profile, Users, Documents, Channels. The Documents tab is selected. The tab shows the documents in a table with the following columns: Definitions, Sender, and Receiver. The Sender and Receiver columns include check boxes.

****

7. Click **Save**.

See Section 8.9, "Changing Document Details," for how to change document protocol versions and document type parameters for a remote trading partner, including using the **Override Version Param** and **Override DocType Param** parameters.

5.5 Configuring Channels

A channel defines how a message is delivered. It specifies trading partner security characteristics, the transport protocol, the exchange protocol, any exchange protocol override elements, and, if defined, support for digital envelopes, encryption credentials, digital signatures, signing credentials, and validation.

When you configure an external delivery channel for the host trading partner, it is available for all remote trading partners when you create agreements. This avoids having to create a delivery channel multiple times, once for each remote trading partner. When you configure an external delivery channel for a remote trading partner, it is available for only that remote trading partner when you create agreements. When you configure an internal delivery channel for the host trading partner—for inbound messages to Oracle B2B using the AQ, File, or JMS transports—the channel is available for only the host trading partner when you create inbound agreements.

You can also create custom JMS exception queues by configuring JMS internal delivery channels for the host trading partner. See the following for more information:

■ Table 17–1, " Configuration Settings"

■ Section G.4, "Using a Custom Exception Queue for Error Message Delivery"

Table 5–1 lists the channels/exchange protocols available in Oracle B2B.

Table 5–1   Channels/Exchange Protocols Available in Oracle B2B

Protocol	Description
AS2-1.1	Applicability Statement 2, version 1.1—specification for using EDI over the Internet. AS2 provides S/MIME support over HTTP or HTTPS. AS2 also works with non-EDI document types such as `.xml`, `.txt`, `.doc`, and `.xls`. AS2 is also called EDI over the Internet, or EDIINT AS2.
MLLP-1.0 (Remote trading partner only)	Minimum Lower Layer Protocol (MLLP) is a minimalistic OSI-session layer framing protocol. MLLP (and the TCP transport protocol) are available for remote trading partners only. It is used with HL7 or Custom documents. With MLLP, the same channel can be used for sending or receiving messages, and can be configured as either the server or the client. MLLP connections can be permanent or transient: Features of a permanent connection: ■ Caches the socket based on the endpoint. ■ Only one socket per endpoint is created. ■ The socket is reused for future messages. Features of a transient connection: ■ A new socket is created for each message. ■ A message is sent and the listener waits for the acknowledgment. ■ When the acknowledgment is received, the socket is closed. See Section 5.5.1, "About MLLP" for more information.
ebMS-2.0 ebMS-1.0	Electronic business Extensible Markup Language (ebXML) Messaging Service (ebMS)—specification used to exchange XML documents. ebMS is built on a SOAP Web services message format. Oracle B2B supports ebMS 1.0 and 2.0 and uses the HTTP, HTTPS, and Email transport protocols and the SOAP packaging protocol. The ebMS protocol supports correlation between documents. Oracle B2B also supports XMLDSig, XML Encrypt, and gZip-based compression for large documents.
RosettaNet-V02.00	RosettaNet 2.0 does not include the proprietary aspects of RosettaNet 1.1, and adds support for multiple transfer protocols, hub-based routing, attachments, payload encryption, and more.
RosettaNet-01.10	Implementation guidelines for creating software applications that provide for the reliable transport of PIPs in XML-format business documents between trading partners. Guidelines are provided for transport, routing, packaging, security, signals, and trading partner agreements. RosettaNet specifies the envelope or container format that remains constant when exchanging business documents (the payloads), whereas the document exchange choreography and the XML schemas vary based on which PIP and document type are used. The RosettaNet envelope format is also independent of the specific transfer protocol you use.
AS1-1.0	Applicability Statement 1—specification for using EDI over SMTP. AS1 also works with non-EDI document types such as XML and TXT files.
Generic File-1.0	Transport by which messages are sent to or received from a file in a local file system.
Generic AQ-1.0	Transport by which messages are sent to or received from Oracle AQ single or multiconsumer queues.
Generic FTP-1.0	Transport by which messages are sent to or received from a file at a remote FTP server.
Generic SFTP-1.0	Transport by which messages are sent to or received from a file at a remote SFTP server.

Table 5–1   (Cont.) Channels/Exchange Protocols Available in Oracle B2B

Protocol	Description
Generic JMS-1.0	Transport by which messages are sent to or received from a JMS queue or topic. If a user name and password are not provided, the local JNDI is used, including in a clustered environment, provided that the destinations are distributed.
	Oracle B2B does not support `javax.jms.ObjectMessage`.
Generic HTTP-1.0	Transport by which messages are sent to or received from a Web server.
Generic Email-1.0	Transport by which messages are sent to or received from an e-mail server.
Generic TCP	Transport that supports customized Exchange Protocol parameters such as Start Block, End Block, Header Length, Message Length Index, and Retain Header.

Depending on the channel/transport protocol selected, you provide channel details such as transport protocol parameters, channel attributes, exchange protocol parameters, and security parameters, as shown in Figure 5–19.

Figure 5–19   Channel Details Tabs
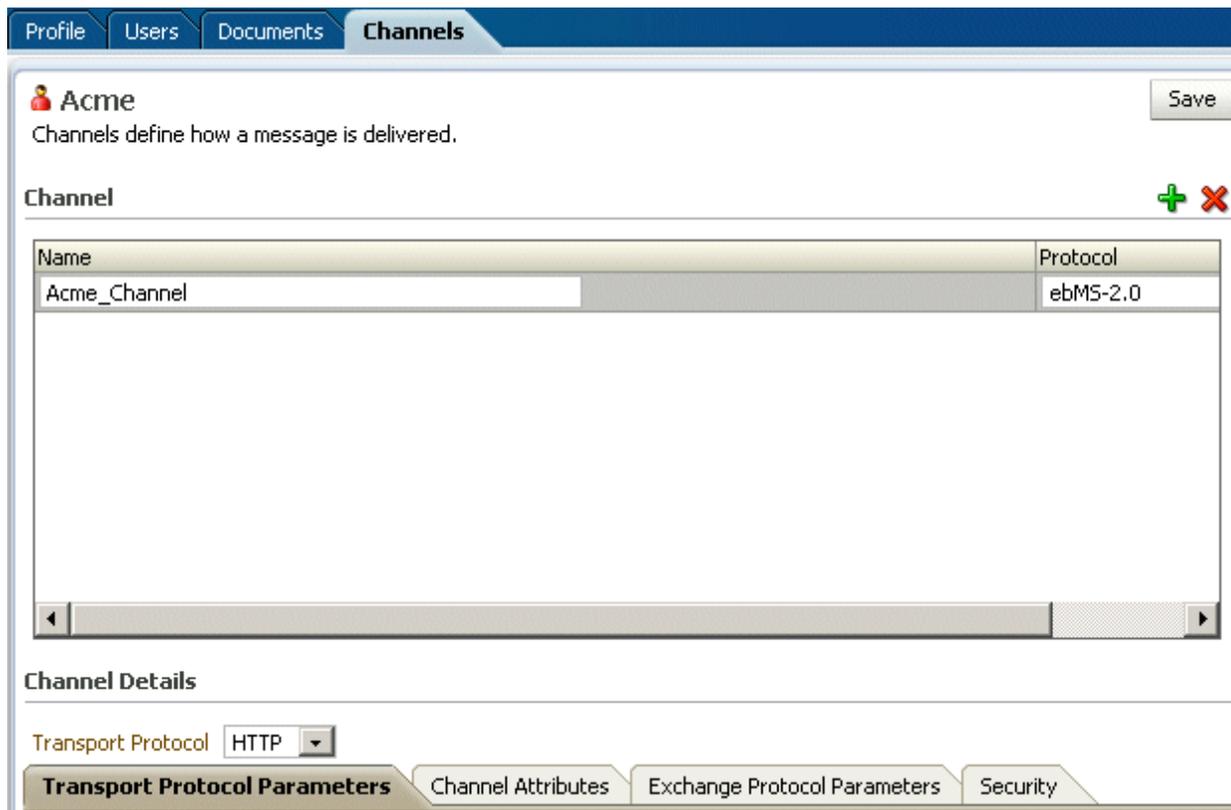
The image shows the Channels tab. The channel details tabs that are displayed are Transport Protocol Parameters, Channel Attributes, Exchange Protocol Parameters, and Security.

****

The transport protocol parameters define the properties specific to a given use of a protocol endpoint. The transport is responsible for message delivery using the selected transport protocol, mode (synchronous or asynchronous), server, and protocol

endpoint address (trading partner address, such as a URI). Table 5–3 describes the transport protocol parameters.

The channel attributes define the communication interface between the host trading partner's host application and its installation. Table 5–4 describes the channel attributes.

The exchange protocol parameters define the headers, acknowledgments, and packaging that puts the headers and payload together (the message exchange mechanism). Table 5–5 describes the exchange protocol parameters.

Security parameters define features such as signing, encryption, and digital signatures. Message encryption using an AES setting is preferable, where available. Table 5–6 describes the security parameters.

Note the following:

- No security parameters are specified for the Generic protocols—Generic File-1.0, Generic AQ-1.0, Generic FTP-1.0, Generic SFTP-1.0, Generic JMS-1.0, Generic HTTP-1.0, and Generic Email-1.0.

- Security parameters do not apply to the MLLP channel.

To configure a channel for a trading partner, do the following:

- Task 1, "Add a Channel"

- Task 2, "Provide Transport Protocol Parameters"

- Task 3, "Provide Channel Attributes"

- Task 4, "Provide Exchange Protocol Parameters"

- Task 5, "Configure Security"

Task 1 Add a Channel

Add a channel for the responder in a B2B transaction.

1. Click the **Partners** link.

2. Select a trading partner.

3. Click the **Channels** tab.

4. Click **Add**.

5. Enter a channel name.

6. Select a protocol, as described in Table 5–1.

   Figure 5–20 shows the list of protocols.

Figure 5–20   Selecting a Protocol
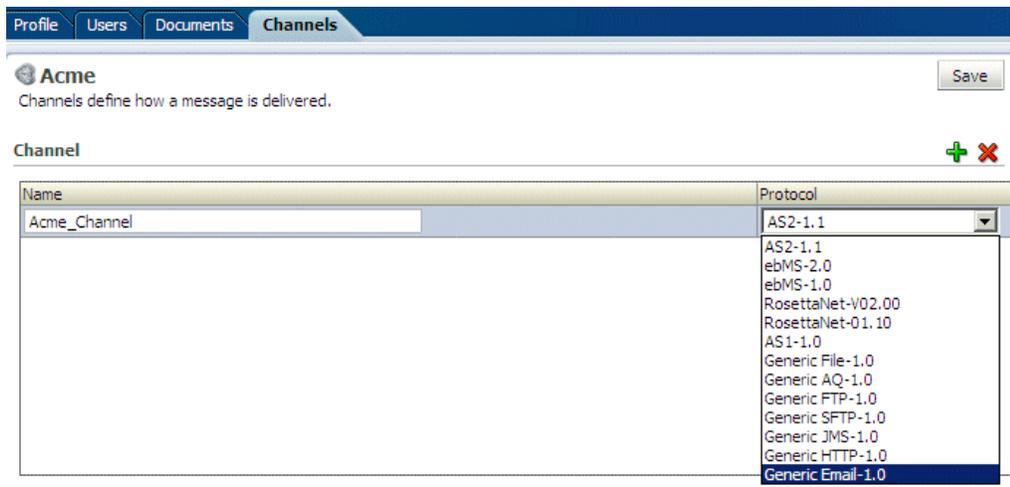
The image shows the following tabs: Profile, Users, Documents, Channels. The Channels tab is selected. The tab shows the channels in a table with the following columns: Name and Protocol. The Protocol column includes a drop-down list. There are plus and minus icons above and to the right of the channels. The Save appears at the top of the tab on the right side.

****

7. Click **Save**.

Based on the delivery channel protocol you selected in Step 6, the applicable protocol is displayed in the **Transport Protocol** field, as shown in Table 5–2.

Table 5–2   Delivery Channels and Transport Protocols

Channel Protocol Selected...	Transport Protocol Displayed...
AS2-1.1	HTTP
ebMS-2.0, ebMS-1.0	
RosettaNet-V02.00, RosettaNet-01.00	
Generic HTTP-1.0	
AS1-1.0	Email
Generic Email-1.0	
MLLP-1.0	TCP
Generic File-1.0	File
Generic AQ-1.0	AQ
Generic FTP-1.0	FTP
Generic SFTP-1.0	SFTP
Generic JMS-1.0	JMS

Task 2 Provide Transport Protocol Parameters

1. Click the **Transport Protocol Parameters** tab.

2. Provide transport protocol parameters, as described in Table 5–3, depending on the channel/transport protocols selected in Task 1.

Table 5–3 describes the transport protocol parameters (listed in alphabetical order) and the protocols to which the parameters apply.

Table 5–3   Transport Protocol Parameters

Parameter	Description	Protocol Used With
Additional transport headers	The custom HTTP headers used to send messages to a trading partner For the sync response process, additional transport headers must be set. For sender to treat the response as an inbound message, add `syncresponse=true` as part of Additional Transport Header.	AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic HTTP (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)
Archival Directory	B2B channels move the processed files to this directory. By default, it is a destructive read—processed files are deleted from the endpoint. In this case, files are moved to the path provided.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Cache Connections	If enabled, file listing and processing of the file occur in the same session (contrary to the default, in which listing and processing occur in different sessions).	Generic FTP-1.0 (optional)
Channel mask	To enable SSL for FTP, enter one of the following: ■ `Control`—Encrypts the control channel ■ `Data`—Encrypts the data channel ■ `Both`—Encrypts both the data and control channels The default is **None** (no SSL).	Generic FTP (optional)
Cipher suites	Provide the preferred cipher for encryption.	Generic FTP (optional)
Connection factory	The JNDI location or Java class name for the connection factory, as in `jms/b2b/B2BQueueConnectionFactory`.	Generic JMS (optional)
Connection Mode	Select from Client or Server.	MLLP-1.0 (required; for remote trading partners only) Generic TCP (required)
Consumer	The client that receives the message.	Generic AQ (optional)
Content type	The content type of the payload being sent over e-mail. The default content type is `text/plain`; other examples include `application/xml` and `application/edi`. This value is used only for the delivery channel (to send e-mail) and not for the listening channel. On the listening channel side, intelligence is built into the transport adapter to deal with different content types, so no configuration is required.	AS1 (optional) Generic Email (optional)
Control port	Provide a value to change the default FTP port value (21)	Generic FTP (optional)
Data port	For active FTP connections, use this option to configure the static/fixed data port of the FTP server.	Generic FTP (optional)
Datasource	The JNDI name of the JDBC data source to access AQ queues.	Generic AQ (optional)
Destination name	The JMS destination name	Generic JMS (optional)
Destination Provider	Enables B2B to connect to JMS queues or topics available on remote servers. JNDI properties required to connected to the target server are expected as the value. Use `;` (semicolon) as the separator for each key/value pair.	Generic JMS-1.0 (optional)
Directory name format	Directory name format to receive	Generic FTP (optional)

Table 5–3   (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Email ID	The e-mail address to which messages are delivered (similar to specifying the path for a file channel or queues in AQ or JMS).	AS1 (required) Generic Email (required)
Email Server	Select **IMAP** or **POP3**.	AS1 (required) Generic Email (required)
Enable CCC	Enables B2B to authenticate in an SSL session and do the rest of the file transfer commands on a plain socket.	Generic FTP-1.0 (optional)
Enable Marker	If enabled, creates a zero-byte file with the same name as the source, indicating completion of reading or writing. The file carries the same name as the source, but with the extension `marker`.	Generic File-1.0 (optional) Generic FTP-1.0 (optional)-1.0 Generic SFTP-1.0 (optional)
Encoding	The encoding used in B2B to convert the contents of the inbound files.	Generic FTP (optional)

Filename format[1]	The following file name formats can be used:
	
	`%FROM_PARTY%`
	`%TO_PARTY%`
	`%DOCTYPE_NAME%`
	`%DOCTYPE_REVISION%`
	`%MSG_ID%`
	`%TIMESTAMP%`
	
	The following file name format can be used for ebMS documents only:
	
	`%ACTIONNAME%`
	
	These file name formats can be used in any combination; for example,
	
	`%TO_PARTY%_%DOCTYPE_NAME%_%DOCTYPE_REVISION%.dat`
	
	produces something like `Acme_4010_850.dat`. Any file extension is allowed.
	
	See footnote below.
		Generic File (optional)
		
		Generic FTP (optional)
		
		Generic SFTP (optional)

Filename Separator	File name format separator to separate the keys. If a character other than _ (underscore) is used as the filename separator, then specify the value in the Filename Separator field. The Filename Separator parameter works with the Filename format parameter, and you should use the same separator value in the Filename format and Filename Separator parameters. For example if using Filename format `%TO_PARTY%_%DOCTYPE_NAME%_%DOCTYPE_REVISION%.dat`, then the Filename Separator should be _ (underscore). The $ and ^ characters cannot be used as filename separators Default value is _ (underscore)	Generic File (optional) Generic FTP (optional) Generic SFTP (optional)
Folder	An absolute directory path is recommended.	AS1 (optional) Generic Email (optional)
Folder name	An absolute directory path is recommended.	Generic File (required) Generic FTP (required)

Table 5–3   (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Host name	The trading partner's transport or e-mail server exchanging messages. For the MLLP 1.0 protocol, if the connection mode is set to Server, then the host name must be the B2B server. If the connection mode is set to Client, then the host name must be the remote B2B server (MLLP server).	AS1 (required) Generic AQ (optional) Generic FTP (required) MLLP-1.0 (required; for remote trading partners only) Generic SFTP (required) Generic Email (required) Generic TCP (required)
Is Map Payload Alone	Indicates that the JMS map message contains only the payload	Generic JMS (optional)
Is topic	Select to indicate that JMS is communicating with a topic (not a queue).	Generic JMS (optional)
Is Van Mailbox	If enabled, B2B treats the endpoint as a VAN Mailbox and operates accordingly.	Generic FTP-1.0 (optional)
Message type	Select a JMS message type: **BYTES**, **TEXT**, or **MAP**.	Generic JMS (optional)
Minimum Age	Files arriving at the endpoint are processed after the time interval entered, in milliseconds.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Pass phrase and Confirm pass phrase	If you enter a private key file location, and if the private key file is pass-phrase protected, then enter the pass phrase.	Generic SFTP (optional)
Password and Confirm Password	To use password authentication, provide a Key Store password, which is used for HTTP basic authentication.	AS1 (optional) AS2 (optional) Generic AQ (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic FTP (optional) Generic HTTP (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)
Path	The absolute directory path where messages are sent from or received.	Generic SFTP (required)
Permanent Connection	When set to false (the default value), a message is sent on a new connection and the connection is closed after the ACK is received. As a receiver of the message, the connection is closed after the ACK is sent back to the trading partner. When set to true, a cached connection is used to exchange all the messages.	MLLP-1.0 (optional; for remote trading partners only) Generic TCP (optional)

Table 5–3   (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Polling interval	The time interval in seconds during which Oracle B2B polls the server for inbound messages.	AS1 (optional) Generic File (not available) Generic AQ (optional) Generic FTP (not available) MLLP-1.0 (optional; for remote trading partners only) Generic SFTP (not available) Generic JMS (optional) Generic Email (not available) Generic TCP (optional)
Port number (or Port)	AQ runs on default port 1521. SFTP runs on default port 22, which can be changed to another port. FTP runs on default port 21, which is not displayed. See the description of **Control Port** for how to change this port number. For the MLLP 1.0 protocol, if the connection mode is set to Server, then the port must be a valid TCP port number. If the connection mode is set to Client, then the port must be the same as the port used on the MLLP server.	Generic AQ (optional) MLLP-1.0 (required; for remote trading partners only) Generic SFTP (required) Generic TCP (required)
Preserve Filename	Retains the file name.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Private key	To use public key authentication, provide the private key file location. You may also need to provide a pass phrase if the private key file is pass-phrase protected.	Generic SFTP (optional)
Queue name	The AQ queue name	Generic AQ (optional)
Recipient	The value used when delivering a message to the AQ queue. For example, if you set the recipient to `testuser`, then the message can be consumed only by the consumer with the name `testuser` (in other words, the recipient is on the sending side and the consumer is on the listening side).	Generic AQ (optional)
Send as attachment	If enabled, the message (payload) is sent as an e-mail attachment instead of the typical delivery in which the payload is the message body.	AS1 (optional) Generic Email (optional)
Sequence (Sequencing)	Enable this property when delivering the incoming EDI, HL7, or Custom message in sequence to the back-end application is required. See Section 5.5.2, "Message Sequencing" for more information.	MLLP-1.0 (optional; for remote trading partners only) Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP (optional) Generic TCP (optional)
SID	System ID to identify an Oracle database	Generic AQ (optional)
Subject	The subject header of the e-mail message	AS1 (optional) Generic Email (optional)
Subscriber ID	The JMS subscriber ID is required if JMS is communicating with a topic.	Generic JMS

Table 5–3   (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Timeout	Defines how long a transient MLLP connection keeps the socket open for the acknowledgment message. The default timeout value is 300 seconds. This parameter applies only to a transient MLLP connection (not to a permanent connection). Timeout can be configured as additional transport header at HTTP delivery channel. Example value: `timeout=30` (timeout value is in seconds)	MLLP-1.0 (optional; for remote trading partners only) Generic TCP (optional)

Timestamp Format
 	
About the timestamp format, timestamp offset, and timestamp source parameters:
 	
When the minimum age of the parameter receiver file has a value greater than zero, B2B ensures that the file has not been modified before picking up the file for processing. In other words, the system timestamp of the FTP server may differ from the system timestamp of the B2B server. The timestamp format, timestamp offset, and timestamp source parameters are used to get the last modified time for the receiver file. (See also the Minimum Age parameter in this table.)
	The receiver read-ordered timestamp format specifies how to parse the timestamp from the response string that is obtained using the FTP command. Index information is also included. For example,
	
	[43,55,'`MMM dd HH:mm`',ucy] - In the response string from the FTP command, the timestamp is present between index 43 and 55.
	
	[4,+14,'`yyyyMMddHHmmss`'] - In the response string from the FTP command, the timestamp is present between index 4 and index (4+4=18).
	
	You can specify multiple format masks, each enclosed by square brackets, as follows:
	
	`format-mask = start "," end "," "'"`
	`time-pattern "'" ["," current-year]`
	
	where
	
	`start = integer`
	`end   = integer`
	`time-pattern =`
	`java.text.SimpleDateFormat | "N"`
	`current-year  = "CY"`
	`TimestampFormats = {"[" format-mask "]"`
	`}+`
	
	For example, [41,53,'`MMM dd HH:mm`',CY][41,53,'`MMM dd yyyy`']
	
	`CY` must be added for time patterns in which a year is not present. The adapter then internally appends the format `yyyy` to the format mask you provide and appends the current year to the actual timestamp string before parsing it.
	
	The `start` and `end` indexes demarcate the timestamp substring of the string, originating either from the FTP `LIST` command or the actual file name. For a given FTP server, you may need to manually experiment with the server to determine which format masks to use.
	
	If `time-pattern` is specified as N, the substring is treated as a regular integral number. In terms of time (`t`), the number (`N`) is interpreted as follows:
	
	`t = "12:00 am, January 1, 1970 UTC" + N`
	`milliseconds`
		Generic FTP-1.0 (optional)

Table 5–3   (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Timestamp Offset	The receiver read-ordered timestamp offset is used to convert the timestamp of the file with respect to the system time where the B2B Server is running. For example, `-25200000`, subtracts 2520 milliseconds from the timestamp obtained from the FTP command to make it compatible with the B2B system time. Note: ■ Even if B2B and FTP are on the same machine, you may need an offset if the timestamp from the FTP command is in a different time zone. ■ Turn on the transport log to see how what timestamp the FTP command returns and then set values appropriately.	Generic FTP-1.0 (optional)
Timestamp Source	The receiver read-ordered timestamp source specifies the format used to get the timestamp, as follows: `LISTTIME` - The last modified time in the format '`MMM dd HH:mm`',`ucy` `TIMESTAMP` - The last modified time in the format '`yyyyMMddHHmmss`'	Generic FTP-1.0 (optional)
Transfer Type	Select **binary** or **ascii** for the file transfer mode.	Generic FTP-1.0 (optional)
URL	The HTTP or HTTPS endpoint URL of the trading partner.	AS2 (required) ebMS-2.0 (required) ebMS-1.0 (required) Generic HTTP (required) RosettaNet-V02.00 (required) RosettaNet-01.10 (required)
Use JMS ID	Uses the JMS message ID as the B2B message ID. This facilitates correlation at the JMS level.	Generic JMS-1.0 (optional)
User name	The user name (login name) to connect to the target server, used for HTTP basic authentication. This value is optional for AQ and JMS because B2B can use the configured JNDI data sources to connect to queues.	AS1 (optional AS2 (optional) Generic AQ (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic FTP (required) Generic HTTP (optional) Generic SFTP (required) Generic JMS (optional) Generic Email (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)
Use proxy	Select a proxy server if used.	AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic FTP (optional) Generic HTTP (optional) Generic SFTP (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)

[1] In File/FTP channels, if the `filename format` is set then the `directory name format` is ignored.

3. Click **Save**.

Task 3 Provide Channel Attributes

1. Click the **Channel Attributes** tab.

2. Provide channel attributes, as described in Table 5–4, depending on the channel/transport protocols selected in Task 1.

Table 5–4   Channel Attributes

Parameter	Description	Protocol Used With
Ack Mode	Select **Sync**, **Async**, or **None**, for the mode in which the trading partner receives messages. Select **None** for all generic exchanges. For MLLP exchanges, select **Sync** or **Async** for a transient connection. Select **None** for a permanent connection.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) MLLP-1.0 (required; for remote trading partners only) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)
Compressed	Select for message compression.	AS1 (optional) AS2 (optional) This parameter is available *only* with AS1 and AS2, although it may appear in the B2B interface for other protocols.
Description	Provide an optional description.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) MLLP-1.0 (optional; for remote trading partners only) Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) Generic TCP (optional)
Enable/Disable Channel	The channel is the communication interface between the host trading partner's host application and its installation.	MLLP-1.0 (required; for remote trading partners only) Generic TCP (required)

Table 5–4   (Cont.) Channel Attributes

Parameter	Description	Protocol Used With
Internal **Caution:** While the B2B interface permits you to select invalid protocols when **Internal** is selected, do not select any protocols other than the generic protocols.	Select this option if the channel is internal to the host trading partner's enterprise.	If this option *is* checked, then only the generic protocols are valid: Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) If this option *is not* checked, all protocols are valid: AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional)

Table 5–4   (Cont.) Channel Attributes

Parameter	Description	Protocol Used With
Retry Count	The number of times that Oracle B2B retries to send the message. See Section 5.5.5, "Configuring Delivery Retry Options" for more information.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) MLLP-1.0 (optional; for remote trading partners only) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) Generic TCP (optional)
Retry Interval	The interval, specified in minutes, after which B2B attempts to resend a message. B2B tries to resend the message if the status of the message is not Complete. For protocols with acknowledgments, B2B waits for the acknowledgment (formerly called the Time to Acknowledge parameter). If it is not received, the retry interval setting causes B2B to retry. See Section 5.5.5, "Configuring Delivery Retry Options" for more information.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) MLLP-1.0 (optional; for remote trading partners only) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) Generic TCP (optional)
Transport Callout	For the inbound message, B2B invokes the transport callout immediately after it receives a message from the transport. For the outbound message, B2B invokes the transport callout immediately before it sends a message to the transport.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) Generic File (optional) Generic AQ (optional) Generic FTP (optional) Generic HTTP (optional) MLLP-1.0 (optional; for remote trading partners only) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional) Generic SFTP (optional) Generic JMS (optional) Generic Email (optional) Generic TCP (optional)

3. Click **Save**.

Task 4 Provide Exchange Protocol Parameters

1. Click the **Exchange Protocol Parameters** tab.

2. Provide exchange protocol parameters, as described in Table 5–5, depending on the channel/transport protocols selected in Task 1.

Table 5–5   Exchange Protocol Parameters

Parameter	Description	Protocol Used With
Carriage Return Character	This value can be only one character. The carriage return character does not appear in the wire message payload. The default value is 0x0D (hexadecimal).	MLLP-1.0 (optional; for remote trading partners only)
Custom Immediate ACK File	Browse for a file with a customized acknowledgment.	MLLP-1.0 (optional; for remote trading partners only)
Discard HL7 ACK	Enable this property for the FA to be correlated at the transport level. This avoids the traditional message correlation that includes trading partner agreement identification, translation, and so on, thus improving performance. Because the ACK is stopped at the transport layer after correlation, it appears in the wire message report, but does not appear in the business message report. To enable the property, select one of the following codes. If the selected code is in the MSA.2 segment, then the ACK is stopped at the transport layer: **AA**—Application acknowledgment: Accept **AE**—Application acknowledgment: Error **AR**—Application acknowledgment: Reject **CA**—Application acknowledgment: Commit Accept **CE**—Application acknowledgment: Commit Error **CR**—Application acknowledgment: Commit Reject Selecting **None** does not enable this property. MSA.2 is the second element of MSA segment.	MLLP-1.0 (optional; for remote trading partners only)
Duplicate Elimination	If enabled, a duplicate elimination header is added for an outbound message. This flag does not apply to the inbound message flow.	ebMS-2.0 (optional) ebMS-1.0 (optional)
End Block	The value for this property can be a string. This property is used to indicate the end of the message. Generally, **End Block** is sent after the message is sent to the trading partner.	Generic TCP (optional)
End Block Character	This value can be only one character. The end block character does not appear in the wire message payload. The default value is 0x1C (hexadecimal).	MLLP-1.0 (optional; for remote trading partners only)
Header Length	This property defines the header size, which is prefixed to the actual data. (This includes the start block, message length, and padded header).	Generic TCP (optional)

Table 5–5   (Cont.) Exchange Protocol Parameters

Parameter	Description	Protocol Used With
Identify TP by Delivery Channel	The trading partner is identified using the delivery channel. Enable this parameter to identify an incoming message by the delivery channel configured for the remote trading partner (rather than by using the MLLP ID). This feature serves as an anonymous trading partner, for situations when identifying the sender is not important. If this parameter is not checked, then the MLLP ID (or some document-level identifier such as HL7 Message Application ID or HL7 Message Facility ID to identify the agreement) is required for MLLP exchanges.	MLLP-1.0 (optional; for remote trading partners only)
Immediate ACK **Note:** The MLLP immediate ACK of an incoming business message (with control number 1017, for example), prefixes the control number with *A*, as in A1017. This indicates to the trading partner that it is an ACK control number. If the prefixed string exceeds the permissible length (for example, if any validation rules are violated at the receiving end), use the Map ACK Control ID parameter.	An immediate acknowledgment is generated and transmitted in the TCP transport layer instead of the document layer. It is an alternative to the functional acknowledgment. It is available when the turnaround time of a functional acknowledgment is undesirable (for example, for some business-critical health care applications), because the functional acknowledgment captures translation and validation errors. Oracle B2B can send an immediate acknowledgment in the following modes: ■ Default: B2B parses the incoming HL7 message and generates an acknowledgment from it. In this mode, B2B can send the acknowledgment to the sending application with correlation details (for example, the control number from the incoming message, the sending application, and so on.) Hence, the trading partner application can correlate the incoming acknowledgment message. If mapping the MSH.10 of the ACK with the MSH.10 of the incoming business message is required, then enable the Map ACK Control ID property. By default, an Immediate ACK is a generic ACK. If generating an ACK with a trigger event is required, then enable the Map Trigger Event property. ■ Simple: B2B sends the predefined acknowledgment message to the sender and does not parse the message. ■ Custom: B2B sends the custom HL7 acknowledgment message based on a configurable file content. If this mode is selected, then specifying the file in the Custom Immediate ACK File property is required. ■ Negative: Select this option to send an immediate ACK only in the case of exceptions.	MLLP-1.0 (optional; for remote trading partners only)
Map ACK Control ID	Select to enable the mapping of the MSH.10 message header of the business message to the MSH.10 message header of the *immediate* acknowledgment.	MLLP-1.0 (optional; for remote trading partners only)

Table 5–5   (Cont.) Exchange Protocol Parameters

Parameter	Description	Protocol Used With
Map Trigger Event	Sends an immediate acknowledgment with a trigger event.	MLLP-1.0 (optional; for remote trading partners only)
Message Order Semantics	A placeholder for CPP/CPA; not involved during run time.	ebMS-2.0 (optional)
Message Length Index	This property indicates the data size available in the header. Start index to end index defines the message size.	Generic TCP (optional)
Persist Duration	A placeholder for CPP/CPA; not involved during run time.	ebMS-2.0 (optional)
Receipt Delivery Option	This parameter is used to configure a URL to which MDN has to be sent back in the case of an asynchronous mode.	AS2 (optional)
Retain Header	Select this property to retain the header while sending the message to the trading partner (for outbound messages) or to Oracle B2B (for inbound messages). When you retain the header, B2B does not not handle the custom header. This is handled using the transport callout.	Generic TCP (optional)
Send Party Type and Value	If enabled, the send party type and value from the message header are sent to the back-end application.	ebMS-2.0 (optional) ebMS-1.0 (optional)
Signed and Compressed	If selected, the message is first signed, and then compressed. If *not* selected, the message is first compressed, and then signed.	AS1 (optional) AS2 (optional)
Start Block	This value can be a string.	Generic TCP (optional)
Start Block Character	This value can be only one character. The start block character does not appear in the wire message payload. The default value is 0X08 (hexadecimal).	MLLP-1.0 (optional; for remote trading partners only)

3. Click **Save**.

Task 5 Configure Security

1. Click the **Security** tab.

2. Provide security parameters, as described in Table 5–6, depending on the channel/transport protocols selected in Task 1.

> **Note:**   The **Digital Signature** and **Encryption** lists are populated with the available certificates when the Key Store location is provided for the host trading partner. See Task 6, "Provide Key Store Information for the Host Trading Partner" for more information.

Table 5–6   Security Parameters

Parameter	Description	Protocol Used With
Ack Signed	Select this option to ensure that the responder acknowledges receipt of the messages; nothing needs to be provided.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)
Digital Signature	To use a digital signature certificate, the Key Store *must* have the corresponding private key. If **Message Signed** is selected, then select the following for AS1: SMIME 3.0 with SHA1 - RSA If **Message Signed** is selected, then select one of the following for AS2: SMIME 3.0 with MD5 - RSA SMIME 3.0 with SHA1 - RSA If **Message Signed** is selected, then select one of the following for ebMS-2.0 and ebMS-1.0: XMLDSIG with SHA1 - RSA XMLDSIG with SHA1 - DSA If **Message Signed** is selected, then select one of the following for RosettaNet-V02.00: SMIME 3.0 with MD5 - RSA SMIME 3.0 with SHA1 - RSA SMIME 2.0 with MD5 - RSA SMIME 2.0 with SHA1 - RSA XMLDSIG with SHA1 - RSA XMLDSIG with SHA1 - DSA If **Message Signed** is selected, then select one of the following for RosettaNet-01.10: SMIME 3.0 with MD5 - RSA SMIME 3.0 with SHA1 - RSA SMIME 2.0 with MD5 - RSA SMIME 2.0 with SHA1 - RSA	AS1 AS2 ebMS-2.0 ebMS-1.0 RosettaNet-V02.00 RosettaNet-01.10

Table 5–6   (Cont.) Security Parameters

Parameter	Description	Protocol Used With
Encryption	To use an encryption certificate, no private key entry is needed. If **Message Encrypted** is selected, then select one of the following for AS1 and AS2: SMIME 3.0 with DES SMIME 3.0 with 3DES SMIME 3.0 with RC2 - 40 SMIME 3.0 with RC2 - 64 SMIME 3.0 with RC2 - 128 If **Message Encrypted** is selected, then select one of the following for ebMS-2.0 and ebMS-1.0: XMLENC with 3DES - RSA-v1.5 XMLENC with AES-128 RSA-OAEP XMLENC with AES-192 RSA-OAEP XMLENC with AES-256 RSA-OAEP	AS1 AS2 ebMS-2.0 ebMS-1.0 RosettaNet-V02.00 (optional)
Message Encrypted	Select this option to enable message encryption. This option requires you to select an encryption schema in the **Encryption** field.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) RosettaNet-V02.00 (optional)
Message Signed	Select this option to provide a digital signature in the Digital Signature field.	AS1 (optional) AS2 (optional) ebMS-2.0 (optional) ebMS-1.0 (optional) RosettaNet-V02.00 (optional) RosettaNet-01.10 (optional)

3. Click **Save**.

> **Note:**   For AS1, B2B supports only SHA1 for signing. MD5 is not supported for AS1 signing.

5.5.1 About MLLP

An MLLP delivery channel is established by a two-way handshake between the server and client. It is always bidirectional, unlike other transports, and is used for both sending and receiving messages. An MLLP delivery channel is configured for the remote trading partner only, and is configured as a server socket or a client socket. As a server socket, the channel accepts connections on the specified port. As a client socket, the channel establishes a connection on the specified IP address and port. For either socket type, you specify a permanent or transient connection type. A permanent connection, once established, is cached and serves as a channel for the message exchange throughout the life cycle of the endpoint. A transient connection serves as a channel only for exchanging one set of messages comprised of the business message and its acknowledgment. See Section 5.5.1.1, "Overriding the Connection Mode."

A recommended configuration is for the sender to configure the MLLP client delivery channel and for the receiver to configure the MLLP server channel. For example, if Acme wants to send an HL7, Custom, or positional flat file message to GlobalChips, Acme can have the client MLLP permanent channel and GlobalChips can have the

server MLLP permanent channel. MLLP connection types (permanent and transient) for the server and client must match (both permanent or both transient). However, in some cases the sender can have the server channel and receiver can have the client channel provided the connection is pre-established.

Because MLLP is a bidirectional channel, it is not considered to be a listening channel and the same MLLP delivery channel can be used for both sending and receiving messages.

Because MLLP operates in single delivery channel mode by default, simply select a delivery channel under the remote trading partner when creating an agreement. If operating in a non-single MLLP delivery channel mode is required, select a different MLLP delivery channel in the other agreements.

5.5.1.1 Overriding the Connection Mode

To override the connection mode for a message without changing the configuration manually, set the following properties:

For changing from a transient to a permanent connection:

```
CONNMODE:Permanent
```

For default integration:

```
b2b.connMode:Permanent
```

You can also change from a permanent connection to a transient connection.

5.5.1.2 Generic Support for TCP

MLLP uses SB (start byte), EB (end byte) and CR to interpret a message. To interpret a message using the length of the data or the start string and end string instead of SB and EB, Oracle B2B provides a generic solution for TCP.

 For generic support for TCP, use with the following parameters on the **Exchange Protocol Parameters** tab (shown in Figure 5–21): **Start Block**, **End Block**, **Header Length**, **Message Length Index**, and **Retain Header**.

Figure 5–21   Parameters for Generic TCP
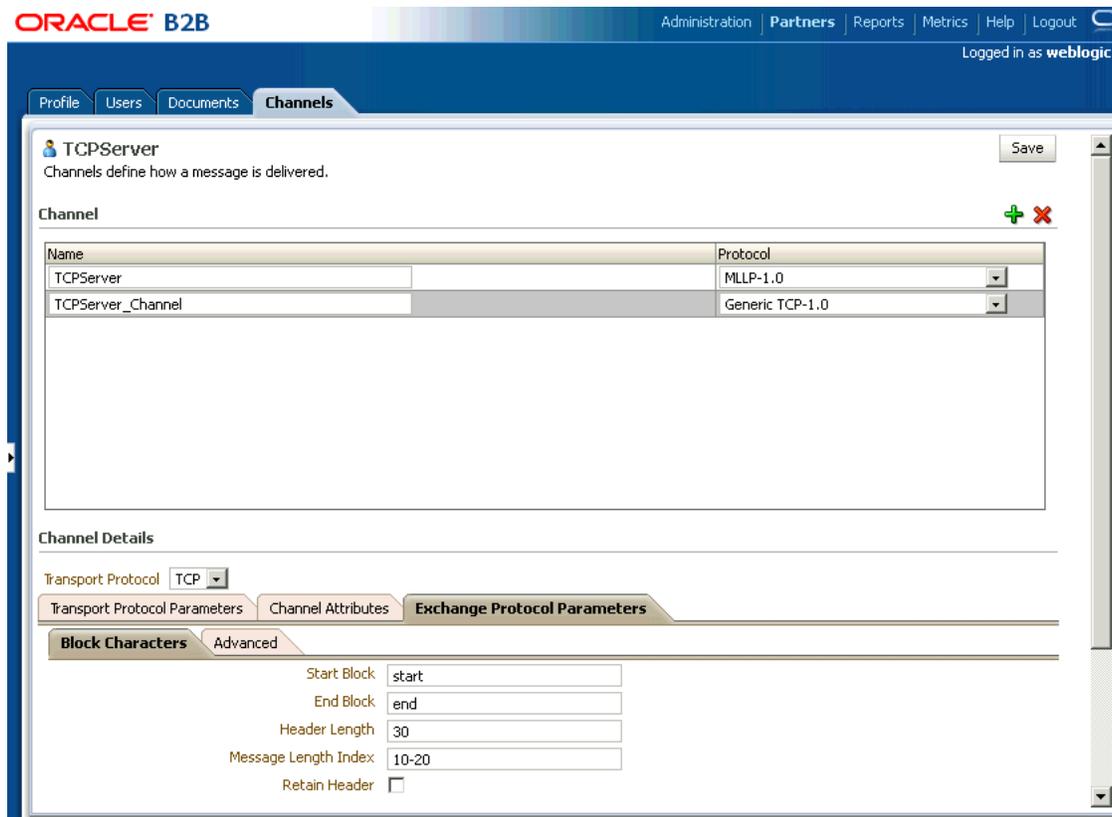
The image shows the Exchange Protocol Parameters tab. It has the following fields: Immediate ACK, Custom Immediate ACK File, Map ACK Control ID, Map Trigger Event, Discard HL7 ACK, Start Block Character, End Block Character, Carriage Return Character, Identify TP by delivery channel, Start Block, End Block, Header Length, Message Length Index, and Retain Header.

****

> **Note:** When you create a generic TCP channel using the MLLP protocol, the parameters on the Exchange Protocol Parameters tab appear as shown in Figure 5–21. After creating the channel, two subtabs appear, with MLLP-specific and generic TCP-specific parameters grouped under them.

See Table 5–5, " Exchange Protocol Parameters" for descriptions of these parameters.

Table 5–7 describes how Oracle B2B processes messages using MLLP when data is sent or received using the parameters that support generic TCP.

Table 5–7   Generic TCP Solutions

Generic TCP Solution	Description
Send or receive data by specifying a start block and end block	Use the **Start Block** and **End Block** parameters available on the **Exchange Protocol Parameters** tab when you select MLLP-1.0 for a remote trading partner. See Table 5–5 for descriptions of the **Start Block** and **End Block** parameters.
	Example: `<start block>`*Data*`<end block>`
Send or receive data by specifying a start block, end block, and data length	Use the **Start Block**, **End Block**, **Message Length Index**, and **Header Length** parameters available on the **Exchange Protocol Parameters** tab when you select MLLP-1.0 for a remote trading partner. See Table 5–5 for descriptions of the parameters.
	Example: `<start block><length>`*Data*`<end block>`
Send or receive data by specifying the data length	Use the **Message Length Index** and **Header Length** parameters available on the **Exchange Protocol Parameters** tab when you select MLLP-1.0 for a remote trading partner. See Table 5–5 for descriptions of the **Message Length Index** and **Header Length** parameters.
	Example: `<length>`*Data*
	Example: `<length+header>`*Data*
	That is, 15HDRDATADATADATA, where you configure:
	Message Length Index=1-2 Header length=5
	15 is the length start after end index of Message Length Index. HDR is the header.
Send or receive data by specifying the start block and data size	Use the **Start Block**, **Message Length Index**, and **Header Length** parameters available on the **Exchange Protocol Parameters** tab when you select MLLP-1.0 for a remote trading partner. See Table 5–5 for descriptions of the **Start Block**, **Message Length Index**, and **Header Length** parameters.
	Note: In this case, the start block is part of the header and the minimum message length index must be more than the start block size.
	Example: `<start block><length>`*Data*
Retain the back-end application header and B2B will not add the start block, data size, and end block.	To send data to the trading partner without adding a header and retain the back-end application header, select the **Retain Header** property. See Table 5–5 for a description of **Retain Header** parameter.

5.5.1.3 Dynamic Endpoints

The dynamic IP feature of MLLP provides flexibility to dynamically change the endpoints associated with a delivery channel. This is done by overriding the IP address of the delivery channel through the `actionName/eventName` attribute in the message enqueue header.

For example:

```
eventName=DynamicIP:GlobalChips:IP_address:port_number
```

or

```
actionName=DynamicIP:GlobalChips:IP_address:port_number
```

This feature is also available in B2B composites (as part of the SOA Service Component Architecture (SCA) assembly model) using the following syntax:

```
b2b.toDynamicIP=GlobalChips:IP_address:port_number
```

The `b2b.toDynamicIP` property is set in a normalized message property that is sent to B2B.

Oracle B2B generates a unique control number for each message. For a broadcasting case involving multiple dynamic endpoints corresponding to the same trading partner, the back-end application must provide the control number. Oracle B2B stores and uses

the dynamic endpoint details for correlation of the acknowledgment. No additional configuration is required.

5.5.1.4 Using a Transport Callout to Extract Custom Headers

To extract a custom header for outbound messages, add the CUSTOM_HEADER property in the actionName property from the back-end application. This property will be available in the callout as a CUSTOM_HEADER parameter of CalloutMessage. You can retrieve the property in the callout for your usage.

For example:

```
eventName= CUSTOM_HEADER:your_value
```

For default integration:

```
b2b.customHeader= your_value
```

To extract a custom header for inbound messages, set the CUSTOM_HEADER property as the CalloutMessage parameter in the callout. The property will be available as part of the actionName properties in the back-end application. See Example 13–1, "Setting and Getting the CUSTOM_HEADER Property" for details.

5.5.2 Message Sequencing

Exchanging messages in sequence can be challenging in a multi-threaded system, because the first message produced may not necessarily arrive at the destination first. For enterprises with this business requirement, Oracle B2B provides a sequencer and a dispatcher. The sequencer sequences a message based on arrival time. The dispatcher dispatches the sequenced message. Message sequencing is available for outbound and inbound directions.

Protocols supported for message sequencing include MLLP Exchange (TCP transport) and Generic Exchange (FILE, FTP, SFTP, JMS, AQ, and HTTP transports).

Notes: Even though it works for all documents, message sequencing is certified only for EDI, HL7, and Custom document protocols.

Sequencing is not supported for transient mode MLLP connections.

EDI batching is not supported for message sequencing. If sequencing is enabled on messages as part of BATCH, it can lead to errors and not all messages are processed. Do not use EDI batching on the messages that are sequenced.

FunctionalAck (FA) and Acknowledgement (ACK/MDN) are not sequenced.

Sequencing does not support the Delivery Channel retry feature. Use the Auto Stack Handler in the Sequencing feature to retry delivery of failed document delivery attempts rather than the retry setting in delivery channel. When documents are sequenced, the delivery channel used for the documents should avoid the use of retry settings.

5.5.2.1 Outbound Message Sequencing

Outbound Message Sequencing for AQ and JMS Delivery Channels

To enable sequencing for an outbound message, for AQ and JMS delivery channels, enqueue the message by setting the `ACTION_NAME` property to `TARGET:`*`sequence_target_name`* as shown in Example 5–1.

However, when using the ENQUEUE utility that is provided with the `b2b.jar`, set `eventName` (not `ACTION_NAME`) to `TARGET:`*`sequence_target_name`*; for example, `eventName=TARGET:sequence1`.

To enable sequencing when using the default channel, use `b2b.sequencingTarget = `*`sequence_target_name`*.

When using a self written AQ Enqueue utility, the AQ Header to be used is `ACTION_NAME`. `eventName` is to be used when using the B2B provided Enqueue utility in `b2b.jar`.

Example 5–1   Outbound Message Sequencing for AQ and JMS Delivery Channels

```
ACTION_NAME = TARGET:sequence_target_name
```

Outbound Message Sequencing for FTP and SFTP Delivery Channels

To enable FTP/SFTP sequencing selecting only the sequencing flag in the delivery channel configuration does not work. You must also configure an FTP listening channel with following parameters:

Sequencing

TimeStamp Format

TimeStamp Offset

TimeStamp Source

For example the following are sample values for the parameters:

Sequencing: true

TimeStamp Format: 43,55,'MMM d yyyy'

TimeStamp Offset: +0000

TimeStamp Source: MMM dd yyyy

Note that, if sequencing is enabled, the order in which files are copied to the folder is the order in which they are processed. If a large payload is copied, the partner must wait until the large payload copy is complete. Then the sender may send the next file, as sequencing is based on the last-modified-timestamp on the file.

Outbound Message Sequencing for HTTP Delivery Channels

Post the message in b2b/sequenceReceiver, and add an additional HTTP header `TARGET:`*`sequence_target_name`*, otherwise the IP address is used as the sequence target.

The messages enqueued with the above header will be sequenced based on the specified sequence target for the transport protocols such as FTP, SFTP, JMS, AQ and HTTP/HTTPS. The enqueued messages with this header would be processed by Oracle B2B, and based on the enqueue time, the messages are sequentially delivered to the trading partner.

To dispatch the sequenced message, configure the **Outbound Dispatcher Count** parameter, shown in Figure 5–22.

Figure 5–22   Dispatcher Configuration: Administration > Configuration Tab

The image shows Miscellaneous (continued) configuration parameters: Generic Message Type (default value set to false), Outbound Dispatcher Count (default value set to 0), Inbound Dispatcher Count (default value set to 0), Auto Stack Handler (default value set to false), and Auto Stack Handler Interval (default value set to 1)

****

By default, the value is 0, which is the setting for sequencing without dispatching (stacking). Depending on the message load, set **Outbound Dispatcher Count** to the appropriate value.

Sequencing in Burst Mode

For sequencing with AQ and JMS, the Oracle B2B server might not be polling while messages get added into the queue. When Oracle B2B begins polling, several messages are available in a burst mode and Oracle B2B cannot maintain the sequence under normal conditions. To ensure sequence in burst mode, the following is recommended:

For AQ, along with the `TARGET:target_name` property, add the flag for `B2B_SEQUENCE_TIMESTAMP` with timestamp as the value.

For example:

```
TARGET:targetname;B2B_SEQUENCE_TIMESTAMP:1284371552121
```

In this example, 1284371552121 is the time in milliseconds (in Java).

For JMS, a new header can be added as `B2B_SEQUENCE_TIMESTAMP` with value as the time in milliseconds.

5.5.2.2 Inbound Message Sequencing

Inbound message sequencing is enabled as part of the delivery channel configuration. The incoming messages received by these delivery channels are processed by Oracle B2B and delivered in a sequenced manner based on the inbound time.

To enable sequencing for an inbound message, enable the **Sequence** property for the delivery channel, as shown in Figure 5–23.

Figure 5–23 Sequencer Configuration

The image shows the Transport Protocol Parameters tab, with the following channel details: Connection Mode (value is Server), Host Name (value is localhost), Port, Permanent Connection (selected), Sequence (selected), Polling Interval (value is 10), and Timeout (value is 300).

****

To dispatch the sequenced message, configure the **Inbound Dispatcher Count** parameter, as shown in Figure 5–22.

For AQ and JMS, in the case of inbound messages received from a partner, if there is a header TARGET (similar to how outbound message sequencing is enabled), then the same value is used and inbound messages are sequenced on the given target.

For HTTP inbound message sequencing, Oracle B2B exposes a URI, /b2b/sequenceReceiver, and requests arriving at this endpoint are processed sequentially. If you want to use multiple sequence targets, you can provide them by adding an HTTP header SEQUENCE_TARGET to the request.

To add SEQUENCE_TARGET as a header to an outbound HTTP message, use the **Additional transport headers** parameter in the delivery channel. See Table 5–3 for more information.

If the incoming message's HTTP headers contains SEQUENCE_TARGET as a header, then the value of this is used as the sequence target. If SEQUENCE_TARGET is not available and there is a FROM HTTP header, then that is used as the sequence target. If FROM HTTP header does not exist, then B2B will use the IP address from which the message originated as the Sequence Target. The originating IP address of the HTTP request is treated as the Sequence Target by default.

5.5.2.3 Sequencing Without Dispatching

Trading partner downtime is typically handled by stacking messages in the back-end application, which requires the entire message processing in B2B after the downtime. This leads to under-utilizing the B2B application during downtime and overloading when the trading partner comes up. This affects the regular message flow, because there is a surge in message processing.

When the auto stack handler is used, then Oracle B2B retries the outbound failed message in sequence. Once the endpoint is up for delivery, all messages in the sequence will be eligible for delivery, and this may cause an overload of message delivery at the endpoint. To reduce the outflow, set the b2b.OutboundDispatchInterval property, which sets the interval between dispatch of messages in milliseconds.

Upon trading partner delivery failure, the corresponding messages are marked not to be picked up by the dispatcher, resulting in stacking the messages in B2B instead of the back-end application. To process the messages, set the following properties:

```
Auto Stack Handler = true
Auto Stack Handler Interval = interval (in seconds)
```

The **Auto Stack Handler** and **Auto Stack Handler Interval** parameters are shown in Figure 5–22. When set to true, the stacked message are eligible for delivery by the dispatcher during an appropriate interval. It is also possible to specify the variable interval with a comma-separated value to **Auto Stack Handler Interval**.

5.5.2.4 Troubleshooting Message Sequencing

B2B Becomes Non-Responsive

If B2B becomes non-responsive after running thousands of messages as part of outbound sequenced messages, check the number of dispatchers. The recommended maximum number of dispatchers is 5.

Sequenced Message Failure

If the failure of a sequenced message is due to

- Agreement not found

- Validation errors which can be fixed by updating the ECS file.

You can recover the failed message by the following options:

1. Inbound case: resubmit the wire message.

2. Outbound case: resubmit the app message.

Otherwise, you can manually delete the row in the sequence manager table, if you do not want to process the failed message, but want to allow the blocked messages (following the failed message in sequence) to be allowed for delivery.

5.5.3 Using Transport Sync Callback

Sync support is provided using callout. This provides a platform to respond to the incoming requests in a synchronous way.

There may be several requirements for an Enterprise to send business responses synchronously. For example, an inbound 270 document may expect a 271 document response synchronously. Enterprises may want to set up sync support for simple custom documents of their choice over HTTP protocol.

Callout is the key component to enable synchronous response. In this model, callout holds the responsibility of delivering the incoming request message to the back end application, and get the corresponding business response from the back end application. Capabilities of back end applications in Enterprises may vary, so the callout implementers can choose their own approaches for sending and receiving messages to and from back end applications.

B2B Engine provides inbound message as an input to configured callout, and expects callout to give the response received from the back end application as its output. The output of callout will be processed as an outbound message in B2B, and the same is streamed back as a response for the inbound message on the same HTTP connection.

To configure sync response:

1. Set up the inbound and outbound agreement.

2. Create callout with the capability to send inbound requests and receive its business response from back end applications.

   See Example 13–3, "Code Example of a Sync Callback Callout" for a code sample.

   Callout output should have all the required values for Oracle B2B to process it as an outbound message. This may include `TO_PARTY`, `DOCTYPE_NAME`, `DOCTYPE_REVISION` and `payload`, among others.

   Callout output parameters such as `TO_PARTY`, `DOCTYPE_NAME`, and `DOCTYPE_REVISION` are case-sensitive. Parameters supported by Oracle B2B JMS adapters can be output valid output parameters.

3. Attach the callout to the Inbound Agreement.

   All the sync requests should be send to the following URL

   ```
   http://host:port_number/b2b/syncreceiver
   ```

4. Test the flow.

   > **Note:** One dummy outbound channel is required to deploy the outbound agreement on the responder side, however Oracle B2B will not use the dummy channel, and the sync response is sent back on the same connection on which it received the inbound request.

   > **Note:** Timeout can be configured as an additional transport header at HTTP delivery channel. For example: `timeout=123`

   Note that the initiator of the sync flow must pay attention to the following issues:

   The initiator of the sync flow must add `syncresponse=true` as part of the Addition Transport Header in a Generic HTTP, AS2, or ebms channel.

   The initiator of the sync flow must set `ack mode none/Async` in AS2 or ebms channel.

5.5.4 Correlating Messages Using JMS Queues

You can correlate inbound and outbound messages using JMS queues, by setting `A2A=true` in the JMS header.

If the message ID (`MSG_ID`) is provided from a back end application, then `MSG_ID` is set to JMS Correlation ID in the B2B output, otherwise the JMS Message ID is set to JMS Correlation ID in the B2B output.

5.5.5 Configuring Delivery Retry Options

In the critical situations of B2B world, it is important to have the message delivered to the destination without fail and receive the exchange level acknowledgement or functional acknowledgement on time.

Oracle B2B provides the ability to retry message delivery at the Channel and Document levels.

5.5.5.1 Delivery Retry at the Channel Level

Channel retry is associated with the delivery channel, and is used to ensure successful delivery. You can configure the Retry Count and Retry Interval parameters for the number of times to retry and the interval between each retry. Oracle B2B retries the message for a successful delivery until all available retries are exhausted before errors are written. See Table 5–4 for information about the channel retry parameters.

Channel Details

Transport Protocol File

Transport Protocol Parameters   **Channel Attributes**

Retry Interval [   ]   Transport Callout [▼]
Retry Count   [   ]
Description   [   ]

Channel Details screen with Channel Attributes tab selected showing Retry Interval, Retry Count, Description, and Transport Callout parameters.

****

For any exchange protocol with Acknowledgement case (for example MDN in AS2 and Acknowledgement in ebMS), the channel retry is used to retry the business message until the message gets to either the COMPLETE or ERROR state after completing the configured retry count. For generic exchange, channel level retry plays a role only in case of transport error.

The number of remaining retries in the retry count and retry interval for a specific message can be seen as part of the business message report.

5.5.5.2 Delivery Retry at the Document Level

Document retry is associated with an Agreement, and is used to ensure the successful integration of the message with the recipient trading partner. This feature consists of configuring a timeout value within which the Functional Acknowledgment has to be received for outbound business messages.

To enable this feature, set the **Document Retry Count** and **Document Retry Interval** parameters as shown in the following graphic.

Agreement

⚇⇄⚇ Acme_GlobalChips_EDIFACT_D98A_Orders_AS2   [Save] [Validate] [Deploy] [Export]

👤 ——— 🗐 ——→ 💰
Acme   Orders_def   GlobalChips

Details

* Agreement Id [Acme_GlobalChips_EDIFACT_D98A_]   Start Date [   ] 📅
* Name   [Acme_GlobalChips_EDIFACT_D98A_]   End Date   [   ] 📅
Description   [   ]   Callout   [▼] Callout Details

Agreement Parameters

Validate   ☑   FA Handled by B2B   [None ▼]
Translate   ☑   Document Retry Interval [1]
Functional Ack ☑   Document Retry Count   [3]

Agreement configuration screen showing the Agreement Parameters in the bottom section of the screen: Validate (checked), Translate (checked), Functional Ack

(checked), FA Handled by B2B (set to None), Document Retry Interval (set to 1), Document Retry Count (set to 3).

****

After successful transmission of a business message, B2B must wait for Functional Acknowledgment for specified time. If FA is not received after retries are exhausted, B2B will raise an exception message to B2B Inbound Queue.

The number of remaining retry and interval for a specific message can be seen as part of the business message report.

Generic Exchange Scenario

For generic exchange, document retry attempts are triggered only upon successful transport Acknowledgment, and in the case of standard based exchange (such as AS1/AS2), only upon receipt of Positive Acknowledgment. That is, for a generic exchange, document retry attempts are triggered only post-transmit, whereas for a standard Acknowledgment case, attempts are triggered only upon receipt of positive Acknowledgment. For a negative Acknowledgment, document retry attempts are not triggered.

Channel-level Retry Interoperability

Channel-level retry attempts are not triggered by document level retries attempts.

In cases where no channel retry parameters are configured, document-level retry attempts are triggered after Document Retry Interval expires.

5.6 Using the Auto Create Agreement Feature

In the **Partner** area, shown in Figure 5–24, you can use the **Auto Create Agreement** icon to create an agreement for a remote trading partner.

Figure 5–24   The Auto Create Agreement Feature
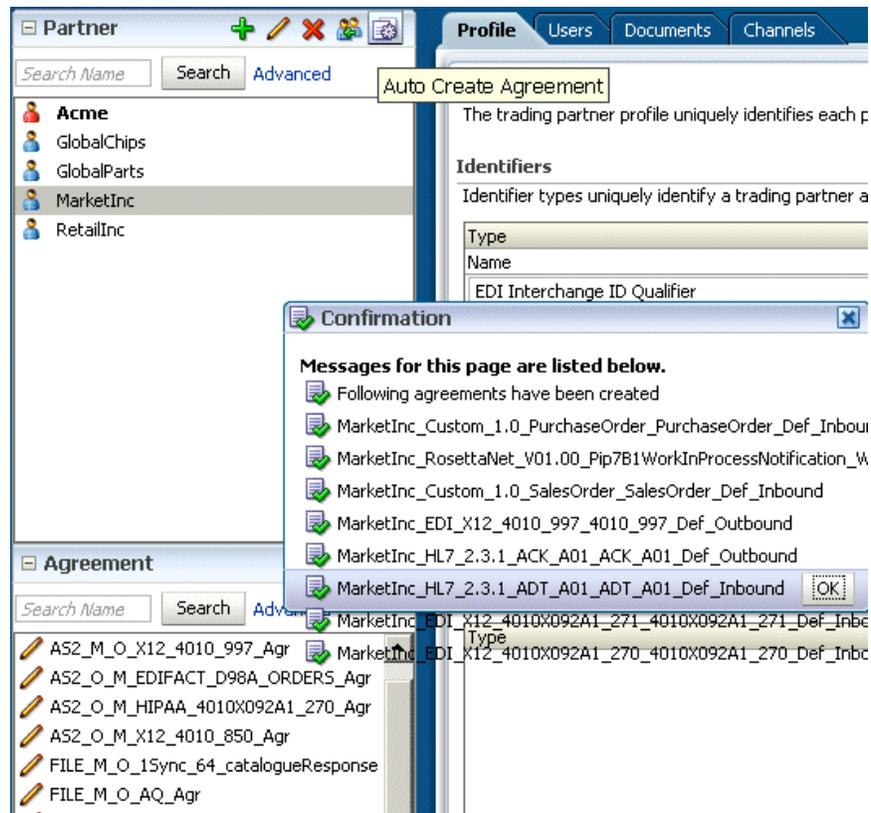
The image shows the Oracle B2B screen. The top left pane is labeled Partner and shows the list of partners (Acme and GChips). The bottom left pane is labeled Agreement and shows the list of agreements. The top right of the screen shows a Confirmation dialog box with messages for the page.

****

This feature creates one agreement for each document definition associated with the selected remote trading partner. You can further customize the agreement on the **Agreement** tab. See Chapter 6, "Creating and Deploying Trading Partner Agreements," for more information about the **Agreement** tab.

5.7 Using Identifiers for Trading Partner Lookup

Identifiers available in design-time data are used to look up trading partners. Identifiers do not need to be part of a deployed, active agreement. The appropriate document and exchange identifiers are used for lookup; for example:

■   For the AS2-1.1 exchange protocol, the AS2 identifier is used.

■   For the EDI X12 document protocol, the Sender Group ID and Sender Interchange ID are used.

5.8 Scheduling Trading Partner Downtime

On occasion a trading partner will need to go offline for planned maintenance. You can configure downtime so that partners are adequately notified and messages are queued for delivery when downtime ends.

See Chapter 12, "Scheduling Trading Partner Downtime" for information about scheduling and managing trading partner downtime.

5.9 Broadcasting Messages to Multiple Trading Partners

Oracle B2B provides the capability to send the same message to multiple trading partners. This gives you the flexibility to create your own groups of trading partners specific to certain business needs. The back-end application interface can then send a message to the group and Oracle B2B sends the message to all the members of the group.

The salient points of the feature are:

1. Prevents BPEL from sending multiple files, reducing the traffic on BPEL-B2B and also in B2B.

2. Introduces the concept of trading partner groups in B2B and uses it for broadcasting. This will prevent BPEL users from knowing about trading partner details as well as the group because it is B2B-specific metadata.

3. Provides an event driven approach to handle the multiple messages to various trading partners.

To use broadcasting, specify the group name as part of the `actionName` attribute. The `actionName` attribute would look like this:

```
actionName = Grouping:name_of_the_group
```

Broadcasting requires an additional identifier to be added to the Trading Partner configuration indicating the group to which the partner is assigned. This is achieved by creating a new custom Identifier and associating the same for the required trading partners. See Section 10.3, "Creating Custom Trading Partner Parameter Types" for more information.

5.10 Validating Certificate in Inbound Message Processing

This feature performs Certificate validation for the inbound signed message processing.

The following configuration is required to validate the certificates in inbound message processing.

- Add a new identifier with the name CertificateAlias.

- Set the required alias in the trading partner profile with the name as CertificateAlias.

- Enable this feature by setting the `b2b.validateMessageCertificate` property in Enterprise Manager console to `true`. By enabling this property, all the messages will be validated against the certificate.

6

Creating and Deploying Trading Partner Agreements

This chapter describes a trading partner agreement that defines the terms that enable two trading partners, the initiator and the responder, to exchange business documents. It also discusses how to use trading partner agreements to identify the trading partners, trading partner identifiers, document definitions, and channels.

