Oracle® Fusion Middleware
Using Oracle B2B

14*c* (14.1.2.0.0)
F81868-01
November 2024

Oracle Fusion Middleware Using Oracle B2B, 14*c* (14.1.2.0.0)

F81868-01

Primary Author: Oracle Corporation

Contents

Preface

Using Oracle B2B 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.

Audience

This guide 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`.

Accessible Access to Oracle Support

Oracle customers who have purchased support 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

Refer to the Oracle Fusion Middleware library on the Oracle Help Center for additional information.

- For Oracle B2B information, see Oracle B2B.
- For Oracle SOA Suite information, see Oracle SOA Suite.
- For versions of platforms and related software for which Oracle products are certified and supported, review the Certification Matrix on OTN.

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.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I

Introduction to Oracle B2B

This part introduces Oracle B2B, an e-commerce gateway, and describes the interface and basic process flow.

This part contains the following chapters:

1 Introduction to Oracle B2B

This chapter discusses how Oracle B2B 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.

This chapter includes the following sections:

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 requirements of the receiver. Table 1-1 provides an example that compares the two kinds of transactions.

Table 1-1 Comparing Traditional and E-Commerce Transactions

Shipping Characteristic	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, OAG, RosettaNet, UCCnet, and more
How is the item packaged?	Box, bubble wrap	Packaging protocols: SMIME, SOAP, XMLDSig, XMLEncrypt

Table 1-1 (Cont.) Comparing Traditional and E-Commerce Transactions

Shipping Characteristic	Traditional Shipping Transaction	E-Commerce Transaction
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?	• Signed receipt • Overnight/next day • Delivery attempts	• Nonrepudiation • Time to acknowledge/respond • Retry counts

This guide describes how to use Oracle B2B to define the document, the packaging, and the delivery. It also covers how to configure trading partners, create and deploy agreements, and monitor a deployment.

Protocols Supported in Oracle B2B

Oracle B2B supports numerous industry-standard e-commerce protocols for a range of industries, including 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) • RosettaNet PIP business documents • OAG • UCCnet • Custom (non-XML)
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 • WS-HTTP

Table 1-2 (Cont.) Protocols Supported in Oracle B2B

Protocol Type	Protocol
Message exchange protocol	• AS1-1.0, AS2-1.1, and AS4-1.0 • 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

Note:

Using the Custom document protocols, you can use many other document types, including W3CXML Schema (OAGIS, xCBL, UBL, ebXML, and more).

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 Managing the MDS Repository in *Administering Oracle Fusion Middleware* for more information about MDS.

Because documents created in Oracle B2B are stored in the metadata repository, the transaction log for the database can become full. If this problem often occurs, 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
```

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 more 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 runtime 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 runtime report data.
- Remote monitor—This role can access reports and view runtime data related to its own exchange with the host trading partner.

For information about assigning roles, see Adding Trading Partner Users.

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, Web service security (OWSM) on WS-HTTP 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 runtime does not support the CLIENT-CERT authentication method. Therefore, B2B is not able to post to OAM-SSO protected URLs.

For more information about security, see Oracle Fusion Middleware Securing Applications with Oracle Platform Security Services.

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 runtime, 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 Administering Oracle SOA Suite and Oracle Business Process Management Suite* and 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 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.

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 Add Document Types That the User Has Permission to Access for information about 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.
- 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.

How Oracle B2B Fits 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 *Developing SOA Applications with 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.

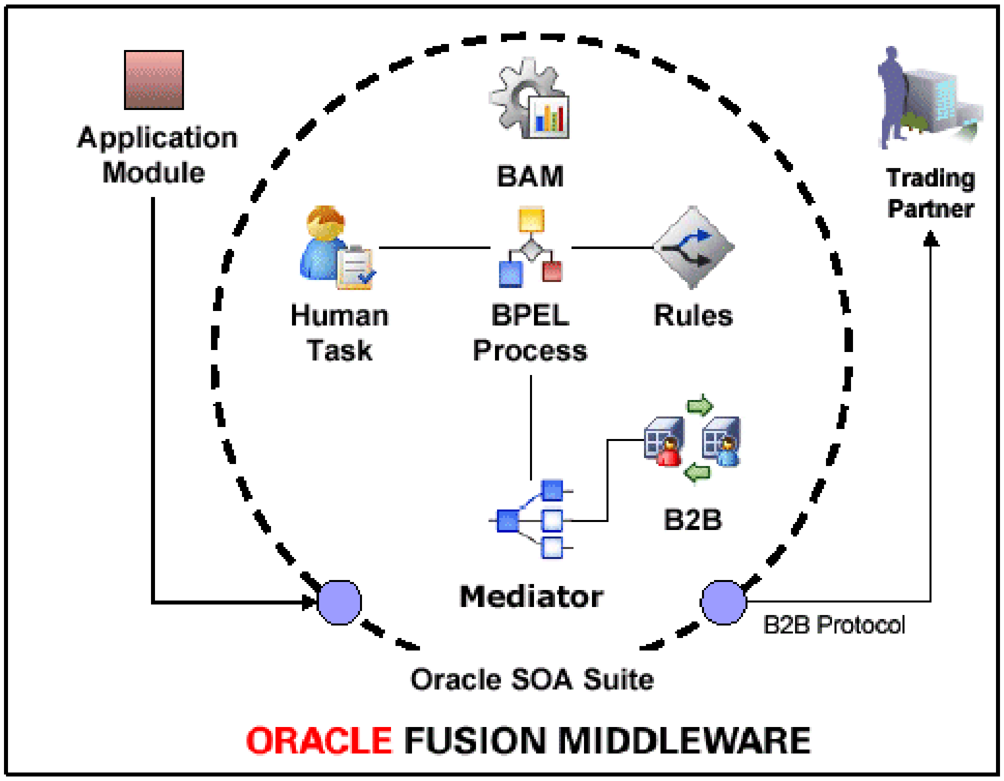
In addition to messages, Oracle B2B can also send attachments and large payloads in a SOA implementation. See Handling Large Payloads, for information about handling large payloads.

Sending a Purchase Order: An Example of a SOA Implementation

This example describes how the components of a SOA composite application are used to send a purchase order that originates from Oracle E-Business Suite.

Figure 1-1is an illustration of this example.

Figure 1-1 An Outbound Purchase Order in a SOA Composite Application



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 the required format if necessary, and manages the interaction with the trading partner.
6. Oracle Business Activity Monitoring (BAM) monitors the end-to-end process.

See Using Oracle B2B in the Oracle JDeveloper Environment for information about how to include a B2B binding component in a SOA composite application.

Administering Oracle B2B in the Oracle Fusion Middleware Environment

Several components provide monitoring, configuration, and performance tuning capabilities for Oracle B2B.

- 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:

  – Properties To Set in Fusion Middleware Control

  – Configuring Oracle B2B and Monitoring Oracle B2B in *Administering 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 Configuring B2B System Parameters .

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

  See Creating Reports.

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

  See Using B2B Metrics .

Accessibility Options

This section describes the accessibility options available with Oracle B2B.

Enabling Accessibility Features in Oracle B2B

Oracle B2B provides the screen reader option, which enables your screen reader to access and read all components of the application.

To enable the screen reader:

1. Click the **Enable screen reader mode** link in the top right corner.
2. The following confirmation message appears: **This will enable screen reader mode for the current session. Do you want to continue?**
3. Click **Yes** to confirm.

Note:

Where the calendar appears in the user interface, it is not available in Screen Reader mode.

2
Getting Started with Oracle B2B

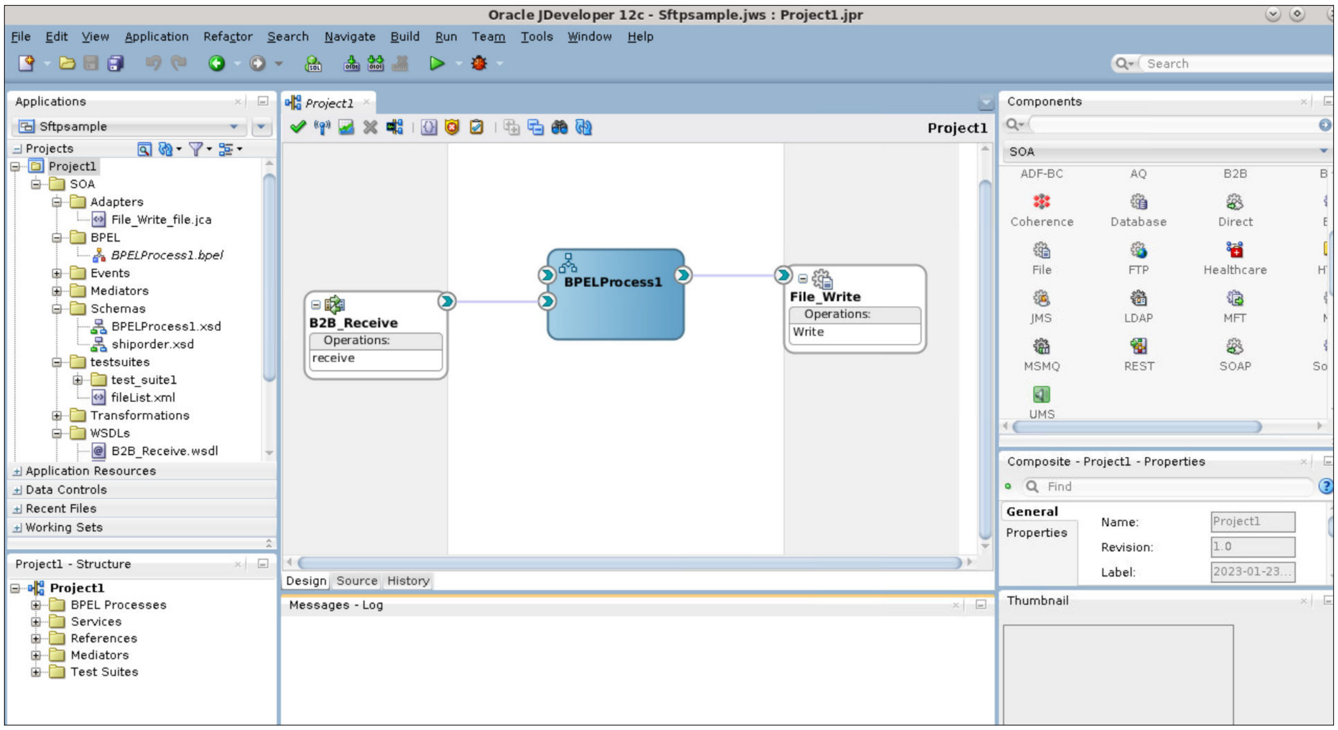
This chapter describes the Web-based interface provided by Oracle B2B for creating B2B transactions. It also discusses the process flow of creating an Oracle B2B transaction and how to use Oracle B2B in the Oracle JDeveloper environment.

This chapter includes the following sections:

What You Need to Get Started with Oracle B2B

Certain document protocols may require document structure guidelines such as XSD Schema. As needed, generate the guideline files using appropriate tools. 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



See the following for more information:

- Creating Guideline Files
- Oracle JDeveloper **Help** menu

Logging in to Oracle B2B

Once you have installed Oracle SOA Suite, including Oracle B2B, you can follow these instructions to log in to Oracle B2B.

You must 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 the 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 Finding Port Information for more information.
   - `/b2bconsole` (or `/b2b`) accesses the B2B interface (`/b2b` is redirected to `/b2bconsole`).

     See 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. Enter the default administrator user name.
3. Enter the administrator password from your Oracle Fusion Middleware 11*g* installation.
4. Click **Login**.

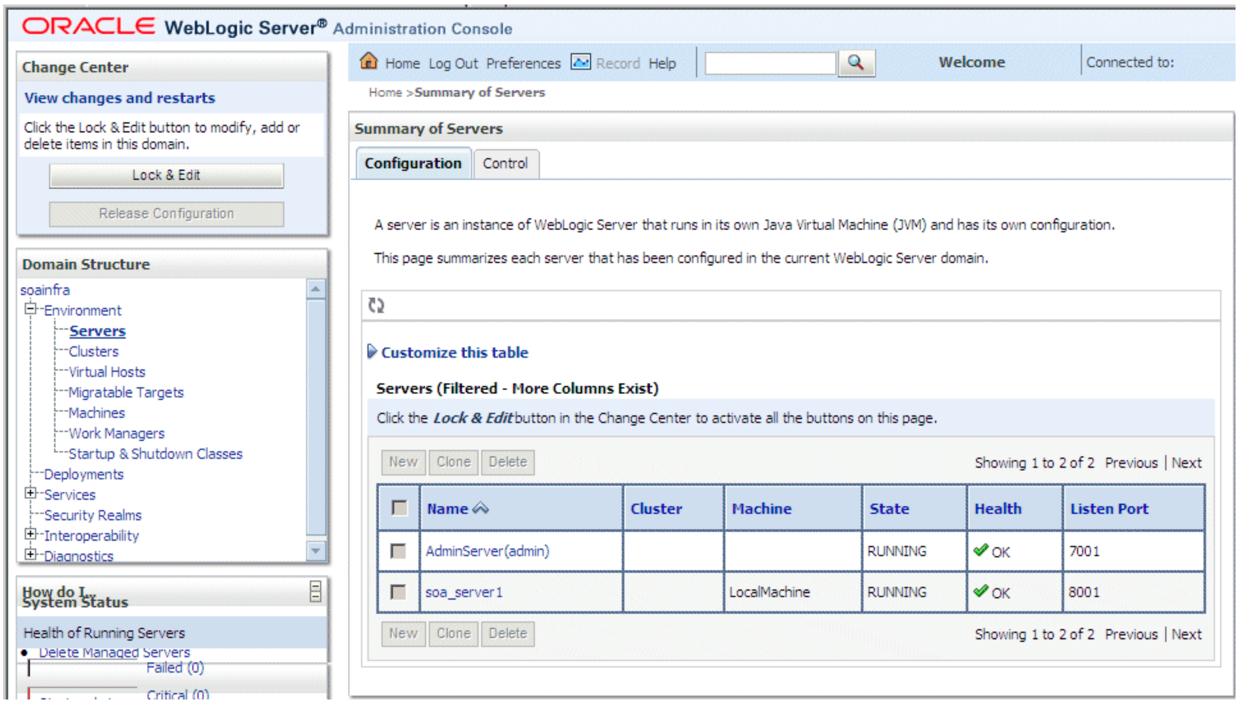
Finding Port Information

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



Click the Lock & Edit button to modify, add or delete items in this domain.

A server is an instance of WebLogic Server that runs in its own Java Virtual Machine (JVM) and has its own configuration.

This page summarizes each server that has been configured in the current WebLogic Server domain.

Servers (Filtered - More Columns Exist)

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

	Name	Cluster	Machine	State	Health	Listen Port
☐	AdminServer(admin)			RUNNING	✓ OK	7001
☐	soa_server1		LocalMachine	RUNNING	✓ OK	8001

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>
```

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>
```

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
   ```

Using the Oracle B2B Interface

B2B activities are grouped as Administration, Partners, Reports, and Metrics.

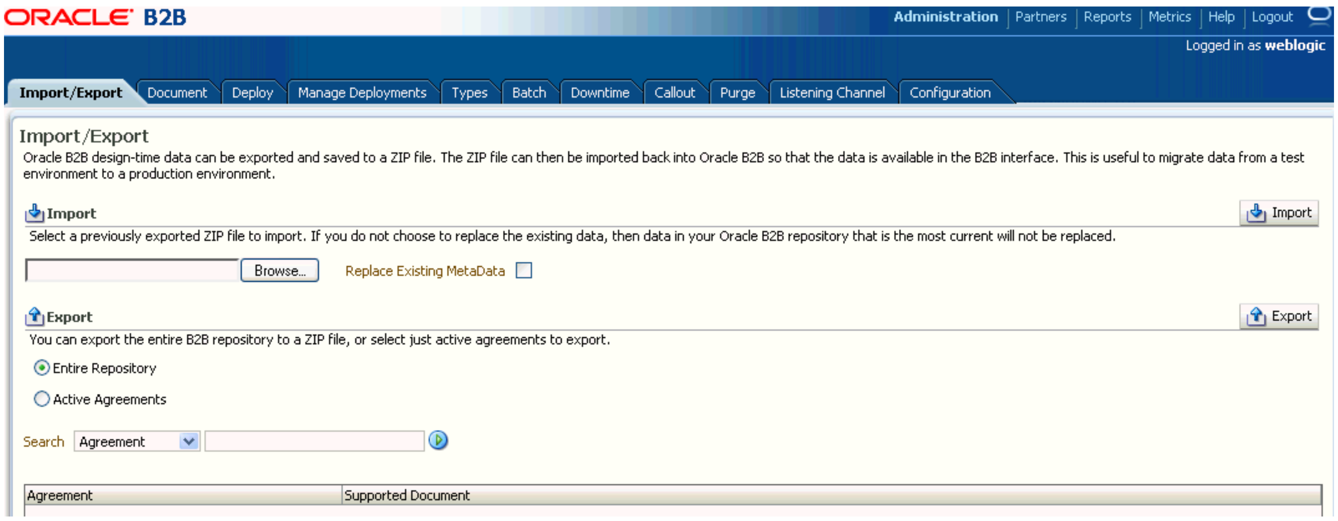
The following sections describe these activities.

- Administration
- Partners
- Reports
- Metrics

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


ORACLE B2B
Administration | Partners | Reports | Metrics | Help | Logout
Logged in as weblogic
Import/Export | Document | Deploy | Manage Deployments | Types | Batch | Downtime | Callout | Purge | Listening Channel | Configuration

Import/Export

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

Import

Select a previously exported ZIP file to import. If you do not choose to replace the existing data, then data in your Oracle B2B repository that is the most current will not be replaced.

Export

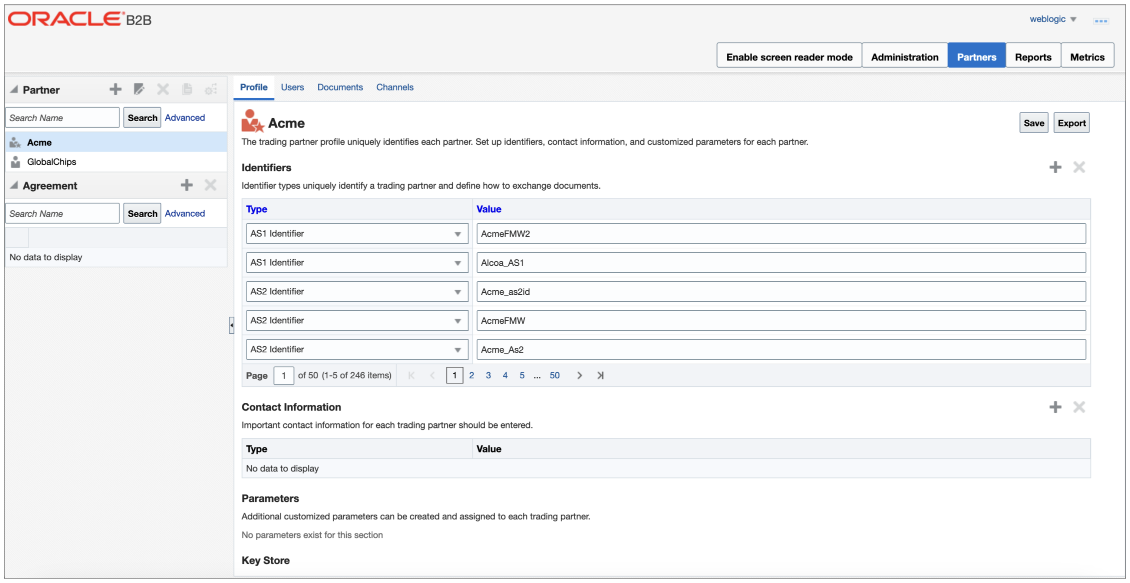
You can export the entire B2B repository to a ZIP file, or select just active agreements to export.

See Oracle B2B Administration for more information.

Partners

Use the tabs of the **Partners** page 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 trading partner profile uniquely identifies each partner. Set up identifiers, contact information, and customized parameters for each partner.

Identifiers

Identifier types uniquely identify a trading partner and define how to exchange documents.

Type	Value
AS1 Identifier	AcmeFMW2
AS1 Identifier	Alcoa_AS1
AS2 Identifier	Acme_as2id
AS2 Identifier	AcmeFMW
AS2 Identifier	Acme_As2

Page 1 of 50 (1-5 of 246 items) 1 2 3 4 5 ... 50

Contact Information

Important contact information for each trading partner should be entered.

Type	Value
No data to display	

Parameters

Additional customized parameters can be created and assigned to each trading partner.

No parameters exist for this section

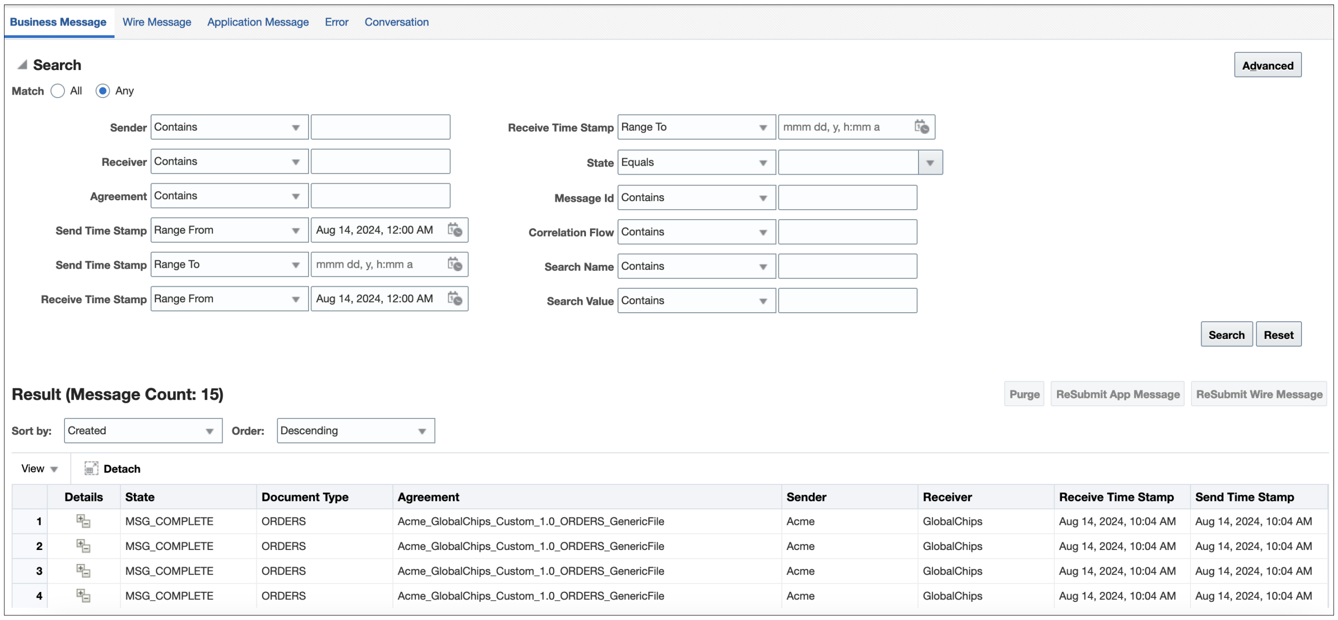
Key Store

See Oracle B2B Process Flow for more information.

Reports

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

Figure 2-5 Reports



Match ( ) All (●) Any

Sender	Contains	
Receiver	Contains	
Agreement	Contains	
Send Time Stamp	Range From	Aug 14, 2024, 12:00 AM
Send Time Stamp	Range To	mmm dd, y, h:mm a
Receive Time Stamp	Range From	Aug 14, 2024, 12:00 AM
Receive Time Stamp	Range To	mmm dd, y, h:mm a
State	Equals	
Message Id	Contains	
Correlation Flow	Contains	
Search Name	Contains	
Search Value	Contains	

Result (Message Count: 15)

Sort by: Created Order: Descending

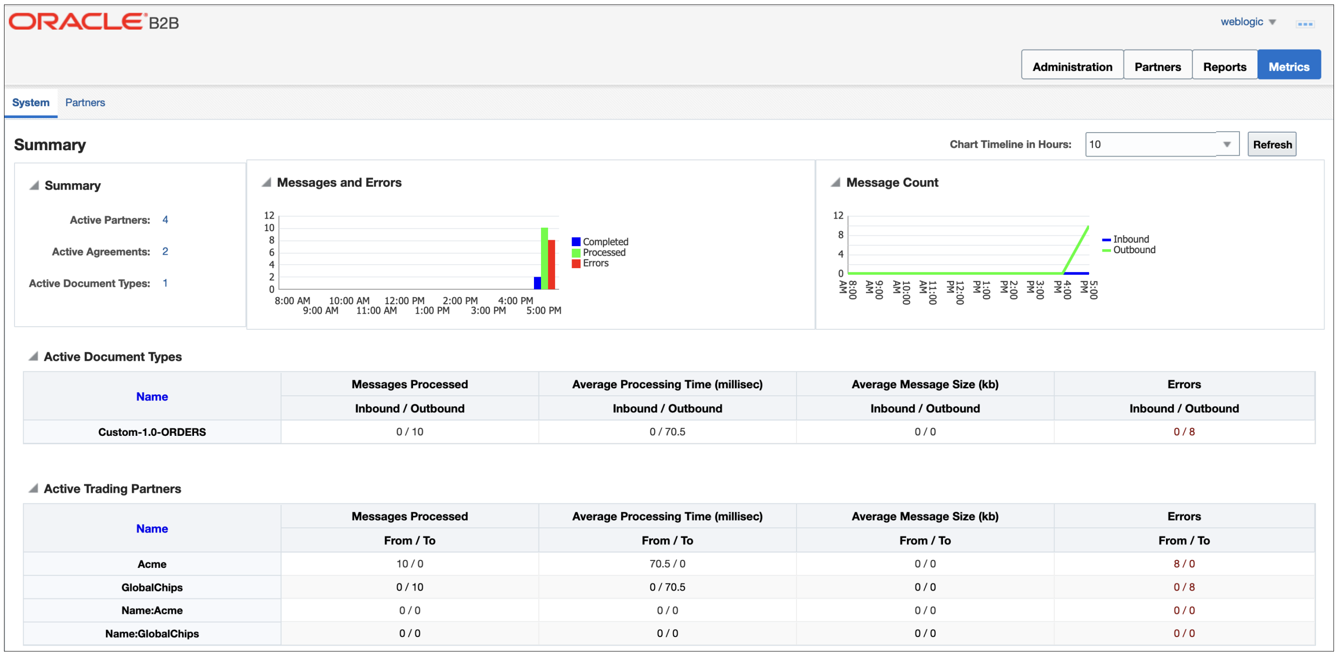
	Details	State	Document Type	Agreement	Sender	Receiver	Receive Time Stamp	Send Time Stamp
1		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
2		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
3		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
4		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM

For more information, see Creating Reports.

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 runtime status, such as error messages and message counts.

Figure 2-6   Metrics



Summary

Chart Timeline in Hours: 10

Summary

Active Partners:	4
Active Agreements:	2
Active Document Types:	1

Active Document Types

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound
Custom-1.0-ORDERS	0 / 10	0 / 70.5	0 / 0	0 / 8

Active Trading Partners

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	From / To	From / To	From / To	From / To
Acme	10 / 0	70.5 / 0	0 / 0	8 / 0
GlobalChips	0 / 10	0 / 70.5	0 / 0	0 / 8
Name:Acme	0 / 0	0 / 0	0 / 0	0 / 0
Name:GlobalChips	0 / 0	0 / 0	0 / 0	0 / 0

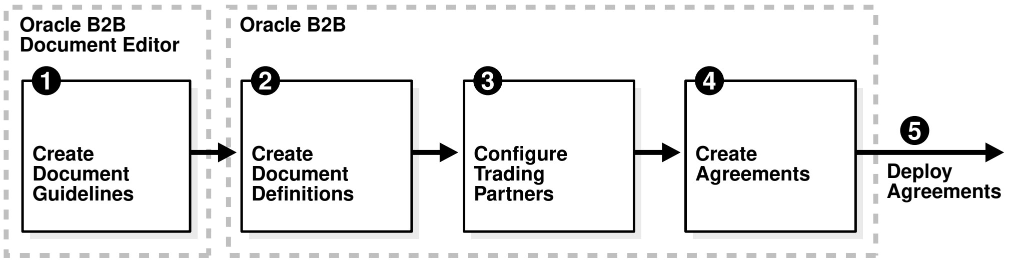
For more information, see Using B2B Metrics .

Creating a B2B Transaction: An Overview of the Process Flow

The B2B process flow starts with creating B2B guideline files such as XSD Schema using appropriate tools and continues with using the Oracle B2B interface to create document definitions, configure trading partners, and create and deploy agreements

Figure 2-7 illustrates the B2B process flow.

Figure 2-7   Oracle B2B Process Flow


Oracle B2B Document Editor
Oracle B2B
1
Create Document Guidelines
2
Create Document Definitions
3
Configure Trading Partners
4
Create Agreements
5
Deploy Agreements

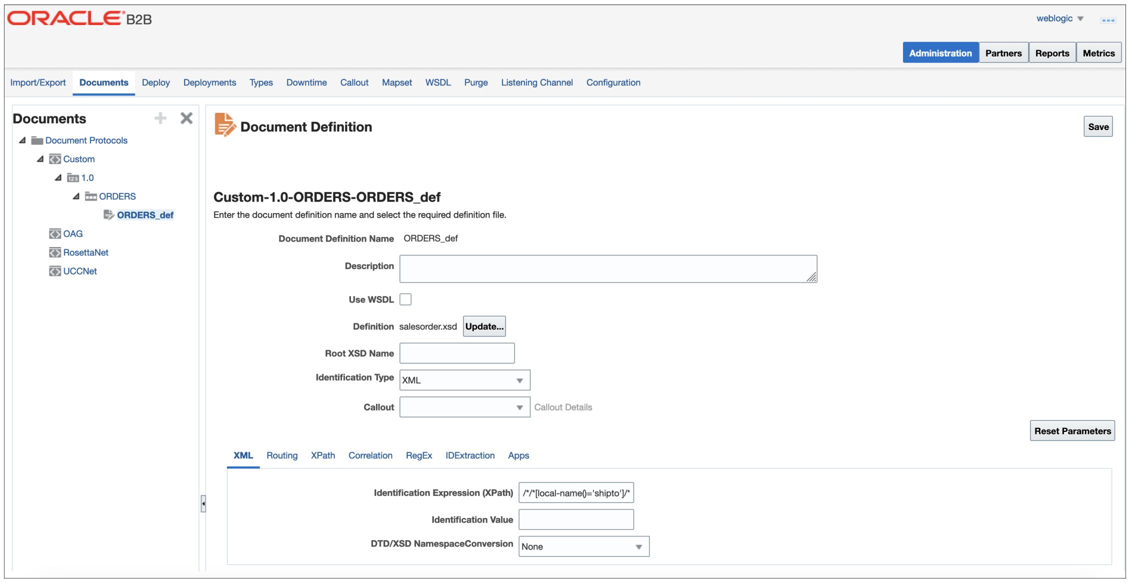
Step 1: Create guideline files

Certain document protocols may require document structure guidelines such as XSD Schema. As needed, generate the guideline files using appropriate tools.

Step 2: Create document definitions

Using the **Administration > Document** tab of Oracle B2B, shown in Figure 2-8, 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 2-8 Creating a Document Definition



Custom-1.0-ORDERS-ORDERS_def

Enter the document definition name and select the required definition file.

Document Definition Name	ORDERS_def
Description	
Use WSDL	☐
Definition	salesorder.xsd
Root XSD Name	
Identification Type	XML
Callout	Callout Details
Identification Expression (XPath)	/*/*[local-name()='shipto']/*
Identification Value	
DTD/XSD NamespaceConversion	None

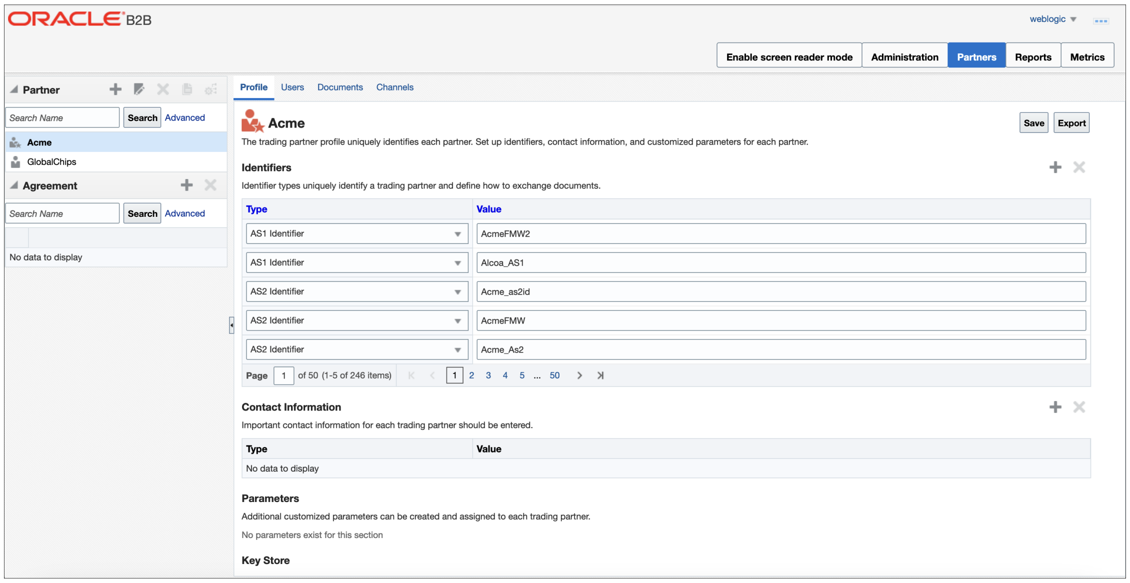
After selecting guideline files (if needed) you created in Step 1, you have created the document definition.

For more information, see Creating Document Definitions.

Step 3: Configure trading partners

Using the tabs of the **Partners** page of Oracle B2B, 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 2-9 Configuring Trading Partners



No data to display

Acme

The trading partner profile uniquely identifies each partner. Set up identifiers, contact information, and customized parameters for each partner.

Identifiers

Identifier types uniquely identify a trading partner and define how to exchange documents.

Type	Value
AS1 Identifier	AcmeFMW2
AS1 Identifier	Alcoa_AS1
AS2 Identifier	Acme_as2id
AS2 Identifier	AcmeFMW
AS2 Identifier	Acme_As2

Page 1 of 50 (1-5 of 246 items) 1 2 3 4 5 ... 50

Contact Information

Important contact information for each trading partner should be entered.

Type	Value
No data to display	

Parameters

Additional customized parameters can be created and assigned to each trading partner.

No parameters exist for this section

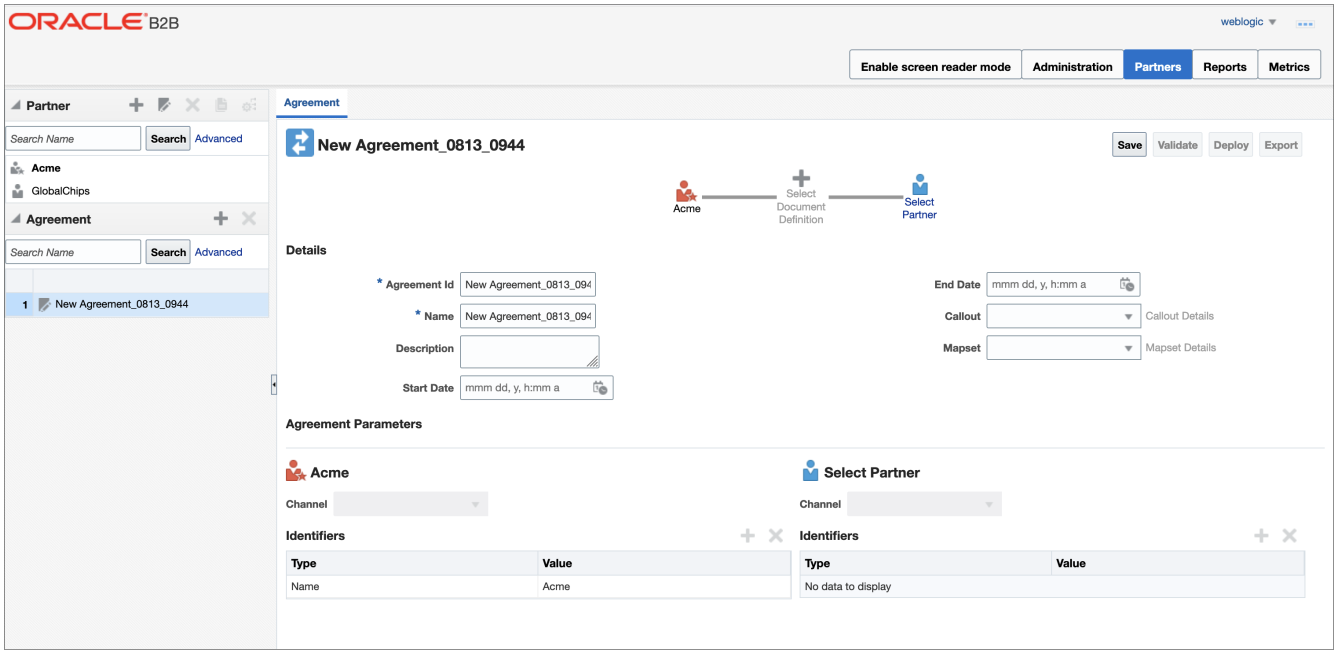
Key Store

For more information, see Configuring Trading Partners .

Step 4: Create agreements

Using the **Partners > Agreement** tab of Oracle B2B, shown in Figure 2-10, create an agreement that specifies the trading partners involved and associates the document definitions, channels, and identifiers with the agreement.

Figure 2-10 Creating a Trading Partner Agreement



New Agreement_0813_0944

Details

* Agreement Id	New Agreement_0813_094
* Name	New Agreement_0813_094
Description	
Start Date	mmm dd, y, h:mm a
End Date	mmm dd, y, h:mm a
Callout	Callout Details
Mapset	Mapset Details

Agreement Parameters

Acme

Channel

Identifiers

Type	Value
Name	Acme

Select Partner

Channel

Identifiers

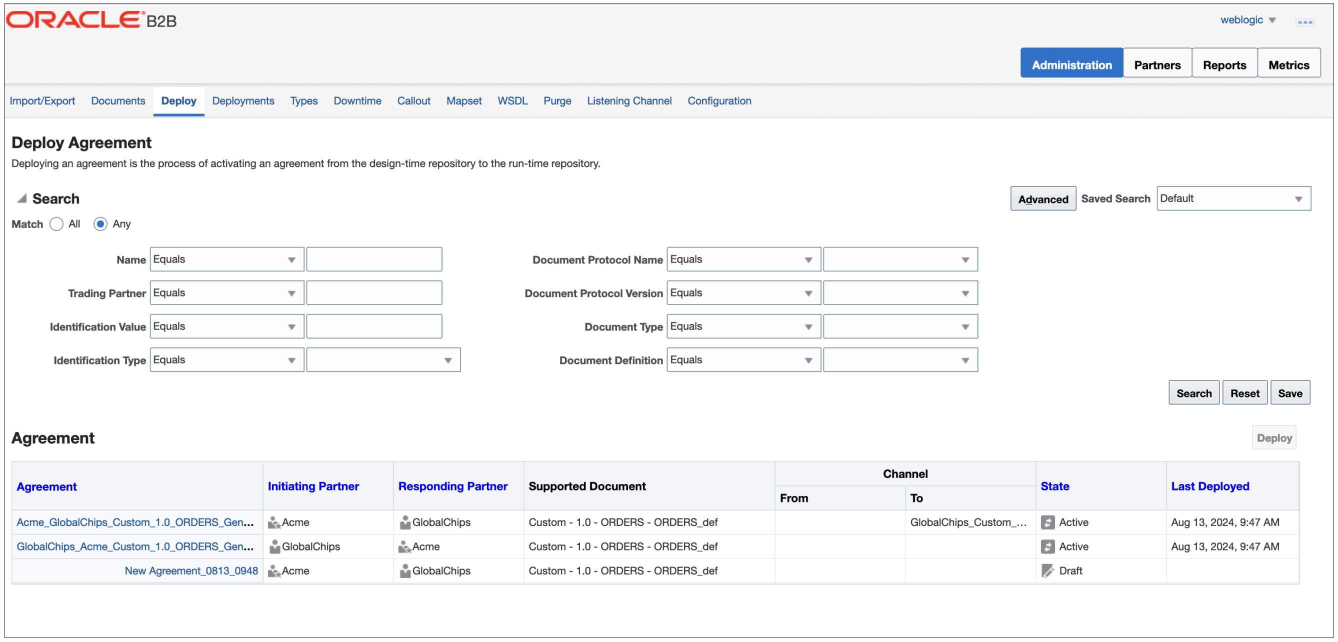
Type	Value
No data to display	

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

Step 5: Deploy agreements

Using the **Administration > Deploy** tab of Oracle B2B, shown in Figure 2-11, search for and deploy agreements.

Figure 2-11 Searching for and Deploying Agreements



Deploy Agreement

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

Search

Saved Search: Default

Match: All [✓ Any]

Name	Equals	Document Protocol Name	Equals
Trading Partner	Equals	Document Protocol Version	Equals
Identification Value	Equals	Document Type	Equals
Identification Type	Equals	Document Definition	Equals

Agreement

Agreement	Initiating Partner	Responding Partner	Supported Document	Channel		State	Last Deployed
				From	To		
Acme_GlobalChips_Custom_1.0_ORDERS_Gen...	Acme	GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def		GlobalChips_Custom_...	Active	Aug 13, 2024, 9:47 AM
GlobalChips_Acme_Custom_1.0_ORDERS_Gen...	GlobalChips	Acme	Custom - 1.0 - ORDERS - ORDERS_def			Active	Aug 13, 2024, 9:47 AM
New Agreement_0813_0948	Acme	GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def			Draft	

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.

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:

- Create a SOA Application and Project
- Add Service Components
- Add a B2B Binding Component

See the following topics in *Developing SOA Applications with Oracle SOA Suite* for more information on creating SOA composite applications:

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

Create a SOA Application and Project

Use these steps to 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 in one of the following ways:
   - If you are opening Oracle JDeveloper for the first time, or it has no applications, then in the Application Navigator in the upper left, click **New Application**.
   - If Oracle JDeveloper has existing applications, then from the **File** main menu or the **Application** menu, select **New** > **Applications** to open the New Gallery, where you can select different application components.
   - In the **Categories** tree, under the **General** node, select **Applications**. In the **Items** pane, select **SOA Application** and click **OK**.
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**.

   > **Note:**
   >
   > Note the following restrictions and recommendations.

   - 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 runtime 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, you have Application1 with Project1, and you create Application2 with Project1. During deployment, the second deployed project (composite) overwrites the first deployed project (composite).

6. In the Configure SOA Settings page, click **Empty Composite**, and click **Finish**.
7. Select **Save All** from the **File** main menu.

For information about creating a SOA application and project, see Creating a SOA Application in *Developing SOA Applications with Oracle SOA Suite*.

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.

   A specific dialog for the selected service components is displayed. Table 2-1 describes the available editors.

Table 2-1 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.

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 below.

   Table 2-2 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 below.

Table 2-3   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 below.

Table 2-4   Selecting Document Definition Handling Options

Option	Description
Import Schema from B2B (**Basic** tab)	Imports the schema from Oracle B2B (the same option as on the **Advanced** tab)
Import Schema from B2B (**Advanced** tab)	Imports the schema from Oracle B2B (the same option as on the **Basic** tab)
Refer to Schema in B2B Repository (**Advanced** tab)	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 (**Advanced** tab)	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 (**Advanced** tab)	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 (**Advanced** tab)	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.

   Other options are described below.

Table 2-5 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.
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. For more information, see Using Document Routing IDs.

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.

    The information displayed when the database connection is established is described below.

Table 2-6 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 runtime.
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 topics in *Developing SOA Applications with Oracle SOA Suite* for more information on binding components:

- Adding Service Binding Components
- Adding Reference Binding Components
- Getting Started with Binding Components

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 2-7 lists the JMS properties to use with the Oracle JCA Adapter for JMS.

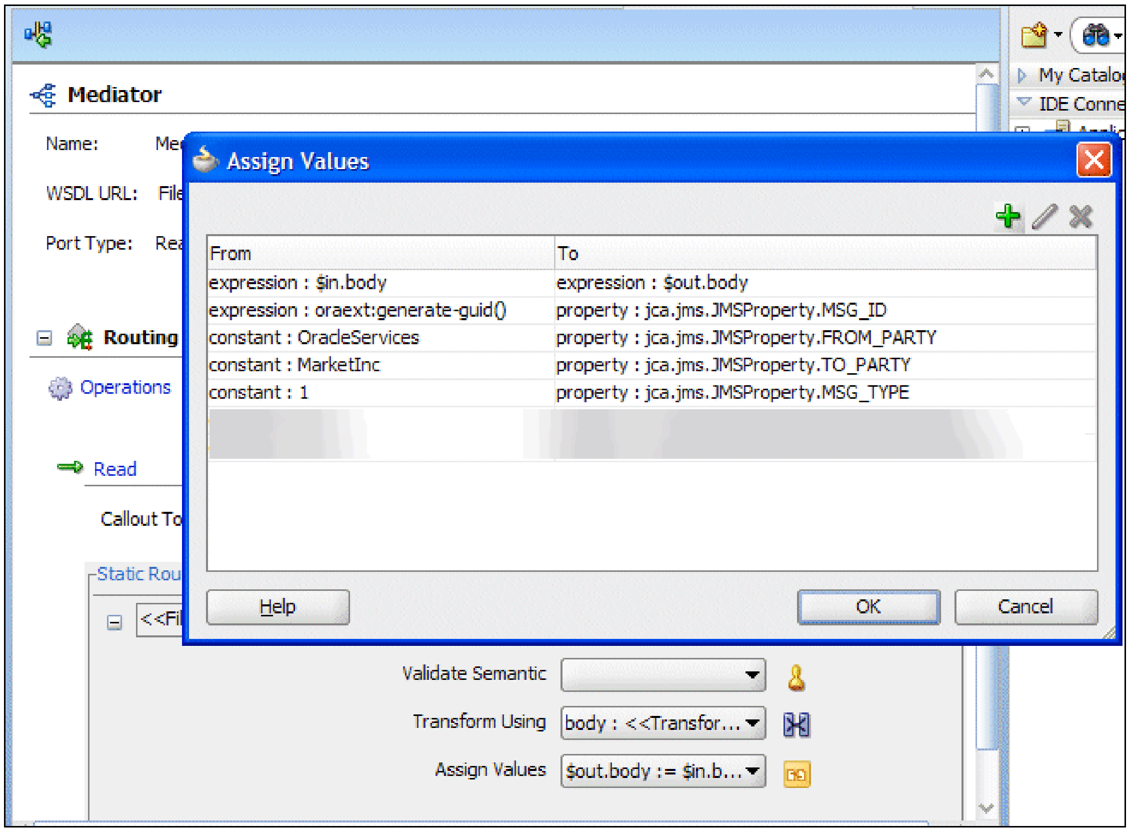
Table 2-7 JMS Adapter Properties

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 2-12.

Figure 2-12 Adding JCA JMS Properties in the Assign Values Dialog



From	To
expression : $in.body	expression : $out.body
expression : oraext:generate-guid()	property : jca.jms.JMSProperty.MSG_ID
constant : OracleServices	property : jca.jms.JMSProperty.FROM_PARTY
constant : MarketInc	property : jca.jms.JMSProperty.TO_PARTY
constant : 1	property : jca.jms.JMSProperty.MSG_TYPE

Validate Semantic

Transform Using: body : <<Transfor...

Assign Values: $out.body := $in.b...

Part II

Oracle B2B Process Flow

This part describes the B2B process flow, with guidelines files, document definitions, trading partners, and deploying agreements.

This part contains the following chapters:

3 Creating Guideline Files

Creating guideline files is the first step in the Oracle B2B process flow. Certain document protocols may require document structure guidelines such as XSD Schema. As needed, User must generate the guideline files using appropriate tools.

4

Creating Document Definitions

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

Creating document definitions is the second step in the Oracle B2B process flow. A document definition specifies the document protocol—the document protocol version and document type—that is used to validate the message. The document definition guideline used will depend on type of message such as XSD/DTD for XML messages.

Note:

In the case of a custom document protocol with XML, then ensure that the Identification Expression (XPath) is unique across the document type/revision/document definition.

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 create the document guideline files (Step 1) and then the Oracle B2B interface to import those files when creating the document definition (Step 2).

This chapter includes the following sections:

- Introduction to Document Protocols
- Creating Document Definitions
- Deleting a Document Definition

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

Introduction to Document Protocols

Using the Custom protocol and the guideline documents, you can define most protocols. When you add a new document protocol, it is always a Custom document.

Oracle B2B supports these document protocols:

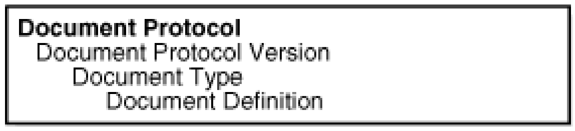
- Custom
- OAG
- RosettaNet
- UCCNet
- UserDefined

As part of the document definition, you may provide the document guideline files. If validation is enabled, then, at runtime, the payload must conform to the document definition file type you use.

The Document Hierarchy

You can think of a document protocol as a hierarchy, as shown in Figure 4-1.

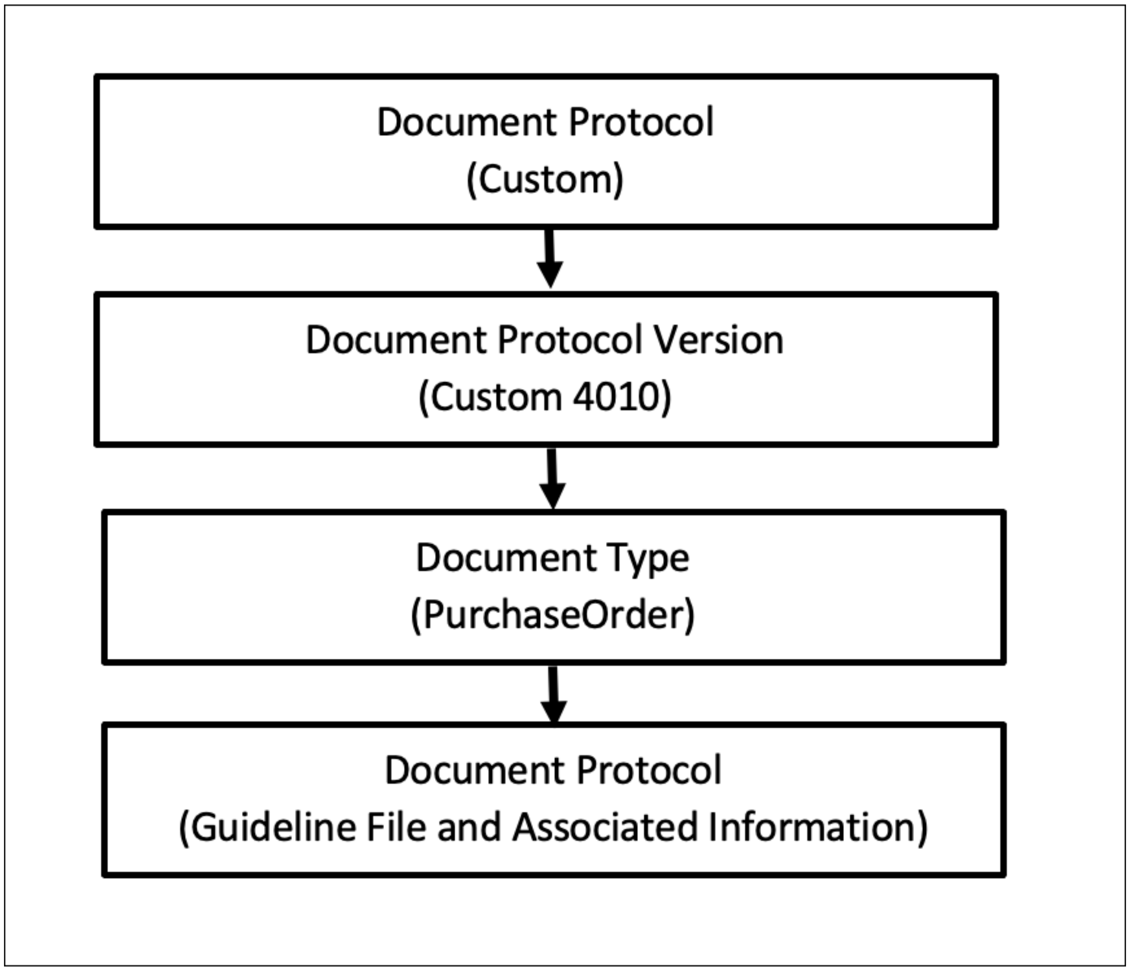
Figure 4-1 Document Hierarchy



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 4-2 shows a document protocol hierarchy as it applies to Custom.

Figure 4-2 Custom Document Hierarchy



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 a Custom document.

Document protocol: CUSTOM

Document protocol version: 4010

Document type: PurchaseOrder

Document definition: PurchaseOrder_def

The resulting document definition is

CUSTOM-4010-PurchaseOrder-PurchaseOrder_def

Creating Document Definitions

After creating required guideline file, use the Oracle B2B interface to create the document definition and import the guideline file.

Note:

You cannot edit the document version, document type, and document definition 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 runtime errors.

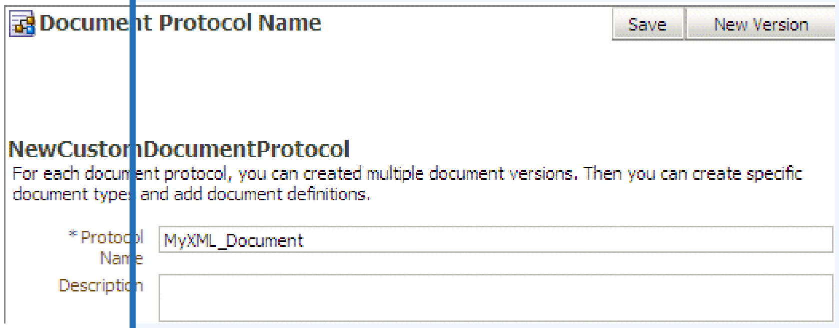
To create a document definition:

1. Click the **Administration** tab.
2. Click the **Document** tab.
3. Select one of the document protocols.

Note:

To create a new Custom document with a name that you provide, click **Document Protocols** folder and click **Add**.

Then enter a protocol name, for example, MyXML_Document. Do not use an existing document protocol name.



Document Protocol Name

NewCustomDocumentProtocol

For each document protocol, you can created multiple document versions. Then you can create specific document types and add document definitions.

* Protocol Name: MyXML_Document

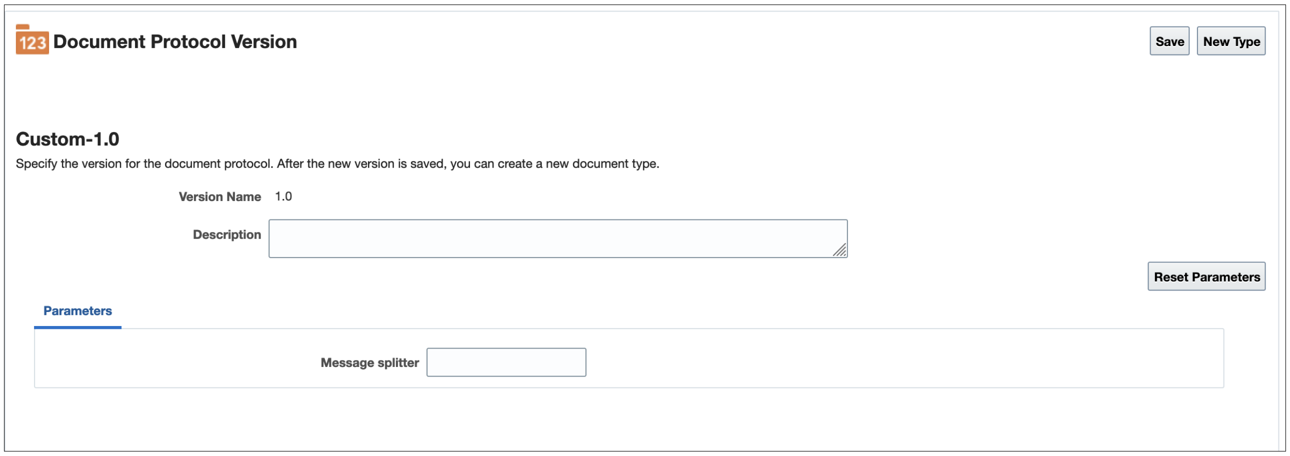
Description:

4. Click **New Version**.
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 4-3 shows the document protocol version page for a Custom document.

Figure 4-3 Entering Document Protocol Parameter Information



Document Protocol Version

Custom-1.0

Specify the version for the document protocol. After the new version is saved, you can create a new document type.

Version Name 1.0

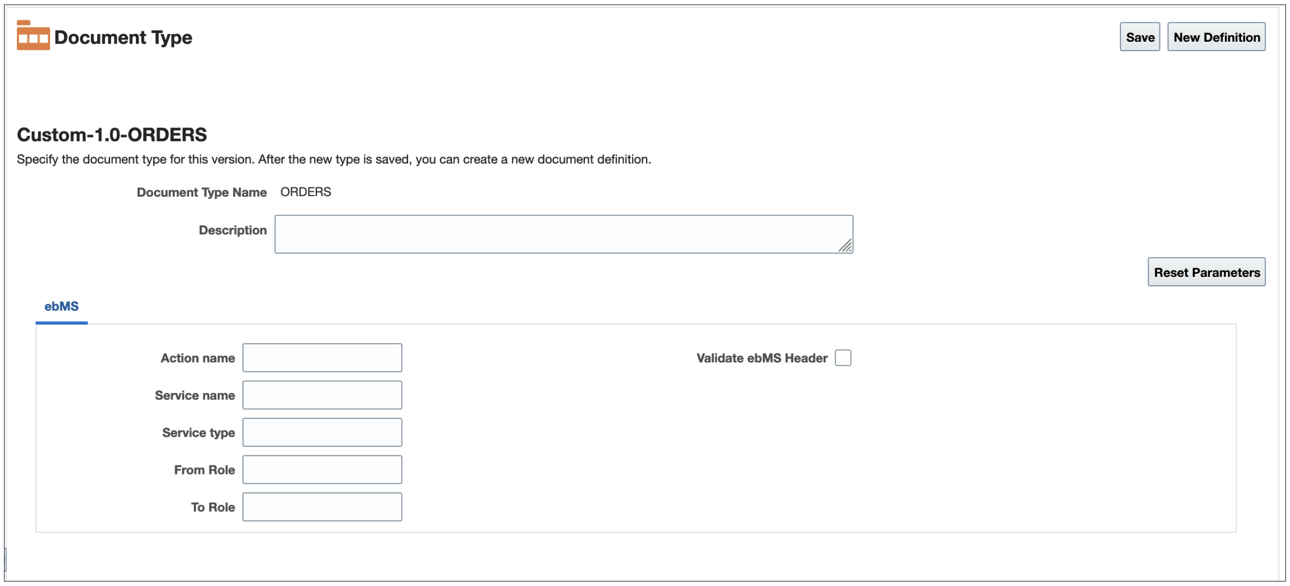
Description

Message splitter

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 4-4 shows the document type parameters page for a Custom document.

Figure 4-4 Entering Document Type Parameter Information



Custom-1.0-ORDERS

Specify the document type for this version. After the new type is saved, you can create a new document definition.

Document Type Name	ORDERS
Description	
Action name	
Service name	
Service type	
From Role	
To Role	
Validate ebMS Header	☐

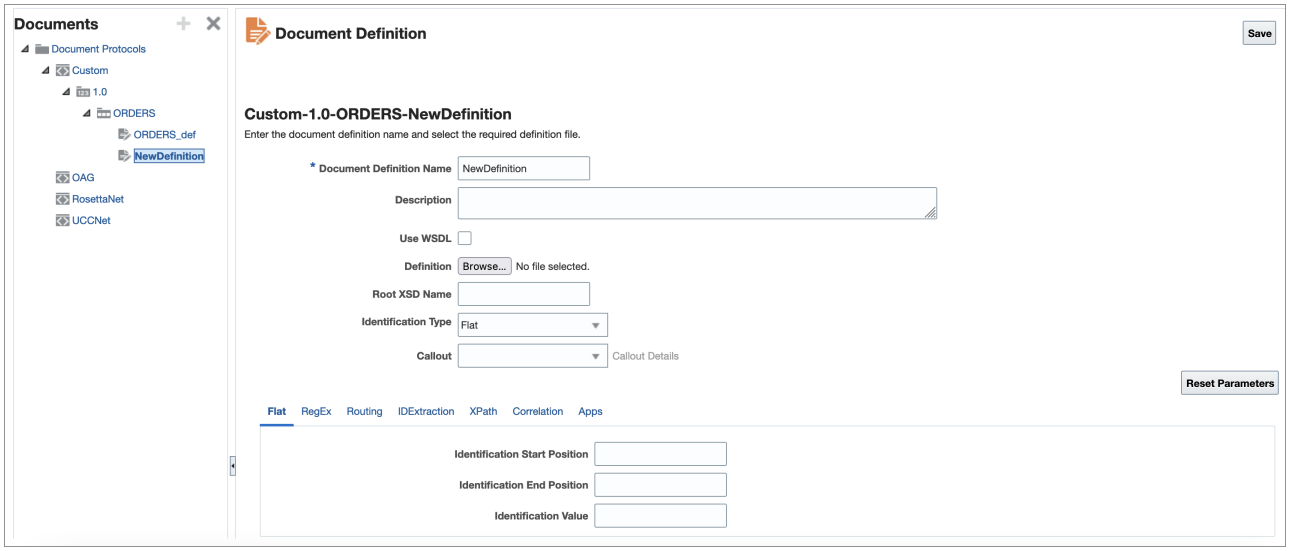
For parameter descriptions, see the following:

- Table 8-1 Document Type Parameters for a Custom Document
- Table 8-6 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 a guideline file for any of the document protocols, if applicable.
   - Provide document definition parameters as applicable.

   Figure 4-5 shows the document definition parameters page for a Custom document.

Figure 4-5 Entering Document Definition Parameter Information



Custom-1.0-ORDERS-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name	NewDefinition
Description	
Use WSDL	☐
Definition	No file selected.
Root XSD Name	
Identification Type	Flat
Callout	
Identification Start Position	
Identification End Position	
Identification Value	

For parameter descriptions, see the following:

- Table 8-2 Document Definition Parameters for a Custom Document
- Table 8-4 Document Definition Parameters for an OAG Document
- Table 8-7 Document Definition Parameters for a RosettaNet Document
- Table 8-8 Document Definition Parameters for a UCCnet Document

10. Click **Save**.

Deleting a Document Definition

To delete a document definition, first delete all agreements that use that document definition.

Then, you 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.

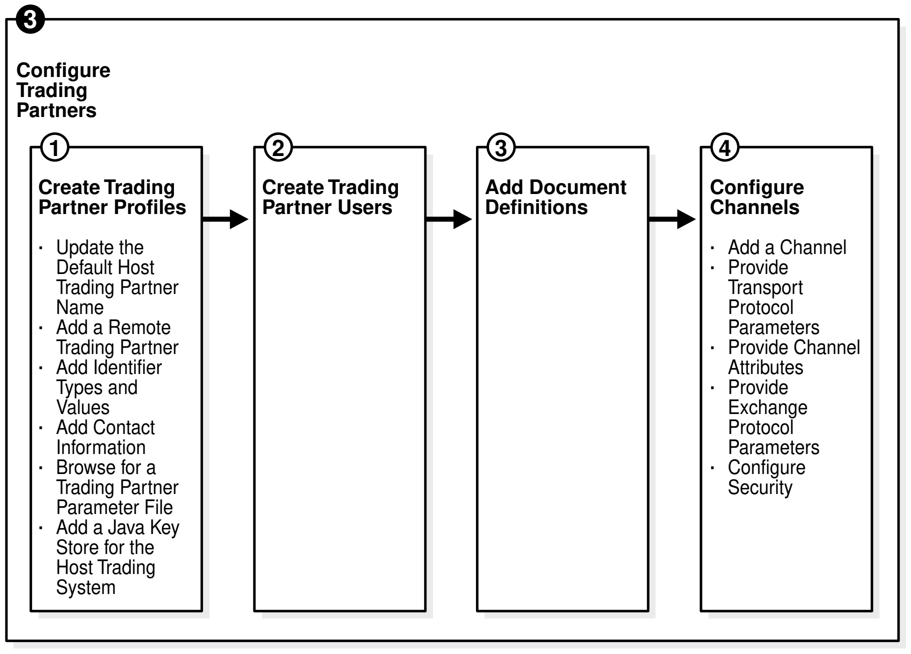
Configuring trading partners is the third step in the Oracle B2B process flow.

This chapter includes the following sections:

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 5-1 shows the steps to configure a trading partner.

Figure 5-1 Configuring Trading Partners


3
Configure Trading Partners
1
Create Trading Partner Profiles
· Update the Default Host Trading Partner Name
· Add a Remote Trading Partner
· Add Identifier Types and Values
· Add Contact Information
· Browse for a Trading Partner Parameter File
· Add a Java Key Store for the Host Trading System
2
Create Trading Partner Users
3
Add Document Definitions
4
Configure Channels
· Add a Channel
· Provide Transport Protocol Parameters
· Provide Channel Attributes
· Provide Exchange Protocol Parameters
· Configure Security

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:

- Update the Default Host Trading Partner Name
- Add a Remote Trading Partner
- Add Identifier Types and Values
- Add Contact Information
- Add a Trading Partner Parameter and Value
- Provide Key Store Information for the Host Trading Partner

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**.

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

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

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

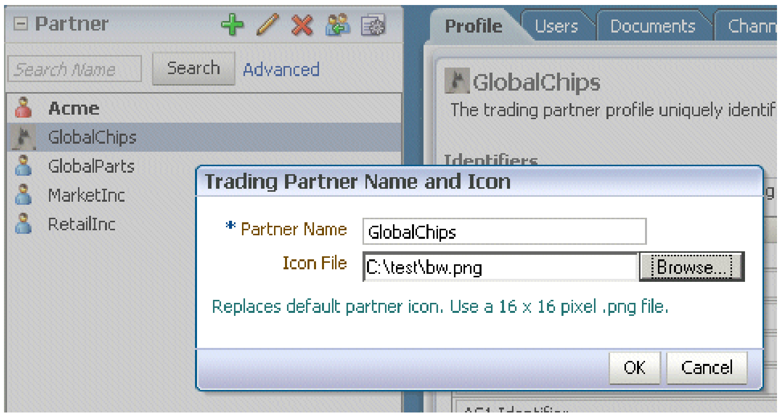
Add a Remote Trading Partner

Do this for each remote trading partner.

1. Click the **Partners** link.
2. Click **Add**.
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 button for the remote trading partner, as shown in Figure 5-2, and click **OK**.

Figure 5-2 Editing a Remote Trading Partner Profile



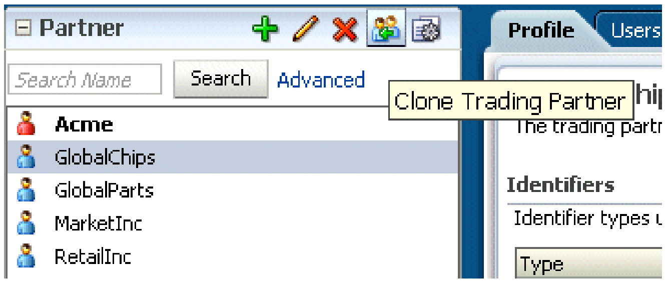
* Partner Name: GlobalChips

Icon File: C:\test\bw.png

Replaces default partner icon. Use a 16 x 16 pixel .png 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** button, as shown in Figure 5-3, and provide the trading partner information that is not cloned: identifiers, contacts, and users.

Figure 5-3 Cloning a Remote Trading Partner



Note:

Use the **Delete** button 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.

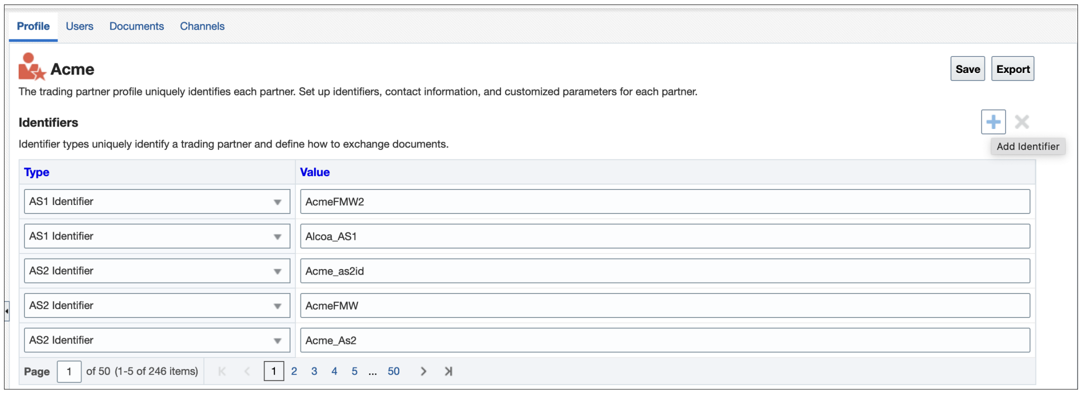
Add Identifier Types and Values

Identifier types enable Oracle B2B to identify a trading partner at runtime. 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.

To add identifier types and values for both the host and remote trading partners:

1. Click the **Partners** link.
2. Click the **Profile** tab.
3. Select a trading partner.
4. In the **Identifiers** area, click **Add**.

Figure 5-4 Adding Identifier Types



Acme

The trading partner profile uniquely identifies each partner. Set up identifiers, contact information, and customized parameters for each partner.

Identifiers

Identifier types uniquely identify a trading partner and define how to exchange documents.

Type	Value
AS1 Identifier	AcmeFMW2
AS1 Identifier	Alcoa_AS1
AS2 Identifier	Acme_as2id
AS2 Identifier	AcmeFMW
AS2 Identifier	Acme_As2

Page 1 of 50 (1-5 of 246 items) 1 2 3 4 5 ... 50

5. From the **Type** list, select an identifier type. For information about how to create the types shown here, see Creating Types .

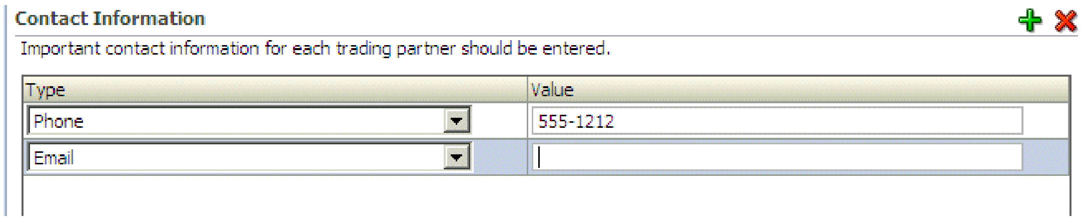
   See Table 10-1 (Identifier Types Defined in Oracle B2B) for descriptions of the identifier types.
6. Provide a value.
7. Repeat Steps 4 through 6 as needed.
8. Click **Save**.

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. For more information, see Creating Custom Contact Information Types.

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 5-5.

Figure 5-5 Adding Contact Information


Contact Information
Important contact information for each trading partner should be entered.

Type	Value
Phone	555-1212
Email	

5. Click **Save**.

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** button does not appear.) For more information, see Creating Types .

1. Click the **Partners** link.
2. Click the **Profile** tab.
3. In the **Parameters** area, click **Add**.
4. Select a parameter and click **OK**.
5. Click **Save**.

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

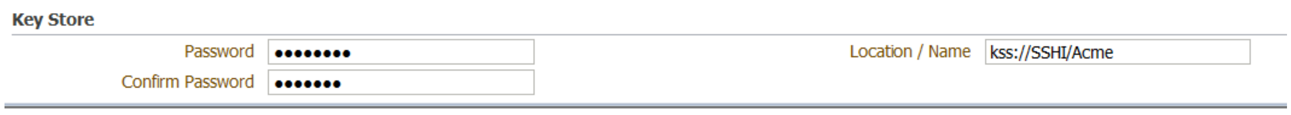
Provide Key Store Information for the Host Trading Partner

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 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 5-6.

Figure 5-6 Entering Key Store Information


Key Store

Password: ••••••••
Confirm Password: •••••••
Location / Name: kss://SSHI/Acme

5. Click **Save**.

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. For more information, see Restricting Access to Document Types.

> **Note:**
>
> A B2B user with the Administrator role must have membership in the WebLogic Administrators group to get the privileges such as seeing all trading partners. In addition, a B2B user with the Monitor role must be a member of the WebLogic Monitors group.

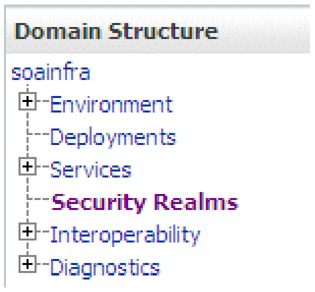
To add users, complete the following tasks:

- Create a New User in the Identity Store
- Add a User in the Oracle B2B Interface
- Add Document Types That the User Has Permission to Access

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 5-7.