The final steps in the Oracle B2B process flow, shown in Figure 6–1, are to create and deploy the agreement.

Figure 6–1   Oracle B2B Process Flow
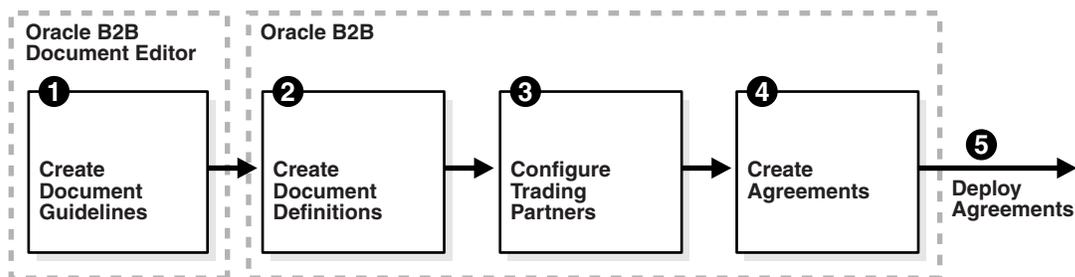
The image shows a step diagram linking the following blocks from left to right: 1. Oracle B2B Document Editor: Create Document Guidelines; 2. Oracle B2B: Create Document Definitions; 3. Oracle B2B: Configure Trading Partners; 4. Oracle B2B: Create Agreements; 5. Deploy Agreements.

****

This chapter contains the following topics:

- Section 6.1, "Introduction to Agreements"
- Section 6.2, "Creating an Agreement"
- Section 6.3, "Deploying an Agreement"
- Section 6.4, "Deleting and Exporting Agreements"

See the following for more information:

- Chapter 9, "Managing Deployments," for how to export agreements and manage deployment states
- Chapter 7, "Importing and Exporting Data," for how to export agreements

6.1 Introduction to Agreements

An agreement consists of two trading partners—the host trading partner and one remote trading partner, and represents one type of business transaction between those partners. For example, if Acme and GlobalChips participate in both EDIFACT and RosettaNet exchanges with each other, you create an agreement for each of the exchanges. Some exchanges are bidirectional, requiring an agreement for each direction.

For example, if Acme sends a sales order to GlobalChips using a Custom document sent using the Generic File protocol, you create an agreement for the outbound direction, where Acme sends the order, and for the inbound direction, where Acme is the receiver. A change to a component of an agreement (for example, a change to the document definition) is effective automatically in the agreement.

Creating an agreement is the last step in the design of a B2B transaction. Before you create an agreement, you must have already created the document definitions and configured the trading partners. See Chapter 4, "Creating Document Definitions," and Chapter 5, "Configuring Trading Partners," for more information.

6.2 Creating an Agreement

Figure 6–2 shows the Oracle B2B interface for working with agreements. Click a remote trading partner name to see its agreements with the host trading partner.

Figure 6–2   Creating an Agreement
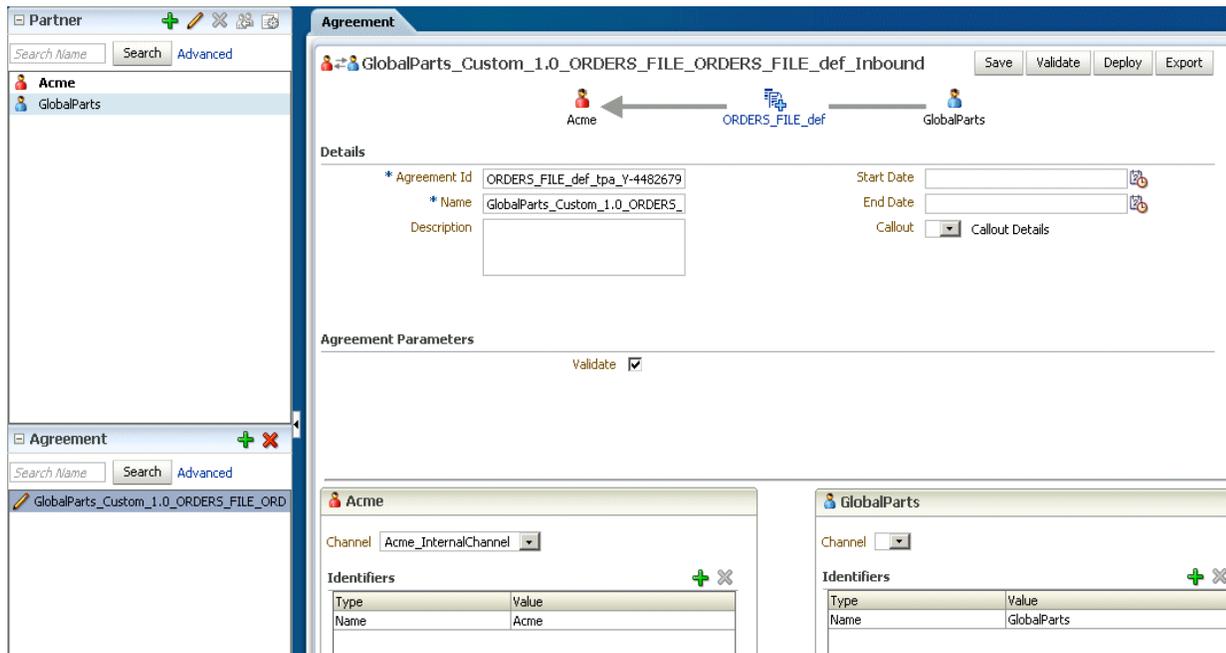
The image shows the Oracle B2B screen. The top left pane shows the Partners and the bottom left pane shows the Agreements. The main pane of the screen shows the Agreement tab.

****

Figure 6–3 shows the steps to create an agreement.

Figure 6–3   Steps to Creating an Agreement (Workflow Overview)
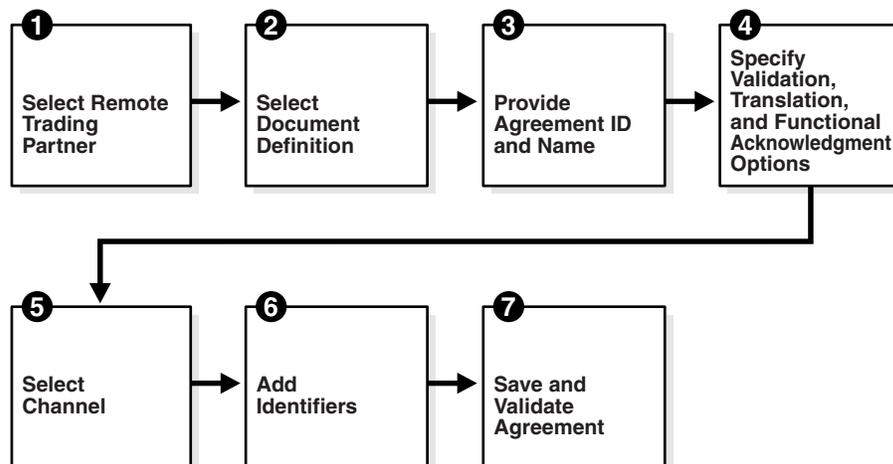
The image shows a step diagram linking the following blocks: 1. Select Remote Trading Partner; 2. Select Documentation Definition; 3. Provide Agreement ID and Name; Specify Validation, Transaction, and Functional Acknowledgement Options; 5. Select Channel; 6. Add Identifiers; 7. Save and Validate Agreement.

****

Step 1: Identify the remote trading partner

The host trading partner is automatically included in an agreement, so you need only identify the remote trading partner. You can do this in two ways: select the partner from the **Partners** region before adding the agreement, or select the host trading partner, click **Add** in the **Agreements** region, and click the **Select Partner** icon in the **New Agreement** region.

Step 2: Select the document definition

The document definition is selected for the host trading partner, as reflected in the Select Document Definition dialog, shown in figure Figure 6–4.

Figure 6–4   Selecting the Document Definition

The image shows a section of the Select Document Definition pane displaying the following two columns: Partner and Document Definition.

****

For an exchange in which you need both outbound and inbound agreements, do the following:

- For the outbound agreement, select the document definition in which the host trading partner is the sender (**Acme --> Globalchips** in Figure 6–4)

- For the inbound agreement, select the document definition in which the host trading partner is the receiver (**Acme <-- GlobalChips** in Figure 6–4)

Step 3: Provide the agreement ID and name

Provide any agreement identifier and agreement name. These fields can have the same value if you need only one for tracking purposes.

Step 4: Select validation, translation, and functional acknowledgment options

Table 6–1 describes the validation, translation, and functional acknowledgments available when you create an agreement.

Table 6–1   Agreement Options

Option	Description
Validate	Select to enable validation of the document against the configured ECS file.
Translate	Select to enable the translation of XML to native format and vice versa (for EDI and HL7, for example). If **Translate** is not selected (no translation), then B2B cannot correlate the business message with the functional acknowledgment, irrespective of the value of the B2B Handle FA property. See Section C.1, "Properties To Set in Fusion Middleware Control," for information about the property.
Functional Ack	Select to enable the functional acknowledgment for success or error criterion.
FA Handled by B2B	If set to true, then B2B automatically generates the functional acknowledgment (FA) message for inbound EDI and HL7 messages. Inbound FA messages are consumed when this option is true. When this option is set to false, B2B does not automatically generate the FA document. The back-end application (middleware) must generate the FA and provide it to B2B as an outbound message. When option is set to false, inbound FA documents are passed back to the back-end application.
	If the document does not require an FA (as indicated by the agreement-level setting), then this option is ignored. The default value for this property is true.
	See Section C.1, "Properties To Set in Fusion Middleware Control," for more information.
	When **Functional Ack Handled by B2B** is set to false, then **Notify Inbound Functional Acks** must be set to false also for the inbound FA to be sent to the back-end application. If **Notify Inbound Functional Acks** is set to true (while **Functional Ack Handled by B2B** is set to false), then the incoming 997 (FA doc) generates only a notification and the 997 document itself is *not* sent back to the back-end application.
Document Retry Interval	Enter the length of time in minutes between document retries. See Section 5.5.5, "Configuring Delivery Retry Options" for more information about configuring retry attempts.
Document Retry Count	Enter the number of times to retry the message. See Section 5.5.5, "Configuring Delivery Retry Options" for more information about configuring retry attempts.

Step 5: Select the channel for the remote trading partner

A list of channels that you created when you set up the remote trading partner is available. (*Listening* channels are not part of an agreement.)

Step 6: Add identifiers

Identifier types for the host and remote trading partners are listed. Select the identifiers that apply to this agreement. You can shift-click to select multiple identifiers.

For outbound agreements, use the identifier types listed in Table 6–2 with the exchange protocols.

Table 6–2   Identifier Types To Use with Exchange Protocols

Exchange Protocol	Identifier Type
Generic File-1.0	Name
Generic FTP-1.0	Name
Generic SFTP-1.0	Name
Generic AQ-1.0	Name
Generic JMS-1.0	Name
AS2-1.1	Name, AS2 Identifier
AS1-1.0	Name, AS1 Identifier
ebMS-1.0, ebMS-2.0	Name, ebMS Identifier
RosettaNet-V02.00, RosettaNet-01.10	Name, DUNS
MLLP exchange	Name, MLLP ID
Generic HTTP-1.0	Name, Generic Identifier
Generic Email-1.0	Name, Generic Identifier

See Chapter 10, "Creating Types," for more information about identifier types.

Step 7: Save and validate the agreement

Clicking **Save** also validates the agreement.

To create an agreement:

1. Click the **Partners** tab.

2. In the **Agreements** region, click **Add**.

3. Click **Select Partner**.

4. Select a remote trading partner.

5. Click **Select Document Definition**.

6. Select a document definition for the initiator.

7. Provide an agreement ID and name.

8. Select from the validation, translation, and functional acknowledgment options, as described in Table 6–1.

9. Provide an optional description, a callout (if previously created), and start and end dates.

Use callouts to transform the formats of messages exchanged between remote and host trading partners. See Chapter 13, "Managing Callouts."

An agreement cannot be deployed after an end date entered here because the agreement will have expired.

10. For the host trading partner, click **Add** and select identifiers.

11. For the remote trading partner, select a channel.

12. In the remote trading partner, click **Add** and select identifiers.

13. Click **Save**.

After you create an agreement, it is ready to be deployed. It is listed on the **Administration** > **Deploy** page. See Section 6.3, "Deploying an Agreement," to continue.

6.3 Deploying an Agreement

Deployment is the process of activating an agreement from the design-time repository to the run-time repository.

After deploying an agreement, use the **Manage Deployments** tab and the **Reports** tab. See the following for more information:

- Chapter 9, "Managing Deployments"

- Chapter 18, "Creating Reports"

After you create, save, and validate an agreement, you can deploy it as follows:

- From the same page (**Partners** > **Agreement** tab), using the **Deploy** button (see Figure 6–2)

- From the **Administration** > **Deploy** page, as shown in Figure 6–5. Use this option to select multiple agreements to deploy at the same time.

Figure 6–5   The Deploy Tab—Lists Valid Agreements

The image shows the following tabs on the Oracle B2B screen: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Callout, Purge, Listening Channel, and Configuration. The Deploy tab is selected. The first area of the tab shows the following Search fields: Name, Trading Partner, Identification Value, Identification Type, Document Protocol Name, Document Protocol Version, Document Type, and Document Definition. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Agreement, Initiating Partner, Responding Partner, Supported Document, Channel (From and To), State, and Last Deployed.

****

> **Note:**   Turn off validation during deployment by setting the property `b2b.deploy.validation=false`.
>
> This property is set in Oracle Enterprise Manager Fusion Middleware Control. Changing the property requires a SOA Server restart. See Section C.1, "Properties To Set in Fusion Middleware Control," for more information.

To deploy an agreement from the Deploy tab:

1.   Click the **Administration** tab.

2. Click the **Deploy** tab.

3. Use the search parameters to find the agreement you want to deploy and click **Search**.

4. Highlight one or more agreements and click **Deploy**.

6.3.1 Redeploying an Agreement

If you deploy a previously deployed agreement, the first version is moved to an inactive state and the most recently deployed agreement is active.

6.4 Deleting and Exporting Agreements

Only agreements in the draft state can be deleted. Purging an agreement returns its status to the draft state. Agreements that have deployed versions in active, inactive, or retired states cannot be deleted.

An agreement can be exported to a ZIP file by using the **Export** button on the **Agreement** tab.

Part III

Oracle B2B Administration

This part describes how to use Oracle B2B administration features.

This part contains the following chapters:

7

Importing and Exporting Data

This chapter describes how to use the Oracle B2B interface to import and export B2B repositories containing design-time data.

The chapter contains the following topics:

- Section 7.1, "Importing and Exporting the Design-Time Repository"
- Section 7.2, "What Is Copied When You Import or Export from the Import/Export Tab"
- Section 7.3, "About the Exported File"

See Chapter 20, "B2B Command-Line Tools," for information on importing and exporting data using `ant`.

7.1 Importing and Exporting the Design-Time Repository

Oracle B2B design-time data can be exported and saved to a ZIP file. The ZIP file can be imported back into Oracle B2B so that the data is available in the B2B interface. This is useful when migrating data from a test environment to a production environment.

> **Caution:** Do *not* manually edit exported files. If you do so, Oracle B2B cannot guarantee their integrity.

You can also export data from other areas of the Oracle B2B interface:

- Click **Partners** > **Profile** to export trading partner data. See Section 5.2, "Creating Trading Partner Profiles," for more information.
- Click **Partners** and then select an agreement to export. See Section 6.4, "Deleting and Exporting Agreements," for more information.
- Click **Administration** > **Manage Deployments** to export deployed agreements. See Section 9.2.3, "Exporting an Active Agreement," for more information.

You can import sample files that use the following document types: Custom, EDI EDIFACT, EDI X12, HL7, and RosettaNet. See Section 1.7, "Oracle B2B Samples and Cookbooks," for the download location and information about the scenarios presented in the samples.

Figure 7–1 shows the **Import/Export** tab, where you import and export design-time data.

Figure 7–1 Importing and Exporting Data
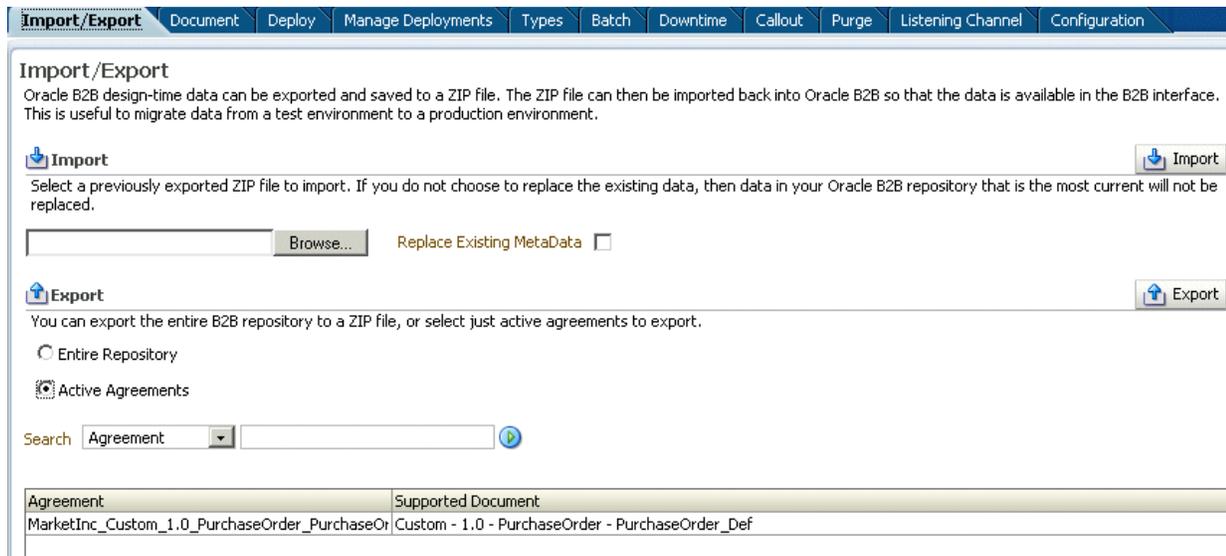
The image shows the following tabs on the Oracle B2B screen: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Import/Export tab is selected. The first area of the tab shows the Import field; to the right is a Browse button. Above the area and to the right is the Import button. The second area of the tab shows the following radio buttons: Entire Repository and Active Agreements. The second area also shows a Search drop-down list. Above the area and to the right is the Export button. The third area of the tab shows a table with the following columns: Agreement and Supported Document.

****

When you import metadata, the updates to your existing B2B are incremental unless you select the **Replace Existing Metadata** option. To delete all existing data before importing metadata, use the **Purge** tab under the **Administration** link. See Chapter 15, "Purging Data," for more information.

> **Caution:** Complete export operations without interruption or idle time. Leaving the browser idle for more than a few minutes during export operations can cause file corruption.

To import data:

1. Click the **Administration** link.

2. Click the **Import/Export** tab.

3. Click **Browse** to find the metadata repository ZIP file.

   The default name for exported metadata is `MDS_EXPORT_DD_MM_YEAR.zip`.

   If you are importing a ZIP file that contains multiple ZIP files within it, you must unzip the containing file and import each ZIP file separately. Individual ZIP files are created when you export multiple agreements at the same time.

4. If you select **Replace Existing Metadata**, then current metadata in the Metadata Service (MDS) repository is overwritten. If it is not selected, then only new data is copied to the MDS repository.

5. Click **Import**.

Depending on the size of the design-time repository contents, this process can take time.

> **Note:** If you export Active Agreements, and if this list has more than one agreement, the export file indicates `all.zip` will consist of many zip files (for example, `a.zip`, `b.zip`, and so on), each consisting of separate Agreements. The underlying ZIP files, `a.zip` and `b.zip`, must be individually imported. If you attempt to import `all.zip`, then the import will return following error:
>
> ```
> Import of file all.zip failed. Error -: B2B-52321: No meta data
> found to import
> ```

To export data:

> **Caution:** Do *not* manually edit exported files.

1. Click the **Administration** link.

2. Click the **Import/Export** tab.

3. Select **Entire Repository** or **Active Agreements**.

The entire repository includes all data in the B2B design-time repository—agreements in all states, all trading partner configurations, and so on.

Active agreements are all deployed agreements that are not inactive, retired, or purged.

4. (Optional) Narrow the list of agreements by using the **Search** option.

   a. Select **Agreement** or **Document Type**.

   b. Enter part or all of an agreement name or document type name and click **Search**.

   c. Click **Search**.

   d. Select one or more agreements from the search results.

      If you select multiple agreements, each agreement is exported in its own ZIP file, and all the individual ZIP files are contained in the export ZIP file.

5. Click **Export**.

6. Select **Open with** or **Save to Disk**.

The system-provided file name is `MDS_EXPORT_DD_MM_YYYY.zip`. As shown in Figure 7–2, you can choose whether you want to open the file or save it, in which case you can specify a file name and download location.

Figure 7–2 Exporting Data
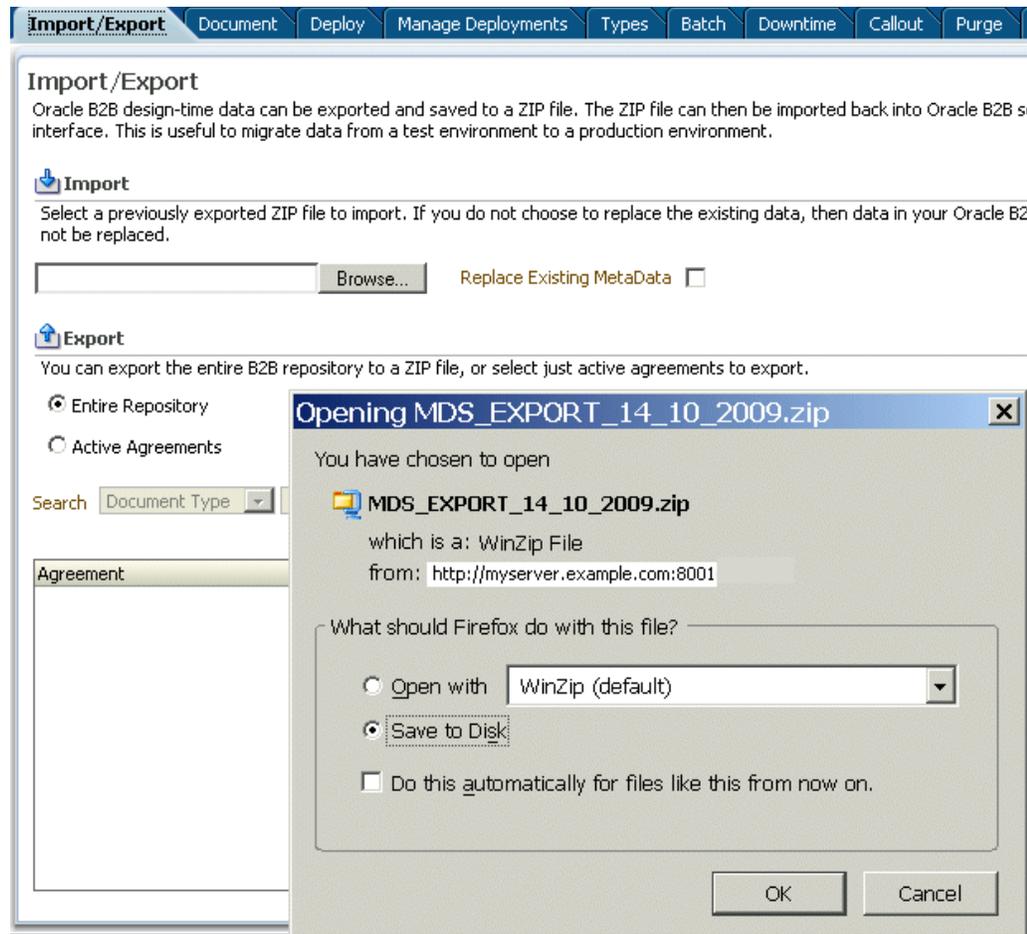
The image shows the File Download dialog box with Open, Save, and Cancel buttons.

****

7.2 What Is Copied When You Import or Export from the Import/Export Tab

Clicking **Import** imports whatever is in the export file (that is, the file that was previously exported), which can possibly include B2BUser and ParameterValue objects. A warning message is displayed to indicate that, if the file contains credential- and policy-related data, then the credential and policy stores must also be imported.

User information—including user permissions for document-type access (see Section 1.4.2, "Restricting Access to Document Types")—is not copied when you export a repository.

ParameterValue objects for passwords are copied when you export a repository.

The B2B import and export functionality is separate from the credential store and policy store import and export functionality. Use the Oracle WebLogic Server tools to import and export identity, credential, and policy stores.

Passwords are not copied when you import a repository. Passwords must be re-created in the destination B2B instance. Passwords are not copied when you export the design-time repository.

Callout library JAR files are not copied during import or export. See Table 13–1, " Callout Details" for more information.

If you export the design-time repository and then continue to make changes to the repository contents in the Oracle B2B interface, and if you later import the exported file (the contents of which are now older), then updates are as follows:

- If **Replace Existing Metadata** is not checked during import, then new data created in the Oracle B2B interface after the file was exported is left untouched.

- If **Replace Existing Metadata** is checked during import, then the existing metadata is replaced with the zip file metadata this option works internally as purge and import).

If an import fails, then the changes are rolled back and the design-time repository remains unchanged. A message appears indicating that the import was unsuccessful.

7.3 About the Exported File

Design-time repository contents that are exported to a file represent a copy of the current data. This file is no longer accessible for changes with the Oracle B2B user interface until it is imported back into Oracle B2B. Do not manually edit exported files.

7.3.1 Exported ZIP Files Containing Agreement Names in Multibyte Character Languages

If you select multiple agreements to export, and any of those agreement names are in a multibyte character language, then in the export ZIP file, which contains a separate ZIP file for each agreement, the ZIP file names for the agreement names in multibyte characters are garbled. This affects your ability to import the ZIP file back into Oracle B2B. Use one of the following approaches for working with this type of file:

- To import a ZIP file containing multiple agreements, in which one or more of the agreement names are in a multibyte character language, use a UTF-8-based unzip tool, such as WinZip version 11.2, to unzip the export file. Then import the individual ZIP files into B2B.

- Alternatively, you can export agreement names that use a multibyte character language one at a time (one per ZIP file). Then import the individual ZIP files as you normally would.

8

Using Document Protocols

This chapter describes the different document protocols supported by Oracle B2B such as EDI EDIFACT, EDI X12, HL7, RosettaNet, and so on.

Oracle B2B document protocols, which define the document type of the message payload are shown in Figure 8–1.

Figure 8–1   Oracle B2B Document Protocols
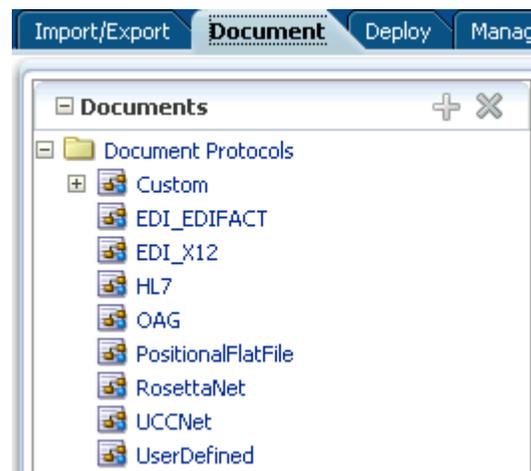
The image shows the following document protocols: Custom, EDI_EDIFACT, EDI_X12, HL7, OAG, PositionalFlatFile, RosettaNet, UCCNET, and UserDefined.

****

You can define nearly any protocol by using the Custom protocol and the many guideline documents in Oracle B2B Document Editor.

This chapter contains the following topics:

- Section 8.1, "Using the Custom Document Protocol"
- Section 8.2, "Using the EDI EDIFACT Document Protocol"
- Section 8.3, "Using the EDI X12 Document Protocol"
- Section 8.4, "Using the HL7 Document Protocol"
- Section 8.5, "Using the OAG Document Protocol"
- Section 8.6, "Using the Positional Flat File Document Protocol"
- Section 8.7, "Using the RosettaNet Document Protocol"

- Section 8.8, "Using the UCCNet Document Protocol"

- Section 8.9, "Changing Document Details"

- Section 8.10, "Using Document Routing IDs"

For related information, see the following:

- Chapter 3, "Creating Guideline Files"

- Chapter 4, "Creating Document Definitions"

8.1 Using the Custom Document Protocol

Oracle B2B supports custom document protocols to create documents needed for proprietary transactions. With XML messages, you have the advantage of schema enforcement (XSDs).

With non-XML messages, you can create trading partner agreements for specific message types.

When creating a Custom document, you specify rules to identify the incoming document. For XML documents, specify an XPath expression and a value, which is the expected result of the expression.

For non-XML documents such as a flat file, you can specify start and end positions or a document routing ID.

Document Version Parameters

No parameters need to be set when you create the document version for a Custom document.

Document Type Parameters

When you create a Custom document type, you can set ebXML messaging service (ebMS) parameters to identify the ebXML document. Figure 8–2 shows the document type parameters for a Custom document.

Figure 8–2   Document Type Parameters for a Custom Document

The image shows a section of the New Document Type screen. The first area of the screen shows the following fields: Document Type Name and Description. The second area of the screen shows the ebMS tab with the following fields: Action name, Service name, Service type, FromRole, ToRole, Vaildate ebMS Header, and CPA File.

****

Table 8–1 describes the document type parameters for a Custom document.

Table 8–1   Document Type Parameters for a Custom Document

Parameter	Description
ebMS Tab	-
Action name	The action name for the ebXML header, which is also an identification criteria for inbound and outbound messages. ebMS documents require an action name to avoid run-time errors.
Service name	The service name for the ebXML header, which is also an identification criteria for inbound messages. ebMS documents require a service name to avoid run-time errors.
Service type	The service type for the ebXML header, which is also an identification criteria for inbound messages. ebMS documents require a service type to avoid run-time errors.
From Role	The trading partner that sends the message. A value provided here overrides the **Identifiers** values supplied on the **Profile** tab.
To Role	The trading partner that receives the message. A value provided here overrides the **Identifiers** values supplied on the **Profile** tab.
Vaildate ebMS Header	When selected, validates inbound ebMS header from role, to role.
CPA File	CPA file

Document Definition Parameters

When you create a Custom document definition, select the file type—XML or Flat—and set parameters in the tabbed areas. Figure 8–3 shows the document definition parameters for an XML-type Custom document.

Figure 8–3   Document Definition Parameters for an XML-Type Custom Document

The image shows the ebMS custom document screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: XML, Routing, XPath, Correlation, and Apps. The XML tab is selected. The tab shows the following fields: Identification Expression (XPath), Identification Value, and DTD/XSD Namespace Conversion. At the top right of the screen is the Save button.

****

Figure 8–4 shows the document definition parameters for a flat-file Custom document.

Figure 8–4   Document Definition Parameters for a Flat-FIle Custom Document

The image shows the ebMS custom document screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Flat, Routing, XPath, Correlation, and Apps. The Flat tab is selected. The tab shows the following information: Identification Start Position, Identification End Position, and Identification Value. At the top right of the screen is the Save button.

****

Table 8–2 describes the document definition parameters for a Custom document.

Table 8–2   Document Definition Parameters for a Custom Document

Parameter	Description
XML Tab	(Available if **XML** is selected from **Identification Type**)
Identification Expression (XPath)	Locates a node in the XML payload
Identification Value	Provides the value to match in the node identified by the Identification Expression. If the values match, then the document is successfully identified. If the value is left blank, then Oracle B2B checks for the existence of the node and the document is successfully identified.
DTD/XSD NamespaceConversion	Select from **None**, **Both**, **Inbound**, or **Outbound**.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document," for more information

Table 8–2   (Cont.) Document Definition Parameters for a Custom Document

Parameter	Description
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload (see Note below table)
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload (see Note below table)
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload (see Note below table)
Correlation Tab	-
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation. (see Note below table)
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation. (see Note below table)
Flat Tab	-
Identification Start Position	Used in combination with the end position to retrieve a value from the payload between the start and end positions
Identification End Position	Used in combination with the start position to retrieve a value from the payload between the start and end positions
Identification Value	A value between the start and end positions
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

> **Note:**   When using EDI documents which have default namespace, the usage of
>
> `//*[local-name()='...']`
>
> can be used, but the more common usage
>
> `//Segment-TH/Field-101-A1/text()`
>
> cannot be used.

8.1.1 How to Configure the XPath Expression for a Custom XML Document

The XPath expression identifies a Custom XML document. You configure the XPath expression when you specify the document type parameters.

The options when configuring an XPath expression are as follows:

- Option 1: Specify the XPath and the Matching Value

- Option 2: Check for the Existence of a Node

- Option 3: Check the Value of an Attribute

8.1.1.1 Option 1: Specify the XPath and the Matching Value

Assume that the transaction ID is 12345. Set the parameters as follows:

Field	Value
Identification Value	12345
Identification Expression	//*[local-name() = 'TransactionID']/text()

Oracle B2B compares the value of **Identification Expression** in the payload to the value specified in **Identification Value**. If the values match, then the document is identified successfully and the corresponding document type and document protocol version are used to identify the agreement. Example 8–1 shows an excerpt of the XML payload for this option.

Example 8–1   Specify the XPath and the Matching Value

```
<?xml version="1.0" encoding="UTF-8" ?>
<Message xmlns:ns1="http://www.example1.org" xmlns:ns2="http://www.example2.org"
  xmlns="http://www.example3.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:ns="http://www.example4.org">
  <MessageHeader>
    <Source>201944019</Source>
    <Destination>205704856</Destination>
    <TransactionID>123456</TransactionID>
    <Version>1-0-0</Version>
  </MessageHeader>
  <Body>
    <ns:Case xsi:schemaLocation="http://www.example4.org" ns1:caseCategoryID="1">
       <ns1:OfficialProvisionNumber>String</ns1:OfficialProvisionNumber>
    </ns:Case>
  </Body>
</Message>
```

8.1.1.2 Option 2: Check for the Existence of a Node

Assume that you are checking for the existence of a node called `registerCommand`. Set the parameters as follows:

Field	Value
Identification Value	*Leave blank.*
Identification Expression	/*[local-name()='envelope']/body/transaction/command/*[local-name()='registerCommand']

When the **Identification Value** field is left blank, Oracle B2B checks for the node identified in **Identification Expression**. If a node in the payload matches, then the document is identified successfully. Example 8–2 shows an excerpt of the XML payload for this option.

Example 8–2   Check for the Existence of a Node

```
<uccnet:envelope xmins:eanucc="http://www.ean-ucc.org/schemas/1.3/eanucc"
```

```
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:uccnet="http://www.uccnet.org/schemas/2.2/uccnet"
        communicationVersion="2.2"
      xsi:schemaLocation="http://www.uccnet.org/schemas/2.2/uccnet
      http://www.testregistry.net/xmlschema/uccnet/2.2/Envelope.xsd">
      <messageHeader>
        <messageIdentifier>
          <value>791:1_EB3CDC749A1F2BABE03014906CC4605A</value>
        </messageIdentifier>
        <userId>oraclesupXSD</userId>
        <representingParty>
          <gin>0060974050142</gin>
        </representingParty>
      </messageHeader>
      <body>
        <transaction>
          <entityIdentification>
            <uniqueCreatorIdentification>856</uniqueCreatorIdentification>
            <globalLocationNumber>
              <gin>0060974050142</gin>
            </globalLocationNumber>
          </entityIdentification>
          <command>
            <uccnet:registerCommand>
              <registerCommandHeader type="ADD" />
            </uccnet:registerCommand>
          </command>
        </transaction>
      </body>
    </uccnet:envelope>
```

8.1.1.3 Option 3: Check the Value of an Attribute

Assume that the value of the country attribute is **US**. Set the parameters as follows:

Field	Value
Identification Value	US
Identification Expression	//*/@country

Oracle B2B compares the value of the country attribute to the value set for
Identification Value. If the values match, then the document is identified successfully.
Example 8–3 shows an excerpt of the XML payload for this option.

Example 8–1   Check the Value of an Attribute

```
<?xml version="1.0" encoding="windows-1252" ?>
<MyAddress country="US" xmlns="http://www.example.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="PO.xsd">
  <name>B2B Buyer</name>
  <street>100 Oracle Parkway</street>
  <city>Redwood City</city>
  <state>CA</state>
  <zip>94065</zip>
</MyAddress>
```

8.2 Using the EDI EDIFACT Document Protocol

Oracle B2B supports message exchanges using UN/EDIFACT, the United Nations Electronic Data Interchange for Administration, Commerce and Transport. These standards prescribe the formats, character sets, and data elements used in purchase orders and invoices.

Oracle B2B supports all versions and document types of EDI EDIFACT, although for some of the newer versions you may need to add the interchange and group guidelines while creating the document version. Table 8–3 lists a few of the transaction sets supported in Oracle B2B.

Table 8–3   Examples of EDI EDIFACT Transaction Sets Supported in Oracle B2B

Set	Description	Version
ORDERS	Purchase Order Message	D98A
ORDRSP	Purchase Order Response Message	D98A
CONTRL	Syntax and Service Report Message	D3

For information about the organization that created and maintains the UN/EDIFACT standards, go to

http://www.unece.org

Document Version Parameters

When you create an EDI EDIFACT document version, you can set various parameters. Figure 8–5 shows document version parameters for an EDI EDIFACT document.

Figure 8–5   Document Version Parameters for an EDI EDIFACT Document

The image shows the EDI_EDIFACT-D98A screen. The first area of the screen shows the following fields: Version Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Interchange, Group, Delimiters, and Miscellaneous. The Interchange tab is selected and shows a number of fields that are described in Table 8–4.

****

Table 8–4 describes the document version parameters for an EDI EDIFACT document.

Table 8–4   Document Version Parameters for an EDI EDIFACT Document

Parameter	Description
Interchange Tab	-
Create UNA	Select from **always**, **never**, or **delimiter-based**. If **delimiter-based** is selected, then UNA is created if the specified delimiters are different from the EDIFACT default value. The **Never** option does not generate UNA for outbound EDIFACT documents, even if nondefault delimiters are used. The **Never** option for inbound messages cannot work for B2B if an EDIFACT document is received without UNA and with nondefault delimiters.
Syntax Identifier	Coded identification of the agency controlling syntax and syntax level used in an interchange. EDI position UNB 010 010 S001 0001. The value **UNOB** is supplied.
Syntax Version Number	Version number of the syntax identified in the syntax identifier (0001). EDI position UNB 010 020 S001 0002. The value **1** is supplied.
Service Code List Directory Version Number	Version number of the service code list directory. EDI position UNB 010 030 S001 0030.
Character Encoding	Coded identification of the character encoding used in the interchange. To be used as specified in the partners' interchange agreement, for the purpose of identifying the character repertoire encoding technique used in the interchange (when the default encoding defined by the character repertoire's associated character set specification is not used). EDI position UNB 010 040 S001 0133.
Interchange Date	Local date when an interchange or a group was prepared. EDI position UNB 030 010 S004 0017. The value **#SystemDate(YYMMDD)#** is supplied.
Interchange Time	Local time of day when an interchange or a group was prepared. EDI position UNB 030 020 S004 0019. The value **#SystemTime(HHMM)#** is supplied.
Recipient's Reference/Password	Reference or password to the recipient's system or to a third-party network as specified in the partners' interchange agreement. To be used as specified in the partners' interchange agreement. It may be qualified by data element 0025. EDI position UNB 060 010 S005 0022.
Recipient's Reference/Password Qualifier	Qualifier for the recipient's reference or password. To be used as specified in the partners' interchange agreement. EDI position UNB 060 020 S005 0025.
Application Reference	Identification of the application area assigned by the sender, to which the messages in the interchange relate; for example, the message type, if all the messages in the interchange are of the same type. Identification of the application area (for example, accounting, purchasing) or of the message type, as applicable. EDI position UNB 070.
Processing Priority Code	Code determined by the sender requesting processing priority for the interchange. To be used as specified in the partners' interchange agreement. EDI position UNB 080.
Interchange Agreement Identifier	Identification by name or code of the type of agreement under which the interchange takes place. Name or code to be specified in the partners' interchange agreement. EDI position UNB 100.
Test Indicator	Indication that the structural level containing the test indicator is a test. EDI position UNB 110.

Table 8–4   (Cont.) Document Version Parameters for an EDI EDIFACT Document

Parameter	Description
Interchange ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file (interchange ecs file of the syntax version number, UNB 010 020) is used.
Group Tab	-
Create Functional Group	Indication of function group (UNG) creation. The value **TRUE** is supplied.
Date of Group Preparation	Local date when an interchange or a group was prepared. EDI position UNG 040 010. The system date stamp is supplied.
Time of Group Preparation	Local time of day when an interchange or a group was prepared. EDI position UNG 040 020. The system time stamp is supplied.
Controlling Agency	Code identifying a controlling agency. EDI position UNG 070 010. The value **UN** is supplied.
Group Association Assigned Code	Code assigned by the association responsible for the design and maintenance of the message type concerned that further identifies the message. EDI position UNG 070 030.
Application Password	Password to the recipient's division, department or sectional application system/process. EDI position UNG 080.
Group ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file is used.
Delimiters Tab	A delimiter is characterized by two levels of separators and a terminator assigned by the sender. Delimiters are also called service characters, data delimiters, or message delimiters. They are specified in the interchange header and cannot be used in a data element value elsewhere in the interchange. In an EDI file, the segment delimiter, the element delimiter, and the subelement delimiter are used. Note: Click **Select Hexadecimal Characters** next to any of the delimiter fields to provide values.
Segment Delimiter	EDIFACT segment delimiter. The value **0x27** is supplied.
Element Delimiter	EDIFACT element delimiter. The value **0x2b** is supplied.
Subelement Delimiter	EDIFACT subelement delimiter. The value **0x3a** is supplied.
Decimal Separator	EDIFACT decimal separator. The value **0x2e** is supplied.
Release Character	EDIFACT release character. The value **0x3f** is supplied.
Replacement Character	EDIFACT replacement character. The value **0x7c** is supplied.
Repeating Separator	EDIFACT repeating separator. The value **0x2a** is supplied.
Miscellaneous Tab	-
Check Duplicate Control Number	When this property is selected (set to true), messages with duplicate interchange control numbers are rejected, meaning that the state of the incoming message is set to ERROR.
Ignore Envelope Parameters	Use this option to provide a list of envelope elements, separated by commas, to be ignored during look-up validation. The possible values depend on the identifiers used in the agreement. Possible values include InterchangeSenderID, InterchangeReceiverID, GroupReceiverID, GroupSenderID, TransactionAssociationAssignedCode, InterchangeReceiverQual, InterchangeSenderQual, and InterchangeControlVersion.

Document Type Parameters

When you create an EDI EDIFACT document type, you can set various parameters. Figure 8–6 shows the document type parameters for an EDI EDIFACT document.

Figure 8–6   Document Type Parameters for an EDI EDIFACT Document

The image shows the EDI_EDIFACT-D98A-ORDERS screen. The first area of the screen shows the following fields: Document Type Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the Transaction tab and displays the following fields: Functional Group Identifier Code, Controlling Agency, Transaction Association Assigned Code, and Common Access Reference. At the top right of the screen are the Save and Reset buttons.

****

Table 8–5 describes the document type parameters for an EDI EDIFACT document.

Table 8–5   Document Type Parameters for an EDI EDIFACT Document

Parameter	Description
Transaction Tab	-
*Functional Group Identifier Code	Code identifying one type of message in a functional group. EDI position UNG 010 0038. Required.
Controlling Agency	Code identifying the agency controlling the specification, maintenance and publication of the message type. EDI position UNH 020 040 S009 0051.
Transaction Association Assigned Code	Code, assigned by the association responsible for the design and maintenance of the message type concerned, which further identifies the message. EDI position UNH 020 050 S009 0057.
Common Access Reference	Reference serving as a key to relate all subsequent transfers of data to the same business case or file. EDI position UNH 030 0068.

Document Definition Parameters

When you create an EDI EDIFACT document definition, you can set various parameters. Figure 8–7 shows document definition parameters for an EDI EDIFACT document.

Figure 8–7 Document Definition Parameters for an EDI EDIFACT Document

The image shows the EDI_EDIFACT Document Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Transaction, Routing, XPath, Correlation, Apps, and EDIEL. The Transaction tab is selected and shows the Transaction Set ecs File field. At the top right of the screen is the Save button.

****

Table 8–6 describes the document definition parameters for an EDI EDIFACT document.

Table 8–6   Document Definition Parameters for an EDI EDIFACT Document

Parameter	Description
Transaction Tab	-
*Transaction Set ecs File	Use the **Browse** button to select the ecs file.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See "How to Configure the XPath Expression for a Custom XML Document" on page 8-6 for more information.
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-

Table 8–6 (Cont.) Document Definition Parameters for an EDI EDIFACT Document

Parameter	Description
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.
EDIEL Tab	-
FA Assoc Assigned Code	Code for the functional acknowledgment
FA Message Version Number	Version number for the functional acknowledgment
FA Message Release Number	Release number for the functional acknowledgment
Remove FA Segments	Remove functional acknowledgment segments
Map Application Reference	Maps the `Application reference` field in the interchange envelope of the incoming EDIEL message to the `Application reference` field in the corresponding outbound CONTRL (FA) message.

8.3 Using the EDI X12 Document Protocol

Oracle B2B supports message exchanges using American National Standards Institute (ANSI) X12. These standards prescribe the formats, character sets, and data elements used in documents such as purchase orders and invoices.

Oracle B2B supports all versions and document types of EDI X12, although for some of the newer versions you may need to add the interchange and group guidelines while creating the document version. Table 8–7 lists a few of the transaction sets supported in Oracle B2B.

Table 8–7 Examples of EDI X12 Transaction Sets Supported in Oracle B2B

Set	Description	Version
850	Purchase Order	4010
855	Purchase Order Acknowledgment	4010
997	Functional Acknowledgment	4010

For information about the organization that created and maintains the ANSI X12 standards, go to

http://www.ansi.org

Document Version Parameters

When you create an EDI X12 document version, you can set various parameters. Figure 8–8 shows document version parameters for an EDI X12 document.

Figure 8–8   Document Version Parameters for an EDI X12 Document

The image shows the EDI_X12-4010 Document Version Parameters screen. The first area of the screen shows the following fields: Version Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Interchange, Group, Delimiters, and Miscellaneous. The Interchange tab is selected and shows several fields described in Table 8–8. At the top right of the screen is the Save button.

****

Table 8–8 describes the document version parameters for an EDI X12 document.

Table 8–8   Document Version Parameters for an EDI X12 Document

Parameter	Description
Interchange Tab	-
Authorization Information Qualifier	Code to identify the type of information in the authorization information. EDI position ISA 01. The value **00** is supplied.
Authorization Information	Information used for additional identification or authorization of the sender or the data in the interchange. The authorization information qualifier sets the type of information. EDI position ISA 02.
Security Information Qualifier	Code to identify the type of information in the security information. EDI position ISA 03. The value **00** is supplied.
Security Information	Information used to identify the security information about the interchange sender or the data in the interchange. The security information qualifier sets the type of information. EDI position ISA 04.
Interchange Date	Date of the interchange. EDI position ISA 09. The system date stamp is supplied (`#SystemDate(YYMMDD)#`).
Interchange Time	Time of the interchange. EDI position ISA 10.The system time stamp is supplied (`#SystemTime(HHMM)#`).

Table 8–8 (Cont.) Document Version Parameters for an EDI X12 Document

Parameter	Description
Interchange Control Standard/Repetition Separator	Code to identify the agency responsible for the control standard used by the message that is enclosed by the interchange header and trailer. EDI position is ISA 11. The value **U** is supplied.
*Interchange Control Version Number	Code specifying the version number of the interchange control segments. EDI position ISA 12. The value **00401** is supplied.
Usage Indicator	Code to indicate whether data enclosed by this interchange envelope is in test or production. EDI position ISA 15. The value **P**, for production, is supplied.
Interchange ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file (interchange ecs file of the interchange control version, ISA 12) is used.
Group Tab	-
Functional Group Date	Date sender generated a functional group of transaction sets. EDI position GS 04. The system date stamp is supplied (`#SystemDate(CCYYMMDD)#`).
Functional Group Time	Time when the sender generated a functional group of transaction sets (local time at sender's location). EDI position GS 05.The system time stamp is supplied (`#SystemTime(HHMM)#`).
Responsible Agency Code	Code used in conjunction with data element 480 to identify the issuer of the standard. EDI position GS 06. The value **X** is supplied.
Version/Release/Industry Identifier Code	Code indicating the version, release, subrelease, and industry identifier of the EDI standard being used, including the GS and GE segments; if the code in DE455 in GS segment is X, then in DE 480 positions 1-3 are the version number; positions 4-6 are the release and subrelease, level of the version; and positions 7-12 are the industry or trade association identifiers (optionally assigned by user); if the code in DE455 in GS segment is T, then other formats are allowed.
Group ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file (group ecs file of EDI X12 version) is used.
Delimiters Tab	Click **Select Hexadecimal Characters** next to any of the delimiter fields to provide values. See Table 8–4 for more about delimiters.
Segment Delimiter	The value **0x7e** is supplied.
Element Delimiter	The value **0x2a** is supplied.
Subelement Delimiter	The value **0x5c** is supplied.
Decimal Separator	The value **0x2e** is supplied.
Replacement Character	The value **0x7c** is supplied.
Repeating Separator	The value **0x5e** is supplied.
Miscellaneous Tab	-
Check Duplicate Control Number	When this property is selected (set to true), messages with duplicate interchange control numbers are rejected, meaning that the state of the incoming message is set to ERROR.
Ignore Envelope Parameters	Use this option to provide a list of envelope elements, separated by commas, to be ignored during look-up validation. The possible values depend on the identifiers used in the agreement. Possible values include InterchangeSenderID, InterchangeReceiverID, GroupReceiverID, GroupSenderID, TransactionAssociationAssignedCode, InterchangeReceiverQual, InterchangeSenderQual, and InterchangeControlVersion.

Document Type Parameters

When you create an EDI X12 document type, you can set various parameters. Figure 8–9 shows the document type parameters for an EDI X12 document.

Figure 8–9   Document Type Parameters for an EDI X12 Document
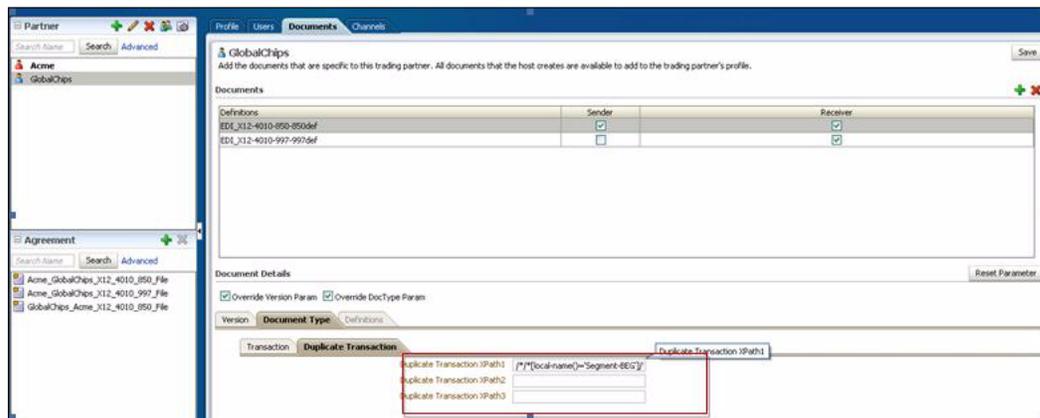
The image shows the EDI_X12-4010-850 Document Type Confirmation screen. The first area of the screen shows the following fields: Document Type Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the Transaction tab and shows the following fields: Functional Group Identifier Code, Implementation Convention Reference, and Transaction Purpose Code. At the top right of the screen are the Save and New Definition buttons.

****

Table 8–9 describes the document type parameters for an EDI X12 document.

Table 8–9   Document Type Parameters for an EDI X12 Document

Parameter	Description
Transaction Tab	-
*Functional Group Identifier Code	Uniquely identifies a transaction set GS 01. Required.
Implementation Convention Reference	Reference assigned to identify Implementation Convention. EDI position ST 03.
Transaction Purpose Code	Code identifying the purpose of the transaction set. EDI position BEG/BGN 01.
Duplicate Transaction Tab	This feature enables you to detect the duplicate transactions by considering specific content in the payload (such as Purchase Order number and invoice number).
	This is achieved by providing the XPath for the specific tag of the payload. The three XPath fields provide the flexibility to arrive at the uniqueness criteria with multiple values of the payload.
	For inbound, this adds an additional check during the Oracle B2B inbound message processing to check duplicate control numbers. The XPath-based duplicate check is done after the control number check.
	Note: You cannot specify the second/third XPath without specifying the first XPath

Document Definition Parameters

When you create an EDI X12 document definition, you can set various parameters. Figure 8–10 shows document definition parameters for an EDI X12 document.

Figure 8–10   Document Definition Parameters for an EDI X12 Document

The image shows the EDI_X12-4010-850 New Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Transaction, Routing, XPath, Correlation, and Apps. The Transaction tab is selected and shows the Transaction Set ecs File field. At the top right of the screen is the Save button.

****

Table 8–10 describes the document definition parameters for an EDI X12 document.

Table 8–10   Document Definition Parameters for an EDI X12 Document

Parameter	Description
Transaction Tab	-
Transaction Set ecs File	Use the **Browse** button to select the ecs file.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document," for more information.
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-

Table 8–10   (Cont.) Document Definition Parameters for an EDI X12 Document

Parameter	Description
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

> **Note:**   For information about behavior and limitations when the `b2b.FAHandledByB2B` property is set to false, see Appendix C, "Setting B2B Configuration Properties in Fusion Middleware Control."

8.4 Using the HL7 Document Protocol

Oracle B2B implements the Health Level 7 (HL7) version 2.x and version 3 standards (version 3 supports Custom document protocols) to exchange documents containing health care information using the Generic exchange or MLLP exchange. When using HL7, the standard Oracle B2B features, such as validation, translation, automatic generation of outbound envelope headers, and acknowledgments, are available.

> **Note:**   While HL7 BATCH and FILE envelopes are supported, batching is not supported in this release.

For information about the organization that created and maintains the HL7 standards, go to

`http://www.hl7.org`

Document Version Parameters

When you create an HL7 document version, you can set various parameters. Figure 8–11 shows document version parameters for an HL7 document.

Figure 8–11   Document Version Parameters for an HL7 Document

The image shows the HL7 Document Version Parameters screen. The first area of the screen shows the following fields: Version Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Message Header, Batch Header, File Header, Delimiters, and Miscellaneous. The Message Header tab is selected and displays several fields described in Table 8–11. At the top right of the screen is the Save button.

****

Table 8–11 describes the document version parameters for an HL7 document.

Table 8–11   Document Version Parameters for an HL7 Document

Parameter	Description
Message Header Tab	-
Security	In some applications of HL7, this field is used to implement security features.
Processing ID	MSH.11 - This field is used to decide whether to process the message as defined in HL7 Application (level 7) processing rules. The first component defines whether the message is part of a production, training, or debugging system (refer to HL7 table 0103 - Processing ID for valid values). The second component defines whether the message is part of an archival process or an initial load (refer to HL7 table 0207 - Processing mode for valid values). This allows different priorities to be given to different processing modes.
Accept Acknowledgement Type	Sets the conditions under which application acknowledgments are required to be returned in response to the message. The value **AL** (always) is supplied.
	B2B checks the payload (MSH.15) of an incoming message to see if an ACK has to be generated. In some HL7 Systems, MSH.15 is not sent in the payload at all and it is expected that an ACK is still sent.

Table 8–11   (Cont.) Document Version Parameters for an HL7 Document

Parameter	Description
Application Acknowledgment Type	MSH.16. The value **AL** (always) is supplied.
Country Code	Sets the country of origin for the message. The value US is supplied.
Character Set	Sets the character set for the entire message. The value ASCII is supplied.
Internationalization Code Identifier	MSH.19
Internationalization Code Text	MSH.19
Internationalization Coding System Name	MSH.19
Internationalization Code Alternate Identifier	MSH.19
Internationalization Code Alternate Text	MSH.19
Internationalization Code Alternate Coding System Name	MSH.19
International Version Identifier	MSH.12
International Version ID Text	MSH.12
International Version ID Coding System Name	MSH.12
International Version ID Alternate Identifier	MSH.12
International Version ID Alternate Text	MSH.12
International Version ID Alternate Coding System Name	MSH.12
Batch Header Tab	-
Create Batch Header	Check the box to create batch headers.
Batch Header ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file is used.
Batch Security	BHS.8
Batch Date	BHS.7. The system date-time stamp is supplied (`#SystemDateTime(CCYYMMDDHHMM)#`).
File Header Tab	-
Create File Header	Check the box to enable.
File Header ecs File	Use the **Browse** button to find an ecs file to override the standard file. If not provided, the B2B-provided default file is used.
File Security	FHS.8
File Date	FHS.7. The system date-time stamp is supplied (`#SystemDateTime(CCYYMMDDHHMM)#`).

Table 8–11   (Cont.) Document Version Parameters for an HL7 Document

Parameter	Description
Delimiters Tab	Click **Select Hexadecimal Characters** next to any of the delimiter fields to provide values. See Table 8–4 for more about delimiters.
Element Delimiter	A single character that follows the segment identifier and separates each data element in a segment except the last. The value **0x7c** is supplied.
Escape Character	The value **0x5c** is supplied.
Repeating Separator	A service character used to separate adjacent occurrences of a repeating data element, or to separate multiple occurrences of a field.The value **0x7e** is supplied.
Segment Delimiter	A syntax character indicating the end of a segment (a logical grouping of data fields) within a message. The value **0x0d** is supplied.
Subcomponent Delimiter	The value 0x26 is supplied.
Subelement Delimiter	The value 0x5e is supplied.
Miscellaneous Tab	-
Ignore Envelope Parameters	Use this option to provide a list of envelope elements, separated by commas, to be ignored during look-up validation. The possible values depend on the identifiers used in the agreement. For an HL7 agreement, the possible values include MessageSendingApp, MessageReceivingApp, MessageSendingFacility, and MessageReceivingFacility.

Document Type Parameters

When you create an HL7 document type, you can set various parameters. Figure 8–12 shows the document type parameters for an HL7 document.

Figure 8–12   Document Type Parameters for an HL7 Document

The image shows the HL7 Document Type screen. The first area of the screen shows the following fields: Document Type Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the Transaction tab and displays the following fields: HL7 Generic ACK (a check box), Map ACK Control

ID (a check box), and Accept Acknowledgement (a drop-down list) At the top right of the screen are the Save and New Definition buttons.

****

Table 8–12 describes the document type parameters for an HL7 document.

Table 8–12   Document Type Parameters for an HL7 Document

Parameter	Description
Transaction Tab	-
HL7 Generic ACK	If selected, Oracle B2B sends a generic ACK immediately upon receiving an HL7 message.
Map ACK Control ID	Select to enable mapping the MSH.10 of the business message to the MSH.10 of the acknowledgment. **Note:** This Map ACK Control ID parameter is for the functional ACK.
Accept Acknowledgement	A functional acknowledgment is generated when MSH.15 has no value. Select **None** to take no action. Acknowledgment generation is dependent on the value in MSH.15 of the business message. Select **AL** (always) to generate the acknowledgment under any conditions. Select **ER** (error/reject) to generate the acknowledgment when the message errors or is rejected. Select **SU** (successful completion) to generate the acknowledgment when the message is successfully processed.

Document Definition Parameters

When you create an HL7 document definition, you can set various parameters. Figure 8–13 shows document definition parameters for an HL7 document.

Figure 8–13   Document Definition Parameters for an HL7 Document

The image shows the HL7 New Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The second area of the screen shows the following tabs: Transaction,

Routing, XPath, Correlation, and Apps. The Transaction tab is selected and shows the following field: Transaction Set ecs File.

****

Table 8–13 describes the document definition parameters for an HL7 document.

Table 8–13   Document Definition Parameters for an HL7 Document

Parameter	Description
Transaction Tab	-
*Transaction Set ecs File	Use the **Browse** button to find the ecs file.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document," for more information.
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

About Using HL7

- No business message is produced for an HL7 immediate acknowledgment (transport-level acknowledgment). When using AS2, you see one acknowledgment business message for MDN (transport-level acknowledgment), and for ebMS, you see one acknowledgment business message in the business message report. In summary, because immediate acknowledgments are sent at the transport level, the entry is available only in the wire message report and not in the business message report.

- Negative acknowledgment messages indicating errors in an HL7 exchange may be truncated because of the 80-character length limitation in HL7 versions 2.1 through 2.5.

8.5 Using the OAG Document Protocol

Oracle B2B implements Open Applications Group (OAG) standards, a robust XML standard used across many industries. This standard defines messages as business object documents (BODs).

For information about the organization that created and maintains the OAG standards, go to

http://www.oagi.org

Document Version Parameters

No parameters need to be set when you create the document version for an OAG document.

Document Type Parameters

When you create an OAG document type, you can set various parameters. Figure 8–14 shows the document type parameters for an OAG document.

Figure 8–14   Document Type Parameters for an OAG Document

The image shows the OAG Document Type screen. The first area of the screen shows the following fields: Document Type Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the Control Area tab and displays the following fields: Logical Identifier, Component, Task, FA on Error, Language, Code Page, Authorization Identifier, and Date Time Qualifier. At the top right of the screen are the Save and New Definition buttons.

****

Table 8–14 describes the document type parameters for an OAG document.

Table 8–14   Document Type Parameters for an OAG Document

Parameter	Description
Control Area Tab	-
Logical Identifier	Logical Identifier
Component	Component
Task	Task
FA on Error	When enabled, CONFIRMATION flag is set to 1.
Language	Language
Code Page	Code Page
Authorization Identifier	Authorization Identifier
Date Time Qualifier	Date Time Qualifier attribute

Document Definition Parameters

When you create an OAG document definition, you can set various parameters. Figure 8–15 shows document definition parameters for an OAG document.

Figure 8–15   Document Definition Parameters for an OAG Document

The image shows the OAG-7.2.1-PROCESS_PO_007--New Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The second area of the screen shows the following tabs: XML, Routing, XPath, Correlation, and Apps. The XML tab is selected and shows the following fields: Identification Expression (XPath), Identification Value, and DTD/XSD Namespace Conversion.

****

Table 8–15 describes the document definition parameters for an OAG document.

Table 8–15   Document Definition Parameters for an OAG Document

Parameter	Description
XML Tab	-
Identification Expression (XPath)	Locates a node in the XML payload
Identification Value	Provides the value to match in the node identified by the identification expression. If the values match, then the document is successfully identified. If the value is left blank, then Oracle B2B checks for the existence of the node and the document is successfully identified.
DTD/XSD Namespace Conversion	Select from **None**, **Both**, **Inbound**, or **Outbound**.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document," for more information.
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

8.6 Using the Positional Flat File Document Protocol

Oracle B2B supports message exchange for positional flat files, for example, NCPDP Telecom documents and SAP iDocs (intermediate documents (text files) used with SAP applications). This adds capabilities beyond handling XML files and traditional EDI files based on various XML and EDI standards.

> **Note:**   Positional flat files must be manually upgraded from 10*g* to 11*g*.

Document Version Parameters

No parameters need to be set when you create the document version for a positional flat file.

Document Type Parameters

No parameters need to be set when you create the document type for a positional flat file.

Document Definition Parameters

When you create a document definition for a positional flat file, you can set various parameters. Figure 8–16 shows document definition parameters for a positional flat file.

Figure 8–16   Document Definition Parameters for a Positional Flat File (including SAP iDocs)

The image shows the PositionalFlatFile Document Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: Transaction, Identification, Routing, XPath, Correlation, and Apps. The Transaction tab is selected and shows the Transaction Set ecs File field. At the top right of the screen is the Save button.

****

Table 8–16 describes the document definition parameters for a positional flat file.

Table 8–16   Document Definition Parameters for a Positional Flat File

Parameter	Description
Transaction Tab	-
*Transaction Set ecs File	Use the **Browse** button to find the ecs file.
Identification Tab	-

Table 8–16   (Cont.) Document Definition Parameters for a Positional Flat File

Parameter	Description
Identification Value	Not applicable
Identification Start Position	Used in combination with the end position to retrieve a value from the payload between the start and end positions
Identification End Position	Used in combination with the start position to retrieve a value from the payload between the start and end positions
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document," for more information.
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

8.7 Using the RosettaNet Document Protocol

Oracle B2B implements the nonproprietary, XML-based RosettaNet standards to exchange documents over the Internet. RosettaNet standards prescribe when information should be exchanged, acknowledged, or confirmed, and how messages in an exchange should be packaged and physically exchanged between trading partners. In addition to using the RosettaNet document guideline files in Oracle B2B Document Editor, you can also download standard DTD files from the RosettaNet Web site.

A RosettaNet DTD, when used with Oracle B2B in a SOA composite application, must be converted to an XSD. An AQ Adapter added to the composite application can convert the inbound DTD to an XSD and manipulate the data as needed. Likewise, the AQ Adapter can convert the outbound XSD to a DTD for Oracle B2B to send the message out.

RosettaNet standards are specified by using of the RosettaNet Partner Interface Process (PIP), RosettaNet Dictionaries, and RNIF. Oracle B2B supports all PIPs. (The RosettaNet Technical Dictionary is not supported in Oracle B2B.)