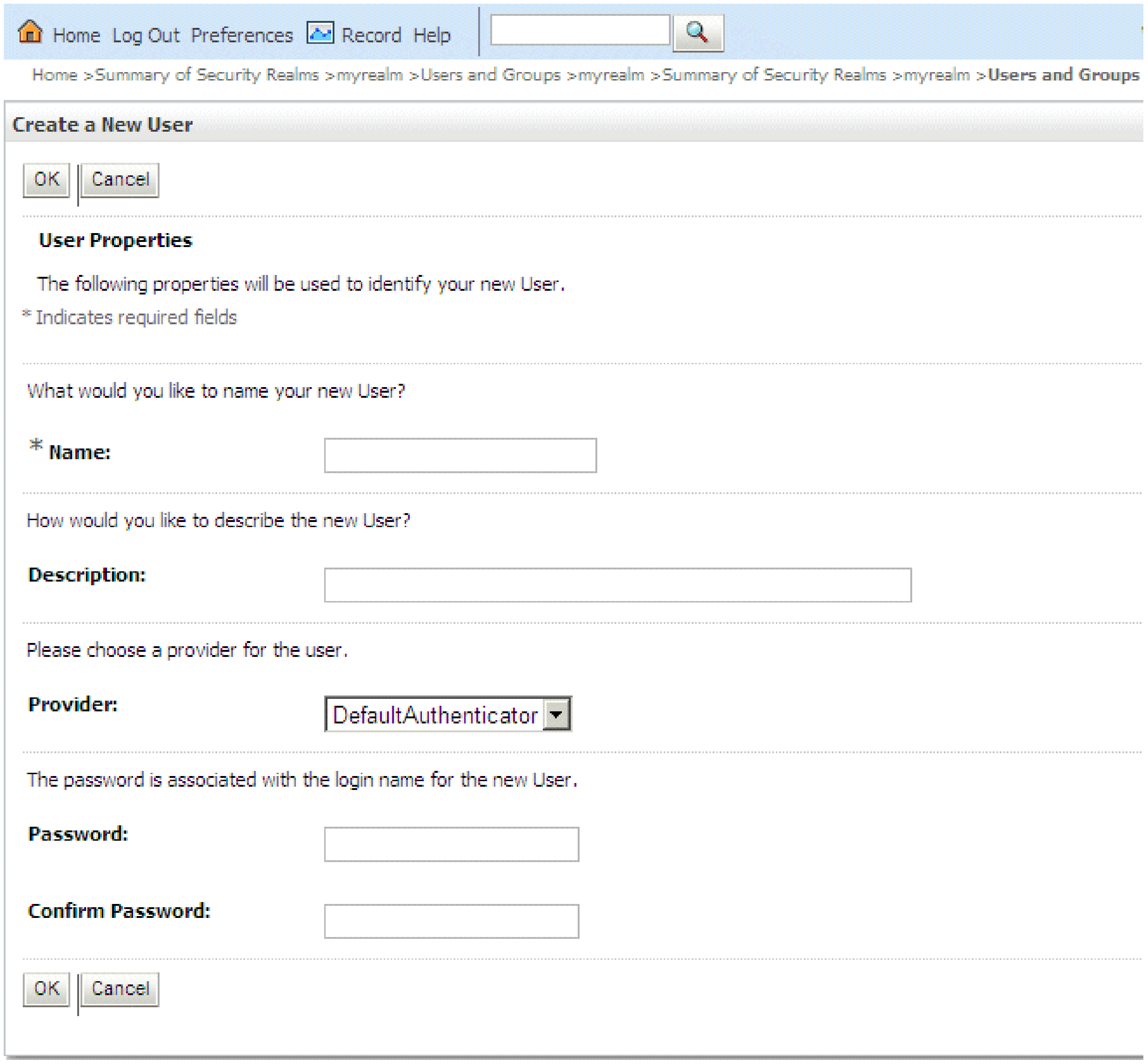
Figure 5-7 Oracle WebLogic Server Administration Console—Security Realms



> **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 5-8.

Figure 5-8 Oracle WebLogic Server Administration Console—Adding a New User


Home Log Out Preferences Record Help
Home >Summary of Security Realms >myrealm >Users and Groups >myrealm >Summary of Security Realms >myrealm >Users and Groups
OK
Cancel

Create a New User

User Properties

The following properties will be used to identify your new User.

* Indicates required fields

What would you like to name your new User?

* Name:

How would you like to describe the new User?

Description:

Please choose a provider for the user.

Provider: DefaultAuthenticator

The password is associated with the login name for the new User.

Password:

Confirm Password:

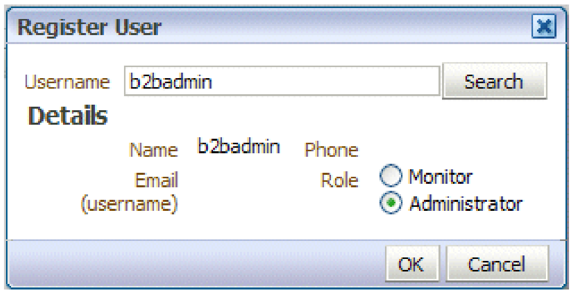
Add a User in the Oracle B2B Interface

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 Create a New User in the Identity Store and click **Search**.

   Enter the user name exactly is it was created.
6. Select the **Monitor** or **Administrator** role, shown in Figure 5-9, and click **OK**.

Figure 5-9 Assigning the Monitor or Administrator Role



Username: b2badmin

Name: b2badmin

Email (username):

Phone:

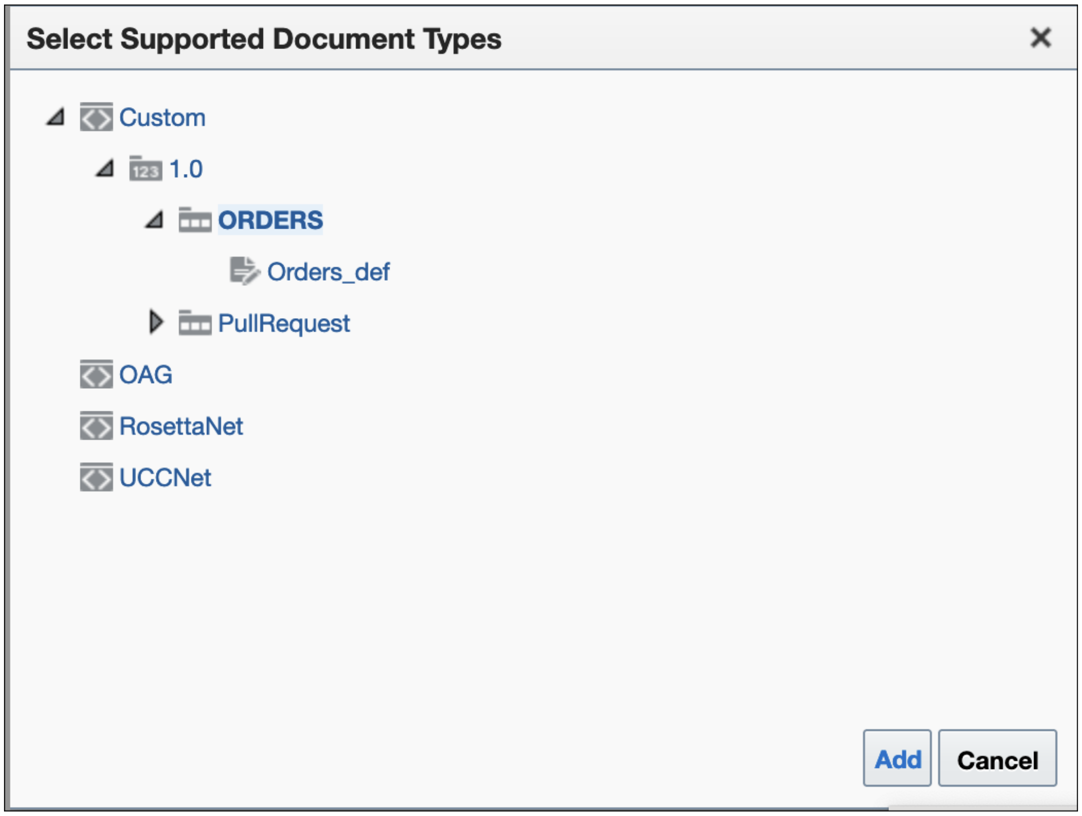
Role: Monitor [✓ Administrator]

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 on the **Users** tab click **Add**.
2. Select a document type and click **Add**, as shown in Figure 5-10.

Figure 5-10 Selecting a Document Type



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.

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 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**).

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

To add document definitions, do the following:

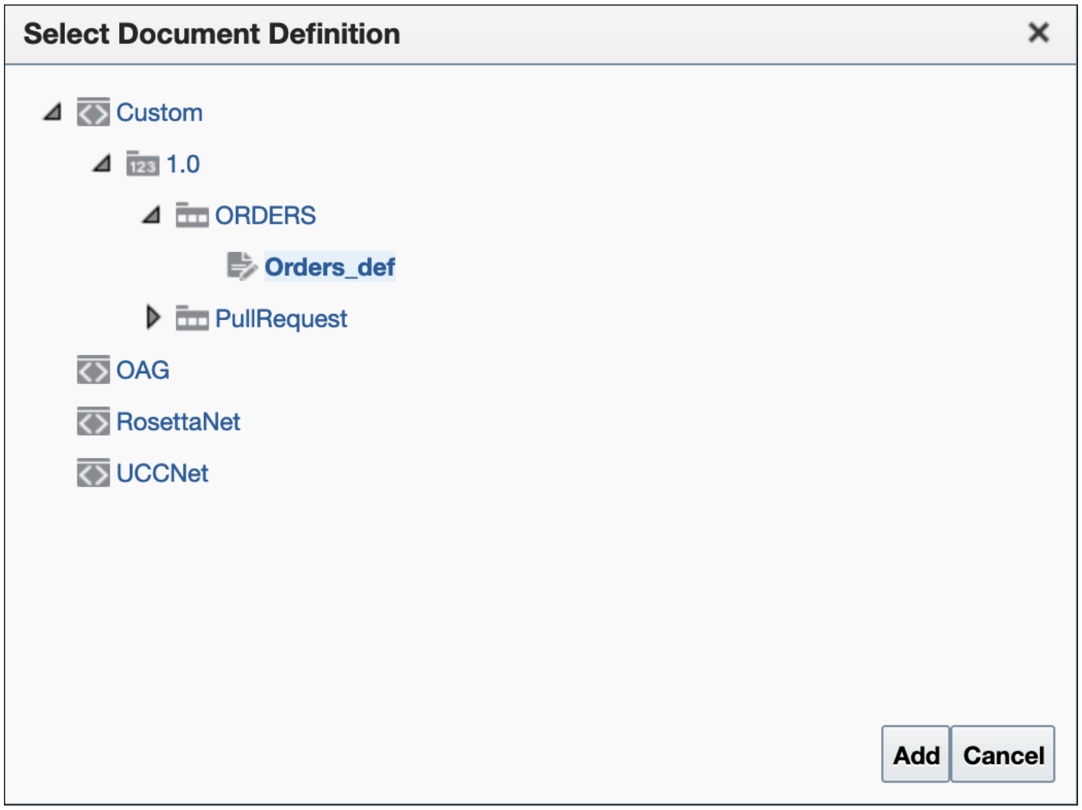
- Add Document Definitions

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. For more information, see Using Document Protocols .

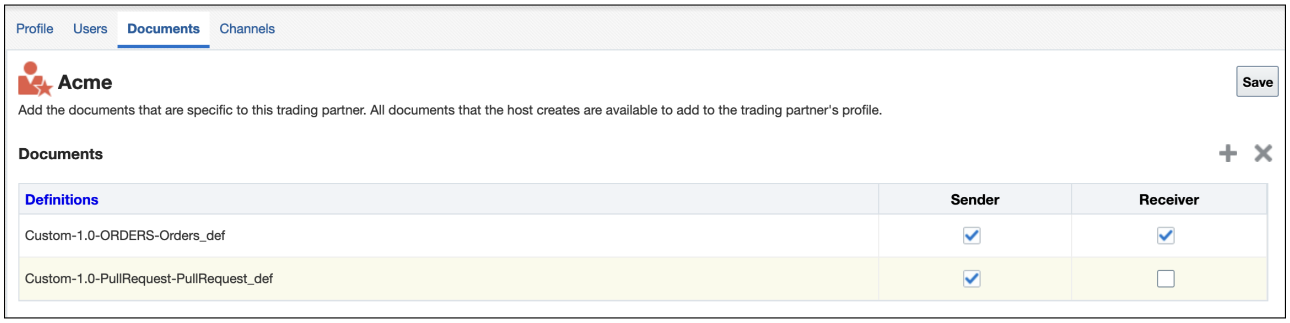
1. Click the **Partners** link.
2. Click the **Documents** tab.
3. Select a trading partner.
4. Click **Add**.
5. Expand the nodes, select a document definition as shown in Figure 5-11, and click **Add**.

Figure 5-11 Selecting a Document Definition



6. For each document listed, identify if the selected partner is the sender or receiver or both, as shown in Figure 5-12.

Figure 5-12 Selecting the Sender and Receiver



Acme

Add the documents that are specific to this trading partner. All documents that the host creates are available to add to the trading partner's profile.

Documents

Definitions	Sender	Receiver
Custom-1.0-ORDERS-Orders_def	✓	✓
Custom-1.0-PullRequest-PullRequest_def	✓	

7. Click **Save**.

See 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.

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 is a typical configuration for a VAN connection; the channel for the VAN can be configured once in the Host Trading Partner, but then used by many different remote Trading Partners that all use the same VAN. This avoids having to create a delivery channel multiple times, 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. For more information, see:

- Setting Configuration Parameters
- 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	Communication Channel
AS4-1.0	AS4-1.0 Applicability Statement 4-specificaiton for using XML over WS-HTTP.	External
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.	External

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

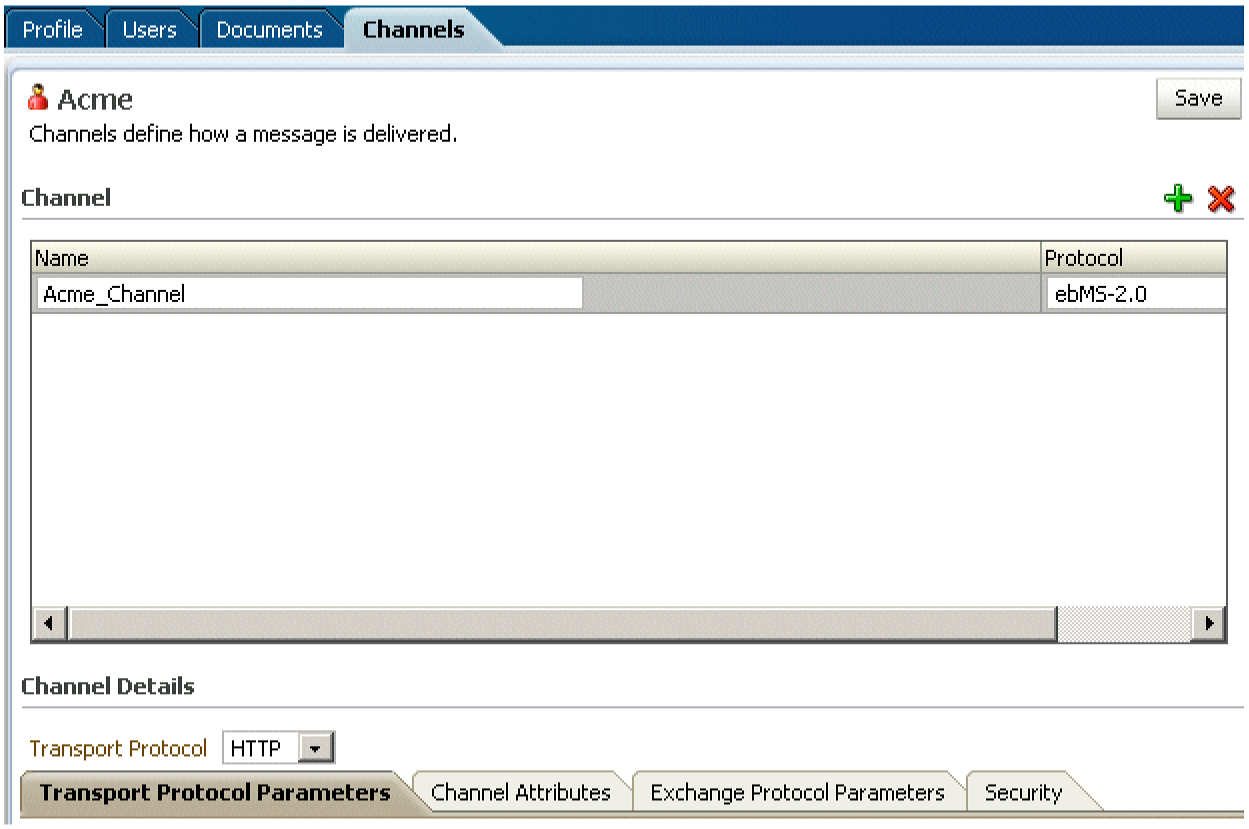
Protocol	Description	Communication Channel
MLLP-1.0 (Remote trading partner only)	Minimum Lower Layer Protocol (MLLP) is a minimalistic OSI-session layer framing protocol. MLLP (and the TCP transport protocol) are available for remote trading partners only. It is used with HL7 or Custom documents. With 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. For more information, see About MLLP.	External
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.	External
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.	External
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.	External
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.	External
Generic File-1.0	Transport by which messages are sent to or received from a file in a local file system.	Internal/External
Generic AQ-1.0	Transport by which messages are sent to or received from Oracle AQ single or multbuttonsumer queues.	Internal/External
Generic FTP-1.0	Transport by which messages are sent to or received from a file at a remote FTP server.	Internal/External
Generic SFTP-1.0	Transport by which messages are sent to or received from a file at a remote SFTP server.	External

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

Protocol	Description	Communication Channel
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`.	Internal/External
Generic HTTP-1.0	Transport by which messages are sent to or received from a Web server.	External
Generic Email-1.0	Transport by which messages are sent to or received from an email server.	External
Generic TCP	Transport that supports customized Exchange Protocol parameters such as Start Block, End Block, Header Length, Message Length Index, and Retain Header.	External
Generic MFT-1.0	Transport by which messages are sent to or received from a file at a remote MFT server.	External

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 5-13.

Figure 5-13 Channel Details Tabs



Acme
Channels define how a message is delivered.

Name	Protocol
Acme_Channel	ebMS-2.0

Transport Protocol: HTTP

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:

- Add a Channel
- Provide Transport Protocol Parameters
- Provide Channel Attributes
- Provide Exchange Protocol Parameters
- Configure Security

Note:

For AS1, B2B supports only SHA1 for signing. MD5 is not supported for AS1 signing.

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.
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...
AS4-1.0	WS-HTTP
AS2-1.1 ebMS-2.0, ebMS-1.0 RosettaNet-V02.00, RosettaNet-01.00 Generic HTTP-1.0	HTTP
AS1-1.0 Generic Email-1.0	Email
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
Generic MFT-1.0	MFT

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 Add a Channel.

   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)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
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 email. 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 email) 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	FTP runs on default port 21, which is not displayed. 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. Per the FTP protocol requirement, ensure that the port is higher than 1023.	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)
Email ID	The email 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)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With

Filename format	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.
	Note:
	In File/FTP channels, if the `filename format` is set then the `directory name format` is ignored.
		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)
FIPS Mode	FIPS mode can be enabled for the channel by selecting this check box. Once selected the SFTP connection to remote server will be established in FIPS mode. When this option is not selected and the user is trying to connect to a remote SFTP server which is running in the FIPS mode, a connection error results. When this option is selected, and the user is trying to connect to a server running in non-FIPS mode, the connection succeeds or fails based on whether the remote server supports the FIPS algorithm.	Generic SFTP
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 email 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) Generic MFT-1.0 (optional)
Path	The absolute directory path where messages are sent from or received.	Generic SFTP (required)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
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)
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. 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 email attachment instead of the typical delivery in which the payload is the message body.	AS1 (optional) Generic Email (optional)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Sequence (Sequencing)	Enable this property when delivering the incoming message in sequence to the back-end application is required. See 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)
Source	The Oracle B2B source name that is created in MFT and added to any transfer with any target. A source specifies from where a file is transferred.	Generic MFT-1.0 (required)
Subject	The subject header of the email message	AS1 (optional) Generic Email (optional)
Subscriber ID	The JMS subscriber ID is required if JMS is communicating with a topic.	Generic JMS
Target	A target specifies to where a file is transferred.	Generic MFT-1.0 (optional)
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)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With

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)

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)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
Transfer	A transfer associates a single source with one or more targets, specifying from where and to where a file is transferred.	Generic MFT-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, for all channels except MFT. For an MFT server, enter a T3 URL with the format `t3://<mft_server_host>:<port>`.	AS2 (required) ebMS-2.0 (required) ebMS-1.0 (required) Generic HTTP (required) RosettaNet-V02.00 (required) RosettaNet-01.10 (required) Generic MFT-1.0 (optional)
Flow Trace EM URL	EM URL of the domain for tracking the instance message flow. The following syntax should be used when providing the Flow Trace EM URL.`:##">http://<host>:<port>#<domain_name>#<domain_type>`	Generic JMS (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)
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) Generic MFT-1.0 (optional)

Table 5-3 (Cont.) Transport Protocol Parameters

Parameter	Description	Protocol Used With
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)

3. Click **Save**.

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 Add a Channel.

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.

Table 5-4 (Cont.) Channel Attributes

Parameter	Description	Protocol Used With
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) Generic MFT-1.0 (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)
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) Generic MFT-1.0 (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 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) Generic MFT-1.0 (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 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) Generic MFT-1.0 (optional)

Table 5-4 (Cont.) Channel Attributes

Parameter	Description	Protocol Used With
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) Generic MFT-1.0 (optional)

3. Click **Save**.

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 Add a Channel.

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)

Table 5-5 (Cont.) Exchange Protocol Parameters

Parameter	Description	Protocol Used With
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)
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)

Table 5-5 (Cont.) Exchange Protocol Parameters

Parameter	Description	Protocol Used With
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)
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 runtime.	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 runtime.	ebMS-2.0 (optional)
Persist HL7 ACK	Indicates if the ACK payload is persisted or not. This is applicable only in the case of Immediate acknowledgments or Discard acknowledgment cases.	MLLP 1.0 (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. It is recommended that you set this parameter when using the ASync mode for the AS2 transport protocol.	AS2 (optional)

Table 5-5 (Cont.) Exchange Protocol Parameters

Parameter	Description	Protocol Used With
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 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**.

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 Add a Channel.

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 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)

Table 5-6 (Cont.) Security Parameters

Parameter	Description	Protocol Used With
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 SHA256 - RSA If **Message Signed** is selected, then select one of the following for ebMS-2.0 and ebMS-1.0: SMIME 3.0 with SHA1 - RSA SMIME 3.0 with SHA256 - RSA 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 SMIME with AES-128 SMIME with AES-192 SMIME with AES-256 Note: For AES-192 and AES-256 support, you must download and apply Java Cryptography Extension (JCE) Unlimited Strength from http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html. Update the JDK/JRE as indicated in the README file and restart the managed servers. 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.

Binary Transfer

Sometimes, you may want to transfer binary content instead of regular XML/Text files. Based on the business needs, binary files, such as PDF, XLS, ZIP, and GIF can also be transferred by using Oracle B2B.

Transfer of binary content can be inbound or outbound.

Outbound Transfer

Outbound binary file transfer in Oracle B2B is used on custom documents over the Generic/AS2 exchange protocols. The File, FTP, SFTP, or JMS internal listening channels can be used for binary file transfer. In case of File, FTP, or SFTP channels, the filename format or the directory name format should contain the document type and document version along with `From Party` and `To Party` information.

In case of JMS internal listening channels, the event name should contain the `BINARYPAYLOAD` property as `true`:

```
eventName=BINARYPAYLOAD:true
```

after you set the preceding property, you need to specify the binary payload (in the form of bytes message) in the JMS Message body:

```
message.setBody(binarypayload);
```

Based on the configured From, To, documents type, and revision, Oracle B2B identifies the agreement and sends it to the trading partner.

Note:

Document validation needs to be turned off at the agreement level for outbound binary transfer.

Inbound Transfer

Inbound binary file transfer in Oracle B2B is used on custom documents over the Generic/AS2 exchange protocols. Incase of File, FTP, or SFTP trading partner listening channels, you need to configure the filename format with From, To, document type and document revision information. Based on the filename format, Oracle B2B identifies the agreement and sends it to the back-end application.

In case of Generic HTTP/AS2, based on the additional HTTP headers, such as `DOCTYPE_NAME` and `DOCTYPE_REVISION`, Oracle B2B identifies the agreement and sends it to the back-end application.

The following is a sample of inbound listening channel filename format:

```
%FROM_PARTY%_%DOCTYPE_NAME%_%DOCTYPE_REVISION%_%TIMESTAMP%.dat
```

The following is a sample filename based on the preceding format:

GlobalChips_ORDERS_1.0.123333333.dat

Note:

Oracle B2B supports filenames with any naming formats, such as 123.dat for Generic Exchange when using File, FTP, or SFTP transport.Oracle B2B processes such files (which do not have their file names based on formats such as `%FROM_PARTY%_%TO_PARTY%_%DOCTYPE%_%DOC_REVISION%`) successfully for outbound and inbound direction by implementing callouts.

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, after 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 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.