For information about the RosettaNet consortium and its history, and for a complete list of PIP clusters and segments, go to

http://www.rosettanet.org

8.7.1 PIPs

A PIP is an XML-based dialog that defines the business processes between trading partners. It defines the structure, sequence of steps, roles (buyer and seller) activities, data elements, values, and value types for each business document message exchanged between trading partners.

Using PIP 3A4 as an example, you can see how a PIP defines a dialog between trading partners, as shown in Figure 8–17.

Figure 8–17   PIP 3A4 Message Exchange Between Buyer and Seller
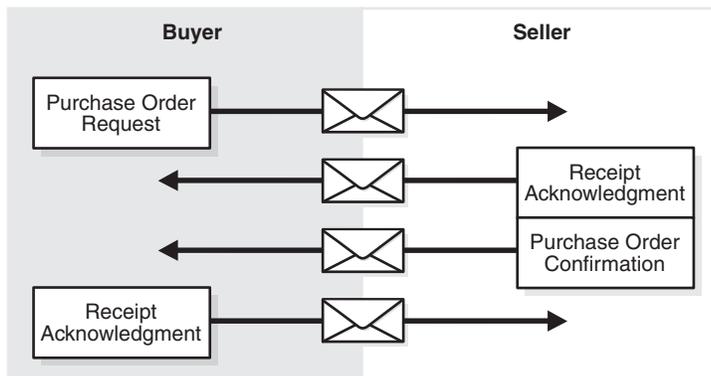
The image shows a PIP3A4 message being exchanged. First, the buyer sends the purchase order request to the seller. Then seller sends a receipt acknowledgment. Next, the seller sends a purchase order confirmation. Finally, the buyer sends a receipt acknowledgment.

****

A PIP sequence combines a cluster, segment, and type. The PIP sequence 3A4, for example, encodes the information shown in Table 8–17.

Table 8–17   PIP 3A4 Breakdown

Element	Description
3	Order manage *cluster*, with which trading partners can: ■ Order catalog products ■ Create custom orders ■ Manage product distribution and delivery ■ Support product returns and financial transactions
3A	Quote and order entry *segment*

Table 8–17   (Cont.) PIP 3A4 Breakdown

Element	Description
3A4	Specific PIP *type*, which supports:
	■ Submittal of a purchase order by a buyer
	■ Submittal of an acceptance purchase order by a seller
	■ Ability of a buyer to cancel or change a purchase order based on the acknowledgment response

Document Version Parameters

No parameters need to be set when you create the document version for a RosettaNet document.

Document Type Parameters

When you create a RosettaNet document type, you can set various parameters. Figure 8–18 shows document type parameters for a RosettaNet document.

Figure 8–18   Document Type Parameters for a RosettaNet Document

The image shows the RosettaNet Document Type screen. The first area of the screen shows the following fields: Document Type Name and Description. The first area also shows the Reset Parameter button. The second area of the screen shows the Service Header tab and displays the following fields: From Role, To Role, From Service, To Service, Business Transaction Name, Business Action, Time to perform for Collaboration, Collaboration Name, and Collaboration Code. At the top right of the screen are the Save and New Definition buttons.

****

Table 8–18 describes document type parameters for a RosettaNet document.

Table 8–18   Document Type Parameters for a RosettaNet Document

Parameter	Description
Service Header Tab	-
*From Role	The trading partner that sends the message (in Partner Role Description of the PIP).
*To Role	The trading partner that receives the message (the role the trading partner receiving the message plays in the PIP).
*From Service	The service that sends the message.
*To Service	The service to which the message is sent.
*Business Transaction Name	The name of the business transaction is required.
*Business Action	The name of the business action is required. The value must be consistent with the Global Business Action Code.
*Time to Perform for Collaboration	The time to perform the business action is required.
*Collaboration Name	The RosettaNet collaboration name signifies the business transaction between trading partners (the roles as buyer and seller) depending on a common transaction. Required.
*Collaboration Code	The textual form of the abbreviated collaboration name. Required.

Document Definition Parameters

When you create a RosettaNet document definition, you can set various parameters. Figure 8–19 shows the document definition parameters for a RosettaNet document.

Figure 8–19   Document Definition Parameters for a RosettaNet Document

The image shows the RosettaNet Document Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and Root XSD Name. The first area also shows the Reset Parameter button. The second

area of the screen shows the following tabs: Parameters, XPath, Correlation, and Apps. The Parameters tab is selected and shows the following fields: Document Routing ID and DTD/XSD Namespace Conversion (a drop-down list). At the top right of the screen is the Save button.

****

Table 8–19 describes the document definition parameters for a RosettaNet document.

Table 8–19   Document Definition Parameters for a RosettaNet Document

Parameter	Description
Parameters Tab	-
Document Routing ID	Sets the consumer name to the back-end application
DTD/XSD Namespace Conversion	A converted document can optionally replace the original RosettaNet document. Select **Both** to replace the RosettaNet document with the converted document for both the inbound and outbound messages. Select **Inbound** to replace the RosettaNet document with the converted document for the inbound message. Select **Outbound** to replace the RosettaNet document with the converted document for the outbound message. Select **None** for no replacement. None passes the DTD instance as-is. Inbound converts the instance DTD to XSD. Outbound converts the instance XSD to DTD. Both convert both inbound and outbound formats.
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document,"
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	Correlation is required for a two-action PIP, for example, a 3A4.
Correlation From XPath Name	The name of the correlation property for initiating the correlation. For example, Pip3A4PurchaseOrderRequest in /*[local-name()='Pip3A4PurchaseOrderRequest']/*[local-name()='thisDocumentIdentifier']/text().
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation. Correlation-to represents the other message that takes part in the correlation. For example, Pip3A4PurchaseOrderConfirmation in/*[local-name()='Pip3A4PurchaseOrderConfirmation']/*[local-name()='requestingDocumentIdentifier']/text().
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

8.7.2 RosettaNet Validation

RosettaNet validation compares the elements in RosettaNet XML-format business documents to the requirements specified in the RosettaNet Message Guideline specification to determine their validity. This specification defines requirements for details such as element datatypes, element lengths, element value lists, and element cardinality. PIPs that require RosettaNet dictionary validation are also validated when a dictionary is present.

The minimum validation-level requirements on the sections of a RosettaNet XML-format business document are as follows. These requirements cover the preamble, delivery header, service header, and service content sections of a document. Documents not following one or more of these requirements are identified as invalid.

1. The XML-format business document requires compliance with its DTD.

2. Elements with datatypes, lengths, or both that are specified in the RosettaNet Message Guideline specification require validation against this specification.

3. An element's list of values specified in the entity instance list in the corresponding RosettaNet Message Guideline specification requires validation against this specification.

4. If the Message Guideline specification defines the cardinality specification of an element differently from the corresponding DTD specification, the Message Guideline specification takes precedence.

5. If a PIP requires dictionary validation, and a dictionary is included, the service content requires validation against the dictionary as a part of action performance.

6. Cross-tag validation is based on message guidelines.

8.8 Using the UCCNet Document Protocol

Oracle B2B implements UCCNet, which enables trading partners—typically retailers and suppliers in the retail and consumer goods industries—to exchange documents with UCCNet. Table 8–20 lists the UCCNet document types supported in Oracle B2B.

Table 8–20   UCCNet Document Types

Standard
registerCommand
confirmCommand
linkCommand
checkComplianceCommand
documentCommand
documentIdentificationCommand
notificationStateCommand
queryCommand
registerLinkCommand
publicationCommand
publishCommand
catalogueItemMaintenanceCommand
priceCommand

Table 8–20   (Cont.) UCCNet Document Types

Standard
validateCommand
registerOwnershipCommand
subscriptionCommand
notifyCommand
response

For information about the organization that created and maintains the UCCNet standards, go to

http://www.1sync.org

Document Version Parameters

No parameters need to be set when you create the document version for a UCCNet document.

Document Type Parameters

No parameters need to be set when you create the document type for a UCCNet document.

Document Definition Parameters

When you create a UCCNet document definition, you can set various parameters. Figure 8–20 shows document definition parameters for a UCCNet document.

Figure 8–20   Document Definition Parameters for a UCCNet Document

The image shows the UCCNet Document Definition screen. The first area of the screen shows the following fields: Document Definition Name, Description, Definition, and

Root XSD Name. The first area also shows the Reset Parameter button. The second area of the screen shows the following tabs: XML, Routing, XPath, Correlation, and Apps. The XML tab is selected and shows the following fields: Identification Expression (XPath) and Identification Value. At the top right of the screen is the Save button.

****

Table 8–21 describes the document definition parameters for a UCCNet document.

Table 8–21   Document Definition Parameters for a UCCnet Document

Parameter	Description
XML Tab	-
Identification Expression (XPath)	Locates a node in the XML payload
Identification Value	Provides the value to match in the node identified by the Identification Expression. If the values match, then the document is successfully identified. If the value is left blank, then Oracle B2B checks for the existence of the node and the document is successfully identified.
Routing Tab	-
Document Routing ID	Sets the consumer name to the back-end application
XPath Tab	See Section 8.1.1, "How to Configure the XPath Expression for a Custom XML Document,"
XPath Name1	The XML XPath name for retrieving the value from the payload
XPath Expression1	The XML XPath expression for retrieving the value from the payload
XPath Name2	The XML XPath name for retrieving the value from the payload
XPath Expression2	The XML XPath expression for retrieving the value from the payload
XPath Name3	The XML XPath name for retrieving the value from the payload
XPath Expression3	The XML XPath expression for retrieving the value from the payload
Correlation Tab	-
Correlation From XPath Name	The name of the correlation property for initiating the correlation.
Correlation From XPath Expression	The XML XPath for retrieving the value from the payload to initiate the correlation.
Correlation To XPath Name	The name of the correlation property for the correlation.
Correlation To XPath Expression	The XML XPath for retrieving the value from the payload for the correlation.
Apps Tab	-
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

8.8.1 Creating a 1Sync Document

The 1Sync document protocol helps in the data synchronization between seller and buyer, which enables the transfer of product and location information with the continuous synchronization of the data over time.

Use the Custom document protocol or the UCCNet document protocol to create a 1Sync XML document.

> **Note:**   The GS-1 organization has changed the standard name from UCCNet to 1Sync. Use either the seeded UCCNet document protocol or create a new Custom document protocol, 1Sync, as illustrated in the figure. The functionality is the same.

Figure 8–21 shows a document definition for a 1Sync document, using the Custom document protocol.

Figure 8–21   1Sync Document Definition
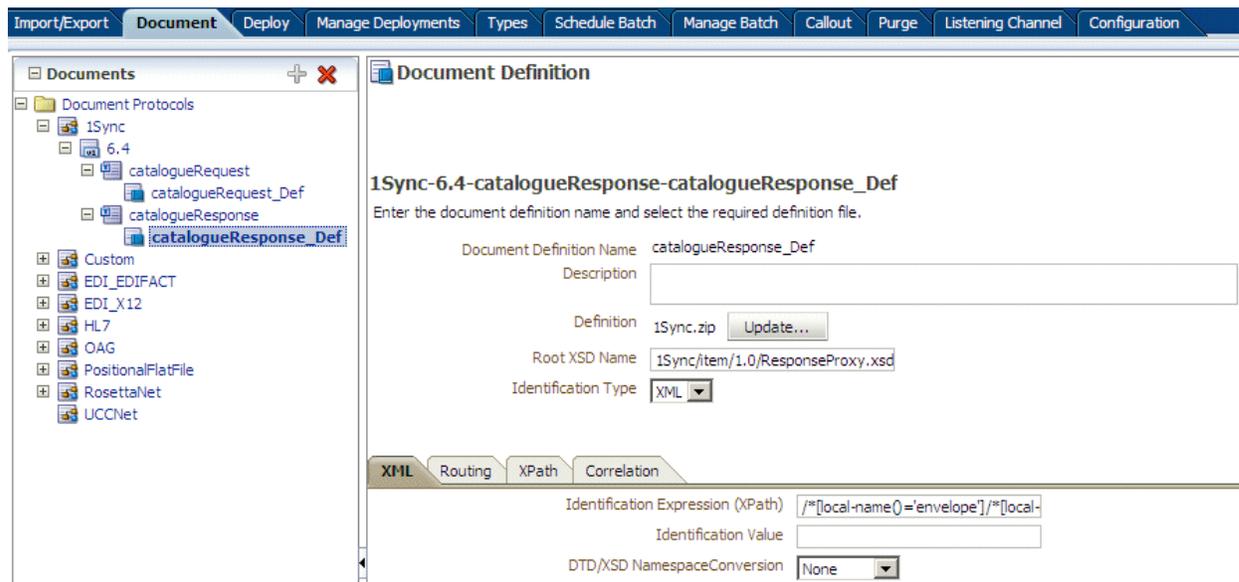
The image is described in the surrounding text.

****

You can correlate 1Sync request and response messages as follows:

- Use the document routing ID on the **Routing** tab. The routing ID **1Sync_64_catalogueRequest** is shown Figure 8–22.

Figure 8–22   The Routing Tab for a 1Sync Document Definition
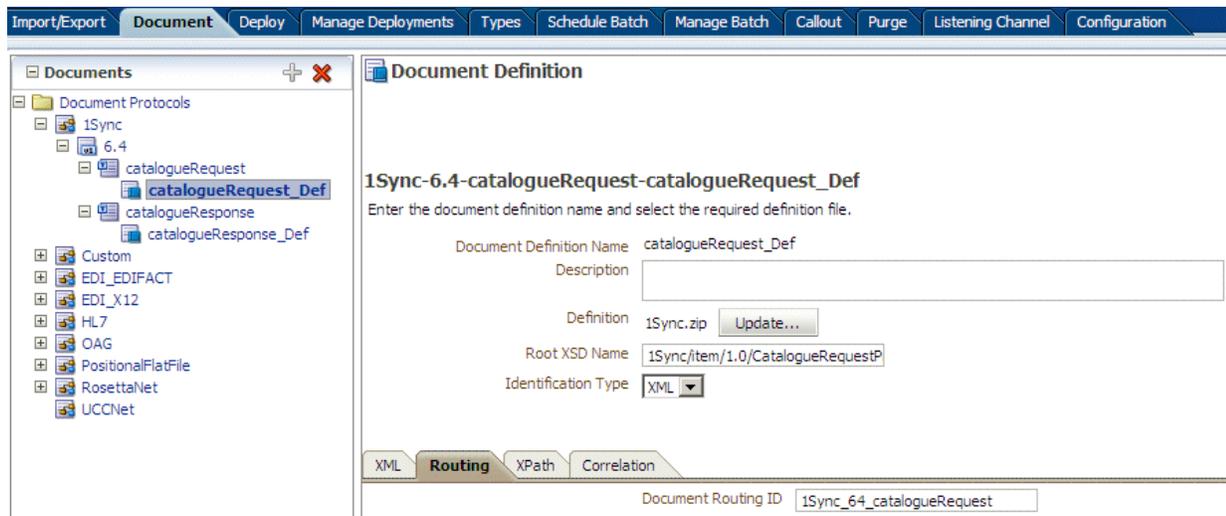
The image shows the catalogueRequest_Def document definition. 1Sync.zip is the definition file. The Root XSD Name fields displays 1Sync/item/1.0/CatalogueRequest. The Identification Type is XML. The Document Routing ID field displays 1Synch_64_catalogueRequest.

****

- Use correlation parameters on the **Correlation** tab. The following parameters and values are shown in Figure 8–23.

    – **Correlation From XPath Name**: From_catalogueRequest_messageId

    – **Correlation From XPath Expression:** /*[local-name()='envelope']/*[local-name()='header']/*[local-name()='messageId']

    This value matches the correlationFrom value in the payload.

Figure 8–23   The Correlation Tab for a 1Sync Document Definition
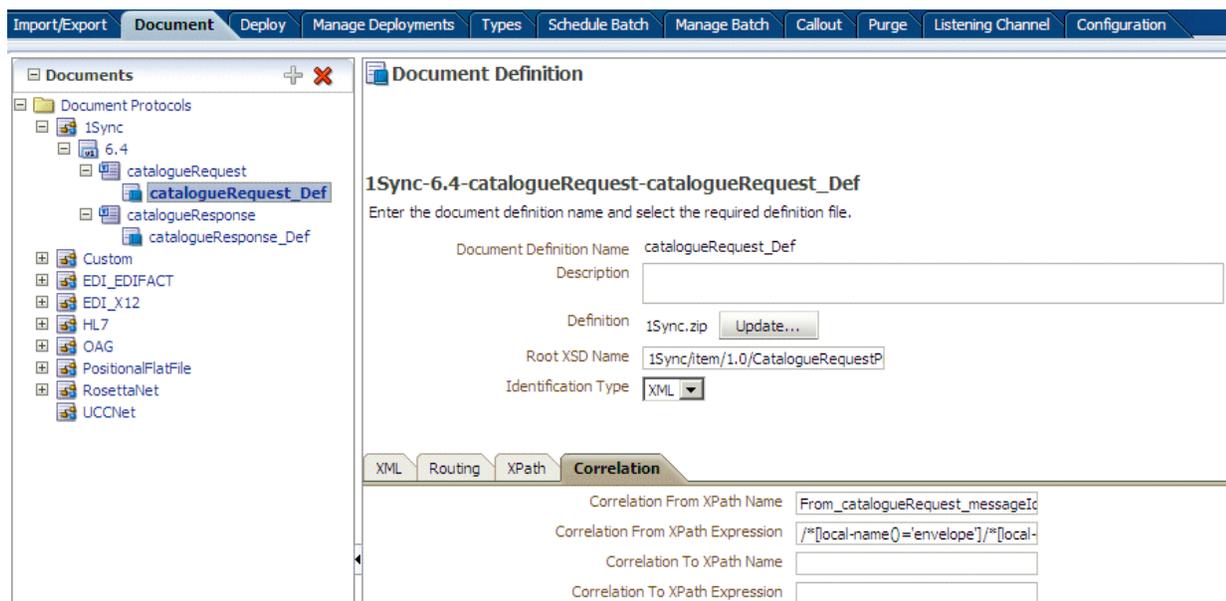
The image shows the Correlation tab for a 1Sync document. Parameters include Correlation From XPath Name, Correlation From XPath Expression, Correlation To XPath Name, and Correlation To XPath Expression. The parameters are explained in the surrounding text.

****

See the following for more information:

■ Section 8.1, "Using the Custom Document Protocol"

■ Section 1.7, "Oracle B2B Samples and Cookbooks." A 1Sync example is provided with the B2B samples.

8.9 Changing Document Details

Document details—document protocol versions and document type parameters—can be changed for a remote trading partner from the **Partners** > **Documents** tab. Host administrators can change any remote trading partner's document details here (host administrators must change their own data on the **Administration** > **Document** tab), and remote administrators can change document details for their own data, if the remote administrator has been granted access to those document types. See Section 1.4.2, "Restricting Access to Document Types," for more information.

Figure 8–24 shows the **Version** tab in the **Document Details** section, where parameters for the document protocol version can be changed.

Figure 8–24   Changing Document Details
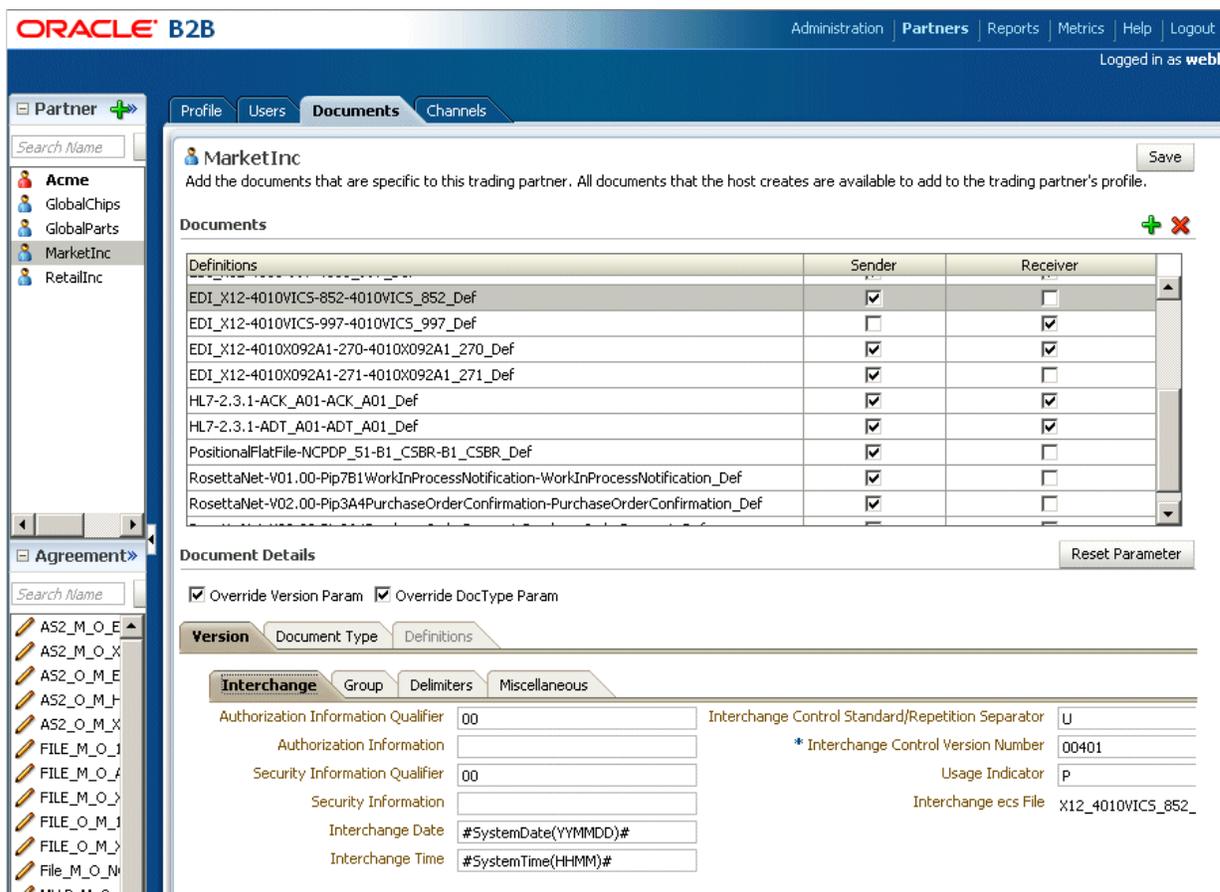
The image shows the Documents tab from the Partners link. Two documents are displayed: 850def and 997def, both EDI X2 documents. GChips is the sender and receiver of the 850def and the receiver of the 997def. The Interchange tab of the Document Protocol Version tab is displayed to show where you can change document details.

****

Delimiters, and parameters such as **Interchange Control Version Number**, **Interchange Date**, and **Interchange Time** are typically changed for different remote trading partners.

Figure 8–25 shows the **Document Type** tab, where parameters for the document type can be changed.

Figure 8–25   Changing Document Details
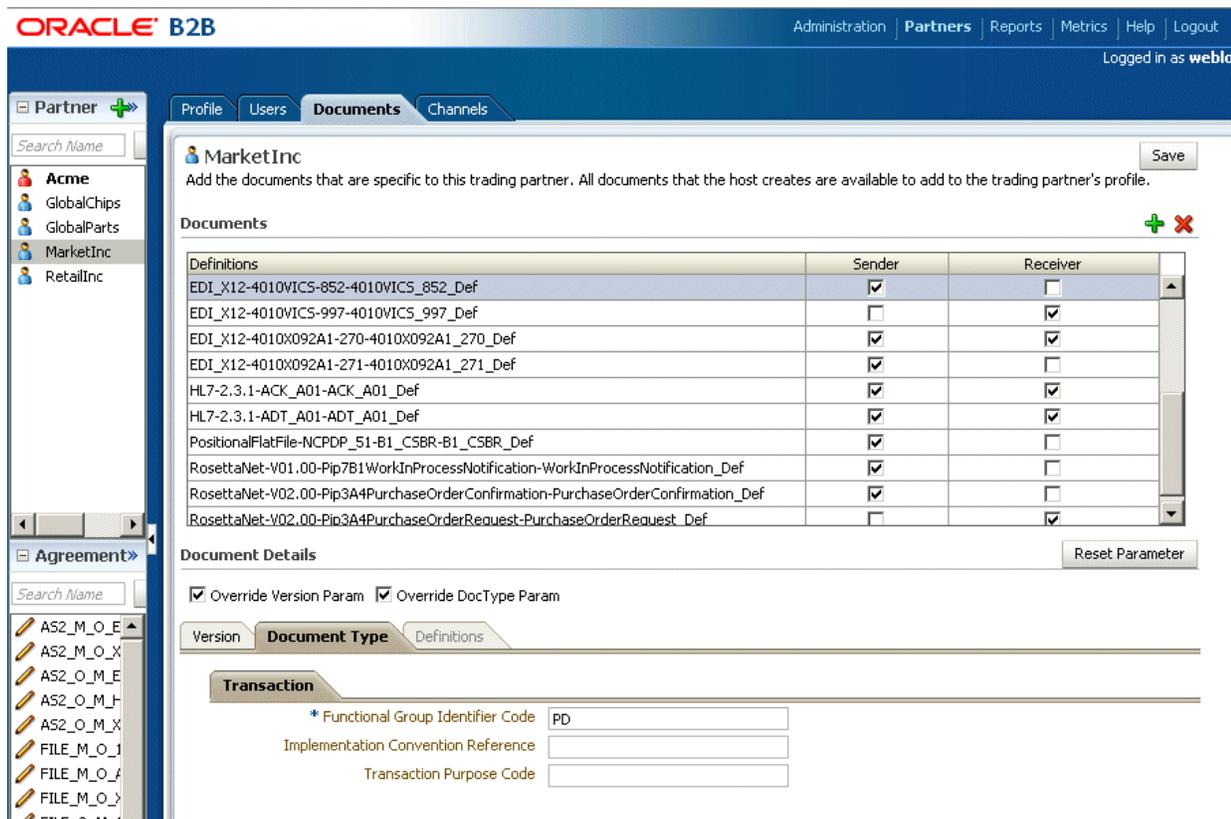
The image shows the Documents tab from the Partners link. Two documents are displayed: 850def and 997def, both EDI X2 documents. GChips is the sender and receiver of the 850def and the receiver of the 997def. The Interchange tab of the Document Protocol Version tab is displayed to show where you can change document details.

****

Use the **Override Version Param** and **Override DocType Param** parameters to indicate that override values are provided. Document type parameter values set for a remote trading partner take precedence over the default document type parameter values set for the document definition when the document was created on the **Administration** > **Document** tab.

To override document details:

1. Click the **Partners** tab.

2. Click the **Documents** tab.

3. Select a remote trading partner.

4. Select a document definition.

5. Select the override types that apply:

   - **Override Version Param**

   - **Override DocType Param**

6. Provide values to override values on the **Version** tabs or the **Document Type** tabs, or both.

7. Click **Save**.

8.9.1 Changing Document Definitions After Deploying an Agreement

Changes to a document definition after an agreement is deployed are not reflected in the trading partner's profile. Use the **Document Details** area on the **Partner**s > **Documents** tab to change document protocol version and document type parameters. Then redeploy the agreement.

8.9.2 Changing Document Definitions After Importing Metadata

If you import B2B metadata and then change the document from the **Administration** > **Document** tab, then you must also make the same changes to the supported document definition for the host and remote trading partners from the **Partner**s > **Documents** tab. Use the **Version**, **Document Type**, and **Definitions** tabs under **Document Details** to make the changes.

8.10 Using Document Routing IDs

A document routing ID is useful in two circumstances: when enqueuing to an AQ queue and when using B2B documents in a SOA composite application. If you set a document routing ID for messages enqueued to an AQ queue (inbound only), then the AQ consumer name is set to the document routing ID. Within a SOA composite application, if you use a document routing ID in your B2B binding component instead of the document definition, then all messages with the same document routing ID are routed to the same SOA composite.

This is useful if you have many different document definitions, but you want them to be handled the same way. The WSDL uses the document routing ID instead of the document definitions. In a SOA composite application, the B2B Configuration Wizard provides an option to use the document routing ID instead of selecting a document definition, as shown in Figure 8–26.

Figure 8–26   Document Routing ID Option in Oracle JDeveloper
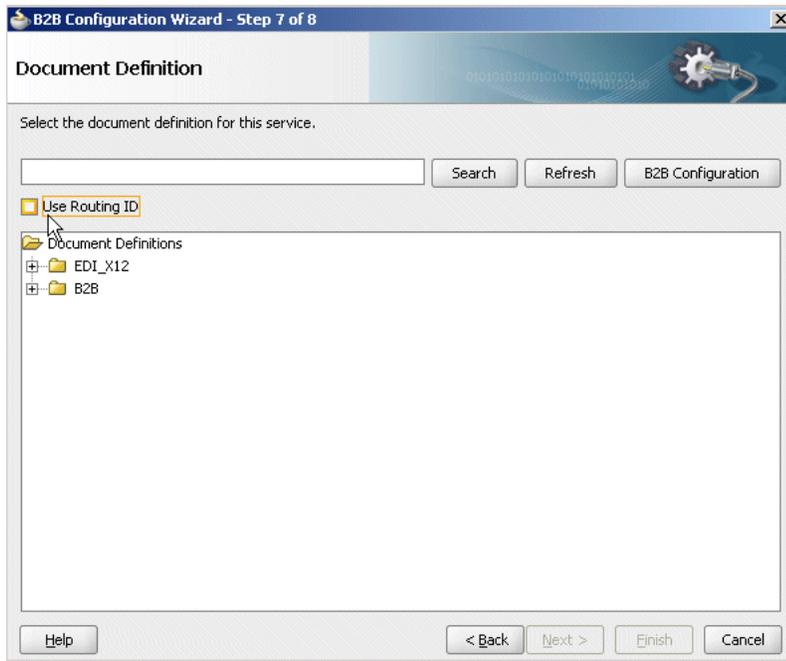
The image shows the Document Definition dialog box of the B2B Configuration Wizard. The first area of the dialog box shows a field to select the document definition of the service. To the right of the field are the Search, Refresh, and B2B Configuration buttons. The second area of the dialog box shows the Use Routing ID check box and an expanded hierarchy of the Document Definitions. The final area of the dialog box shows the Help, Back, Next, Finish, and Cancel buttons.

****

When using AQ, if you set the routing ID value instead of using the default b2buser, then do not set it to a numeric value. Use a combination of alphabetic and numeric values.

9

Managing Deployments

This chapter describes how to deploy an agreement to validate and activate a set of run-time data that is used for run-time transactions.

The chapter contains the following topics:

- Section 9.1, "Introduction to Agreement Deployment States"
- Section 9.2, "Managing Deployed Agreements"

See Chapter 6, "Creating and Deploying Trading Partner Agreements," for more information about how to deploy an agreement.

9.1 Introduction to Agreement Deployment States

You can manage the state of a deployment—Active, Inactive, Retired, or Purged—as shown in Figure 9–1. You can also search on the deployed agreements in the run-time repository, as well as export an agreement.

Figure 9–1 Managing a Deployed Agreement

The image shows the following tabs on the Oracle B2B screen: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Manage Deployments tab is selected. The first area of the tab shows an iconic representation of a step diagram with the following blocks from left to right: Active, Inactive (selected), Retire, and Purge. The second area of the tab shows the following Search fields: Name, Responding Partner, Initiating Partner, State, and Document Definition. At the top of the area is the Match radio button with the following options: All or Any. The third area of the tab shows the search results in a table with the following columns: Agreement, User, State, First Deployed Date, and Last Deployed Date. Above the table and to the right are the Search and Reset buttons.

****

9.2 Managing Deployed Agreements

A deployed agreement is initially in the Active state. Table 9–1 describes the deployment states.

Table 9–1   Deployed Agreement States

State	Description	When to Use
Active	The agreement has been successfully deployed and is ready to process messages. From an Active state, a deployed agreement can move to an Inactive state only.	When you are ready to receive or send messages using the agreement.
Inactive	The agreement can be changed to Active or Retired states. The agreement will not accept any new messages. However, all in-flight messages will be processed successfully. From an Inactive state, a deployed agreement can be moved to a Retired state or can be moved back to an Active state.	When a newer version of the same agreement is made Active, the previous version is changed to the Inactive state automatically. Also, when you do not want to receive new messages, but want to continue the in-flight messages, you can change the agreement to Inactive.
Retired	The agreement cannot be redeployed. No messages will be processed. From a Retired state, a deployed agreement can be purged only.	When you no longer want to receive or send messages using this agreement
Purged	The agreement is deleted from the system.	When you want to clean up unused agreements. Differs from Retired agreements, where you can still see the agreement in the system for information purposes.

Table 9–2 describes the information displayed for a deployed agreement.

Table 9–2   Deployed Agreements

Field	Description
Agreement	The name of the agreement, as created on the **Partners** > **Agreement** page.

Table 9–2   (Cont.) Deployed Agreements

Field	Description
User	The logged-in user name.
State	The state can be active, inactive, retired, or purged.
First Deployed Date	The date that the agreement was first deployed.
Last Deployed Date	The date that the agreement was last deployed.

9.2.1 Searching for Deployed Agreements

Use the search parameters described in Table 9–3 to search for deployed agreements.

Table 9–3   Search Parameters for Searching on Deployed Agreements

Parameter	Description
Name	Enter a string that is contained in the agreement name, equals the name, or is at the end of the name.
Responding Partner	Enter a string that is contained in the responding partner name, equals the name, or is at the end of the name.
Initiating Partner	Enter a string that is contained in the initiating partner name, equals the name, or is at the end of the name.
*State	Select from **All**, **Active**, **Inactive**, or **Retire**.
Document Definition	Select from one of the document definitions you previously created. See Chapter 4, "Creating Document Definitions," for more information.

- Click **Reset** to return the search parameters shown in Table 9–3 to their previous settings.
- Click **Advanced** to select additional search parameters, as shown in Figure 9–2.

Figure 9–2   Advanced Search Parameters

The image shows the following tabs on the Oracle B2B screen: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Manage Deployments tab is selected. The first area of the tab shows an iconic representation of a step diagram with the following blocks from left to right: Active, Inactive (selected), Retire, and Purge. The second area of the tab shows the following Search fields: Name, Responding Partner, Initiating Partner, State, and Document Definition. At the top of the area is the Match radio button with the following options: All or Any. The third area of the tab shows the search results in a table with the following columns: Agreement and User. Above the table and to the right are the Search, Reset, Save, and Add Fields buttons. The Add Fields button is expanded to show the following menu items: User, Document Type, Document Protocol Version, Name, Label, Initiating Partner, State, Document Definition, Document Protocol Name, and Responding Partner.

****

If you select the document search parameters from the **Add Fields** list, use them as follows: Select a document protocol name first to populate the list of document protocol versions; next select a document protocol version to populate the list of document types; and then select a document type to populate the list of document definitions.

The Saved Search feature is not available.

9.2.2 Changing the Deployment State

To change the deployment state:

1. Click the **Administration** link.

2. Click the **Manage Deployments** tab.

3. Select an agreement.

4. Click one of the available actions:

   - If the state is Active, then **Inactive** is available.

   - If the state is Inactive, then **Active** or **Retire** is available.

   - If the state is Retired, then **Purge** is available.

9.2.3 Exporting an Active Agreement

You can export active agreements. For agreements that use HTTPS or digital signature and encryption, the key store password of the host trading partner is not included as part of the export file. This is because a key store is specific to each computer. Therefore, when the export file is imported on a different computer, you must re-create the key store password and update the key store location (if needed) for the host trading partner in the B2B interface. If the export file is imported back or the key store and its location have not changed on the target computer, then the key store password and location may be identical to the first key store and key store password you used. This applies only to the host trading partner.

> **Caution:** Do *not* manually edit exported files. If you do so, Oracle B2B cannot guarantee their integrity.

To export an active agreement:

1. Click the **Administration** link.

2. Click the **Manage Deployments** tab.

3. Select an agreement (or multiple agreements).

4. Click **Export**.

    The system-provided file name is `MDS_EXPORT_DD_MM_YYYY.zip`. You can choose whether you want to open the file or save it, in which case you can specify a file name and download location. Each agreement is a separate ZIP file within `MDS_EXPORT_DD_MM_YYYY.zip`.

    Exporting can take some time based on the agreement metadata.

10

Creating Types

This chapter describes how to create identifier types, contact information types, and trading partner parameter types. It also discusses how Oracle B2B can meet individual specifications for document exchange, contact information, and trading partner parameters by using custom types.

This chapter contains the following topics:

- Section 10.1, "Creating Custom Identifier Types"
- Section 10.2, "Creating Custom Contact Information Types"
- Section 10.3, "Creating Custom Trading Partner Parameter Types"

See Section 5.2, "Creating Trading Partner Profiles," for information on *adding* custom types and values to a trading partner profile.

10.1 Creating Custom Identifier Types

Identifier types, or identifiers, help in identifying a trading partner (as exchange identifiers) or can be used to define additional inputs for various document protocols.

Oracle B2B has preseeded many of the commonly required identifiers. A new custom identifier can be created as required.

To create an identifier type:

1. Click the **Administration** link.

2. Click the **Types** tab.

3. In the **Identifiers** area, click **Add**.

4. Provide a name and optional description, as shown in Figure 10–1.

Figure 10–1   Creating an Identifier Type
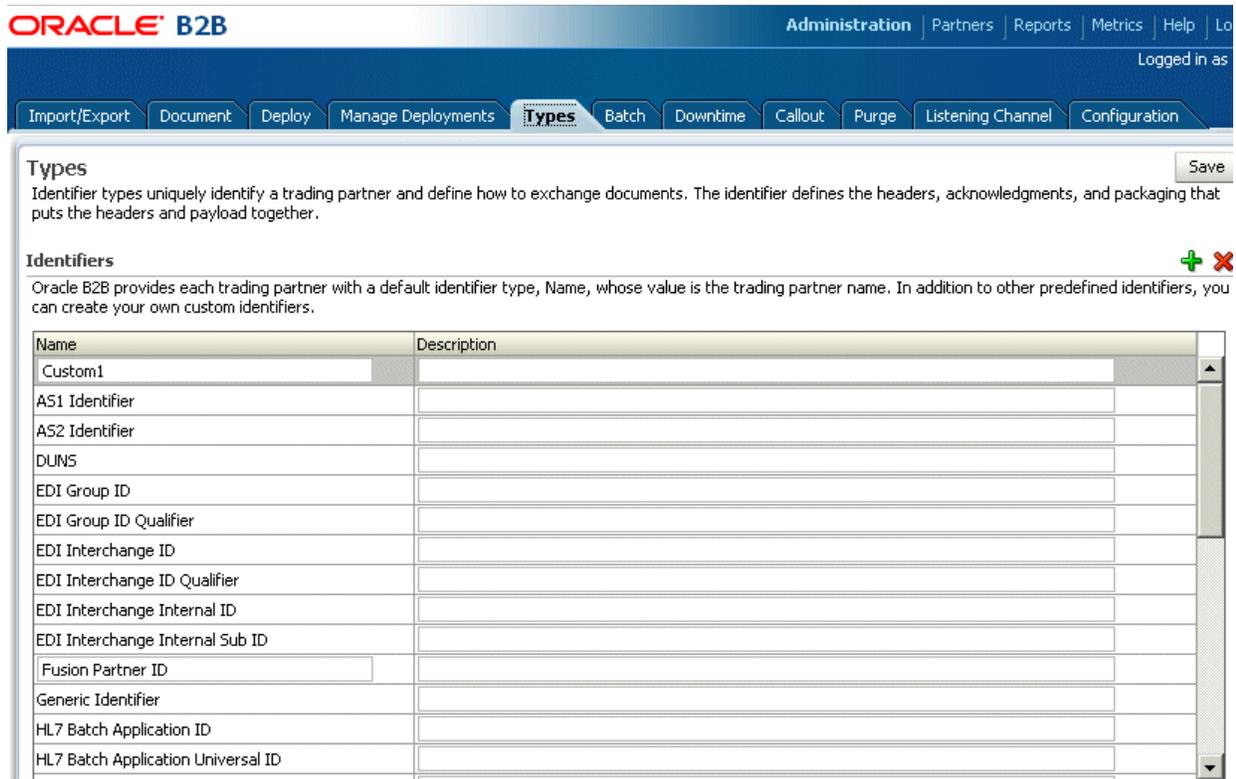
The image shows the following tabs on the Oracle B2B screen: Import/Export, Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Types tab is selected. The tab shows a table labeled Identifiers with the following columns: Name and Description. Above the table and to the right are plus and minus buttons. At the top right of the tab is the Save button.

****

5.   Click **Save**.

See Task 3, "Add Identifier Types and Values" on page 5-5 for how to add the new type and a value to a trading partner's profile.

Oracle B2B provides predefined identifiers for the supported document protocols, as listed in Table 10–1. You can deleted unused types to further customize your B2B environment. A type that is used by a trading partner cannot be deleted.

Table 10–1 (Cont.) Identifier Types Defined in Oracle B2B

Name	Description
AS1 Identifier	The specification for using EDI over SMTP to transmit data using e-mail. AS1 also works with non-EDI document types such as XML and TXT files. The AS1 Identifier and the Name identifier are required for AS1 exchanges.
AS2 Identifier	An alias for the service address (specified by the AS2-From/AS2-To fields) inside an AS2 transaction. The value can be any unique name that a trading partner recognizes. The AS2 Identifier and the Name identifier are required for AS2 exchanges.
DUNS	A unique, sequentially-generated, nine-digit number that is obtained from Dun and Bradstreet, formally as a D-U-N-S number. The DUNS Identifier and the Name identifier are required for RNIF exchanges.
EDI Group ID	Used to identify multiple branches within a trading partner's company. The group ID can be the same as the interchange ID.
EDI Group ID Qualifier	Similar to EDI Interchange ID Qualifier, this qualifier applies to EDIFACT only (not X12).
EDI Interchange ID	A unique identifier for a trading partner that can come from different sources. For example, if the trading partner has a Dun & Bradstreet number, that number can be used for the interchange ID. In most cases, the selected VAN assigns the interchange ID.
EDI Interchange ID Qualifier	Informs the network of the type of interchange ID that follows. Typical qualifiers include ZZ, indicating that the interchange ID that follows is mutually defined; 01, indicating that the interchange ID is the trading partner's Dun and Bradstreet number; 12, indicating that the interchange ID is a telephone number.
EDI Interchange Internal ID	Identifies the trading partner based on the EDI interchange internal ID.
EDI Interchange Internal Sub ID	Identifies the trading partner based on the EDI interchange internal sub-ID.
Generic Identifier	The IP address to use for identifying trading partners if you are using the generic exchange protocol (EDI X12 over Generic Exchange, EDI EDIFACT over Generic Exchange, or Custom Document over Generic Exchange) with the HTTP or HTTPS transport protocol. Do *not* enter the host name. The Generic Identifier and the Name identifier are required for Generic HTTP and Generic Email exchanges. Wildcard characters are not permitted in the IP address.
HL7 Batch Application ID	Identifies the trading partner based on the HL7 batch application ID. BHS.3 and BHS.5 have the same definition as the corresponding field in the MSH segment.
HL7 Batch Application Universal ID	Identifies the trading partner based on the HL7 batch application universal ID.
HL7 Batch Application Universal ID Type	Identifies the trading partner based on the HL7 batch application universal ID type.
HL7 Batch Facility ID	Identifies the trading partner based on the HL7 batch facility ID.
HL7 Batch Facility Universal ID	Identifies the trading partner based on the HL7 batch facility universal ID.
HL7 Batch Facility Universal ID Type	Identifies the trading partner based on the HL7 batch facility universal ID type.
HL7 File Application ID	Identifies the trading partner based on the HL7 file application ID. FSH.3 and FSH.5 have the same definition as the corresponding field in the MSH segment.

Table 10–1   (Cont.) Identifier Types Defined in Oracle B2B

Name	Description
HL7 File Application Universal ID	Identifies the trading partner based on the HL7 file application universal ID.
HL7 File Application Universal ID Type	Identifies the trading partner based on the HL7 file application universal ID type.
HL7 File Facility ID	Identifies the trading partner based on the HL7 file facility ID. This field further describes the sending/receiving application. The facility ID can have an organizational entity, unit, product or vendor's identifier.
HL7 File Facility Universal ID	Identifies the trading partner based on the HL7 file facility universal ID.
HL7 File Facility Universal ID Type	Identifies the trading partner based on the HL7 file facility universal ID type.
HL7 Message Application ID	Identifies the sending/receiving application.
HL7 Message Application Universal ID	For outbound messages, this field is used to override the Message Application Universal ID, which is MSH.3 for the sender and MSH.5 for the receiver. For inbound messages, this field is used for lookup.
HL7 Message Application Universal ID Type	For outbound messages, this field is used to override the Message Application Universal ID Type, which is MSH.3 for the sender and MSH.5 for the receiver. For inbound messages, this field is used for lookup.
HL7 Message Facility ID	Identifies the trading partner based on the HL7 message facility ID.
HL7 Message Facility Universal ID	For outbound messages, this field is used to override the Message Facility Universal ID, which is MSH.4 for the sender and MSH.6 for the receiver. For inbound messages, this field is used for lookup.
HL7 Message Facility Universal ID Type	For outbound messages, this field is used to override Message Facility Universal ID, which is MSH.4 for the sender and MSH.6 for the receiver. For inbound messages, this field is used for lookup.
MLLP ID	The TCP/IP Minimum Lower Layer Protocol (MLLP) is the standard for HL7. The MLLP ID and the Name identifier are required for MLLP exchanges.
Name	Identifies the trading partner by its name. The value for this type is automatically supplied when you create or edit the trading partner name, for example, Acme or GlobalChips. The Name identifier is required for Generic File, Generic FTP, Generic SFTP, Generic AQ, and Generic JMS exchanges.
ebMS Identifier	This type, OASIS ebXML Messaging Services (ebXML), specifies a secure and reliable way to exchange messages using HTTP, HTTPS, SOAP, XMLDsig, and XMLEncrypt. The ebMS Identifier and the Name identifier are required for ebMS exchanges.

10.2 Creating Custom Contact Information Types

Oracle B2B provides a centralized location for trading partner contact information. After you create a type, you can add it to a trading partner's profile and change its value.

You can create any type of contact information. You may want to create types for contact names, e-mail addresses, telephone and fax numbers, and so on. You can deleted unused types to further customize your B2B environment. A type that is used by a trading partner cannot be deleted.

To create a contact information type:

1. Click the **Administration** link.

2. Click the **Types** tab.

3. In the **Contact Information** area, click **Add**.

4. Provide a name for the contact information type, an optional description, and click **Save**.

The string that you provide in the **Name** field is displayed in a list under the **Type** field on the **Partners** > **Profile** page.

See Task 4, "Add Contact Information" on page 5-6 for how to add the new type and a value to a trading partner's profile.

10.3 Creating Custom Trading Partner Parameter Types

Trading partner parameter types are string types. After you create a type, you can add it to a trading partner's profile and change its value.

To create a trading partner parameter type and default value:

1. Click the **Administration** link.

2. Click the **Types** tab.

3. In the **Trading Partner Parameters** area, click **Add**.

4. Provide the following information and click **Save**.

- Name (required)

- Default Value (optional)

- Group Name (optional)

- Display Name (optional; however, the value of Display Name, not Name, appears when you add this type to a trading partner profile)

- Description (optional)

See Task 5, "Add a Trading Partner Parameter and Value" on page 5-6 for how to add the new type and a value to a trading partner's profile.

There are no predefined trading partner parameter types. You may want to create a type named `Country`, for example. Then the value—a specific country code—can be configured for each trading partner. You can deleted unused types to further customize your B2B environment. A type that is used by a trading partner cannot be deleted.

11

Batching EDI Messages

This chapter describes how to use the Oracle B2B interface to batch, schedule, and send outbound EDI X12 and EDI EDIFACT messages for outbound messages (Inbound messages to Oracle B2B are automatically debatched.)

The chapter contains the following topics:

- Section 11.1, "Setting Up a Batch"
- Section 11.2, "Managing Batched Messages"

See the following for more information about EDI:

- Section 8.2, "Using the EDI EDIFACT Document Protocol"
- Section 8.3, "Using the EDI X12 Document Protocol"

11.1 Setting Up a Batch

Batching is often used to group messages by document type; for example, you may want to send out a batch of purchase orders or a batch of invoices, to one or more trading partners. You can also batch multiple document types, sent to one or more trading partners.

When batching more than 1000 documents, where the payload size is greater than 30 KB, specific settings are required. See "Settings for Batching a Large Number of Documents" in *Oracle Fusion Middleware Performance Guide*.

Figure 11–1 shows where you set up a batch transmission of EDI messages.

Figure 11–1 Scheduling a Batch
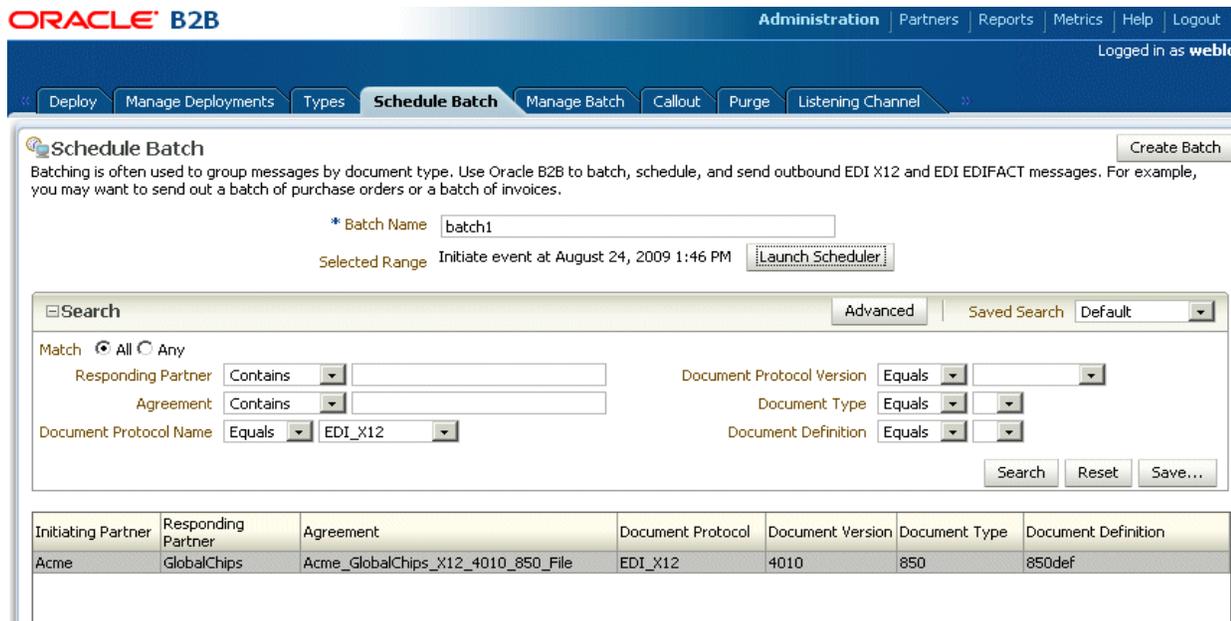
The image shows the following tabs on the Oracle B2B screen: Manage Deployments, Types, Import/Export, Schedule Batch, Manage Batch, Callout, and Purge. The Schedule Batch tab is selected. The first area of the tab shows the Batch Name field. Above the field is the Create Batch button and below the field is the Launch Scheduler button. The second area of the tab shows the following Search fields: Responding Partner, Agreement, Document Protocol Name, Document Protocol Version, Document Type, and Document Definition. At the top of the area is the Match radio button with the following options: All or Any. The third area of the tab shows the search results in a table with the following columns: Initiating Partner, Responding Partner, Agreement, Document Protocol, Document Version, and Document Type.

****

To set up a batch, do the following:

- Task 1, "Search for Agreements to Batch"

- Task 2, "Create the Batch"

- Task 3, "Schedule the Batch"

Task 1 Search for Agreements to Batch

1. Click the **Administration** link.

2. Click the **Schedule Batch** tab.

3. Use the search parameters described in Table 11–1 to identify which agreements you want to batch.

   Use the document search parameters as follows: Select a document protocol name first to populate the list of document protocol versions; next select a document protocol version to populate the list of document types; and then select a document type to populate the list of document definitions.

Table 11–1   Search Parameters for Creating a Batch

Parameter	Description
Match All or Any	If you select **All**, then fields with values are matched using an *and* condition. If you select **Any**, then fields with values are matched using an *or* condition.
Responding Partner	Select **Starts With**, **Contains**, **Equals**, or **Ends With**, and type the appropriate portion of the name of the responding trading partner.
Agreement	Select **Starts With**, **Contains**, **Equals**, or **Ends With**, and type the appropriate portion of the name of the agreement.
Document Protocol Name	Select **EDI_EDIFACT** or **EDI_X12**.
Document Protocol Version	Select a document protocol version that you previously created.
Document Type	Select a document type that you previously created.
Document Definition	Select a document definition that you previously created.

4. Click **Search**.

Active, deployed agreements (outbound) that meet your search criteria are displayed.

5. Go to Task 2, "Create the Batch".

Task 2 Create the Batch

1. Enter a name for the batch.

2. Select the agreements you want to batch.

3. Click **Create Batch**.

4. Go to Task 3, "Schedule the Batch".

Task 3 Schedule the Batch

1. Click **Launch Scheduler.**

2. Select the **Non-Repeating Event** tab or the **Repeating Event** tab.

3. For a nonrepeating event, do one of the following:

- Enter the date in the format shown in the Scheduler dialog and click **OK**.

- Or, click the **Calendar** icon, specify a date and time, and click **OK**.

For a repeating event (see Figure 11–2), enter details on the interval to trigger the event by specifying the minutes, hour, month, year, and date details. Then click **OK**.

Figure 11–2   Scheduling the Batch
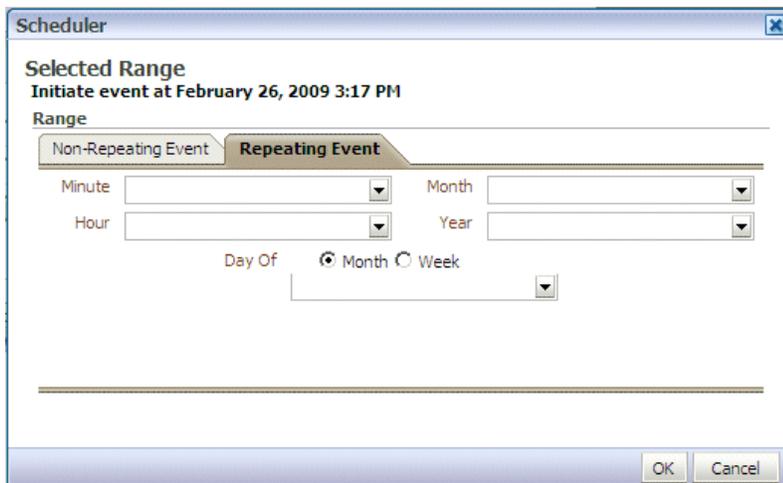
The image shows the Scheduler dialog box showing the following tabs: Non-Repeating Event and Repeating Event. The Repeating Event tab is selected. The tab shows the following drop-down lists: Minute, Hour, Month, Year, and Day of Month or Day of Week.

****

You can see the batches you create on the **Manage Batch** tab.

11.2 Managing Batched Messages

Figure 11–3 shows where you can search for batches that you previously created, view details of a batch, and disable, update, or delete a batch.

Figure 11–3   Managing Batched EDI Messages
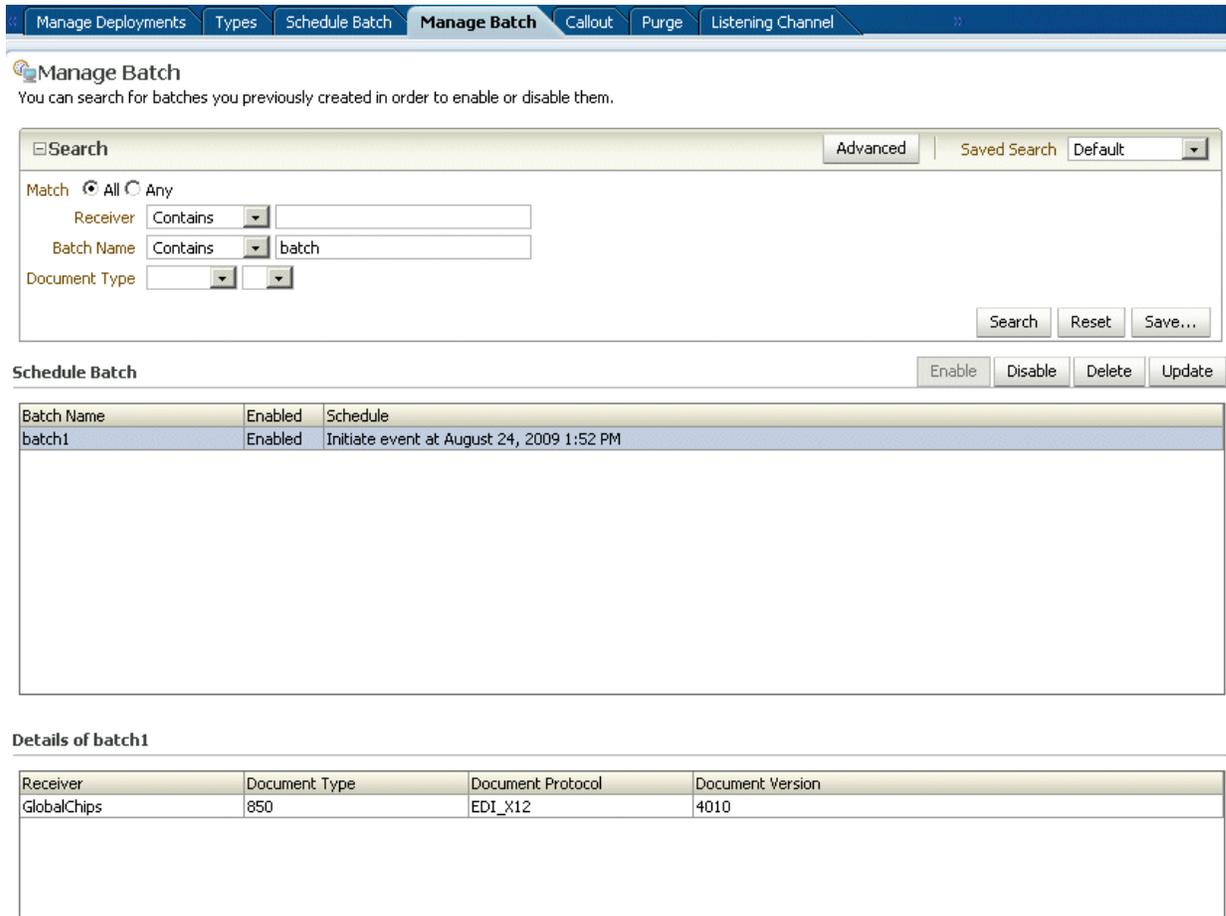
The image shows the following tabs on the Oracle B2B screen: Deploy, Manage Deployments, Types, Import/Export, Schedule Batch, Manage Batch, Callout, Purge, Listening Channel, and Configuration. The Manage Batch tab is selected. The first area of the tab shows the following Search fields: Receiver, Batch Name, and Document Type. At the top of the area is the Match radio button with the following options: All or Any. Above the fields are the Advanced button and the Saved Search drop-down list. Below the fields are the Search, Reset, and Save buttons. The second area of the tab shows the search results in a table with the following columns: Batch Name, Enabled, and Schedule. Above and to the right of the table are the Enable, Disable, Delete, and Update buttons. The third area of the tab shows details of the selected batch in a table with the following columns: Receiver, Document Type, Document Protocol, and Document Version.

****

For the **Update** batch action, only the documents definitions selected can be updated. Ensure that you reselect all the documents that are to be part of the batch and not just the new ones.

In some cases, B2B may not pick up the batched messages when you update the batching schedule. If you see that batched messages are not being picked up, delete the batch and create a new batch schedule with the same name as the previous batch. The same name must be used so that B2B picks up the previous messages in WAIT_BATCH status.

12

Scheduling Trading Partner Downtime

This chapter describes how to schedule Trading Partner downtime, which allows Trading Partners to notify each other about downtime and to delay message delivery during downtime.

In a competitive B2B world with high volume message flow, it is not possible to compromise on the processing capability of the B2B infrastructure, thereby creating the need of scheduling Trading Partner downtime. This results in processing the messages, even during trading partner's downtime, from the sender's perspective, and managing the maintenance or load of the recipient.

This chapter contains the following topics:

- Section 12.1, "Introduction to Scheduling Trading Partner Downtime"
- Section 12.2, "Scheduling Trading Partner Downtime"
- Section 12.3, "Deleting Scheduled Downtime"
- Section 12.4, "Extending Trading Partner Downtime"

12.1 Introduction to Scheduling Trading Partner Downtime

Various Trading Partners schedule their downtimes for different reasons and notify their partners about the downtime.

Scheduling the trading partner downtime in Oracle B2B ensures that messages are not delivered during the downtime period, yet are processed by Oracle B2B such that the messages are ready for delivery when the Trading Partner becomes available after downtime.

Trading Partner Downtime can be scheduled using a command line utility. See Section 20.16, "Scheduling Trading Partner Downtime" for more information.

It is not possible to reduce the length of scheduled downtime using either the console or the command line utility. To interrupt a scheduled downtime, or end an open-ended downtime, you can delete the schedule. See Section 12.3 for information.

At present the functionality to extend the time of a schedule using the console is not available. To extend the length of a scheduled downtime use the command line utility. See Section 12.4 for an explanation of the options.

12.2 Scheduling Trading Partner Downtime

Note: As a pre-requisite, it is required to enable the dispatcher in Oracle B2B. The number of dispatchers depends on the load and configuration of the system. The Outbound Dispatcher Count should be at least 1.

For more information, refer to Chapter 17, "Configuring B2B System Parameters."

To schedule downtime:

1. Click the **Administration** link, and select the **Downtime** tab.

The image shows the following tabs on the Oracle B2B screen: Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel. The Downtime tab is selected. The Downtime tab contains two subtabs: Schedule Downtime and Manage Downtime, and the Schedule Downtime tab is selected.

The first area of the Schedule Downtime tab shows the Downtime Name field, followed by the Schedule Downtime Start Time and End Time fields which are controlled with calendar selection buttons to select date and time values. Above these fields is the Schedule Downtime button.

The middle area of the tab shows the Responding Partner Search field. The bottom area of the tab displays the search results in a table with the following columns: Responding Partner and Channel.

****

2. With the **Schedule Downtime** subtab selected, enter a name in **Downtime Name** field.

3. Select the dates and times between which the downtime interval is desired in the **Start Time** and **End Time** fields.

   You can start the downtime immediately by selecting an end time only, configured in the **End Time** field. The downtime starts immediately upon creation until the end time.

   You can configure an open-ended downtime by selecting a start time only, configured in the **Start Time** field. The **End Time** field is left empty. The open-ended downtime is stopped by deleting the schedule.

4. Select an entry in the **Responding Partner-Channel** table and click **Schedule Downtime**.

12.3 Deleting Scheduled Downtime

To interrupt a scheduled downtime, or end an open-ended downtime, you can delete the schedule.

To delete a scheduled downtime:

1. Click the **Administration** link, and select the **Downtime** tab.

2. With the **Manage Downtime** subtab selected, search for the schedule using the **Downtime Name** and/or **Responding Partner**.

The image shows the following tabs on the Oracle B2B screen: Document, Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Downtime tab is selected. The Downtime tab contains two subtabs: Schedule Downtime and Manage Downtime, and the Manage Downtime tab is selected.

The top area of the Manage Downtime subtab shows the Search field, Responding Partner. At the top of the area is the Match radio button with the following options: All or Any.

The middle area of the tab displays the search results in a table with the following columns: Downtime Name, Start Date and End Date. The bottom area of the tab displays the Responding Partner and Channel of the Downtime schedule selected in the Scheduled Downtime area.

****

3. Select the result in the **Scheduled Downtime(s)** list and click **Delete**.

12.4 Extending Trading Partner Downtime

The following options enable you to alter an existing schedule.

12.4.1 Overlapping Schedules

Overlap is allowed using different schedule names. Overlap is not allowed with the same schedule name.

Consider a schedule X between 4:00 AM to 6:00 AM. It is possible to create a new schedule Y with start time 5:00 AM to end time 7:00 AM. This makes the effective schedule from 4:00 AM to 7:00 AM.

12.4.2 Using the Extend Option

By default an error is thrown while extending an earlier created schedule. User can extend the schedule using the command line option `-Dextend`.

Consider a schedule X between 4:00 AM to 6:00 AM, it is possible to extend it by creating a new schedule X with option `-Dextend=true` between 6:00 AM to 8:00 AM. This makes the effective schedule from 4:00 AM to 8:00 AM.

See Section 20.16, "Scheduling Trading Partner Downtime" for information and examples about using the command line options.

13

Managing Callouts

This chapter describes how to create and use Java callouts, which transform the formats of messages exchanged between the host and remote trading partners. You can use callouts to invoke an XSLT style sheet, and any Java program in general.

This chapter contains the following topics:

- Section 13.1, "Introduction to Callouts"
- Section 13.2, "Creating a Callout"
- Section 13.3, "Including a Callout in an Agreement"
- Section 13.4, "Implementing a Callout"

13.1 Introduction to Callouts

Callouts are used in environments in which a host trading partner application does not use the same message format as the remote trading partner. For example, a remote trading partner sends a RosettaNet XML-formatted purchase order request to a host trading partner, as shown in Figure 13–1.

Figure 13–1   A Purchase Order Example: Using Callouts for Differently Formatted XML Messages

Purchase Order Request
(Oracle E-Business Suite XML format)

Callout Usage
(Transforms RosettaNet XML to Oracle E-Business Suite XML)

Host Application (Oracle E-Business Suite XML)	Internal Delivery Channel Responding Internal Delivery Channel	**A** **Host Trading Partner** (seller) **B**	Purchase Order Request (RosettaNet XML format) Purchase Order Acceptance (RosettaNet XML format)	**Remote Trading Partner** (buyer) (RosettaNet XML)

Responding Callout Usage
(Transforms Oracle E-Business Suite XML to RosettaNet XML)

Purchase Order Acceptance
(Oracle E-Business Suite XML format)

A Source Document Definition: PurchaseOrderRequest
Target Document Definition: PurchaseOrderAcceptance

B Source Document Definition: PurchaseOrderAcceptance
Target Document Definition: PurchaseOrderRequest

The image shows a box on the far right labeled Remote Trading Partner (buyer) (RosettaNet XML). There are two arrows on the left side of this box. The top arrow labeled Purchase Order Request (RosettaNet XML format) points out to the left to a box labeled Host Trading Partner (seller). The bottom arrow labeled Purchase Order Acceptance (RosettaNet XML format) points from the box labeled Host Trading Partner (seller) into the box labeled Remote Trading Partner (buyer) (RosettaNet XML).

The box labeled Host Trading Partner (seller) has a smaller box inside in the upper left that is labeled A and a smaller box inside in the lower left that is labeled B. An arrow labeled Callout Usage (Transforms RosettaNet XML to Oracle E-Business Suite XML) points down to the box labeled A. An arrow labeled Responding Callout Usage (Transforms Oracle E-Business Suite XML to RosettaNet XML) points up to the box labeled B.

There are two arrows on the left side of box labeled Host Trading Partner (seller). The top arrow labeled Purchase Order Request (Oracle E-Business Suite XML format) points out to the left to a box labeled Host Application (Oracle E-Business Suite XML). This top arrow is also labeled Internal Delivery Channel. The bottom arrow labeled Purchase Order Acceptance (Oracle E-Business Suite XML format) points from the box labeled Host Application (Oracle E-Business Suite XML) into the box labeled Host Trading Partner (seller). This bottom arrow is also labeled Responding Internal Delivery Channel.

The Host Application (Oracle E-Business Suite XML) box and the Host Trading Partner (seller) box are inside a larger dashed box.

Below the illustration is a key with two letters: A and B. The key letter A (which denotes the smaller upper left box inside the Host Trading Partner (seller) box) says Source Document Definition: PurchaseOrderRequest and Target Document Definition: PurchaseOrderAcceptance.

The key letter B (which denotes the smaller lower left box inside the Host Trading Partner (seller) box) says Source Document Definition: PurchaseOrderAcceptance and Target Document Definition: PurchaseOrderRequest.

****

In this example, the host application of the host trading partner is an Oracle E-Business Suite application that does not use RosettaNet XML-formatted messages. To enable communication between these two different formats, you create two callouts, as follows:

- One callout, `callout_inbound`, for example, transforms the RosettaNet XML-formatted purchase order request into an Oracle E-Business Suite XML format understood by the Oracle E-Business Suite application. The Oracle E-Business Suite application, in turn, responds to the request message with a purchase order acceptance message in Oracle E-Business Suite XML format.

- The other callout, `callout_outbound`, for example, transforms the Oracle E-Business Suite XML format back into a RosettaNet XML-formatted message for the remote trading partner.

These two callouts are then associated with the two agreements created for this exchange, as follows:

- Include `callout_outbound` in the agreement for the outbound message, that is, the agreement for the initiating purchase order request.

- Include `callout_inbound` in the agreement for the inbound message, that is, the agreement for the responding purchase order acceptance.

Because a document definition is a component of an agreement, a callout is associated with a specific document definition.

This purchase order example depicts a simple association of one callout to one agreement. In reality, however, the same callout can be included in many different agreements by changing the value of one or more callout parameters. See Figure 13–3 for where you add parameters and see Table 13–2 for a list of parameter attributes.

13.1.1 Transport Callouts

Another type of callout is the transport callout, which is associated with a channel. For the inbound message, B2B invokes the transport callout immediately after it receives a message from the transport. For the outbound message, B2B invokes the transport callout immediately before it sends a message to the transport. Transport callouts can be selected in the channel configuration, as shown in Figure 13–2, and can be used with any protocol.

Figure 13–2   Transport Callouts
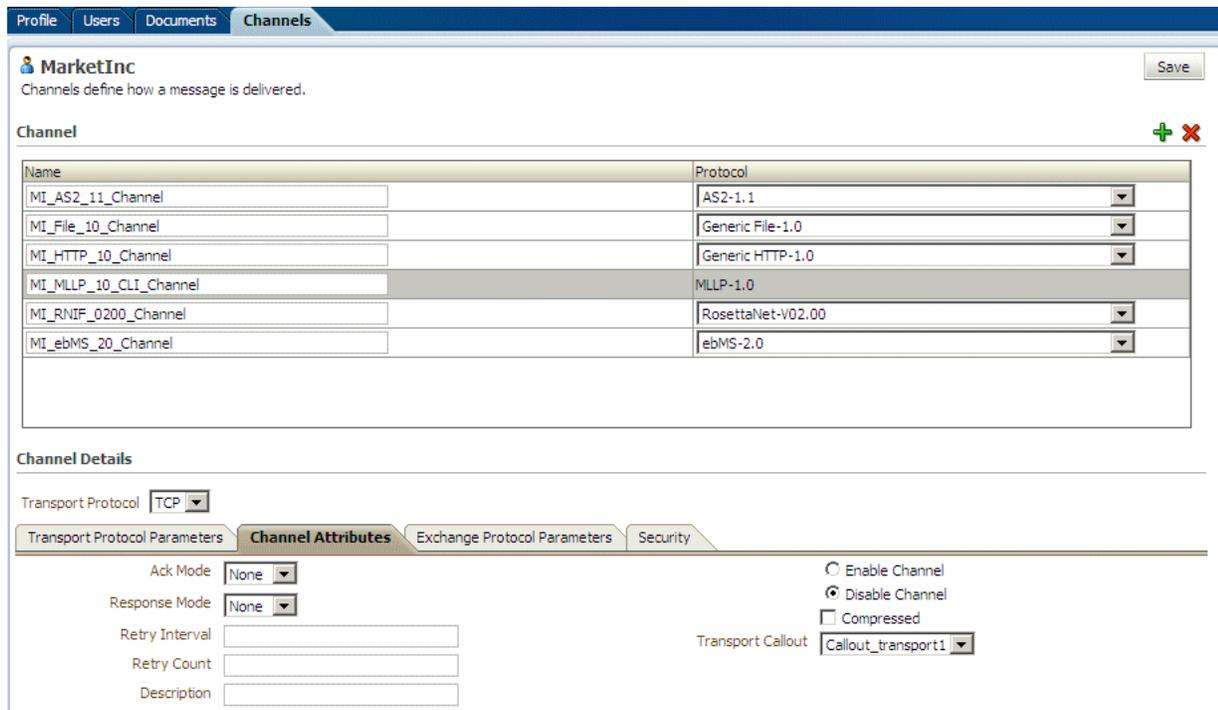
The image shows that Callout_transport1 is selected from the Transport Callout list. This field is on the Channel Attributes tab. Any protocol can be selected.

****

You can use transport callouts to extract custom headers for inbound and outbound messages using the MLLP protocol. Example 13–1 shows how to set and get the CUSTOM_HEADER property in the callout.

Example 13–1   Setting and Getting the CUSTOM_HEADER Property

```
import java.util.*;
import oracle.tip.b2b.callout.*;
import oracle.tip.b2b.callout.exception.*;

    public class SampleCallout implements Callout {
    public void execute(CalloutContext context,List input,List output)
                throws CalloutDomainException, CalloutSystemException {
      try {
       CalloutMessage cmIn = (CalloutMessage)input.get(0);
       String s =cmIn.getBodyAsString();

       //for getting the CUSTOM_HEADER
       Properties params =  (Properties)cmIn.getParameters();
       String customHeader = (String)params.get("CUSTOM_HEADER");

       //for setting the CUSTOM_HEADER
       CalloutMessage cmOut = new CalloutMessage(s);
       cmOut.setParameter("CUSTOM_HEADER", "your_value");
       output.add(cmOut);

       } catch (Exception e) {
       throw new CalloutDomainException(e);
       }
```

```
    }
}
```

See Section 5.5.1.4, "Using a Transport Callout to Extract Custom Headers," for more information.

Transport callouts are created like other callouts, from the **Callout** tab, as described in Section 13.2, "Creating a Callout." Although a transport callout is not added to an agreement, all transport callouts appear in the **Callouts** list on the **Agreement** tab; therefore, it is available for selection. To avoid confusion, when you create a transport callout, provide a name that indicates its type so that you do not select it from the **Callouts** list on the **Agreement** tab.

13.1.2 Creating a Callout Library JAR File

If the callout JAR file provided with Oracle B2B is not sufficient for your needs, you can create your own callout JAR file outside of Oracle B2B, following the standards described in the *Oracle Fusion Middleware B2B Callout Java API Reference*. Use the **Configuration** tab of the **Administration** link to specify the directory location of this external JAR file. It is recommended that you create an external JAR file for your callouts; do not bundle your callouts with b2b.jar.

> **Note:** MySampleCallout is a restricted keyword and should not be used. It is already packaged into b2b.jar.

13.2 Creating a Callout

To create a callout, provide callout details—the implementation class name and library name—and callout parameters, as shown in Figure 13–3.

Figure 13–3 Creating a Callout
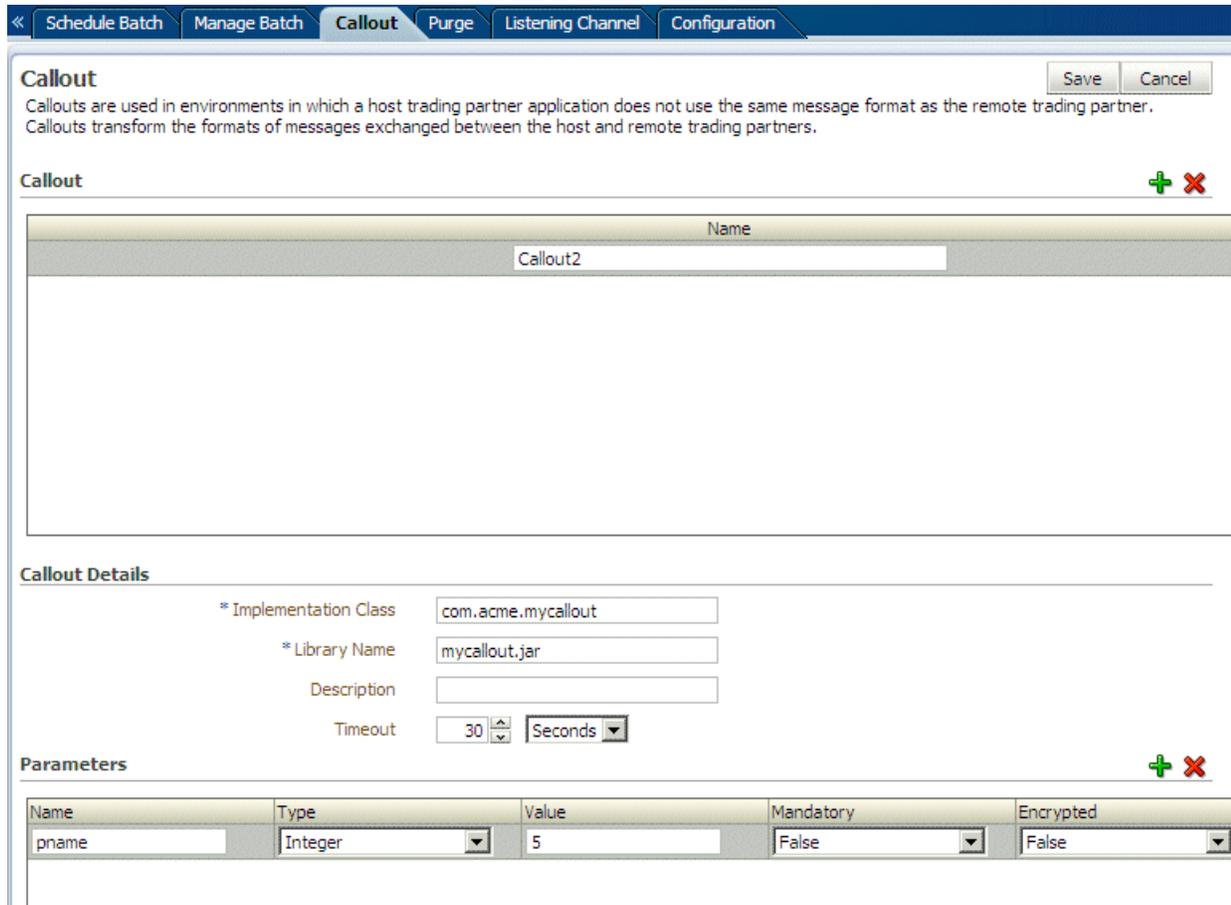
The image shows the following tabs on the Oracle B2B screen: Manage Batch, Callout, Purge/Archive, and Listening Channel. The Callout tab is selected. The first area of the tab shows the Callout list. There are plus and minus icons above the list and to the right. The second area of the tab shows the Callout Details with the following fields: Implementation Class, Library Name, Description, and Timeout. The third area of the tab shows the Parameters of the selected callout in a table with the following columns: Name, Type, Value, and Mandatory. There are Save and Cancel buttons at the top of the tab on the right side.

****

You can create multiple callouts with the same name if you assign them different implementation names. You cannot delete a callout that is included in an agreement.

Table 13–1 lists the callout details that you provide.

Table 13–1   Callout Details

Field	Description
*Implementation Class	Enter the class file name without `.class`. **Note:** Oracle B2B includes a predefined class file named XSLTCalloutImpl that you can use for XML-to-XML transformations.

Table 13–1   (Cont.) Callout Details

Field	Description
*Library Name	Enter the JAR file name that has the callout implementation classes.
	Note: If you specify one or more of your own callout JAR files, you must specify the directory location. Use the **Configuration** tab from the **Administration** link. The directory location for the default `b2b.jar` file included with Oracle B2B does not need to be specified.
	The callout library must be manually migrated from one environment to another. The B2B export/import feature does not migrate the callout library JAR.
	See Section 17.1, "Setting Configuration Parameters," for information on specifying the callout directory for your own callout JAR files.
	The image shows a section of the Oracle Enterprise Manager screen. The first area of the screen shows icons for Farm, SOA Infrastructure, and Topology. The left pane of the screen shows an expanded hierarchy with the soa-infra(oc4j_soa) node selected. The right pane shows the B2B Server Properties with the following fields: Callout Directory and Log Payload (a check box).
	*** *** ***
Description	Enter a description.
Timeout (seconds)	Enter the time limit in which to process the callout.
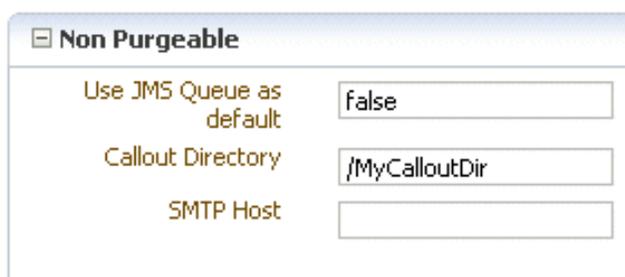
Callout parameters are similar in concept to global variables to which you can assign local values that are applicable only to a specific callout use. Or, you can create a callout parameter and assign it a default value that is applicable to all callout uses. Changes to callout parameters for an existing callout affect all agreements that use that callout.

Table 13–2 lists the optional callout parameter attributes.

Table 13–2   Callout Parameter Attributes

Field	Description
Name	Enter a parameter name.
Type	Select from **Integer**, **Float**, **String**, **Boolean**, or **Date** types. The format for the **Date** type is MM/DD/YYYY. **Note:** Changing a type can invalidate the parameter default value.
Value	Enter a value. If **Encrypted** is set to **True**, then this value is encrypted.
Mandatory	Select **True** or **False**.
Encrypted	Select **True** or **False**.
Description	Enter an optional description.

After you create a callout, it is available to include in an agreement. See Section 13.3, "Including a Callout in an Agreement," for more information. If you change a callout after it is deployed with an agreement, a server restart is required.

To create a callout:

1. Click **Administration**, and then **Callout**.

2. In the **Callout** section, click **Add**.

3. Enter a name for the callout.

   (You may want to indicate if you are creating a transport callout in the name.)

4. Enter callout details, as described in Table 13–1.

5. (Optional) Click **Add** in the **Parameters** section.

6. Enter a parameter name and attributes, as described in Table 13–2.

7. Click **Save**.

You can edit the details, parameters, or parameter values at any time, but not the callout name.

13.3 Including a Callout in an Agreement

After you create a callout, it is available to include in an agreement, as shown in Figure 13–4.

Figure 13–4   Specifying a Callout in an Agreement

The image shows the Agreement tab. The first area of the tab shows an iconic representation of a step diagram with the following blocks from left to right: Acme, ORDERS_def, and GlobalChips. The second area of the tab shows the agreement details with the following fields: Agreement Id, Name, Description, Start Date, End Date, and Callout.

****

To include a callout in an agreement:

1. Click **Partners**.

2. Click an agreement name.

3. Select a callout.

4. Click **Save**.

To update the value of a callout parameter for a specific agreement:

1. Click **Partners**.

2. Click an agreement name.

3. Select a callout.

4. Click **Callout Details**.

5. Enter a value for the parameter name, as shown in Figure 13–5.

Figure 13–5   Entering Callout Details
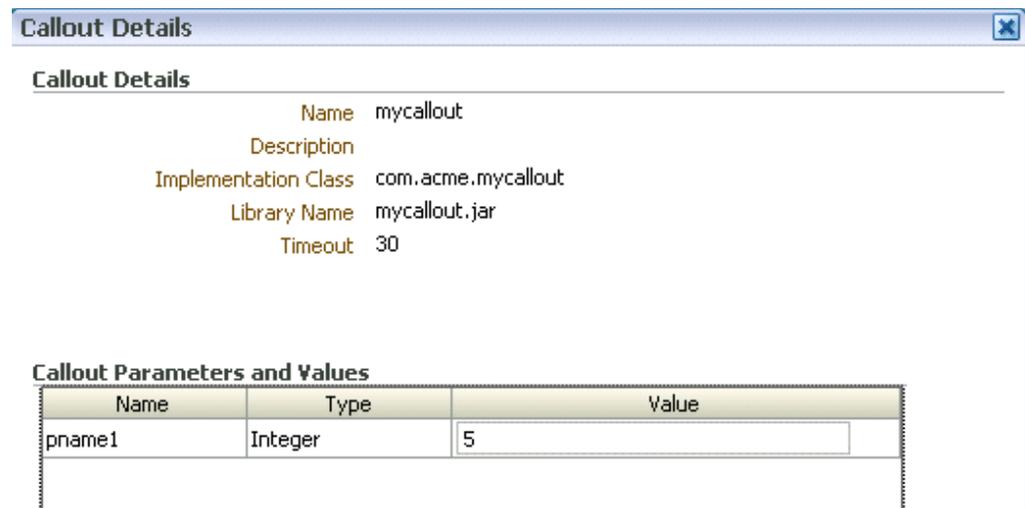
The image shows the Callout Details dialog box. The first area shows of the tab the Callout Details with the following fields: Name, Description, Implementation Class, Library Name, and Timeout. The second area of the tab shows the Callout Parameters and Values as a table with the following columns: Name, Type, and Value.

****

6. Click **OK**.

13.4 Implementing a Callout

Example 13–2 shows how an incoming XML document is transformed to another XML document. The directory structure is oracle.tip.callout. In this example, note that setting the output CalloutMessage in the output list is required (output.add(cmOut)).

Example 13–3 shows how to create a synchronous callback callout for use with Transport Synch Callback. See Section 5.5.3, "Using Transport Sync Callback" for more information.

Example 13–2   Code Example of an XML-to-XML Transformation

```
import java.io.*;
import java.net.*;
import java.util.*;
import oracle.xml.parser.v2.*;
import oracle.tip.b2b.callout.Callout;
import oracle.tip.b2b.callout.CalloutMessage;
import oracle.tip.b2b.callout.CalloutContext;
import oracle.tip.b2b.callout.exception.*;

/**
 * This sample callout transforms the incoming XML document
 * to another XML document. It also shows how to generate
 * Functional Ack and Error message.
 */
public class XSLTCalloutImpl implements Callout {
   public void execute(CalloutContext context,
                       List input,
                       List output)
              throws CalloutDomainException, CalloutSystemException {
     try {

      // (1) Retrieve the callout properties from CalloutContext
      String xsltFile     = context.getStringProperty("xsltFile");

      // (2) Get the input callout message
      CalloutMessage cmIn = (CalloutMessage)input.get(0);

      // (3) Process the message
      // instantiate a stylesheet
      URL xslURL = new URL("file://" + xsltFile);
      XSLProcessor processor = new XSLProcessor();
      XSLStylesheet xsl = processor.newXSLStylesheet(xslURL);

      // parser input XML content
      DOMParser parser = new DOMParser();
      parser.setPreserveWhitespace(true);
      parser.parse(new StringReader(cmIn.getBodyAsString()));
      XMLDocument xml = parser.getDocument();
      processor.showWarnings(true);
      processor.setErrorStream(System.err);

      // Transform the document
      StringWriter strWriter = new  StringWriter();
      processor.processXSL(xsl, xml, new PrintWriter(strWriter));

      // (4) Create a output callout message
      // create a callout output message
```

```
        CalloutMessage cmOut =
            new CalloutMessage(strWriter.getBuffer().toString());
        strWriter.close();

// create Functional Ack callout message
// this is an optional step
CalloutMessage fa = new CalloutMessage(/*set FA payload here*/);
fa.setParameter("functional_ack", "true");
//setting your own doctype and revision
//set the doc type name and revision as defined in b2b ui
fa.setParameter("doctype_name", "fa");
fa.setParameter("doctype_revision", "1.0");

// create Error callout message
// this is an optional step
CalloutMessage err = new CalloutMessage(/* set the payload that causes this
error */);
err.setParameter("error_message", "true");
err.setParameter("error_desc", "set the error desc");

        output.add(cmOut);
        output.add(fa);
        output.add(err);

        //(5) Throw an exception, if any
    } catch (Exception e) {
        throw new CalloutDomainException(e);
    }
  }
}
```

Example 13–3   Code Example of a Sync Callback Callout

```
import java.io.FileInputStream;
import java.util.List;
import java.util.Properties;
import oracle.tip.b2b.callout.Callout;
import oracle.tip.b2b.callout.CalloutContext;
import oracle.tip.b2b.callout.CalloutMessage;
import oracle.tip.b2b.callout.exception.CalloutDomainException;
import oracle.tip.b2b.callout.exception.CalloutSystemException;
import oracle.tip.b2b.domain.B2BParameters;
import oracle.tip.b2b.system.B2BRuntimeException;
import oracle.tip.b2b.system.ErrorKeys;

public class SyncSampleCallout implements Callout {

        public void execute(CalloutContext calloutContext, List input, List output)
                    throws CalloutDomainException, CalloutSystemException {
                try
                {
                        CalloutMessage message = new CalloutMessage();
                        Properties properties = new Properties();

                        properties.put("FROM_PARTY", "MarketInc");
                        properties.put(B2BParameters.TO_PARTY, "OracleServices");
                        properties.put(B2BParameters.DOCTYPE_NAME, "271");
                        properties.put(B2BParameters.DOCTYPE_REVISION, "4010X092A1");

                        message.setParameters(properties);
```

```
                FileInputStream inStream = new FileInputStream("/tmp/271.dat");

                byte[] content = new byte[inStream.available()];

                inStream.read(content);

                inStream.close();

                message.setBody(content);

                output.add(message);
        }
        catch(Exception e) {
                new B2BRuntimeException(ErrorKeys.B2B_RUNTIME_ERROR, e);
        }
    }
}
```

14

Using Maps

This chapter describes how to use a mapset to define the mapping logic when the messages in your source and target systems are defined by different document definitions, and you need to map the data from one message to another.

This chapter contains the following sections:

- Section 14.1, "Introduction to Mapping"
- Section 14.2, "Using Mapsets in the Oracle B2B Console"
- Section 14.3, "Using Oracle MapBuilder to Create Prebuilt Maps"

14.1 Introduction to Mapping

Mapsets provide data transformation for cases where it is better to map a native data format to a native data format instead of the standard translation of mapping native formatting to XML, XML to XML, and then XML back to native formatting. You can use mapsets when you need to map data between messages that are defined by different document definitions. For example, you might have a system that sends messages in HL7 2.3.1 format, but a receiving system needs the data in HL7 2.5 format. Or you might need to transform HIPAA 4010 messages to HIPAA 5010 and back again.

A mapset includes a predefined or user-defined map file and two document definitions. The map file defines how data is converted from one format to the other. Currently, the Oracle SOA Suite healthcare integration user interface supports mapping different versions of HL7 messages to each other. The Oracle B2B Console supports mappings for different HIPAA X12 messages to each other, as well as different versions of HL7 to each other.

> **Note:** Currently, using mapsets to transform HL7 2.x messages to HL7 v3.0 is not supported, but this can be achieved using the XSLT mapper in Oracle JDeveloper.

Oracle B2B provides the functionality of directly transforming one native data format to another native format. For example, Oracle B2B provides prebuilt maps for Health Insurance Portability and Accountability Act (HIPAA) documents version 4010 to version 5010 and vice versa (for document type 837 Health Care Claim). This enables you to use a standard map file for the transformation. However, if you want to create customized maps, Oracle B2B provides a MapBuilder application (available only on the Windows platform) where the you can design and build your own maps.

14.1.1 About Mapsets

When you create a mapset, you associate a map file, a source document definition, and target document definition within one mapset. A mapset groups the definitions and mapping together into one unit that can be reused in multiple Oracle B2B or healthcare integration applications. Each mapset uses two different document definitions, and these definitions need to be created in the design-time repository in order to create the mapset. Mapsets give you the option of using the default document definitions provided for each protocol or customized definitions that you have created or modified.

When you create a mapset in Oracle SOA Suite for healthcare integration, you associate it with the appropriate endpoints to incorporate the mapping logic into healthcare integration projects. When you create a mapset in Oracle B2B, you associate it with trading partner agreements. When you associate a mapset with an endpoint or agreement, you can only select an mapset whose document definitions and message flow match that of the endpoint or agreement.

14.1.2 Predefined and Custom Mapsets

In Oracle B2B, you have the option of creating your own custom mapsets using the MapBuilder feature of the Oracle Document Editor or purchasing predefined map files provided by Edifecs. The prebuilt maps include maps for the Health Insurance Portability and Accountability Act (HIPAA); for example, to transform HIPAA 4010 messages to HIPAA 5010 messages.

14.2 Using Mapsets in the Oracle B2B Console

Use the Oracle B2B Console to create mapsets when you are mapping a data standard other than HL7 or when you want to use the features of Oracle B2B instead of the healthcare integration features. You can use this feature to map HIPAA 4010 message types to HIPAA 5010, for example. You can use Oracle B2B for mapping HL7 messages, but it is recommended you use Oracle SOA Suite for healthcare integration instead.

Perform the following steps to incorporate a mapset into an Oracle B2B trading partner agreement:

- 14.2.1 , "Creating a Mapset in the Oracle B2B Console"
- 14.2.2 , "Associating a Mapset with a Trading Partner Agreement"
- 14.2.3 , "Deleting a Mapset in the Oracle B2B Console"

14.2.1 Creating a Mapset in the Oracle B2B Console

Before you begin this step, make sure the map file is available on the computer from which you are accessing the Oracle B2B Console, and that the required document definitions are already created in Oracle B2B for both of the standards you are mapping. The mapset cannot be created without these three components.

For more information about working with document definitions in Oracle B2B, see Chapter 8, "Using Document Protocols".

To create a mapset

1. On the Oracle B2B Console, click **Administration** and then click the **Mapset** tab.

Figure 14–1   Mapset Page of the Oracle B2B Console
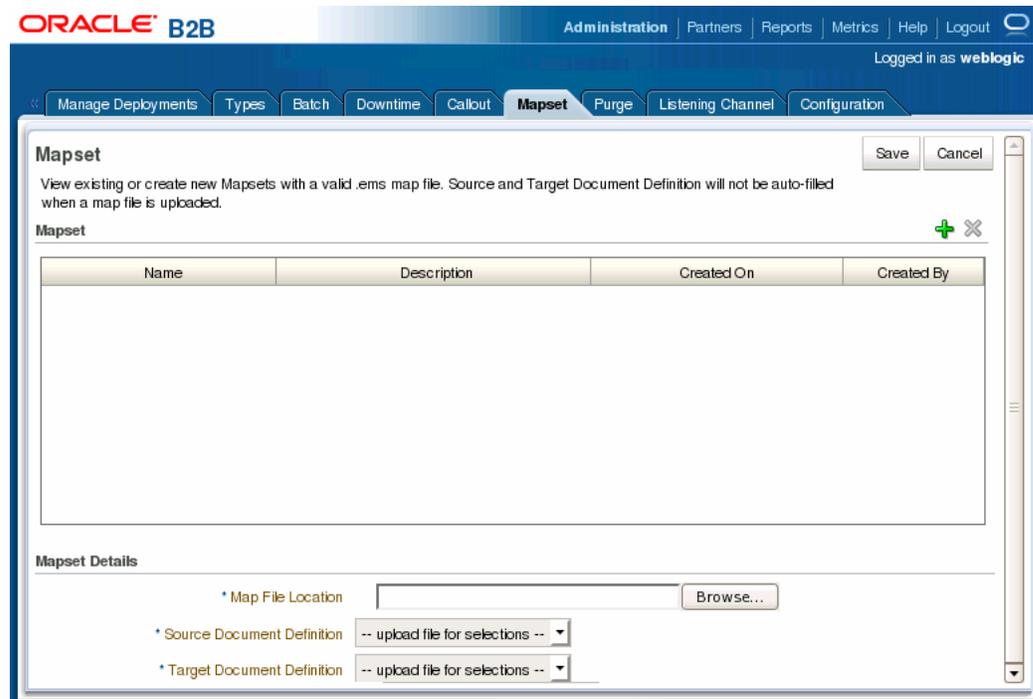
The image shows the Mapset page of the Oracle B2B Console. This page includes the name and description of the mapset as well as the date it was created and the user ID of the person who created it. It also shows the name of the map file and the document definitions being mapped.

****

2. On the Mapset page, click **Add Mapset** (the plus icon).

    A new row appears in the Mapset list.

3. In the new row, enter a name and a brief description of the new mapset.

4. Next to the **Map File Location** field, click **Browse**.

5. Browse to the location of your map file, select the file, and then click **Open**.

    The map file you selected is loaded, and the Source Document Definition and Target Document Definition fields are populated with the appropriate document definitions for the mapping.

Figure 14–2   Mapset Defined in the Oracle B2B Console
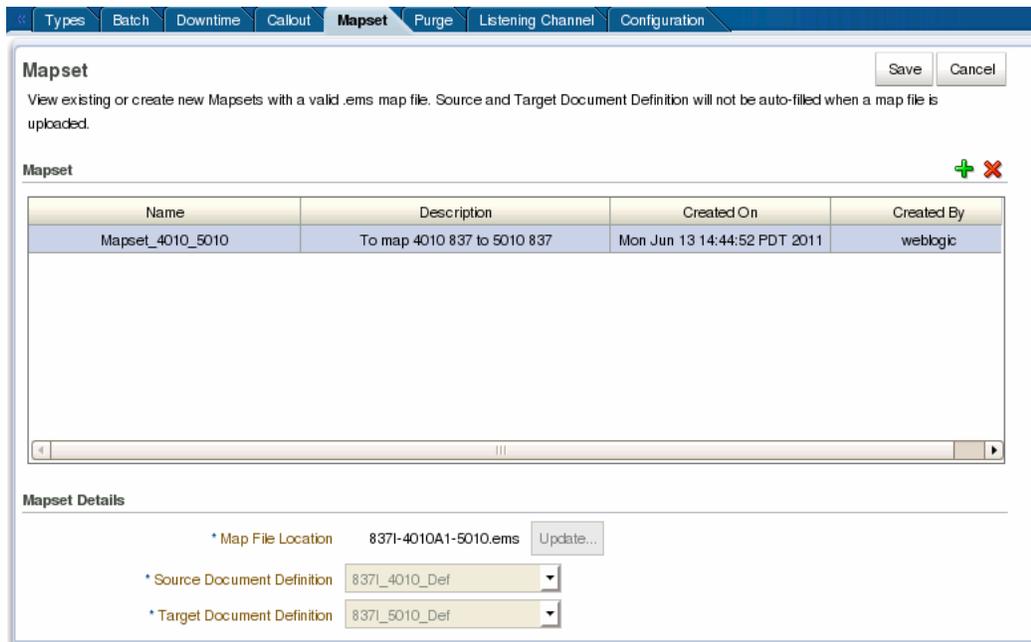
The image shows the Mapset page of the Oracle B2B console.

****

> **Note:**   If the document definitions do not already exist in Oracle B2B, this step will fail.

6.   Click **Save** and then click **OK** on the confirmation dialog that appears.

14.2.2 Associating a Mapset with a Trading Partner Agreement

You associate a mapset with a trading partner agreement in order to incorporate the mapping into the B2B processing logic. For information and instructions on working with trading partner agreements, see Chapter 6, "Creating and Deploying Trading Partner Agreements".

Before You Begin:

Make sure all required B2B components have been created and configured. At a minimum, this includes document definitions, trading partners, and the mapset.

To associate a mapset with a trading partner agreement

1.   On the Oracle B2B Console, select **Partners** in the upper right.

   The Partner page appears.

2.   In the Partner panel on the left, select the name of the remote trading partner, and then click the **Create New Agreement** icon in the Agreement panel.

   The Agreement page appears.

3.   Deselect **Translate**.

> **Note:** If Translation is selected, it is ignored when used in conjunction with a mapset in the endpoint. The mapset performs the data transformation and translation is not needed.

4. Select the trading partner and document definition for the agreement. For the document definition, select the native format that will be exchanged with the trading partner.

5. Click in the **Mapset** field and select the mapset you created for this agreement.

Figure 14–3   Mapset Page in Oracle B2B
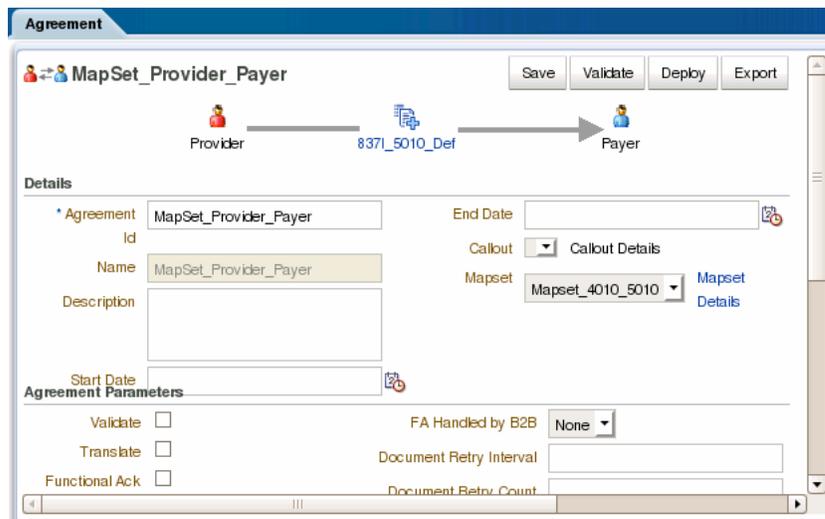
The image shows the Agreement page of the Oracle B2B Console. In the Mapset field, the mapset Mapset_4010_5010 is selected.

****

> **Note:** When using mapsets, whether the selected document definition is the source or target in the mapset depends on the direction of the message flow in the agreement. The above image shows an outbound message, so the selected document definition (HIPAA 5010 837) is the target document definition in the mapset, which maps HIPAA 4010 837 message to HIPAA 5010 837 messages.

6. To view information about the selected mapset, click **Mapset Details**.

A dialog appears with summary information for the mapset.

Figure 14–4 Mapset Details Dialog

The figure shows the Mapset Details dialog accessed from a trading partner agreement. The dialog shows the mapset name, descriptions, create date, user name, and the source and target document definitions.

****

7. Configure the agreement as described in Chapter 6, "Creating and Deploying Trading Partner Agreements".

8. When you are done configuring the agreement, click **Save** and then click **OK** on the confirmation dialog that appears.

9. To validate the agreement, click **Validate**.

10. When you are ready to deploy the agreement to the application server, click **Deploy**.

14.2.3 Deleting a Mapset in the Oracle B2B Console

Only mapsets that are not currently being used in an agreement can be deleted.

To delete a mapset in the Oracle B2B Console

1. On the Oracle B2B Console, click **Administration** and then click the **Mapset** tab.

2. On the Mapset page, select the row containing the mapset to delete.

3. Click **Delete Mapset** (the X icon) above the Mapset table and then click **Yes** on the Confirm Delete dialog.

4. Click **OK** on the confirmation dialog.

14.3 Using Oracle MapBuilder to Create Prebuilt Maps

You can create mapset files by using the Oracle MapBuilder application that is installed as a part of the Oracle Document Editor.

Please refer to the Oracle MapBuilder Help for more details about creating custom mapset files.

15

Purging Data

This chapter describes how to use the Oracle B2B interface to purge design metadata and instance data.

The chapter contains the following topics:

- Section 15.1, "Purging Design Metadata and Instance Data"

See the following for alternate methods of purging:

- Chapter 20, "B2B Command-Line Tools"

> **Caution:** Archiving is required before purging data. Purging is an irreversible operation. Ensure that you first archive any important data.

15.1 Purging Design Metadata and Instance Data

Use the Oracle B2B interface to purge design metadata and instance data. Design metadata contains partner profile data, identifiers, document definitions, channels, and agreements. When you purge this data, predefined data that is part of the installation (the host trading partner name, protocols, and identification types, for example) is not purged. Instance data is created during run time when messages are processed. Instance, or run-time, data contains the business messages and message-related data.

Specific instance data can be purged from the **Business Message** tab of the **Reports** link. See Section 18.1.2, "Purging Messages," for more information.

Purging does not remove artifacts that B2B creates in the Credential Store, such as passwords. See *Oracle Fusion Middleware Security Guide* for more information about the Credential Store.

With an instance message purge, you can optionally purge control number information. Control numbers are used in EDI (X12 and EDIFACT) and HL7 message standards. B2B keeps track of control numbers for inbound and outbound messages. For outbound messages, B2B generates the control numbers in a sequence from an internal control number table. Because purging instance data and control numbers resets the sequence (the control number table is reset), an outbound message after a purge may have the same control number as a message before the purge. If this is undesirable, do not purge control numbers.

Purging is useful for:

- Managing disk space and improving performance
- Removing repositories on a test system

To purge design metadata or instance data:

1. Click the **Administration** tab, and then the **Purge** tab.

2. (Optional if purging instance data.) Select **Purge Control Number** to reset the sequence.

3. Click **Purge Design Metadata** or **Purge Instance Data**, as shown in .

Figure 15–1   Purging Design Metadata or Instance Data

The image shows the Purge tab, which displays the Purge Design Metadata button and the Purge Instance Metadata button. The Purge Instance Metadata button is inaccurate. It should be labeled Purge Instance Data. The Purge Control Number checkbox is also displayed.

****

If you select **Purge Design Metadata**, then the message **Do you want to delete all the design metadata from the repository permanently?** appears.

If you select **Purge Instance Data**, then the message **Do you want to delete all the runtime data from the repository permanently?** appears.

4. Click **Yes**.

16

Configuring Listening Channels

This chapter describes how to create a listening channel to send messages to Oracle B2B. It also covers how to configure document sequencing.

A listening channel listens on an endpoint for messages. If a listening channel is marked as internal, then it can be used by any internal business application. If it is used as an external channel, then any trading partner can send a message to Oracle B2B using this channel.

This chapter contains the following topics:

- Section 16.1, "Adding a Listening Channel and Protocol"
- Section 16.2, "Using Transport Protocols"
- Section 16.3, "Adding Listening Channel Details"
- Section 16.4, "Configuring a Listening Channel"
- Section 16.5, "Configuring Document Sequencing"

16.1 Adding a Listening Channel and Protocol

Listening channels are used globally. You do not need to select a listening delivery channel in an agreement. Listening channels are used for any trading partner to send inbound messages to Oracle B2B or for any back-end business application to send outbound messages to Oracle B2B.

When you add a listening channel, you also specify the protocol that the channel uses, as shown in Figure 16–1.

Figure 16–1 Adding a Protocol for a Listening Channel
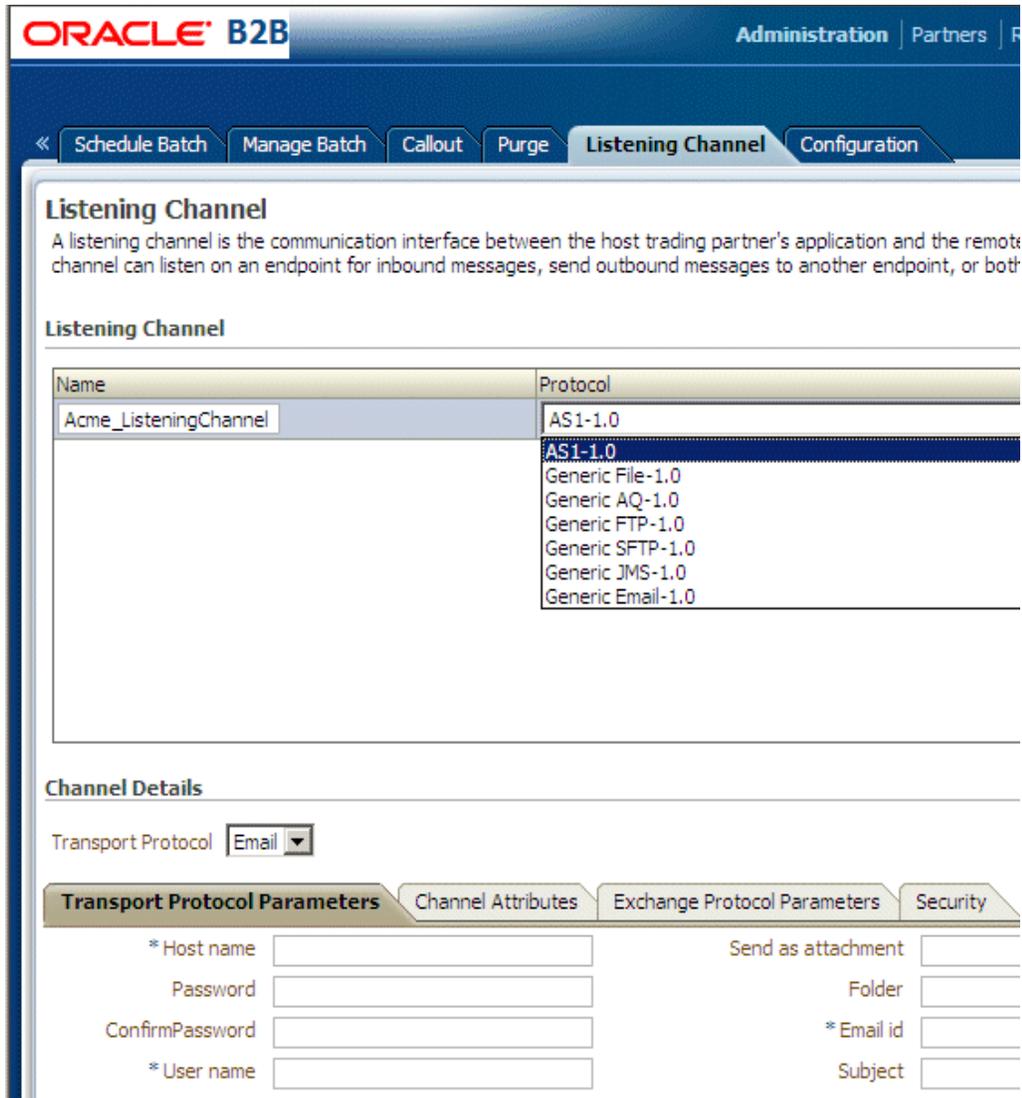
The image shows the following tabs on the Oracle B2B screen: Schedule Batch, Manage Batch, Callout, Purge, Listening Channel, and Configuration. The Listening Channel tab is selected. The first area of the tab shows the Listening Channel list as a table with the following columns: Name and Protocol. The second area of the tab shows the Channel Details and displays the Transport Protocol field (a drop-down list). The third area of the tab shows the following sub-tabs: Transport Protocol Parameters, Channel Attributes, Exchange Protocol Parameters, and Security. The Transport Protocol Parameters tab is selected and shows the following fields: Host name, Password, ConfirmPassword, User name, Send as attachment, Folder, Email Id, and Subject.

****

By using a global listening channel, you can keep all messages in one directory from which Oracle B2B pulls. This approach is useful for File, FTP, and SFTP (SSH FTP) exchanges.

Table 16–1 describes the listening channel protocols supported by Oracle B2B.
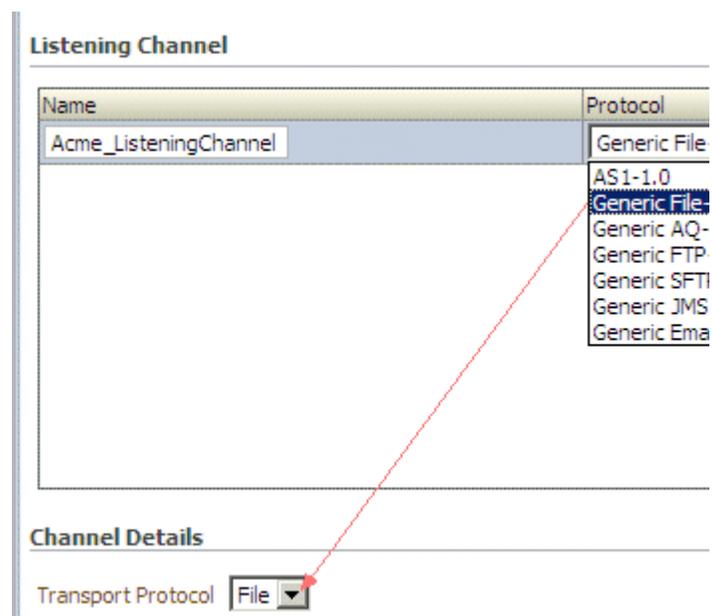
Table 16–1   Listening Channel Protocols

Protocol	Description
AS1-1.0	Applicability Statement 1 (AS1) provides S/MIME and uses SMTP to transmit data using e-mail. Security, authentication, message integrity, and privacy are assured by the use of encryption and digital signatures. Use nonrepudiation to make it impossible for the intended recipient of a message to deny having received it. AS1 works with almost any type of data.
Generic File-1.0, Generic AQ-1.0, Generic FTP-1.0, Generic SFTP-1.0, Generic JMS-1.0, Generic Email-1.0	Using the Generic options, you can send messages with or without security. The Generic exchange protocol supports MIME and S/MIME, including S/MIME 3.0-based signing and encryption. There is no receipt acknowledgment support with the Generic protocols (the acknowledgment mode must be set to None).

16.2 Using Transport Protocols

The transport protocol used to send the message is determined by the listening channel you select, as shown in the **Channel Details** area in Figure 16–2.

Figure 16–2   Channel Details: The Transport Protocol

The image shows a section of the Listening Channel tab with the Protocol drop-down menu expanded.

****

Table 16–2 describes the transport protocols available in Oracle B2B.

Table 16–2   Transport Protocols Available in Oracle B2B

Protocol	Description
Email	Use Email for AS1 and Email listening channels.
File	The File transport enables files to be picked up from a shared file directory.
AQ	Oracle AQ provides secure, bidirectional, asynchronous communication. The location of the application location is transparent, using any number of Oracle connectivity options, including OCI, JDBC, or PL/SQL. Both XML and non-XML message payloads are supported.

Table 16–2 (Cont.) Transport Protocols Available in Oracle B2B

Protocol	Description
FTP	FTP enables files to be passed with FTP between applications. FTP runs on default port 21. To change to another port, provide the value in the **Control Port** field. To enable SSL, use the **Channel Mask** field. The default is None (no SSL).
SFTP	SFTP enables files to be passed using SSH FTP. SFTP runs on default port 22, which can be changed to another port. SFTP supports two modes of authentication, password authentication and public key authentication. To use password authentication, provide a password, which is used for authentication. To use public key authentication, provide the private key file location. You may also need to provide a pass phrase if the private key file is pass-phrase protected.
JMS	JMS enables applications to send and receive messages to and from the queues and topics administered by any Java Message Service (JMS) provider, including Oracle WebLogic JMS and non-Oracle providers such as MQSeries JMS (IBM). If a user name and password are not provided, the local JNDI is used, including in a clustered environment, provided that the destinations are distributed. Oracle B2B does not support `javax.jms.ObjectMessage`.

16.3 Adding Listening Channel Details

Listening channel details include transport protocol parameters, channel attributes, exchange protocol parameters, and security specifications.

16.3.1 Transport Protocol Parameters

A transport protocol defines the properties specific to a given use of a protocol endpoint. The transport is responsible for message delivery using the selected transport protocol, mode (synchronous or asynchronous), server, and protocol endpoint address (the trading partner address, such as a URI). Table 16–3 describes the transport protocol parameters and lists the protocols to which the parameters apply.

Table 16–3   Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Archival Directory	B2B channels move the processed files to this directory. By default, it is a destructive read—processed files are deleted from the endpoint. In this case, files are moved to the path provided.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Cache Connections	If enabled, file listing and processing of the file occur in the same session (contrary to the default, in which listing and processing occur in different sessions).	Generic FTP-1.0 (optional)
Channel mask	To enable SSL for FTP, enter one of the following: ■ `Control`—Encrypts the control channel ■ `Data`—Encrypts the data channel ■ `Both`—Encrypts both the data and control channels The default is **None** (no SSL).	Generic FTP-1.0 (optional)
Cipher suites	Sets of ciphers defined in SSL.	Generic FTP-1.0 (optional)
Connection factory	The JNDI location or Java class name for the connection factory, as in `jms/b2b/B2BQueueConnectionFactory`.	Generic JMS-1.0 (optional)
Consumer	The client that receives the message.	Generic AQ-1.0 (optional)
Content type	The content type of the payload being sent over e-mail. The default content type is `text/plain`; other examples include `application/xml` and `application/edi`. This value is used only for the delivery channel (to send e-mail) and not for the listening channel. On the listening channel side, intelligence is built into the transport adapter to deal with different content types, so no configuration is required.	AS1-1.0 (optional) Generic Email-1.0 (optional)

Table 16–3   (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Control port	Provide a value to change the default FTP port value (21)	Generic FTP-1.0 (optional)
Data port	For active FTP connections, use this option to configure the static/fixed data port of the FTP server.	Generic FTP-1.0 (optional)
Datasource	The JNDI name of the JDBC data source to access AQ queues.	Generic AQ-1.0 (optional)
Destination name	The JMS destination name.	Generic JMS-1.0 (optional)
Destination Provider	Enables B2B to connect to JMS queues or topics available on remote servers. JNDI properties required to connected to the target server are expected as the value. Use ; (semicolon) as the separator for each key/value pair.	Generic JMS-1.0 (optional)
Email ID	The e-mail address to which messages are delivered (similar to specifying the path for a file channel or queues in AQ or JMS).	AS1-1.0 (required) Generic Email-1.0 (required)
Email Server	Select **IMAP** or **POP3**.	AS1-1.0 (required) Generic Email-1.0 (required)
Enable CCC	Enables B2B to authenticate in an SSL session and do the rest of the file transfer commands on a plain socket.	Generic FTP-1.0 (optional)
Enable Marker	If enabled, creates a zero-byte file with the same name as the source, indicating completion of reading or writing. The file carries the same name as the source, but with the extension `marker`.	Generic File-1.0 (optional) Generic FTP-1.0 (optional)-1.0 Generic SFTP-1.0 (optional)
Encoding	The encoding used in B2B to convert the contents of the inbound files.	Generic FTP-1.0 (optional)

Filename format[1]	The following filename formats can be used:
	 `%FROM_PARTY%`
	 `%TO_PARTY%`
	 `%DOCTYPE_NAME%`
	 `%DOCTYPE_REVISION%`
	 `%MSG_ID%`
	 `%TIMESTAMP%`
	 This filename format can be used for ebMS documents only:
	 `%ACTIONNAME%`
	 These formats can be used in any combination; for example,
	 `%TO_PARTY%_%DOCTYPE_NAME%_%DOCTYPE_REVISION%.dat`
	 produces something like `Acme_4010_850.dat`. Any file extension is allowed.
	 See footnote below.
		Generic File-1.0 (optional)
		 Generic FTP-1.0 (optional)
		 Generic SFTP-1.0 (optional)

Folder	An absolute directory path is recommended.	AS1-1.0 (optional) Generic Email-1.0 (optional)
Folder name	An absolute directory path is recommended.	Generic File-1.0 (required) Generic FTP-1.0 (required)
Host name	The trading partner's transport or e-mail server exchanging messages.	AS1-1.0 (required) Generic AQ-1.0 (optional) Generic FTP-1.0 (required) Generic SFTP-1.0 (required) Generic Email-1.0 (required)
Is Binary	Treats the message as binary content, with no translation or validation. Agreements are identified based on the file naming convention.	This parameter is *not* available with Generic File-1.0, Generic FTP-1.0, and Generic SFTP-1.0, although it appears in the B2B interface for these protocols.
Is Map Payload Alone	Indicates that the payload is sent alone as part of a JMS message of type `javax.jms.MapMessage`	Generic JMS-1.0 (optional)

Table 16–3   (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Is topic	Select to indicate that JMS is communicating with a topic (not a queue).	Generic JMS-1.0 (optional)
Is Van Mailbox	If enabled, B2B treats the endpoint as a VAN Mailbox and operates accordingly.	Generic FTP-1.0 (optional)
Message type	Select a JMS messages type: **BYTES**, **TEXT**, or **MAP**.	Generic JMS-1.0 (optional)
Minimum Age	Files arriving at the endpoint are processed after the time interval entered, in milliseconds.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Pass phrase and Confirm pass phrase	If you enter a private key file location, and if the private key file is pass-phrase protected, then enter the pass phrase.	Generic SFTP-1.0 (optional)
Password and Confirm Password	To use password authentication, provide a key store password, which is used for authentication.	AS1-1.0 (optional) Generic AQ-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional) Generic JMS-1.0 (optional) Generic Email-1.0 (optional)
Path	The absolute directory path where messages are sent from or received.	Generic SFTP-1.0 (required)
Polling interval	The time interval in seconds during which Oracle B2B polls the server for inbound messages.	AS1-1.0 (optional) Generic File-1.0 (optional) Generic AQ-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional) Generic JMS-1.0 (optional) Generic Email-1.0 (optional)
Port number	AQ runs on default port 1521. SFTP runs on default port 22, which can be changed to another port. FTP runs on default port 21, which is not displayed. See the description of **Control Port** for how to change this port number.	Generic AQ-1.0 (optional) Generic SFTP-1.0 (required)
Preserve Filename	Retains the file name.	Generic File-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional)
Private key	To use public key authentication, provide the private key file location. You may also need to provide a pass phrase if the private key file is pass-phrase protected.	Generic SFTP-1.0 (optional)
Queue name	The AQ queue name.	Generic AQ-1.0 (optional)
Recipient	The value used when delivering a message to the AQ queue. For example, if you set the recipient to `testuser`, then the message can be consumed only by the consumer with the name `testuser` (in other words, the recipient is on the sending side and the consumer is on the listening side).	Generic AQ-1.0 (optional)
Send as attachment	If enabled, the message (payload) is sent as an e-mail attachment instead of the typical delivery in which the payload is the message body.	This parameter is *not* available with AS1-1.0 and Generic Email-1.0, although it appears in the B2B interface for these protocols.
SID	System ID to identify an Oracle database.	Generic AQ-1.0 (optional)

Table 16–3   (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Subject	The subject header of the e-mail message.	This parameter is *not* available with AS1-1.0 and Generic Email-1.0, although it appears in the B2B interface for these protocols.
Subscriber ID	The JMS subscriber ID is required if JMS is communicating with a topic.	Generic JMS-1.0
Transfer Type	Select **binary** or **ascii** for the file transfer mode.	Generic FTP-1.0 (optional)
Use JMS ID	Uses the JMS message ID as the B2B message ID. This facilitates correlation at the JMS level.	Generic JMS-1.0 (optional)
Use proxy	Select this option if a proxy server is used.	Generic FTP-1.0 (optional)
		Generic SFTP-1.0 (optional)
User name	The user name (login name) to connect to the target servers. This value is optional for AQ and JMS because B2B can use the configured JNDI data sources to connect to queues.	AS1-1.0 (required)
		Generic AQ-1.0 (optional)
		Generic FTP-1.0 (required)
		Generic SFTP-1.0 (required)
		Generic JMS-1.0 (optional)
		Generic Email-1.0 (required)

[1] In File/FTP channels, if the `filename format` is set then the `directory name format` is ignored.

16.3.2 Channel Attributes

The channel is the communication interface between the host trading partner's host application and its installation. Table 16–4 describes the channel attributes and lists the protocols to which the attributes apply.

Table 16–4   Channel Attributes

Protocol/Parameter	Description	Protocol Used With
Ack Mode	Select **Sync**, **Async**, or **None** for the mode in which the trading partner receives messages. Select **None** for all generic exchanges.	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.
Description	Provide an optional description.	AS1-1.0 (optional)
		Generic File-1.0 (optional)
		Generic AQ-1.0 (optional)
		Generic FTP-1.0 (optional)
		Generic SFTP-1.0 (optional)
		Generic JMS-1.0 (optional)
		Generic Email-1.0 (optional)
Enable/Disable Channel	The channel is the communication interface between the host trading partner's host application and its installation.	AS1-1.0 (required)
		Generic File-1.0 (required)
		Generic AQ-1.0 (required)
		Generic FTP-1.0 (required)
		Generic SFTP-1.0 (required)
		Generic JMS-1.0 (required)
		Generic Email-1.0 (Required)

Table 16–4   (Cont.) Channel Attributes

Protocol/Parameter	Description	Protocol Used With
Internal	Select this option if the channel is internal to the host trading partner's enterprise. This feature is disabled for AS1.	Generic File-1.0 (optional) Generic AQ-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional) Generic JMS-1.0 (optional) Generic Email-1.0 (optional)
Response Mode	Select **Sync**, **Async**, or **None**,	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocols.
Retry Count	The number of times that Oracle B2B retries sending the message.	This parameter is *not* available with AS1-1.0, Generic File-1.0, Generic AQ-1.0, Generic FTP-1.0, Generic SFTP-1.0, Generic JMS-1.0, and Generic Email-1.0, although it appears in the B2B interface for these protocols.
Retry Interval	The time interval in seconds during which Oracle B2B attempts to resend the message. A time interval of 2 minutes increments the HH:MM:SS timestamp as follows: If the sent timestamp is 3:42:58, then 42 minutes is incremented by 2 minutes and the retry is sent at 3:44:00. The seconds are dropped in the retry increment. Subsequent retries are at 2 minute intervals. For protocols with acknowledgments, B2B waits for the acknowledgment (formerly called the Time to Acknowledge parameter). If it is not received, the retry interval setting causes B2B to retry	This parameter is *not* available with AS1-1.0, Generic File-1.0, Generic AQ-1.0, Generic FTP-1.0, Generic SFTP-1.0, Generic JMS-1.0, and Generic Email-1.0, although it appears in the B2B interface for these protocols.
Transport Callout	For the inbound message, B2B invokes the transport callout immediately after it receives a message from the transport. For the outbound message, B2B invokes the transport callout immediately before it sends a message to the transport.	AS1-1.0 (optional) Generic File-1.0 (optional) Generic AQ-1.0 (optional) Generic FTP-1.0 (optional) Generic SFTP-1.0 (optional) Generic JMS-1.0 (optional) Generic Email-1.0 (optional)

16.3.3 Exchange Protocol Parameters

The exchange protocol defines the headers, acknowledgments, and packaging that puts the headers and payload together (the message exchange mechanism). The exchange protocol also defines signing and compression. Table 16–5 describes the exchange protocol parameters and lists the protocols to which the parameters apply.

Table 16–5   Exchange Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Signed and Compressed	If selected, the message is first signed, and then compressed. If *not* selected, the message is first compressed, and then signed.	AS1-1.0 (optional)

16.3.4 Security Parameters

Security parameters are not available for any of the protocols, although the B2B interface displays security parameters for the AS1-1.0 protocol, as described in Table 16–6.

Table 16–6   Security Parameters: Not Available for Listening Channel Protocols

Protocol/Parameter	Description	Protocol Used With
Ack Signed	Select this option to ensure that the responder acknowledges receipt of the messages; nothing needs to be provided.	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.
Digital Signature	If **Message Signed** is selected, then select one of the following: SMIME 3.0 with MD5 - RSA SMIME 3.0 with SHA1 - RSA	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.
Encryption	If **Message Encrypted** is selected, then select one of the following: SMIME 3.0 with DES SMIME 3.0 with 3DES SMIME 3.0 with RC2 - 40 SMIME 3.0 with RC2 - 64 SMIME 3.0 with RC2 - 128	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.
Message Encrypted	Select this option to enable message encryption. This option requires you to select an encryption schema in the **Encryption** field.	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.
Message Signed	Select this option to provide one of the digital signatures in the **Digital Signature** field.	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for this protocol.

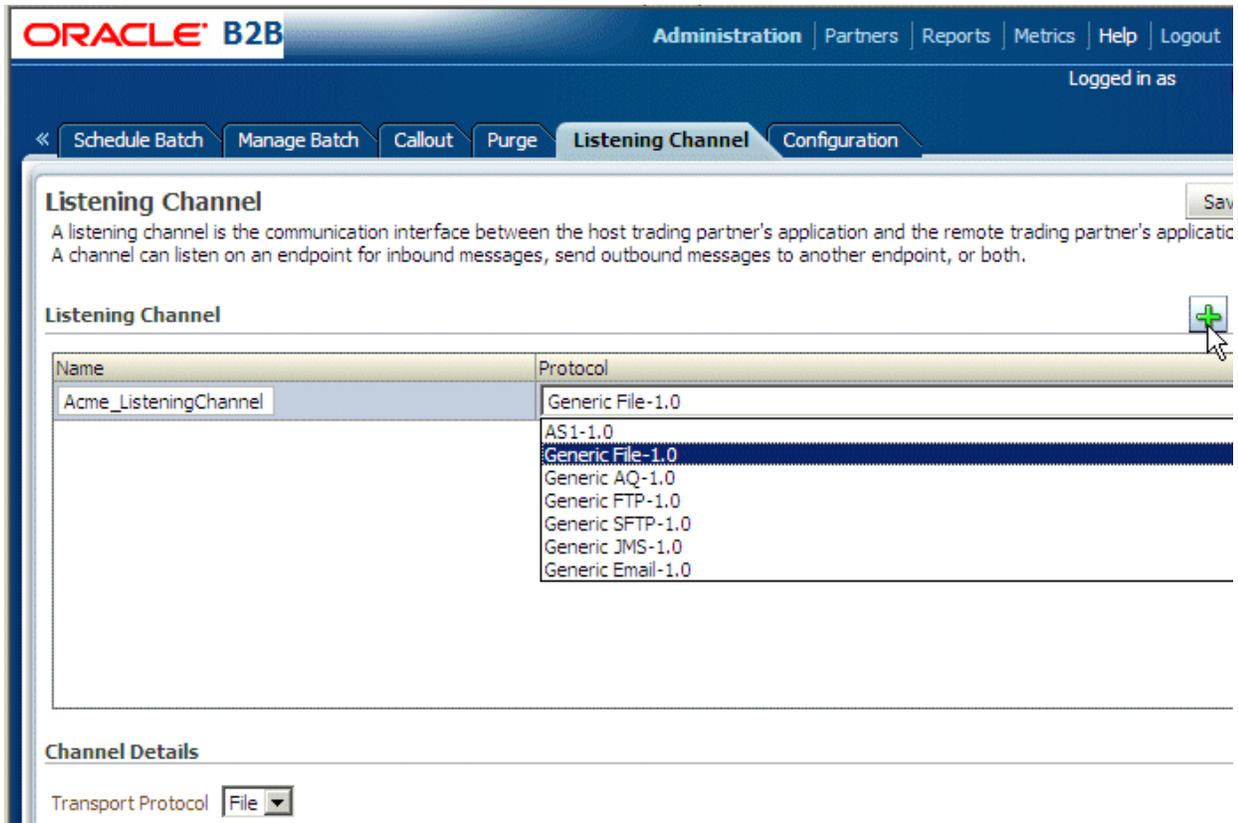## 16.4 Configuring a Listening Channel

To configure a listening channel, add a listening channel protocol, and then transport protocol parameters, channel attributes, exchange protocol parameters, and security parameters, depending on the channel protocol you selected.

To add a listening channel protocol:
1. Click the **Administration** link.
2. Click the **Listening Channel** tab.
3. Click **Add**.
4. Provide a name for the listening channel.
5. Select a protocol.

   Figure 16–3 shows the list of protocols.