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
```

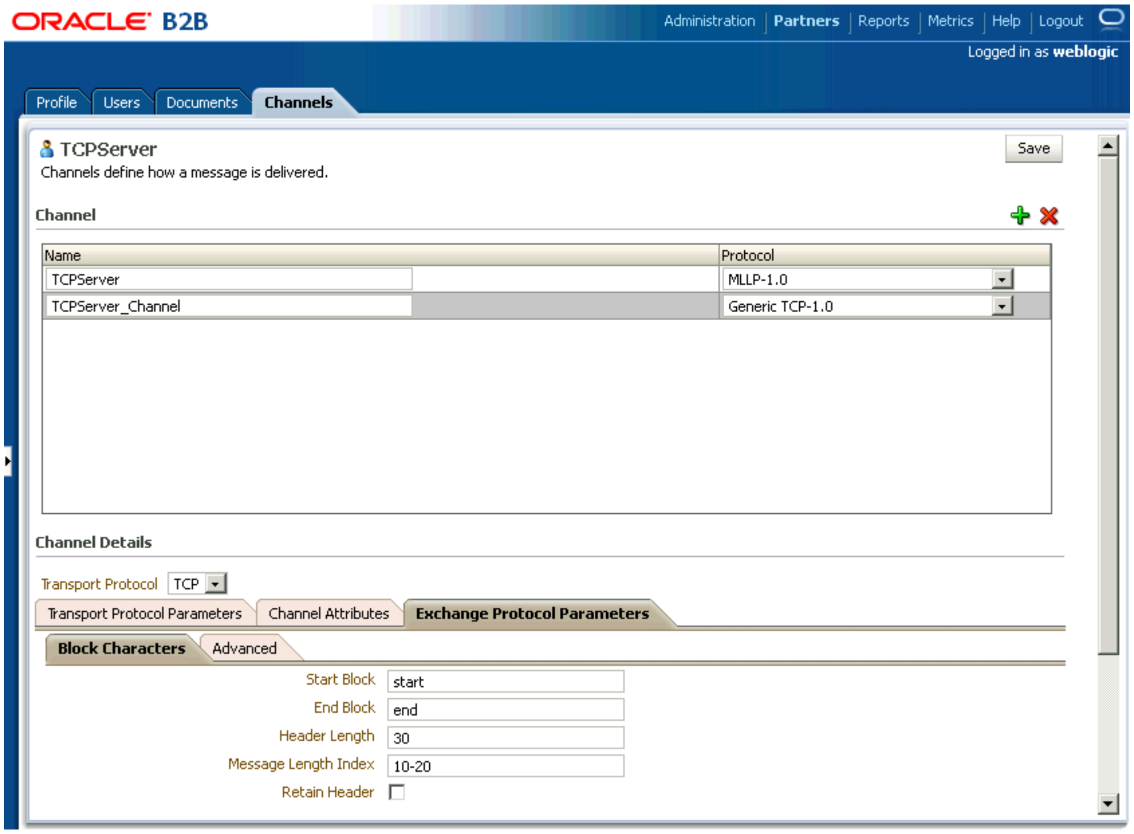
You can also change from a permanent connection to a transient connection.

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 5-14): **Start Block**, **End Block**, **Header Length**, **Message Length Index**, and **Retain Header**.

Figure 5-14 Parameters for Generic TCP


ORACLE B2B
Administration | Partners | Reports | Metrics | Help | Logout
Logged in as weblogic
Profile
Users
Documents
Channels
Save
Transport Protocol Parameters
Channel Attributes
Exchange Protocol Parameters
Block Characters
Advanced

TCPServer

Channels define how a message is delivered.

Channel

Name	Protocol
TCPServer	MLLP-1.0
TCPServer_Channel	Generic TCP-1.0

Channel Details

Transport Protocol: TCP

Start Block	start
End Block	end
Header Length	30
Message Length Index	10-20
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 5-14. After creating the channel, two subtabs appear, with MLLP-specific and generic TCP-specific parameters grouped under them.

See Table 5-5 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.

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.

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 12-1 for details.

Multiple Channel Support for ebMS

For ebMS, Oracle B2B provides multiple channels for Actions, Services, and Service Types. From the multiple channels, Oracle B2B identifies the relevant outbound channel that is associated with a specific Action, Service, and Service Type and uses the channel to communicate with a trading partner.

This feature allows you to override the following parameters of an ebMS channel:

- URL
- Retry Count
- Retry Interval
- Duplicate Elimination
- Ack Signed
- Ack Required

Depending on the Action, Service, and Service Type along with the optional From Role and To Role parameters, the corresponding delivery channel is used for trading partner communication and its associated acknowledgment.

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).

Note:

Sequencing is not supported for transient mode MLLP connections.

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.

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`.

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: trueTimeStamp Format: 43,55,'MMM d yyyy'TimeStamp Offset: +0000TimeStamp 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.

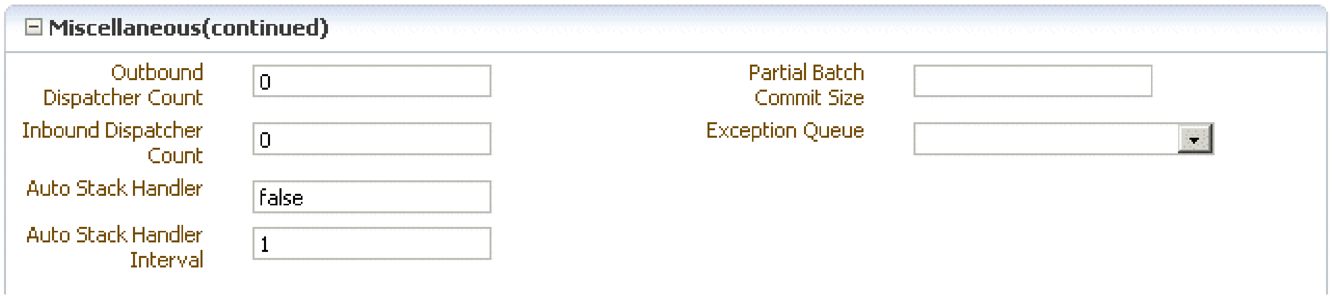
Outbound Message Sequencing for HTTP Delivery Channels

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 5-15.

Figure 5-15 Dispatcher Configuration: Administration > Configuration Tab


Miscellaneous(continued)

Outbound Dispatcher Count	0	Partial Batch Commit Size	
Inbound Dispatcher Count	0	Exception Queue	
Auto Stack Handler	false		
Auto Stack Handler Interval	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.

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

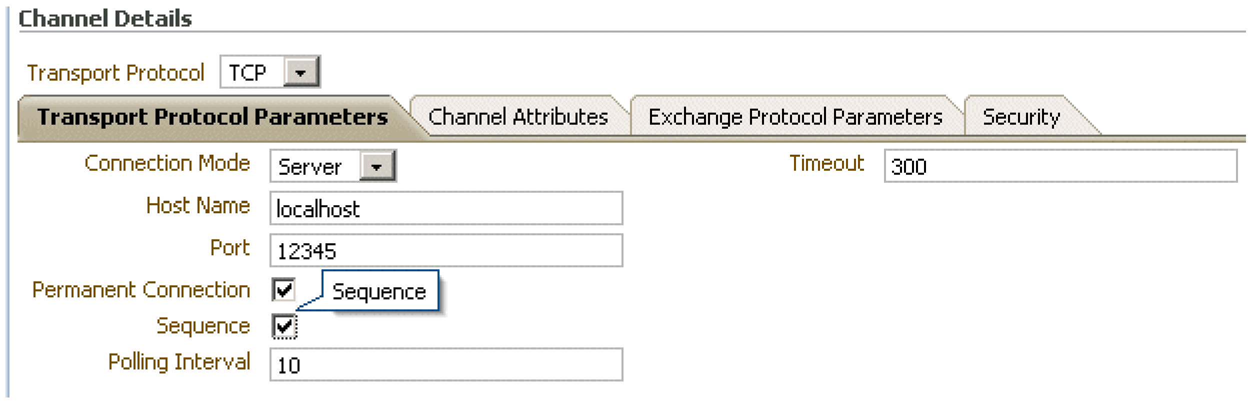
```
ACTION_NAME = TARGET:sequence_target_name
```

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 5-16.

Figure 5-16 Sequencer Configuration


Channel Details

Transport Protocol: TCP

Connection Mode: Server
Host Name: localhost
Port: 12345
Permanent Connection: ✓
Sequence: ✓
Polling Interval: 10
Timeout: 300

To dispatch the sequenced message, configure the **Inbound Dispatcher Count** parameter, as shown in Figure 5-15.

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.

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. after 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 5-15. 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**.

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 guideline 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.

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.

[need sample location in install footprint]

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 12-3 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.

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.

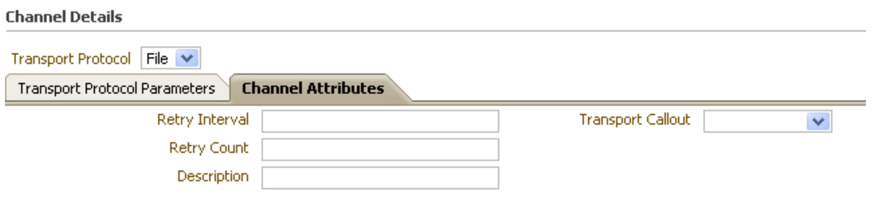
Configuring Delivery Retry Options

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 acknowledgment or functional acknowledgment on time.

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

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.



For any exchange protocol with acknowledgment case (for example MDN in AS2 and acknowledgment 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.

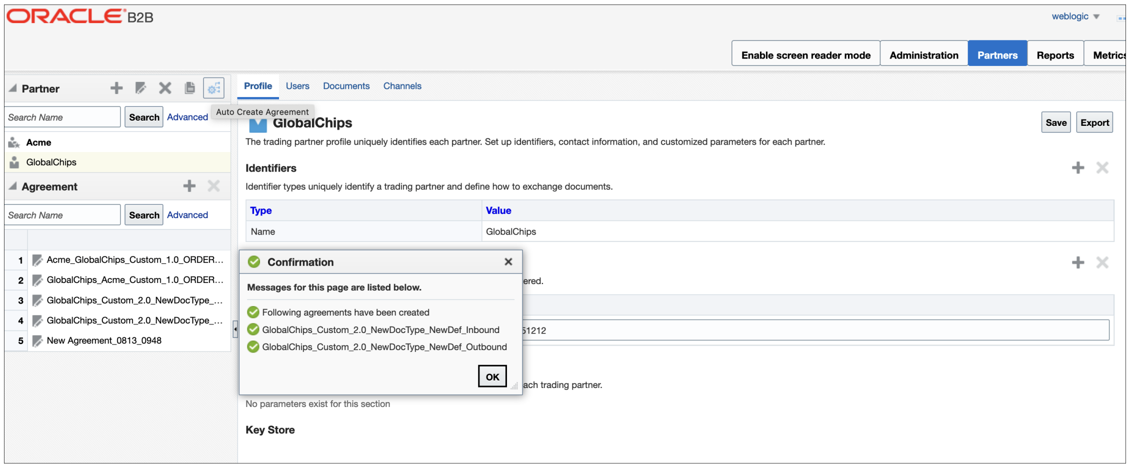
The number of remaining retries in the retry count and retry interval for a specific message can be seen as part of the business message report.

Using the Auto Create Agreement Feature

This feature creates one agreement for each document definition associated with the selected remote trading partner.

In the **Partner** area, shown in Figure 5-17, you can use the **Auto Create Agreement** button to create an agreement for a remote trading partner.

Figure 5-17 The Auto Create Agreement Feature



1 Acme_GlobalChips_Custom_1.0_ORDER...
2 GlobalChips_Acme_Custom_1.0_ORDER...
3 GlobalChips_Custom_2.0_NewDocType_...
4 GlobalChips_Custom_2.0_NewDocType_...
5 New Agreement_0813_0948

GlobalChips

The trading partner profile uniquely identifies each partner. Set up identifiers, contact information, and customized parameters for each partner.

Identifiers

Identifier types uniquely identify a trading partner and define how to exchange documents.

Type	Value
Name	GlobalChips

Confirmation

Messages for this page are listed below.

Following agreements have been created
GlobalChips_Custom_2.0_NewDocType_NewDef_Inbound
GlobalChips_Custom_2.0_NewDocType_NewDef_Outbound

ered.

51212

ach trading partner.

No parameters exist for this section

Key Store

You can further customize the agreement on the **Agreement** tab. See Creating and Deploying Trading Partner Agreements, for more information about the **Agreement** tab.

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.

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 the chapter Scheduling Trading Partner Downtime for information about scheduling and managing trading partner downtime.

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 Creating Custom Trading Partner Parameter Types for more information.

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.
Creating and deploying the agreement are the final steps in the Oracle B2B process flow.

This chapter includes the following sections:

- Introduction to Agreements
- Creating an Agreement
- Deploying an Agreement
- Deleting and Exporting Agreements

For more information, see:

- Managing Deployments for how to export agreements and manage deployment states
- Importing and Exporting Data for how to export agreements

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 Custom 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. 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.

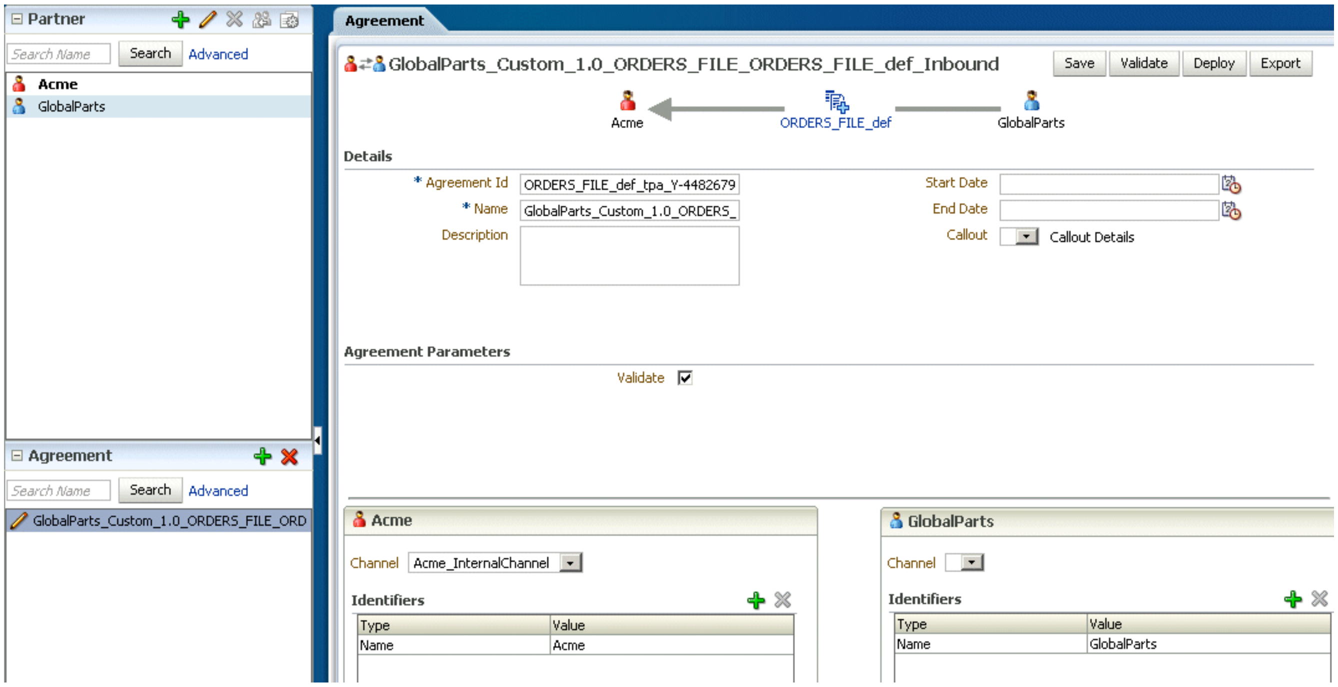
Creating an agreement is the last step in the design of a B2B transaction. Before you create an agreement, you must have already created the document definitions and configured the trading partners. For more information, see Creating Document Definitions and Configuring Trading Partners .

Creating an Agreement

This section shows the overall steps needed to create an agreement between two trading partners.

Figure 6-1 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 6-1 Creating an Agreement



Acme
GlobalParts

GlobalParts_Custom_1.0_ORDERS_FILE_ORD

GlobalParts_Custom_1.0_ORDERS_FILE_ORDERS_FILE_def_Inbound

Details

* Agreement Id	ORDERS_FILE_def_tpa_Y-4482679	Start Date	
* Name	GlobalParts_Custom_1.0_ORDERS_	End Date	
Description		Callout	Callout Details

Agreement Parameters

Validate ☑

Acme

Channel Acme_InternalChannel

Identifiers

Type	Value
Name	Acme

GlobalParts

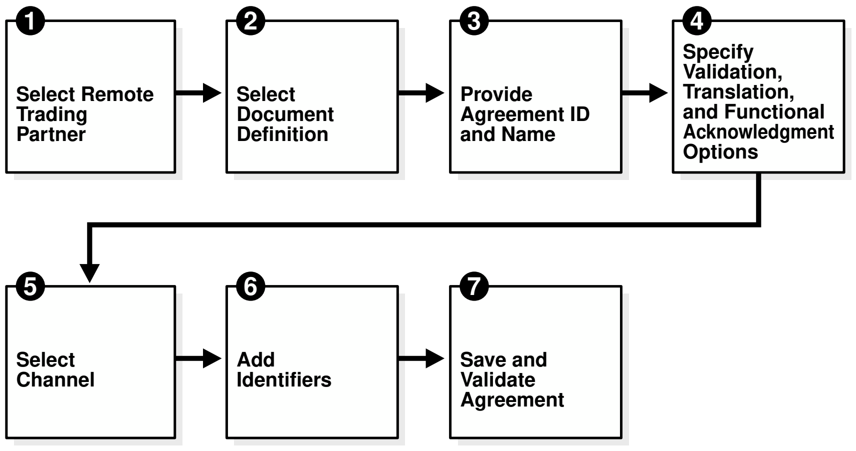
Channel

Identifiers

Type	Value
Name	GlobalParts

Figure 6-2 shows the steps to create an agreement.

Figure 6-2 Steps to Creating an Agreement (Workflow Overview)


1 Select Remote Trading Partner
2 Select Document Definition
3 Provide Agreement ID and Name
4 Specify Validation, Translation, and Functional Acknowledgment Options
5 Select Channel
6 Add Identifiers
7 Save and Validate Agreement

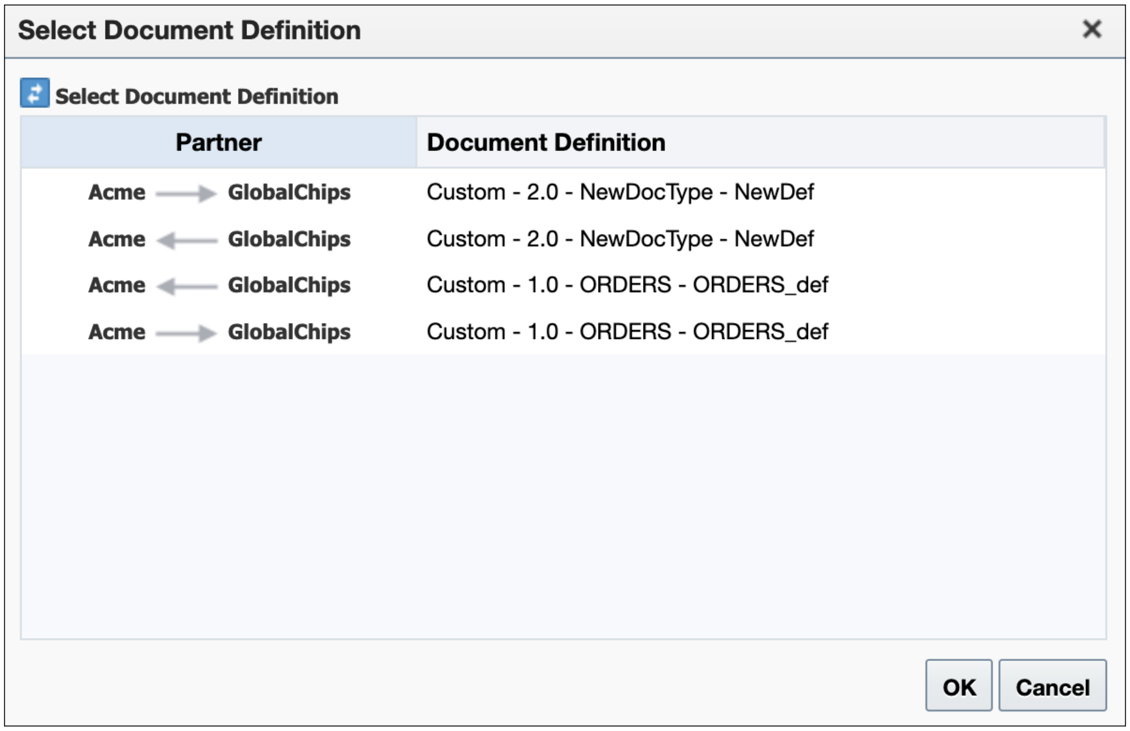
Step 1: Identify the remote trading partner

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** button 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 6-3.

Figure 6-3 Selecting the Document Definition



Partner	Document Definition
Acme → GlobalChips	Custom - 2.0 - NewDocType - NewDef
Acme ← GlobalChips	Custom - 2.0 - NewDocType - NewDef
Acme ← GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def
Acme → GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def

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 6-3)
- For the inbound agreement, select the document definition in which the host trading partner is the receiver (**Acme <-- GlobalChips** in Figure 6-3)

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 guideline file such as XSD.

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

For more information about identifier types, see Creating 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 the appropriate 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 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 listed on the **Administration** > **Deploy** page and ready to be deployed.

Deploying an Agreement

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

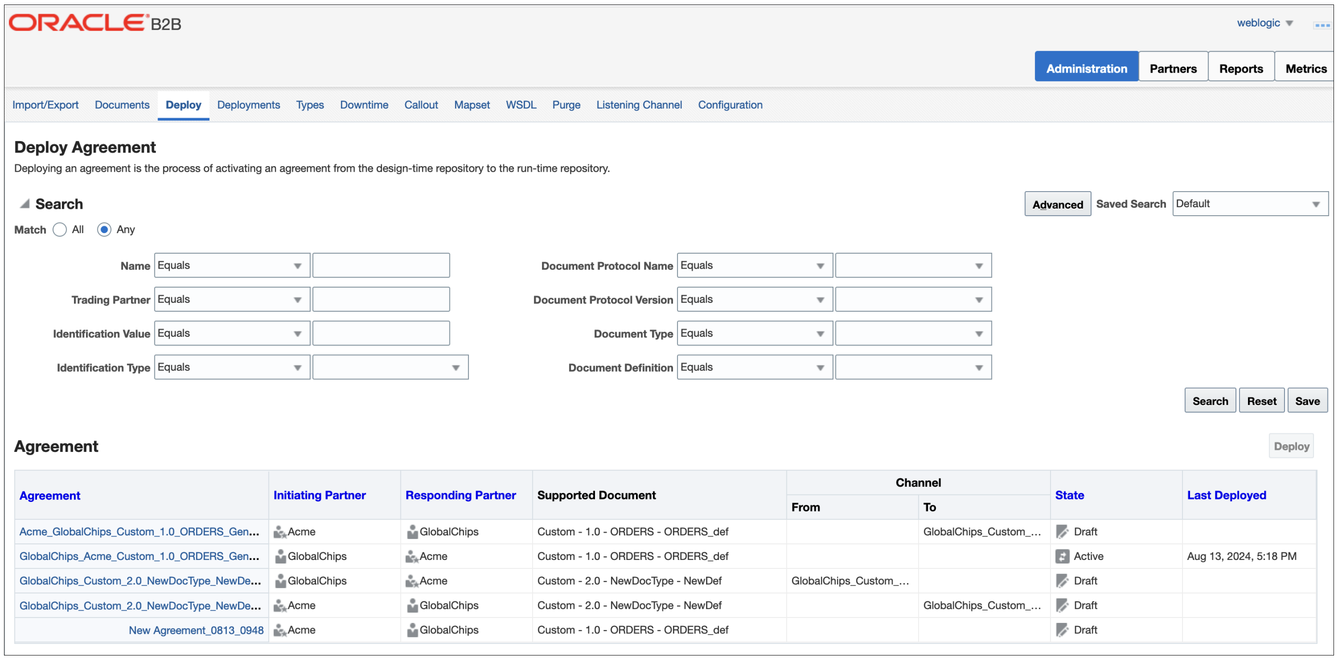
After deploying an agreement, use the **Manage Deployments** tab and the **Reports** tab. See the following topics for more information:

- Managing Deployments
- 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 6-1).
- From the **Administration** > **Deploy** page, as shown in Figure 6-4. Use this option to select multiple agreements to deploy at the same time.

Figure 6-4 The Deploy Tab—Lists Valid Agreements



Deploy Agreement

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

Search

Advanced | Saved Search: Default

Match: ( ) All (●) Any

Name	Equals	Document Protocol Name	Equals
Trading Partner	Equals	Document Protocol Version	Equals
Identification Value	Equals	Document Type	Equals
Identification Type	Equals	Document Definition	Equals

Search | Reset | Save

Agreement

Deploy

Agreement	Initiating Partner	Responding Partner	Supported Document	Channel From	Channel To	State	Last Deployed
Acme_GlobalChips_Custom_1.0_ORDERS_Gen...	Acme	GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def		GlobalChips_Custom_...	Draft	
GlobalChips_Acme_Custom_1.0_ORDERS_Gen...	GlobalChips	Acme	Custom - 1.0 - ORDERS - ORDERS_def			Active	Aug 13, 2024, 5:18 PM
GlobalChips_Custom_2.0_NewDocType_NewDe...	GlobalChips	Acme	Custom - 2.0 - NewDocType - NewDef	GlobalChips_Custom_...		Draft	
GlobalChips_Custom_2.0_NewDocType_NewDe...	Acme	GlobalChips	Custom - 2.0 - NewDocType - NewDef		GlobalChips_Custom_...	Draft	
New Agreement_0813_0948	Acme	GlobalChips	Custom - 1.0 - ORDERS - ORDERS_def			Draft	

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. For more information, see Properties to Set in Fusion Middleware Control.

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**.

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.

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 includes the following sections:

- Importing and Exporting the Design-Time Repository
- What Is Copied When You Import or Export from the Import/Export Tab
- About the Exported File

For information about importing and exporting data using `ant`, see B2B Command-Line Tools .

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.

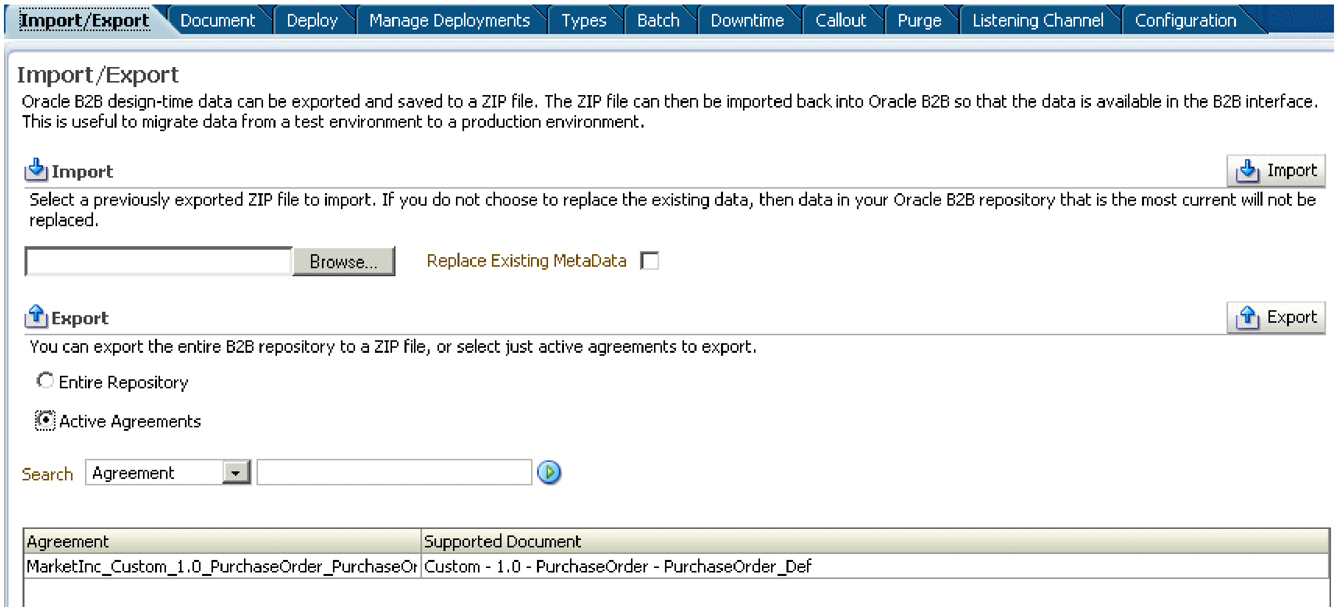
You can also export data from other areas of the Oracle B2B interface:

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

Imported files can use the following document types: Custom, EDI EDIFACT, EDI X12, HL7, and RosettaNet.

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

Figure 7-1   Importing and Exporting Data



Import/Export

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

Import

Select a previously exported ZIP file to import. If you do not choose to replace the existing data, then data in your Oracle B2B repository that is the most current will not be replaced.

Export

You can export the entire B2B repository to a ZIP file, or select just active agreements to export.

Entire Repository [✓ Active Agreements]

Search: Agreement

Agreement	Supported Document
MarketInc_Custom_1.0_PurchaseOrder_PurchaseOr	Custom - 1.0 - PurchaseOrder - PurchaseOrder_Def

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 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:

Note:

Do *not* manually edit exported files.

1. Click the **Administration** link.
2. Click the **Import/Export** tab.
3. Select **Entire Repository** or **Active Agreements**.

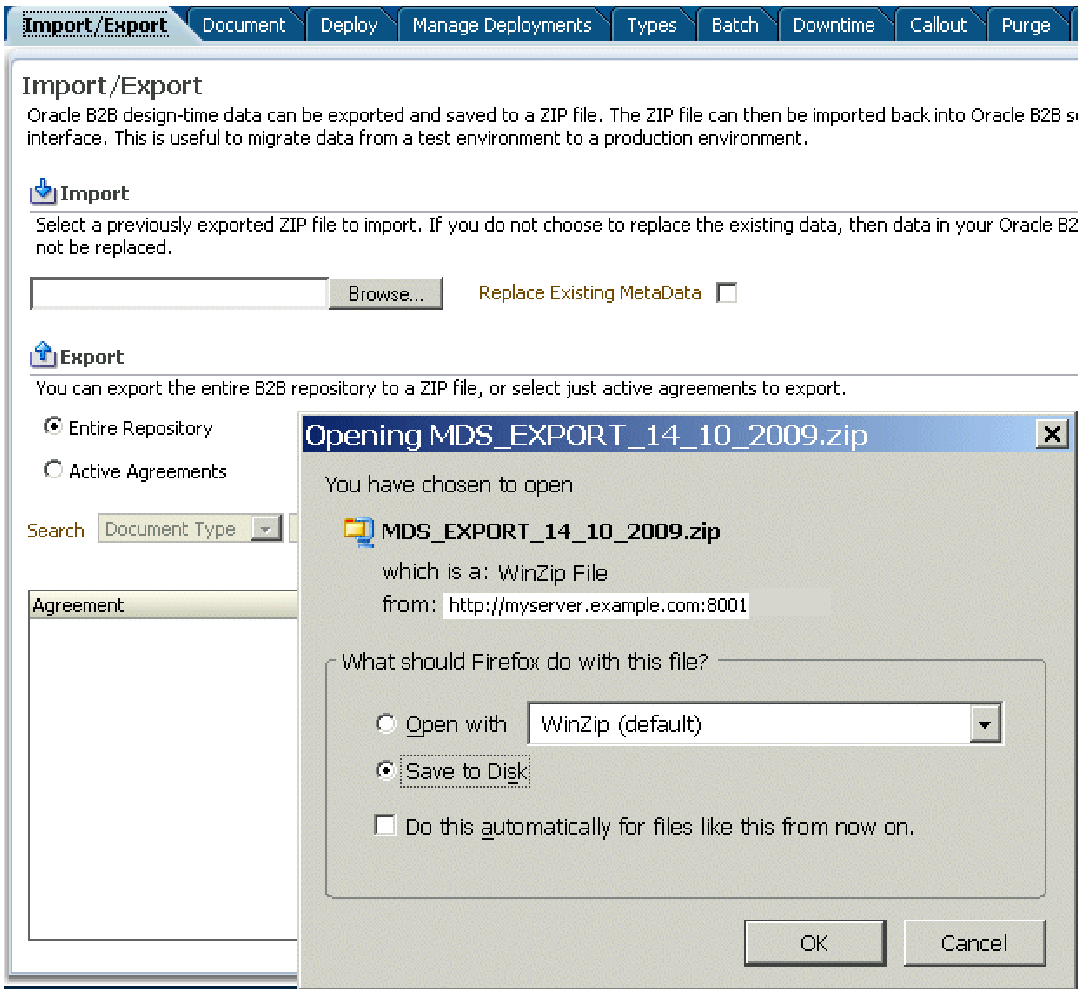
   The entire repository includes all data in the B2B design-time repository—agreements in all states, all trading partner configurations, and so on.

   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



Import/Export

Oracle B2B design-time data can be exported and saved to a ZIP file. The ZIP file can then be imported back into Oracle B2B s
interface. This is useful to migrate data from a test environment to a production environment.

Import

Select a previously exported ZIP file to import. If you do not choose to replace the existing data, then data in your Oracle B2
not be replaced.

Browse... Replace Existing MetaData ☐

Export

You can export the entire B2B repository to a ZIP file, or select just active agreements to export.

◉ Entire Repository
○ Active Agreements

Search Document Type

Agreement

Opening MDS_EXPORT_14_10_2009.zip

You have chosen to open

MDS_EXPORT_14_10_2009.zip

which is a: WinZip File
from: http://myserver.example.com:8001

What should Firefox do with this file?

○ Open with WinZip (default)
◉ Save to Disk
☐ Do this automatically for files like this from now on.

OK Cancel

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 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 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.

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.

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.

This chapter includes the following sections:

- Using the Custom Document Protocol
- Using the OAG Document Protocol
- Using the RosettaNet Document Protocol
- Using the UCCNet Document Protocol
- Changing Document Details
- Using Document Routing IDs

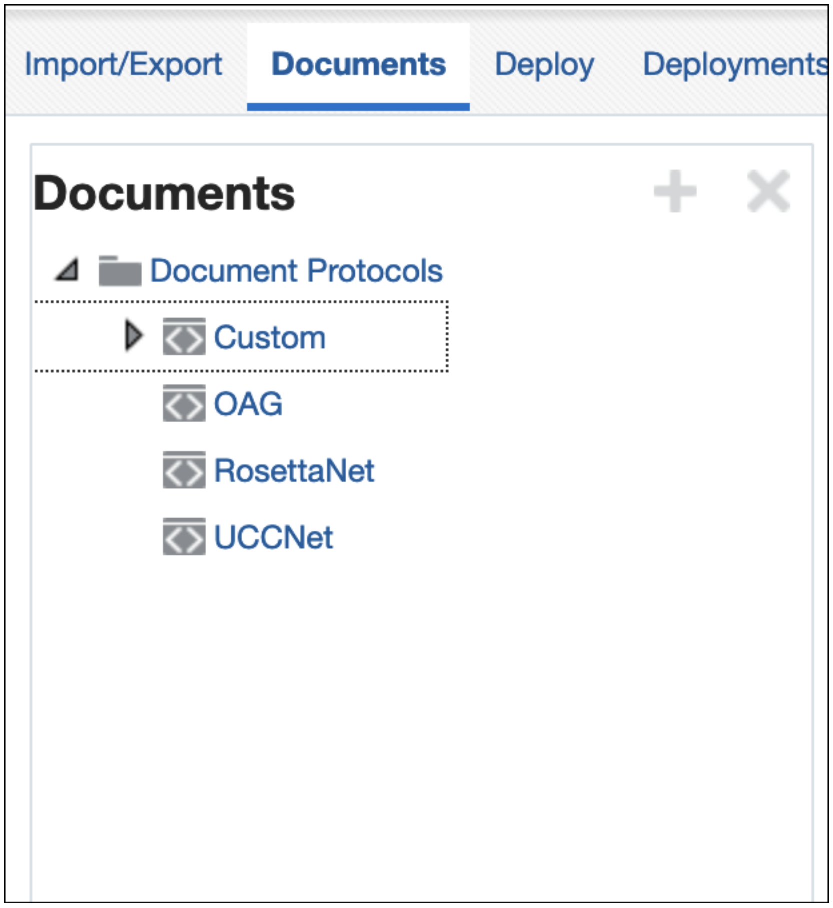
For related information, see the following:

- Creating Guideline Files
- Creating Document Definitions

Using the Custom Document Protocol

Document protocols are shown in Figure 8-1.

Figure 8-1 Oracle B2B Document Protocols



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.

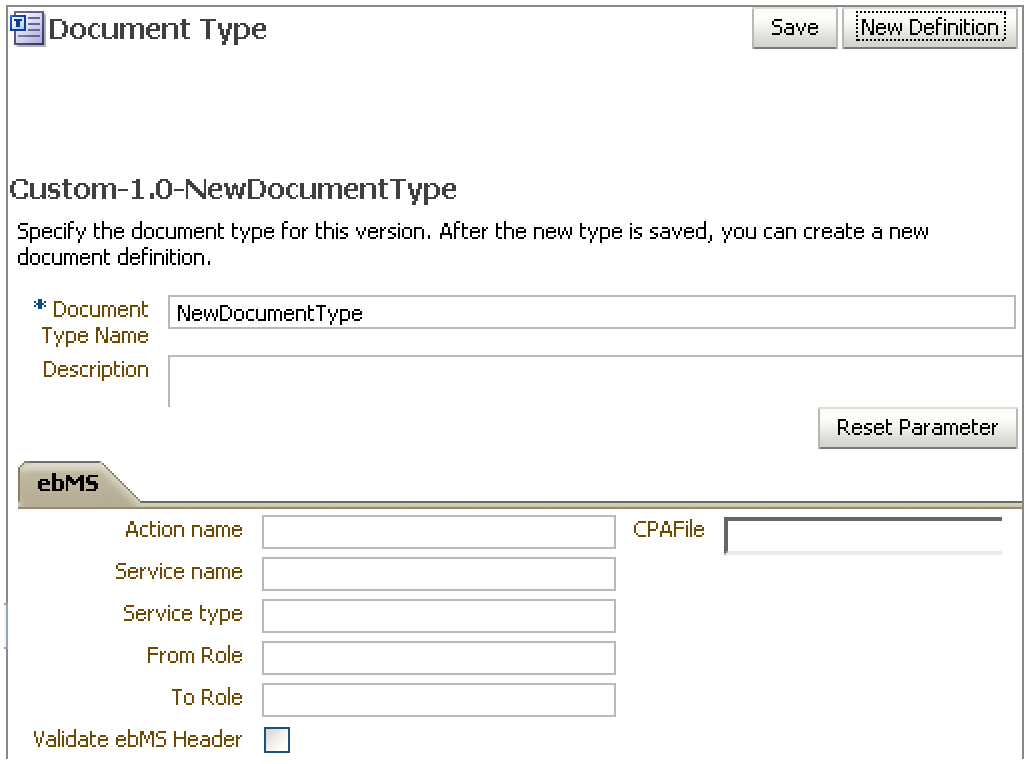
Document Version Parameters

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



Custom-1.0-NewDocumentType

Specify the document type for this version. After the new type is saved, you can create a new document definition.

* Document Type Name: NewDocumentType

Description:

Table 8-1 describes the document type parameters for a Custom document.

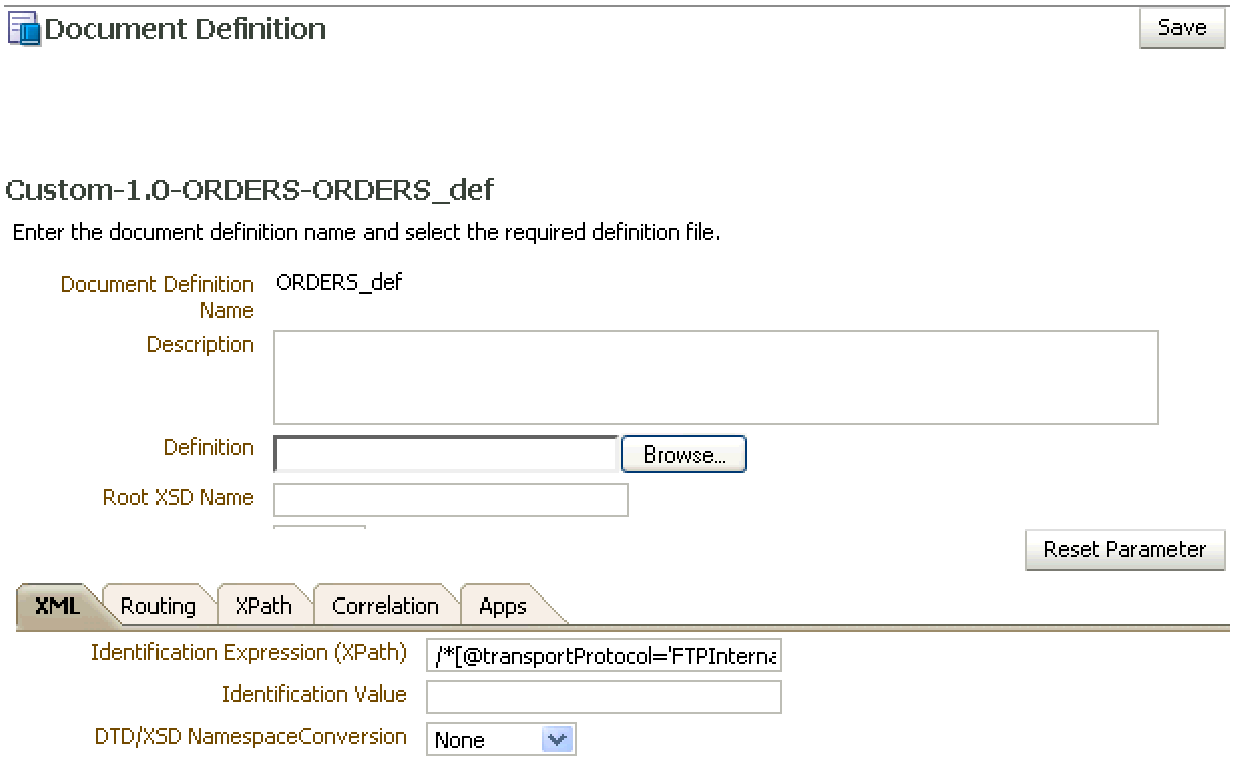
Table 8-1 Document Type Parameters for a Custom Document

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 runtime 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 runtime 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 runtime 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.
Validate 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



Document Definition

Save

Custom-1.0-ORDERS-ORDERS_def

Enter the document definition name and select the required definition file.

Document Definition Name	ORDERS_def
Description	
Definition	Browse...
Root XSD Name	

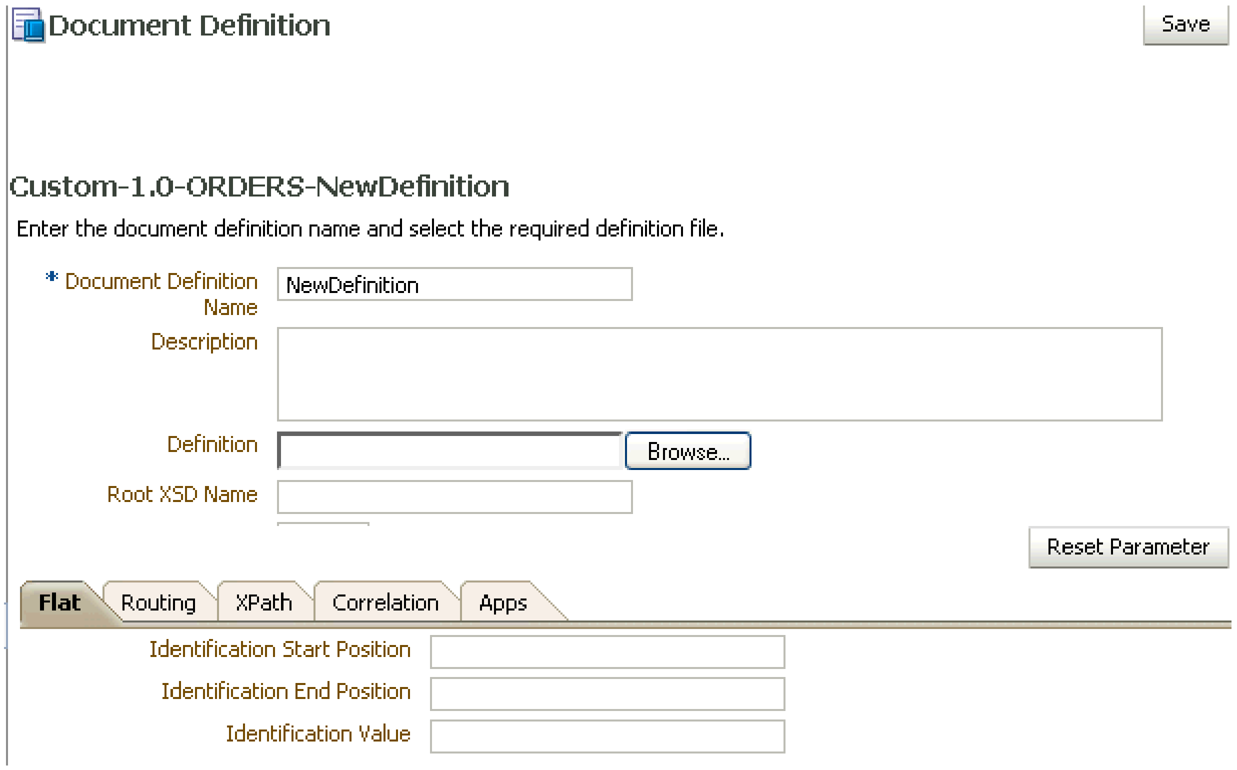
Reset Parameter

XML | Routing | XPath | Correlation | Apps

Identification Expression (XPath)	/*[@transportProtocol='FTPInterna
Identification Value	
DTD/XSD NamespaceConversion	None

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



Document Definition

Save

Custom-1.0-ORDERS-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name	NewDefinition
Description	
Definition	Browse...
Root XSD Name	

Reset Parameter

Flat | Routing | XPath | Correlation | Apps

Identification Start Position	
Identification End Position	
Identification Value	

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	For more information, see 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 (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.

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:

- 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

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>
```

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>
```

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 8-3   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>
```

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`.

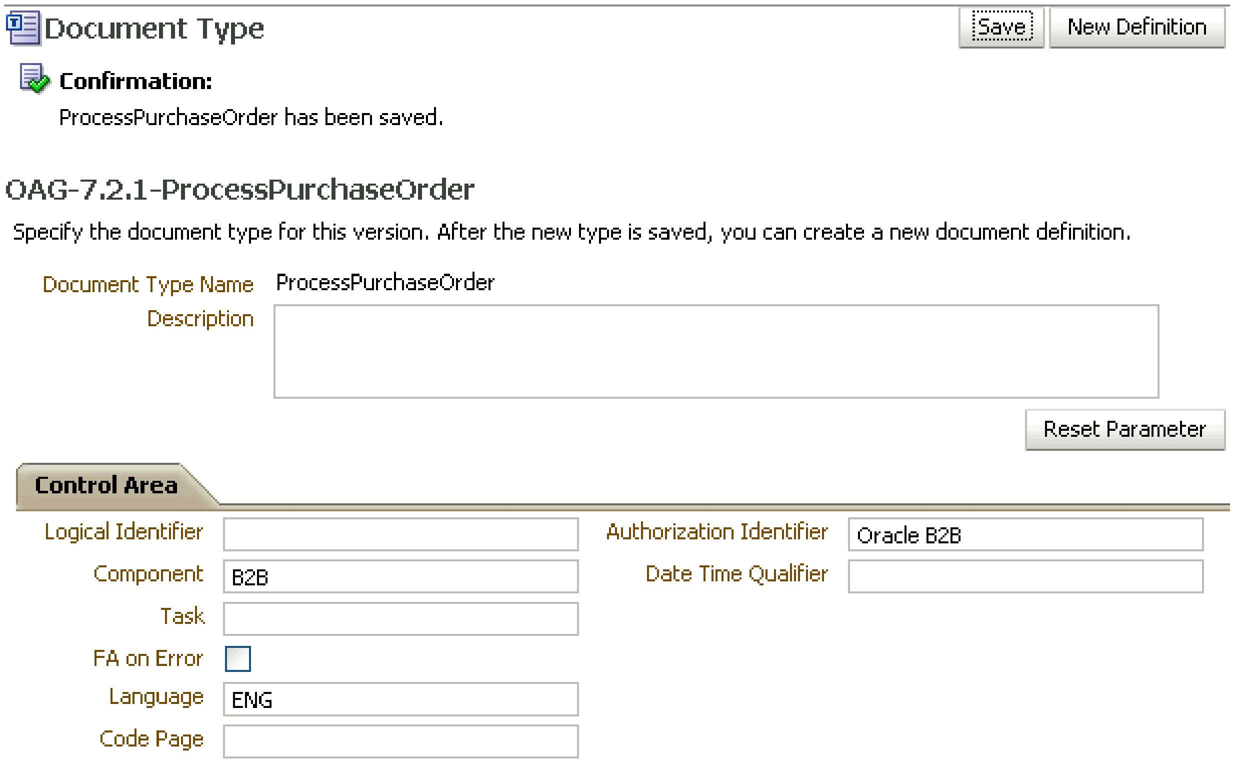
Document Version Parameters

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 8-5 shows the document type parameters for an OAG document.