Figure 16–3 Adding a Listening Channel Protocol

The image shows the following tabs on the Oracle B2B screen: Schedule Batch, Manage Batch, Callout, Purge, Listening Channel, and Configuration. The Listening Channel tab is selected. The first area of the tab shows the Listening Channel list as a table with the following columns: Name and Protocol. The Protocol drop-down list is expanded and shows the following options: AS1-1.0, Generic File-1.0, Generic AQ-1.0, Generic FTP-1.0, Generic SFTP-1.0, Generic JMS-1.0, and Generic Email-1.0.

****

See Table 16–1 for a description of the protocols.

The transport protocol that appears under **Channel Details** is based on your protocol selection in Step 5.

6. Click **Save**.

To add transport protocol parameters:

1. Click the **Transport Protocol Parameters** tab.

2. Provide transport protocol parameters, depending on the channel/transport protocols.

   Table 16–3 describes the transport protocol parameters (listed in alphabetical order) and the protocols to which the parameters apply.

3. Click **Save**.

To add channel attributes:

1. Click the **Channel Attributes** tab.

2. Provide channel attributes, depending on the channel/transport protocols selected.

   Table 16–4 describes the channel attributes (listed in alphabetical order) and the protocols to which the attributes apply.

3. Click **Save**.

To add exchange protocol parameters:

1. Click the **Exchange Protocol Parameters** tab.

2. Provide exchange protocol parameters, depending on the channel/transport protocols selected.

   Table 16–5 describes the exchange protocol parameters (listed in alphabetical order) and the protocols to which the attributes apply.

3. Click **Save**.

16.5 Configuring Document Sequencing

There are two options available to enable document sequencing for an AQ or JMS Trading Partner-facing listening channel.

- The Trading Partner can send messages with sequence target if inbound message sequencing is desired.

- The host Trading Partner can enable the sequencing option in the AQ or JMS Trading Partner-facing listening channel to sequence inbound messages. In this case select the queue name as the sequence target.

Note that if both sequencing options are enabled then first option takes precedence.

For a full discussion of sequencing on Trading Partner channels see Section 5.5.2, "Message Sequencing."

17

Configuring B2B System Parameters

This chapter describes how to access configuration settings that are available in the Oracle B2B interface on the **Configuration** tab.

Settings on the **Configuration** tab override property settings configured in the Oracle Enterprise Manager Fusion Middleware Control Console. See Appendix C, "Setting B2B Configuration Properties in Fusion Middleware Control" for information about properties set in Oracle Enterprise Manager Fusion Middleware Control Console.

This chapter contains the following topics:

- Section 17.1, "Setting Configuration Parameters"

17.1 Setting Configuration Parameters

Table 17–1 shows the configuration settings available in the Oracle B2B interface.

Figure 17–1 Configuration Parameters in the Oracle B2B Interface

The image shows the following tabs on the Oracle B2B screen: Deploy, Manage Deployments, Types, Batch, Downtime, Callout, Purge, Listening Channel, and Configuration. The Configuration tab is selected. The tab is divided into nine areas: Acknowledgement (upper left), Miscellaneous (upper right), Miscellaneous (continued) (middle left), Non Purgeable (center right), Performance (lower left), and UI (lower right). The fields in each section of the tab are described in Table 17–1. A Save button appears at the top right of the screen.

****

Table 17–1 describes the configuration parameters.

Table 17–1   Configuration Settings

Field	Description
Acknowledgment	-
Functional Ack Handled by B2B	To specify how a functional acknowledgment is handled, set this property to true or false.
	If set to true, then B2B autogenerates the functional acknowledgment (FA) message for inbound EDI and HL7 messages. Inbound FA messages are consumed when this option is true.
	When this option is set to false, B2B does not autogenerate the FA document. The back-end application (middleware) must generate the FA and provide it to B2B as an outbound message. When option is set to false, inbound FA documents are passed back to the back-end application.
	If the document does not require an FA (as indicated by the agreement-level setting), then this option is ignored. The default value for this property is true.
	If the FA is marked as expected in an agreement, then the message is placed into the MSG_WAIT_FA state and the back-end application is expected to generate the FA and push it to B2B as an outbound message back to the partner.
	See Section C.1, "Properties To Set in Fusion Middleware Control," for more information.
	When **Functional Ack Handled by B2B** is set to false, then **Notify Inbound Functional Acks** must be set to false also for the inbound FA to be sent to the back-end application. If **Notify Inbound Functional Acks** is set to true (while **Functional Ack Handled by B2B** is set to false), then the incoming 997 (FA doc) generates only a notification and the 997 document itself is *not* sent back to the back-end application.
	The following limitations apply when generating the FA from the back-end application:
	■ The FA is correlated with the original message based on the ReferToMsgID value set in the enqueue properties. The FA is correlated based on control numbers also.
	■ If the FA indicates that there was an error in the received message, the status of the correlated message is not updated to indicate an error. The correlated message is updated to MSG_COMPLETE.
	These limitations are not present when the FA is generated by B2B (that is, when this property is set to `true`).
Functional Ack Internal Properties	Generates the internal properties structure in the functional acknowledgment XML for EDI transactions. A document type 997 (for X12) or CONTRL (for EDIFACT) must exist.
	The default value is false, which means that the functional acknowledgment uses the original message-internal properties.
	If true, then the FA message autogenerated by B2B contains interchange/group envelope information from the original message. To swap the interchange and group IDs in the generated FA from the sender to the receiver and vice versa, set this property to `true`.
	The test usage indicator parameter is also mapped; that is, an inbound message set as a test message generates an FA with the flag set to test message. Setting this property to `true`, by default, does not map ISA segment elements (1-4).
Notify Inbound Receipt Acks	If set to true, B2B sends an acknowledgment notification to the application when an exchange acknowledgment is received.

Table 17–1 (Cont.) Configuration Settings

Field	Description
Notify Inbound Functional Acks	If set to true, B2B sends an acknowledgment notification to the application when a functional acknowledgment is received.
	When **Functional Ack Handled by B2B** is set to false, then **Notify Inbound Functional Acks** must be set to false also for the inbound FA to be sent to the back-end application. If **Notify Inbound Functional Acks** is set to true (while **Functional Ack Handled by B2B** is set to false), then the incoming 997 (FA doc) generates only a notification and the 997 document itself is *not* sent back to the back-end application.
Miscellaneous	-
Default Trading Partner	Defaults to this trading partner if trading partner agreement identification fails. Used for HL7 documents.
Ignore Correlation	When an acknowledgment is received from a trading partner, it is correlated to the actual business message of the sender. If the correlation fails, an exception is generated and the acknowledgment processing stops. To ignore the correlation and process the acknowledgment, set this property to true.
Additional MIME Types	Use to specify attachments (additional MIME types) in addition to the default MIME types supported by B2B for ebxml exchanges. By default, B2B supports `text/plain : image/jpeg: text/xml : application/xml : application/octet-stream : application/EDIFACT : application/EDI-X12 : application/jpg : application/gzip : application/x-gzip : application/pkcs7-signature`.
Log Payload	If true, B2B logs the payload in a diagnostic log (also depends on log level setting). Error messages are logged by default. Payload logging is useful for diagnostic purposes, but may be undesirable for security reasons. The default value is false.
Reconnect on Error	If set to true, the AQ adapter retries the enqueue operation when the initial enqueue fails. This parameter is not available in this release.
HTTP Header Delimiter	A delimiter to separate the HTTP headers provided in the **Additional Transport Headers** field for HTTP delivery channel configuration.
Treat Reply to Message as Request	Used in ebMS to indicate that the conversation message is to be considered as a request message.
Generic Message Type	If this property is enabled (set to true), B2B finds the agreement for the specific message type first, and then the generic message type. The default value is false.
Miscellaneous (continued)	-
Outbound Dispatcher Count	The number of dispatchers used for handling the outbound messages. Used in message sequencing for MLLP. The default value is 0.
Inbound Dispatcher Count	The number of dispatchers used for handling the inbound messages. Used in message sequencing for MLLP. The default value is 0.
Auto Stack Handler	Used in stacking for MLLP. If true, the stack handler processes stacked messages in automatic mode. The default value is false.
Auto Stack Handler Interval	Used in stacking for MLLP. Enter comma-separated values for the time interval in seconds for the stack handler to process the stacked messages. The default value is 1.
Partial Batch Commit Size	Used while debatching an EDI message that has large transaction sets. Instead of using a single commit, which is prone to errors in case of failure, this feature enables specifying a partial batch commit size. For example, if Partial Batch Commit Size=10, then, for an envelope with more than 10 transactions, B2B calls a separate commit for every 10 transaction sets.

Table 17–1   (Cont.) Configuration Settings

Field	Description
Exception Queue	Select a JMS internal delivery channel for the host trading partner to use as the exception queue. A null default value for this parameter means that exceptions are sent to the JMS queue (B2B_IN_QUEUE) if **Use JMS Queue as default** is set to true or to the AQ queue (IP_IN_QUEUE) if **Use JMS Queue as default** is set to false.
	AQ queues are not supported for use as custom exception queues.
Enable BAM	Enables B2B to send runtime information to Oracle BAM. See Chapter 24, "Monitoring Instance Message Data With Oracle BAM" for more information.
BAM Polling Interval	Polling interval in minutes for Oracle BAM. See Chapter 24, "Monitoring Instance Message Data With Oracle BAM" for more information.
Non Purgeable	The nonpurgeable parameters retain their values even after a metadata repository purge is invoked.
Use JMS Queue as default	If this option is set to true, then B2B starts to poll on the JMS queue, B2B_OUT_QUEUE for outbound messages, and delivers all inbound messages to B2B_IN_QUEUE. Polling on IP_OUT_QUEUE is stopped.
	If this option is set to false (the default), then B2B starts to poll on the AQ queue, IP_OUT_QUEUE for outbound messages, and delivers all inbound messages to IP_IN_QUEUE. Polling on B2B_OUT_QUEUE is stopped. When a non-Oracle database is used and therefore no AQ queues are available, the JMS queues are used no matter how this option is set.
	If you select a queue from **Exception Queue**, then exception messages are sent to that configured queue. The default queues continue to be valid for other messages, depending on the setting for **Use JMS Queue as default**.
	If the value of **Use JMS Queue as default** is set to true before purging the metadata repository, then after a purge, the value continues to be true and does not revert back to the default value, false.
Callout Directory	Specify a directory for the callout JAR file location if you do not use the default callout. The callout directory path cannot end with / or \.
	The default file location, /MyCalloutDir, is retained after purging the metadata repository.
SMTP Host	Specify the host name of the SMTP server in the enterprise to send the negative MDN to the trading partner for an AS1 exchange.
Webservice Policy	Specify a security policy to secure the Web service. Enter only a security policy URI or complete <policy> tag. For example:
	oracle/wss_username_token_service_policy
	See Section 21.5, "Securing Oracle B2B Web Services" for more information about the use of this field.
Performance	-
Large Payload Size	Specify a large payload size, in bytes. The default value is 2,000,000 (2MG).
Large Payload Directory	The default directory is /tmp. For Windows-based systems, change the directory to an appropriate directory, such as C:\temp.
UI	-
Show Payload	Enables the payload to be displayed in reports accessible from the **Reports** tab. If set to true, the database is automatically searched with the default search parameters and the results are displayed.
Enable Auto Search	Enables automatic searching in reports accessible from the **Reports** tab. The default value is true. If set to false, a blank result table is displayed on the report pages until the **Search** button is clicked.
Payload Display Size	The default value is 1,048,576 KB. This parameter (in bytes) is used to display the payload only if its size is less than the value configured in the interface.

To set configuration parameters:

1. Click the **Administration** link.

2. Click the **Configuration** tab.

3. Provide values for the configuration parameters, as described in Table 17–1.

4. Click **Save**.

Part IV

Reports and Metrics

This part contains the following chapters:

-
-

18

Creating Reports

This chapter describes Oracle B2B reports that provide real-time status on the run-time behavior of deployed data. It discusses business message report, wire message report, application message report, error report, and conversation report.

The chapter contains the following topics:

- Section 18.1, "Introduction to Reports"
- Section 18.2, "Creating Business Message Reports"
- Section 18.3, "Creating Wire Message Reports"
- Section 18.4, "Creating Application Message Reports"
- Section 18.5, "Creating Error Reports"
- Section 18.6, "Creating Conversation Reports"

18.1 Introduction to Reports

Use the **Reports** link to search on data in the run-time repository. The Saved Search function is not available.

The following message types are available for searching:

- Business messages—See Section 18.2, "Creating Business Message Reports"
- Wire messages—See Section 18.3, "Creating Wire Message Reports"
- Application messages—See Section 18.4, "Creating Application Message Reports"
- Error messages—See Section 18.5, "Creating Error Reports"
- Conversation messages—See Section 18.6, "Creating Conversation Reports"

> **Note:** In a cluster environment, if system time stamps are not synchronized for all nodes in the cluster, then you may see message time stamps that look incorrect, but are not. For example, given an unsynchronized, multinode cluster, if an outbound message is received on one node, but the reply is sent from another node, it is possible for a report to show message receipt at 4 a.m., but an acknowledgment sent at 3:55 a.m.

18.1.1 The Monitor User Role

For individuals such as business analysts who create and analyze message reports, Oracle B2B provides a monitor user role that an administrator can assign to trading

partner users. This role provides a user with access to only the functionality of the **Reports** tab of Oracle B2B. A user with the Monitor role cannot see or access the other parts of the interface or see data for other trading partners. See Section 5.3, "Adding Trading Partner Users," for how to assign the Monitor role.

18.1.2 Purging Messages

From the **Business Message** tab, use the **Purge** button to purge one or more messages that display after you search the instance data.

18.1.3 Resubmitting Messages from Oracle B2B

If errors that occur when sending an inbound or outbound message are internal to Oracle B2B, then you can correct the problem and resend the message. For example, if B2B attempts to send a message to an endpoint that is not configured correctly, or if the agreement is not configured correctly, correct the error and use **Resubmit** for application messages or wire messages.

Resubmitting an application message, for an outbound message, replays the message from the time of receipt of the message and goes through agreement lookup, message translation (for EDI) and then finally the delivery is attempted. An application message resubmit is helpful when the agreement settings or document configuration is not as required and the message must be restructured with updated settings.

Resubmitting an application message, for an inbound message, attempts to deliver the message again to the back-end application. Resubmitting is useful when the back-end application is down and the delivery must be retried.

Resubmitting a wire message, for an outbound message, tries to redeliver only the previously processed message. There is no repackaging or other message transformation. This is helpful when the problem was with the delivery endpoint (for example, the partner's server is down and unable to receive the message).

Resubmitting a wire message, for an inbound message, replays the message from the time of receipt from the trading partner. The exchange and document are re-identified and an agreement lookup is done. The processed message is then delivered to the back-end. This is useful when the agreement or document setting are not correct and the message must be translated and validated again.

> **Note:** If you resubmit an inbound AS2 synchronous wire message, the MDN is generated, but it is not returned to the sender in synchronous mode. This is because the sender is not the one who is initiating the originating message. In this scenario, the MDN message state is in the `MSG_COMPLETE` state.

18.2 Creating Business Message Reports

Business message status reports identify business message instance details for a document protocol. These details include the sending and receiving trading partners, the agreement name, the business action, the business message ID, the status, the exchange protocol and document protocol, and message details.

Figure 18–1 shows a business message report.

Figure 18–1 Business Message Report
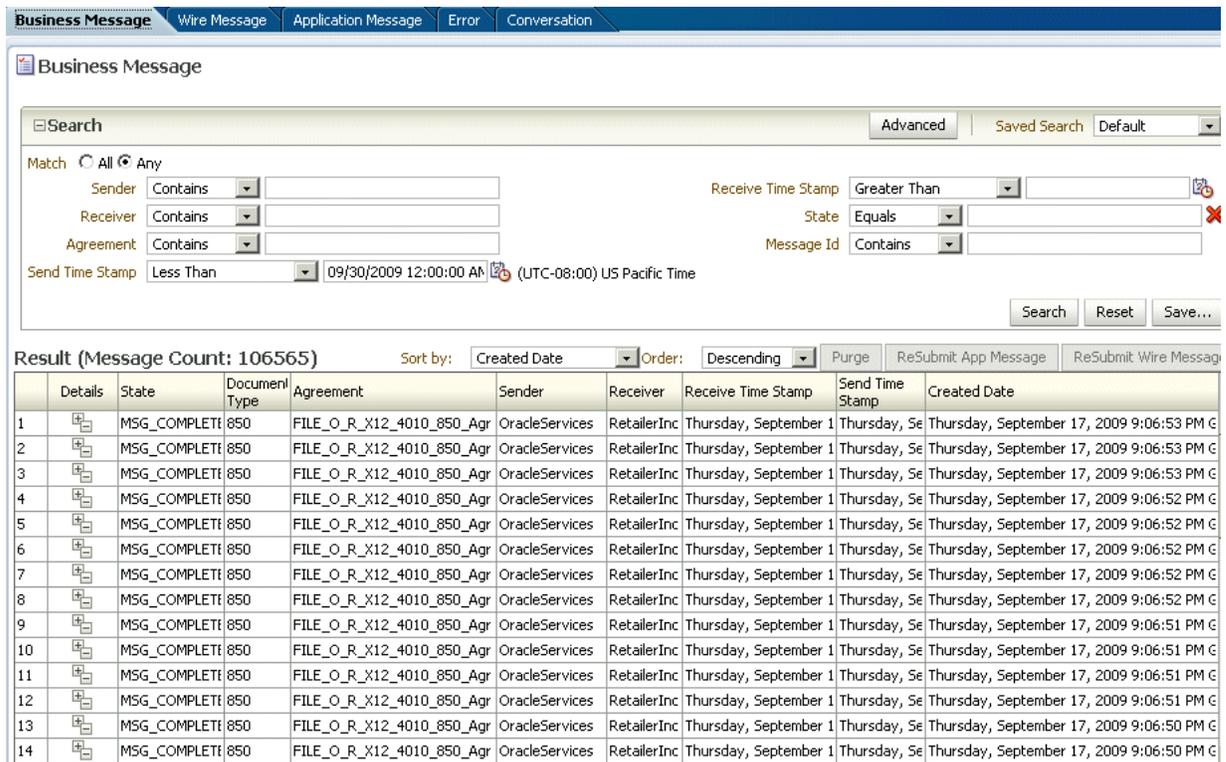
The image shows the following tabs on the Oracle B2B screen: Business Message, Wire Message, Application Message, Error, and Conversation. The following Search fields are displayed: Sender, Receiver, Agreement, Send Time Stamp, Receive Time Stamp, State, and Message Id. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Details, State, Document Type, Agreement, Sender, Receiver, Receive Time Stamp, and Send Time Stamp.

****

To create a business message report:

1. Click **Reports**, and then **Business Message**.

2. Provide search parameters.

Field	Description
Match	Select **All** or **Any**.
Sender	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a trading partner name.
Receiver	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a trading partner name.
Agreement	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a trading partner agreement name.
Send Time Stamp	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.

Field	Description
Receive Time Stamp	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.

State	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message state:
	
	`MSG_COMPLETE`
	`MSG_ERROR`
	`MSG_WAIT_TRANSMIT`
	`MSG_WAIT_FA`
	`MSG_WAIT_BATCH`

Message ID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message ID.

3. To add more search fields, click **Advanced** and select from **Add Fields**.

Field	Description
Document Protocol Name	Enter Custom, EDI_EDIFACT, EDI_X2, HL7, OAG, PositionalFlatFile, RosettaNet, or UCCNet. (**Equals** is the only operator.)
Document Type	Select from a previously created document type, for example, 850 for EDI X12. (**Equals** is the only operator.)
Document Protocol Version	Select from a previously created document protocol version. (**Equals** is the only operator.)
Document Definition	Select from a previously created document definition. (**Equals** is the only operator.)

Use the document search parameters as follows: Select a document protocol name first to populate the list of document protocol versions; next select a document protocol version to populate the list of document types; and then select a document type to populate the list of document definitions.
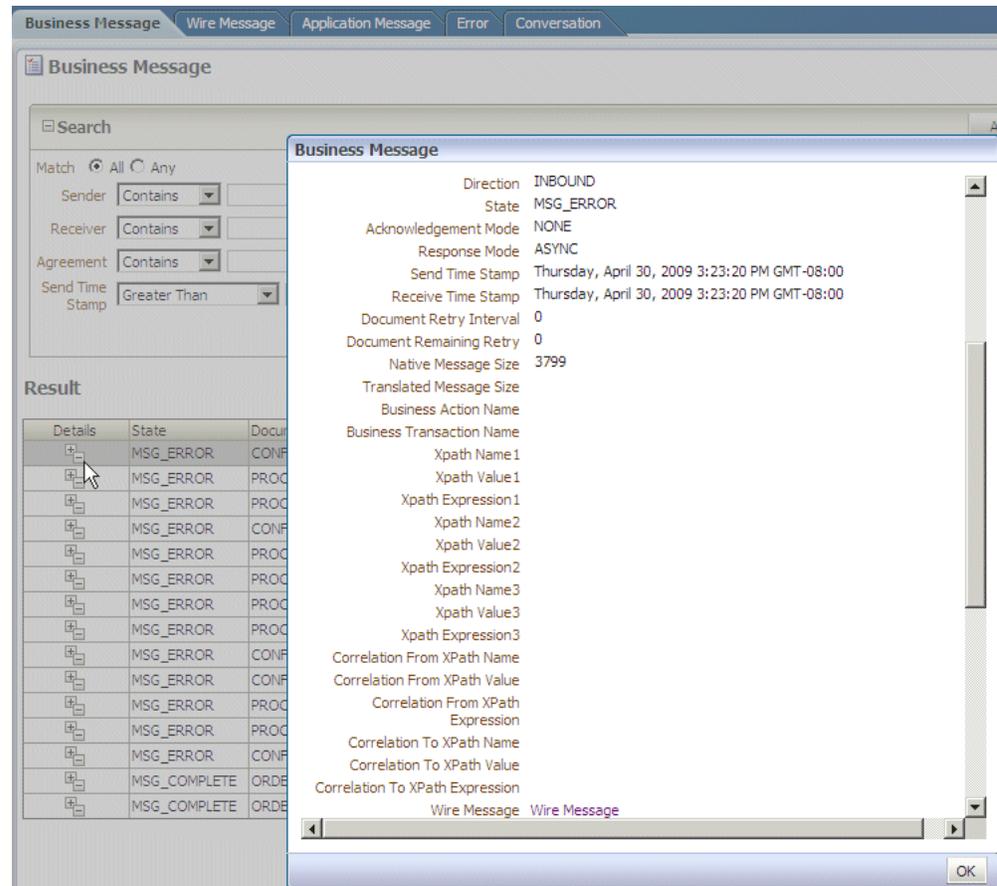
4. Click **Search**.

View the results, as shown in Figure 18–1.

5. In the **Details** column of the **Results** area, click the icon to see report details.

Figure 18–2 shows the business message details.

Figure 18–2   Business Message Details

The image shows details of the business message. The following fields are displayed: Direction: Inbound; State: MSG_ERROR; Acknowledgement Mode: None; Response Mode: Async; Send Time Stamp: Thursday, April 30, 2009 3:23:20 PM GMT-08:00; Receive Time Stamp: Thursday, April 30, 2009 3:23:20 PM GMT-08:00; Document Retry Interval: 0; Document Remaining Retry: 0; Native Message Size: 3799; Translated Message Size; Business Action Name; Business Transaction Name; Xpath Name1; Xpath Value1; Xpath Expression1; Xpath Name2; Xpath Value2; Xpath Expression2; Xpath Name3; Xpath Value3; Xpath Expression3; Correlation From XPath Name; Correlation From XPath Value; Correlation From XPath Expression; Correlation to XPath Name; Correlation to XPath Value; Correlation To XPath Expression; Wire Message: Wire Message. The image shows an OK button at the bottom right.

****

18.3 Creating Wire Message Reports

Wire messages are the native format of data sent from trading partners. Wire messages can contain several sections, such as payloads, attachments, or trailers. Wire message status reports identify details about wire message instances, such as the transport protocol name, the transport protocol revision, and the protocol message identification and its state. The reports enable you to go from a business message to its corresponding wire message and from a wire message to its corresponding business messages.

Figure 18–3 shows a wire message report.

Figure 18–3   Wire Message Report
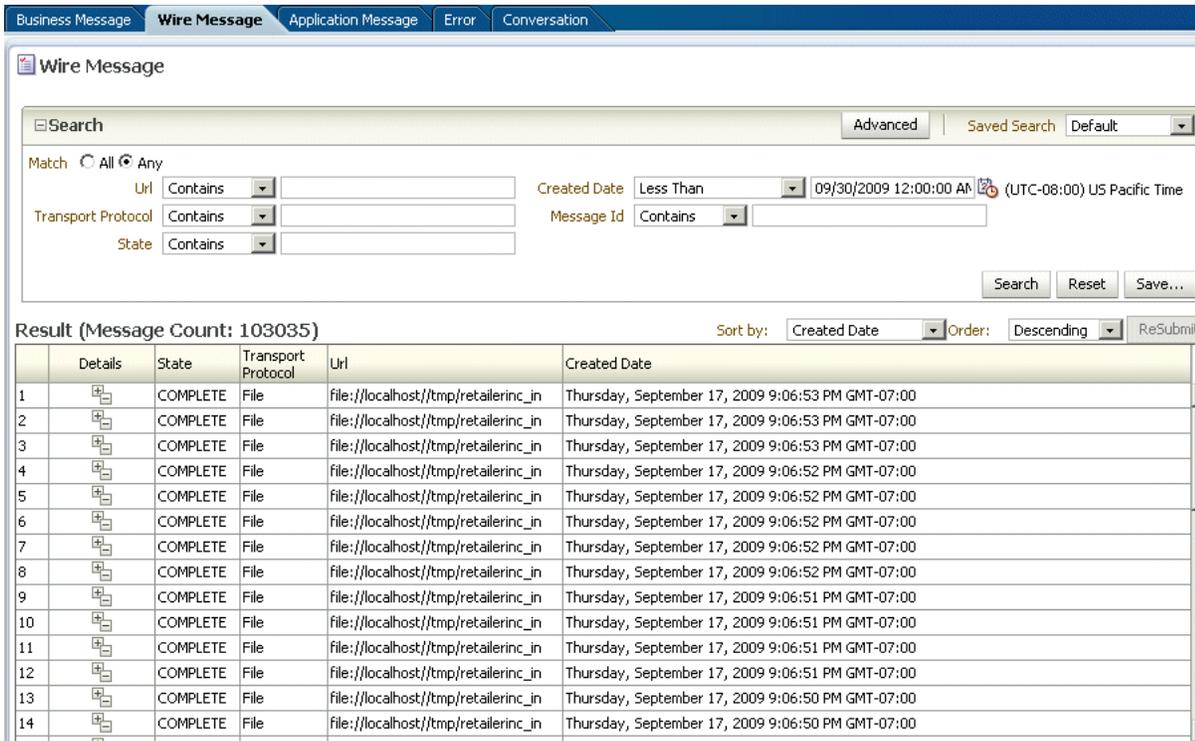
The image shows the following tabs on the Oracle B2B screen: Business Message, Wire Message, Application Message, Error, and Conversation. The Wire Message tab is selected. The following Search fields are displayed: Url, Transport Protocol, State, Create Date, and Message Id. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Details, State, Transport Protocol, Url, and Created Date.

****

To create a wire message report:

1. Click **Reports**, and then **Wire Message**.

2. Provide search parameters.

| Field	Value
URL	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the URL.
Transport Protocol	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the transport protocol.

State	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message state:
	
	`MSG_COMPLETE`
	`MSG_ERROR`
	`MSG_WAIT_TRANSMIT`
	`MSG_WAIT_FA`
	`MSG_WAIT_BATCH`

Field	Value
Created Date	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.
Message ID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message ID.

3. To add more search fields, click **Advanced** and select from **Add Fields**.

Field	Description
Document Protocol Name	Select from Custom, EDI_EDIFACT, EDI_X2, HL7, OAG, PositionalFlatFile, RosettaNet, or UCCNet. (**Equals** is the only operator.)
Document Type	Select from a previously created document type, for example, 850 for EDI X12. (**Equals** is the only operator.)
Document Protocol Version	Select from a previously created document protocol version. (**Equals** is the only operator.)
Document Definition	Select from a previously created document definition. (**Equals** is the only operator.)

4. Click **Search**.

View the results, as shown in Figure 18–3.

5. In the **Details** column of the **Results** area, click the icon to see report details.

Figure 18–4 shows wire message details.

Figure 18–4   Wire Message Details
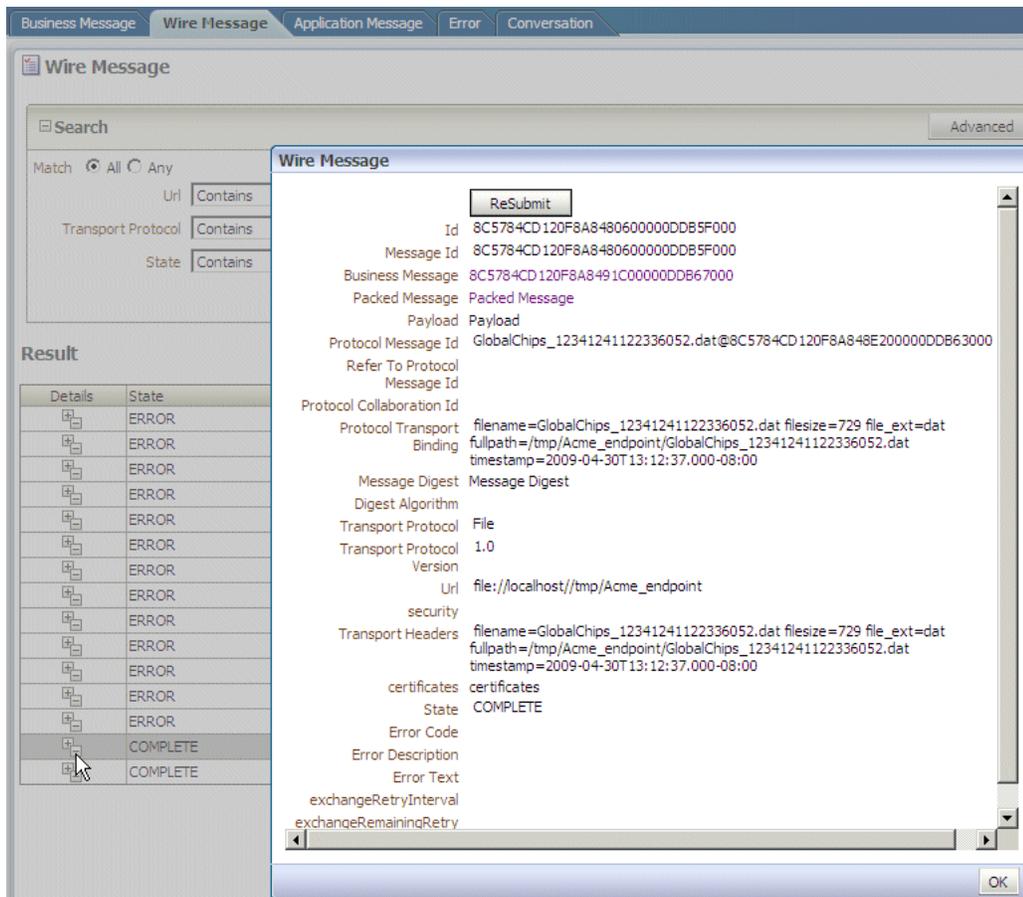
The image shows details of the wire message. The following fields are displayed: Id, Message Id, Business Message, Packaged Message, Payload, Protocol Message Id, Refer to Protocol Message Id, Protocol Collaboration Id, Protocol Transport Binding, Message Digest, Digest Algorithm, Transport Protocol, Transport Protocol Version, Url, Security, Transport Headers, Certificates, State, Error Code, Error Description, Error Text, Exchange Retry Interval, and Exchange Remaining Retry. A Resubmit button is available at the top of the screen, and the OK button is shown at the bottom right.

****

18.4 Creating Application Message Reports

This report provides information related to the SOA Composite—the name, version, and so on, if a back-end composite application sent or received the message.

Figure 18–5 shows an application message report.

Figure 18–5 Application Message Report
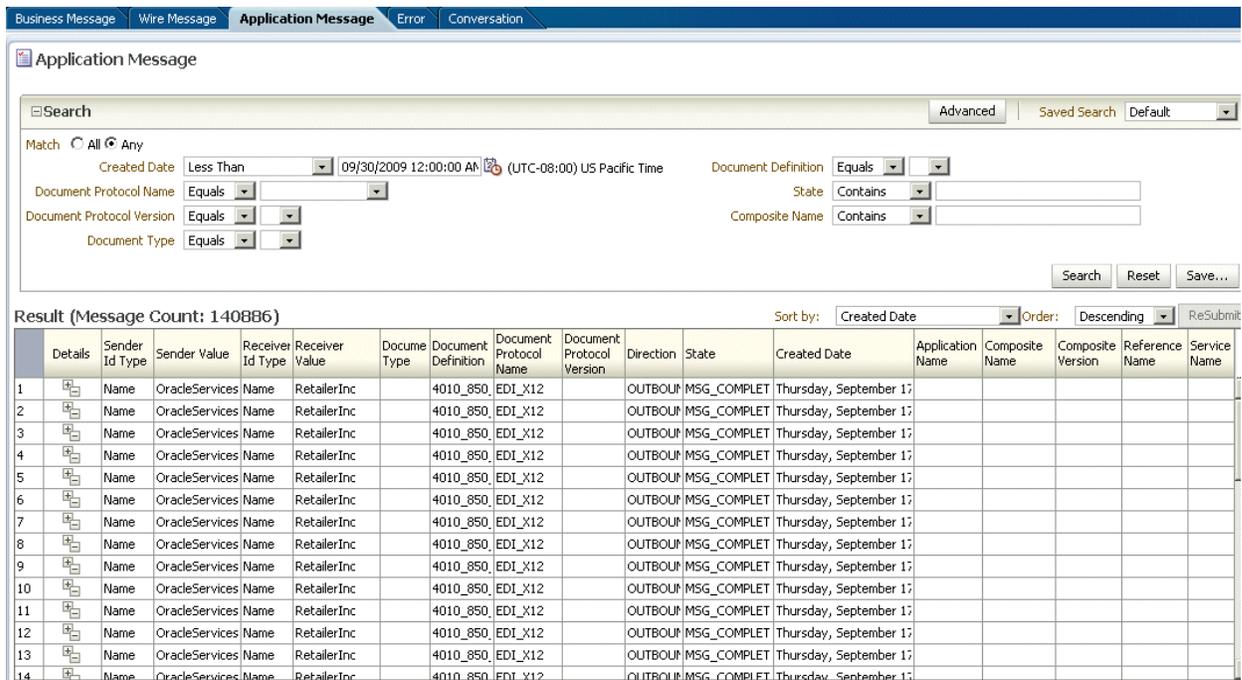
The image shows the following tabs on the Oracle B2B screen: Business Message, Wire Message, Application Message, Error, and Conversation. The Application Message tab is selected. The following Search fields are displayed: Created Date, Document Protocol Name, Document Protocol Version, Document Type, Document Definition, State, and Composite Name. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Details, Sender Id Type, Sender Value, Receiver Id Type, Receiver Value, Document Type, Document Definition, Document Protocol Name, Document Protocol Version, Direction, State, and Created Date. A Resubmit button is available at the top right of the second area.

****

To create an application message report:

1. Click **Reports**, and then **Application Message**.

2. Provide search parameters.

| Field	Description
Match	Select **All** or **Any**.
Created Date	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.
Document Protocol Name	Select from Custom, EDI_EDIFACT, EDI_X2, HL7, OAG, PositionalFlatFile, RosettaNet, or UCCNet. (**Equals** is the only operator.)
Document Protocol Version	Select from a previously created document protocol version. (**Equals** is the only operator.)

Field	Description
Document Type	Select from a previously created document type, for example, 850 for EDI X12. (**Equals** is the only operator.)
Document Definition	Select from a previously created document definition. (**Equals** is the only operator.)
State	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message state: MSG_COMPLETE MSG_ERROR MSG_WAIT_TRANSMIT MSG_WAIT_FA MSG_WAIT_BATCH
Composite Name	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the SOA composite application name.

3. To add more search fields, click **Advanced** and select from **Add Fields**.

Field	Description
Application Name	Provide the name of the application.
Composite Version	Provide the version of the SOA composite application in Oracle JDeveloper.
ECID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide an instance ID.
Sender ID Type	Provide the sender's identifier type, such as Name, DUNS, or MLLP ID.
Service Name	Provide the name of the B2B service binding component.
Receiver ID Type	Provide the receiver's identifier type, such as Name, DUNS, or MLLP ID
Receiver Value	Provide the value of the receiver's identifier type. For example, if DUNS is the Receiver ID Type, provide the DUNS number.
Sender Value	Provide the value of the sender's identifier type. For example, if Name is the Sender ID Type, provide the trading partner name as set in the identifier type in the trading partner's profile.
Reference Name	Provide the name of the B2B reference binding component.
Fabric CompositeDn	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the composite name.

4. Click **Search**.

View the results, as shown in Figure 18–5.

5. In the **Details** column of the **Results** area, click the icon to see report details.

Figure 18–6 shows application message details.

Figure 18–6   Application Message Details
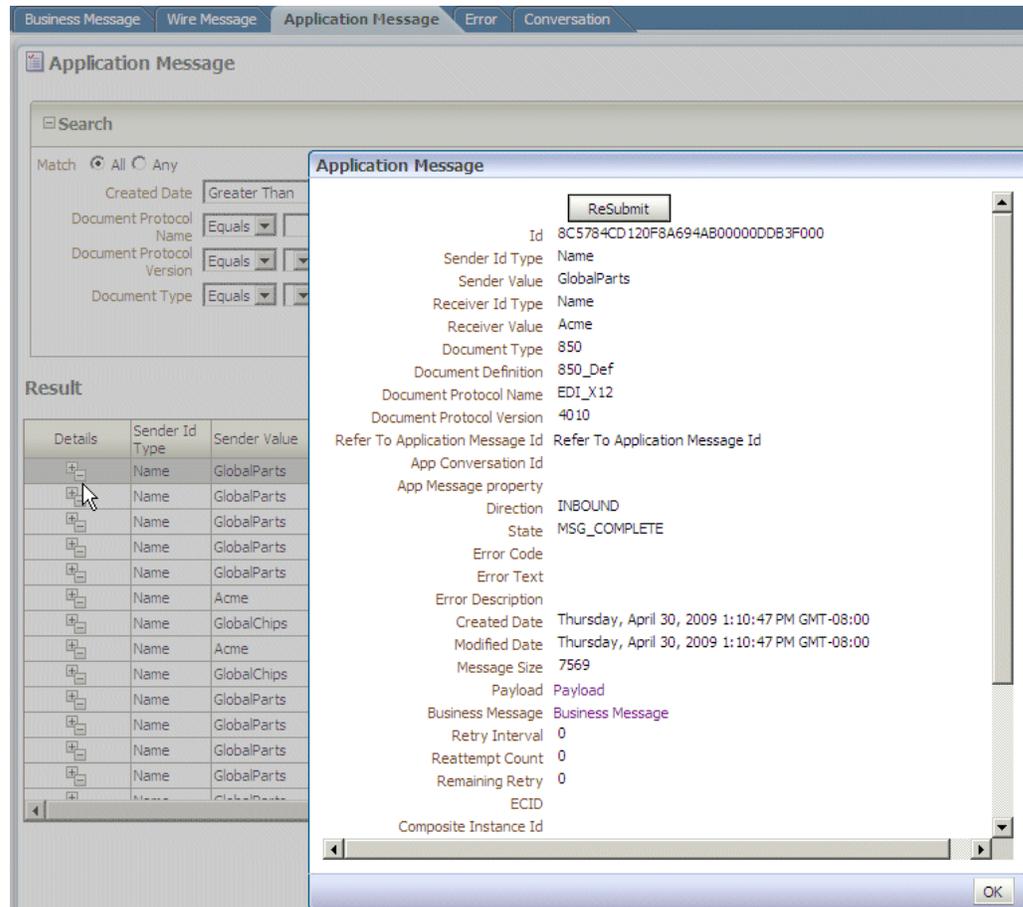
The image shows details of the application message. The following fields are displayed: Id; Sender Id Type: Name; Sender Value: GlobalParts; Receiver Id Type: Name; Receiver Value: Acme; Document Type: 850; Document Definition: 850_Def; Document Protocol Name: EDI_X12; Document Protocol Version: 4010; Refer to Application Message Id: Refer to Application Message Id; App Conversation Id; App Message Property; Direction: Inbound; State: MSG_COMPLETE; Error Code; Error Text; Error Description; Created Date: Thursday, April 30, 2009 1:10:47 PM GMT-08:00; Modified Date: Thursday, April 30, 2009 1:10:47 PM GMT-08:00; Message Size: 7569; Payload: Payload; Business Message: Business Message; Retry Interval: 0; Reattempt Count: 0; Remaining Retry: 0; ECID; and Composite Instance Id. A Resubmit button is available at the top of the screen, and the OK button is shown at the bottom right.

****

18.5 Creating Error Reports

Error status reports provide error message details. These details include the error code, error text, business message identification, message date, and message details.

Figure 18–7 shows an error report.

Figure 18–7 Error Report
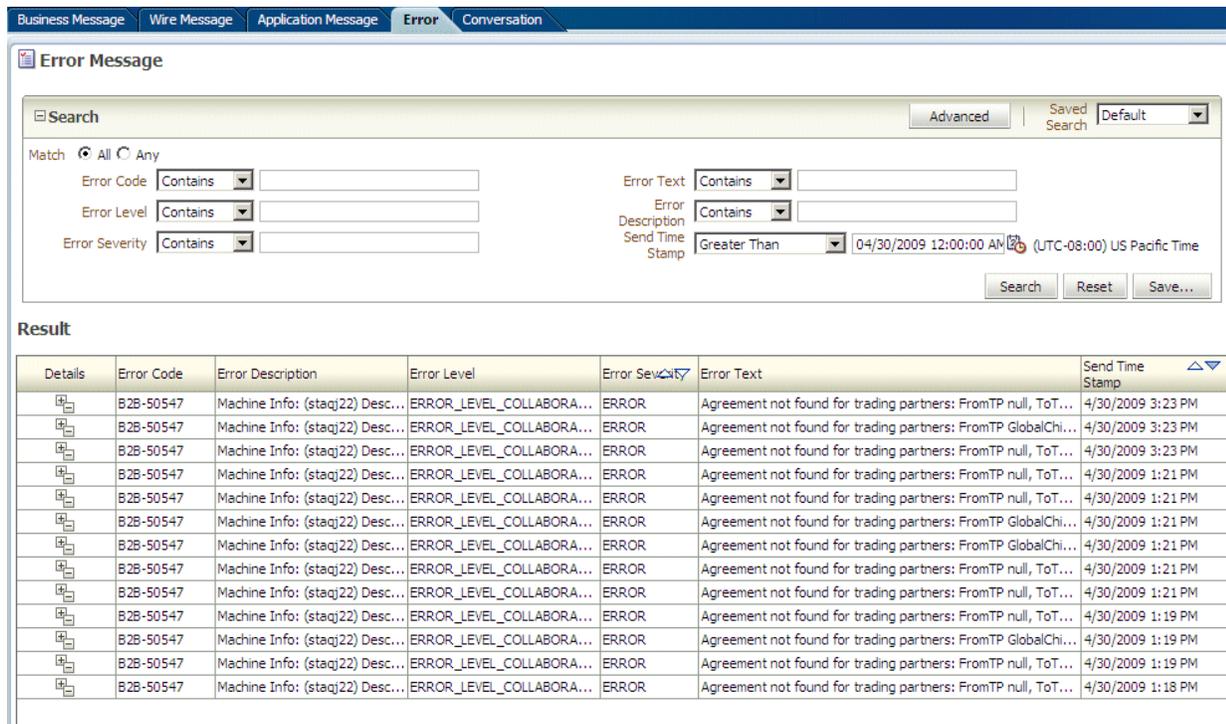
The image shows the following tabs on the Oracle B2B screen: Business Message, Wire Message, Application Message, Error, and Conversation. The Error tab is selected. The following Search fields are displayed: Error Code, Error Level, Error Severity, Error Text, Error Description, and Send Time Stamp. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Details, Error Code, Error Description, Error Level, Error Severity, Error Text, and Send Time Stamp.

****

To create an error report:

1. Click **Reports**, and then **Error**.

2. Provide search parameters.

Field	Description
Match	Select **All** or **Any**.
Error Code	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of an error code.
Error Level	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of an error level
Error Severity	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of an error severity.
Error Text	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the error text.
Error Description	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the error description.

Field	Description
Send Time Stamp	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.

3. To add more search fields, click **Advanced** and select from **Add Fields**.

| Field	Description
Document Definition	Select from a previously created document definition. (**Equals** is the only operator.)
Document Type	Select from a previously created document type, for example, 850 for EDI X12. (**Equals** is the only operator.)
Document Protocol Version	Select from a previously created document protocol version. (**Equals** is the only operator.)
Document Protocol Name	Select from Custom, EDI_EDIFACT, EDI_X2, HL7, OAG, PositionalFlatFile, RosettaNet, or UCCNet. (**Equals** is the only operator.)

4. Click **Search**.

View the results, as shown in Figure 18–7.

5. In the **Details** column of the **Results** area, click the icon to see report details.

Figure 18–8 shows error report details.

Figure 18–8   Error Reports
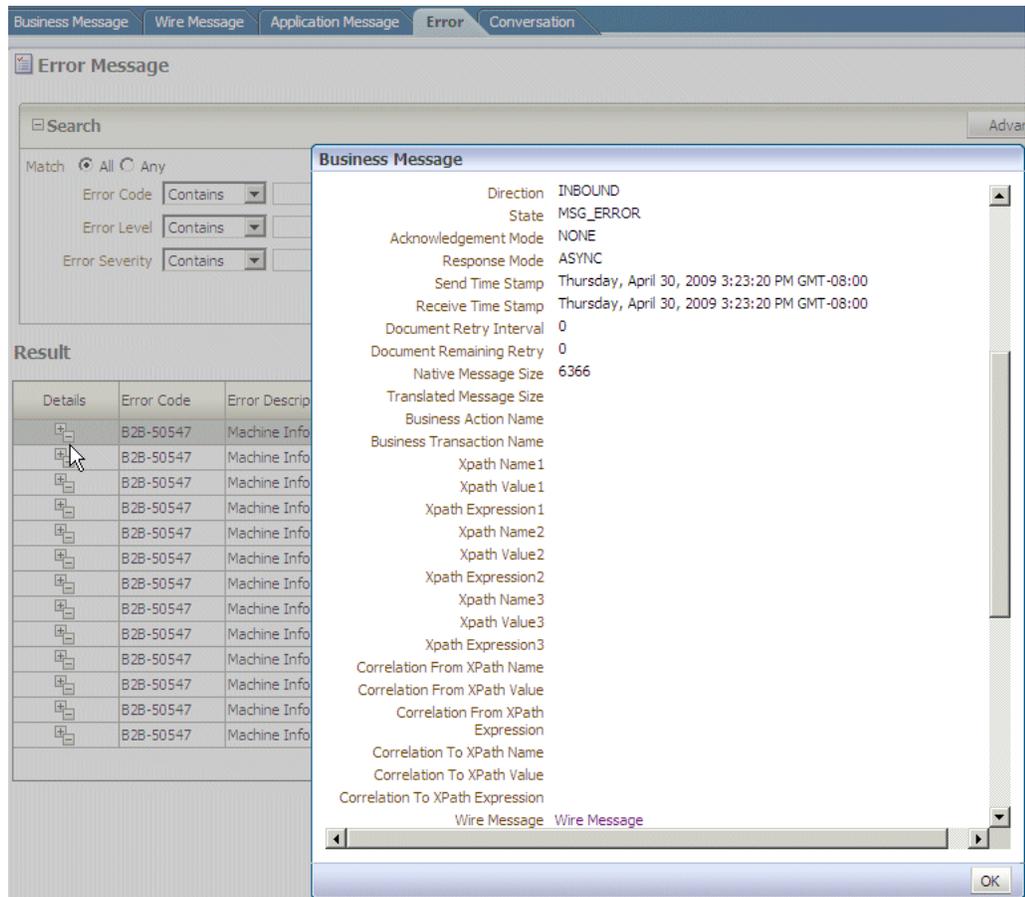
The image shows details of the business message. The following fields are displayed: Direction: Inbound; State: MSG_ERROR; Acknowledgement Mode: None; Response Mode: Async; Send Time Stamp: Thursday, April 30, 2009 3:23:20 PM GMT-08:00; Receive Time Stamp: Thursday, April 30, 2009 3:23:20 PM GMT-08:00; Document Retry Interval: 0; Document Remaining Retry: 0; Native Message Size: 6366; Translated Message Size; Business Action Name; Business Transaction Name; Xpath Name1; Xpath Value1; Xpath Expression1; Xpath Name2; Xpath Value2; Xpath Expression2; Xpath Name3; Xpath Value3; Xpath Expression3; Correlation From XPath Name; Correlation From XPath Value; Correlation From XPath Expression; Correlation to XPath Name; Correlation to XPath Value; Correlation To XPath Expression; Wire Message: Wire Message. The image shows an OK button at the bottom right.

****

18.6 Creating Conversation Reports

A conversation message results when the correlation XPath is set in a document definition to correlate messages. A correlation message also shows messages that are correlated automatically. For example, an AS2 message and its acknowledgment (MDN) are automatically correlated as part of a conversation. In RosettaNet, request and response messages are also correlated, in addition to the acknowledgments sent and received. These related messages are displayed on the **Conversation** tab.

Figure 18–9 shows a conversation report.

Figure 18–9   Conversation Report
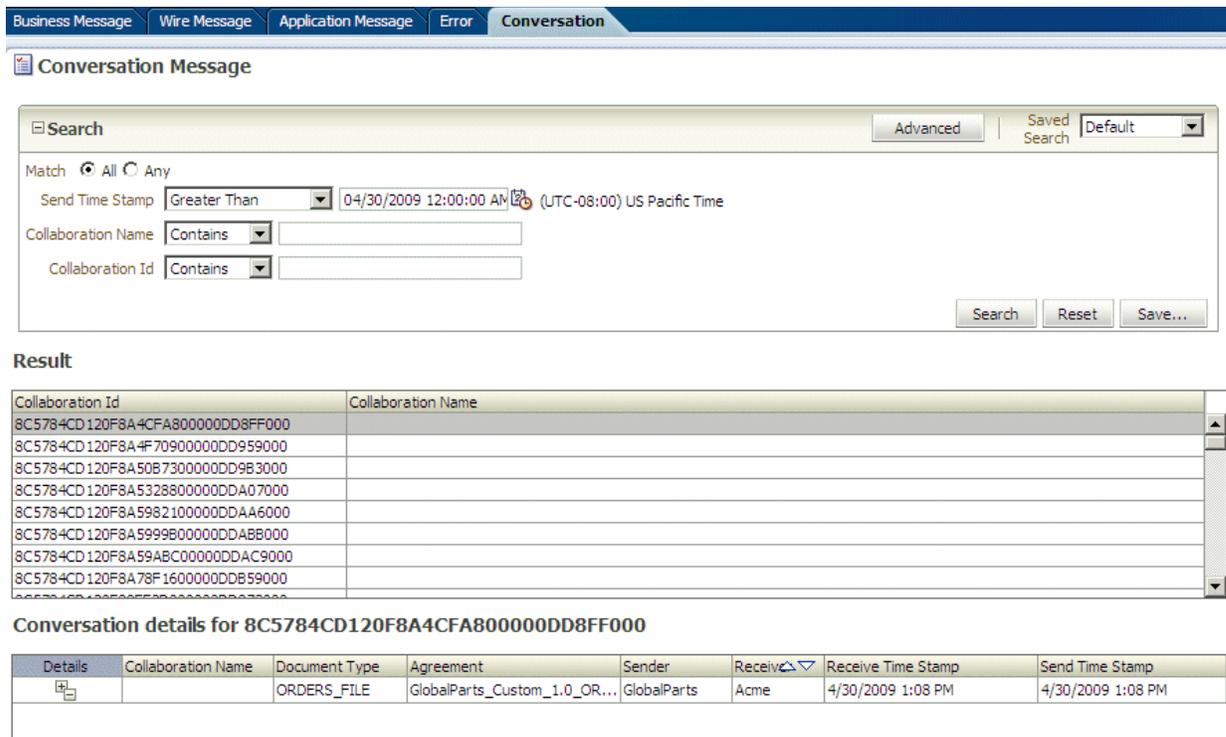
The image shows the following tabs on the Oracle B2B screen: Business Message, Wire Message, Application Message, Error, and Conversation. The Conversation tab is selected. The following Search fields are displayed: Send Time Stamp, Collaborator Name, and Collaboration Id. At the top of the area is the Match radio button with the following options: All or Any. The second area of the tab shows the search results in a table with the following columns: Collaboration Id and Collaboration Name. The third area of the tab shows the conversation details in a table with the following columns: Details, Collaboration Name, Document Type, Agreement, Sender, Receiver, Receive Time Stamp, and Send Time Stamp.

****

To create a conversation report:

1. Click **Reports**, and then **Conversation**.

2. Provide search parameters.

Field	Description
Match	Select **All** or **Any**.
Send Time Stamp	Select from **Less Than**, **Greater Than**, **Greater Than Equals**, **Equals**, or **Less Than Equals**. Provide a date and time in the format shown (MM/DD/YYYY HH:MM:SS AM/PM) or click the **Select Date and Time** icon.
Collaboration Name	Applies to ebMS and RosettaNet documents and is available from header information.
Collaboration ID	Applies to ebMS and RosettaNet documents and is available from header information.

No additional fields can be added using the **Advanced** search button.

3. Click **Search**.

   View the results, as shown in Figure 18–9.

4. In the **Details** column of the **Results** area, click the icon to see report details.

   Figure 18–10 shows conversation report details.

Figure 18–10   Conversation Report Details
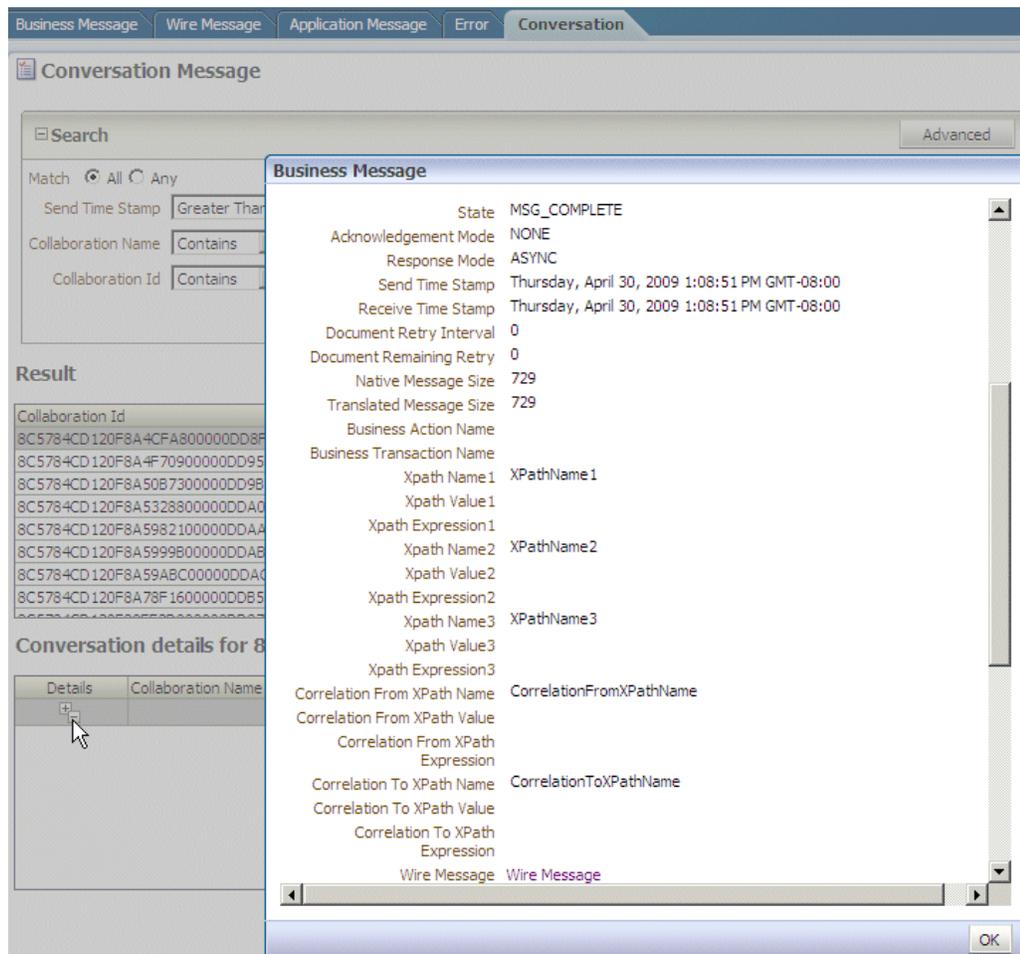
The image shows details of the business (conversation) message. The following fields are displayed: Direction: State: MSG_COMPLETE; Acknowledgement Mode: None; Response Mode: Async; Send Time Stamp: Thursday, April 30, 2009 1:38:51 PM GMT-08:00; Receive Time Stamp: Thursday, April 30, 2009 1:38:51 PM GMT-08:00; Document Retry Interval: 0; Document Remaining Retry: 0; Native Message Size: 729; Translated Message Size: 729; Business Action Name; Business Transaction Name; Xpath Name1: XPathName1; Xpath Value1; Xpath Expression1; Xpath Name2: XPathName2; Xpath Value2; Xpath Expression2; Xpath Name3: XPathName3; Xpath Value3; Xpath Expression3; Correlation From XPath Name: CorrelationFromXPathName; Correlation From XPath Value; Correlation From XPath Expression; Correlation to XPath Name: CorrelationToXPathName; Correlation to XPath Value; Correlation To XPath Expression; Wire Message: Wire Message. The image shows an OK button at the bottom right.

****

19

Using B2B Metrics

This chapter describes how Oracle B2B metrics provide system-level and partner-level status on B2B run-time data. It also discusses status on messages and errors, message counts, active document types and trading partners, and error messages.

This chapter contains the following topics:

- Section 19.1, "Introduction to B2B Metrics"
- Section 19.2, "B2B System Metrics"
- Section 19.3, "B2B Partner Metrics"

19.1 Introduction to B2B Metrics

Use the **Metrics** tab to view current run-time data in the repository. The **Metrics** tab reflects changes that occur in the run-time repository (for example, purging the run-time instance data).

Metrics data shown in the **Messages and Errors** chart and the **Message Count** chart, shown in Figure 19–1, display data for the last 10 hours or the last 20 hours.

Figure 19–1 The Messages and Errors Chart and Message Count Chart
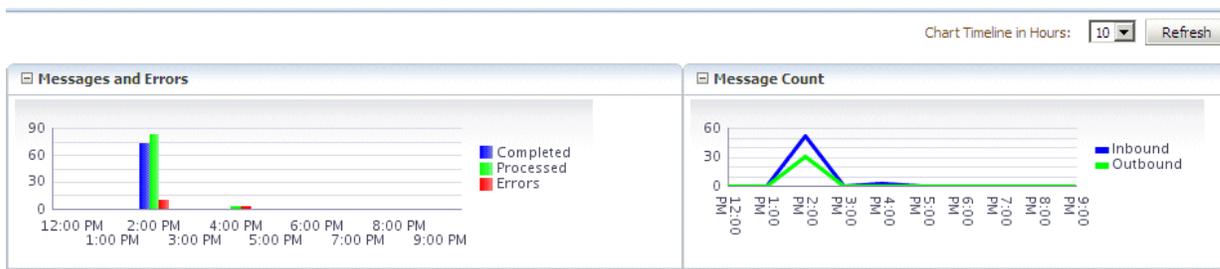
The image shows the Messages and Errors chart (on the left) and the Message Count chart (on the right) The Messages and Errors chart is a bar graph showing completed messages, processed messages, and errors The Message Count chart is a line graph showing inbound and outbound messages.

****

The metrics tables show all data from the time the first message was received. Current data is available by using the **Refresh** button. In contrast, changes are *not* immediately reflected in Oracle Enterprise Manager Fusion Middleware Control, which is based on dynamic monitoring service (DMS) metrics collected from the Weblogic managed server node. Enterprise Manager also shows limited information (the top 5 partners,

the top 5 documents) and the data is available only from the last restart of the server. See *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite* for more information.

Most fields in the active document types, active trading partners, and errors tables can be sorted in ascending or descending order, as shown in Figure 19–2.

Figure 19–2 Sorting Columns

The image shows the Active Document Types (top pane) and Errors (bottom pane). The Active Document Types pane shows a table with the following columns: Name, Number of Messages Processed (From and To), Average Processing Time in milliseconds (From and To), and Average Message Size in kilobytes (From and To). The Errors pane shows a table with the following columns: Error (Error Code and Error Text), Initiating Partner, Responding Partner, Document Type, Timestamp, and Business Message Id.

****

This is useful to identify the largest average message size or to group all the responding partner error messages, for example. You can resize columns to see any text that may be obscured. For error text, place the mouse over the text to see the entire message. The business message IDs in the **Errors** area link to business message details, as shown in Figure 19–3.

Figure 19–3   Business Message Details
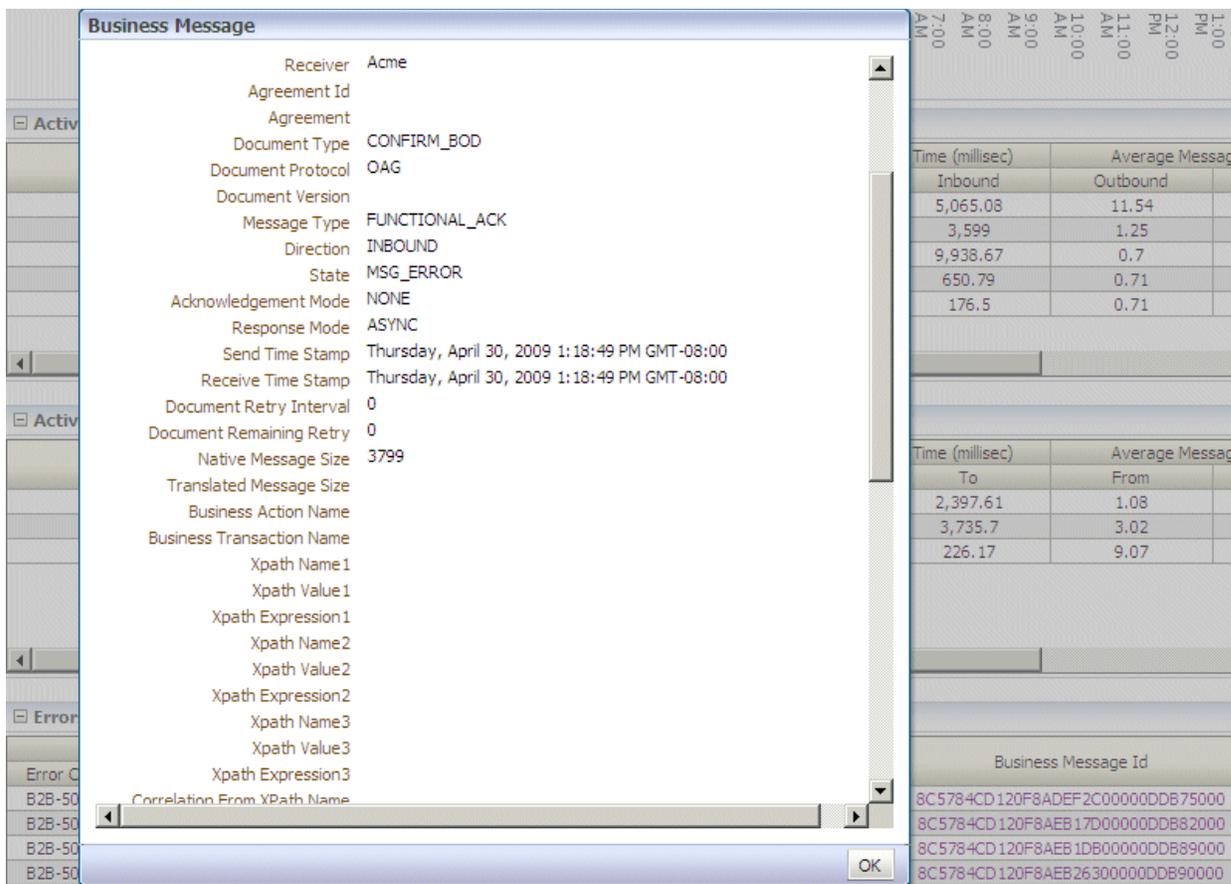
The image shows details of the business message. The following fields are displayed: Receiver: Acme; Agreement Id; Agreement; Document Type: CONFIRM_BOD; Document Protocol: OAG; Document Version; Message Type: FUNCTIONAL_ACK; Direction: Inbound; State: MSG_ERROR; Acknowledgement Mode: None; Response Mode: Async; Send Time Stamp: Thursday, April 30, 2009 1:18:49 PM GMT-08:00; Receive Time Stamp: Thursday, April 30, 2009 1:18:49 PM GMT-08:00; Document Retry Interval: 0; Document Remaining Retry: 0; Native Message Size: 3799; Translated Message Size; Business Action Name; Business Transaction Name; Xpath Name1; Xpath Value1; Xpath Expression1; Xpath Name2; Xpath Value2; Xpath Expression2; Xpath Name3; Xpath Value3; Xpath Expression3; Correlation From XPath Name; Correlation From XPath Value; and Correlation From XPath Expression. The image shows an OK button at the bottom right.

****

19.2 B2B System Metrics

Figure 19–4 shows system metrics summary data.

Figure 19–4   System Metrics
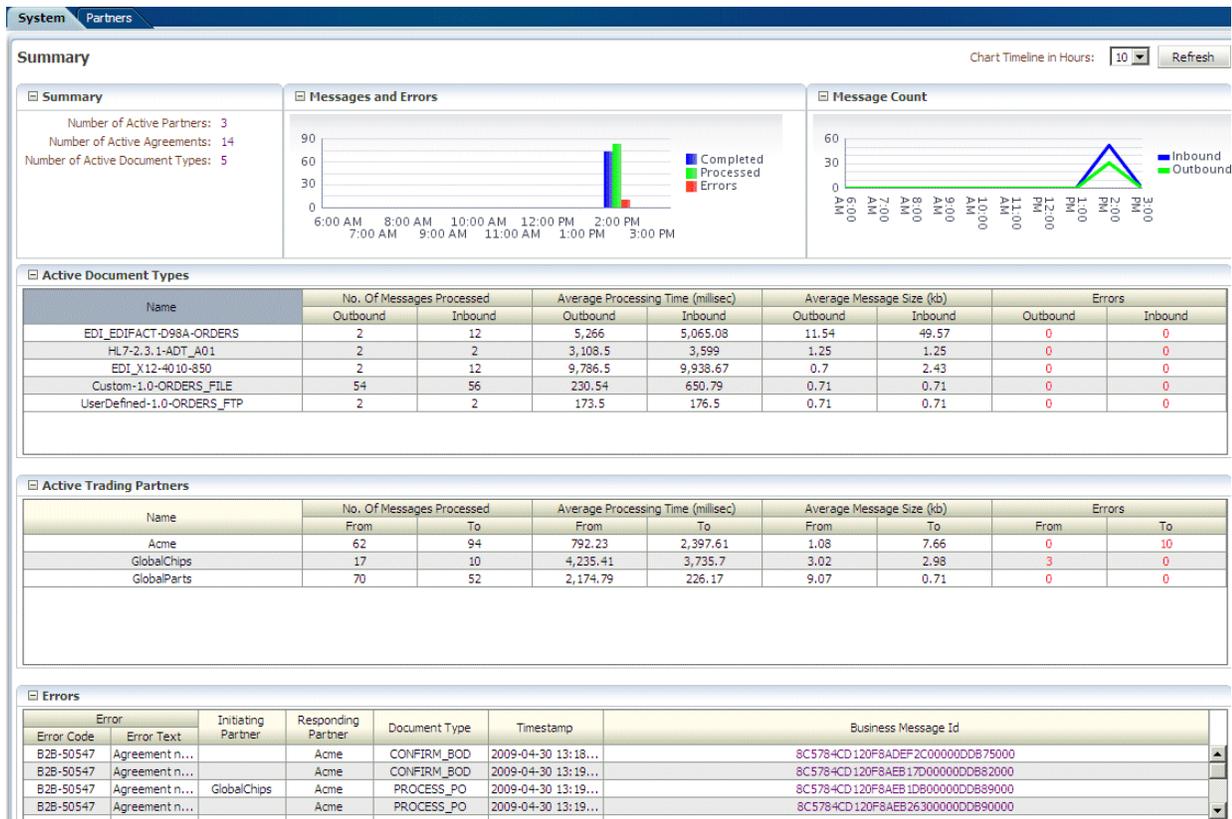
The image shows the following tabs on the Oracle B2B screen: System and Partner. The System tab is selected. The first area of the tab shows the Summary information presented in the following panes (from left to right): Summary, Messages and Errors, and Message Count. The second area of the tab shows the Active Document Types in a table with the following columns: Name, Number of Messages Processed (Outbound and Inbound), Average Processing Time in milliseconds (Outbound and Inbound), Average Message Size in kilobytes (Outbound and Inbound), and Errors (Outbound and Inbound). The third area of the tab shows the Active Trading Partners in a table with the following columns: Name, Number of Messages Processed (From and To), Average Processing Time in milliseconds (From and To), Average Message Size in kilobytes (From and To), and Errors (From and To). The fourth area of the tab shows the Errors in a table with the following columns: Error (Error Code and Error Text), Initiating Partner, Responding Partner, Document Type, Timestamp, and Business Message Id. There is a Refresh button at the top of the tab on the right side.

****

Table 19–1 describes the information on the **System** metrics tab.

Table 19–1   B2B System Metrics

Area/Field	Description
Summary	Active partners are partners for which at least one agreement has been deployed. Active agreements are agreements that have been deployed and are in the active state. Active document types are document types that have been included in deployed and active agreements.
Messages and Errors	Processed messages = Completed messages + Errored messages Details of the errored messages are listed under **Errors**.
Message Count	Active messages are shown in this trend of inbound and outbound message quantity over time.
Active Document Types	Active document types are document types that have been included in active agreements. Details of the errors are listed under **Errors**. Messages processed include completed plus errored messages, that is, active messages.
Name	Name of the document definition
No. of Messages Processed	Shows the number of document messages exchanged between the host and trading partners. **Outbound** indicates messages sent from the host to the trading partner and **Inbound** indicates messages sent from the trading partner to the host.
Average Processing Time (millisec)	Shows the average document processing time, in milliseconds, for exchanged messages. **Outbound** indicates messages sent from the host to the trading partner and **Inbound** indicates messages sent from the trading partner to the host.
Average Message Size (kb)	Shows the average document size, in kilobytes, for outbound and inbound messages.
Errors	Shows the document error count.
Active Trading Partners	Active trading partners are partners for which an agreement has been deployed and is in an active state. The host trading partner is included in the list. Messages processed include completed plus errored messages, that is, active messages.
Name	Name of the trading partner
No. of Messages Processed	Shows the number of messages sent by (**From** column) and received by (**To** column) the specified trading partner.
Average Processing Time (millisec)	Shows the average document processing time, in milliseconds, for the specified trading partner.
Average Message Size (kb)	Shows the average document size, in kilobytes, for the specified trading partner.
Errors	Shows the document error count.
Errors	Error message text is available from the Java resource bundle. The business message IDs link to business message details.

19.3 B2B Partner Metrics

Figure 19–5 shows metrics summary data for a selected trading partner.

Figure 19–5   Partner Metrics
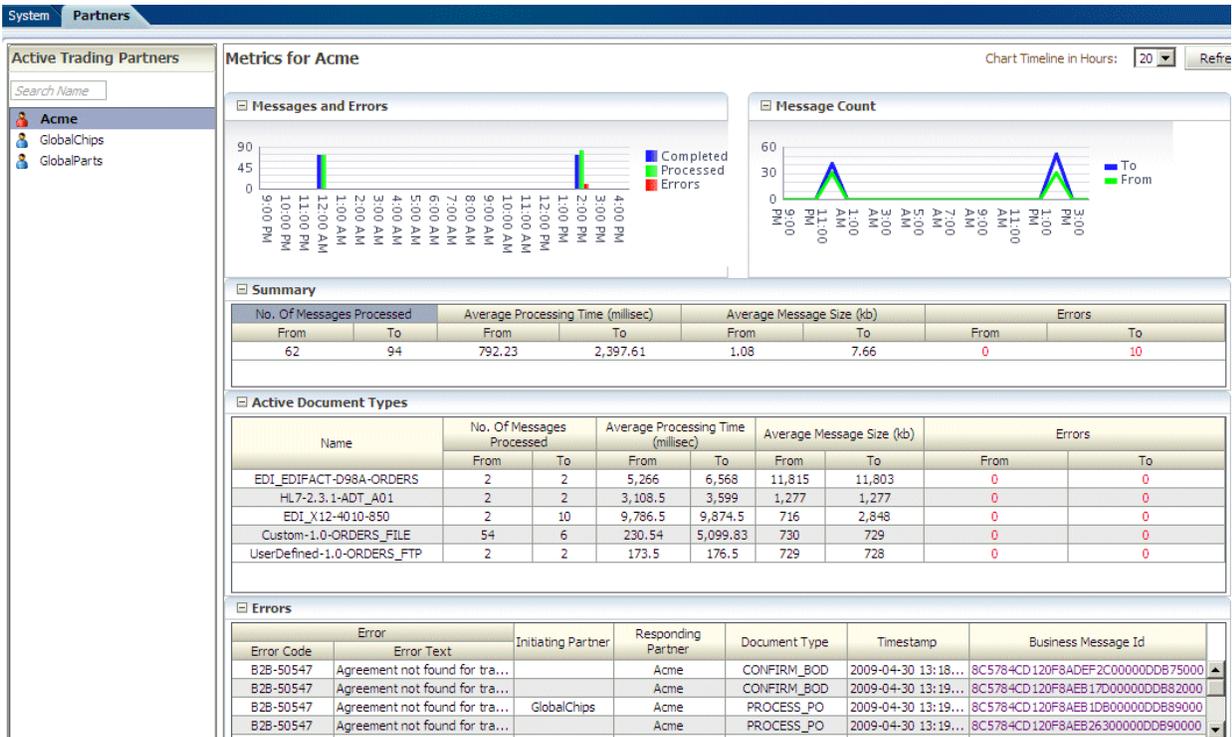
The image shows the following tabs on the Oracle B2B screen: System and Partner. The Partner tab is selected. The left pane of the tab shows a list of Active Trading Partners. The first area of the right pane shows the Summary information presented in the following panes (from left to right): Messages and Errors and Message Count. The second area of the right pane shows the Summary in a table with the following columns: No. of Messages Processed (From and To), Average Processing Time in milliseconds (From and To), Average Message Size in kilobytes (From and To), and Errors (From and To). The third area of the right pane shows the Active Document Types in a table with the following columns: Name, No. of Messages Processed (Outbound and Inbound), Average Processing Time in milliseconds (Outbound and Inbound), Average Message Size in kilobytes (Outbound and Inbound), and Errors (Outbound and Inbound). The fourth area of the tab shows the Errors in a table with the following columns: Error (Error Code and Error Text), Initiating Partner, Responding Partner, Document Type, Timestamp, and Business Message Id. There is a Refresh button at the top of the tab on the right side.

****

Table 19–2 describes the information on the **Partners** metrics tab. The data displayed is specific to the trading partner selected under **Active Trading Partners**.

Table 19–2   B2B Partner Metrics

Area/Field	Description
Messages and Errors	Processed messages = Completed messages + Errored messages Details of the errored messages are listed under **Errors**.
Message Count	Active messages are shown in this trend of inbound and outbound message quantity over time.
Summary	The number of messages processed, the average processing time, the average message size, and the number of errors are summarized for the selected trading partner.
No. of Messages Processed	Shows the number of messages sent by (**From** column) and received by (**To** column) the specified trading partner.
Average Processing Time (millisec)	Shows the average document processing time, in milliseconds, for the specified trading partner.
Average Message Size (kb)	Shows the average document size, in kilobytes, for the specified trading partner.
Errors	Shows the document error count.
Active Document Types	Active document types are document types that have been included in active agreements. Details of the errors are listed under **Errors**. Messages processed include completed plus errored messages, that is, active messages.
Name	Name of the document definition
No. of Messages Processed	Shows the number of document messages exchanged between the host and trading partners. **Outbound** indicates messages sent from the host to the trading partner and **Inbound** indicates messages sent from the trading partner to the host.
Average Processing Time (millisec)	Shows the average document processing time, in milliseconds, for exchanged messages. **Outbound** indicates messages sent from the host to the trading partner and **Inbound** indicates messages sent from the trading partner to the host.
Average Message Size (kb)	Shows the average document size, in kilobytes, for outbound and inbound messages.
Errors	Shows the document error count.
Errors	Error message text is available from the Java resource bundle. The business message IDs link to business message details.

Part V

Scripts and Utilities

This part describes how to do various tasks using scripts and utilities that are provided in Oracle B2B.

This part contains the following chapters:

- Chapter 20, "B2B Command-Line Tools"
- Chapter 21, "Using the Oracle B2B Web Services"
- Chapter 22, "Scripts for Archiving and Restoring Data"
- Chapter 23, "Utilities for Enqueuing and Dequeuing"
- Chapter 24, "Monitoring Instance Message Data With Oracle BAM"
- Chapter 25, "Programmatically Accessing Instance Message Data"

20

B2B Command-Line Tools

This chapter describes the B2B command-line tools that are available for a number of tasks such as archiving date, purging data, importing and exporting data, batching operations, deploying agreements, and so on.

This chapter contains the following topics:

20.1 Prerequisites for Running the Command-line Tools

Do the following before using the command-line tools:

1. Set the `ORACLE_HOME`, `ANT_HOME`, and `JAVA_HOME` environment variables.

   `ORACLE_HOME` is set to your Oracle Fusion Middleware installation directory. For example:

   ```
   set ORACLE_HOME=C:\oracle\wls_home
   set ANT_HOME=%ORACLE_HOME%\modules\org.apache.ant_1.7.1
   set JAVA_HOME=%ORACLE_HOME%\jdk160_18
   ```

2. Create `jndi.properties`.

   ```
   cd $ORACLE_HOME\bin
   ```

```
ant -f ant-b2b-util.xml b2bcreate-prop
```

3. Edit the `jndi.properties` file to include the `weblogic` password.

Notes:

1. Command-line tools are for administrator use only. No security or permission checks are performed to prevent the logged-in user from purging, importing, or exporting data.

2. After running any command-line tool, you should re-log into the B2B Console. The B2B Console caches some metadata and any command-line action which may have updated the metadata could lead to invalid cached data. Therefore, it is advisable to always re-login into the B2B Console after using command-line operations.

3. All of the command-line tools can be run without any JNDI credentials. To restrict the command-line tools from anonymous use, provide the following information in the `jndi.properties` file:

```
java.naming.security.principal=weblogic
java.naming.security.credentials=weblogic_password
```

Note: For any B2BCommandline utility, by default, the ANT run shows `BUILD SUCCESSFUL` and system code `0` (zero) is returned even in the case of client or server side error. In case you want the ANT run to FAIL for each client or server side error, then you need to set `exitonerror` parameter to `true`.

You can do this in either of the following two ways:

- Specify `-Dexitonerror=true` on ANT command line

  ```
  ant -f ant-b2b-util.xml b2bpurge -Dagreement=<AGR_NAME>
  -Dmode=DT -Dexitonerror=true
  ```
- Setting the `exitonerror` parameter as global setting:

  Create a properties file called ant_general.properties with the value `exitonerror=true` in the directory where ant-b2b-util.xml is present (*$ORACLE_HOME*/bin).

With this configuration, for all client or server errors, all ANT commands will fail with the message `BUILD FAILED` and the system exit code will be set to `-1`.

20.2 Archiving Data

Oracle B2B uses Oracle Data Pump as the archiving mechanism for Oracle B2B runtime instance data in Oracle database. Oracle B2B enables archiving functionality only when the Oracle B2B repository is Oracle database. You can configure start date, end date, and message state to archive and/or purge the runtime data.

In order to improve the performance and synchronize archive and purge activity, a new column (`JOB_ID`) is added in each runtime table - `B2B_BUSINESS_MESSAGE`, `B2B_EXT_BUSINESS_MESSAGE`, `B2B_APP_MESSAGE`, `B2B_WIRE_MESSAGE` and `B2B_DATA_STORAGE`.

Oracle B2B will mark the target runtime data (by start date, end date, and message state) with a unique `JOB_ID`. If you select to archive the runtime data, Oracle B2B will

invoke Data Pump PL/SQL API with `JOB_ID` to export the runtime data. Oracle B2B will purge the runtime messages by `JOB_ID` if you also want to purge them.

After the runtime data is archived/exported, Oracle B2B can also use Oracle Data Pump to import the runtime data into an Oracle B2b repository.

Before archiving data, you must set up the permissions and the archival directory.

> **Note:** The `b2b_archive` utility only supports Oracle database. No other databases are supported.

To do initial setup:

1. On the machine running the database, create a directory in which to dump the archive file. For example:

```
mkdir /tmp/archive
```

2. Grant permissions to this directory so that the database process can write to it.

```
chmod 777 /tmp/archive
```

3. Log in to the database as `sysdba`.

```
sqlplus / as sysdba
```

4. Set up the `B2B_EXPORT_DIR`.

```
SQL> create or replace DIRECTORY B2B_EXPORT_DIR as '/tmp/archive'
```

5. If your SOA schema user is `b2b_soainfra`, the user needs to be granted permission for the export.

```
SQL> grant read, write on directory B2B_EXPORT_DIR to b2b_soainfra;
SQL> grant exp_full_database  to b2b_soainfra;
```

> **Note:** Make sure that the initial setup steps detailed above are performed before attempting to archive.

To archive data:

1. Log in as the soainfra schema user.

```
$ sqlplus b2b_soainfra/password
```

2. Execute the archive procedure. For example:

```
SQL> exec B2B_ARCHIVE_INSTANCE_MSGS('2010/06/23 12:23:23','2010/06/24
12:46:24','MSG_COMPLETE',null,null,null,null,null,null,'myDump.dmp')
```

The signature of the procedure is

```
exec B2B_ARCHIVE_INSTANCE_MSGS(p_startDate,p_endDate,p_msgState,p_tpName,
p_direction,p_msgType,p_tpaName,p_idType,p_idVal,filename);
```

Table 20–1   Options for B2B_ARCHIVE_INSTANCE_MGS utility

Options	Type	Description
startDate	varchar2	Starting date for archival

Table 20–1 (Cont.) Options for B2B_ARCHIVE_INSTANCE_MGS utility

Options	Type	Description
endDate	varchar2	Ending date for archival
msgState	varchar2	State of the business message
tpName	varchar2	Trading Partner name
direction	varchar2	Message direction
msgType	varchar2	Message type
tpaName	varchar2	Trading Partner Agreement name
idType	varchar2	ID type
idVal	varchar2	ID value
filename	varchar2	Name of the archive file to be created by the database. File names MUST be unique. Verify that a file with this name does not exist in that directory.

20.3 Purging Data

> **Note:** Before purging run-time data, the configuration setup described in Section 20.2, "Archiving Data" is required. If those steps are not done, then the archive will fail with a misleading error.
>
> No security or permission checks are performed to prevent the logged-in user from purging data.
>
> The `b2bpurge` utility only supports Oracle database. No other databases are supported.

The following utility purges both design-time and run-time data and resets the environment to the installation time.

```
ant -f ant-b2b-util.xml b2bpurge
```

Table 20–2 lists the options for this utility.

Table 20–2 Options for b2bpurge utility

Option	Description	Domain	Required
mode	Specifies purging design-time or run-time data. (see Note below)	DT RT	Yes
msgState	Deletes messages with the specified message state. Used for run-time data.	MSG_COMPLETE MSG_ERROR MSG_WAIT_TRANSMIT MSG_WAIT_FA MSG_WAIT_BATCH	No. If `msgstate` is present, then `start` and `end` must be used.
purgecontrolnumber	Deletes control numbers. Used for run-time data.	true false (default)	No
fromdate	Deletes all messages, which created on or after this date.	Date format dd-mm-yyyy hh:mm AM/PM	No

Table 20–2 (Cont.) Options for b2bpurge utility

Option	Description	Domain	Required
todate	Deletes all messages, which created on or before this date.	Date format `dd-mm-yyyy hh:mm AM/PM`	No
tp	Based on trading partner.		No
direction	Direction of the message		No
msgtype	Type of the message		No
agreement	Name of the agreement		No
idtype			No
idvalue			No
archive	Should archive		Default value is `true`
archivename	File name of archived file		No

> **Note:** When only `-Dmode=RT -Dtp=trading_partner_name` options are used, this option deletes all records matching Trading Partner name with SenderName or in Receiver Name.

Example 20–1 Removes Design-Time Data

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=DT
```

Example 20–2 Purges Run-Time Data

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT
```

Example 20–3 Purges Run-Time Data, Including Control Numbers

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT -Dpurgecontrolnumber=true
```

Example 20–4 Purges Messages with the Specified State Between the Specified Dates

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT -Dfromdate="01/02/2009 12:00 AM"
-Dtodate="10/02/2009  12:00 AM" -Dmsgstate=MSG_COMPLETE
```

> **Note:** When using `archivename` the value must be a unique file name. An existing file name used with `archivename` throws an exception.

> **Note:** You can also perform document type based purge and archive of the instance data. To do this, first perform the prerequisite steps listed in Section 20.1, "Prerequisites for Running the Command-line Tools" and the initial steps listed in Section 20.2, "Archiving Data".
>
> Example:
>
> ```
> ant -f ant-b2b-util.xml b2bpurge -Dmode=RT -Ddoctype="ORDERS_FILE"
> -Darchive=true -Darchivename="docType.dmp"
> ```
> You can also execute this by using SQL.

20.4 Importing Data

> **Note:** No security or permission checks are performed to prevent the logged-in user from importing data.

The `b2bimport` utility imports the B2B metadata ZIP file to the repository. Basic validation is performed, but it is not a complete validation as with deployment validation. No data is overwritten unless you use the `overwrite` option.

The following usage imports data from `tmp/export.zip` to a location on the same server without overwriting.

```
ant -f ant-b2b-util.xml b2bimport -Dlocalfile=true -Dexportfile="/tmp/export.zip"
```

Table 20–3 lists the options for this utility.

Table 20–3 Options for b2bimport utility

Option	Description	Domain	Required
`exportfile`	Location of the export (ZIP) file	-	Yes
`overwrite`	Overwrites the existing business elements. For example, an existing delivery channel with the same trading partner name as a delivery channel in the import file is replaced if this option is set to `true`.	`true` `false` (default)	No
`localfile`	If the export file location exists on the server, then set this option to true to improve performance. The export file must be on the server on which B2B is running.	`true` `false` (default)	No

20.5 Exporting Data

The `b2bexport` utility exports metadata from the Oracle B2B repository. If no options are specified then the entire repository is exported.

> **Note:** No security or permission checks are performed to prevent the logged-in user from exporting data.

The following usage exports the entire repository (without policy details) to `/tmp/export.zip` if no other options are specified.

```
ant -f ant-b2b-util.xml b2bexport
```

Table 20–4 lists the options for this utility.

Table 20–4   Options for b2bexport utility

Option	Description	Domain	Required
tpanames	One or more agreement names to be exported. If one agreement is exported, then the ZIP file contains the folder /soa/b2b. If multiple agreements are exported, then the ZIP file contains an individual ZIP file for each of the agreements.	Agreement names must be separated by a comma	No
tpname	The trading partner name to be exported	Name of the trading partner	No
exportfile	Location of the ZIP file where the exported data is stored	/tmp/export.zip (default)	No
active	Exports agreements that have been deployed and are in active state.	true false (default)	No
policies	Set to true to export the entire repository with user and role details, which is needed for the policy store. A warning is displayed to remind you to export the policy store also. See Section 7.2, "What Is Copied When You Import or Export from the Import/Export Tab," for more information.	true false (default)	No
localfile	Set to true for improved performance *if* the export file is on the same computer as Oracle B2B.	true false (default)	No

Example 20–5   Export entire repository with policy details to /tmp/export.zip

```
ant -f ant-b2b-util.xml b2bexport -Dexportfile="/tmp/export.zip" -Dpolicies=true
```

Example 20–6   Export entire repository w/o details to /tmp/exportinserver.zip on same server

```
ant -f ant-b2b-util.xml b2bexport -Dexportfile="/tmp/exportinserver.zip"
-Dlocalfile=true
```

Example 20–7   Exports the Trading Partner Acme to /tmp/Acme.zip

```
ant -f ant-b2b-util.xml b2bexport -Dtpname="Acme" -Dexportfile="/tmp/Acme.zip"
```

Example 20–8   Exports an Agreement from Design-Time with Listening Channel Details to /tmp/acmeGc.zip

```
ant -f ant-b2b-util.xml b2bexport -Dtpanames="Acme_GC_Agreement1"
-Dexportfile="/tmp/AcmeGc.zip"
```

Listening channels are deactivated while exporting and must be reactivated after you import data.

Example 20–9   Exports Multiple Deployed and Active Agreements to /tmp/export.zip

```
ant -f ant-b2b-util.xml b2bexport -Dtpanames="Acme_GC_Agreement1,
GC_Acme_Agreement1" -Dactive=true
```

No listening channels are exported.

20.6 Batching Operations

The command line `b2bbatch` utility enables you to create batches, based on various criteria. This utility is a supplement to the available batch creation/deletion feature in the Oracle B2B Console for EDI documents. This is an ANT based command, and provides the flexibility to selectively set the criteria to create batches.

To create multiple trading partners and document versions in a single step, you must use the Oracle B2B Console; to achieve it on command-line you must run the batch creation utility multiple times. For advanced functionality, such as disabling or updating a batch you must use the Oracle B2B Console.

The usage is as follows:

```
ant -f ant-b2b-util.xml b2bbatch -Dtp=<tpName> -Dbatchtime=<batchTriggerTime>
-Dbatchname=<batchName> -Ddocument=<documentProtocolName>
-Ddocrevision=<docRevision> -Ddoctype=<docType> -Disrepetitive=<true|false>
```

Table 20–5 lists the options for this utility.

Table 20–5   Options for b2bbatch utility

Option	Description	Domain	Required
tp	Trading Partner name.	Specify the trading partner name.	Yes[1]
batchname	Batch name.	Specify a name for the batch.	Yes
batchtime	Batch trigger time.	The trigger time can be a cron String or the date in `dd/MM/yyyy HH:mm AM/PM` format. See also Example 20–13.	Yes
document	Document Protocol name.	Valid values: EDI_EDIFACT, EDI_X12	Yes
docrevision	Document revision.	-	Yes
doctype	Document type.	-	Yes
isrepetitive	To enable repetitive batching when batch created using cron string.	Valid values: true or false	No
mode	Mode	Set to `deletebatch` to delete the batch.	No

[1] To create a batch the tp, batchtime, batchname, document, docrevision, doctype options are required, to delete a batch only the mode and batchname options are required.

> **Note:** While creating a cron job for the commandline operation, you need to follow the cron syntax.

Example 20–10   Create a batch operation

This command creates a batch operation with GlobalChips for the X12/4010/850 document which is executed in a repetitive mode for the given cron String.

```
ant -f ant-b2b-util.xml b2bbatch -Dtp=GlobalChips -Dbatchtime="0 4850 11 7 5 ?
2010" -Dbatchname=batch1234 -Ddocument=EDI_X12 -Ddocrevision=4010 -Ddoctype=850
-Disrepetitive=true
```

Example 20–11   Creating batches for multiple document types

```
ant -f ant-b2b-util.xml b2bbatch -Dtp=GlobalChips -Dbatchtime="0 58 11 7 5 ? 2010"
-Dbatchname=batch1234 -Ddocument=EDI_X12 -Ddocrevision=4010 -Ddoctype=850,997

ant -f ant-b2b-util.xml b2bbatch -Dtp=GlobalChips -Dbatchtime="07/05/2010 11:45
AM" -Dbatchname=batch1234 -Ddocument=X12 -Ddocrevision=4010 -Ddoctype=850,997
```

Example 20–12   Delete a batch operation

```
ant -f ant-b2b-util.xml b2bbatch -Dmode=deletebatch -Dbatchname=batch1234
```

Example 20–13   Using special characters in batchtime option

If the value for the batchtime contains special characters such as * or # then the
character must be escaped using double quotation marks.

```
ant -f ant-b2b-util.xml b2bbatch -Dtp=GlobalChips -Dbatchtime='0
5,10,15,20,25,30,35,40,45,50,55,59 "*" "*" "*" ? 2010' -Dbatchname=batch1234
-Ddocument=EDI_X12 -Ddocrevision=4010 -Ddoctype="850,855" -Disrepetitive=true
```

20.7 Resetting Channel Passwords

The following utility sets or resets a channel password:

```
ant -f ant-b2b-util.xml resetchannelpassword
```

Table 20–6 lists the options for this utility.

Table 20–6   Options for resetchannelpassword utility

Options	Description	Required
channelname	Name of the existing channel	Yes
tp	Name of the trading partner	Yes
password	Password to be set for the specified channel. This is an optional parameter. If this parameter is not provided, the existing password (if any) of the channel is removed.	No

> **Note:**   For listening channels, the modified password takes effect
> only after the listening channel is restarted. For delivery channels, the
> modified password takes effect only after deployment.

Example 20–14   Setting the password for a listening channel

```
ant -f ant-b2b-util.xml resetchannelpassword -Dchannelname=AcmeInboundListening
-Dtp=Acme  -Dpassword=welcome1
```

The preceding example sets `welcome1` as the password for the
`AcmeInboundListening` listening channel.

Example 20–15   Resetting the password for a listening channel

```
ant -f ant-b2b-util.xml resetchannelpassword -Dchannelname=AcmeInboundListening
-Dtp=Acme
```

The preceding example resets the password for the `AcmeInboundListening` listening channel.

20.8 Activating or Deactivating Listening Channels

The following utility activates or deactivates a listening channel:

```
ant -f ant-b2b-util.xml updatechannel
```

Table 20–7 lists the options for this utility.

Table 20–7   Options for updatechannel utility

Options	Description	Value	Required
channelname	Name of the existing listening channel	-	Yes
state	State of the listening channel. If you specify `state=active`, then the listening channel goes into an active state and is restarted. If the channel is not already started, then `state=active` starts the channel. If you specify `state=inactive`, then the listening channel goes into an inactive state and is stopped.	`active` / `inactive`	Yes

Example 20–16   Starting a listening channel

```
ant -f ant-b2b-util.xml updatechannel -Dchannelname="AcmeInboundListening"
-Dstate=active
```

The preceding example starts the `AcmeInboundListening` channel.

20.9 Deploying Agreements

The `b2bdeploy` utility validates and deploys all agreements in the repository. If an agreement is already deployed, then it is deployed again. The older version of the agreement is then in an inactive state. Turning off validation is useful when deploying large numbers of agreements, where you are certain that the data is valid. It requires a SOA Server restart. Validation can be turned off by setting the property `b2b.deploy.validation` to false.

To deploy all agreements in the repository, execute:

```
ant -f ant-b2b-util.xml b2bdeploy
```

Table 20–8 lists the options for this utility.

Table 20–8   Options for b2bdeploy utility

Options	Description	Domain	Required
tpanames	One or more names of agreements to be deployed	Agreement names must be separated by a comma	No

Example 20–17   Deploys the Agreements Acme_GC_Agreement1 and GC_Acme_Agreement1

```
ant -f ant-b2b-util.xml b2bdeploy
-Dtpanames="Acme_GC_Agreement1,GC_Acme_Agreement1"
```

20.10 Validating B2B Metadata

The `b2bvalidate` utility validates Oracle B2B metadata, including agreements, trading partners, and documents. All agreements are validated if no options are specified.

```
ant -f ant-b2b-util.xml b2bvalidate [-Dargs="comma_separated_argumants"]
```

Table 20–9 lists the options for this utility.

Table 20–9   Options for b2bvalidate utility

Options	Description	Domain	Required
args	File names of the trading partner, agreement, or document protocol	File names must be separated by a comma	Yes

Example 20–18   Validates All Agreements

```
ant -f ant-b2b-util.xml b2bvalidate
```

Example 20–19   Validates Agreement tpa_ID1234.xml

```
ant -f ant-b2b-util.xml b2bvalidate -Dargs="tpa_ID1234.xml"
```

Example 20–20   Validates Trading Partner tp_MyCompany.xml and Agreement tpa_ID1234.xml

```
ant -f ant-b2b-util.xml b2bvalidate -Dargs="tp_MyCompany.xml,tpa_ID1234.xml"
```

20.11 Using the ebXML CPP/CPA Utilities

The ebXML CPP/CPA utilities convert to and from a standard ebXML CPA file in the Oracle B2B metadata.

20.11.1 Creating CPP/CPA Properties Templates

The `b2bcreate-cpaprop` utility creates a property file template that can be customized and then used to supply parameters for the `b2bcpaimport` and `b2bcpaexport` utilities.

The following usage creates a `cpp_cpa.properties` template file, which is used in the `propfile` option.

```
ant -f ant-b2b-util.xml b2bcreate-cpaprop
```

Table 20–10 lists the options for this utility.

Table 20–10   Options for b2bcreate-cpaprop utility

Option	Description	Domain	Required
propfile	Property file that stores configuration details for `b2bcpaimport` and `b2bcpaexport`	-	Yes

Example 20–21   Creates a Property File Template That Is Used in the propfile Option

```
ant -f ant-b2b-util.xml b2bcreate-cpaprop
```

20.11.1.1 Properties of cpp_cpa.properties

The following properties can be configured as part of the `cpp_cpa properties` file:

- CPA Import Properties

- CPA Export Properties

- Common Properties

20.11.1.1.1   CPA Import Properties   The CPA import properties are as follows:

`oracle.tip.b2b.ebms.BPSSDocument (Optional Property)`
This property holds the absolute path for the BPSS document, which is used to get the BPSS document details to be imported into the Oracle B2B repository. If the property does not exist, then the values are imported from the CPA document. Multiple BPSS documents are separated by `;` (semi-colon).

`oracle.tip.b2b.ebms.CPADocument (Required Property)`
This property is used to get the absolute path of the CPA document to be imported into the Oracle B2B repository.

`oracle.tip.b2b.ebms.xsdLocation (Optional Property)`
This property is used to specify the absolute path of the schema file location. This schema file is used for document validation. It is used only when a BPSS document is specified.

`oracle.tip.b2b.ebms.internalDeliveryChannel.protocol (Optional Property)`
The default internal delivery channel is an AQ queue. If you want to add a specific internal delivery channel (JMS/FTP/FILE/SFTP), then this property is used in Oracle B2B configuration. Specify all the required properties with respect to the specific transport protocol. Then use the specific channel to send messages to back-end applications.

20.11.1.1.2   CPA Export Properties   The CPA export properties are as follows:

`oracle.tip.b2b.ebms.OutputFolder (Required Property)`
This property is used to place the generated CPP/CPA files in the specified location.

`oracle.tip.b2b.ebms.Host (Required Property)`
This property is used to set the host trading partner.

`oracle.tip.b2b.ebms.HostEndPoint (Required Property)`
This property is used to set the host endpoint while generating the CPP/CPA export.

`oracle.tip.b2b.ebms.HostCertificateAlias (Optional Property)`
For secure message transfer, this property is used to get the host certificate details to the CPP/CPA export.

`oracle.tip.b2b.ebms.TPCertificateAlias (Optional Property)`

For secure message transfer, this property is used to get the trading partner certificate details to the CPP/CPA export.

```
oracle.tip.b2b.ebms.BPSSExport (Optional Property)
```
This Boolean property is used to generate the BPSS document.

20.11.1.1.3 Common Properties The common properties are as follows:

```
oracle.tip.b2b.ebms.LogDirectory (Required Property)
```
This property is used to store the log files.

```
oracle.tip.b2b.ebms.LogLevel (Required Property)
```
This property is used to specify the mode of the logs, such as DEBUG, INFO, or ERROR.

```
oracle.tip.b2b.ebms.LogType (Required Property)
```
This property is used to specify whether the log file is stored as text or XML.

20.11.2 CPP/CPA Import

The following utility converts an ebXML standard `cpa.xml` file to an Oracle B2B metadata file, which must then be imported into Oracle B2B.

```
ant -f ant-b2b-util.xml b2bcpaimport
```

Table 20–11 lists the options for this utility.

Table 20–11   Options for b2bcpaimport utility

Option	Description	Domain	Required
propfile	Property file that stores configuration details for b2bcpaimport and b2bcpaexport	-	Yes

Example 20–22   Converts CPA-Formatted XML to an Oracle B2B ZIP File

```
ant -f ant-b2b-util.xml b2bcpaimport -Dpropfile="/tmp/cpp_cpa.properties"
```

20.11.3 CPP/CPA Export

The following utility converts an Oracle B2B metadata file (data exported from Oracle B2B) to an ebXML standard `cpa.xml` file (a CPA-ready configuration).

```
ant -f ant-b2b-util.xml b2bcpaexport
```

Table 20–12 lists the options for this utility.

Table 20–12   Options for b2bcpaimport utility

Option	Description	Domain	Required
propfile	Property file that stores configuration details for b2bcpaimport and b2bcpaexport	-	Yes

Example 20–23   Converts an Oracle B2B ZIP File to a CPA-Formatted XML File

```
ant -f ant-b2b-util.xml b2bcpaexport -Dpropfile="/tmp/cpp_cpa.properties"
```

20.12 Verifying Agreement Availability

The b2bcheckcpaid utility enables you to verify the availability of an agreement for a given CPAID and trading partner. Based on the result, you can import the agreement with the option overwrite it.

Table 20–13   Options for b2bcheckcpaid utility

Option	Description	Domain	Required
args	Oracle B2B metadata file	file name	Yes

Example 20–24

```
ant -f ant-b2b-util.xml b2bcheckcpaid -Dargs="soa_file.zip"
```

20.13 Creating Oracle B2B Metadata Based on selfservice.xsd

The following utility creates the selfservice.xsd file, which is used to understand or create selfservice XML based on the XSD structure.

```
ant -f ant-b2b-util.xml b2bselfservicexsd
```

The following utility creates Oracle B2B metadata from the XML file that is created based on selfservice.xsd.

```
ant -f ant-b2b-util.xml b2bselfservice
```

See Appendix H, "Self Service Utility Protocols, Identifications, Security Specifications, and Parameters" for information about selfservice protocols, identifications, and security specifications.

Table 20–14 lists the options for these utilities.

Table 20–14   Options for b2bselfservice utility

Option	Description	Domain	Required
input	XML file absolute location	-	Yes
output	Location for Oracle B2B metadata stored as a ZIP file	-	No

Example 20–25   Converts b2bselfservicexsd-Generated XML to an Oracle B2B Metadata ZIP file (stored in /tmp/soa.zip)

```
ant -f ant-b2b-util.xml b2bselfservice -Dinput="/tmp/selfservice1.xml"
```

Example 20–26   Converts b2bselfservicexsd-Generated XML to an Oracle B2B Metadata ZIP file (stored in /tmp/as11b2b.zip)

```
ant -f ant-b2b-util.xml b2bselfservice -Dinput="/tmp/selfservice1.xml"
-Doutput="/tmp/as11b2b.zip"
```

20.13.1 Using Selfservice to Batch Create Document Protocols, Trading Partners, and Agreements

When using a multiple file approach, you must make sure that the names of the selfservice XML files are in following order, otherwise, an error claiming that the referenced object does not exist might be thrown.

1. Document Protocols selfservice XML files.

2. Trading Partner selfservice XML files (HOST must be first in this list).

3. Trading Partner Agreement selfservice XML files.

For example:

1. `doc_selfservice.xml` to hold Document Protocols.

2. `tp_selfservice.xml` to hold Trading Partner details.

3. `tpa_selfservice.xml` to hold Agreements details.

Example 20–27   Converts Several b2bselfservicexsd-Generated XML Files Within a Folder to an Oracle B2B Metadata ZIP file (stored in /tmp/as11b2b.zip)

```
ant -f ant-b2b-util.xml b2bselfservice -Dinput="/folder"
-Doutput="/tmp/as11b2b.zip"
```

20.13.2 Using Self Service Samples

Self service utility (`b2bselfservice`) XML samples are bundled along with its schemas and ECS files, which are together with the B2B metadata of B2B Samples configuration.

Self service samples are found in *$samples*/`selfservice` folder.

Samples are not available for all of the document and exchange protocol combinations. However, by replacing the document protocol details and exchange protocol details in the existing samples, you can create new XML files.

> **Note:**   Self service XML files can also be created using the XSD, which is useful for the advanced user who wants to start from scratch.

X12 Samples

The X12 samples are located in:

$samples/selfservice/x12/b2b-201-X12_4010_850_File/x12_ss.xml

The schemas are located in:

$samples/selfservice/x12/b2b-201-X12_4010_850_File/schemas

Use the following command:

```
ant -f ant-b2b-util.xml b2bselfservice
-Dinput="$samples/selfservice/x12/b2b-201-X12_4010_850_File/x12_ss.xml"
```

Custom Samples

The Custom samples are located in:

$samples/selfservice/custom/b2b-101-Custom_1.0_orders_generic_file/custom_ss.xml

The schemas are located in;

$samples/selfservice/custom/b2b-101-Custom_1.0_orders_generic_file/schemas

Use the following command:

```
ant -f ant-b2b-util.xml b2bselfservice -Dinput="$samples/selfservice/
custom\b2b-101-Custom_1.0_orders_generic_file/custom_ss.xml"
```

ebMS Samples

The ebMS samples are located in:

```
$samples/selfservice/custom/
b2b-106-Custom_1.0_orders_ebMS/buyer_setup_selfservice/ ebms_buyer_ss.xml
```

The schemas are located in:

```
$samples/selfservice/custom/
b2b-106-Custom_1.0_orders_ebMS/buyer_setup_selfservice/schemas
```

Use the following command:

```
ant -f ant-b2b-util.xml b2bselfservice -Dinput="$samples/selfservice/custom/
b2b-106-Custom_1.0_orders_ebMS/buyer_setup_selfservice/ ebms_buyer_ss.xml"
```

20.14 Resubmitting a Message

The following utility resubmits an application message or a wire message for a selected business message.

```
ant -f ant-b2b-util.xml b2bresubmit
```

> **Note:** The resubmit functionality for payload rectification will only work for Outbound Message for AppMessage only. This is not valid for inbound cases.
>
> ```
> ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG
> -Dmsgid=12345
> -Dpayloadpath=/scratch/viramamo/fmwhome/AS11gR1SOA/bin/3a4_req.xml
> ```

Table 20–15 lists the options for this utility.

Table 20–15   Options for b2bresubmit utility

Option	Description	Domain	Required
direction	The direction of the message	INBOUND OUTBOUND	Yes
msgsource	The message source	APPMSG WIREMSG	Yes
msgid	The message ID	-	Yes
doctype	Document Type		
msgstate	Message State		
fromdate	The sendTimestamp of the message	Date format to be provided within Double Quotes dd-mm-yyyy hh:mm AM/PM	
todate	The sendTimestamp of the message		
agreement	Agreement Name		

Table 20–15   (Cont.) Options for b2bresubmit utility

Option	Description	Domain	Required
payloadpath	This option is applicable for outbound application message resubmission, by providing the rectified file path.		

Example 20–28   Resubmits an Outbound Message with Message ID 12345

```
ant -f ant-b2b-util.xml b2bresubmit -Ddirection=OUTBOUND -Dmsgsource=APPMSG
-Dmsgid=12345\
```

Other Examples

```
ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Ddoctype=850

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dfromdate="29/11/2009 5:40
AM" -Dtodate="30/11/2009 7:39 AM"

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG
-Dagreement="Acme_GlobalChips_X12_4010_850_File"

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=WIREMSG -Dmsgstate=MSG_ERROR

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dfromdate="29/11/2009 5:40
AM" -Dtodate="30/11/2009 7:39 AM" -Ddirection=OUTBOUND

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dfromdate="29/11/2009 5:40
AM" -Dtodate="30/11/2009 7:39 AM" -Ddirection=INBOUND

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dmsgid=12345
-Dpayloadpath="/tmp/850.xml"
```

20.15 Resubmitting Messages Based on the Protocol Message ID

Protocol message ID can be provided in three ways.

The following examples illustrates how to resubmit messages based on the protocol message IDs.

Example 20–29   Specifying a single protocol message ID

```
ant -f ant-b2b-util.xml b2bresubmit
-Dprotocolmessageid="@0AE4A13C13049AF1C940000011B831E8"
```

Example 20–30   Specifying a comma-separated list of protocol message IDs

```
ant -f ant-b2b-util.xml b2bresubmit
-Dprotocolmessageid="@0AE4A13C13049AF1C940000011B831E8,@0AE56A6F13049AF21500000016
729990"
```

*Example 20–31   Specifying a list of protocol message IDs (one protocol message ID in
one row) in a file*

```
ant -f ant-b2b-util.xml b2bresubmit -Dsourceid=protocolmessageid
-DfilePath="inputProtMsgId.txt"
```

> **Note:** Use only one of the preceding options at a time. Using a combination of the preceding options in a single command may return unexpected results.

20.16 Scheduling Trading Partner Downtime

The following utility schedules downtime for a Trading Partner.

```
ant -f ant-b2b-util.xml b2bschedule
```

Table 20–16 lists the options for this utility.

Table 20–16   Options for b2bschedule utility

Option	Description	Domain	Required
mode	Indicates if the script will schedule or unschedule a downtime.	`schedule` or `unschedule`	Yes
schedulename	A descriptive name for the scheduled downtime	-	Yes
tp	Trading Partner Name	-	Yes (except in `unschedule` mode)
fromate	The date and time at which to begin the downtime.	Date format to be provided within Double Quotes `dd/mm/yyyy hh:mm AM/PM`	No
todate	The date and time at which to end the downtime.	Date format to be provided within Double Quotes `dd/mm/yyyy hh:mm AM/PM`	No
channelname	Channel name	-	No
extend	Extends a previously created schedule. See Example 20–37.	`true`	No

The following are examples of scheduling Trading Partner downtime using the `b2bschedule` utility. The command does not need to be in a single line.

Example 20–32   Schedule Trading Partner downtime for a specific channel and duration

```
ant -f ant-b2b-util.xml b2bschedule
-mode=schedule
-Dtp="OracleServices"
-Dfromdate="28/05/2010 06:10 AM"
-Dtodate="28/05/2010 06:15 AM"
-Dchannelname="MarketInc_AS2_DC"
-Dschedulename= "Maintenance"
```

Example 20–33   Schedule a particular channel for downtime

```
ant -f ant-b2b-util.xml b2bschedule
-mode=schedule
```

```
-Dtp="MarketInc"
-Dchannelname="MarketInc_ebMS_DC"
-Dschedulename= "Maintenance"
```

Example 20–34   Schedule the Trading Partner for downtime

In this example, all channels of this Trading Partner will be down for an unknown duration. Executing the `unschedule` command separately, as shown in Example 20–36, brings the Trading Partner back to an active state.

```
ant -f ant-b2b-util.xml b2bschedule
-mode=schedule
-Dtp="MarketInc"
-Dschedulename= "Maintenance"
```

Example 20–35   Schedule downtime for a particular duration of time

```
ant -f ant-b2b-util.xml b2bschedule
-mode=schedule
-Dtp="MarketInc"
-Dfromdate="28/05/2010 03:05 AM"
-Dtodate="28/05/2010 03:08 AM"
-Dschedulename= "Maintenance"
```

Example 20–36   Un-scheduling the scheduled event

```
ant -f ant-b2b-util.xml b2bschedule
-Dmode=unschedule
-Dschedulename="Maintenance"
```

Example 20–37   Extend an existing schedule

Create a schedule for a particular time frame for all the channels for a remote tp by entering the following:

```
ant -f ant-b2b-util.xml b2bschedule -Dtp=GlobalChips-Dfromdate="31/08/2010 10:47
AM" -Dtodate="31/08/2010 10:57 AM" -Dschedulename=Load
-Dchannelname=GlobalChips_File_Endpoint
```

Extend the schedule by creating another schedule with the same name, with the start time as the end time of the previously created schedule (31/08/2010 10:57 AM), and the end time to which you want to extend the schedule to, and then pass the parameter `-Dextend=true`:

```
ant -f ant-b2b-util.xml b2bschedule -Dtp=GlobalChips-Dfromdate="31/08/2010 10:57
AM" -Dtodate="31/08/2010 11:57 AM" -Dschedulename=Load
-Dchannelname=GlobalChips_File_Endpoint -Dextend=true
```

20.17 Managing the Keystore

Certificate expiry must be identified and notified to the Administrator as it affects the message flow with security. Oracle B2B exposes Public API/Ant commands to check the certificate for expiry. You can use `b2bmanagekeystore` to define a schedule and encapsulate these APIs for proper alert mechanism.

> **Note:** As a pre-requisite, it is required to configure the Java key store in Oracle B2B.

```
ant -f ant-b2b-util.xml b2bmanagekeystore [-Dmode=list | -Dalias="cert_name" |
-Ddays=num_of_days]
```

Table 20–17   Options for b2bmanagekeystore utility

Option	Description	Domain	Required
mode	List all the certificate alias in the keystore and its expiry dates	list	No
alias	Certificate status and its expiry date of the provided alias		No
days	List all the certificates which will expire within the days specified	integer	No

Example 20–38   Get a list of certificate aliases

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Dmode=list
```

Example 20–39   Get the status of a certificate

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Dalias="MarketInc_Cert"
```

Example 20–40   Get a list of certificates expiring at a future date

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Ddays=15
```

20.18 Updating the Keystore

The following utility updates the keystore location and password:

```
ant -f ant-b2b-util.xml b2bkeystoreupdate
```

Table 20–18 lists the mandatory parameters for this utility.

Table 20–18   Parameters for b2bkeystoreupdate utility

Option	Description
keystorelocation	List all the certificate alias in the keystore and its expiry dates
keystorepassword	Certificate status and its expiry date of the provided alias

The following is a sample usage of the utility:

```
ant -f ant-b2b-util.xml b2bkeystoreupdate -Dkeystorelocation="/tmp/acme.jks"
-Dkeystorepassword="welcome"
```

20.19 Errors During Import

If you get the following broken pipe error, use Oracle WebLogic Server Administration Console to increase Maximum Message Size to 200000000.

```
[java] Exception in thread "main" java.lang.Exception: java.rmi.UnmarshalException: Broken pipe;
nested exception is:
[java]  java.net.SocketException: Broken pipe
```

```
[java] at
oracle.tip.b2b.utility.B2BCommandLineUtility.upgradeRepository(B2BCommandLineUtility.java:548)
[java]  at oracle.tip.b2b.utility.B2BCommandLineUtility.main(B2BCommandLineUtility.java:601)
[java] Caused by: java.rmi.UnmarshalException: Broken pipe; nested exception is:
[java] java.net.SocketException: Broken pipe
```

21

Using the Oracle B2B Web Services

This chapter provides information about using the Oracle B2B Web Services such as Outbound web service and Translation Web service. It also discusses how to secure the Web services.

The chapter contains the following topics:

- Section 21.1, "Introduction to Oracle B2B Web Services"
- Section 21.2, "Using the Outbound Web Service"
- Section 21.3, "Using the Translation Web Service"
- Section 21.4, "Using the Query API"
- Section 21.5, "Securing Oracle B2B Web Services"

21.1 Introduction to Oracle B2B Web Services

Oracle B2B exposes web services to retrieve the document details in Oracle JDeveloper. These are partner-facing web services, where the user can send messages to Oracle B2B. Similarly, Oracle B2B provides a web service exposed to the back end to receive the messages, and process and send to partner, based on the agreements deployed in Oracle B2B.

To use the Oracle B2B web services, create a web service proxy in your application in Oracle JDeveloper. The Web Services Description Language (WSDL) files for the web services are available at the following URLs on the system where Oracle B2B is installed.

```
http://host_name:port_number/b2b/services/
```

A URL suffixed with just a URL pattern of Servlet entered in the Web browser, without any URI, provides the list of WSDLs available for download (with no security restrictions).

Oracle B2B provides the following web service APIs:

- **Outbound Web Service API** provides a built-in internal listening channel. See Section 21.2, "Using the Outbound Web Service" for more information.

- **Translation Web Service API** is exposed to translate the native payload, such as HL7, EDI, to XML format. See Section 21.3, "Using the Translation Web Service" for more information.

- **Query API** helps to retrieve the configured details in Oracle B2B. See Section 21.4, "Using the Query API" for more information.

Security For Oracle B2B Web Services

Oracle Web Services Manager allows integrating various types of policies without impacting the run time and flow of the web service, and it provides security of service infrastructure. Signing, encryption/decryption, authentication, authorization, auditing, and reporting will be provided by this tool kit. Hence, this web service implementation will not address the details of security, signing, and encryption. Web service methods hold the business logic instead of security details. Based on the policy enforced in Oracle Weblogic Server console, the web service client must attach security details with the web services port.

21.2 Using the Outbound Web Service

The Outbound Web Service is exposed as a built-in internal listening channel, and messages are enqueued to this service. Based on the details and headers in soap:body, agreement identification is done and the message are routed to the partner.

Table 21–1, Table 21–2, and Table 21–3 describe the Outbound Web Service request, response and fault notification message parameters.

Table 21–1   Outbound Web Service Request Parameters

Header	Data Type	Description	Required
from	String	Host name or identification value	No
to	String	Trading Partner name or identification value	Yes
@type	String Example: DUNS	Identification type	No
documentProtocolVersion	String Example: 4010	Document type version	Yes [1]
documentTypeName	String Example: 850	Document type name	
action	String Example: PurchaseOrder	ebMS action name	
service	String Example: OrderProcessing	ebMS service name	
serviceType	String Example: string Default: string	ebMS service type	
messageId	String	Message ID given in this parameter is used to create APP_Message.	No
replyToMessageId	String Example: <reply_msgID>:collaborationID	Holds the message ID of which message this reply goes to, along with the collaboration ID.	No
messageType	String	Type of the message	No
encoding	String Example: ISO-8859-1 Default: UTF-8	Encoding format	No
payload	Xsd:anyType	Holds the payload	Yes
attachment	Xsd:anyType	Attachment, if any	No

[1] Either documentProtocolVersion and documentTypeName must be present, or action, service, and serviceType must be present. A custom generic case only requires action and not the others.

Table 21–2   Outbound Web Service Response Parameters

Header	Data Type	Description	Required
isTransmitted	Boolean	If `true`, the message was successfully transmitted; otherwise `false`.	Yes

Table 21–3   Outbound Web Service Fault Message Parameters

Header	Data Type	Description
ExceptionMessage	String	If a fault is found, the Exception Stack Trace is transmitted.

21.3 Using the Translation Web Service

The Translation Web Service translates the XML payload, such as HL7, EDI, to native format as well as translates the native documents to XML.

> **Note:**   If a non-EDI XML is sent, then the translated content contains the same payload as it is, provided the agreement matches in Oracle B2B server.
>
> If the EDI XML is sent for translation, and the document is involved as part of a batch, then the translated response will not translate the payload.

> **Note:**   Translation Web service requests are not differentiated from other outbound requests in reporting and metrics.

> **Note:**   The `TranslateService` Web service has been deprecated.

21.3.1 XMLToNativeService

This service translates a given XML payload to the native document. Translation is possible in two levels, complete envelope or just the transaction set alone.

21.3.1.1 Complete Envelope

This translation produces a complete native document.

Prerequisites:

- Valid deployed outbound agreement
- Trading Partner name, document type name, document protocol version and payload are mandatory for the service invocation

Supported protocols:

- EDI X12
- EDI EDIFACT
- HL7

21.3.1.2 Transaction Set Alone

This translation produces a transaction set only in the resultant native content.

Prerequisites:

- The service expects the document type and revision to be configured in Oracle B2B.

- The `transactionSetOnly` element should be set to `true` and the Trading Partner name should not be part of service invocation request.

- The elements, `transactionSetOnly`, `documentTypeName`, `documentProtocolVersion`, and `payload` are mandatory for the service invocation.

Supported protocol:

- EDI X12

21.3.2 NativeToXMLService

This service translates a given native document to XML. It can translate either a complete native document or just the transaction set alone.

21.3.2.1 Complete Envelope

The service translates a complete native document to XML.

Prerequisites:

- Valid deployed inbound agreement

- Trading Partner name and payload are mandatory for the service invocation

Supported protocols:

- EDI X12

- EDI EDIFACT

- HL7

21.3.2.2 Transaction Set Alone

This service produces the XML content for a given transaction set.

The prerequisites and supported protocol are the same as that of "XMLToNativeService".

21.4 Using the Query API

The Query API retrieves the configured details from Oracle B2B, and share them with applications.

Before initiating a message transmission from applications, a health check request is made for the given parameters. This check finds if any configurations exist, and how many are active. If no configuration is found, the application can stop message flow in its layer with the message "no configuration found in B2B."

The following APIs are provided:

- `Is Trading Partner Agreement Setup` returns true if agreement found for the given inputs. See Section 21.4.1, "Is Trading Partner Agreement Setup Parameters" for parameter details.

- `Get Trading Partner Agreement Information` returns agreement details. See Section 21.4.2, "Get Trading Partner Agreement Information Parameters" for parameter details.

21.4.1 Is Trading Partner Agreement Setup Parameters

Table 21–4   Is Trading Partner Agreement Setup Request Parameters

Header	Data Type	Description	Required
from	String	Host name or identification value	No
to	String	Trading Partner name or identification value	No
@type	String Example: DUNS	Identification type	No
document	String Example: Sales Order	Name of the internal application document or AIA EBO to be sent out	Yes
action	String Example: Update	A sub-classification which identifies the specific interaction with the Trading Partner	Yes

Table 21–5   Is Trading Partner Agreement Setup Response Parameters

Header	Data Type	Description	Required
MatchedTPACount	integer	Number of Agreements present in an active state in the Oracle 2B Repository	Yes

Table 21–6   Is Trading Partner Agreement Setup Fault Message Parameters

Header	Data Type	Description
ExceptionMessage	String	If a fault is found, the Exception Stack Trace is transmitted.

21.4.2 Get Trading Partner Agreement Information Parameters

Table 21–7   Get Trading Partner Agreement Information Request Parameters

Header	Data Type	Description	Required
from	String	Host name or identification value	No
to	String	Trading Partner name or identification value	No
@type	String Example: DUNS	Identification type	No
document	String Example: Sales Order	Name of the internal application document or AIA EBO to be sent out	No
action	String Example: Update	A sub-classification which identifies the specific interaction with the Trading Partner	No

Table 21–8   Get Trading Partner Agreement Information Response Parameters

Header	Data Type	Description	Required
AgreementID	String	Unique Agreement ID of the matching agreement	Yes
B2BDocumentDef	String	Document definition in Oracle B2B used for creating the Oracle B2B document type	Yes
B2BDocumentType	String	Document type defined in Oracle B2B for the requested application document and action	Yes
B2BDocumentRevision	String	Document revision defined in Oracle B2B for the requested application document and action	Yes
B2BDocumentProtocol	String	DocumentProtocol name defined in Oracle B2B for the requested application document and action	Yes
Direction	String	Document direction	Yes
XSLTFile	String	XSLT file to be used by the AIA layer to generate the Oracle B2B TP document	No

Table 21–9   Get Trading Partner Agreement Information Fault Message Parameters

Header	Data Type	Description
ExceptionMessage	String	If a fault is found, the Exception Stack Trace is transmitted.

21.5 Securing Oracle B2B Web Services

Web services exposed in B2B must be secured to hide the configuration details from intruders. The Oracle Web Services Manager policy approach provides the facility to secure the web services based on your requirements.

Web services endpoints are registered dynamically and programmatically using the `oracle.webservices.provider.ProviderConfig.addService(...)` API. Because these endpoints are not displayed in Oracle Enterprise Manager Fusion Middleware Control Console, Oracle B2B maintains the lifecycle of Web services and their policies.

To specify a policy and attach it to the web services endpoints:

1. In Oracle B2B Console, go to **Administration>Configuration** tab.

2. Enter the appropriate value in the **Webservice Policy** field in the **Non Purgeable** section.

    Enter either the Oracle Web Services Manager policy URI to secure just the endpoints, or enter the whole `<policy>` tag to uptake RM, Addressing, and Logging. See the examples below.

    See "What Oracle WSM Security Policies Are Available?" in *Oracle Fusion Middleware Securing WebLogic Web Services for Oracle WebLogic Server* for a listing of available Oracle WSM policies.

3. Click **Save**.

    Based on the policy attached here, WSDL URL will start publishing/describing the policy details which need to be used while creating the proxy client for this service.

    No restart of B2B is required to uptake the policy changes.

Examples

Example 1: Enter the following URI in **Webservice Policy** to apply security policy `oracle/wss_username_token_service_policy`.

```
oracle/wss_username_token_service_policy
```

Example 2: Enter the following XML in **Webservice Policy**, which applies Security and RM policy.

```
<policy><policy-references><policy-reference uri="oracle/wss_username_token_
service_policy" category="security"/><policy-reference uri="oracle/wsrm11_policy"
category="wsrm"/></policy-references></policy>
```

Limitations

There is no way to control the policy only for the particular endpoint. Whatever the policy specified, it is applicable for all the endpoints.

Removing already specified policy URI by clearing the **Webservice Policy** field does not work. You must enter some string in the field, such as "none".

No metrics are displayed for Oracle B2B Web service usage in Oracle Enterprise Manager Fusion Middleware Control Console.

22

Scripts for Archiving and Restoring Data

This chapter describes how to archive and restore B2B business messages using SQL scripts.

This chapter contains the following topics:

- Section 22.1, "Introduction to Archiving and Restoring B2B Business Messages"
- Section 22.2, "Archiving B2B Business Messages"
- Section 22.3, "Restoring B2B Business Messages"

See Chapter 7, "Importing and Exporting Data," for information on importing and exporting design-time data.

22.1 Introduction to Archiving and Restoring B2B Business Messages

Oracle B2B uses Oracle Data Pump, an Oracle Database 11*g* feature that enables fast bulk data and metadata movement, to archive B2B run-time instance data *in Oracle databases*.[1]

You can specify criteria for archiving (and optionally purging) business messages based on start date, end date, and message state. The targeted business messages are marked with `JOB_ID`, a column in the B2B run-time tables that is used to synchronize archive and purge activity. B2B invokes the Data Pump PL/SQL API using `JOB_ID`. Hence, when you archive business messages, all the associated tables are also archived. Archived business messages can also be restored by using the Data Pump to import the run-time data into Oracle B2B (Oracle Metadata Service repository) and accessing it through B2B reports.

22.2 Archiving B2B Business Messages

To archive business messages, set up the archive directory and permissions and then run the archive procedure. The procedure provides an option to purge the archived rows.

To set up the archive directory and permissions:

1. On the computer running the database, create a directory for the archive file. For example,

   ```
   mkdir /tmp/archive
   ```

---

[1] For non-Oracle databases, external database archiving tools can be used to export and import run-time data.

2. Give permissions to this directory so that the database process can write to it. For example,

```
chmod 777 /tmp/archive
```

3. Log in to the database as sysdba.

```
sqlplus /as sysdba
```

4. Set up B2B_EXPORT_DIR.

```
SQL> create or replace directory B2B_EXPORT_DIR as '/tmp/import'
```

5. Grant the SOA schema user (for example, b2b_soainfra) permission for the export.

```
SQL> grant read, write on directory B2B_EXPORT_DIR to b2b_soainfra;
SQL> grant exp_full_database to b2b_soainfra;
```

To archive, with an option to purge:

Set up the archive directory and permissions before using the following PL/SQL API.

1. Log in as the SOA schema user.