Figure 8-5 Document Type Parameters for an OAG Document



Confirmation:
ProcessPurchaseOrder has been saved.

OAG-7.2.1-ProcessPurchaseOrder

Specify the document type for this version. After the new type is saved, you can create a new document definition.

Document Type Name	ProcessPurchaseOrder
Description	
Logical Identifier	
Component	B2B
Task	
FA on Error	☐
Language	ENG
Code Page	
Authorization Identifier	Oracle B2B
Date Time Qualifier	

Table 8-3 describes the document type parameters for an OAG document.

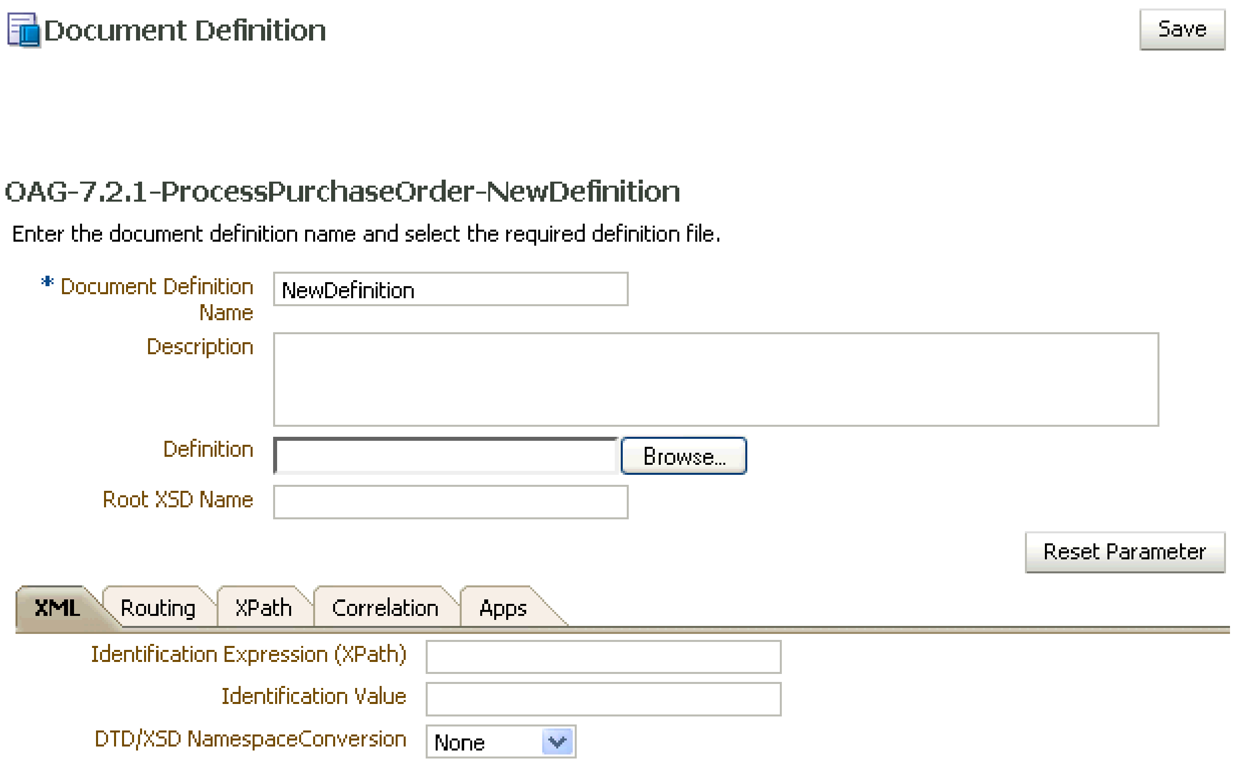
Table 8-3 Document Type Parameters for an OAG Document

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 8-6 shows document definition parameters for an OAG document.

Figure 8-6 Document Definition Parameters for an OAG Document



OAG-7.2.1-ProcessPurchaseOrder-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name	NewDefinition
Description	
Definition	
Root XSD Name	
Identification Expression (XPath)	
Identification Value	
DTD/XSD NamespaceConversion	None

Table 8-4 describes the document definition parameters for an OAG document.

Table 8-4 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	For more information, see 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	-

Table 8-4 (Cont.) Document Definition Parameters for an OAG 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.

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.

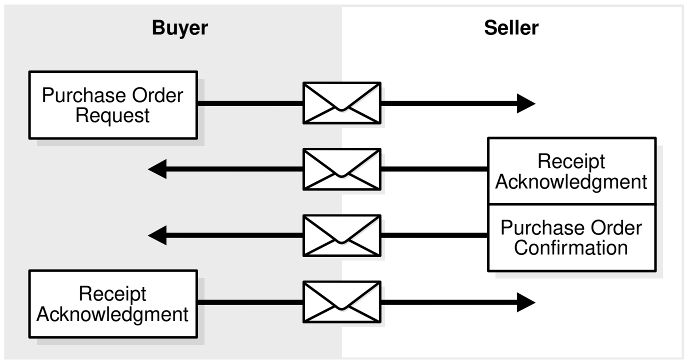
RosettaNet standards are specified by using of the RosettaNet Partner Interface Process (PIP), RosettaNet Dictionaries, and RNIF. Oracle B2B supports all PIPs. (The RosettaNet Technical Dictionary is not supported in Oracle B2B.)

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 8-7.

Figure 8-7 PIP 3A4 Message Exchange Between Buyer and Seller



A PIP sequence combines a cluster, segment, and type. The PIP sequence 3A4, for example, encodes the information shown in Table 8-5.

Table 8-5 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*.
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

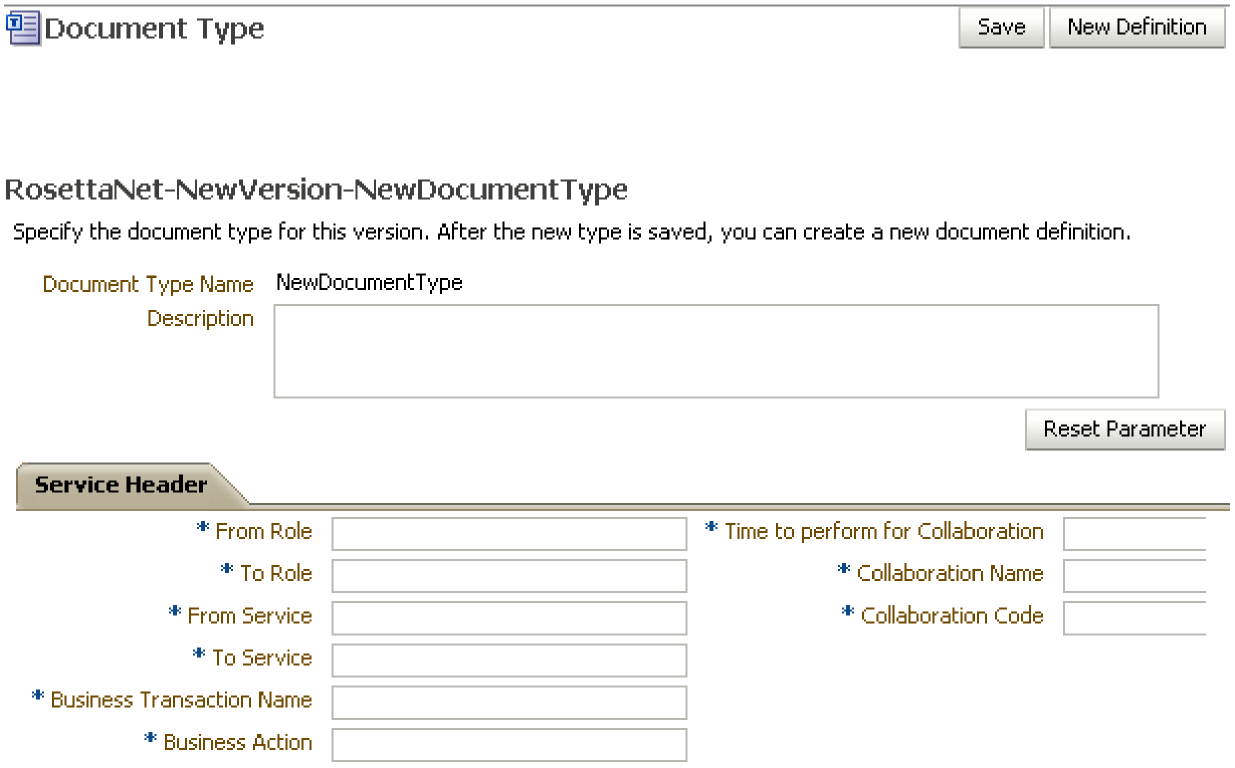
Document Version Parameters

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 8-8 shows document type parameters for a RosettaNet document.

Figure 8-8 Document Type Parameters for a RosettaNet Document



Document Type

Save | New Definition

RosettaNet-NewVersion-NewDocumentType

Specify the document type for this version. After the new type is saved, you can create a new document definition.

Document Type Name	NewDocumentType
Description	

Reset Parameter

Service Header

* From Role		* Time to perform for Collaboration	
* To Role		* Collaboration Name	
* From Service		* Collaboration Code	
* To Service			
* Business Transaction Name			
* Business Action			

Table 8-6 describes document type parameters for a RosettaNet document.

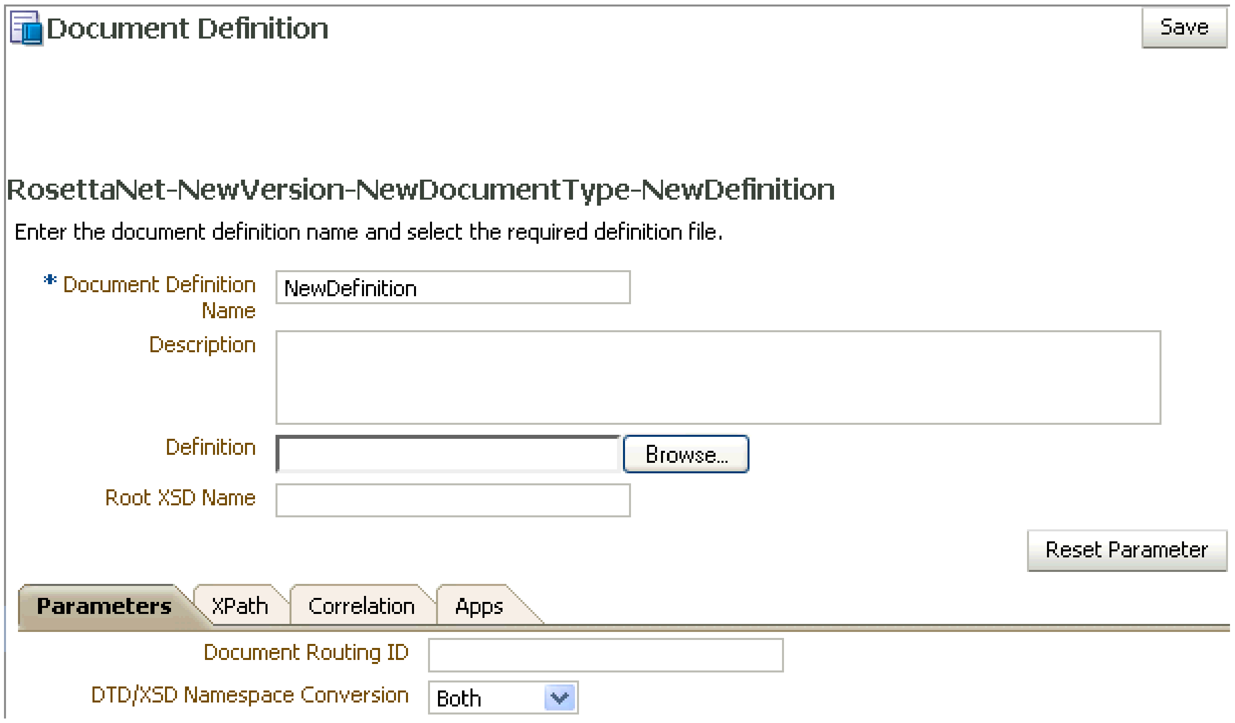
Table 8-6 Document Type Parameters for a RosettaNet Document

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 8-9 shows the document definition parameters for a RosettaNet document.

Figure 8-9 Document Definition Parameters for a RosettaNet Document



RosettaNet-NewVersion-NewDocumentType-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name	NewDefinition
Description	
Definition	
Root XSD Name	
Document Routing ID	
DTD/XSD Namespace Conversion	Both

Table 8-7 describes the document definition parameters for a RosettaNet document.

Table 8-7 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	For more information, see 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.

Table 8-7 (Cont.) Document Definition Parameters for a RosettaNet Document

Parameter	Description
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.

Using the partnerDefinedPIPPayloadBindingId and LocationId Service Header Parameters

Oracle B2B provides support for the following Service Header parameters:

- partnerDefinedPIPPayloadBindingID
- LocationID

Configuring partnerDefinedPIPPayloadBindingID

You can configure the partnerDefinedPIPPayloadBindingID parameter by using the Service Header section under DocumentType in the Oracle B2B console.

Configuring locationID

To configure Rosettanet sender and receiver LocationID using the Oracle B2B console:

1. Create a Trading Partner Identification Type with the value `RN Location ID`.

   Note:

   The value is case-sensitive.

2. Use the `RN Location ID` Identifier Type to create Trading Partner Identifiers for both the host and remote trading partners.
3. Include these new Trading Partner Identifiers of the host and remote trading partners into the Agreement (in addition to usual DUNS Identifier.)
4. Deploy the agreement.

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 data types, 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 data types, 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.

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.

The UCCNet document types supported in Oracle B2B are list below.

- registerCommand
- confirmCommand
- linkCommand
- checkComplianceCommand
- documentCommand
- documentIdentificationCommand
- notificationStateCommand
- queryCommand
- registerLinkCommand
- publicationCommand
- publishCommand
- catalogueItemMaintenanceCommand
- priceCommand
- validateCommand

- registerOwnershipCommand
- subscriptionCommand
- notifyCommand
- response

Document Version Parameters

No parameters need to be set when you create the document version for a UCCNet document.

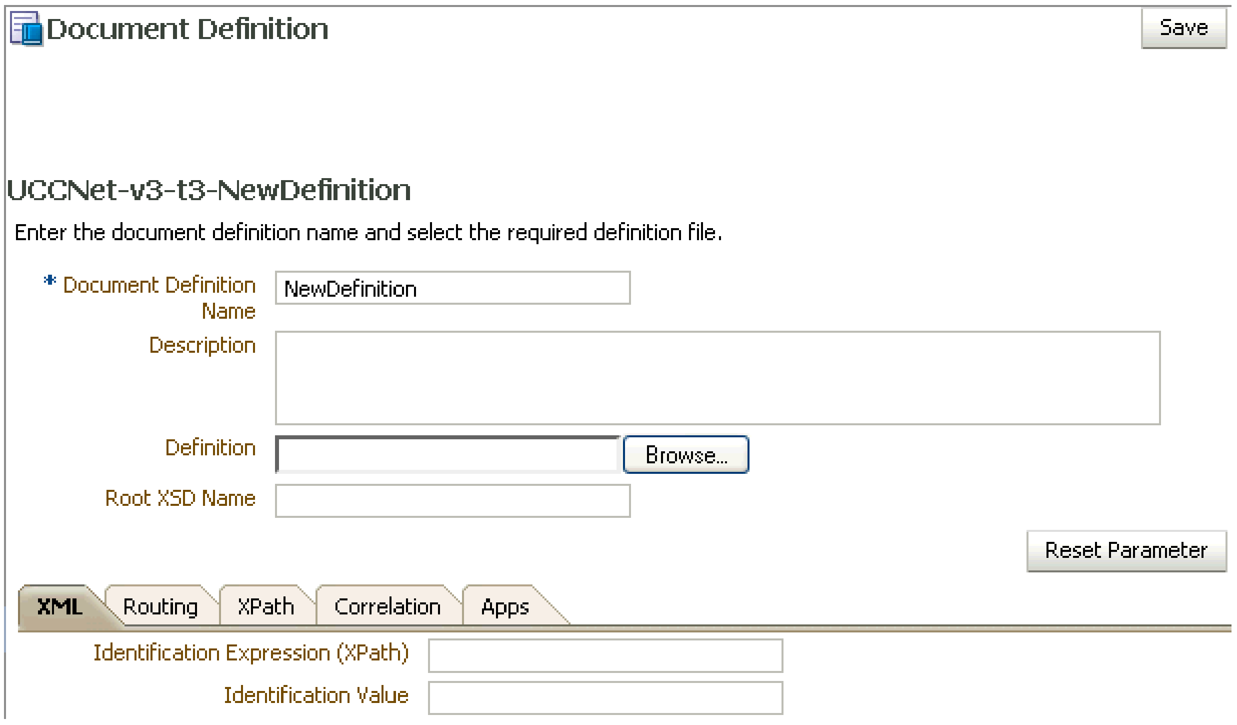
Document Type Parameters

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 8-10 shows document definition parameters for a UCCNet document.

Figure 8-10 Document Definition Parameters for a UCCNet Document


Document Definition

UCCNet-v3-t3-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name	NewDefinition
Description	
Definition	
Root XSD Name	

Identification Expression (XPath)	
Identification Value	

Table 8-8 describes the document definition parameters for a UCCNet document.

Table 8-8 Document Definition Parameters for a UCCnet Document

Parameter	Description
XML Tab	The following parameters are on the XML tab.
Identification Expression (XPath)	Locates a node in the XML payload

Table 8-8 (Cont.) Document Definition Parameters for a UCCnet Document

Parameter	Description
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	The following parameters are on the Routing tab.
Document Routing ID	Sets the consumer name to the back-end application.
XPath Tab	For more information, see 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	The following parameters are on the 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	The following parameters are on the Apps tab.
Document	The name of the internal application document.
Action	A sub-classification within the document.
XSLTFile	The name of the XSLT file.

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.

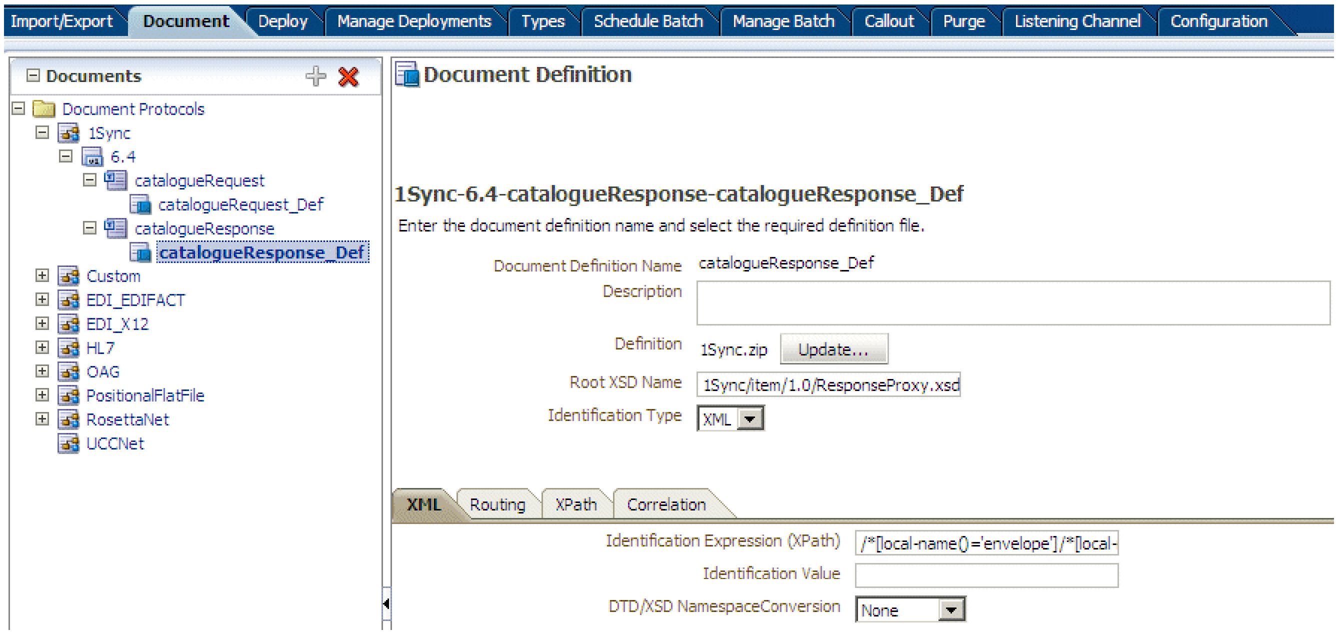
Use the Custom document protocol or the UCCNet document protocol to create a 1Sync XML document.

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 8-11 shows a document definition for a 1Sync document, using the Custom document protocol.

Figure 8-11 1Sync Document Definition



1Sync-6.4-catalogueResponse-catalogueResponse_Def

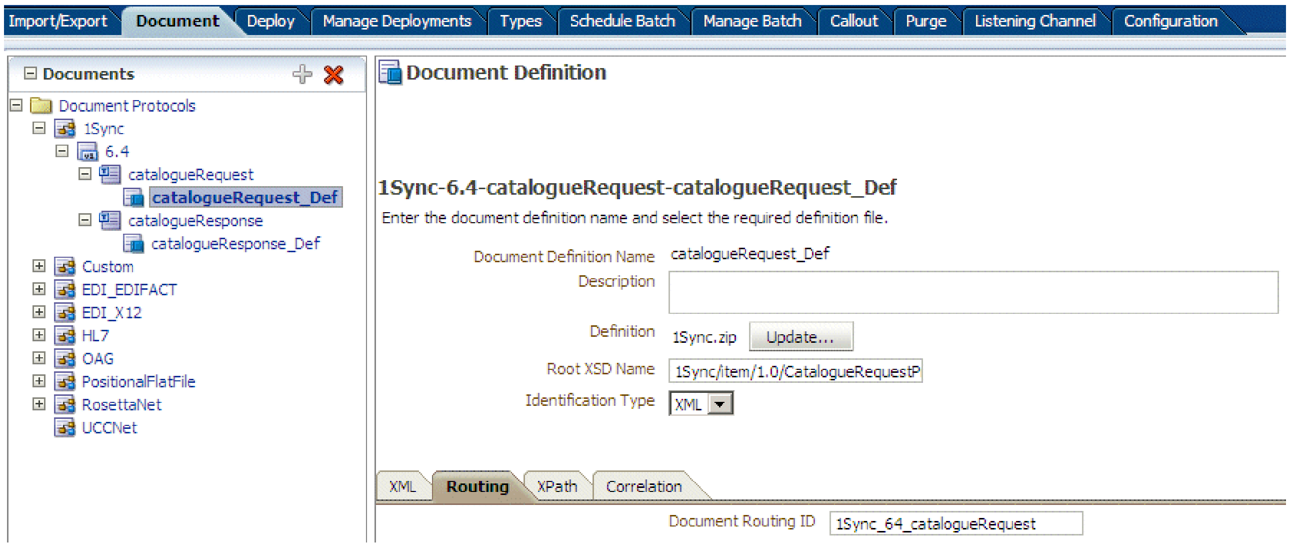
Enter the document definition name and select the required definition file.

Document Definition Name: catalogueResponse_Def
Description:
Definition: 1Sync.zip
Root XSD Name: 1Sync/item/1.0/ResponseProxy.xsd
Identification Type: XML
Identification Expression (XPath): /*[local-name()='envelope']/*[local-
Identification Value:
DTD/XSD NamespaceConversion: None

You can correlate 1Sync request and response messages as follows:

- Use the document routing ID on the **Routin**g tab. The routing ID **1Sync_64_catalogueRequest** is shown Figure 8-12.

Figure 8-12 The Routing Tab for a 1Sync Document Definition



1Sync-6.4-catalogueRequest-catalogueRequest_Def

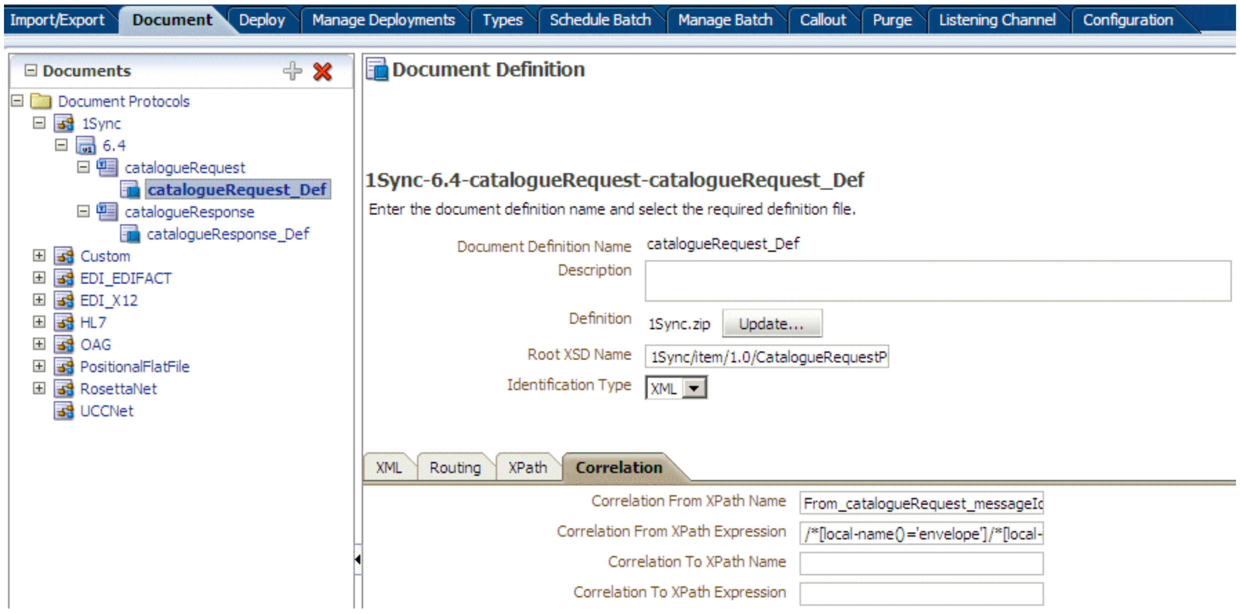
Enter the document definition name and select the required definition file.

Document Definition Name: catalogueRequest_Def
Description:
Definition: 1Sync.zip
Root XSD Name: 1Sync/item/1.0/CatalogueRequestP
Identification Type: XML
Document Routing ID: 1Sync_64_catalogueRequest

- Use correlation parameters on the **Correlation** tab. The following parameters and values are shown in Figure 8-13.
  - **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 8-13 The Correlation Tab for a 1Sync Document Definition



1Sync-6.4-catalogueRequest-catalogueRequest_Def

Enter the document definition name and select the required definition file.

Document Definition Name	catalogueRequest_Def
Description	
Definition	1Sync.zip
Root XSD Name	1Sync/item/1.0/CatalogueRequestP
Identification Type	XML

Correlation From XPath Name	From_catalogueRequest_messageIc
Correlation From XPath Expression	/*[local-name()='envelope']/*[local-
Correlation To XPath Name	
Correlation To XPath Expression	

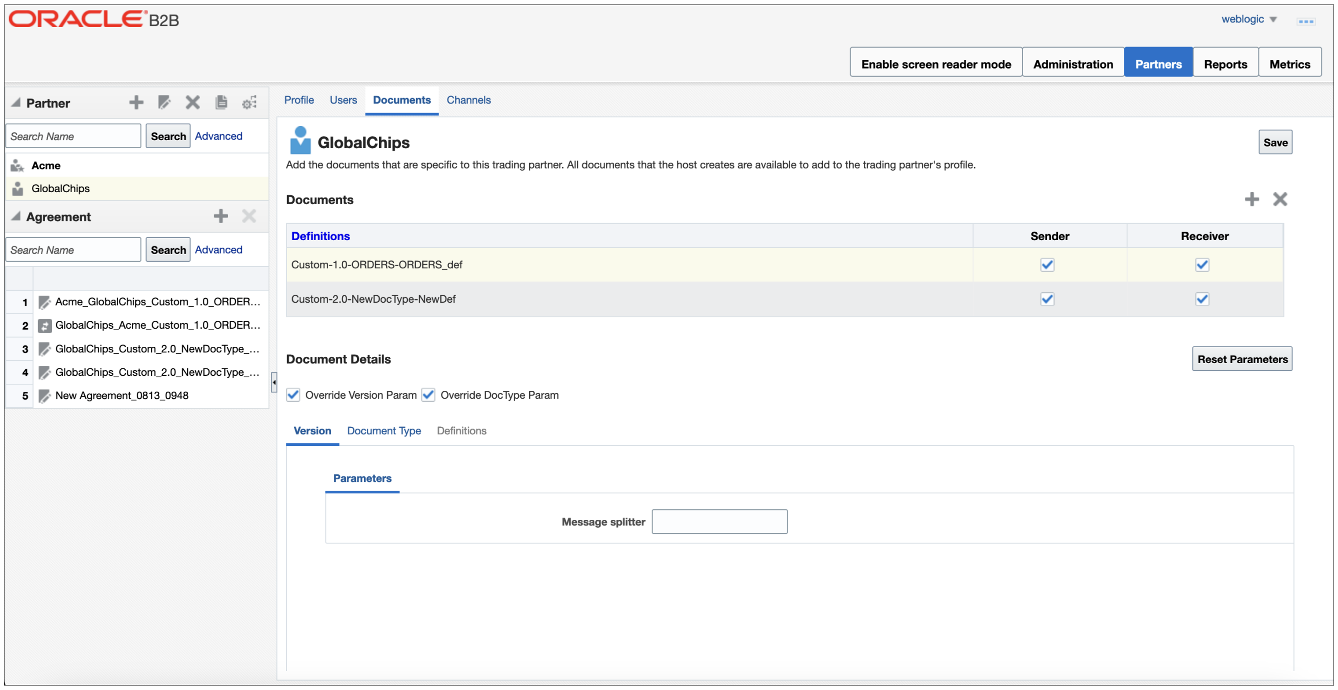
For more information, see Using the Custom Document Protocol.

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.
For more information, see Restricting Access to Document Types.

Figure 8-14 shows the **Version** tab in the **Document Details** section, where parameters for the document protocol version can be changed.

Figure 8-14 Changing Document Details



GlobalChips

Add the documents that are specific to this trading partner. All documents that the host creates are available to add to the trading partner's profile.

Documents

Definitions	Sender	Receiver
Custom-1.0-ORDERS-ORDERS_def	✓	✓
Custom-2.0-NewDocType-NewDef	✓	✓

1	Acme_GlobalChips_Custom_1.0_ORDER...
2	GlobalChips_Acme_Custom_1.0_ORDER...
3	GlobalChips_Custom_2.0_NewDocType_...
4	GlobalChips_Custom_2.0_NewDocType_...
5	New Agreement_0813_0948

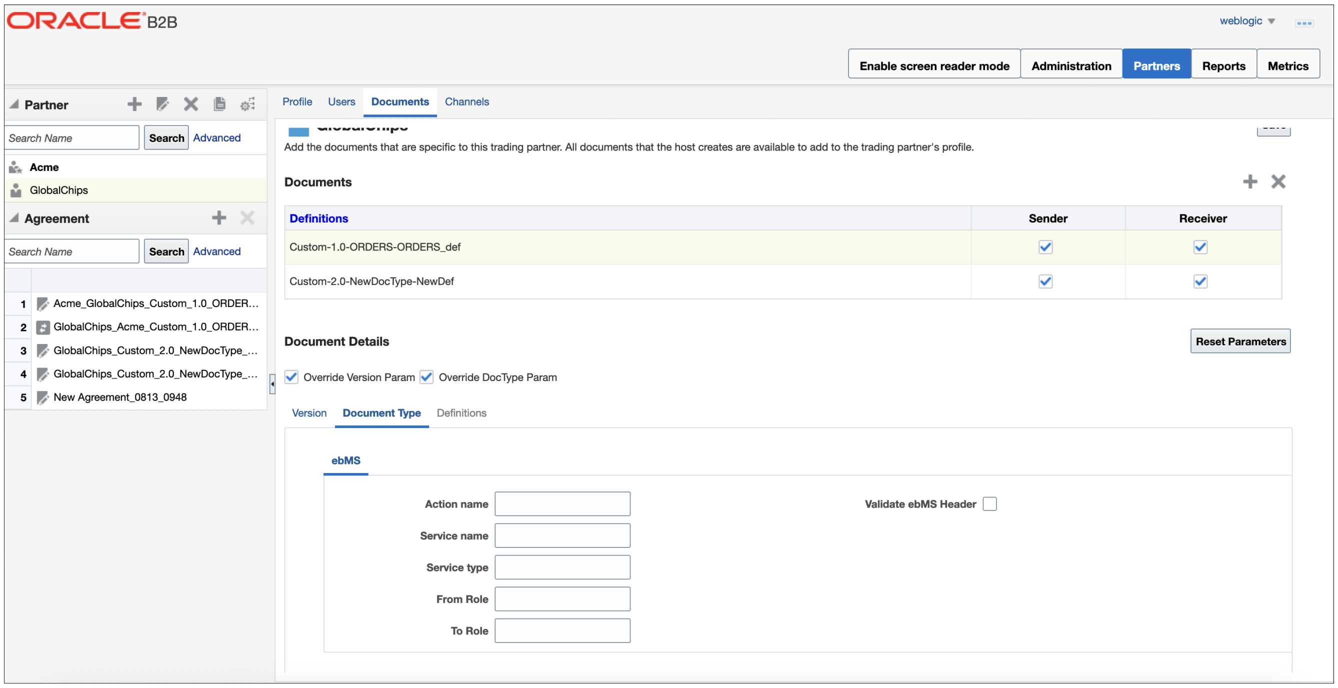
Document Details

✓ Override Version Param ✓ Override DocType Param

Message splitter

Figure 8-15 shows the **Document Type** tab, where parameters for the document type can be changed.

Figure 8-15 Changing Document Details



Add the documents that are specific to this trading partner. All documents that the host creates are available to add to the trading partner's profile.

Documents

Definitions	Sender	Receiver
Custom-1.0-ORDERS-ORDERS_def	✓	✓
Custom-2.0-NewDocType-NewDef	✓	✓

1	Acme_GlobalChips_Custom_1.0_ORDER...
2	GlobalChips_Acme_Custom_1.0_ORDER...
3	GlobalChips_Custom_2.0_NewDocType_...
4	GlobalChips_Custom_2.0_NewDocType_...
5	New Agreement_0813_0948

Document Details

✓ Override Version Param ✓ Override DocType Param

Action name

Service name

Service type

From Role

To Role

Validate ebMS Header ☐

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**.

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.

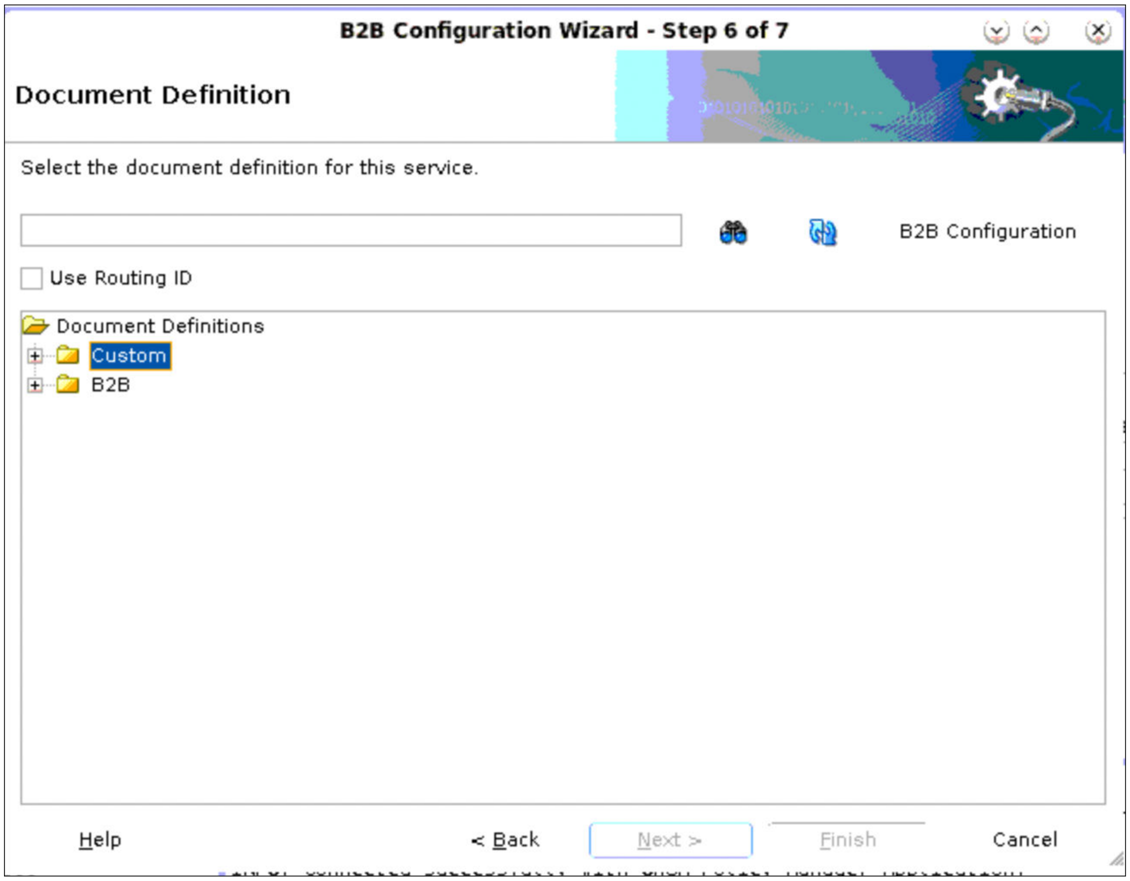
Changing Document Definitions After Importing Metadata

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.

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 8-16.

Figure 8-16   Document Routing ID Option in Oracle JDeveloper



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 runtime data that is used for runtime transactions.

The chapter includes the following sections:

- Introduction to Agreement Deployment States
- Managing Deployed Agreements

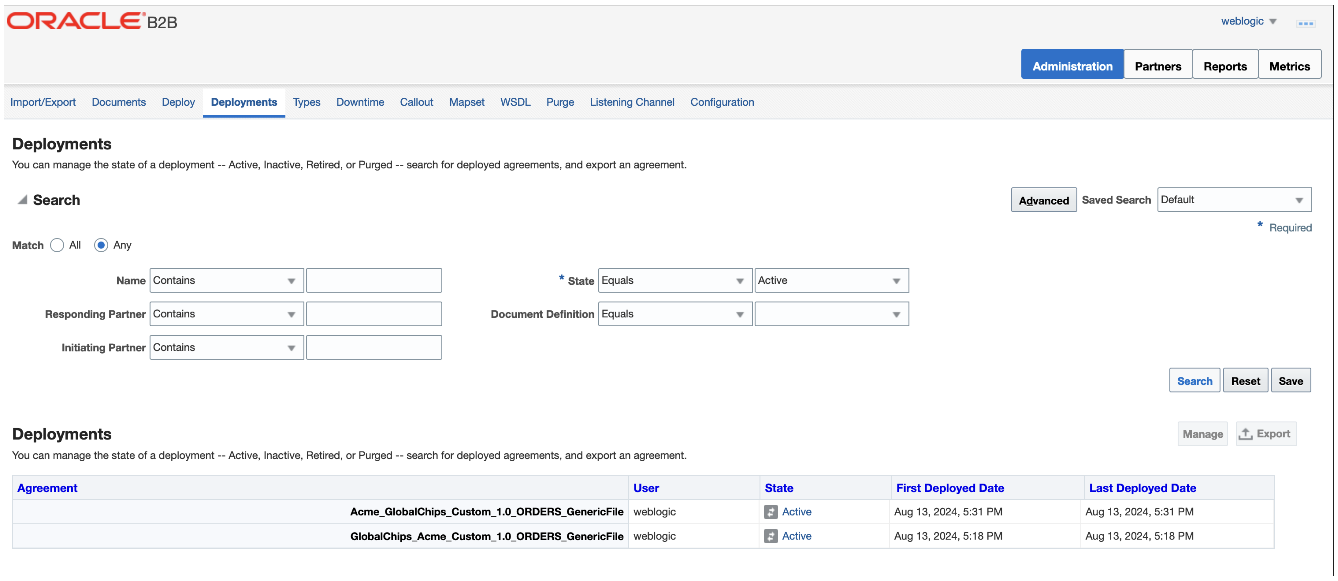
For more information about how to deploy an agreement, see Creating and Deploying Trading Partner Agreements.

Introduction to Agreement Deployment States

You can manage the state of a deployment—Active, Inactive, Retired, or Purged. You can also search on the deployed agreements in the runtime repository, as well as export an agreement.

Figure 9-1 shows the Manage Deployment tab.

Figure 9-1 Managing a Deployed Agreement



Deployments

You can manage the state of a deployment -- Active, Inactive, Retired, or Purged -- search for deployed agreements, and export an agreement.

Search

Saved Search: Default

* Required

Match: All [✓ Any]

Name	Contains			* State	Equals	Active
Responding Partner	Contains			Document Definition	Equals	
Initiating Partner	Contains					

Deployments

You can manage the state of a deployment -- Active, Inactive, Retired, or Purged -- search for deployed agreements, and export an agreement.

Agreement	User	State	First Deployed Date	Last Deployed Date
Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	weblogic	Active	Aug 13, 2024, 5:31 PM	Aug 13, 2024, 5:31 PM
GlobalChips_Acme_Custom_1.0_ORDERS_GenericFile	weblogic	Active	Aug 13, 2024, 5:18 PM	Aug 13, 2024, 5:18 PM

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.
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.

Searching for Deployed Agreements

You can search for deployed agreements using search parameters.

The search parameters aredescribed in Table 9-3.

Table 9-3 Search Parameters for Searching on Deployed Agreements

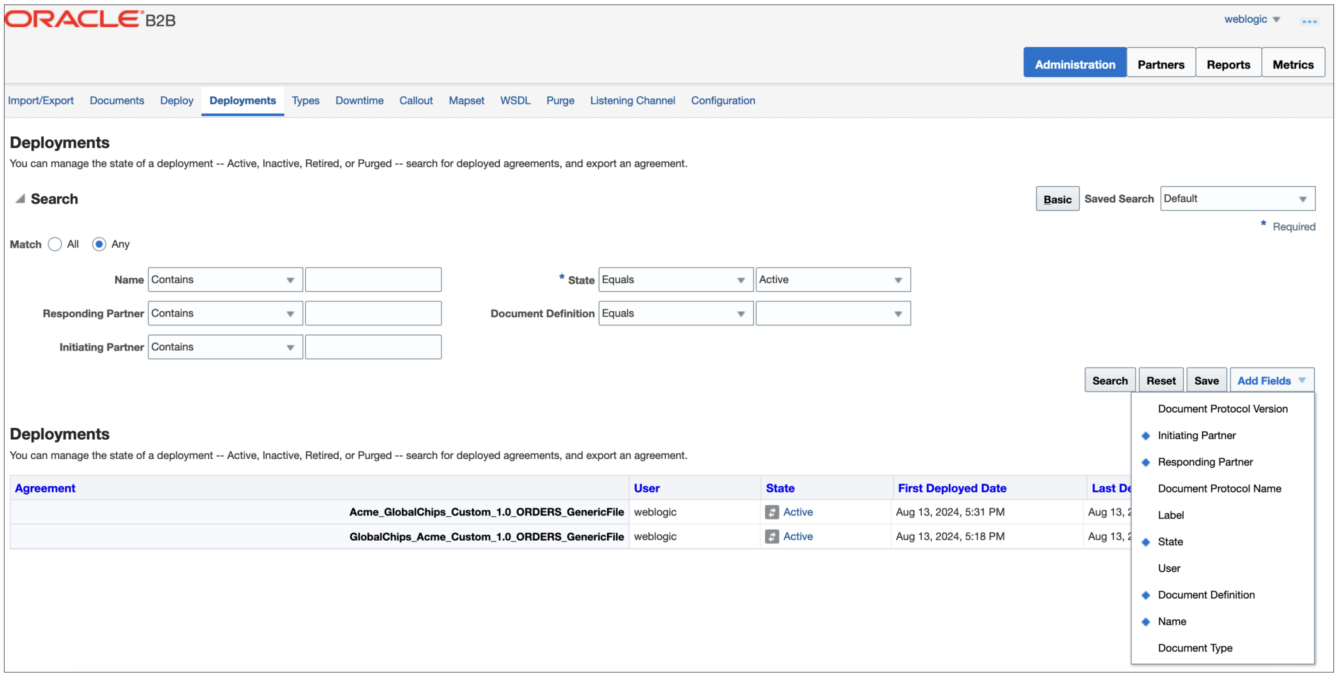
Parameter	Description
Name	Enter a string that is contained in the agreement name, equals the name, or is at the end of the name.
Responding Partner	Enter a string that is contained in the responding partner name, equals the name, or is at the end of the name.
Initiating Partner	Enter a string that is contained in the initiating partner name, equals the name, or is at the end of the name.

Table 9-3 (Cont.) Search Parameters for Searching on Deployed Agreements

Parameter	Description
*State	Select from **All**, **Active**, **Inactive**, or **Retire**.
Document Definition	Select from one of the document definitions you previously created. For more information, see Creating Document Definitions.

- 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



Deployments

You can manage the state of a deployment -- Active, Inactive, Retired, or Purged -- search for deployed agreements, and export an agreement.

Search

Basic | Saved Search: Default

* Required

Match ( ) All [✓ Any]

Name	Contains		* State	Equals	Active
Responding Partner	Contains		Document Definition	Equals	
Initiating Partner	Contains				

Search | Reset | Save | Add Fields

Deployments

You can manage the state of a deployment -- Active, Inactive, Retired, or Purged -- search for deployed agreements, and export an agreement.

Agreement	User	State	First Deployed Date	Last De
Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	weblogic	Active	Aug 13, 2024, 5:31 PM	Aug 13, 2
GlobalChips_Acme_Custom_1.0_ORDERS_GenericFile	weblogic	Active	Aug 13, 2024, 5:18 PM	Aug 13, 2

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.

Note after the user has logged in, and the host trading partner is selected by default, no agreement is shown. The user can use quick search and advanced search to search for agreements in the system.To do so, the user can clear the quick search text, then search and no agreement is shown. If the user selects remote Trading Partner, all agreements related to that Trading Partner are shown, and quick search matches the names among only those agreements. Clear the quick search text, then search and all agreements related to that Trading Partner is shown.

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.

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 includes the following sections:

- Creating Custom Identifier Types
- Creating Custom Contact Information Types
- Creating Custom Trading Partner Parameter Types

For information about *adding* custom types and values to a trading partner profile, see Creating Trading Partner Profiles.

Creating Custom Identifier Types

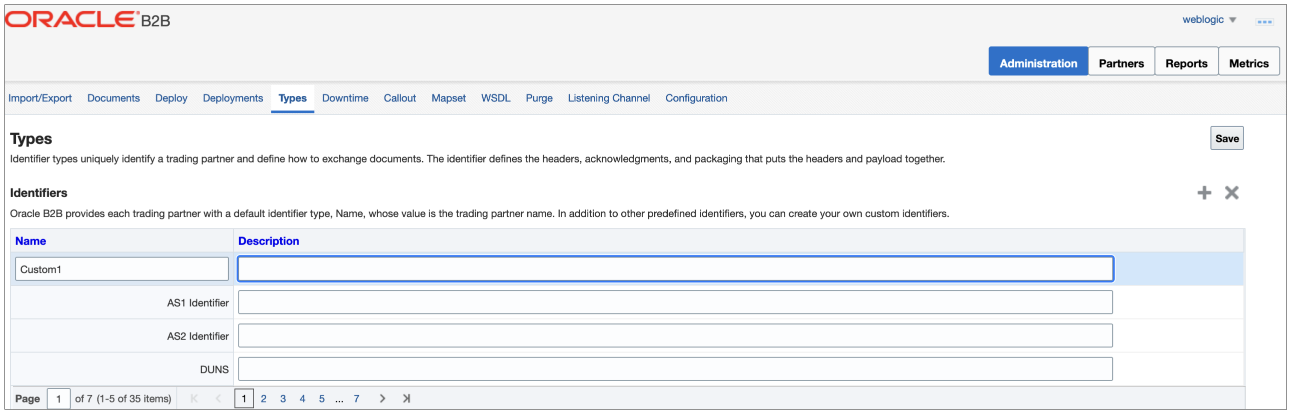
Identifier types, or identifiers, help in identifying a trading partner (as exchange identifiers) or can be used to define additional inputs for various document protocols.

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



Types

Identifier types uniquely identify a trading partner and define how to exchange documents. The identifier defines the headers, acknowledgments, and packaging that puts the headers and payload together.

Identifiers

Oracle B2B provides each trading partner with a default identifier type, Name, whose value is the trading partner name. In addition to other predefined identifiers, you can create your own custom identifiers.

Name	Description
Custom1	
AS1 Identifier	
AS2 Identifier	
DUNS	

Page 1 of 7 (1-5 of 35 items) 1 2 3 4 5 ... 7

5. Click **Save**.

See Add Identifier Types and Values 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 Identifier Types Defined in Oracle B2B

Name	Description
AS1 Identifier	The specification for using EDI over SMTP to transmit data using email. 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.
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.
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.

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, email 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 Add Contact Information for how to add the new type and a value to a trading partner's profile.

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 Add a Trading Partner Parameter and Value 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

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 includes the following sections:

- Introduction to Scheduling Trading Partner Downtime
- Scheduling Trading Partner Downtime
- Deleting Scheduled Downtime
- Extending Trading Partner Downtime

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 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 Deleting Scheduled Downtime 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 Extending Trading Partner Downtime for an explanation of the options.

Scheduling Trading Partner Downtime

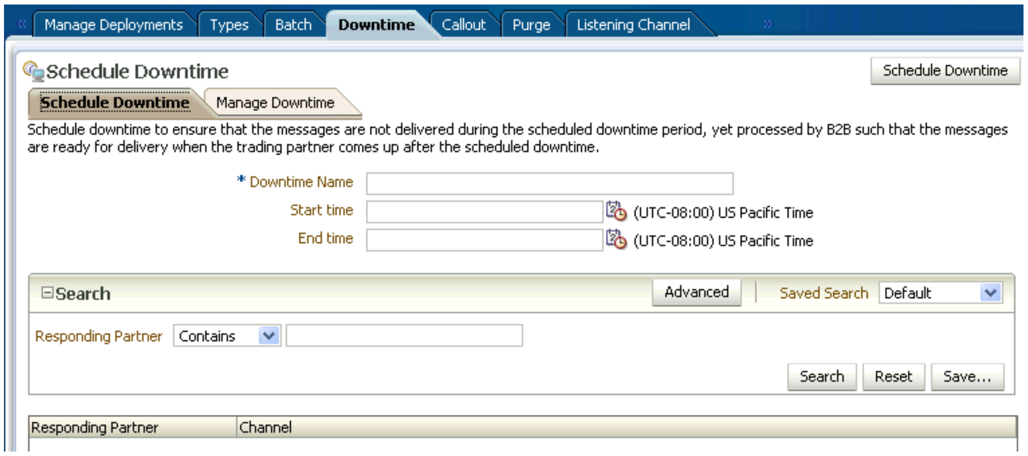
As a pre-requisite, it is required to enable the dispatcher in Oracle B2B. The number of dispatchers depends on the load and configuration of the system. The Outbound Dispatcher Count should be at least 1.

Note:

For more information, refer to Configuring B2B System Parameters .

To schedule downtime:

1. Click the **Administration** link, and select the **Downtime** tab.



Schedule downtime to ensure that the messages are not delivered during the scheduled downtime period, yet processed by B2B such that the messages are ready for delivery when the trading partner comes up after the scheduled downtime.

* Downtime Name
Start time (UTC-08:00) US Pacific Time
End time (UTC-08:00) US Pacific Time
Saved Search: Default
Responding Partner: Contains

Responding Partner	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.

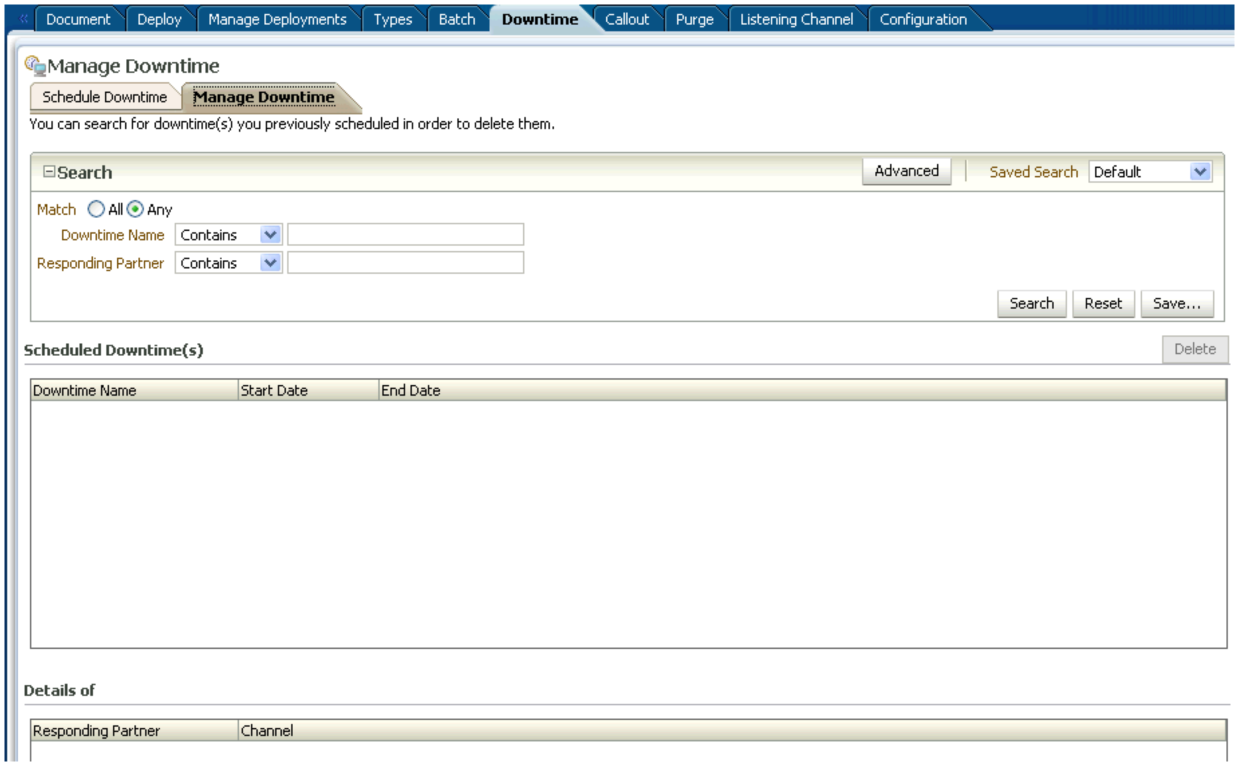
   You can configure an open-ended downtime by selecting a start time only, configured in the **Start Time** field. The **End Time** field is left empty. The open-ended downtime is stopped by deleting the schedule.
4. Select an entry in the **Responding Partner-Channel** table and click **Schedule Downtime**.

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**.



You can search for downtime(s) you previously scheduled in order to delete them.

Search

Advanced | Saved Search: Default

Match ○ All [✓ Any]

Downtime Name	Contains	
Responding Partner	Contains	

Scheduled Downtime(s)

Downtime Name	Start Date	End Date

Details of

Responding Partner	Channel

3. Select the result in the **Scheduled Downtime(s)** list and click **Delete**.

Extending Trading Partner Downtime

You can overlap schedules or use the `Extend` option to alter an existing schedule.

The following options enable you to alter an existing schedule.

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.

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 Scheduling Trading Partner Downtime for information and examples about using the command line options.

12

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.

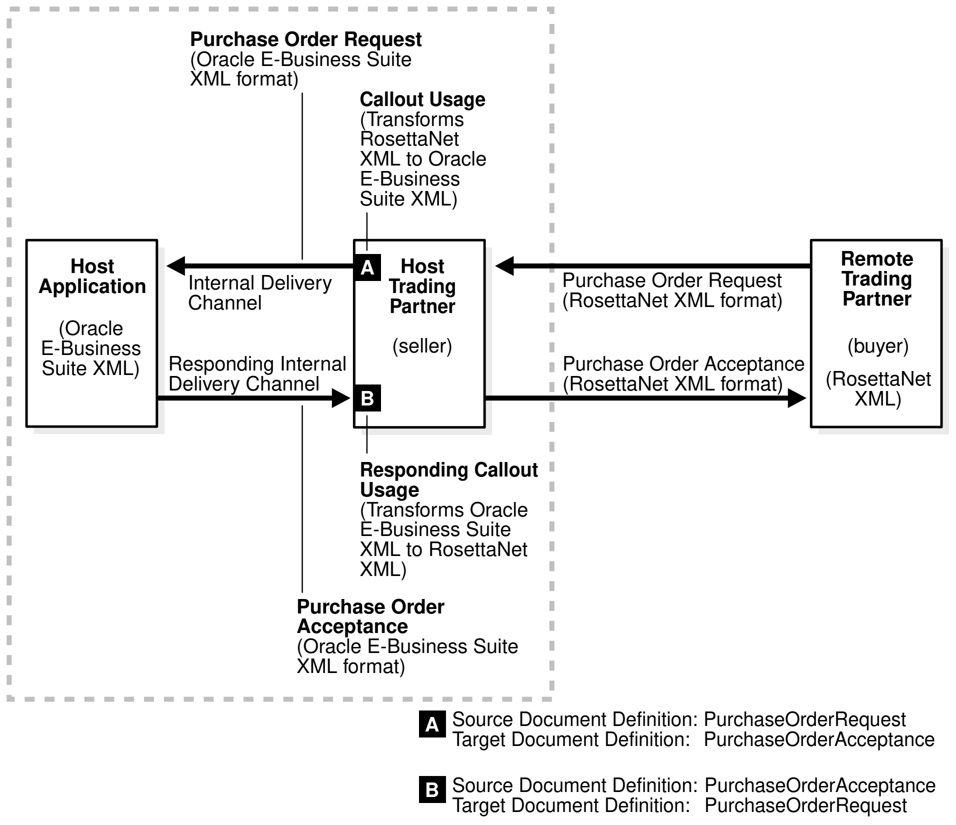
This chapter includes the following sections:

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 12-1.

Figure 12-1 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
B
Host Trading Partner (seller)
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

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.

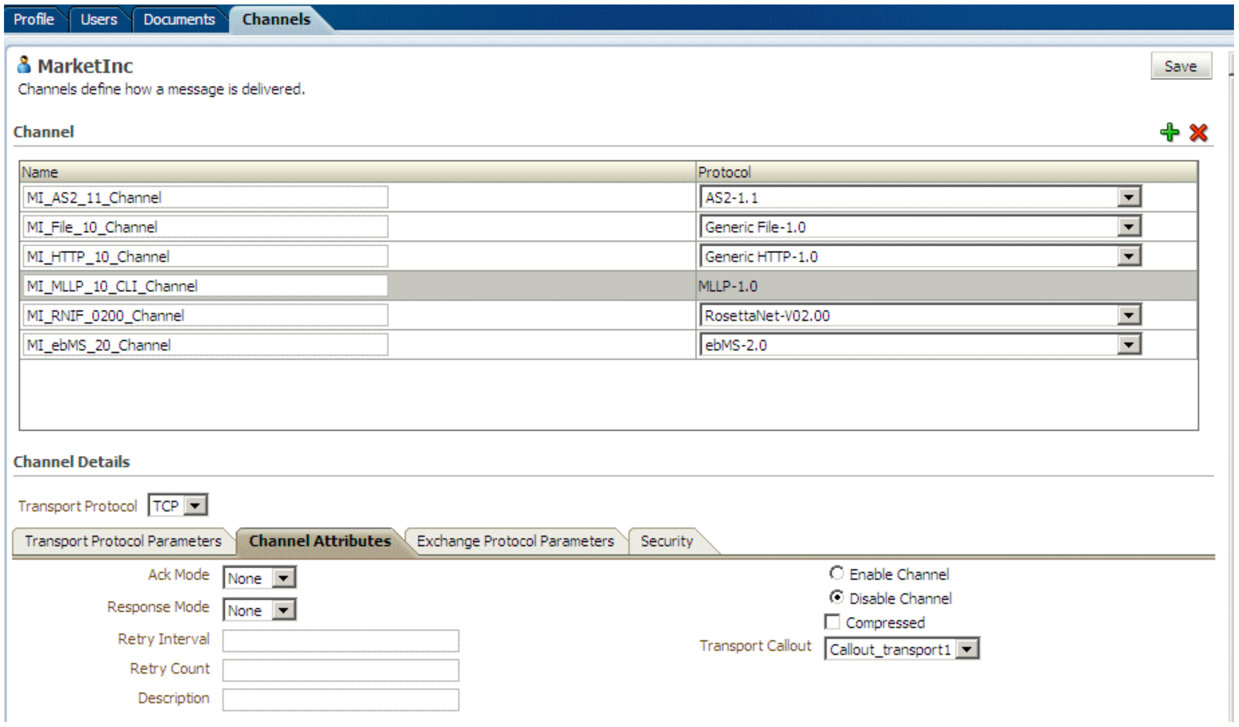
Because a document definition is a component of an agreement, a callout is associated with a specific document definition.

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 12-3 for where you add parameters and see Table 12-2 for a list of parameter attributes.

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 12-2, and can be used with any protocol.

Figure 12-2 Transport Callouts



MarketInc
Channels define how a message is delivered.

Channel

Name	Protocol
MI_AS2_11_Channel	AS2-1.1
MI_File_10_Channel	Generic File-1.0
MI_HTTP_10_Channel	Generic HTTP-1.0
MI_MLLP_10_CLI_Channel	MLLP-1.0
MI_RNIF_0200_Channel	RosettaNet-V02.00
MI_ebMS_20_Channel	ebMS-2.0

Channel Details

Transport Protocol TCP

Transport Protocol Parameters | Channel Attributes | Exchange Protocol Parameters | Security

Ack Mode None
Response Mode None
Retry Interval
Retry Count
Description

Enable Channel [✓ Disable Channel]
Compressed
Transport Callout Callout_transport1

You can use transport callouts to extract custom headers for inbound and outbound messages using the MLLP protocol. Example 12-1 shows how to set and get the `CUSTOM_HEADER` property in the callout.

For more information, see Using a Transport Callout to Extract Custom Headers.

Transport callouts are created like other callouts, from the **Callout** tab, as described in 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.

Example 12-1 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);
       }
    }
 }
```

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:

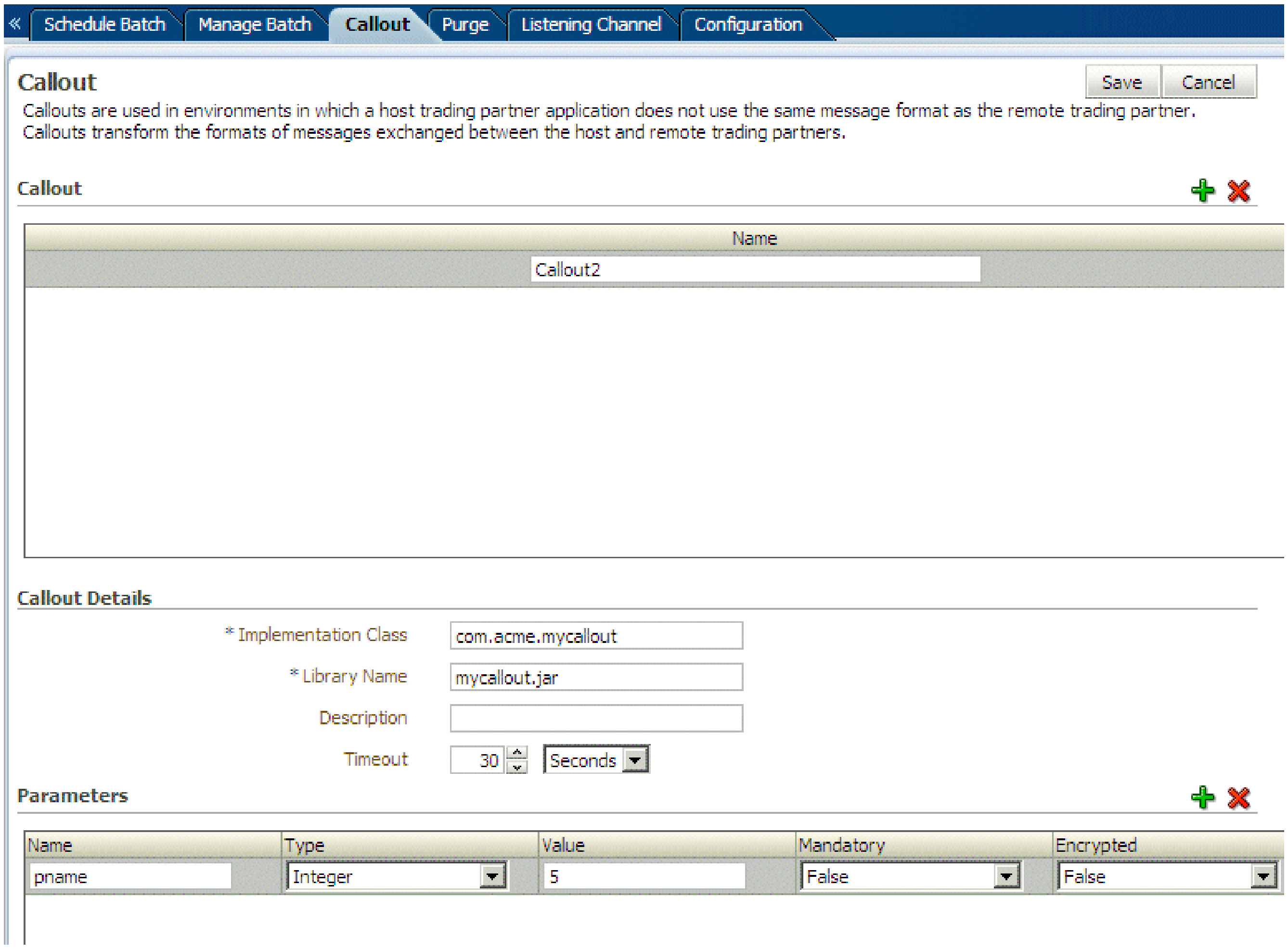
MySampleCallout is a restricted keyword and should not be used. It is already packaged into b2b.jar.

Creating a Callout

To create a callout, provide callout details—the implementation class name and library name—and callout parameters.

These are shown in Figure 12-3.

Figure 12-3 Creating a Callout



Callout

Callouts are used in environments in which a host trading partner application does not use the same message format as the remote trading partner. Callouts transform the formats of messages exchanged between the host and remote trading partners.

Callout

Name
Callout2

Callout Details

* Implementation Class	com.acme.mycallout
* Library Name	mycallout.jar
Description	
Timeout	30 Seconds

Parameters

Name	Type	Value	Mandatory	Encrypted
pname	Integer	5	False	False

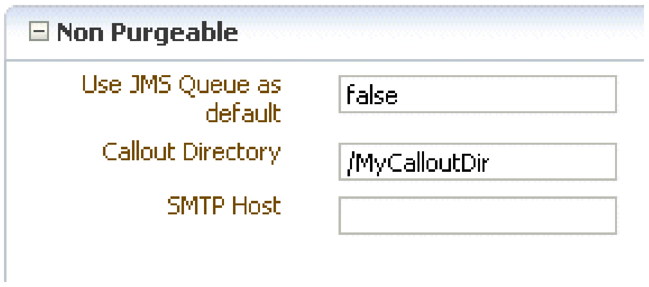
You can create multiple callouts with the same name if you assign them different implementation names. You cannot delete a callout that is included in an agreement.

Table 12-1 lists the callout details that you provide.

Table 12-1 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 12-1 (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. For information about specifying the callout directory for your own callout JAR files, see Setting Configuration Parameters.  Non Purgeable Use JMS Queue as default false Callout Directory /MyCalloutDir SMTP Host
Description	Enter a description.
Timeout (seconds)	Enter the time limit in which to process the callout.

Callout parameters are similar in cafterpt 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 12-2 lists the optional callout parameter attributes.

Table 12-2 Callout Parameter Attributes

Field	Description
Name	Enter a parameter name.
Type	Select from **Integer**, **Float**, **String**, **Boolean**, or **Date** types. The format for the **Dat**e 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. For more information, see Including a Callout in an Agreement. 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 12-1.
5. (Optional) Click **Add** in the **Parameters** section.
6. Enter a parameter name and attributes, as described in Table 12-2.
7. Click **Save**.

You can edit the details, parameters, or parameter values at any time, but not the callout name.

Securing Messages with PGP

Oracle B2B and Healthcare support message level security using PGP (Pretty Good Privacy). PGP combines features of both symmetric and public key cryptography. PGP creates a session key which is a onetime only, randomly-generated key for the text. The session key encrypts the plain text, and then the session key is then encrypted to the recipient's public key. Decryption works in reverse.
You can use this functionality as a transport callout only in B2B and Healthcare, using the PGP callout jar provided. You must have the capability to create PGP keys.

Usage Scenario

Inbound: Oracle B2B/Healthcare receives Encrypted PGP messages from an external trading partner. As part of the message processing, the encrypted PGP message will be decrypted using the enterprise's Private key. In case the PGP encrypted message contains the digital signature, the digital signature will be verified using the sender's public key. Also, in case the PGP encrypted message contains the message digest, then as part of decryption, data integrity will be verified.

Outbound: Oracle B2B/Healthcare sends encrypted PGP messages to an external trading partner. As part of the processing, the message will be encrypted using PGP APIs using the external trading partner's public key. During encryption, enterprise has the option to add the digital signature for message origin authentication and message digest for integrity check. For message signature, the Enterprise's private key will be used.

Callout Parameters

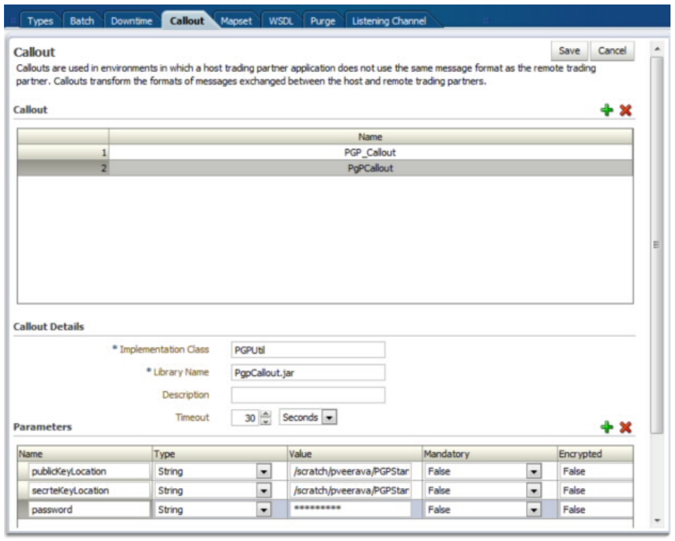
The PGP callout needs the following callout parameters while configuring the callout:

- `publicKeyLocation`
- `secretKeyLocation`
- `password`
- `direction`
- `encryptionType`

`encryptionType` is used to specify the type of algorithm used for encryption. The table below lists the supported types and the integer values that must be provided to the callout. For example, to use the AES_256 algorithm, the callout parameter value for `encryptionType` would be `9`.

EncryptionType	Integer value
`TRIPLE_DES`	`2`
`CAST5`	`3`
`BLOWFISH`	`4`
`DES`	`6`
`AES_128`	`7`
`AES_192`	`8`
`AES_256`	`9`
`TWOFISH`	`10`

Figure 12-4   PGP Callout Parameters



Callout

Callouts are used in environments in which a host trading partner application does not use the same message format as the remote trading partner. Callouts transform the formats of messages exchanged between the host and remote trading partners.

Callout

	Name
1	PGP_Callout
2	PgPCallout

Callout Details

* Implementation Class: PGPUtil
* Library Name: PgpCallout.jar
Description:
Timeout: 30 Seconds

Parameters

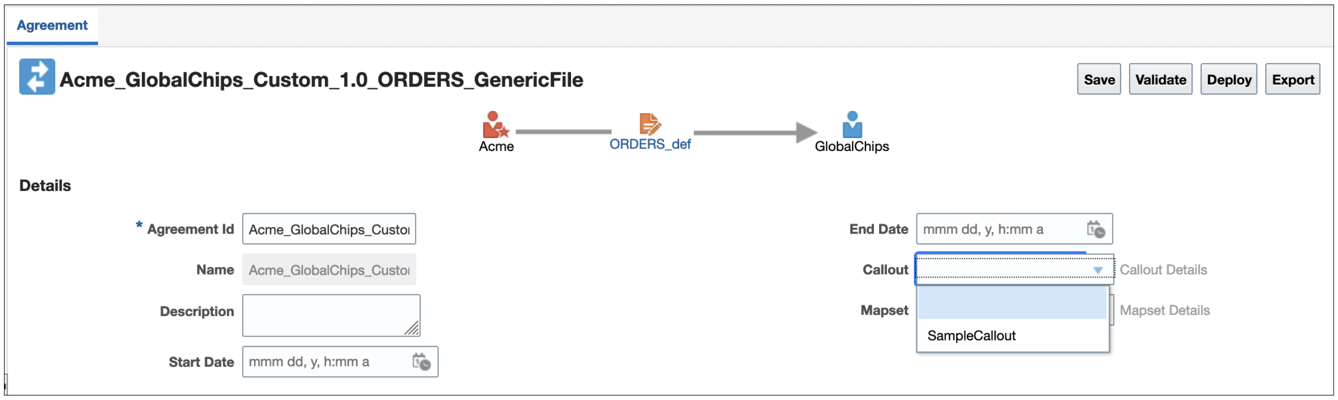
Name	Type	Value	Mandatory	Encrypted
publicKeyLocation	String	/scratch/pveerava/PGPStar	False	False
secrteKeyLocation	String	/scratch/pveerava/PGPStar	False	False
password	String	*********	False	False

Including a Callout in an Agreement

After you create a callout, it is available to include in an agreement

This is shown in Figure 12-5.

Figure 12-5 Specifying a Callout in an Agreement



* Agreement Id	Acme_GlobalChips_Custo	End Date	mmm dd, y, h:mm a
Name	Acme_GlobalChips_Custo	Callout	
Description		Mapset	
Start Date	mmm dd, y, h:mm a		

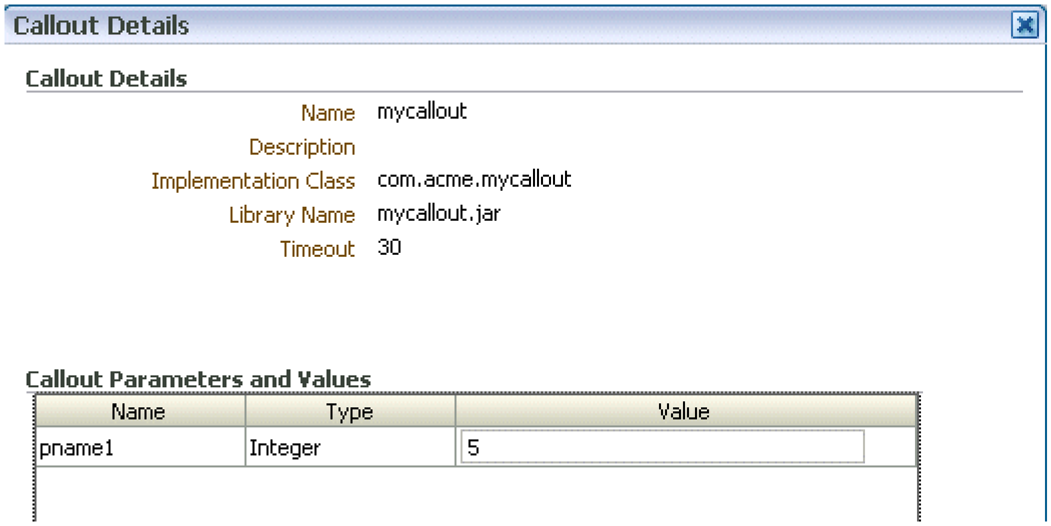
To include a callout in an agreement:

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 12-6.

Figure 12-6 Entering Callout Details



Callout Details

Name	mycallout
Description	
Implementation Class	com.acme.mycallout
Library Name	mycallout.jar
Timeout	30

Callout Parameters and Values

Name	Type	Value
pname1	Integer	5

6. Click **OK**.

Implementing a Callout

These examples illustrate the code for implementing a callout.

Example 12-2 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 12-2   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 12-3 shows how to create a synchronous callback callout for use with Transport Synch Callback. See Using Transport Sync Callback for more information.

Example 12-3   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);
                }
        }
}
```

13

Purging Data

This chapter describes how to use the Oracle B2B interface to purge design metadata and instance data.