```
$ sqlplus b2b_soainfra/password
```

2. Execute the archive procedure, for example,

```
SQL> exec
b2b_archive_procedure('21-JAN-2008','28-JAN-2008','MSG_COMPLETE','JAN.dmp','N')
;
```

The signature of the procedure is

```
b2b_archive_procedure(fromDate, toDate, messageState, fileName, should_purge);
```

Table 22–1 lists the parameters for the b2b_archive_procedure API.

Table 22–1   b2b_archive_procedure Parameters

Parameter	Example	Description
fromDate	21-JAN-2008	Starting date for archival, DD-MON-YYYY
toDate	28-JAN-2008	Ending date for archival, DD-MON-YYYY
messageState	MSG_COMPLETE	State of the business message. The MSG_COMPLETE state is typically archived. Other possible states are MSG_INVALID, MSG_CONTINUE_PROCESS, MSG_COLLAB_WAIT, MSG_PROCESS_ACK, MSG_SEND_ACK, MSG_WAIT_ACK, MSG_ERROR, MSG_WAIT_TRANSMIT, MSG_SEND_EXP, MSG_PROCESS_EXP, MSG_ABORTED, MSG_TRANSMITFAILED, MSG_WAIT_FA, MSG_SEND_FA, MSG_WAIT_BATCH
fileName	JAN.dmp	Name of the archive file to be created by the database. Ensure that a file with this name does not exist in the archive directory.
should_purge	N	Y removes the archived rows. The default is N.

22.3 Restoring B2B Business Messages

To restore business messages, set up the import directory and permissions and then run the restore procedure.

To set up the import directory and permissions:

1. On the PC running the database, create a directory for the import file.

    ```
    mkdir /tmp/import
    ```

2. Give permissions to this directory so that the database process can read from it.

    ```
    chmod 777 /tmp/import
    ```

3. Log in to the database as `sysdba`.

    ```
    sqlplus /as sysdba
    ```

4. Set up `B2B_IMPORT_DIR`.

    ```
    SQL> create or replace B2B_IMPORT_DIR as '/tmp/import'
    ```

5. Grant the SOA schema user (`b2b_soainfra`) permission for the import.

    ```
    SQL> grant read, write on directory B2B_IMPORT_DIR to b2b_soainfra;
    SQL> grant imp_full_database to b2b_soainfra;
    ```

To restore business messages:

Set up the import directory and permissions before using the following PL/SQL API.

1. Log in as the SOA infra schema user.

    ```
    $ sqlplus soa_infra_user/password
    ```

2. Execute the import procedure, for example

    ```
    SQL> exec b2b_restore_procedure('JAN.dmp');
    ```

    The signature of the procedure is

    ```
    b2b_restore_procedure(fileName)
    ```

    Use the **Reports** tab to search for and display the imported data.

23

Utilities for Enqueuing and Dequeuing

This chapter describes the utilities provided by Oracle B2B to send and receive business messages to and from Oracle B2B through the default AQ queue interface or the JMS queue interface.

Oracle B2B provides utilities to test and verify your installation and configuration before connecting to the host (back-end) applications. These utilities can be used to send and receive business messages to and from Oracle B2B through the default AQ queue interface or the JMS queue interface. Other AQ internal delivery channels can be handled in the same way. See the B2B samples for examples of how to implement these utilities (See Section 1.7, "Oracle B2B Samples and Cookbooks," for information about the samples.)

This chapter contains the following topics:

- Section 23.1, "AQ Enqueue and Dequeue Utilities"
- Section 23.2, "JMS Enqueue and Dequeue Utilities"
- Section 23.3, "Using the attachmentsDescriptor.xsd"

23.1 AQ Enqueue and Dequeue Utilities

You can enqueue to and dequeue from an AQ queue using Java. `IPEnqueue` and `IPDequeue` must be executed in the Oracle B2B environment.

23.1.1 AQ Enqueue

Table 23–1 lists the Java AQ enqueue utility (`oracle.tip.b2b.data.IPEnqueue`) properties.

Table 23–1   IPEnqueue Properties

Name	Description
queue	The outbound AQ queue name. If unspecified, the Java enqueue utility uses the default outbound queue `IP_OUT_QUEUE`.
replyToMsgID	The message ID to which the sending message is replying, typically used for the response message type.
from	Trading partner that sends the message
to	Trading partner that receives the message
doctypeName	Document type name for the message
doctypeRevision	Document protocol revision for the message
payload	Payload file name

Table 23–1 (Cont.) IPEnqueue Properties

Name	Description
attachment_name	Attachment file name. Use this property to assign a name to the attachment file that is something other than the e-mail subject name.
	For a custom outbound message over e-mail with AQ do the following:
	`actionName=ATTACHMENT_NAME:Sample.txt`
url	The database URL format is `jdbc:oracle:thin:@host:port:sid`
user	The database user
password	The database password
eventName	Action name
msgID	Message ID (optional). B2B generates its own message ID if it is not provided as part of an enqueue.
msgType	Provide an optional message type: ■ Request = 1 (default) ■ Response = 2 ■ Functional Ack = 9
dateFormat	Used to convert the date format used in e-mail delivery channel
dynamicemail	Provides the `to` party e-mail address as part of `actionName`. For example: `actionName=DYNAMICEMAIL:email_id`
dynamic_from_email	Provides the `from` party e-mail address as part of `actionName`. For example: `actionName=DYNAMIC_FROM_EMAIL:email_id`

Example: `ipenqueue.properties`

```
queue           =
url             = jdbc:oracle:thin:@host:1521:sid
user             = user1
password         = password
replyToMsgID    =
from            = Acme
to              = GlobalChips
doctypeName     = 850
doctypeRevision = 4010
payload         = Acme_850.xml
attachment      =
```

> **Note:** In Windows ja_JP locale instances, the VARCHAR/String values are not enqueued correctly to the queue. The INT and CLOB values are enqueued correctly. This causes some fields, such as the `from` and `to` fields, to be null when the IPEnqueue utility is used to enqueue a file. As a workaround, in ja_JP locales, orai18n.jar should be added to the classpath while using oracle.tip.b2b.data.IPEnqueue.

23.1.2 AQ Dequeue

To dequeue messages, use the `IPDequeue` utility.

Table 23–2 lists the Java AQ dequeue utility (`oracle.tip.b2b.data.IPDequeue`) properties.

Table 23–2   IPDequeue Properties

Name	Description
queue	The inbound AQ queue name. If unspecified, the Java dequeue utility uses the default inbound queue `IP_IN_QUEUE`.
count	The number of messages to dequeue. If unspecified, only one message is dequeued.
output	Output file name
url	The database URL format is `jdbc:oracle:thin:@host:port:sid`
user	The database user
password	The database password

Example: `ipdequeue.properties`:

```
queue           =
count           = 1
output          = t1.trc
url             = jdbc:oracle:thin:@host:1521:sid
user            = user1
password        = password
```

23.2 JMS Enqueue and Dequeue Utilities

You can enqueue to and dequeue from a JMS destination (queue or topic) using utilities. If a user name and password are not provided, the local JNDI is used, including in a clustered environment, provided that the destinations are distributed. Oracle B2B does not support javax.jms.ObjectMessage.

23.2.1 JMS Enqueue

Use the JMS enqueue utility, `oracle.tip.b2b.data.JMSEnqueue`, to send a message to a JMS destination (queue or topic). This utility expects a property file to be provided as a command-line argument where it reads the details to be sent.

Table 23–3 lists the properties that can be configured in the file.

Table 23–3   JMS Enqueue Properties

Name	Description
destination	JNDI name of queue or topic to send message to
cf	JNDI name of connection factory to use
factory	Factory provider class
isTopic	Indicator for topic (optional)
url	The JNDI URL format is `url=t3://host_name:port_number/`
user	The application server administrator userID.
password	The application server administrator password
from	From party

Table 23–3   (Cont.) JMS Enqueue Properties

Name	Description
to	To party
eventName	Action name
doctypeName	Document type name
doctypeRevision	Document type revision
payload	Payload file path
attachment	Attachment file path
msgID	Message ID (optional). B2B generates its own message ID if it is not provided as part of an enqueue.
replyToMsgID	Reply to message (optional)
msgType	Message type; the default is Request (optional).
attachment_name	Attachment file name. Use this property to assign a name to the attachment file that is something other than the e-mail subject name. For a custom outbound message over e-mail with JMS do the following: `constant : Sample.xml property : jca.jms.JMSProperty.ATTACHMENT_NAME`
dateFormat	Used to convert the date format used in e-mail delivery channel
dynamicemail	Provides the to party e-mail address. For example: `constant : email_id property : jca.jms.JMSProperty.DYNAMICEMAIL`
dynamic_from_email	Provides the from party e-mail address. For example: `constant : email_id property : jca.jms.JMSProperty.DYNAMIC_FROM_EMAIL`

Example 23–1 shows the sample `jms_enqueue.properties` file.

Example 23–1   Sample jms_enqueue.properties File

```
####### Destination Details       #######
destination = jms/b2b/B2B_IN_QUEUE
cf = jms/b2b/B2BQueueConnectionFactory

####### Server and Factory Details #######
factory=weblogic.jndi.WLInitialContextFactory
url=t3://host_name:port_number/
#user=<uncomment and provide you username>
#password=<uncomment and proivde you password if required>

####### Payload Details #######
from=Acme
to=GlobalChips
#eventName=SampleEvent
doctypeName=Custom
doctypeRevision=1.0
payload=/scratch/work/GlobalChips_1234.dat
```

See the sample documentation for how to run these utilities.

23.2.2 Enqueue—Using a JMS JCA Adapter or Custom Utilities

The properties used by the AQ and JMS utilities are translated internally before the message is sent to the destination. Ensure that the properties in Table 23–4 are set as part of the javax.jms.Message delivered to the destination that B2B listens on.

Table 23–4 How AQ/JMS Properties Are Translated for Custom Utilities

AQ/JMS Utilities	Translated Value—For Custom Utilities	JMS Message
from	FROM_PARTY	Sent as a string type message property
to	TO_PARTY	Sent as a string type message property
doctypeName	DOCTYPE_NAME	Sent as a string type message property
doctypeRevision	DOCTYPE_REVISION	Sent as a string type message property
eventName	ACTION_NAME	Sent as a string type message property
msgID	MSG_ID	Sent as a string type message property
replyToMsgID	INREPLYTO_MSG_ID	Sent as a string type message property
msgType	MSG_TYPE	Sent as a string type message property
attachment	ATTACHMENT	Sent as a string type message property
payload	-	Sent as the message body

23.2.3 JMS Dequeue

The JMS dequeue utility, oracle.tip.b2b.data.JMSDequeue, receives messages from the destination. The count property can be specified to control the number of messages to be picked up from the destination. Retrieved messages are written to the file JMSDequeue.txt at the current path (where you run the utility).

See the samples documentation on Oracle Technology Network for how to run these utilities.

Example 23–2 shows the sample JMS dequeue properties file.

Example 23–2 Sample jms_dequeue.properties File

```
####### Destination Details#######
destination = jms/b2b/B2B_IN_QUEUE
cf = jms/b2b/B2BQueueConnectionFactory
count=1

####### Server and Factory Details #######
factory=weblogic.jndi.WLInitialContextFactory
url=t3://host_name:port_number/
#user=<uncomment and provide your username>
#password=<uncomment and provide your password if required>
```

23.3 Using the attachmentsDescriptor.xsd

Use the `attachmentsDescriptor.xsd` file for sending attachments.

Example 23–3 shows a sample attachment XML file.

Example 23–3   Sample Attachment XML File

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--Sample XML file generated by XMLSpy v2005 sp1 U  (http://www.xmlspy.com)-->
 <Attachments xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="AttachmentsDescriptor.xsd" version="1.0" boundary="boundary---">
       <AttachmentPart>
               <Location>file:///home/user_dir/data.xml</Location>
               <Content-Type>
                       <Top-Level-Type>text</Top-Level-Type>
                       <Sub-Type>plain</Sub-Type>
                       <Parameter Value="charset" Name="us-ascii"/>
               </Content-Type>
               <!--Content-Transfer-Encoding>BASE64</Content-Transfer-Encoding-->
               <Content-ID/>
               <Content-Description/>
       </AttachmentPart>
</Attachments>
```

24

Monitoring Instance Message Data With Oracle BAM

This chapter provides end-to-end instructions for setting up Oracle B2B instance message monitoring with Oracle BAM.

The Oracle B2B-BAM integration does not work with Oracle XE database. It will work with Oracle 10g EE and Oracle 11g EE versions.

This chapter contains the following topics:

- Section 24.1, "Introduction to Monitoring Oracle B2B with Oracle BAM"
- Section 24.2, "Creating a Data Source in Oracle WebLogic Server"
- Section 24.3, "Creating a Foreign JMS Server"
- Section 24.4, "Create a B2B Data Object using Oracle BAM Architect"
- Section 24.5, "Define a JMS Message Source Using an Oracle BAM Enterprise Message Source"
- Section 24.6, "Mapping Oracle B2B Instance Messages to Oracle BAM Data Objects"
- Section 24.7, "Creating a Dashboard to Monitor Oracle B2B"
- Section 24.8, "Enabling the Oracle BAM Integration in Oracle B2B"

24.1 Introduction to Monitoring Oracle B2B with Oracle BAM

Oracle BAM provides a framework for creating dashboards that display real-time data inflow and creating rules to send alerts under specified conditions. Oracle BAM can be configured to monitor an Oracle B2B instance by following the procedures in the remaining sections of this chapter.

> **Note:** While it is possible to create an Oracle BAM dashboard on an external data source defined on the instance message table, active data is not supported on external data sources, so the resulting report does not display real-time updates.

24.2 Creating a Data Source in Oracle WebLogic Server

You could use an existing data source if it exists or create a new data source following the steps below:

1. Open the Oracle WebLogic Server Administration Console and log in.

```
http://host_name:port_number/console
```

The *host_name* is the name of the machine where Oracle BAM is installed, and the default *port_number* for Oracle WebLogic Server is 7001.

2. Select **Data Sources** in the **JDBC** section, and click **New**.

3. Configure the data source:

   a. Enter a **Name** for the data source (for example, BAMAQDataSource).

   b. Enter a JNDI name from the data source (for example, jdbc/oracle/bamaq). This name is used to configure a foreign JMS server.

   c. Select **Oracle** for the **Database Type**.

   d. Select **Oracle's Driver (Thin)** in the **Database Driver** field.

   e. Click **Next**.

4. Uncheck **Support Global Transaction**, and click **Next**.

5. Enter your Database SID into the **Database Name** field (for example, ORCL).

6. Enter the host name of the machine where the database is installed as the **HostName** (for example, localhost).

7. Enter the database **Port** number (for example, 1521).

8. Enter the **Username** and **Password**, and click **Next**.

9. Click **Test Configuration**.

10. Once the test is successful, click **Next.**

11. Select the server where Oracle BAM is deployed, and click **Finish**.

24.3 Creating a Foreign JMS Server

To create a foreign JMS server:

1. Add an Oracle WebLogic Server JMS module.

   a. In Oracle WebLogic Server Administration Console, on the Home page navigate to the **JMS Modules** page.

   b. Click New to create a new Oracle WebLogic Server JMS module.

   c. Enter a name for the JMS module (for example, BAMAQsystemModule), and click **Next**.

   d. Assign the target as the server where Oracle BAM is deployed, click **Next**, and click **Finish**.

   > **Note:** It is recommended that you create a Subdeployment. Select **JMS Module>SubDeployment>New**. Use the subdeployment in the subsequent steps where you have the **Subdeployment** tab.

2. Target the Subdeployment to the JMS Server by selecting it from the list.

3. Add an AQ JMS Foreign Server to the JMS module.

   a. Select the JMS module that was previously created.

   b. Click **New**, and go to the list of JMS resources.

    c. Select the **Foreign Server** option, and click **Next**.

    d. Enter a **Name** for the Foreign Server (for example, `BAMAQForeignServer`), and click **Finish**.

4. Configure the AQ JMS Foreign Server.

    a. Select the AQ JMS Foreign Server that was previously created.

    b. Enter `oracle.jms.AQjmsInitialContextFactory` in the **JNDI Initial Context Factory** field.

    c. Enter `datasource=`*`data_source_jndi_location`* In the **JNDI Properties** field, where *`data_source_jndi_location`* is the JNDI location of your data source (for example, `jdbc/oracle/bamaq`).

    d. Check the **Default Targeting Enabled** checkbox.

5. Add Connection Factories to the AQ JMS Foreign Server.

    a. Select the AQ JMS Foreign Server that was previously created.

    b. Select the **Connection Factories** tab.

    c. Enter a name for the connection factory. This is a logical name referenced by Oracle WebLogic Server.

    d. Enter the local JNDI name that will be used by Oracle BAM Enterprise Message Source to look up this connection factory in the **Local JNDI Name** field (for example, `jms/BAMAQQueueCF`).

    e. Enter `QueueConnectionFactory` in the **Remote JNDI Name** field.

Note: The following are the alternative options available. If you use this connection factory in a global transaction, select an XA-based connection factory, otherwise select a non-XA based connection factory.

- QueueConnectionFactory
- ConnectionFactory
- XAQueueConnectionFactory
- XATopicConnectionFactory
- XAConnectionFactory

---

    f. Click **OK**.

6. Add Destinations to the AQ JMS Foreign Server.

    a. Select the AQ JMS Foreign Server that was previously created.

    b. Select the **Destinations** tab

    c. Enter a name for the destination. This is a logical name referenced by Oracle WebLogic Server. It is not related to the destination name.

    d. Enter the local JNDI name that will be used by Oracle BAM Enterprise Message Source to look up the destination in the **Local JNDI Name** field (for example, `jms/BAMAQQueue`).

    e. In the **Remote JNDI Name** field, enter `Queues/queue_name` if the destination is a queue, or enter `Topics/topic_name` if the destination is a topic.

    For example, the default name for an AQ queue created for the Oracle B2B - Oracle BAM integration is `Queues/B2B_BAM_QUEUE`.

    f. Click **OK**.

7. Restart Oracle WebLogic Server.

24.4 Create a B2B Data Object using Oracle BAM Architect

To create a B2B data object:

1. Log in to Oracle BAM at the following URL:

```
http://host_name:port_number/OracleBAM
```

The default port number is 9001.

2. Select **Architect** to open the Oracle BAM Architect application.

3. Select **Data Objects** in the drop down list.

4. Select the **Samples** folder, and create a subfolder named **B2B**.

5. Click on the **Create Data Object** link

6. Enter `B2BDO` in the **Name** field.

7. Click **Add a Field**.

8. Enter values for **Fields in Data Object**.

9. Repeat, clicking **Add a Field** to add all fields as listed in Table 24–2.

For more information about creating Oracle BAM data objects, see "Defining and Managing Oracle BAM Data Objects" in *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

24.5 Define a JMS Message Source Using an Oracle BAM Enterprise Message Source

To define a JMS message source:

1. Log in to Oracle BAM at the following URL:

```
http://host_name:port_number/OracleBAM
```

The default port number is 9001.

2. Select **Architect** to open the Oracle BAM Architect application.

3. Select **Enterprise Message Sources** in the drop down list.

4. Click **Create**, and enter the Enterprise Message Source information for AQ JMS Queue as indicated in Table 24–1, leaving unmentioned fields with default or blank entries.

Table 24–1   Enterprise Message Source Properties for AQ JMS Queue

Property	Values and Examples
Name	Enter a display name for the Enterprise Message Source (for example, B2B_EMS).
Initial Context Factory	Enter weblogic.jndi.WLInitialContextFactory
JNDI Service Provider URL	Enter t3://localhost:9001
	Note: If Oracle BAM is installed is on an Administrator Server use port 9001. If Oracle BAM is installed on a managed server use default port 7001, or the port number configured by the system administrator.
Topic/Queue Connection Factory Name	Enter the name of your connection factory (for example, jms/BAMAQQueueCF).
Topic/Queue Name	Enter the name of your JJMS topic (or queue) (for example, jms/BAMAQQueue).
Data Object Name:	Choose the B2B data object to send the values received from AQ JMS server
Operation	Choose the **Insert** or **Upsert** operation, depending on the use case.

5. To configure **Source Value Formatting**, check the **DateTime Specification** checkbox, and select the **MM/dd/yy H:mm:ss** format in the drop-down list.

> **Note:**   The incoming XML value for the date, in the format `MM/dd/yy HH24:MI:SS`, must be converted into the `MM/dd/yy H:mm:ss` format that the Enterprise Message Source expects so that it can be stored correctly in the data object

6. Complete the **Source to Data Object Field Mapping** so that data from the incoming XML can be mapped to an appropriate field in the B2B data object. See Table 24–2 for a list of fields in the incoming XML which will be retrieved from the B2B instance table.

7. Click **Save**.

8. Select the B2B_EMS Enterprise Message Source, and click **Start**. The status changes to **Started** in a few seconds.

After starting the Enterprise Message Source, any messages that are inserted by the database job into the AQ JMS queue are instantly available in the B2BDO data object.

To view the contents of the B2BDO data object, select Data Objects from the Oracle BAM Architect drop-down menu. Browse and select the B2BDO object. Click **Contents** to view the contents.

New rows are inserted into B2BDO as the database job reads incoming messages from the B2B instance table and inserts them onto the AQ JMS queue.

For more information on creating Oracle BAM Enterprise Message Sources, see "Creating Oracle BAM Enterprise Message Sources" in *Oracle Fusion Middleware Developer's Guide for Oracle SOA Suite*.

24.6 Mapping Oracle B2B Instance Messages to Oracle BAM Data Objects

Table 24–2 lists the Oracle B2B instance message fields. These fields can be used to design data objects and Enterprise Message Sources in Oracle BAM to monitor Oracle B2B performance in real time.

Note that the payloads (`APP_PAYLOAD`, `PAYLOAD`, `WIREPAYLOAD`) are not included in the list below. They are not transferred for performance reasons.

Table 24–2   Oracle B2B Instance Message Fields

Field	Type
ID	NOT NULL VARCHAR2(256)
REFERTOID	VARCHAR2(256)
B2BMESSAGEID	NOT NULL VARCHAR2(256)
ACKNOWLEDGEMODE	VARCHAR2(256)
MESSAGEDATETIME	DATE
MESSAGETYPE	VARCHAR2(256)
STATE	VARCHAR2(256)
REMAININGRETRY	NUMBER(10)
DIRECTION	VARCHAR2(256)
TIMETOACK	VARCHAR2(256)
TPA_NAME	VARCHAR2(256)
XPATH_EXPRESSION1	VARCHAR2(1024)
XPATH_EXPRESSION2	VARCHAR2(1024)
XPATH_EXPRESSION3	VARCHAR2(1024)
XPATH_NAME1	VARCHAR2(256)
XPATH_NAME2	VARCHAR2(256)
XPATH_NAME3	VARCHAR2(256)
XPATH_VALUE1	VARCHAR2(256)
XPATH_VALUE2	VARCHAR2(256)
XPATH_VALUE3	VARCHAR2(256)
DOCUMENT_DEFINITION	VARCHAR2(256)
CREATED	DATE
MODIFIED	DATE
SEND_TIMESTAMP	DATE
RECEIVE_TIMESTAMP	DATE
NATIVE_MSG_SIZE	NUMBER(10)
TRANSLATED_MSG_SIZE	NUMBER(10)
BM_RESUBMIT_COUNT	NUMBER(10)
ERRORCODE	VARCHAR2(256)
ERRORTEXT	VARCHAR2(1024)

Table 24–2 (Cont.) Oracle B2B Instance Message Fields

Field	Type
ERRORDESCRIPTION	VARCHAR2(2000)
PROCESSING_TIME	NUMBER(10)
DOCUMENTTYPE	VARCHAR2(513)
CORRELATIONID	VARCHAR2(256)
FABRIC_ECID	VARCHAR2(512)
FABRIC_COMPOSITE_NAME	VARCHAR2(512)
AM_RESUBMIT_COUNT	NUMBER(10)
TRANSACTIONCONTROLNUMBER	VARCHAR2(256)
GROUPCONTROLNUMBER	VARCHAR2(256)
INTERCHANGECONTROLNUMBER	VARCHAR2(256)
PROTOCOLWORKAREA	CLOB
B2BWIREMESSAGEID	VARCHAR2(256)
B2BTIMESTAMP	DATE
CONVERSATIONID	VARCHAR2(1024)
PROTOCOLMESSAGEID	VARCHAR2(1024)
URL	VARCHAR2(1024)
TRANSPORTPROTOCOL	VARCHAR2(513)
TRANSPORTHEADERS	VARCHAR2(2000)
WM_RESUBMIT_COUNT	NUMBER(10)
DOCUMENTPROTOCOL	VARCHAR2(513)
EXCHANGEPROTOCOL	VARCHAR2(513)
AGREEMENTID	VARCHAR2(256)
SENDERNAME	VARCHAR2(256)
SENDERID	VARCHAR2(513)
RECEIVERNAME	VARCHAR2(256)
RECEIVERID	VARCHAR2(513)

24.7 Creating a Dashboard to Monitor Oracle B2B

Real-time monitoring dashboards are created using Oracle BAM Active Studio. Create a dashboard to monitor Oracle B2B using the B2BDO data object.

To create a simple report:

1. Log in to Oracle BAM at the following URL:

    ```
    http://host_name:port_number/OracleBAM
    ```

    The default port number is 9001.

2. Select **Active Studio** to open the Oracle BAM Active Studio application.

3. Click **Create A New Report**.

4. Select a **Tiled Report** with four quadrants.

5. In the upper left quadrant, select **Bar Chart** (you may need to click twice).

   A sample bar chart is displayed, and the view editor opens at the bottom of the Oracle BAM Active Studio window.

6. In the view editor Data Objects selection, select **B2BDO** and click **Next**.

7. In the **Group By** list, select **state**.

8. In the **Chart Values** list, select **b2bMessageId** and click **Next**.

9. Click **Change View Properties** and enter the following information:

   Enter `B2B State` in the **View Title** field.

   Check the **Display Legend** checkbox.

10. Click **OK**.

11. Similarly, follow the instructions in the *Oracle Fusion Middleware User's Guide for Oracle Business Activity Monitoring* to create display information for the other quadrants in the dashboard.

    For example, you could group by `documentProtocol`, `exchangeProtocol` or `direction` of the messages in the other three quadrants.

12. Once you are finished configuring view for each of the quadrants, save the dashboard.

13. Go to the main Oracle BAM page and select **Active Viewer**.

14. Select the dashboard that you created to view it in Oracle BAM Active Viewer.

15. The views in the dashboard refresh automatically when any new instances are created in the `b2b_instancemessage` view, and the data is pushed into the `B2BDO` data object. The changes are reflected instantaneously.

For more information about creating Oracle BAM dashboards, see *Oracle Fusion Middleware User's Guide for Oracle Business Activity Monitoring*.

24.8 Enabling the Oracle BAM Integration in Oracle B2B

To enable the Oracle BAM integration in Oracle B2B, set the following properties in the Oracle B2B **Administration>Configuration** tab:

Enable BAM - Set to `true`. When `b2b.enableBAM` is set to true, the Oracle B2B user interface automatically starts the database job in the database to monitor new instances in the `b2b_instancemessage` view. When `b2b.enableBAM` is set to `false`, the user interface automatically removes the database job from the database.

BAM Polling Interval – the interval in minutes when the database job polls the Oracle B2B instance table for any updates and puts them on the AQ JMS queue for uptake by Oracle BAM. Values less than a minute can also be specified, for example, 0.5 minutes.

The Miscellaneous (continued) section of the Administration>Configuration tab in Oracle B2B console. The Enable BAM (set to true) and BAM Polling Interval (set to 5) properties are the last two properties shown on the bottom right side of this section of the tab.

****

25

Programmatically Accessing Instance Message Data

This chapter provides an overview of the Instance Message Java API for programmatically accessing instance message data.

This chapter contains the following topics:

For more information about the Instance Message API, `InstanceMessageUtil` class, see *Oracle Fusion Middleware B2B Instance Message Java API Reference*.

25.1 Programmatically Accessing Instance Message Data

Instance Message Java API is a Java API provided in Oracle B2B to retrieve runtime data that is exchanged between Trading Partners.

Some Instance Message Java API use cases include:

- Resubmitting from a back-end application

- Checking the status for a particular order number

- Payload introspection by a back-end application

- Checking the remaining retries for the message

- Checking the wire message of the payload

Instance Message API queries the runtime data and provides results in the form of Instance data for various criteria. Instance Message API is a java wrapper on top of Instance Message View which encapsulates both business and wire messages.

> **Note:** When using this API, you must set additional libraries as part of the classpath in addition to `b2b.jar` and `weblogic.jar` or `wlthint3client.jar`.

Instance Message API Query Options

The Instance Message Java API enables you to retrieve the instance messages based on the following criteria:

- Agreement name

- Control number, group control number, transaction control number, Trading Partner name, and document type

- Date range and Trading Partner

- Date range, direction, and Trading Partner

- Message ID

- Message state

- Protocol message ID

- Trading Partner

- Trading Partner and direction

- Trading Partner and document type

- Trading Partner and message state

- Xpath expression (can be evaluated on one or two XPath expressions)

- XPath value (can be evaluated on one or two XPath expressions)

- Conversation ID

For more information about the Instance Message API, see *Oracle Fusion Middleware B2B Instance Message Java API Reference*.

25.2 Instance Message Java API Examples

You can write your logic based on the out put of

```
Vector messages = (Vector) imUtil.getInstanceMessagesForTP("GlobalChips")
```

as shown in Example 25–1.

Another similar exposed method is `getInstanceMessagesForMsgId(String messageId)`. All of the methods are documented in *Oracle Fusion Middleware B2B Instance Message Java API Reference*.

Example 25–1   Retrieval based on Trading Partner Name
```
import java.util.Vector;
import oracle.tip.b2b.utility.InstanceMessageUtil;

public class MyInstanceData {
    public static void main(String[] args) throws Exception {

        InstanceMessageUtil imUtil = new InstanceMessageUtil();
        Vector messages = (Vector) imUtil.getInstanceMessagesForTP("GlobalChips");
        System.out.println(messages.size());
    }
}
```

25.3 Troubleshooting Instance Message Java API

See the following topics for troubleshooting information:

- Section 25.3.1, "Handling a Large Number of Messages"

- Section 25.3.2, "Handing Large Payloads"

■ Section 25.3.3, "Executing Queries Remotely Using Weblogic.jar"

25.3.1 Handling a Large Number of Messages

If the query returns a large number of messages then the
`weblogic.socket.MaxMessageSizeExceededException` may occur.

To recover from this issue, increase `weblogic.MaxMessageSize`, and pass
`-Dweblogic.MaxMessageSize=100000000` in the client command line. This
parameter setting will allow you to receive 100 MB messages.

For example:

```
java -cp
.:$ORACLE_HOME/wlserver_10.3/server/lib/wlthint3client.jar:$SOA_HOME/
/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar
-Dweblogic.MaxMessageSize=100000000 TestInstanceMsg
```

25.3.2 Handing Large Payloads

Rather than return the payload itself, a large inbound or outbound payload will return
as a reference to the location of the payload.

For example:

```
/tmp/GlobalChips_1_custnonxml_largepayload.dat@9844C4341297D3EB7B60000011A0CC9D
.dat
```

See Appendix A, "Handling Large Payloads" for more information about how Oracle
B2B handles large payloads.

25.3.3 Executing Queries Remotely Using Weblogic.jar

If while using `weblogic.jar` and executing queries remotely you may see classpath
issues for Oracle WebLogic Server, and the
`java.lang.ClassNotFoundException:weblogic.security.subject.Abstr`
`actSubject` exception.

To recover, execute queries using `wlthint3client.jar`.

For example:

```
java -cp
.:$ORACLE_HOME/wlserver_10.3/server/lib/wlthint3client.jar:$SOA_HOME/
/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar TestInstanceMsg
```

Part VI

Appendixes

This part contains the following appendixes:

- Appendix A, "Handling Large Payloads"

- Appendix C, "Setting B2B Configuration Properties in Fusion Middleware Control"

- Appendix D, "Back-End Applications Interface"

- Appendix G, "Exception Handling"

- Appendix H, "Self Service Utility Protocols, Identifications, Security Specifications, and Parameters"

A

Handling Large Payloads

This appendix describes how to use Oracle B2Bto handle large payloads by using the SOA Infrastructure and JMS internal queues.

The appendix contains the following topics:

A.1 Handling Large Payloads

Oracle B2B can handle large payloads through the SOA Infrastructure and JMS internal queues.

A.1.1 Introduction to Large Payload Support

Inbound Setup

Figure A–1 shows the properties to set for inbound cases. Go to **Administration** > **Configuration**.

Figure A–1   Large Payload Size

The image shows the following Performance fields: Large Payload Size (the default value is 2000000) and Large Payload Directory (the default value is /tmp).

****

If a composite is deployed to handle the large payload, this is the only configuration needed. If B2B is not delivering the payload to a composite, set **Use JMS Queue as default** to true, as shown in Figure A–2. Go to **Administration** > **Configuration**.

Figure A–2   Use JMS Queue

The image shows the following Generic fields: Use JMS Queue (the value shown is true), Use B2B Queue (the value shown is false), and Callout Directory (the value shown is /MyCalloutDir).

****

With **Use JMS Queue as default** set to true, the payload is delivered to `B2B_IN_QUEUE`, a JMS-based queue.

Outbound Setup

Figure A–3 shows the properties to set for the outbound case.

Figure A–3   Large Payload Directory
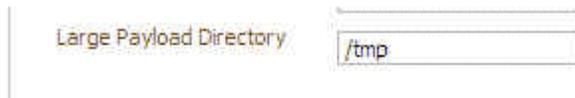
The image shows the Large Payload Directory field (the default value is /tmp).

****

You must also notify B2B that a service engine is sending a large payload. The change involves two steps:

The `b2b.largePayload` property must be set in the BPEL process when sending a large payload to Oracle B2B. For composite samples, which do not handle large payload, there is no change.

Code change in Oracle B2B to handle this flag

1. Declare the Variable_largePayload variable in an outbound BPEL process in the `<variables>` section.

   ```
   <variable name="Variable_largePayload" type="xsd:string"/>
   ```

2. In the Assign activity, copy `'true'` into the variable.

   ```
   <copy>
    <from expression="'true'"/>
    <to variable="Variable_largePayload"/>
   </copy>
   ```

3. Assign the variable to `b2b.largePayload` in the Invoke activity.

   ```
   <bpelx:inputProperty name="b2b.largePayload"
    variable="Variable_largePayload"/>
   ```

Notes:

If BPEL is not sending a large payload to Oracle B2B, this property should not be set.

Once the code is checked in, any Large Payload Sample must be updated to confirm to this.

In BPEL and Mediator, if `b2b.largePayload` is set to `true`, then `largePayloadDir` must be present (set it in Oracle B2B). If `b2b.largePayload` is not set, then this directory should not matter.

Oracle B2B retains the large payloads in the large payload processing directory, after sending the payload to corresponding endpoints.

About Large Payload Support

1. If you are doing large payload testing, set **Log Payload** on the **Administration** > **Configuration** tab to false.

2. If you are doing large payload testing, set **Show Payload** on the **Administration** > **Configuration** tab to false to avoid listing the payload in reports.

3. If an enqueue script is used when working with large payloads, add

   ```
   eventName=LARGE_PAYLOAD=true
   ```

4. Increase the maximum heap size to use `-Xmx2048m`.

5. Increase the database tablespace size for soadatasource to have autoextend on and increase the tablespace file size maximum limit.

   ```
   alter database datafile
   '/scratch/$user/auto_work/db230/oradata/db230/SH_soainfra.dbf
   ' autoextend on next 10M maxsize 4096M
   ```

6. Set the transaction timeout in Oracle WebLogic Administration Server:

   - Weblogic Console Services -> JTA Timeout Seconds=720 seconds

   - Weblogic Console Services -> JDBC->DataSources->SOADataSource - increase XA timeout to 120-180 seconds

7. If Oracle B2B is used alone (without the SOA Infrastructure), the JTA timeout can be set in `b2b.jtaTimeout` by using Oracle Enterprise Manager Fusion Middleware Control. See *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite* for more information.

8. For an outbound SOA composite, always select the **Use file streaming** option for the File Adapter, as shown in Figure A–4.

Figure A–4   The File Adapter Use File Streaming Option

The image shows the following File Adapter operation types: Read File (selected), Write File, Synchronous Read File, and List Files. The Operation Name field displays Read. The Do not read file content option is not checked. The Use file streaming option is checked. Other fields displayed are Character Set, Encoding, and Content Type (no values are shown).

****

A.1.2 Large Payloads and 32-Bit Windows PCs

On a 32 bit Windows computer, the payload size limit is 50 MB. This is because the heap size cannot be set to more than 1536m due to Windows-specific limits. Java VM throws an out-of-memory exception.

A.1.3 Settings for a Large Dataset Scenario

The following suggested settings are based on a dataset with approximately 2,500 trading partners, an export ZIP file that is approximately 253 MB in size, and assumes a 6 GB computer. Using these settings can considerably reduce data upload time when using the Upgrade Assistant.

1. Use Oracle WebLogic Server Administration Console to increase the

    ■   JTA transaction timeout from 30 to 350

    ■   Maximum message size from the default size to 200000000

2. Add indices for better performance. Using Oracle Database 11g Enterprise Edition Release 11.1.0.7.0 - Production, with the Partitioning, OLAP, Data Mining and Real Application Testing options, do the following:

```
SQL> create index idx_mds_attr on
rc1_mds.MDS_ATTRIBUTES("ATT_VALUE","ATT_LOCALNAME");
Index created.

SQL> create index idx_mds_path on
rc1_mds.MDS_PATHS("PATH_CONTENTID","PATH_PARTITION_ID");
Index created.

SQL> commit;
```

3. Start the managed server with the following updated memory setting:

```
DEFAULT_MEM_ARGS="-Xms1024m -Xmx2048m"
```

4. Change *ORACLE_HOME*/bin/UA default memory from the default 256 to 2048. The default is

```
$JAVA_HOME/bin/java ${JAVAMODE} -Xmx256m -classpath ${CLASSPATH}
-Dua.home=$base_dir -Dice.pilots.html4.ignoreNonGenericFonts=true
-Dsun.lang.ClassLoader.allowArraySyntax=true
-Doracle.installer.oui_loc=$OUI_HOME oracle.ias.upgrade.UpgradeDriver
$ARGUMENTS
```

Change the default to

```
$JAVA_HOME/bin/java ${JAVAMODE} -Xmx2048m -classpath ${CLASSPATH}
-Dua.home=$base_dir -Dice.pilots.html4.ignoreNonGenericFonts=true
-Dsun.lang.ClassLoader.allowArraySyntax=true
-Doracle.installer.oui_loc=$OUI_HOME oracle.ias.upgrade.UpgradeDriver
$ARGUMENTS
```

5. Change the value of Stuck Thread Max Time from 600 to 2000.

B

Synchronous Request/Reply Support

This appendix describes how to configure Oracle JDeveloper and Oracle B2B to enable synchronous request and reply support in Oracle B2B.

The appendix contains the following topics:

- Section B.1, "Introduction"
- Section B.2, "Configuring Sender"
- Section B.3, "Configuring Receiver"
- Section B.4, "Resubmitting Messages"

B.1 Introduction

Oracle B2B provides synchronous request/response between from Fabric to Oracle B2B. However, only HTTP transport is supported for the initial release. Figure B–1 shows the end-to-end scenario.

Figure B–1 Synchronous Request/Reply
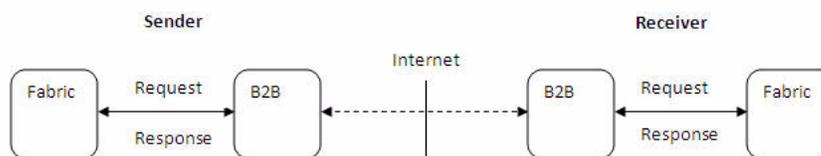
The figure displays the end-to-end scenario of synchronous request/reply support in Oracle B2B.

****

B.2 Configuring Sender

To configure the sender, you need to perform the following tasks:

- "Configuring Oracle JDeveloper"
- "Configuring Oracle B2B"

B.2.1 Configuring Oracle JDeveloper

You need to use B2B Configuration Wizard in Oracle JDeveloper to configure the sender for synchronous request/reply support:

■ In the **Operations** page of B2B Configuration Wizard, select **Synchronous Request/Reply** option, and then select **Outbound** as shown in Figure B–2.

Figure B–2 B2B Configuration Wizard: Operation Page
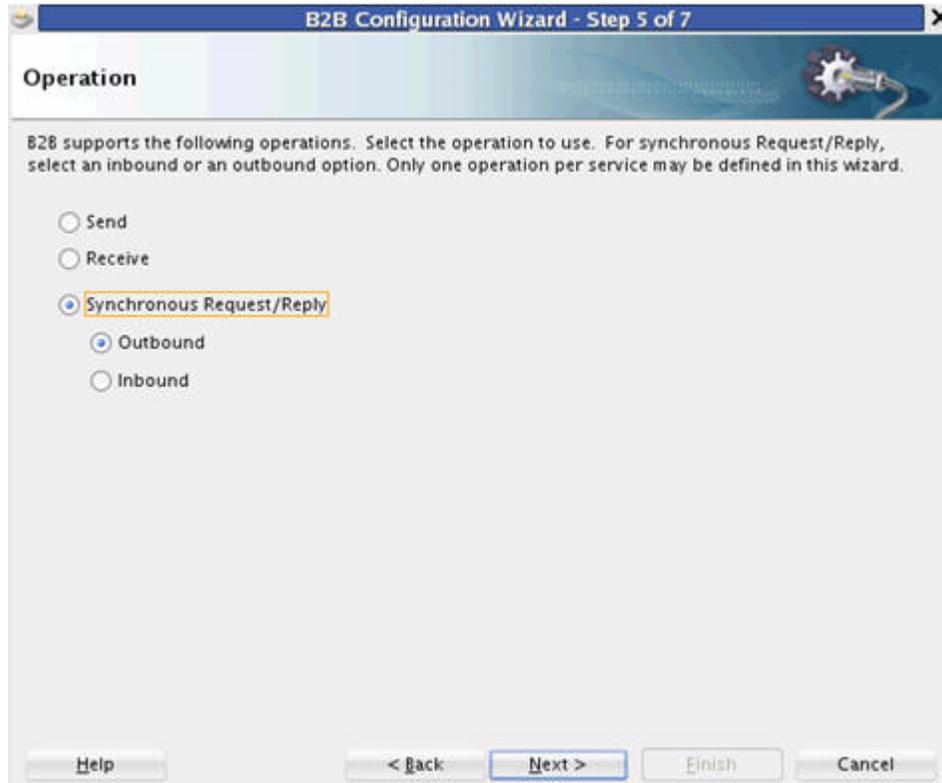
The figure displays that you need to select Synchronous Request/Reply option, and then select Outbound in theB2B Configuration Wizard: Operation Page.

****

■ In the **Document Definition** page of B2B Configuration Wizard, specify the definitions for both Request and Reply.

The **Routing ID** selection only appears for the Request document. The Response is returned to Fabric on the same request call, so no Routing ID is required to route the response as shown in Figure B–3.

Figure B–3   B2B Configuration Wizard: Document Definition Page
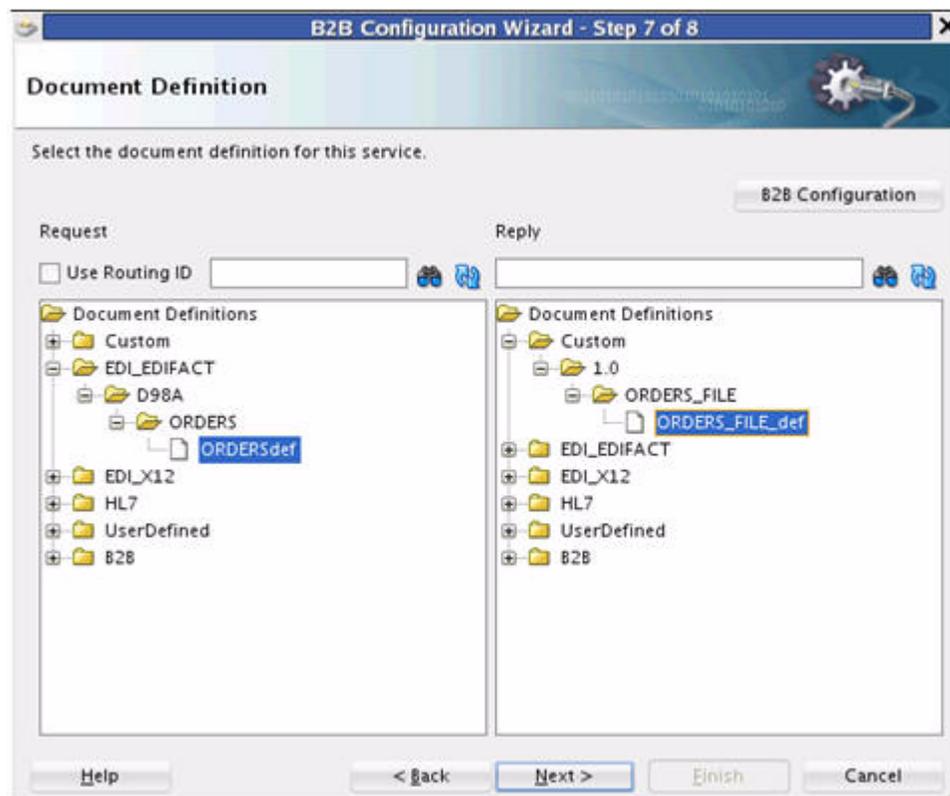
The figure displays how to specify the document definition for the sender.

****

B.2.2 Configuring Oracle B2B

In Oracle B2B console, you need to:

- **Set up Oracle B2B agreements:** You need to define two agreements. One agreement is required to send the Request, and the other agreement is needed to receive the Response as shown in Figure B–4. You do not need to set up any Oracle B2B property for this. Based on the call from the Fabric to Oracle B2B, the message type "sync request" is determined.

Figure B–4   Setting Up Oracle B2B Agreements
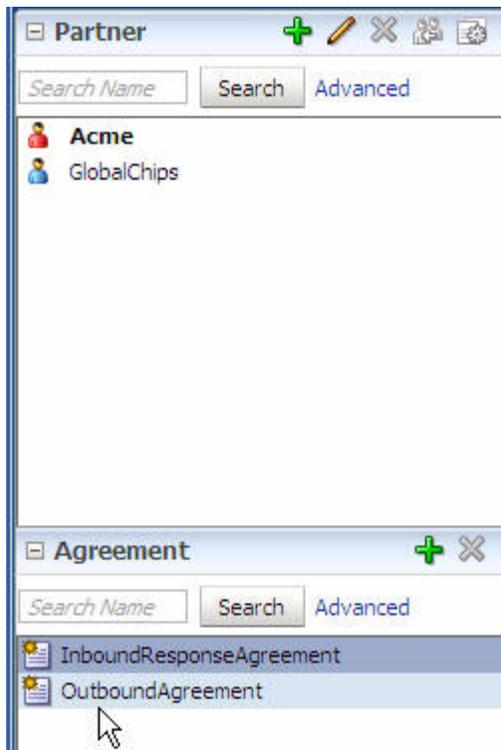
This figure displays how to set up Oracle B2B agreements in the Oracle B2B console.

****

- **Create a Trading Partner channel:** You need to create a remote Trading Partner Channel to point to `http://server:port/b2b/syncreceiver` as shown in Figure B–5. This is required if the communication is between two Oracle B2B products. Please note that the URL will be different if the communication takes place between two different B2B products.

Figure B–5   Creating a Trading Partner Channel

The figure displays the creation of a trading partner channel for the sender.

****

B.3 Configuring Receiver

To configure the receiver, you need to perform the following tasks:

- "Configuring Oracle JDeveloper"
- "Configuring Oracle B2B"

B.3.1 Configuring Oracle JDeveloper

You need to use B2B Configuration Wizard in Oracle JDeveloper to configure the receiver for synchronous request/reply support:

■ In the **Operations** page of B2B Configuration Wizard, select **Synchronous Request/Reply** option, and then select **Inbound** as shown in Figure B–6.

Figure B–6   B2B Configuration Wizard: Operation Page
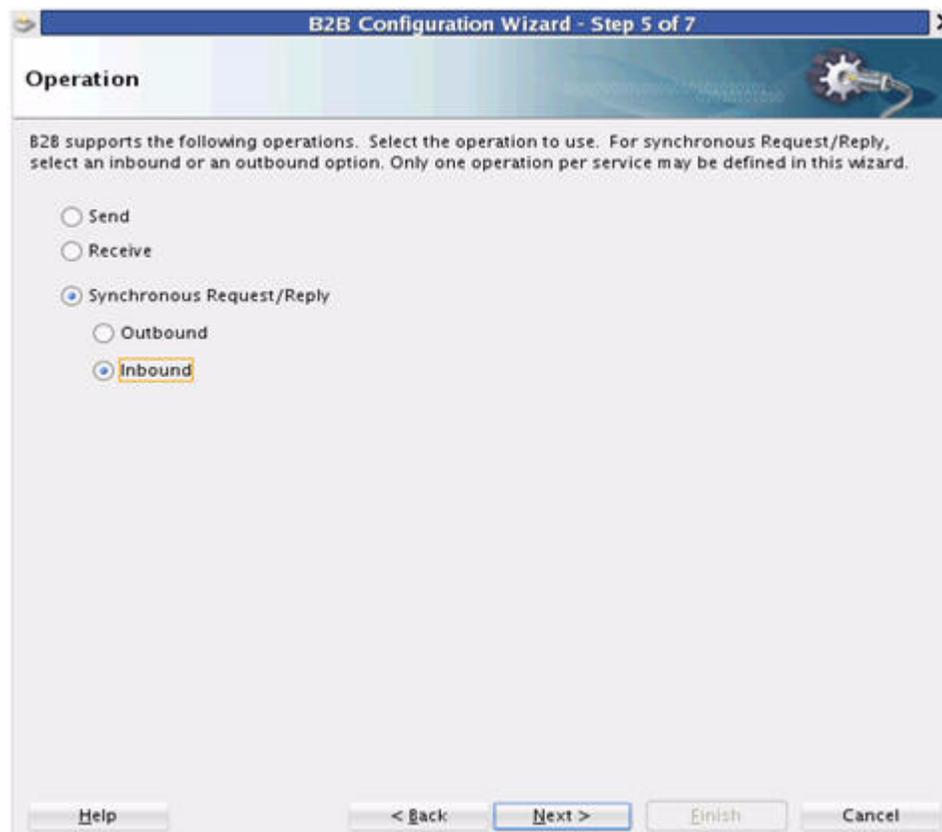
The figure displays that you need to select Synchronous Request/Reply option, and then select Inbound in theB2B Configuration Wizard: Operation Page.

****

■ In the **Document Definition** page of B2B Configuration Wizard, specify the definitions for both Request and Reply.

The **Routing ID** selection only appears for the Request document. The Response is returned to Fabric on the same request call, so no Routing ID is required to route the response as shown in Figure B–7.

Figure B–7   B2B Configuration Wizard: Document Definition Page
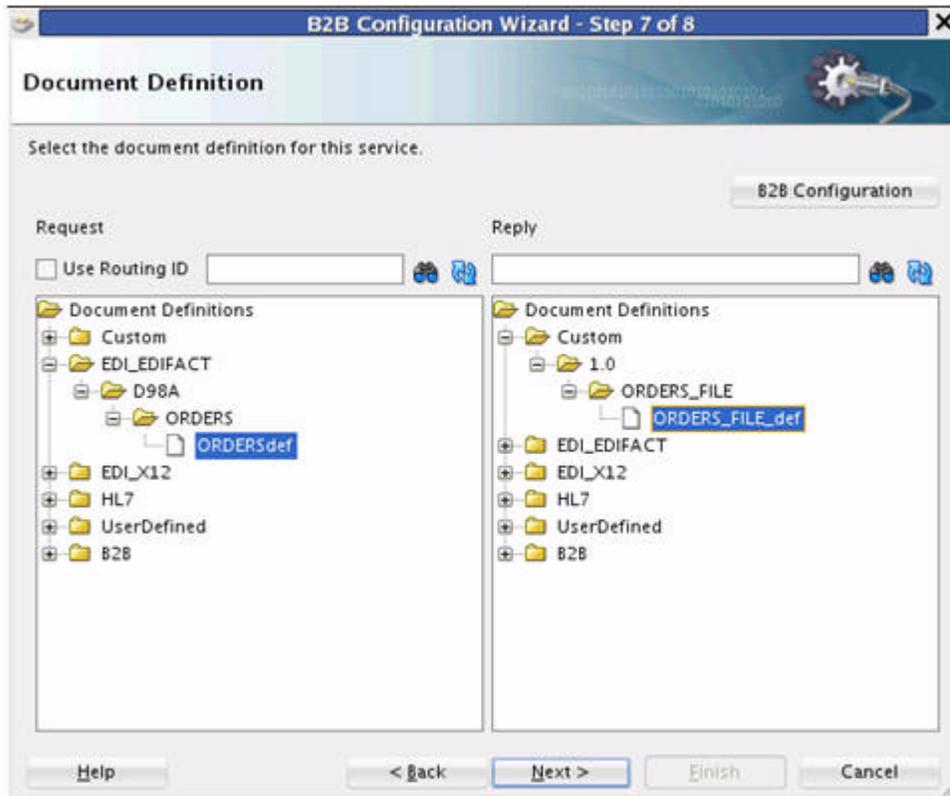
The figure displays how to specify the document definition for the receiver.

****

B.3.2 Configuring Oracle B2B

In Oracle B2B console, you need to:

- **Set up Oracle B2B agreements:** You need to define two agreements. One agreement is required to receive the Request, and the other agreement is needed to send the Response as shown in Figure B–8. You do not need to set up any Oracle B2B property for this. The synchronous HTTP receiver decides the message type as "sync request".

Figure B–8   Setting Up Oracle B2B Agreements
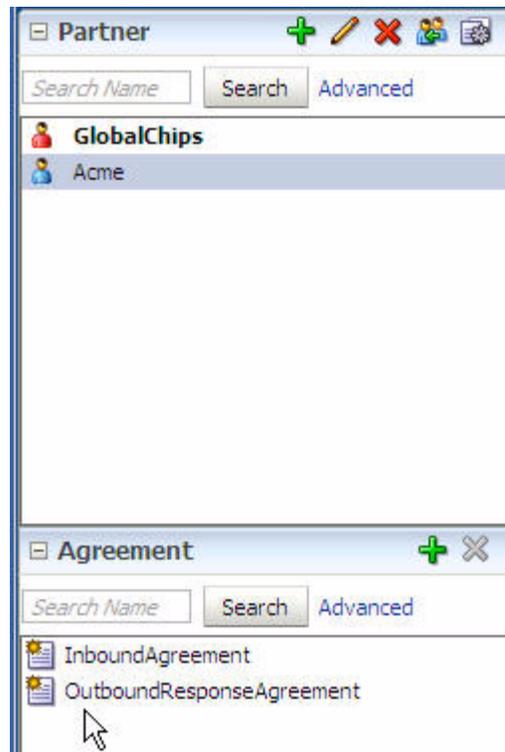
This figure displays how to set up Oracle B2B agreements in the Oracle B2B console.

****

- **Create a dummy Trading Partner channel:** You need to create a dummy remote Trading Partner Channel, but this channel is not used at run time. You can specify any URL value, but the value needs to be in the format `http://host:port/xyz` as shown in Figure B–9.

Figure B–9   Creating a Dummy Trading Partner Channel

The figure displays the creation of a dummy trading partner channel.

****

B.4 Resubmitting Messages

Synchronous request or response resubmit *must* be done from the Application side. Although the Oracle B2B console allows resubmit of Application or Wire messages from Oracle B2B, however, it resubmits the messages in the as nonsynchronous mode. Sometimes, the message processing logic is nondeterministic. For example, if an Application or a Wire message is resubmitted from Sender side Oracle B2B, the

message is posted to `http://server:port/b2b/syncreceiver`. The Receiver Oracle B2B processes the message as synchronous request or response. The response message on the Sender side is ignored.

C

Setting B2B Configuration Properties in Fusion Middleware Control

This appendix describes how to use Oracle Enterprise Manager Fusion Middleware Control to set B2B configuration properties for properties that are not set on the **Configuration** tab of the Oracle B2B interface (see Chapter 17, "Configuring B2B System Parameters"). It also discusses how to set B2B properties by using the `configmbeanutil` utility.

The appendix contains the following topics:

- Section C.1, "Properties To Set in Fusion Middleware Control"
- Section C.2, "Using the configmbeanutil Utility"

C.1 Properties To Set in Fusion Middleware Control

The following properties can be set in Oracle Enterprise Manager Fusion Middleware Control. See *Oracle Fusion Middleware Administrator's Guide for Oracle SOA Suite and Oracle Business Process Management Suite* for how to set the properties.

> **Note:** Restarting the SOA Server is required for changes to B2B properties.

Table C–1   Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.addebMSHeaders`	Used to add the incoming ebMS message details of ACTION, SERVICE, SERVICE TYPE, CPAID, FROMROLE, and TOROLE to the ACTION_NAME header in AQ.
	By default only the ACTION is passed as part of ACTION_NAME.
`b2b.attachments.dir`	To specify an attachments directory location, set this property.
	If enabled, this property allows users to specify a directory into which all of the attachments will be written.
	Changes to this property require a server restart for the new value to take effect.
`b2b.certificatevalidation`	This property is used to enable or disable the validation of the certificate received on ebMS. For example, an expired certificate normally throws an error; however, if the property is set to `false`, it allows the certificate without throwing validation errors. The default value is `true`, indicating that by default, Oracle B2B validates the certificate.

Table C–1   (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
b2b.checkDuplicate	To check for duplicate messages, set this property to `true`.
	If this property is set to `false`, a check for a duplicate of the incoming business message is not performed.
	By default, Oracle B2B checks for duplicate messages based on the business message ID of the incoming message.
b2b.deploy.validation	To turn off validation during deployment, set this property to `false`.
	This is useful when deploying a large number of agreements where you are certain that the data is valid.
b2b.docPluginList	Using this property, document plugins are loaded as per the order specified in this property. For example,
	`b2b.docPluginList=Custom,EDI_X12,EDI_EDIFACT,HL7,RosettaNet`
	ensures that document plugins are loaded as per the specified order. If no value is provided for this property, then all document plugins from the system (`protocol.xml`) are loaded.
b2b.edi.enablePreprocess	Set this property to `true` to pre-process data to remove the trailing CRLF (new line characters) and/or unwanted CRLF that maybe present in file.
	The default value is `false`.
	Changes to this property require a server restart for the new value to take effect.
oracle.tip.b2b.edi.binaryDocType	This property is used by Oracle B2B to handle the BIN segment of EDI payloads. You need to specify a comma-separated list of document types that can have BIN segments as the following:
	`oracle.tip.b2b.edi.binaryDocType=841`
	Setting this property ensures that Oracle B2B processes the corrupt BIN segment content by treating the content as binary.
b2b.edi.identifyToTP	To identify the toTP from the TP information present in the incoming message, use this property.
	This property is used for inbound message processing. If it is set to `true`, the toTP is identified from the TP information present in the incoming message. The default value is `false`.
b2b.edi.ignoreValidation	To turn off the EDI envelope validation and ignore interchange validation errors for EDI messages, you can set comma separated envelope header with this property.
	For example:
	`b2b.edi.ignoreValidation=InterchangeSenderID, InterchangeReceiverID`
	or use the value `ALL` to turn off the complete envelope header validation.
	Note: If you add or update this flag, a server restart is not needed. However, if you remove the flag, the server must be restarted for the change to take effect.
b2b.encoding	This property can be used to specify encoding other than default UTF-8.

Table C–1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.errorsCumulativeReported`	To indicate whether reported errors are cumulative or not, set this property to `true` or `false`.
	Set this property to `true` (the default) to report errors in a cumulative fashion.
	If set to false, the error text and error description fields are not concatenated with the description starting on a new line.
	This property is most useful with EDI batching error messages.
`b2b.fa.inbound.validation` `b2b.fa.outbound.validation`	Use these properties to control inbound and outbound FA validation.
	To turn off the outbound FA validation, set the `b2b.fa.outbound.validation` property to `false`. The default value is `true`.
	To turn off the inbound FA validation, set the `b2b.fa.inbound.validation` property to `false`. The default value is `true`.
`b2b.FACorrelatedByInterchangeId`	To correlate inbound and outbound FAs correctly if the same control numbers are used between the same trading partners, set this property to `true`.
	By default Oracle B2B does not correlate inbound and outbound FAs correctly if the same control numbers are used between the same trading partners. Set this property to `true` so that Oracle B2B will correlate based on Control Number + (Interchange Sender ID + Interchange Receiver ID) for both Inbound and Outbound messages.
`b2b.FailedMessagesDirectory`	To change the location of the failed application message that is written to the file system, when `b2b.jmsRedeliveryLimit` is reached, set this property. The location can be changed from the default to another location by setting this property to another folder.
`b2b.HAInstance`	Set this property to `true` to ensure only one node in an High Availability (HA) cluster starts a polling thread for the configured File, FTP, or Email channels. Without this property being set, each node starts its own polling thread and could pick the same file multiple times.
	The default value of this property is `false`.
	A restart of all the nodes in the cluster is required to ensure each node works in the expected HA Mode.
`b2b.IDENTIFY_TP_BY_HOSTNAME`	In cases of TCP-based exchange protocols, such as MLLP and Generic TCP, where the host name is masked, to enable Oracle B2B to determine the host name of each message, set this `boolean` property to `true`.
	If set to `true`, then Oracle B2B retrieves the host name from the IP address of the host.
	If set to `false`, then Oracle B2B does not lookup the host name from the IP address.
	The default value for this property is `true`.

Table C–1   (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
b2b.TPBasedSeqTarget	To enable a channel name or a trading partner name as a sequence target, set this property to true.
	If a message has a sequence target as a part of the action name, then priority is be given to that message property.
	However, if there is no message level sequence target, and the property (b2b.TPBasedSeqTarget) is enabled, then the trading partner name is considered as the sequence target.
	To enable sequencing in case of:
	■ TCP: If sequencing is enabled (in trading partner facing channel) and TARGET is not specified, use the trading partner name as the TARGET to query the cache. This happens at the entry point in Oracle B2B.
	■ Non-TCP protocols: Because in non-TCP protocols, the channel is not bidirectional, use the TO-PARTY name as TARGET.
	■ Nonfabric: Use the sequencing flag in the channel of outbound listening internal channel.
b2b.hl7.ignoreValidation	To turn off the hl7 envelope validation, you can set comma separated envelope header with this property.
	For example:
	b2b.hl7.ignoreValidation = MessageReceivingFacility, MessageSendingApp
	or set it to ALL to turn off the complete envelope header validation.
b2b.inboundThreadCount b2b.inboundThreadSleepTime b2b.outboundThreadCount b2b.outboundThreadSleepTime b2b.defaultThreadCount b2b.defaultThreadSleepTime	To set the number of threads and thread sleep time to improve message processing, set these properties.
	The recommended values for b2b.inboundThreadCount and b2b.outboundThreadCount depend on your system. For a 2 GB computer, a setting of 3 to 5 is recommended.
	If Oracle B2B is running in a single thread mode (default), then a blocked outbound HTTP message could cause failure in Oracle B2B to process the subsequent message. The HTTP delivery has a default timeout of 60 seconds, and the timeout would push the message to error and unblock the processing. To avoid delays in processing, it is recommended that you increase the thread count property b2b.outboundThreadCount to 3 or 4 threads.
	The b2b.inboundThreadSleepTime and b2b.inboundThreadSleepTime properties put a thread to sleep after message processing. A setting between 10 and 1000 (milliseconds) is recommended.
b2b.jmsRedeliveryLimit	You can change the retry limit for reading messages from the JMS queue by setting this property, causing Oracle B2B to retry reading the message for the number of times specified there. The default retry limit is 5.
	When the limit count expires, the message and the header contents are written into the file system (the default location is the /tmp folder). Oracle B2B also sends an exception message to the JMS - B2B_IN_QUEUE, reporting the error and providing the location of the saved message.
	You can specify the location for failed messages by setting b2b.FailedMessagesDirectory.
b2b.listening.channel.restart.wait	Sometimes Oracle B2B creates multiple file monitor threads for a listening channel. Setting a longer interval using this property helps to avoid the race condition of threads during initialization.

Table C–1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.MaxTimeinAquiredState`	Set this property to avoid an issue in which the autostack handler does not resume processing of messages after restart.
	The time unit value of `b2b.MaxTimeinAquiredState` property is in minutes and 30 is the default value.
`b2b.mdsCache cache_size`	To set the Metadata Service (MDS) instance cache size, set this property.
	A ratio of 5:1 is recommended for the xmx-to-mdsCache values. For example, if the xmx size is 1024, maintain mdsCache at 200 MB.
`b2b.OutboundDispatchInterval`	To control the delay between every cycle of dispatch, use this property. Set this property to specify the amount of time to wait (in milliseconds) between dispatching of sequenced messages.
	If enabled, the property will control the delay between every cycle of dispatch. The property can be used along with Message Sequencing and Trading Partner Downtime Schedule features.
	When auto stack handler is used, then Oracle B2B retries the outbound failed messages in sequence. Once the endpoint is reached for delivery, all messages in the sequence will be eligible for delivery and this may cause an overload of message delivery at the endpoint. To reduce the load, this property can be used to set the interval between dispatch of messages in milliseconds.
`b2b.outboundOneErrorAllError`	To specify how errors are flagged, set this parameter to `true` or `false`.
	For outbound messages, `b2b.outboundOneErrorAllError` can be set to true or false. The default is false. If the property is set to true, then for outbound messages, even with a single message in error, all outbound batch messages are set to the error state. None of the batched messages are sent to the trading partner.
	When using the `b2b.outboundOneErrorAllError` property, inbound messages behave as if `b2b.outboundOneErrorAllError` is set to false; that is, if an error occurs during an inbound message process, then only that message is flagged with the error and other messages are passed. There is no option to flag every message as failed.
`b2b.payloadObfuscation`	To turn on payload obfuscation, set this property to true.
	See Section 1.4.1, "Payload Obfuscation," for more information.
`b2b.rowLockingForCorrelation`	To enable row level locking, set this property to `true`.
	When Oracle B2B receives an EDI FA message and an AS2 MDN message (Acknowledgment message) simultaneously, a race condition can occur which causes the outbound EDI message to remain in the MSG_WAIT_FA state (although FA was successfully processed).
	Use this row level locking parameter so that the original EDI message can be accessed sequentially when the correlated FA and Acknowledgment messages are processed concurrently. It avoids a racing condition when Oracle B2B updates the original message.
`b2b.setDynamicNameSpace`	To use EDI ecs and xsd files from Oracle B2B 10g version, set this property to `true`.
	When using EDI ecs and xsd files in Oracle B2B 11g which were used in Oracle B2B 10g, the XEngine may generate dynamic namespace for the translated xml. For example,
	`xmlns="NS_31CA8D0F33324F95A0BF15D85539C27E20060518215520"`
	To turn off dynamic namespace generation for inbound EDI messages, set this property to `false`.

Table C–1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.SyncAppDelivery`	When using callouts, to enable delivery of messages to the back-end message queue, set this property to `true`. To disable the message delivery, set this property to `false`. By default, this property is disabled (set to `false`). This property is case-sensitive.
`b2b.setisLargePayloadPropertyForSmallMsg`	By default Oracle B2B sets the `LARGE_PAYLOAD` header property to `true` only if the payload is large (as per configured size). In case the payload is small, the `LARGE_PAYLOAD` header property is not set. if you want to set the `LARGE_PAYLOAD` property for small payloads also, set the `b2b.setisLargePayloadPropertyForSmallMsg` property to `true`. By default, the value of the property is `false`.
`b2b.listening_channels.continue_reconnect`	By default, if the resource server, such as SFTP or JMS is down, Oracle B2B tries to reconnect only a specified number of times. However, if Oracle B2B cannot reconnect within the specified retries, it simply stops the particular listening channel. When this property is set to `true`, Oracle B2B makes continuous attempts to reconnect to resource server in downtime. In case of SFTP, the resource server is an SFTP server, and in case of JMS, the resource server is an JMS server. By default, the value of the property is `false`.
`b2b.reuseHttpConnections`	Set the value of this property to `true` to reuse HTTP connections. By default, the value of this property is `false`. When the default value is used, HTTP connections are opened and closed repeatedly when accessing the same URL consecutively. However, this property depends on other property values to take effect. If you call `HTTPClient.HTTPConnection.setCurrentProxy` when reusing an HTTP connection, side effect changes (such as reusing HTTP connections, closing the socket) are invoked only if the proxy settings have actually changed for the obtained HTTP connection instance. If invoking `setCurrentProxy` applies values currently set in the HTTP connection instance, the `setCurrentProxy` call is ignored. So you need to compare `HTTPClient.HTTPConnection.getProxyHost` and `HTTPConnection.getProxyPort` values against proposed new values, and call `HTTPConnection.setCurrentProxy` only if the values are different.
`b2b.deliverPingPongToBackend`	In ebMS, you can check the server state by using a set of messages called 'Ping and Pong'. Set the b2b.deliverPingPongToBackend property to `true` for enabling Oracle B2B to pass these messages to the backend applications on receipt of such messages. By default, the value of the property is `false`, which forces Oracle B2B to consume these Ping and Pong messages on receipt.
`b2b.trackInboundCtlNum`	By default, the value of this property is `true`, which controls the Oracle B2B behavior of saving the control numbers of inbound EDI messages in the database (the control numbers of inbound EDI message from a given partner will be inserted, or updated if the partner entry is already present). However, if set to `false`, then Oracle B2B does not attempt to insert the control numbers for inbound messages.