This chapter includes the following sections:

- Purging Design Metadata and Instance Data
- Purging Data Based on Database Partitions

See the following for alternate methods of purging:

- B2B Command-Line Tools

Caution:

Archiving is required before purging data. Purging is an irreversible operation. Ensure that you first archive any important data.

Purging Design Metadata and Instance Data

Use the Oracle B2B interface to purge design metadata and instance data. Purging does not remove artifacts that B2B creates in the Credential Store, such as passwords.

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 runtime when messages are processed and contains the business messages and message-related data. Specific instance data can be purged from the **Business Message** tab of the **Reports** link. For more information, see Purging Messages.

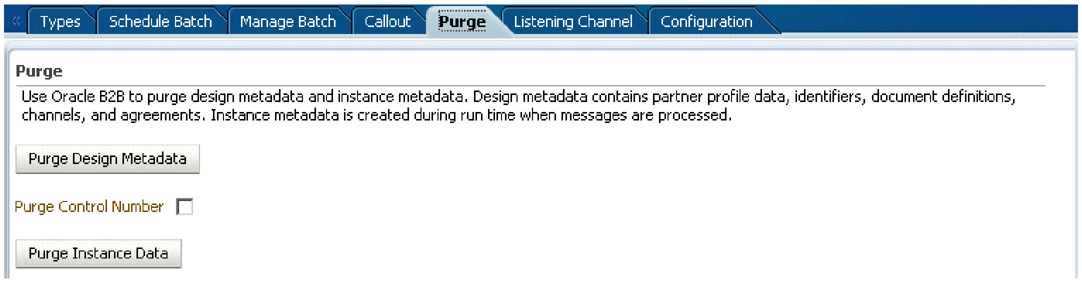
Purging is useful for:

- 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 13-1.

Figure 13-1 Purging Design Metadata or Instance Data



Purge

Use Oracle B2B to purge design metadata and instance metadata. Design metadata contains partner profile data, identifiers, document definitions, channels, and agreements. Instance metadata is created during run time when messages are processed.

4. Respond to the confirmation prompt.
5. Click **Yes**.

Purging Data Based on Database Partitions

Oracle B2B allows you to purge data from the B2B tables, based on the conditions provided by using the `PURGE_INSTANCE_MSGS_AUDIT` procedure. Typically, Oracle B2B uses the `B2B_BUSINESS_MESSAGE` table for performing database partition based purge.

The conditions that control the partition-based purge are:

- `p_startDate` - the start date for the records to be purged.
- `p_endDate` - The end Date for the records to be purged.
- `p_msgState` - The state of the Business Message.
- `p_tpName` - The Trading Partner name (if specified, `direction` needs to be specified as well.)
- `p_direction` - Indicates the direction of the method. Valid values:
  - `INBOUND`
  - `OUTBOUND`
- `p_msgType` - Indicates the request type. Valid values:
  - `REQ`
  - `RESP`
- `p_tpaName` - The Trading Partner Agreement (TPA) name.
- `p_idType` - The sender `ID_TYPE`, or identifier type, which uniquely identifies a trading partner and defines how to exchange documents.
- `p_idVal` - The sender `ID_VALUE`. The value associated with the identifier type.
- `p_shouldArchive` - Indicates whether the messages to be purged are be archived. Valid values:
  - `true`
  - `false`
- `p_force_del` - Indicates whether data will be forcefully deleted.

  <Reviewers:> Please check for technical accuracy.
- `p_archiveFileName` - The file name of the archive.

- `p_tpaId` - The TPA ID.
- `p_action` - The EBMS action to be performed.
- `p_service` - The EBMS service to be invoked.
- `p_docType` - The document type.
- `p_commitFrequency` - The commit frequency for batching up deletes; default value is `5000 rec/batch`.
- `p_auditId` - The Audit ID to track messages deleted against an `auditId`.
- `p_partitionMode` - Indicates how partitions should be handled. Valid values:
  - `DROP`
  - `TRUNCATE`
  - `STATEMENT`
- `p_logMode` - Indicates the Log level and whether the logging is console or file based. Valid values:
  - `DEBUG`
  - `INFO`
  - `ERROR`
  - `CONSOLE_DEBUG`
  - `CONSOLE_INFO`
  - `CONSOLE_ERROR`
- `p_rowLimit` - Indicates the maximum number of Business Message records that can be deleted.
- `p_stop_time` - Indicates the time by which the purge operation should be completed. The procedure continues to delete records in batch until the time is elapsed.
- `p_refreshMW` - Indicates whether the `b2b_system` materialized view should be completely refreshed after the purge. By default, the purge refreshes the view only when a partition pruning is done. Valid values:
  - `true`
  - `false`
  - `null`
- `p_numOfRecordsPurged` - Indicates the number of records that are purged in the run.

Oracle B2B provides a command-line utility called `b2bpurge` that allows partition-based purge of data. See Purging Data to know more about the utility.

14 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.

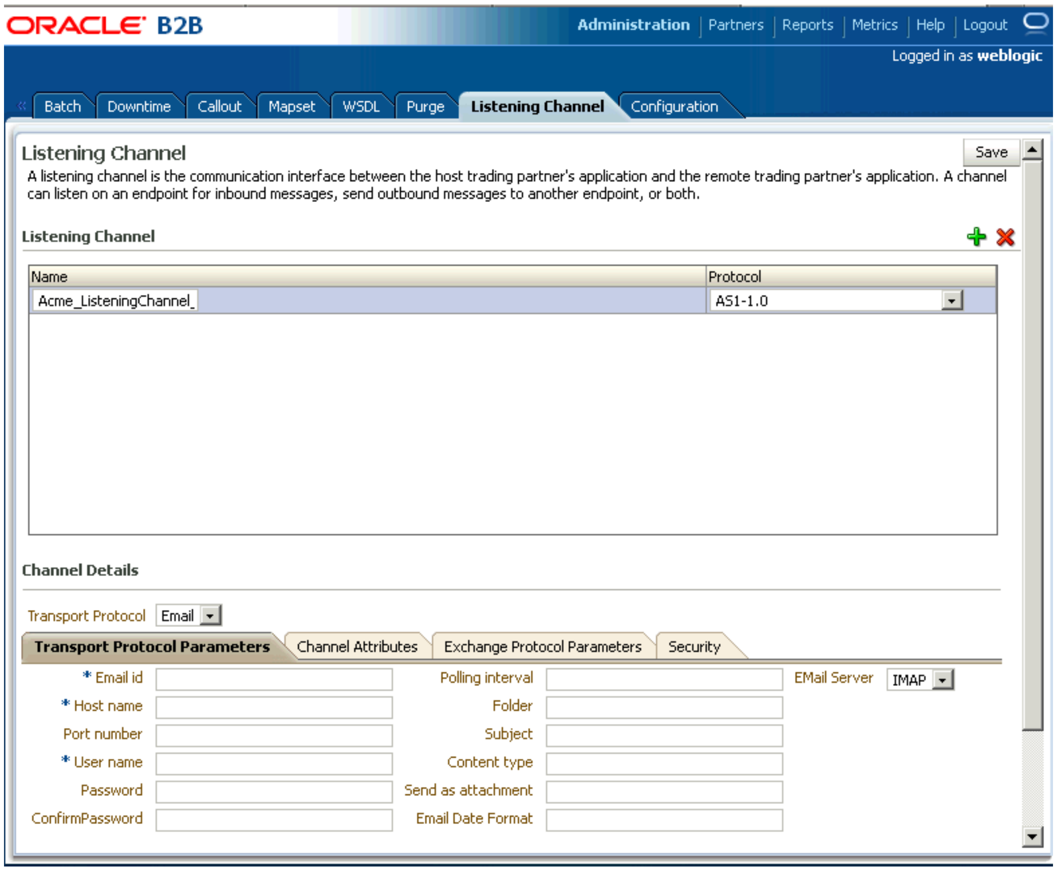
This chapter includes the following sections:

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 14-1.

Figure 14-1 Adding a Protocol for a Listening Channel



Listening Channel

A listening channel is the communication interface between the host trading partner's application and the remote trading partner's application. A channel can listen on an endpoint for inbound messages, send outbound messages to another endpoint, or both.

Listening Channel

Name	Protocol
Acme_ListeningChannel_	AS1-1.0

Channel Details

Transport Protocol: Email

* Email id		Polling interval		EMail Server	IMAP
* Host name		Folder			
Port number		Subject			
* User name		Content type			
Password		Send as attachment			
ConfirmPassword		Email Date Format			

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.

Note:

Ensure that the archive directory is not in the polling directory for File, FTP, and SFTP channels.

Table 14-1 describes the listening channel protocols supported by Oracle B2B.

Table 14-1 Listening Channel Protocols

Protocol	Description
AS1-1.0	Applicability Statement 1 (AS1) provides S/MIME and uses SMTP to transmit data using email. 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.

Table 14-1 (Cont.) Listening Channel Protocols

Protocol	Description
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).

Using Transport Protocols

The transport protocol used to send the message is determined by the listening channel you select.

This is shown in the **Channel Details** area in Figure 14-2.

Figure 14-2 Channel Details: The Transport Protocol

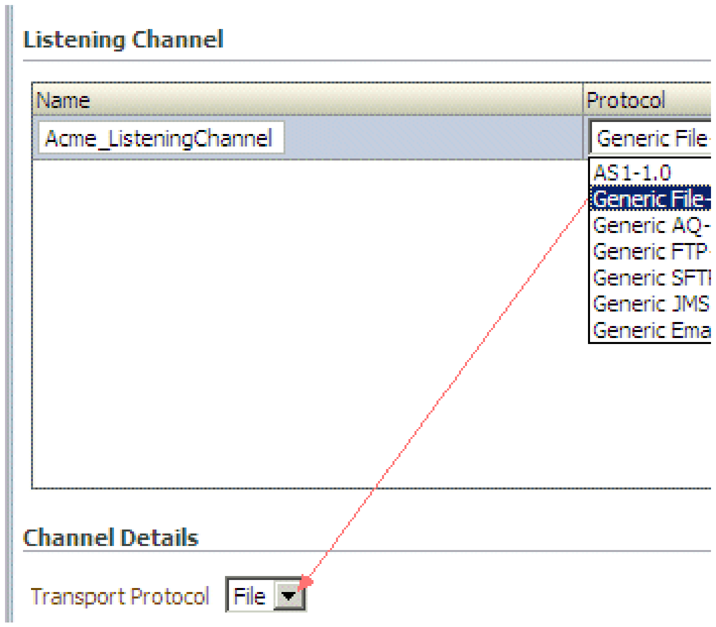


Table 14-2 describes the transport protocols available in Oracle B2B.

Table 14-2 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 14-2 (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`.

Adding Listening Channel Details

Listening channel details include transport protocol parameters, channel attributes, exchange protocol parameters, and security specifications.

The sections below describe each of these details.

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 14-3 describes the transport protocol parameters and lists the protocols to which the parameters apply.

Table 14-3 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)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Channel mask	To enable SSL for FTP, enter one of the following: • `Control`—Encrypts the control channel • 30532218 `Data`—Encrypts the data channel • `Both`—Encrypts both the data and control channels The default is **None** (no SSL). Note that the keystore should be configured in the host profile, In case of SSL there is a additional configuration required where in the user has to select the trust store.The certificate will be used from the trust store. This SSL feature is related to FTPS. The certificates are stored in the B2B Key Store. This can be JKS or KSS. No specific certificate needs to be indicated in the FTPS transport parameter as B2B uses the available certificates in the key store to negotiate the SSL hand shake.	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 email. 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 email) 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)
Control port	Provide a value to change the default FTP port value (21)	Generic FTP-1.0 (optional)
Copy Contents	Select this checkbox to allow the SFTP receiver to read the payload into a local temporary file location (within the ava tmp directory). This allows the SFTP receiver to *not* load the entire payload into memory while reading from the remote location. This can be used when the listening channel is expected to read large payloads.	
Data port	For active FTP connections, use this option to configure the static/fixed data port of the FTP server. Per the FTP protocol requirement, ensure that the port is higher than 1023.	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). You can specify comma-separated multiple e-mail addresses, so that a message can be delivered to multiple e-mail addresses.	AS1-1.0 (required) Generic Email-1.0 (required)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
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)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With

Filename format	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 similar to `Acme_850_4010.dat`. Any file extension is allowed.
	For Inbound Processing of Messages over File/FTP/SFTP Listening Channel, B2B expects the file name to contain `FROM_PARTY`.
	Where you have no control over name(s) for directory and file, you can use the `PARTNER PROFILE NAME` parameter available with FILE/FTP/SFTP Listening Channel.
	If you provide a value for this parameter in the Listening Channel, all messages arrived through such Listening Channel are assumed to be coming from Trading Partner specified in the `PARTNER PROFILE NAME` parameter.
	The `PARTNER PROFILE NAME` parameter takes precedence over any other form or forms of getting `FROM_PARTY` such as `FILE NAME FORMAT` or `DIRECTORY NAME FORMAT`.
	Example: If you specify `GlobalChips` in the `PARTNER PROFILE NAME` parameter for `FILE` Listening Channel, all messages arriving on this channel will have Sender as `GlobalChips`.
	Note:
	In File/FTP channels, if the `FILE NAME FORMAT` is set then the `DIRECTORY NAME FORMAT` is ignored.
		Generic File-1.0 (optional)
		Generic FTP-1.0 (optional)
		Generic SFTP-1.0 (optional)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
FIPS Mode	FIPS mode can be enabled for the channel by selecting this check box. Once selected the SFTP connection to remote server will be established in FIPS mode. When this option is not selected and the user is trying to connect to a remote SFTP server which is running in the FIPS mode, a connection error results. When this option is selected, and the user is trying to connect to a server running in non-FIPS mode, the connection succeeds or fails based on whether the remote server supports the FIPS algorithm.	Generic SFTP
Flow Trace EM URL	When Oracle B2B is hosted in one domain and the Oracle SOA composite (Oracle B2B adapter with the JMS option) is deployed in a different domain (Oracle Enterprise Manager Fusion Middleware Control in a different domain), Oracle B2B needs a mechanism to provide the link to the domain where the composite is deployed. Use this field to provide the domain URL details of the Oracle SOA composite for tracking the instance message flow based on the ECID. The format for setting the value of this property is: `http://<host>:<port>#<domain_name>#<domain_type>` You can also specify this URL details by setting the `b2b.flowTraceEMURL` property in Oracle Enterprise Manager Fusion Middleware Control. This URL is available as a **Flow Trace** link for individual messages in the Oracle B2B Application Message reports. While creating the Flow Trace link, Oracle B2B first checks if the value is provided at the channel level. If the value is present, it is used to construct the URL link. If the value is not present, Oracle B2B checks whether `b2b.flowTraceEMURL` is set in Oracle Enterprise Manager Fusion Middleware Control. If it is set, then the property value is used to construct the URL link. If both values are not set, by default, Oracle B2B assumes that the composite is available in the local domain and constructs the URL link accordingly.	Generic JMS-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)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With

Header File Extension	When transferring messages using The Generic File transfer protocol, Oracle B2B requires specific mandatory message headers to process the message. You need to provide all the mandatory headers by using the filename format or the directory name format. Even though Oracle B2B has no limitation regarding filename and directory name format, there may be limitations in the operating system for the file name or directory name length. For example, Microsoft Windows allows a maximum limit of 260 characters for file names or directory names.
	So, there is a necessity to read the business message headers (such as `CPAID`, `SERVICE`, `FROM/TO Party`, and so on) from a properties file for the respective payload files, because the message headers cannot be a part of file name or directory name due to operating system restriction on the length of the file or directory name.
	For the Generic File transfer protocol, for all the outbound message payload files (to be sent to partner), there needs to be a corresponding properties file in the same directory that dictates the headers to be used for the particular payload. Based on the Generic File listening channel configuration, Oracle B2B picks the payload file and reads the properties file (which is in the same location as the payload file and having the same name as that of the payload file, but with a different extension, such as `.properties`).
	For example:
	• Payload File Name: acme_purchaseorder_123.xml
	• Properties File Name: acme_purchaseorder_123.properties
	Oracle B2B listens at the configured folder and picks the payload file, and then it picks the respective properties file. It reads the properties file and sets all the necessary headers for the payload. Based on the properties provided for the payload, Oracle B2B then processes the message.
	Property files are of the format.
	`MSG_TYPE=1`
	`TO_PARTY=GlobalChips`
	`MSG_ID=123456`
	`ACTION_NAME=ACTION\:rsProcessTestRequest;SERVICE\:bcRequestTest;SERVICETYPE\:string;CPAID\:TP2Channel_Openreach_CPA;FROMROLE\:WholesaleProvider;TOROLE\:ServiceProvider;`
	`FROM_PARTY=Acme`
	Note: It is important that the properties file is present in the folder before the payload file is made available to Oracle B2B.
	Specify the extension of the properties file in this field.
	The message header properties map to the JMS properties as listed in Table G-1 in Back-End Applications Interface.
		Generic File-1.0 (optional)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
Host name	The trading partner's transport or email 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)
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 milliseconds 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)

Table 14-3 (Cont.) Transport Protocol Parameters

Protocol/Parameter	Description	Protocol Used With
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 email 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)
Subject	The subject header of the email 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)

Channel Attributes

The channel is the communication interface between the host trading partner's host application and its installation. Table 14-4 describes the channel attributes and lists the protocols to which the attributes apply.

Table 14-4 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)
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**. **Sync**: Oracle B2B provides a synchronous request/response between the fabric (back end application) to Oracle B2B and the trading partner. It is a two-way AS4 message exchange. The initiating message service handler (MSH) sends the request message and the responding MSH sends a response message in the same HTTP session. **Async** or **None**: Oracle B2B provides an asynchronous request/response between the fabric to Oracle B2B and the trading partner. It is a one-way AS4 message exchange. The initiating message service handler (MSH) sends the request message and the responding MSH sends a response message in a separate HTTP session.	This parameter is *not* available with AS1-1.0, although it appears in the B2B interface for these 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.

Table 14-4 (Cont.) Channel Attributes

Protocol/Parameter	Description	Protocol Used With
Retry Interval	The time interval in minutes 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)

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 14-5 describes the exchange protocol parameters and lists the protocols to which the parameters apply.

Table 14-5 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)

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 14-6.

Table 14-6 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.

Table 14-6 (Cont.) Security Parameters: Not Available for Listening Channel Protocols

Protocol/Parameter	Description	Protocol Used With
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.

FIPS 140 Compliance

FIPS 140-2 specifies the security requirements that must be met by a cryptographic module to protect sensitive information. The standard provides four increasing, qualitative levels of security to cover the wide range of potential applications and environments in which cryptographic modules may be employed.

Oracle Fusion Middleware Release 12c (12.2.1) supports the use of FIPS 140-2 enabled cryptographic libraries. B2B certifies FIPS 140-2 compliance for SFTP transports. When FIPS mode is enabled through a channel level attribute, this transport operates using only FIPS approved algorithms for its needs like key exchange and public key encryption.

This table provides the FIPS and non-FIPS algorithms for SFTP.

Type	Non-FIPS Algorithms	FIPS Algorithms
Key Exchange	DHG1	DHG14
Ciphers	BlowfishCBC	AES128CBC, TripleDESCBC, AES192CBC, AES256CBC
Message Authentication	HMACMD5, HMACMD596, HMACSHA196	HMACSHA1
Signature	No non-FIPS algorithm	RSA, DSA

For more information about FIPS 140–2 use in Oracle B2B, see FIPS 140 Support in Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

Enabling FIPS Mode

You enable FIPS mode as part of the transport protocol parameters of a channel, either inbound or outbound. If it is an inbound listening channel, you start from the Administration tab and then use the Listening Channel tab. If it is an outbound channel, you start from the Partners tab and then use the Channels tab. Either way, you land on the Channel Details page.

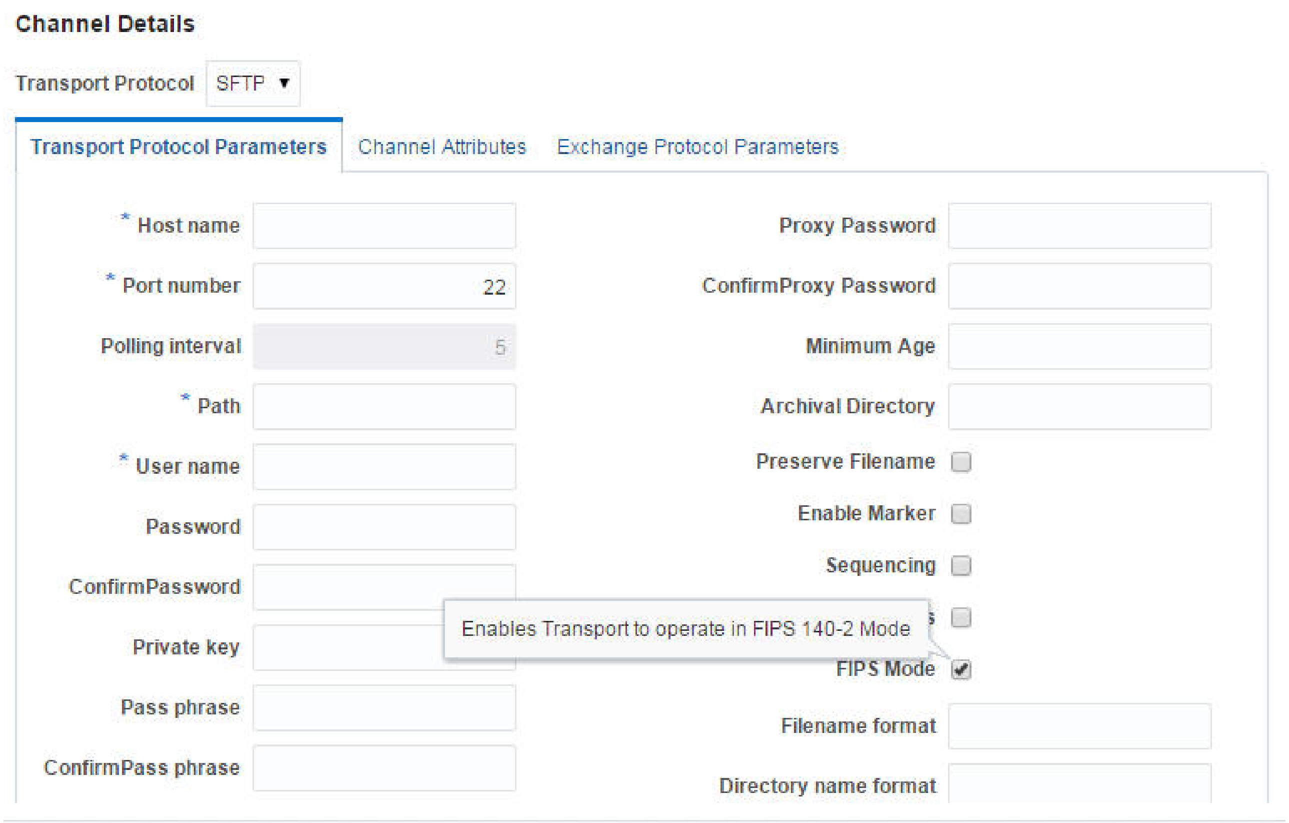
You can create a new channel and then enable FIPS mode, or you can edit an existing channel to enable FIPS mode.

> **Note:**
>
> You can only enable FIPS mode on a channel that uses the SFTP transport protocol.

To enable FIPS mode:

1. Log in the to B2B console, if necessary, and then do one of the following:
   a. For an inbound listening channel, click the **Administration** tab and then **Listening Channel**.
   b. For an outbound channel, click the **Partners** tab and then **Channels**.
2. Select the desired channel. Make sure the **Transport Protocol** is set to **SFTP**, and click the **Transport Protocol Parameters** tab, if necessary.
3. Select the **FIPS Mode checkbox** in the **Channel Details** section.

Figure 14-3 Enabling FIPS Mode


Channel Details

Transport Protocol: SFTP

* Host name		Proxy Password	
* Port number	22	ConfirmProxy Password	
Polling interval	5	Minimum Age	
* Path		Archival Directory	
* User name		Preserve Filename	☐
Password		Enable Marker	☐
ConfirmPassword		Sequencing	☐
Private key		s	☐
Pass phrase		FIPS Mode	✓
ConfirmPass phrase		Filename format	
		Directory name format	

Enables Transport to operate in FIPS 140-2 Mode

4. Click **Save**.

Configuring a Listening Channel

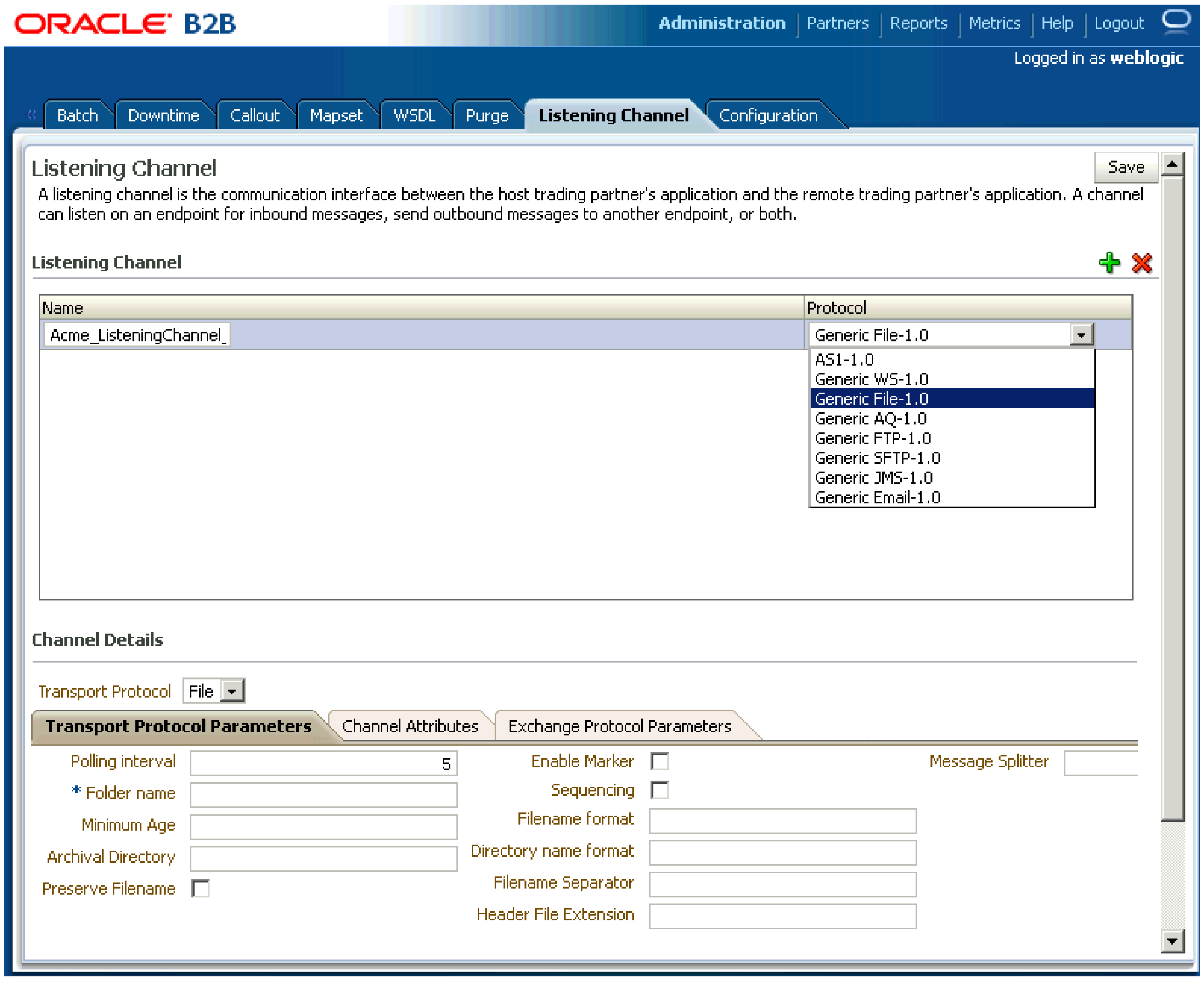
To configure a listening channel, add a listening channel protocol, and then transport protocol parameters, channel attributes, exchange protocol parameters, and security parameters, depending on the channel protocol you selected.

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 14-4 shows the list of protocols.

Figure 14-4 Adding a Listening Channel Protocol



Listening Channel

A listening channel is the communication interface between the host trading partner's application and the remote trading partner's application. A channel can listen on an endpoint for inbound messages, send outbound messages to another endpoint, or both.

Listening Channel

Name	Protocol
Acme_ListeningChannel_	Generic File-1.0

Channel Details

Transport Protocol: File

Polling interval: 5
* Folder name
Minimum Age
Archival Directory
Preserve Filename
Enable Marker
Sequencing
Filename format
Directory name format
Filename Separator
Header File Extension
Message Splitter

See Table 14-1 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 14-3 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 14-4 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 14-5 describes the exchange protocol parameters (listed in alphabetical order) and the protocols to which the attributes apply.
3. Click **Save**.

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 Message Sequencing.

Working with Default Channels

Oracle B2B provides a set of default listening and delivery channels. The external default listening channels provided by Oracle B2B are all HTTP channels.

Table 14-7 lists the default external listening channels:

Table 14-7 Default External Listening Channels

Channel Name	Description
httpReceiver	Used to receive HTTP messages. You can use the channel by accessing http://<*hostname*>:<*port*>/b2b/httpReceiver, where *hostname* is the name of the computer running Oracle SOA Suite, and *port* is the port number where Oracle B2B listens, typically `8001`. The HTTP Channel is used by the exchange protocols like AS2 and ebMS as well.
syncReceiver	Used to receive synchronous HTTP request messages. You can use the channel by accessing http://<*hostname*>:<*port*>/b2b/syncReceiver, where *hostname* is the name of the computer running Oracle SOA Suite, and *port* is the port number where Oracle B2B listens, typically `8001`
sequenceReceiver	Used to receive HTTP request messages sequentially. You can use the channel by accessing http://<*hostname*>:<*port*>/b2b/sequenceReceiver, where *hostname* is the name of the computer running Oracle SOA Suite, and *port* is the port number where Oracle B2B listens, typically `8001`.
calloutReceiver	Used to enable callouts. You can use the channel by accessing http://<*hostname*>:<*port*>/b2b/calloutReceiver, where *hostname* is the name of the computer running Oracle SOA Suite, and *port* is the port number where Oracle B2B listens, typically `8001`.

Oracle B2B also provides default internal listening channels that are listed in Table 14-8:

Table 14-8 Default Internal Listening Channels

Name	Description
In the case of Fabric	If a composite is deployed, then the channels pick up the messages from the composite.
In the case of JMS	If in the **Configuration** tab on the Administration page of the Oracle B2B console, JMS has been specified as the default, then Oracle B2B channels listen and pick up the messages from `B2B_OUT_QUEUE`.
In the case of AQ	If JMS is not specified as default, then the messages are picked up from `IP_OUT_QUEUE`.

Oracle B2B provides default internal delivery channels that:

- In the case of JMS: If in the **Configuration** tab on the Administration page of the Oracle B2B console, JMS has been specified as the default, then Oracle B2B channels listen and pick up the messages from `B2B_IN_QUEUE`.
- In the case of AQ: If JMS is not specified as default, then the messages are delivered to `IP_IN_QUEUE`.

Message Flow Throttling

Oracle B2B can pause, or throttle, the endpoint to publish messages to `B2B_EVENT_QUEUE` if the messages in `B2B_EVENT_QUEUE` grow large.

By default, `b2b.useJMSDataSourceCache` is set to true. If you want to use throttling, `b2b.useJMSDataSourceCache` cannot be set to false.

For more information about controlling the message flow, see Controlling the Flow of Messages on JMS Servers and Destinations in *Oracle Fusion Middleware Tuning Performance of Oracle WebLogic Server*guide.

To pause the endpoint:

1. In the B2B Console, go to the B2BEventQueueConnectionFactory settings page and click the **Flow Control** tab.
2. Set the **Flow Maximum**, **Flow Minimum**, **Flow Interval**, and **Flow Steps** values as appropriate for your environment.
3. Go to the B2BEventQueue settings page and click the **Thresholds and Quotas** tab.
4. Set the Messages Threshold High and Messages Threshold Low values as appropriate for your environment.

15

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 Setting B2B Configuration Properties in Fusion Middleware Control).