Table C–1   (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
b2b.SingleTransactionAtInbound	In the case of an inbound MLLP HA, if the server crashes after wire message has been committed to the database but before the event gets enqueued to Event Queue, it is perpetually stuck in the Sequence Manager table and is not processed. This blocks the inbound message flow in the sequencing case.
	Set the b2b.SingleTransactionAtInbound to true only in the case of MLLP HA to enable the JMS and database commit to take place in a single transaction. It is suitable only for the MLLP case where only one inbound message is received at a time.
	The default value of the property is false.
b2b.retainmsgid	When this value is set to true, @msgid is retained for the specific message ID, protocol message ID and collaboration ID during an Application Message resubmit.
	If this value is set to false, @msgid is retained for the message ID, but a new ID is generated for the protocol message ID and the collaboration ID.
	The default value for this property is false.
b2b.TPACache	Use this property to enable or disable Trading Partner Agreement (TPA) layer processing cache. By default, this property is not enabled.
	Limitation of the property: The TPA cache relies on agreement deployments to refresh the change in values. So, when this parameter is enabled, hot uptake of channel parameter changes does not work. It would require the agreements to be deployed. You need to restart the server for applying any changes to this parameter.
	The valid value for this parameter is local.
b2b.DispatcherCache	Use this property to enable or disable the Dispatcher processing cache. By default, this property is not enabled. You need to restart the server for applying any changes to this parameter.
	The valid value for this parameter is local.
b2b.enableDeliveryHelper	Use this property to spawn a separate thread for dispatching messages to the back-end application. Enable this property to improve the performance of the inbound dispatcher. You need to restart the server for applying any changes to this parameter.
	The valid values for this property are true or false.
b2b.sequencingFetchSize	Use this property to define the batch size for sequencing fetches. The Inbound or outbound Dispatcher attempt to pick *<batchsize>* number of messages for a given sequence target in a single fetch. The default value for this property is 20. You need to restart the server for applying any changes to this parameter.
	The valid value for this parameter is Integer.
Coherence system properties coherence.b2bDispatcherCache.size coherence.b2bDispatcherCache.expiry coherence.b2bTPACache.size coherence.b2bTPACache.expiry	These system properties help you to tune the cache size and expiry durations. The following are the default values of the system properties: coherence.b2bDispatcherCache.size - 20000 coherence.b2bDispatcherCache.expiry - 3 minutes coherence.b2bTPACache.size - 20000 coherence.b2bTPACache.expiry - 30 minutes

C.2 Using the configmbeanutil Utility

You can also use the configmbeanutil utility to set properties.

To use the configmbeanutil utility:

1. Set the *MW_HOME* environment variable to point to the Fusion Middleware installation directory. For example,

   ```
   setenv MW_HOME /scratch/$user/fmwhome
   ```

   Note: To access the Fusion Middleware directory, you must provide an mbean property file (`mbeanutil.properties`) that contains host, port, user, and password information. For example,

   ```
   host=myfmw.com
   port=7001
   user=weblogic
   password=mypwd
   ```

   ---

2. Set the *JAVA_HOME* environment variable. For example,

   ```
   setenv JAVA_HOME ${MW_HOME}/jdk160_14_R27.6.4-18
   ```

3. Add the Java bin directory to the *PATH* environment variable. For example,

   ```
   setenv PATH ${JAVA_HOME}/bin:${PATH}
   ```

Example C–1, Example C–2, Example C–3, and Example C–4 show uses for this utility.

Example C–1   To Print All Properties to the Console

```
java -cp $MW_HOME/AS11gR1SOA/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME/
wlserver_10.3/server/lib/wljmxclient.jar:$MW_HOME/modules/
glassfish.jaxb_1.2.0.0_2-1-7.jar oracle.tip.b2b.utility.ConfigMBeanUtility
```

Example C–2   To Add a Property

```
java -cp $MW_HOME/AS11gR1SOA/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME/
wlserver_10.3/server/lib/wljmxclient.jar:$MW_HOME/modules/
glassfish.jaxb_1.2.0.0_2-1-7.jar oracle.tip.b2b.utility.ConfigMBeanUtility add
b2b.test cool ok
```

Example C–3   To Update a Property

```
java -cp $MW_HOME/AS11gR1SOA/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME/
wlserver_10.3/server/lib/wljmxclient.jar:$MW_HOME/modules/
glassfish.jaxb_1.2.0.0_2-1-7.jar oracle.tip.b2b.utility.ConfigMBeanUtility
update b2b.test thru
```

Example C–4   To Remove a Property

```
java -cp $MW_HOME/AS11gR1SOA/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME/
wlserver_10.3/server/lib/wljmxclient.jar:$MW_HOME/modules/
glassfish.jaxb_1.2.0.0_2-1-7.jar oracle.tip.b2b.utility.ConfigMBeanUtility
remove b2b.test
```

D

Back-End Applications Interface

This appendix lists the message properties supported by Oracle B2B.

The appendix contains the following topics:

- Section D.1, "Mapping B2B IP_MESSAGE_TYPE to SCA Normalized Message Properties"
- Section D.2, "Normalized Message Properties"

D.1 Mapping B2B IP_MESSAGE_TYPE to SCA Normalized Message Properties

Table D–1 maps the `B2B IP_MESSAGE_TYPE` to SCA normalized message properties.

Table D–1   B2B IP_MESSAGE_TYPE to AS11 SCA Normalized Message Property Mapping

AQ (IP_MESSAGE_TYPE)	SCA	JMS
MSG_ID	b2b.messageId	MSG_ID
INREPLYTO_MSG_ID	b2b.replyToMessageId	INREPLYTO_MSG_ID
FROM_PARTY	b2b.fromTradingPartnerId	FROM_PARTY
-	b2b.fromTradingPartnerIdType	-
TO_PARTY_ID_VALUE	b2b.toTradingPartnerId	TO_PARTY
TO-PART_ID_TYPE	b2b.toTradingPartnerIdType	-
ACTION_NAME	-	ACTION_NAME
DOCTYPE_NAME	b2b.documentTypeName	DOCTYPE_NAME
DOCTYPE_REVISION	b2b.documentProtocolVersion	DOCTYPE_REVISION
-	b2b.documentProtocolName	-
-	b2b.documentDefinitionName	-
MSG_TYPE	b2b.messageType	MSG_TYPE
-	b2b.conversationId	-
PAYLOAD	body	-
ATTACHMENT	attachments	-

D.2 Normalized Message Properties

Header manipulation and propagation are key business integration messaging requirements. Like other SOA components such as Oracle BPEL Process Manager, Oracle Mediator, and Oracle JCA, Oracle B2B relies on header support to solve integration needs. For example, you can preserve a file name from the source directory to the target directory by propagating it through message headers.

Normalized messages have two parts, properties and payload. Typically, properties are name-value pairs of scalar types. To fit the existing complex headers into properties, properties are flattened into scalar types.

Manipulating headers in design time is simplified by using predetermined complex properties. In B2B, you can manipulate headers with reserved key words. However, some properties are dynamically generated based on your input. These definitions are not predetermined and hence cannot be accounted for in the list of predetermined property definitions. You cannot design header manipulation of the dynamic properties before they are defined. To address this limitation, you must generate all the necessary services (composite entry points) and references. This restriction applies to services that are expected to generate dynamic properties. After dynamic properties are generated, they are stored for each composite, and can be manipulated in the composite editor.

Figure D–1 shows the Properties tab of an Invoke activity, part of a BPEL process that includes a B2B binding component. Enter values and specify the input or output type for B2B properties on this dialog.

Figure D–1   Invoke Activity Showing B2B Normalized Message Properties
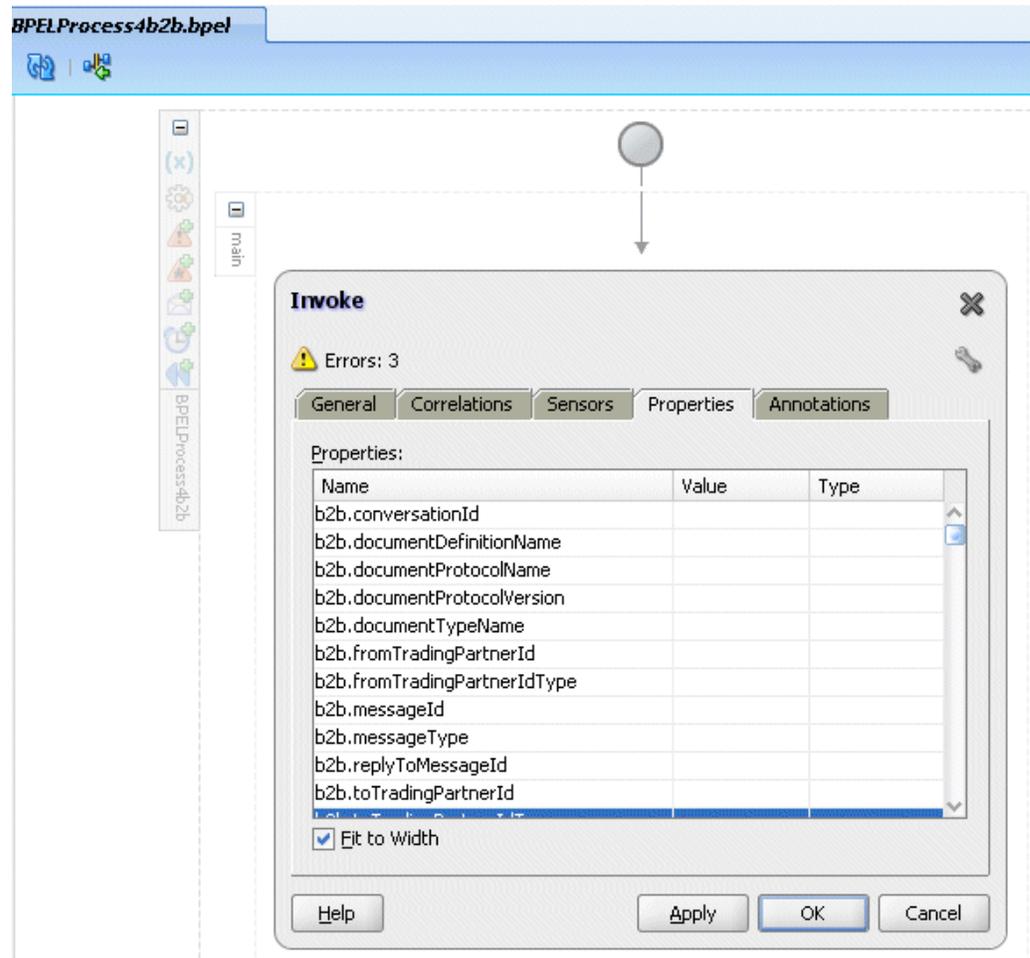
The image shows the Invoke activity, with a list of the properties for a normalized message in Oracle B2B. The properties are listed in the following table.

****

Table D–2 lists the predetermined properties of a normalized message for Oracle B2B.

Table D–2   Properties for Oracle B2B

Property Name	Propagable (Yes/No)	Direction (Inbound /Outbound)	Data Type	Range of Valid Values	Description
b2b.conversationId	No	Both	String	-	The ID used to relate the message to the message response
b2b.documentDefinitionName	No	Both	String	-	The document definition, for example, 850_def for an EDI X2 document
b2b.documentProtocolName	No	Both	String	-	The document protocol, for example, X12 for an EDI X12 document
b2b.documentProtocolVersion	No	Both	String	-	The document version, for example, 4010 for an EDI X12 document

Table D–2   (Cont.) Properties for Oracle B2B

Property Name	Propagable (Yes/No)	Direction (Inbound /Outbound)	Data Type	Range of Valid Values	Description
b2b.documentTypeName	No	Both	String	-	The document type, for example, 850 for an EDI X12 document
b2b.fromTradingPartnerId	No	Both	String	-	The trading partner identifier of the sender, for example, the name, such as Acme, or a DUNS number
b2b.fromTradingPartnerIdType	No	Both	String	-	The trading partner identifier type for the sender, for example, Name or DUNS
b2b.messageId	No	Both	String	-	A unique message ID, not directly related to ECID. (ECID information is stored in the B2B AppMessage table.)
b2b.messageType	No	Both	String	-	Message type values are: ■ Request = 1 ■ Response = 2 ■ Functional Ack = 9
b2b.replyToMessageId	No	Both	String	-	The message ID to which the sending message is replying
b2b.toTradingPartnerId	No	Both	String	-	The trading partner identifier of the receiver, for example, the name, such as Acme, or a DUNS number.
b2b.toTradingPartnerIdType	No	Both	String	-	The trading partner identifier type for the receiver, for example, Name or DUNS. If no value is found, the Name type is assumed.

E

Sequence Message Management

This appendix discusses the ANT commands and Public APIs used by Oracle B2B for managing message sequencing.

The appendix contains the following sections:

- Section E.1, "Why Do You Need to Manage Sequenced Messages"
- Section E.2, "ANT Commands and Public APIs"

E.1 Why Do You Need to Manage Sequenced Messages

Preserving the sequence of messages is important in the Oracle B2B message processing. This also needs robust management options. This includes the efficient reporting by using a dashboard and managing a particular target both within Oracle B2B as well as externally by other applications.

You need to have a good sequence message management framework for the following reasons:

- A better view of sequence messages (dashboard)
- Support for public API to enable external use
- Deleting messages based on target, state, message Id
- Viewing the pending messages based on target and state
- Option to pause and resume the message flow

E.2 ANT Commands and Public APIs

Oracle B2B provides ANT commandline utilities as well as public APIs to perform sequence message management.

This section lists the various management tasks that you can perform on sequenced messages.

List all the endpoints or targets with state:

This activity is like generating a dashboard of all the targets that explains all the states of the each target. This helps you to know the health of a target or endpoint (whether these are processing fine or these are in Error state).

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=listTargets
```

Public API

```
public java.util.List<java.lang.String> listSequenceTargets() throws
java.lang.Exception
```

List pending sequence messages based on state and endpoint:

This activity is required to know the number of messages on a given target or
endpoint with different states for the given target or endpoint. This information is
useful to know how many messages are piled up for the given target, incase, if there is
any issue with respect to the endpoint or target.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=report -Dstate=PROCESSED
-Dtarget=Pha01
```

Public API

```
public java.util.List<java.lang.String>
getSequenceMessagesByTargetAndState(java.lang.Stringtarget,
java.lang.String state)
throws java.lang.Exception
```

List pending sequence messages based on state:

This retrieves all the messages for a given state pertaining to all the endpoints or
targets.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=report -Dstate=PROCESSED
```

Public API

```
public java.util.List<java.lang.String> getSequenceMessagesByTarget
(java.lang.Stringtarget)throws java.lang.Exception
```

List pending sequence messages based on endpoint:

This lists all the pending messages with for a given endpoint or target.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=report -Dtarget=Pha01
```

Public API

```
public java.util.List<java.lang.String> getSequenceMessagesByTarget
(java.lang.String target) throws java.lang.Exception
```

Discard messages based on endpoint:

This deletes the messages based on a given target or endpoint. Essentially, this deletes
all the message pertaining to a target or endpoint from the Sequence Manager table.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discard -Dtarget=Pha01
```

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discard -Dtarget=Pha01
-Ddirection=OUTBOUND
```

Public API

```
public java.util.List<java.lang.String>
discardSequencedMessageByTarget(java.lang.String target) throws
java.lang.Exception
```

```
public java.util.List<java.lang.String>
discardSequencedMessageByTarget(java.lang.String target, java.lang.String
direction) throws java.lang.Exception
```

Discard messages based on state:

This deletes all the messages for a given state from all the endpoints or targets.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discard -Dstate=STACKED
```

Public API

```
public java.util.List<java.lang.String>
discardSequencedMessageByState(java.lang.String state) throws
java.lang.Exception
```

Discard messages based on state and endpoint:

This deletes all the messages for both a given state and endpoint or target.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discard -Dtarget=Pha01
-Dstate=PROCESSED
```

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discard -Dtarget=Pha01
-Dstate=PROCESSED -Ddirection=OUTBOUND
```

Public API

```
public java.util.List<java.lang.String>
discardSequencedMessageByStateAndTarget(java.lang.String target,
java.lang.String state) throws java.lang.Exception
```

```
public java.util.List<java.lang.String>
discardSequencedMessageByStateAndTarget(java.lang.String target,
java.lang.String state, java.lang.String direction) throws
java.lang.Exception
```

Discard messages based on message ID:

In case a particular message has some processing problems and should not be processed in the sequence manner. In this case, you can delete that message based on the message ID.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager
-Dmode=discard-Dmsgid=0AE851ED131B3D6103A00000152F97E9
```

Public API

```
public java.util.List<java.lang.String>
discardSequencedMessageByMessageId(java.lang.String msgId) throws
java.lang.Exception
```

Discard the first message of the endpoint:

This is a useful option to delete the first message of the endpoint or target. This may be required where the message is in stacked state and is not been able to be processed. Because this would be the first record in the stack, by deleting this, all the other messages can be processed successfully

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discardFirst
-Dtarget=Pha01

ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=discardFirst
-Dtarget=Pha01 -Ddirection=OUTBOUND
```

Public API

```
public java.util.List<java.lang.String>
discardFirstSequenceMessageByTarget(java.lang.String target)
throws java.lang.Exception

public java.util.List<java.lang.String>
discardFirstSequenceMessageByTarget(java.lang.String target,
java.lang.String direction)
throws java.lang.Exception
```

Re-process the messages based on message id (with and without payload)

Re-processing of the message is a useful option. Assume that due to some problem, a particular target or endpoint is down. After the target or endpoint starts functioning normally, the re-processing of the message enables you to send the message to the target or endpoint in sequence.

Public API

```
public boolean processSequenceMessageByMessageId(java.lang.String
messageId) throws java.lang.Exception
```

Pause / resume endpoint:

There may be a scenario, where the sequence messages need to be paused from processing. If the receiving application is down, then these messages can be paused till the application starts functioning. The pausing of messages is based on the given endpoint or target.

ANT Command

```
ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=pause -Dtarget=Pha01

ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=pause -Dtarget=Pha01
-Ddirection=OUTBOUND

ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=resume -Dtarget=Pha01

ant -f ant-b2b-util.xml b2bsequencemanager -Dmode=resume -Dtarget=Pha01
-Ddirection=OUTBOUND
```

Public API

```
public boolean pauseSequenceTarget(java.lang.String target)throws
java.lang.Exception

public boolean pauseSequenceTarget(java.lang.String target,
java.lang.String direction) throws java.lang.Exception

public boolean resumeSequenceTarget(java.lang.String target) throws
java.lang.Exception

public boolean resumeSequenceTarget(java.lang.String target,
java.lang.String direction)throws java.lang.Exception
```

Get the state for given Target / endpoint:

This retrieves the state of a given target or endpoint.

Public API

```
public String getStateByTarget(java.lang.String target)throws
java.lang.Exception
```

```
public String getStateByTarget(java.lang.String target, java.lang.String
direction)throws java.lang.Exception
```

F

Setting Up B2B Communication By Using Remote JNDI Queue

In Oracle B2B, you can configure third-party JMS providers to exchange messages. For this, you can configure a remote JNDI queue so that B2B can publish messages to a queue or topic on a remote server, or a queue or topic on a third-party JMS server.

This appendix contains the following topic:

- Section F.1, "Configuring B2B Communication By Using Foreign JNDI"

F.1 Configuring B2B Communication By Using Foreign JNDI

You can bind any remote server JNDI context to a Weblogic server by using the Foreign JNDI Provider option in the Weblogic Server console.

This procedure has two steps:

1. Creating the target JMS server, module, and queue.

2. Setting up an environment for B2B to write to the JMS queue.

F.1.1 Creating a Target JMS Server, Module, and Queue

To set up a target JMS server, module, and queue in a computer (say Host1):

1. Install Weblogic server.

2. Install Admin server and SOA server.

3. Create a test JMS server by selecting the SOA server as the target as shown in Figure F–1.

Figure F–1   Creating a Test JMS Server
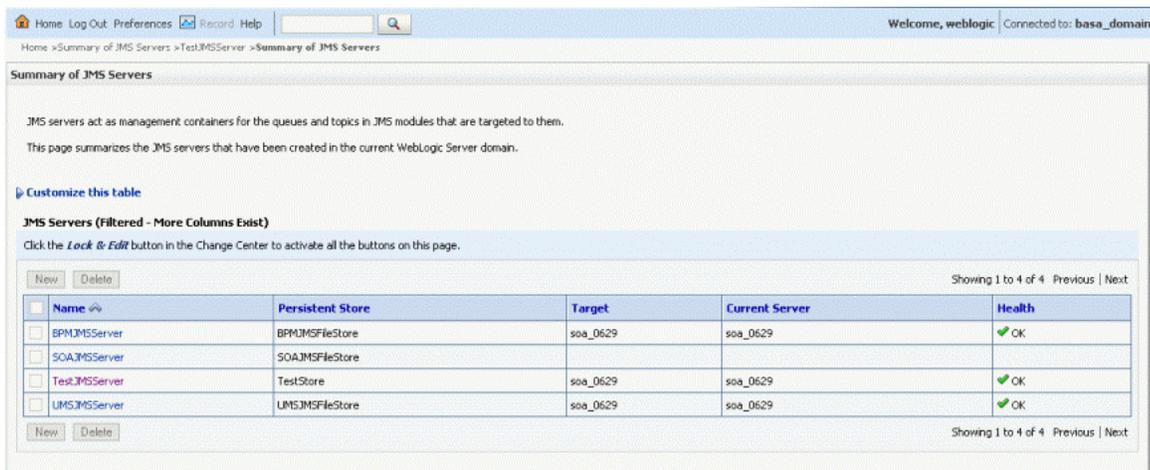
The figure displays a test JMS server listing in a Weblogic server console.

****

4. Create a test JMS module as shown in Figure F–2.

Figure F–2   Creating a Test JMS Module
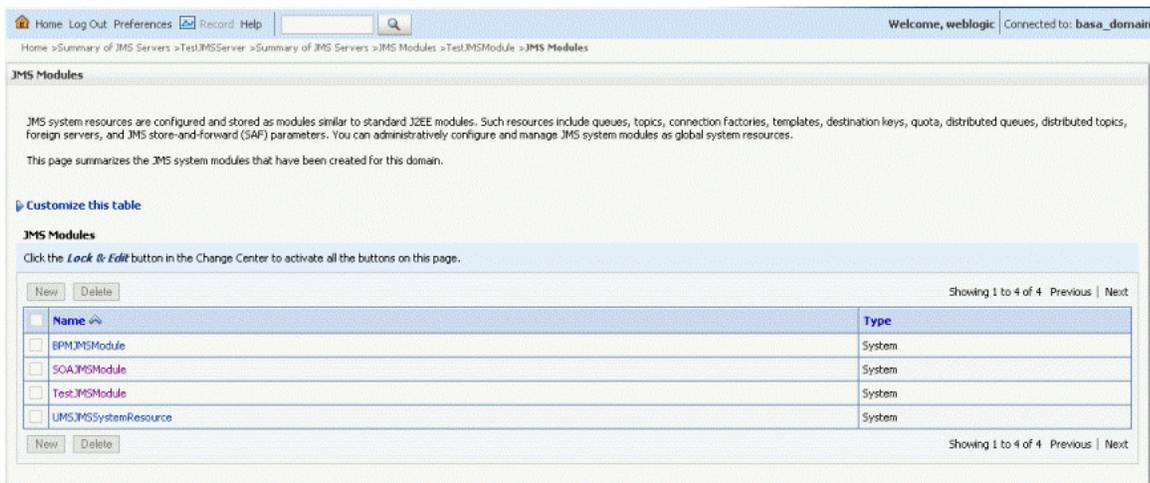
The figure displays a test JMS module listing in a Weblogic server console.

****

5. Launch the test JMS module that you created and create a connection factory with target as SOA server.

6. Create a JMS Queue and with the target as the created test JMS server as shown in Figure F–3.

Figure F–3   Creating a Test JMS Queue
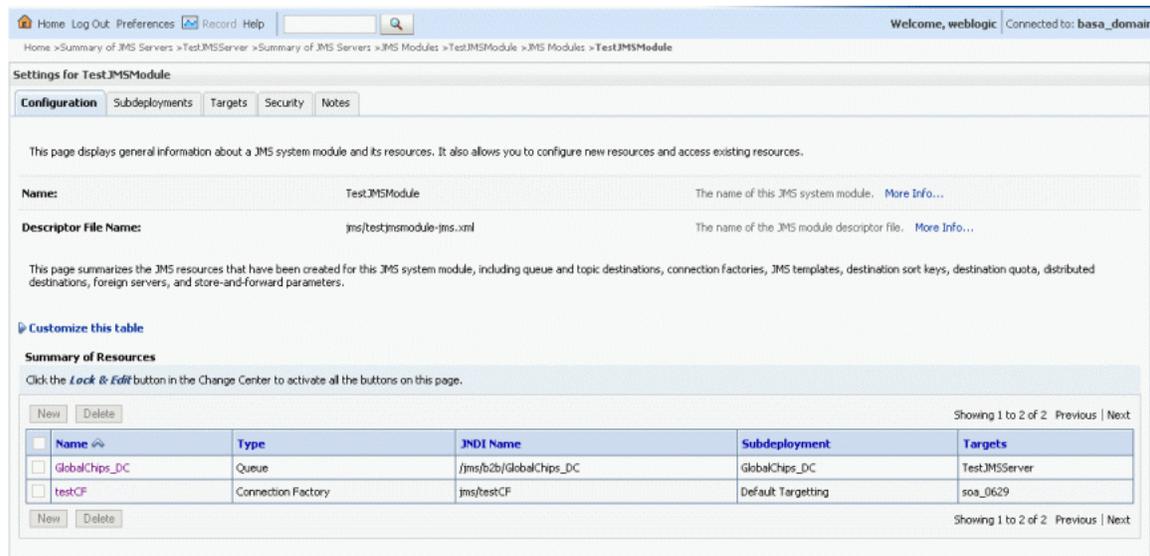
The figure displays a test JMS queue listing in a Weblogic server console.

****

F.1.2 Setting Up an Environment to Connect to the JMS queue

Once you have created a test JMS server, module, and queue on Host1, you need to set up another computer (say Host2) to enable B2B to write to the JMS queue on Host1.

To set up Host2:

1. Install Weblogic server and SOA server.

    > **Note:**   Ensure that all the server names and the domain names for Host2 are different from the ones that you have specified for Host1.

2. Log on to the Weblogic server console.

3. Navigate to **Foreign JNDI Providers** under **Services** on the left-hand navigation panel.

4. In the Foreign JNDI Providers page, click **New** to add a test JNDI provider called `TestJNDIProvider`.

5.   Click the newly added JNDI provider name and under the General tab, enter values corresponding to the Weblogic server credentials of Host1.

    For example, specify `weblogic.jndi.WLInitialContextFactory` as the Initial Context Factory name and `t3://<host:port>` for the Provider URL as shown in Figure F–4.

Figure F–4   Configuring Connection to Remote Server
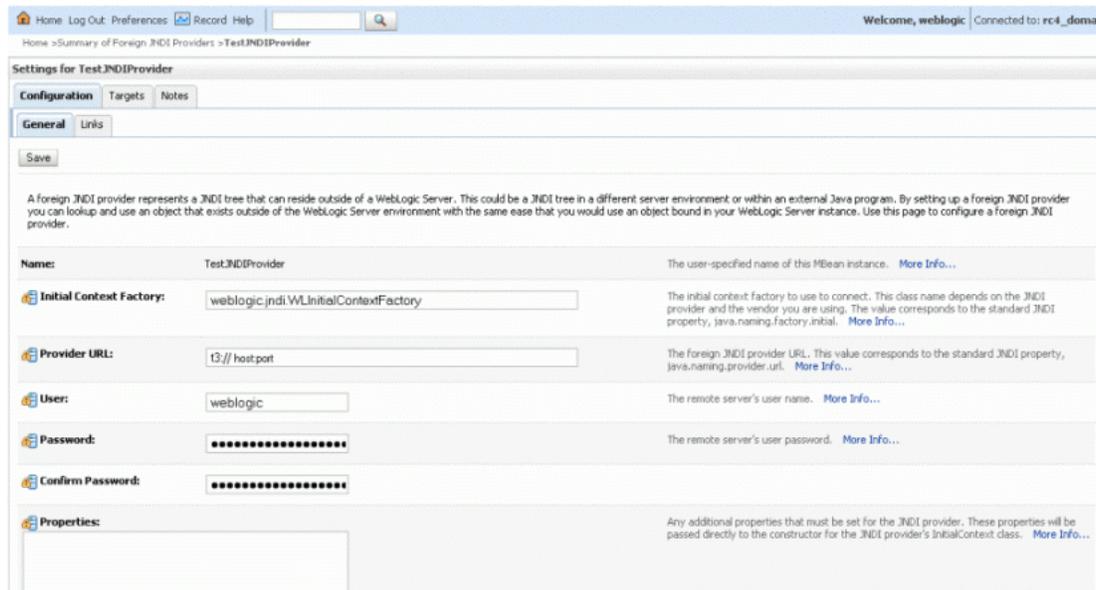
The figure displays how to set up the connection to the remote server.

****

6. Save the changes to complete setting up connection to the remote server.

7. Click the **Links** tab.

8. Create a link to point to the connection factory and to the queue on the JMS server on Host1 by specifying the following:

    - Name: Any user-defined name

    - Local JNDI Name: Any user-defined JNDI name

    - Remote JNDI Name: JNDI names of the connection factory and queue on the JMS server on Host1

      Figure F–5 shows the Foreign JNDI Links table displaying a list of foreign JNDI links.

Figure F–5   Foreign JNDI Links
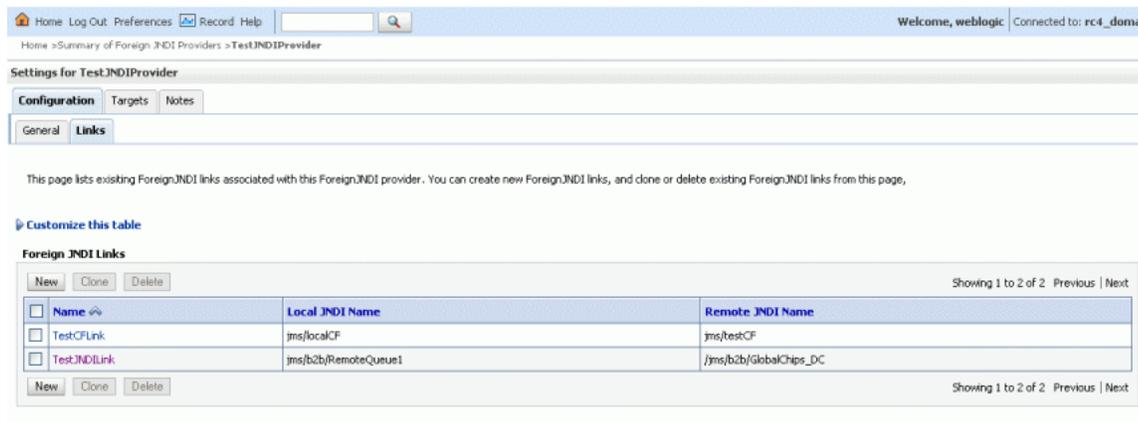
The figure displays a Weblogic server console with a list of foreign JNDI links.

****

9. In the B2B user interface, create a JMS channel by specifying the local JNDI names (created to point to the remote JNDI queue and remote connection factory) in the Destination name and Connection factory fields as shown in Figure F–6.

Figure F–6   Creating a JMS Channel in the B2B User Interface
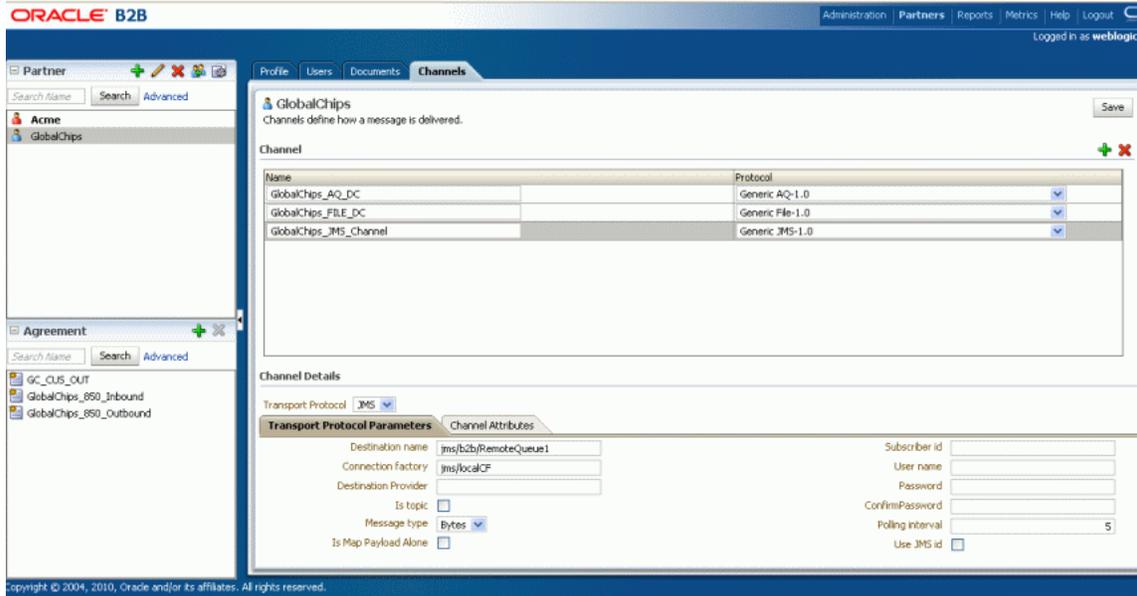
The figure displays the Oracle B2B UI where an JMS channel is created by specifying the destination name and the connection factory name.

****

G

Exception Handling

Oracle B2B handles exceptions for inbound and outbound messages. This appendix describes the exception handling, error messages, and structures for Oracle B2B.

The appendix contains the following topics:

- Section G.1, "Inbound Messages"
- Section G.2, "Outbound Messages"
- Section G.3, "Using a JMS Queue for Error Message Delivery"
- Section G.4, "Using a Custom Exception Queue for Error Message Delivery"
- Section G.5, "Inbound Exception Handling Scenarios"
- Section G.6, "Exception Payload Definition"

Note: Oracle B2B does not support the various error codes specified by the ebMS 2.0 specification. For exception messages, Oracle B2B sets the error code to "Unknown". The expected error codes are:

ValueNotRecognized

NotSupported

Inconsistent

OtherXml

DeliveryFailure

TimeToLiveExpired

SecurityFailure

MimeProblem

Unknown

G.1 Inbound Messages

This section describes the following inbound message types:

- Request or Response Messages
- Acknowledgment Messages
- Exception Messages

G.1.1 Request or Response Messages

For an incoming request, response, or functional acknowledgment message that results in an exception, the following actions occur when you use the default error handling settings:

- An exception message is sent to the application.

  The exception message is enqueued to `IP_IN_QUEUE` and has the recipient name `b2berroruser`. The enqueued exception is based on `ipException.xsd` and contains information such as the error message (`errorText` has a short description and `errorDescription` has a longer description) and the error code.

- An exception message is sent to the trading partner, if mandated by the exchange specification.

  The exception message is sent back to the trading partner only if there is enough information to identify the outgoing trading partner agreement. For this purpose, the flag `B2BHeader.sendException` is used. The flag is set to true when enough information is extracted from the incoming message to send the exception message to the trading partner.

- Oracle B2B catches exceptions thrown by exchange or document layers.

  If the `B2Bheader.sendException` flag is set to `true`, the outgoing trading partner agreement is processed and an exception message is sent to the trading partner.

G.1.1.1 Inbound ebMS, AS1, and AS2 Messages

If the following types of failure occur while an incoming message is processing, then the receiving trading partner sends a negative acknowledgment to the sender.

- Decryption fails

- Verification fails

- Agreement is not found

- Document identification fails

- Document validation fails (and so on)

The negative acknowledgment message has the reference for the original (request) message details to correlate at the sender side.

G.1.2 Acknowledgment Messages

For an incoming acknowledgment message that results in an exception, the following actions occur when you use the default error handling settings:

- An exception message is sent to the application.

  The exception message is enqueued to `IP_IN_QUEUE` and has the recipient name `b2berroruser`. The enqueued exception is based on `ipException.xsd` and contains information such as error text and error code.

- No exception message is sent back to the trading partner.

G.1.3 Exception Messages

For an incoming exception message, the following actions occur when you use the default error handling settings:

- The original message is updated so that it is in an errored state. The incoming exception is processed and delivered to the application normally.

- If the incoming exception message itself results in an exception, an exception message is sent to the application.

  The exception message is enqueued to `IP_IN_QUEUE` and has the recipient name `b2berroruser`. The enqueued exception is based on `ipException.xsd` and contains information such as error text and error code. No exception message is sent back to the trading partner in this case.

Exceptions can be delivered to default queues (`B2B_IN_QUEUE` or `IP_IN_QUEUE`) or custom JMS queues configured for exception messages. See Section G.4, "Using a Custom Exception Queue for Error Message Delivery" for more information.

G.2 Outbound Messages

If an exception occurs while an outbound message is being sent (for example, if the trading partner identification fails), then an exception message is sent to the application. When you use the default error handling settings, the exception message is enqueued to `IP_IN_QUEUE` and has the recipient name `b2berroruser`. The enqueued exception is based on `ipException.xsd` and contains information such as error text and error code.

If an exception occurs during Oracle B2B startup, then an exception message is enqueued to `IP_IN_QUEUE` and has the recipient name `b2berroruser`. The enqueued exception is based on `ipException.xsd` and contains information such as error text and error code. The correlation ID is not populated in this case.

Note the following:

- When the exception message is sent back to the application, the document type is `Exception` instead of the original message document type.

- When the exception message is sent back to the application, `inReplyToMessageId` is populated with the correlation ID value.

- For inbound exception handling, a business message is always created and populated with the available information. It also points to the corresponding wire message. The wire message is updated so that it is in an errored state. For the outbound direction, only the business message is updated, because the wire message does not exist. However, if a transmission failure occurs, then the wire message table does have an entry.

- The error reports are updated to show only business messages; a business message is always created in the inbound and outbound directions.

G.3 Using a JMS Queue for Error Message Delivery

The default settings, as described in Section G.1 and Section G.2, use an AQ queue, `IP_IN_QUEUE`, as the exception queue. You can configure B2B to use a JMS queue by setting the **Use JMS Queue as default** parameter to true on the **Configuration** tab. The JMS queue, `B2B_IN_QUEUE`, becomes the default exception queue unless you have configured a custom JMS exception queue and selected it as the value for the **Exception Queue** parameter (see Section G.4, "Using a Custom Exception Queue for Error Message Delivery.") In general, B2B sends inbound messages to `B2B_IN_QUEUE` and polls on `B2B_OUT_QUEUE` for outbound messages.

Because JMS queues cannot use b2berroruser as the consumer, a JMS message property is used to filter exception messages for error handling. Specifically, when the

MSG_TYPE value equals 3 (MSG_TYPE='3'), all exception messages are received by the JMS receiver. (For successful messages, MSG_TYPE='1'.) All JMS message properties are of type string.

See Table 17–1, " Configuration Settings" for more information on the **Use JMS Queue as default** parameter.

G.4 Using a Custom Exception Queue for Error Message Delivery

You can create custom JMS exception queues by configuring JMS internal delivery channels (JMS queues or topics) for the host trading partner on the **Partners** > **Channels** tab, as shown in Figure G–1.

Figure G–1   Creating a Custom Exception Queue
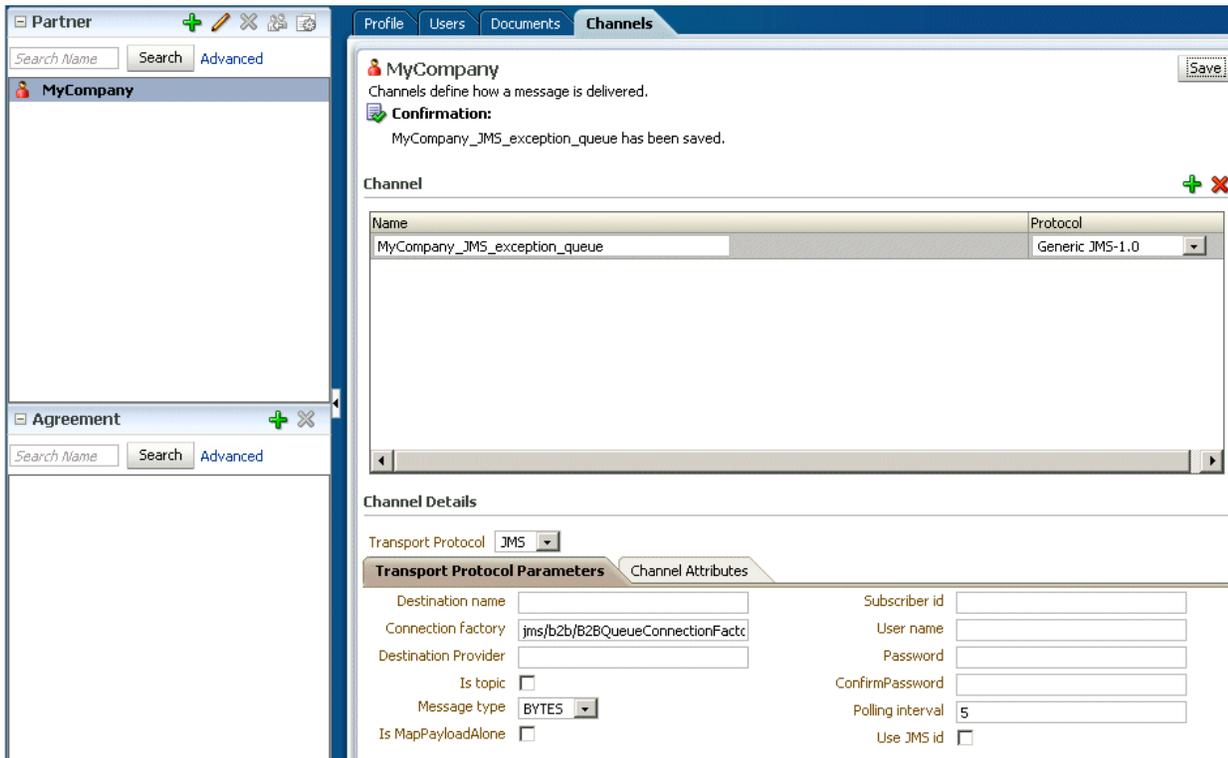
The image shows the Channels tab with the Channel Name input field and the Protocol dropdown list. The Channel Details sections shows the JMS transport protocol and the following fields on the Transport Protocol Parameters tab: Destination Name, Connection Factory, Destination Provider, Is topic (check box), Message type (dropdown list with BYTES displayed), IsMapPayloadAlone (check box), Subscriber id, User name, Password, ConfirmPassword, Polling Interval, and Use JMS id (check box).

****

Then select the queue from the **Exception Queue** parameter on the **Configuration** tab. The **Exception Queue** dropdown lists all JMS internal delivery channels from the host trading partner.

A null default value for this parameter means that the JMS queue, B2B_IN_QUEUE, is the exception queue if **Use JMS Queue as default** is set to true, and that the AQ

queue, `IP_IN_QUEUE`, is the exception queue if **Use JMS Queue as default** is set to false.

If B2B fails to deliver an exception message to the selected custom exception queue, then the exception message is sent to the default internal delivery channel.

See Table 17–1, " Configuration Settings" for more information on the **Exception Queue** parameter.

G.5 Inbound Exception Handling Scenarios

Table G–1 describes inbound exception handling scenarios.

Table G–1 Inbound Exception Handling Scenarios

If an exception occurs because. . .	Then Oracle B2B does . . .
The identification of the exchange fails or the exchange is not supported	- Notifies the middleware - Updates the wire message as in an errored state - Creates a business message in an errored state for the wire message - Sends a transport error message to the trading partner if the sendException flag is set in the exchange layer
Message unpacking fails	- Notifies the middleware - Updates the wire message as in an errored state - Creates a business message in an errored state for the wire message
Incoming message decoding fails	- Notifies the middleware - Updates the wire message as in an errored state - Creates a business message in an errored state for the wire message - Sends an exception message to the trading partner, if the `sendException` flag is set in the exchange layer
The message is duplicated	- Notifies the middleware - Updates the wire message as a duplicated message error - Creates a business message as a duplicated message error for the wire message

Table G–1   (Cont.) Inbound Exception Handling Scenarios

If an exception occurs because. . .	Then Oracle B2B does . . .
Document identification fails	■ Notifies the middleware ■ Updates the wire message as in an errored state ■ Creates a business message in an errored state for the wire message ■ Sends an exception message to the trading partner, if the `sendException` flag is set in the exchange layer
Incoming trading partner agreement processing fails	■ Notifies the middleware ■ Updates the wire message as in an errored state ■ Creates a business message in an errored state for the wire message ■ Sends an exception message to the trading partner, if the `sendException` flag is set in the exchange layer
Incoming document processing fails	■ Notifies the middleware ■ Updates the wire message as in an errored state ■ Creates a business message in an errored state for the wire message ■ Sends an exception message to the trading partner, if the `sendException` flag is set in the exchange layer

Note the following:

- The exception is sent back to the trading partner only for RosettaNet exchanges. For other exchanges, a failure is reported as mandated in the respective specifications. For example, for an ebMS exchange, an acknowledgment is sent along with the error list that is defined. For an AS2 exchange, the acknowledgment is sent indicating an error, without exception details.

- An exception is sent back to the trading partner for all message types except acknowledgments.

G.6 Exception Payload Definition

Example G–1 shows the definition for the exception payload, `ipException.xsd`.

Example G–1   Exception Payload Definition

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://integration.oracle.com/B2B/Exception"
targetNamespace="http://integration.oracle.com/B2B/Exception">

  <xs:element name="Exception">
    <!--xs:complexType name="Exception"-->
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="correlationId"/>
        <xs:element ref="b2bMessageId"/>
        <xs:element ref="errorCode"/>
        <xs:element ref="errorText"/>
        <xs:element ref="errorDescription"/>
        <xs:element ref="errorSeverity"/>
        <xs:element ref="errorDetails" minOccurs="0" />
```

```
        </xs:sequence>
      </xs:complexType>
  </xs:element>
  <xs:element name="correlationId" type="xs:string"/>
  <xs:element name="b2bMessageId" type="xs:string"/>
  <xs:element name="errorCode" type="xs:string"/>
  <xs:element name="errorText" type="xs:string"/>
  <xs:element name="errorDescription" type="xs:string"/>
  <xs:element name="errorSeverity" type="xs:string"/>
  <xs:element name="errorDetails">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="parameter" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="parameter">
    <xs:complexType>
      <xs:attribute name="name" type="xs:string" use="required" />
      <xs:attribute name="value" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

H

Self Service Utility Protocols, Identifications, Security Specifications, and Parameters

This appendix gives the values for protocols, identifications, security specifications, and parameters used in the `selfservice` utility.

The appendix contains the following topics:

- Section H.1, "Protocols"
- Section H.2, "Identifications"
- Section H.3, "Security Specifications"
- Section H.4, "Exchange Protocols Parameter Values"
- Section H.5, "Transport Protocols Parameter Values"
- Section H.6, "Document Protocol Parameter Values"

H.1 Protocols

Table H–1   Document Protocols

Document Protocol Name
EDI_X12
EDI_EDIFACT
HL7
RosettaNet
Custom

Table H–2   Exchange Protocols

Exchange Protocol Name
AS2
MLLP
ebMS2
ebMS1
RNIF20
RNIF11
AS1

Table H–2   (Cont.) Exchange Protocols

Exchange Protocol Name
Generic-File
Generic-AQ
Generic-FTP
Generic-SFTP
Generic-JMS
Generic-HTTP
Generic-Email

Table H–3   Transport Protocols

Transport Protocol Name
HTTP
File
AQ
JMS
FTP
SFTP
AS1
TCP
Email

H.2 Identifications

Any of the following columns value can be used as Identification name in Self-Service. Name Identifier will be created by Self-Service using the Trading Partner name.

Table H–4   Identifications

Identification Name	Identification ID
Generic Identifier	Generic
DUNS	DUNS
ebMS Identifier	ebMS
AS2 Identifier	AS2
MLLP ID	MLLP
AS1 Identifier	AS1
EDI Interchange ID	InterchangeID
EDI Interchange ID Qualifier	InterchangeIDQualifier
EDI Group ID	GroupID
EDI Interchange Internal ID	InterchangeInternalID
EDI Interchange Internal Sub ID	InterchangeInternalSubID
EDI Group ID Qualifier	GroupIDQualifier

Table H–4   (Cont.) Identifications

Identification Name	Identification ID
HL7 Message Application ID	Message-ApplicationID
HL7 Message Application Universal ID	Message-ApplicationUniversalID
HL7 Message Application Universal ID Type	Message-ApplicationUniversalIDType
HL7 Message Facility ID	Message-FacilityID
HL7 Message Facility Universal ID	Message-FacilityUniversalID
HL7 Message Facility Universal ID Type	Message-FacilityUniversalIDType
HL7 Batch Application ID	Batch-ApplicationID
HL7 Batch Application Universal ID	Batch-ApplicationUniversalID
HL7 Batch Application Universal ID Type	Batch-ApplicationUniversalIDType
HL7 Batch Facility ID	Batch-FacilityID
HL7 Batch Facility Universal ID	Batch-FacilityUniversalID
HL7 Batch Facility Universal ID Type	Batch-FacilityUniversalIDType
HL7 File Application ID	File-ApplicationID
HL7 File Application Universal ID	File-ApplicationUniversalID
HL7 File Application Universal ID Type	File-ApplicationUniversalIDType
HL7 File Facility ID	File-FacilityID
HL7 File Facility Universal ID	File-FacilityUniversalID
HL7 File Facility Universal ID Type	File-FacilityUniversalIDType

H.3 Security Specifications

Any of the following columns value can be used as Identification name in Self-Service.

Table H–5   Security Specifications

Security Specifications Name	Security Specifications ID
SMIME 3.0 with MD5 - RSA	SMIME-3_0-MD5-RSA
SMIME 3.0 with SHA1 - RSA	SMIME-3_0-SHA-RSA
SMIME 2.0 with MD5 - RSA	SMIME-2_0-MD5-RSA
SMIME 2.0 with SHA1 - RSA	SMIME-2_0-SHA-RSA
XMLDSIG with SHA1 - DSA	XMLDSIG-1_0-SHA-DSA
XMLDSIG with SHA1 - RSA	XMLDSIG-1_0-SHA-RSA
SMIME 3.0 with DES	SMIME-3_0-DES
SMIME 3.0 With 3DES	SMIME-3_0-3DES
SMIME 3.0 with RC2-40	SMIME-3_0-RC2-40
SMIME 3.0 with RC2-64	SMIME-3_0-RC2-64
SMIME 3.0 with RC2-128	SMIME-3_0-RC2-128
SMIME 2.0 with DES	SMIME-2_0-DES
SMIME 2.0 With 3DES	SMIME-2_0-3DES

Table H–5 (Cont.) Security Specifications

Security Specifications Name	Security Specifications ID
SMIME 2.0 with RC2-40	SMIME-2_0-RC2-40
SMIME 2.0 with RC2-64	SMIME-2_0-RC2-64
SMIME 2.0 with RC2-128	SMIME-2_0-RC2-128
XMLENC with 3DES - RSA-v1.5	XMLENC-1_0-3DES-RSA-V1_5
XMLENC with AES-128 - RSA-OAEP	XMLENC-1_0-AES128-RSA-OAEP
XMLENC with AES-192 - RSA-OAEP	XMLENC-1_0-AES192-RSA-OAEP
XMLENC with AES-256 - RSA-OAEP	XMLENC-1_0-AES256-RSA-OAEP

H.4 Exchange Protocols Parameter Values

Table H–6   Exchange Protocols Parameter Values

Exchange	Parameter Value Name	Domain	Required
AS2	Receipt-Delivery-Option	String, any URL	No
	Signed-And-Compressed	Boolean, false (default), true	No
MLLP	ImmediateACK	String, any of these values - None(default), Default, Simple, Custom	No
	ImmediateACK-Custom-File	Absolute File Path	No
	ImmediateACK-mapAckControlID	Boolean, false (default), true	No
	ImmediateACK-MapImmTriggerEvt	Boolean, false (default), true	No
	DiscardHL7ACK	String, any of these values - None (default), AA, AE, AR, CA, CE, CR	No
	Start-Block-Char	Hexadecimal, 0x0B(default)	No
	End-Block-Char	Hexadecimal, 0x1C(default)	No
	Carriage-Return-Char	Hexadecimal, 0x0D(default)	No
	Identify-TP-by-delivery-channel	Boolean, false (default), true	No
ebMS2.0	Duplicate-Elimination	Boolean, false (default), true	No
	messageOrderSemantics	String	No
	PersistDuration	String	No
	SendPartyTypeAndValue	Boolean, false (default), true	No
ebMS1.0	Duplicate-Elimination	Boolean, false (default), true	No
	SendPartyTypeAndValue	Boolean, false (default), true	No
AS1	Signed-And-Compressed	Boolean, false (default), true	No

H.5 Transport Protocols Parameter Values

Table H–7

Transport	Parameter Name Value	Domain	Required
HTTP	url	String, any URL	Yes
	user	String	No
	password	String	No
	additional_headers	String	No
	use_proxy	Boolean, false (default), true	No

Table H–7   (Cont.)

Transport	Parameter Name Value	Domain	Required
File	polling_interval	Integer, 5(default)	No
	folder	String	Yes
	filename_format	String	No
AQ	sid	String, orcl(default)	No
	port	Integer, 1521(default)	No
	schema	String	No
	queue_name	String	No
	password	String	No
	host	String	No
	polling_interval	Integer	No
	recipient	String	No
	consumer	String	No
	datasource	String, either `datasource` or `jdbc(host,sid,port,schema)`	No
JMS	queue_name	String	No
	jndi_connection_factory_location	String	No
	Is_topic	Boolean, false (default), true	No
	is_map_message	String, Any of these values BYTES,TEXT,MAP	No
	is_map_payload_alone	Boolean, false (default), true	No
	Subscriber_ID	String	No
	user	String	No
	password	String	No
	polling_interval	Integer, 5(default)	No
FTP	host	String	Yes
	polling_interval	Integer, 5(default)	No
	folder	String	Yes
	user	String	Yes
	password	String	No
	channel_mask	String, None (default)	No
	cipher_suites	String	No
	control_port	Integer	No
	data_port	Integer	No
	use_proxy	Boolean, false (default), true	No
	filename_format	String	No
	sourcefile_encoding	String	No

Table H–7   (Cont.)

Transport	Parameter Name Value	Domain	Required
SFTP	host	String	Yes
	port	Integer	No
	polling_interval	Integer, 5(default)	No
	folder	String	Yes
	user	String	Yes
	password	String	No
	private_key	String	No
	pass_phrase	String	No
	use_proxy	Boolean, false (default), true	No
	filename_format	String	No
TCP	sockettype	String, Either of Server, Client (default)	No
	host	String	No
	port	Integer	No
	PermanentConnectionType	Boolean, false (default), true	No
	Sequencing	Boolean, false (default), true	No
	pollinterval	Integer, 10(default)	No
	timeout	Integer, 300(default)	No
Email	host	String	Yes
	password	String	No
	user	String	No
	polling_interval	Integer	No
	content-type	String	No
	send_as_attachment	String	No
	folder	String	No
	email-id	String	Yes
	subject	String	No
	server	String, either of IMAP (default), PoP3	Yes

H.6 Document Protocol Parameter Values

This section conatins the following tables:

- Table H–8, " EDI_X12 Document Protocol Parameter Values"
- Table H–9, " EDI_EDIFACT Document Protocol Parameter Values"
- Table H–10, " HL7 Document Protocol Parameter Values"
- Table H–11, " RosettaNet Document Protocol Parameter Values"
- Table H–12, " Custom Document Protocol Parameter Values"
- Table H–13, " Common Parameter Values"
- Table H–14, " AgrDocType Parameter Values"

Table H–8   EDI_X12 Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Document Protocol Version	InterchangeAuthorizationInfoQual	String, 00(default)	No
	InterchangeAuthorizationInfo	String	No
	InterchangeSecurityInfoQual	String, 00(default)	No
	InterchangeSecurityInfo	String	No
	InterchangeDate	Date, #SystemDate(YYMMDD)#(default)	No
	InterchangeTime	Time, #SystemTime(HHMM)#(default)	No
	InterchangeControlStandard_RepeatingSeparator	String, U (default)	No
	InterchangeControlVersion	String, 00401(default)	No
	InterchangeAckRequested	String, 0(default)	No
	InterchangeUsageIndicator	String, P (default)	No
	InterchangeECSFileBlob	Absolute file path of ecs	No
	GroupDate	Date, #SystemDate(CCYYMMDD)# (default)	No
	GroupTime	Time, #SystemTime(HHMM)#(default)	No
	GroupAgencyCode	String, X (default)	No
	GroupVersionNumber	String	No
	GroupECSFileBlob	Absolute file path of ecs	No
	SegmentDelimiter	Hexadecimal, 0x7e(default)	No
	ElementDelimiter	Hexadecimal, 0x2a(default)	No
	SubelementDelimiter	Hexadecimal, 0x5c (default)	No
	DecimalSeparator	Hexadecimal, 0x2e (default)	No
	ReplacementChar	Hexadecimal, 0x7c (default)	No
	RepeatingSeparator	Hexadecimal, 0x5e (default)	No
Document Type	GroupID	String	Yes
	TransactionID	String	Yes
	TransactionImplementationReference	String	No
	TransactionPurposeCode	String	No
Document Definition	TransactionECSFileBlob	Absolute file path of ecs	Yes
	DocumentRoutingID	String	No
	Common Parameter Values (see Table H–13, " Common Parameter Values")		

Table H–9   EDI_EDIFACT Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Document Protocol Version	UNACreation	String, any of these values - always, never, delimiter-based	No
	CharSet	String, UNOB(default)	No
	InterchangeControlVersion	String, 1(default)	No
	InterchangeServiceCodeList	String	No
	InterchangeCharacterEncoding	String	No
	InterchangeDate	Date, #SystemDate(YYMMDD)# (default)	No
	InterchangeTime	Date, #SystemTime(HHMM)# (default)	No
	InterchangeRecipientRefPassword	String	No
	InterchangeRecipientRefPasswordQual	String	No
	InterchangeApplicationRef	String	No
	InterchangeProcessingPriority	String	No
	InterchangeAgreementIdentifier	String	No
	InterchangeTestIndicator	String	No
	InterchangeECSFileBlob	Absolute file path of ecs	No
	FunctionalGroupCreation	String	No
	GroupDate	Date, #SystemDate(YYMMDD)# (default)	No
	GroupTime	Time, #SystemTime(HHMM)# (default)	No
	GroupAgencyCode	String, UN (default)	No
	GroupAssociationCode	String	No
	GroupApplicationPassword	String	No
	GroupECSFileBlob	Absolute file path of ecs	No
	SegmentDelimiter	Hexadecimal, 0x27 (default)	No
	ElementDelimiter	Hexadecimal, 0x2b (default)	No
	SubelementDelimiter	Hexadecimal, 0x3a (default)	No
	DecimalSeparator	Hexadecimal, 0x2e (default)	No
	ReleaseCharacter	Hexadecimal, 0x3f (default)	No
	ReplacementChar	Hexadecimal, 0x7c (default)	No
	RepeatingSeparator	Hexadecimal, 0x2a (default)	No
Document Type	GroupID	String	Yes
	TransactionControllingAgency	String	No
	TransactionAssociationAssignedCode	String	No
	TransactionCommonAccessRef	String	No
Document Definition	TransactionECSFileBlob	Absolute file path of ecs	Yes
	DocumentRoutingID	String	No
	FAAssocAssignedCode	String	No
	FAMessageVersionNumber	String	No
	FAMessageReleaseNumber	String	No
	RemoveFASegments	String	No
	Common Parameter Values (see Table H–13, " Common Parameter Values")		

Table H–10   HL7 Document Protocol Parameter Values

Document	Parameter Name Value	Domain	Required
Document Protocol Version	Security	String	No
	ProcessingID	String, P (default)	No
	AcceptAcknowledgementType	String, AL (default)	No
	AppAcknowledgementType	String, AL (default)	No
	CountryCode	String, US (default)	No
	CharacterSet	String, ASCII (default)	No
	BatchCreation	Boolean, false (default), true	No
	BatchECSFile	Absolute file path of ecs	No
	FileCreation	Boolean, false (default), true	N o
	FileECSFile	Absolute file path of ecs	No
	InternatCodeID	String	No
	InternatCodeText	String	No
	InternatCodeSystem	String	No
	InternatCodeAlternateID	String	No
	InternatCodeAlternateText	String	No
	InternatCodeAlternateSystem	String	No
	InternationalVersionID	String	No
	InternationalVersionIDText	String	No
	InternationalVersionIDSystem	String	No
	InternationalVersionIDAlternateID	String	N o
	InternationalVersionIDAlternateText	String	No
	InternationalVersionIDAlternateSystem	String	No
	ElementDelimiter	Hexadecimal, 0x7c (default)	No
	ReleaseCharacter	Hexadecimal, 0x5c (default)	No
	Repeating-Separator	Hexadecimal, 0x7e (default)	No
	SegmentDelimiter	Hexadecimal, 0x0d (default)	No
	SubcomponentDelimiter	Hexadecimal, 0x26 (default)	No
	SubelementDelimiter	Hexadecimal, 0x5e (default)	N o
	BatchSecurity	String	No
	FileSecurity	String	No
	FileDate	Date, #SystemDateTime(CCYYMMDDHHMM)# (default)	No
	BatchDate	Date, #SystemDateTime(CCYYMMDDHHMM)# (default)	No
Document Type	hl7-gen-Ack	Boolean, false (default), true	No
	hl7-mapAckControlID	Boolean, false (default), true	No
	hl7-acceptAcknowledgement	String, Any of these values - None, AL, ER, SU	No
Document Definition	MessageECSFileBlob	Absolute file path of ecs	Yes
	DocumentRoutingID	String	No
	Common Parameter Values (see Table H–13, " Common Parameter Values")		

Table H–11   RosettaNet Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Document Type	FromRole	String	Yes
	ToRole	String	Yes
	ServiceHeader	String	Yes
	ToService	String	Yes
	BusinessTransaction	String	Yes
	ServiceHeader	String	Yes
	CollaborationTimeToPerform	String	Yes
	CollaborationName	String	Yes
	CollaborationCode	String	Yes
Document Definition	DocumentRoutingID	String	No
	DTDXSDNamespaceConversion	String, any of these values - Both, Inbound, Outbound, None	No
	Common Parameter Values (see Table H–13, " Common Parameter Values")		

Table H–12   Custom Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Document Type	ActionName	String	No
	Service	String	No
	ServiceType	String	No
	FromRole	String	No
	ToRole	String	No
Document Definition	IdentificationExpression	String	No
	IdentificationExpressionValue	String	No
	DTDXSDNamespaceConversion	String, any of these values - None, Both, Inbound, Outbound	No
	IdentificationStartPosition	String	No
	IdentificationEndPosition	String	No
	FlatIdentificationExpressionValue	String	No
	DocumentRoutingID	String	No
	Common Parameter Values (see Table H–13, " Common Parameter Values")		

Table H–13   Common Parameter Values

Parameter Value Name	Domain	Required
XPathName1	String, XPathName1 (default)	No
XPathExpression1	String	No
XPathName2	String, XPathName2 (default)	No
XPathExpression2	String	No
XPathName3	String, XpathName3 (default)	No
XPathExpression3	String	No
CorrelationFromXPathName	String, CorrelationFromXPathName (default)	No
CorrelationFromXPathXPR	String	No

Table H–13   (Cont.) Common Parameter Values

Parameter Value Name	Domain	Required
CorrelationToXPathName	String, CorrelationToXPathName (default)	No
CorrelationToXPathXPR	String	No

Table H–14   AgrDocType Parameter Values

Parameter Value Name	Domain	Required	Applicable Documents
validate	boolean	No	X12, EDIFACT, HL7, RosettaNet, Custom
translate	boolean	No	X12, EDIFACT, HL7
fa	boolean	No	X12, EDIFACT, HL7

Index