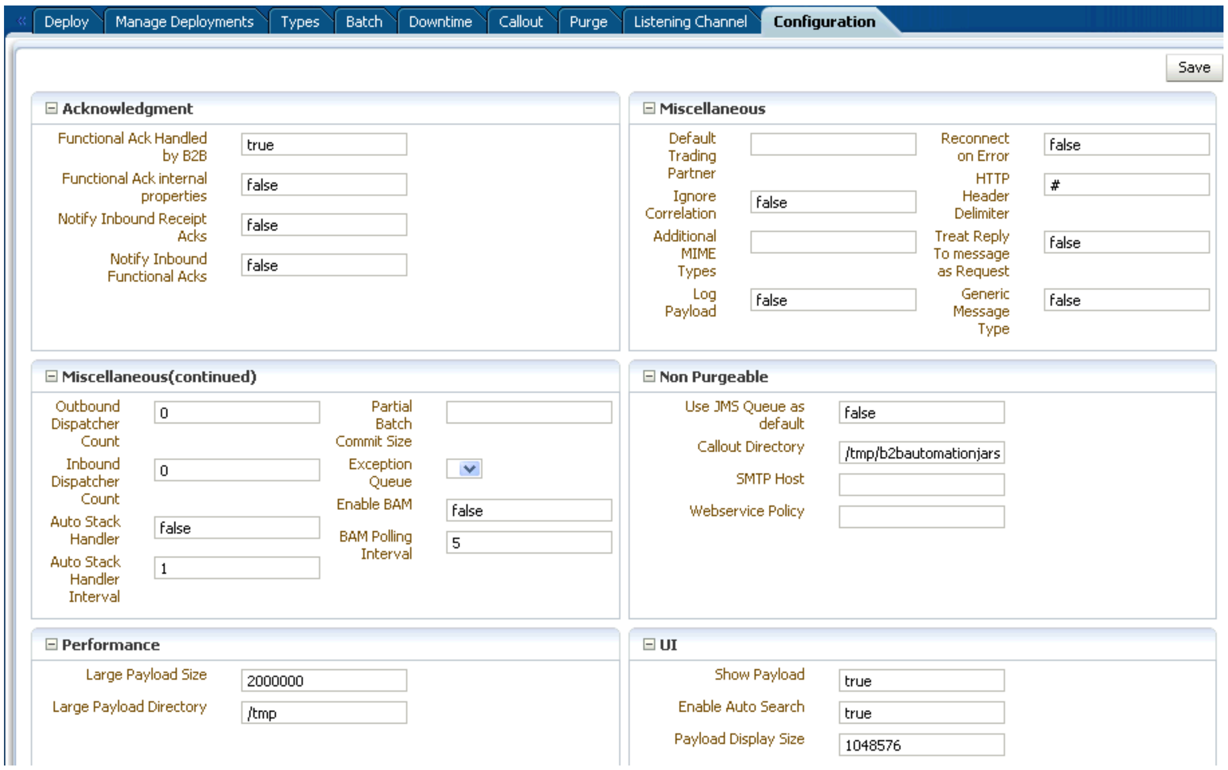
This chapter includes the following section:

- Setting Configuration Parameters

Setting Configuration Parameters

Figure 15-1 shows the configuration settings available in the Oracle B2B interface.

Figure 15-1 Configuration Parameters in the Oracle B2B Interface



Acknowledgment

Functional Ack Handled by B2B	true
Functional Ack internal properties	false
Notify Inbound Receipt Acks	false
Notify Inbound Functional Acks	false

Miscellaneous

Default Trading Partner		Reconnect on Error	false
Ignore Correlation	false	HTTP Header Delimiter	#
Additional MIME Types		Treat Reply To message as Request	false
Log Payload	false	Generic Message Type	false

Miscellaneous(continued)

Outbound Dispatcher Count	0	Partial Batch Commit Size	
Inbound Dispatcher Count	0	Exception Queue	
Auto Stack Handler	false	Enable BAM	false
Auto Stack Handler Interval	1	BAM Polling Interval	5

Non Purgeable

Use JMS Queue as default	false
Callout Directory	/tmp/b2bautomationjars
SMTP Host	
Webservice Policy	

Performance

Large Payload Size	2000000
Large Payload Directory	/tmp

UI

Show Payload	true
Enable Auto Search	true
Payload Display Size	1048576

Table 15-1 describes the configuration parameters.

Table 15-1 Configuration Settings

Field	Description
Acknowledgment	-
Notify Inbound Receipt Acks	If set to `true`, B2B sends an acknowledgment notification to the application when an exchange acknowledgment is received.
Miscellaneous	-
Default Trading Partner	Defaults to this trading partner if trading partner agreement identification fails.
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`.
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. For more information, see Monitoring Instance Message Data With Oracle BAM.

Table 15-1 (Cont.) Configuration Settings

Field	Description
BAM Polling Interval	Polling interval in minutes for Oracle BAM. For more information, see Monitoring Instance Message Data With Oracle BAM.
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 Securing Oracle B2B Web Services for more information about the use of this field.
SSL Private Key Password	When configuring in B2B SSL(with client authentication required) or signing, if private key password (`keypassword`) is different from keystore password, when doing handshake (SSL) or trying to encrypt message this error occurs: "Could not establish SSL due to `java.security.UnrecoverableKeyException: Cannot recover key`. So, B2B is not able to access its private key from a keystore if keypass is not the same as keystore password. Therefore, if the private key password is different from the keystore password, the private key password can be configured using **Administration->Configuration-> "SSL Private Key Password**
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.

Table 15-1 (Cont.) Configuration Settings

Field	Description
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.
Session Timeout	Enables you to set when the session will timeout due to inactivity. It is measured in seconds. The default is 300 seconds. Any changes will take effect after you log out and log back in.

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 15-1.
4. Click **Save**.

Part IV
Reports and Metrics

This part discusses securing B2B, creating reports, and metrics.

This part contains the following chapters:

16 Securing Oracle B2B

This chapter discusses the security features of Oracle B2B such as the OPSS, roles, users, keystores, and security policies.

This chapter contains the following sections:

Introduction to OPSS

Oracle B2B uses the Oracle Platform Security Services (OPSS) framework to implement security. Oracle Platform Security Services (OPSS) is a security platform that can be used to secure applications deployed on any of the supported platforms or in standalone applications.OPSS is a standards-based, portable, integrated, enterprise-grade security framework for Java SE and Java EE applications.

OPSS is the underlying security platform that provides security to Oracle Fusion Middleware including WebLogic Server, Server Oriented Architecture (SOA) applications, Oracle WebCenter, Oracle Application Development Framework (ADF) applications, and Oracle Entitlement Server. OPSS is portable to third-party application servers, so you can use OPSS as the single security framework for both Oracle and third-party environments. This helps in decreasing application development, administration, and maintenance overheads.

OPSS provides an abstraction layer in the form of application programming interfaces (APIs) that insulate developers from security and identity management implementation details. With OPSS, developers do not need to know the details of, for example, cryptographic key management, repository interfaces, or other identity management infrastructures. Using OPSS, in-house developed applications, third-party applications, and integrated applications benefit from the same, uniform security, identity management, and audit services across the enterprise.

OPSS conforms to the following standards: role-based-access-control (RBAC), Java Enterprise Edition (Java EE), and Java Authorization and Authentication Services (JAAS) and provides a security platform that supports:

- Authentication
- Identity assertion
- Authorization, based on fine-grained JAAS permissions
- The specification and management of application policies
- Secure storage and access of system credentials through the Credential Store Framework
- Secure storage and access of keys and certificates through the Keystore Service
- Auditing
- Role administration and role mappings
- The User and Role API
- Identity Virtualization

- Security configuration and management
- SAML and XACML
- Oracle Security Developer Tools, including cryptography tools
- Policy Management API
- Java Authorization for Containers (JAAC)

Securing Oracle B2B

Using the OPSS framework, Oracle B2B provides security options in terms of users and groups, credentials, identity authentication and authorization, policies, keystores, and auditing.

See the following sections for more information.

- Managing Users and Groups
- Managing Credentials
- Managing Identity, Authentication, and Authorization
- Managing Policies
- Managing Keystore
- Managing Auditing

Managing Users and Groups

Using the Oracle Fusion Middleware Control console, you can create, edit, view, and modify users and groups for Oracle B2B. You can access the Fusion Middleware Control console from the following URL:

http://<*hostname*>:<*port*>/console, where,

hostname is the name of the computer running Oracle Weblogic server and the port number is typically `7001`.

To access Fusion Middleware Control and perform tasks, you must have the appropriate role. Fusion Middleware Control uses the Oracle WebLogic Server security realm and the roles defined in that realm. If a user is not granted one of these roles, the user cannot access Fusion Middleware Control.

Each role defines the type of access a user has. For example, a user with the role Admin has full privileges. A user with the role Monitor has privileges only to view the configuration.

For more information about creating users and groups, see Users, Groups, and Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Managing Credentials

A credential can hold user names, passwords, and tickets; credentials can be encrypted. Credentials are used during authentication, when principals are populated in subjects, and, further, during authorization, when determining what actions the subject can perform. A principal is the identity to which the authorization in the policy is granted. A principal can be a user, an external role, or an application role. Most frequently, it is an application role.

Oracle B2B uses the OPSS Credential Store Framework (CSF), a set of APIs that applications can use to create, read, update, and manage credentials securely. A typical use of the credential store is to store user names and passwords to access some external system, such as a database or an LDAP-based repository.

Managing Identity, Authentication, and Authorization

Oracle B2B uses the Identity Governance Framework (IGF) of OPSS to implement authentication and authorization of all users and roles. It replaces the old User/Role API model.

All the authorization related entries and user role association is stored in the OPSS policy store. The OPSS policy store is maintained per domain basis and each application running under a domain has its own scope in the policy store. This scope is called Application Stripe, and all the policy data for an application are managed under the application stripe.

Oracle B2B uses IGF for the following:

- Authentication: Authenticating the user against the LDAP user store.
- Retrieving user details: Retrieving the user profile/attributes such as user name, department, phone, and so on.
- Search operations: Searching user role.
- Role operations: Creating, deleting, and managing roles.

Oracle B2B makes use of a resource-based model for managing authentication and authorization.The resource-based model allows Oracle B2B to:

- Provide administrators with human readable and manageable resource definition and policies.
- Enable rich querying of the policy store for policies, grantable resources, and policies-granted resources
- Provide a single unified authorization administration console across Oracle products that externalize authorization using the OPSS Authorization model.
- Leverage advanced authorization policy enforcement and management capabilities in the Oracle Identity Management suite.

All the resources that are to be protected should be defined along with the resource type in the Resource Catalog before granting the permission to resource. Each resource is unique and represented by a name. The resource type is a group representing all the resources of same type. Resource types are defined with matching class and list of actions. However, this approach neither changes the way authorization is done, nor it has any effect on the authorization. This approach is recommended for administration and audit purpose.

For more information about Resource Catalog and the OPSS authorization models, see The JAAS/OPSS Authorization Model in *Securing Applications with Oracle Platform Security Services*.

Oracle B2B also uses the OPSS server authentication providers, which are components that validate user credentials or system processes based on a user name-password combination or a digital certificate. Authentication providers also make user identity information available to other components in a domain (through subjects) when needed.

Managing Policies

Fusion Middleware Control allows managing system and application policies in a WebLogic domain running Oracle B2B, regardless of the type of policy store provider used in the domain.

A Java 2 policy specifies the permissions granted to signed code loaded from a given location.

A JAAS policy extends Java 2 grants by allowing an optional list of principals; the semantics of the permissions are granted to only code from a given location, possibly signed, and run by a user represented by those principals.

An application policy is a collection of Java 2 and JAAS policies, which is applicable to just that application (in contrast to a Java 2 policy, which is applicable to the whole JVM).

The policy store refers to the portion of the security store where application and system policies are kept. The type of the policy store can be file, LDAP, or DB. A file store is an XML file. The only LDAP policy store type supported is Oracle Internet Directory. Application roles can include enterprise users and groups specific to the application (such as administrative roles). A policy can use any of these groups or users as principals.

Using Fusion Middleware Control, you can manage Application Policies, Application Roles, and System Policies pertaining to Oracle B2B. For more information, see Managing Policies with Fusion Middleware Control in *Securing Applications with Oracle Platform Security Services*.

Note:

When configuring Application Policies, and Application Roles for Oracle B2B, select **b2bui** as the Application Stripe.

Managing Keystore

Oracle B2B supports Farm Keystore Service (FKS) primarily along with Oracle Key Store Service (KSS) for keystore-related operations.

Earlier, there was no centralized keystore for Oracle B2B and other J2EE applications. Each application created and managed its own keystores and it relied on using system properties to locate relevant keystores. There was also no standard or guideline for keystore storage, thus leading to several keystore files scattered all over the file system. It was difficult for the security administrator to ensure common policies, such as key strength, algorithm, and so on across applications. There was also the problem of trust management. Multiple truststores meant that even applications within the same domain do not necessarily trust each other.

KSS provides a single storage of keystores that all participating entities (Java EE applications, WLS, OVD, system components) can use across the farm. Keystores are stored in a single repository, which facilitates backup, export/import, and migration. Access to this keystore repository is provided through standard JDK KeyStore interfaces so that the applications do not need to change much when switching between keystore types. The framework also supports a common truststore to provide easy trust management. There is a root-level Certificate Authority that signs certificates in a staging environment.

Following are the key features of KSS:

- Provides a common trust store for the whole domain for centralized management of trust
- Supports generation and storage of secure keys in hardware devices
- Provides secure access to keys and certificates using code-based permissions
- Supports key password and keystore password if you want additional password-based security, which is in addition to code-based security
- Manages keystores within separate stripes – system and applications

- Provides management support through Fusion Middleware Control and Weblogic Scripting Tool (WLST)
- Supports LDAP and database-based providers for keystores

Oracle B2B uses KSS for providing transport protocol-based security for HTTP, FTP, AS2, and SMTP exchanges such as:

- Digital envelopes and certificates
- Digital signatures for host and remote trading partners
- Integration with KSS for storing all passwords and security credentials
- Secure HTTP (using Secure Socket Layer (SSL))
- Encrypted Key Store password for a host trading partner

In the KSS framework, there is a single truststore file available at the system level (trust.jks). This file is used to store trusted certificates for all applications (such as Oracle B2B) by default, unless explicitly specified as something else. For example, an application A can either specify a file stored under its context to be used as truststore (say app2.jks), or choose to use the common domain level truststore (trust.jks).

Application stripe and key stores are managed either by using WLST commands or from a Fusion Middleware Control console. To access the keystores in any application stripe, you must have the `KeyStoreAccess` permission on either the entire application stripe or a particular keystore. You can be define this in the authorization policy either through WLST or Fusion Middleware Control (System-jazn-data.xml).

Note:

The stripe for B2B should be B2B. By default, this stripe is not available in KSS. You need to first create the stripe and then create corresponding keystores under this stripe.

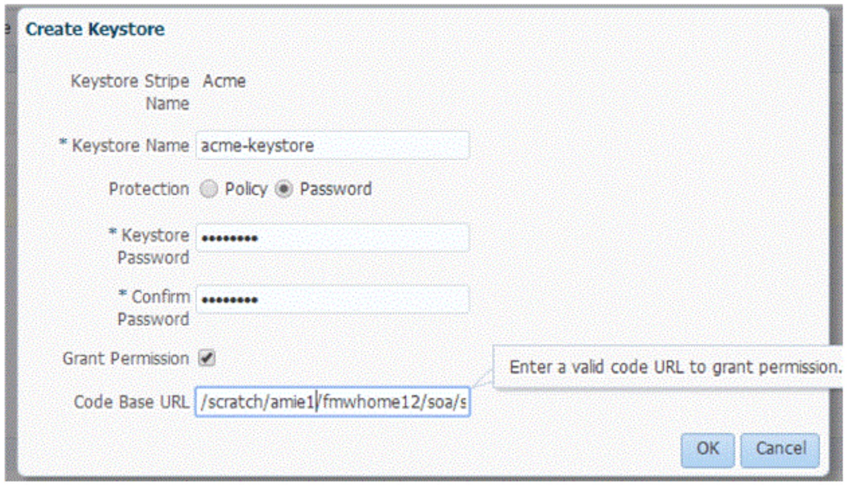
You should use this stripe, which you create in the format: kss://<keystore-stripe>/<keystore-name>

You should use this as all keystores under B2B stripe are provided with necessary permission to use. For B2B, the uri typically resembles: `kss://B2B/Acme`

Using Another Stripe

If you want to use any other stripe, then while creating the key store you must perform additional steps to provide code source permission as shown in Figure 16-1 and described below.

Figure 16-1 Using Another Stripe


Create Keystore

Keystore Stripe Name	Acme
* Keystore Name	acme-keystore
Protection	Policy [✓ Password]
* Keystore Password	••••••••
* Confirm Password	••••••••
Grant Permission	✓
Code Base URL	/scratch/amie1/fmwhome12/soa/s

Enter a valid code URL to grant permission.

Note that you must specify:

- Grant Permission = checkbox selected
- The Code Base URL = `$MW_HOME/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar` (that is, the absolute or relative path of the `b2b.jar`)

If the keystore location already exists, then you must configure system policies to provide necessary privileges within system policies.

For more information, see Managing Keys and Certificates and Developing with the Keystore Service in *Securing Applications with Oracle Platform Security Services*.

Features of FKS

FKS provides the following services:

- Support for Provider Types
- Support for Keystore Type
- Support for Governance policy
- Support for certificate expiration

Provider Types

KSS supports provider-based architecture like other OPSS services. By default, the configuration for these services is available on a file-based provider. KSS also supports LDAP-based providers as supported by other OPSS services. You need to set these configuration parameters in jps-config.xml located under *$DOMAIN_HOME*/config/fmwconfig.

Keystore Type

A parameter indicating the type of keystore being used in KSS needs to be set in the jps-config.xml file. This is required by the runtime provider.

```
<property name=" kss.type" value=" KSS" />
```

Governance Policy

Governance policy determines what types of certificates, keys, and so on are allowed in KSS. For example, an administrator can define rules that allow only keys that are 1024 bits and higher, disallow MD5 hash algorithm, allow only RSA signing algorithm, or have key validity of at least 2 years from current time.

All these parameters are configurable in jps-config.xml as the following:

```
<property name=" kss.min.keysize" value=" 1024" />
<property name=" kss.disallowed.algorithms.hashing" value=" MD5" />
<property name=" kss.disallowed.algorithms.signing" value=" DSA" />
<property name=" kss.allowed.keyusage" value=" digsig,crlsign" />
```

Certificate Expiration

KSS provides a mechanism to configure policies to manage certificate expiration alert. It is integrated with User Messaging Service (UMS) to provide notifications and alerts.

Policy: KSS allows administrators to configure that an alert needs to be sent a specific number of days before a certificate expires. You can configure it in jps-config.xml as:

```
<property name=" kss.daysbefore.cert.expiration" value=" 60" />
```

Alerts: KSS allows administrators to configure channels to manage alerts and notifications for certificate expiration. By default, a channel is configured to log warning message to a log file. An administrator can additionally configure alerts through e-mail, SMS, and so on. Oracle B2B supports the alert mechanisms supported by UMS.You can configure multiple channels for KSS, and it is possible to specify more than one to be used for sending alerts. You can configure it in jps-config.xml as:

```
<property name=" kss.channel.log" value=" /tmp/alert.log" />
<property name=" kss.channel.email" value=" admin@foo.com,smtp.foo.com" />
<property name=" kss.channel.cell" value=" 16505067000,mobilegateway.foo.com" />
<property name=" kss.alert.channel" value=" kss.channel.log, kss. channel.email" />
```

Using KSS, the following keystore service operations are possible:

- Create keystore
- Delete keystore
- Update keystore
- Change keystore password
- Change key password
- Export keystore
- Import keystore
- Migrate Keystore in
- Migrate keystore out
- Validate Certificate
- Merge Certificate

Naming Convention for Keystores

Adhere to the following naming conventions for KSS keystores:

- Do not use a name longer than 256 characters.
- Do not use any of the following characters in a keystore name:

  ```
  | ; , ! @ # $ ( ) < > / \ " ' ` ~ { } [ ] = + & ^ space tab
  ```

- Do not use non-ASCII characters in a keystore name.
- Additionally, follow the operating-system-specific rules for directory and file names.

Setting Up Keystore for Oracle B2B

You can set up KSS keystores for Oracle B2B either by using the Fusion Middleware Control console or by using WLST commands.

Setting Up Keystore by Using Fusion Middleware Control Console

For more information about setting up keystores, see Managing Keystores with Fusion Middleware Control in *Securing Applications with Oracle Platform Security Services*.

Setting Up Keystore by Using WLST

Perform the following steps to set up the keystore for Oracle B2B using WLST commands:

1. Start Oracle Weblogic servers (Admin server and managed servers).
2. On a console window, run the following commands:

   a. `cd $FMW_HOME/oracle_common/common/bin`

   b. `./wlst.sh`

      to set the WLST environment.

   c. `connect("<Admin user>","<Admin user password>","t3://localhost:<Admin server port>")`

      to connect to the online server.

   d. `svc = getOpssService(name='KeyStoreService')`

      to initialize and retrieve all the keystore services.

   e. `svc.createKeyStore(appStripe='<StripeName>', name='<Keystorename>', password='<keystorePassword_Optional>', permission=true/false)`

      to create the keystore. For example:

      ```
      svc.createKeyStore(appStripe='B2B', name='B2BKeyStore', password='weblogic1',
      permission=false)
      ```

   f. `svc.generateKeyPair(appStripe='<StripeName>', name='<Keystorename>', password='<keystorePassword_Optional>', dn='cn=www.abc.com', keysize='1024', alias='<keyAlias>', keypassword='<KeyPassword>')`

      to generate a private/public key pair with key alias *<keyAlias>* under the *<Keystorename>* store of the *<StripeName>* stripe.

      For example:

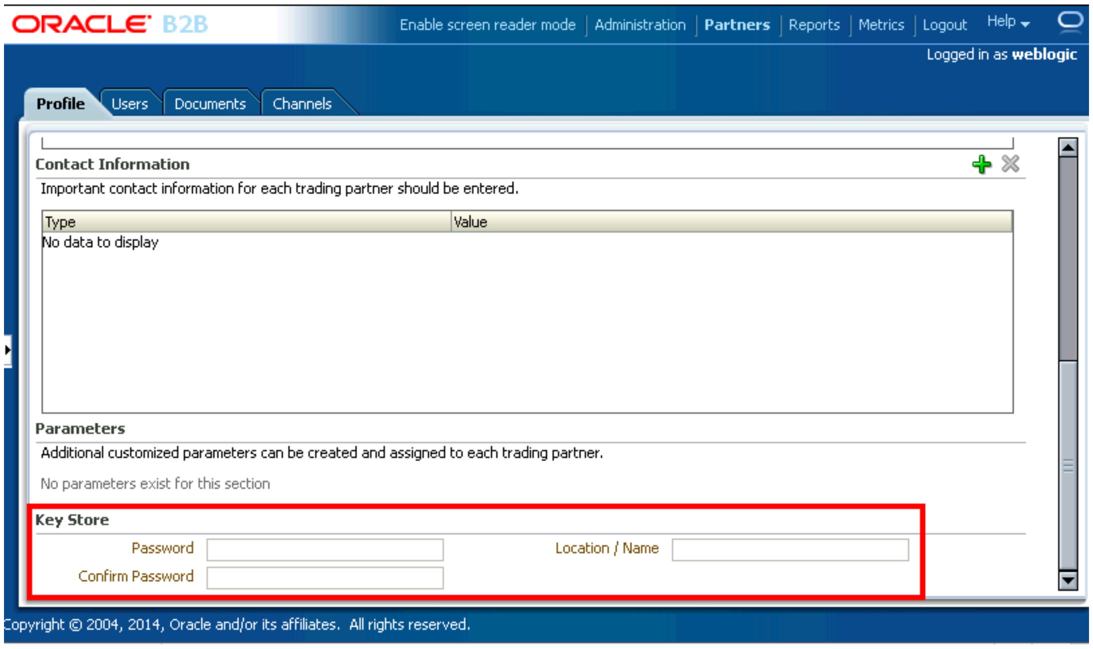
      ```
      svc.generateKeyPair(appStripe='B2B', name='B2BKeyStore',
      password='<keystorePassword_Optional>', dn='cn=www.abc.com',
      keysize='1024', alias='Acme', keypassword='<KeyPassword>')
      ```

Note:

Permission for code source for `b2b.jar` is applicable only to the keystore under stripe B2B. By default, Oracle B2B has provisioned with code source permission for all stripes under Oracle B2B.

In the Oracle B2B console under the host trading partner **Profile** tab, you can specify the keystore credentials, such as **Password** and **Location/Name** as shown in Figure 16-2:

Figure 16-2 Providing Keystore Credentials



Contact Information

Important contact information for each trading partner should be entered.

Type	Value
No data to display	

Parameters

Additional customized parameters can be created and assigned to each trading partner.

No parameters exist for this section

Key Store

Password

Confirm Password

Location / Name

Copyright © 2004, 2014, Oracle and/or its affiliates. All rights reserved.

When the trustStore is specified in the JRE, B2B uses that trustStore for establishing TLS connections. You therefore have the option to use the default `.jks` file that is installed with WLS or the `.jks` file specified by the —`Djavax.net.ssl.trustStore` parameter in the WLS startup script. If the `b2b.httpsTrustStore` server property is set to `true`, or it is running in a Fusion Application environment, then B2B uses the default `.jks` file to send for the https delivery channel.

Note:

If the keystore credentials are not provided, then the default KSS weblogic keystore is used.

Managing Auditing

Oracle B2B uses the OPSS audit framework to perform auditing.

For more information, see Managing Audit in *Securing Applications with Oracle Platform Security Services*.

17

Creating Reports

This chapter describes Oracle B2B reports that provide real-time status on the runtime behavior of deployed data. It discusses business message report, wire message report, application message report, error report, and conversation report.

The chapter includes the following sections:

Introduction to Reports

Use the **Reports** link to search on data in the runtime repository. The Saved Search function is not available.

The following message types are available for searching:

- Business messages—See Creating Business Message Reports
- Wire messages—See Creating Wire Message Reports
- Application messages—See Creating Application Message Reports
- Error messages—See Creating Error Reports
- Conversation messages—See 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.

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 Adding Trading Partner Users, for how to assign the Monitor role.

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.

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.

When messaging errors are because of endpoint Hostname/IPAddress or port change, then the messages are in MSG_WAIT_STACK or MSG_WAIT_TRANSMIT. Correct the Hostname/IPAddress or port of the endpoint and resubmit the first errored message which is in MSG_WAIT_STACK. All the pending messages in the sequence then use the updated Hostname/IPAddress or port of the endpoint and are processed. There is no need to resubmit every error message for the endpoint.

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:

For certain documents, the user can set XPath expressions to check for, and thus avoid duplicates. If two messages arrive with the same XPath values, the latter of the messages is marked as duplicate and it errors out.

When you resubmit this errored duplicate message (a wire message resubmit), Oracle B2B processes the message ignoring the fact that it is a duplicate, because the resubmission is done intentionally. So, if you do not want Oracle B2B to process duplicate messages, you should not resubmit those messages.

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.

Correlation Flow Id Response in B2B During Resubmission

Wire message-based inbound resubmission results in a new correlation flow id. It is treated as new business flow.

Inbound Application resubmission re-uses the correlation information because the B2B business information is not altered.

If you want to retain the correlation information, you can use Application message resubmission; if you are initiating a new flow completely, you can use wire resubmit.

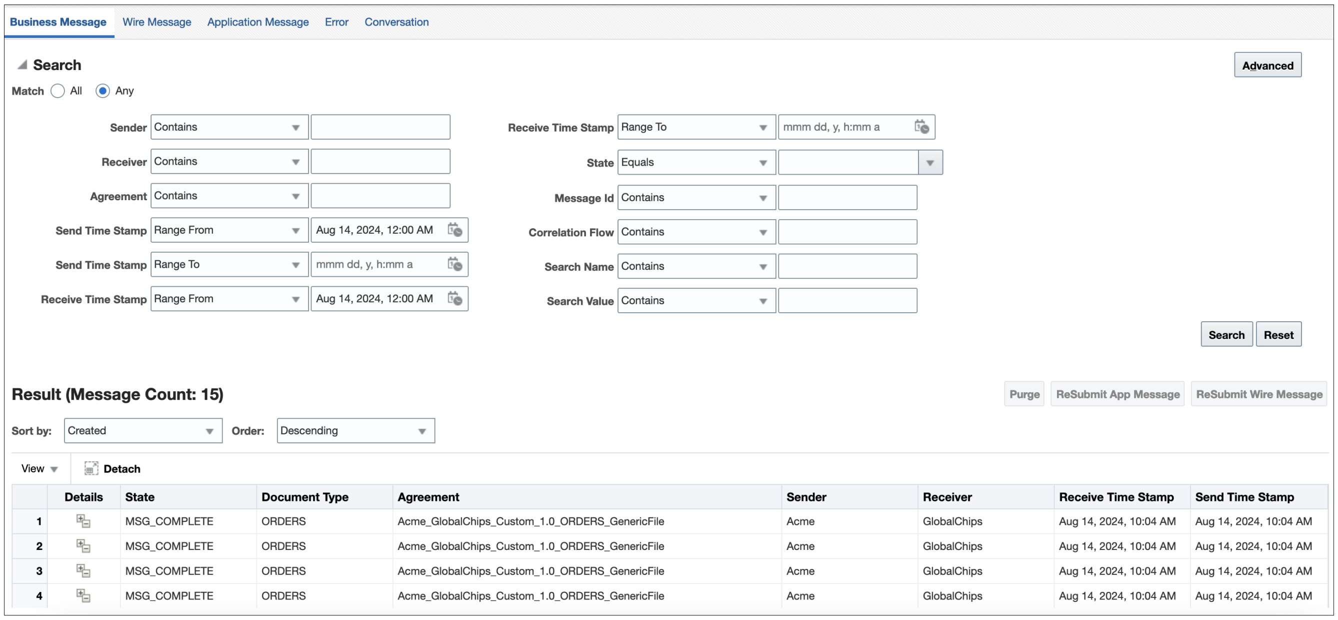
For Outbound Application and Wire resubmission, existing correlation information is re-used as it is the same business flow in the background that is used.

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 17-1 shows a business message report.

Figure 17-1 Business Message Report



Match: All [✓ Any]

Sender	Contains		Receive Time Stamp	Range To	mmm dd, y, h:mm a
Receiver	Contains		State	Equals	
Agreement	Contains		Message Id	Contains	
Send Time Stamp	Range From	Aug 14, 2024, 12:00 AM	Correlation Flow	Contains	
Send Time Stamp	Range To	mmm dd, y, h:mm a	Search Name	Contains	
Receive Time Stamp	Range From	Aug 14, 2024, 12:00 AM	Search Value	Contains	

Result (Message Count: 15)

Sort by: Created Order: Descending

	Details	State	Document Type	Agreement	Sender	Receiver	Receive Time Stamp	Send Time Stamp
1		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
2		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
3		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM
4		MSG_COMPLETE	ORDERS	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Acme	GlobalChips	Aug 14, 2024, 10:04 AM	Aug 14, 2024, 10:04 AM

To create a business message report:

1. Click **Reports**, and then **Business Message**.
2. Provide search parameters.

Field	Description
Generate URL from current search criteria	A URL is generated for the current search criteria. This URL can then be used for repeatable searches using the same search criteria without setting up the search criteria again.
Basic/Advanced	Click this toggle button to display a list of basic or advanced search criteria.
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** button.
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** button.

State	Select from the following message states (**Equals** is the only operator).
	`All`
	`MSG_COMPLETE`
	`MSG_ERROR`
	`MSG_WAIT_TRANSMIT`
	`MSG_WAIT_FA`
	`MSG_SEND_FA`
	`MSG_WAIT_BATCH`
	`MSG_INVALID`
	`MSG_CONTINUE_PROCESS`
	`MSG_COLLAB_WAIT`
	`MSG_PROCESS_ACK`
	`MSG_SEND_ACK`
	`MSG_WAIT_ACK`
	`MSG_SEND_EXP`
	`MSG_PROCESS_EXP`
	`MSG_ABORTED`
	`MSG_TRANSMITFAILED`
	`MSG_WAIT_STACK`
	`MSG_WAIT_TA1`
	`MSG_SEND_TA1`

Message ID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message ID.
Correlation Flow	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a correlation flow.

Field	Description
Search Name	This field enables user to search for the values of XPath Name1, XPath Name2, XPath Name3.
Search Value	Search for the values of XPath Value1, XPath Value2, XPath Value3 in B2B UI Reports.
Search	Click this button to start searching for the messages.
Reset	Click this button to reset the search criteria to the default state.

3. To add more search fields, click **Advanced** and select from **Add Fields**. The following list shows the fields available:
   - Document Remaining Retry
   - Batch Id
   - Content Type
   - Sender Id Type
   - XPath Name1
   - XPath Name2
   - XPath Name3
   - Acknowledgement Mode
   - Delivered Endpoint
   - Created
   - Receiver Value
   - STATUS
   - Native Message Size
   - Collaboration Name
   - Document Protocol Version
   - Collaboration State
   - Correlation To XPath Value
   - Payload Name
   - Business Action Name
   - Group Control Number
   - Label
   - Document Definition
   - Modified
   - Attribute1, 2, 3, 4, 5
   - Business Transaction Name
   - Translated Message Size
   - Document Retry Interval
   - Business Transaction Id
   - Queue Time
   - Processing Time

- Business Transaction Version
- Correlation From XPath Name
- Exchange Protocol Name and Version
- Doc Attempt Count
- Refer To Message Id
- Job Id
- Protocol Work Area
- Direction
- Response Mode
- Sender Value
- Document Protocol Name

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.

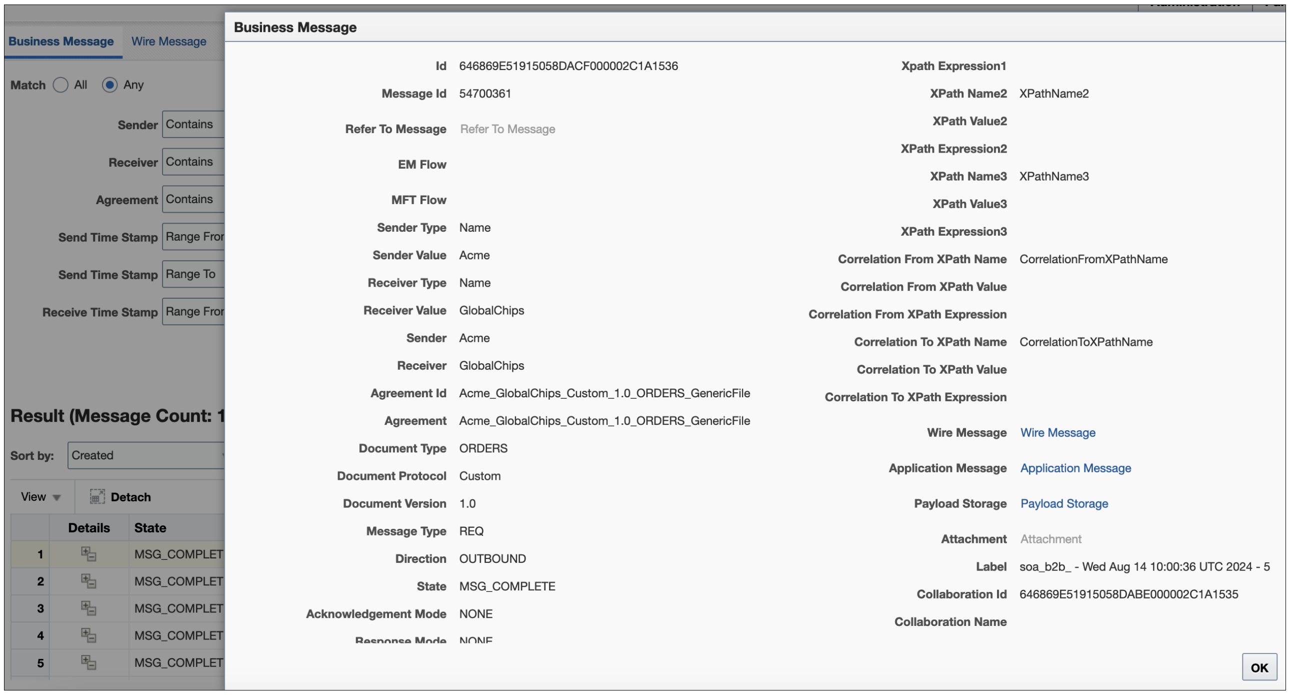
4. Click **Search**.

   View the results, as shown in Figure 17-1.

5. In the **Details** column of the **Results** area, click the button to see report details.

   Figure 17-2 shows the business message details. The screen is in two columns, which makes reading it easier, without the need to scroll.

Figure 17-2 Business Message Details



Sender	Contains
Receiver	Contains
Agreement	Contains
Send Time Stamp	Range Fror
Send Time Stamp	Range To
Receive Time Stamp	Range Fror

Result (Message Count: 1

Sort by: Created

	Details	State
1		MSG_COMPLET
2		MSG_COMPLET
3		MSG_COMPLET
4		MSG_COMPLET
5		MSG_COMPLET

Business Message

Id	646869E51915058DACF000002C1A1536	Xpath Expression1	
Message Id	54700361	XPath Name2	XPathName2
Refer To Message	Refer To Message	XPath Value2	
EM Flow		XPath Expression2	
MFT Flow		XPath Name3	XPathName3
Sender Type	Name	XPath Value3	
Sender Value	Acme	XPath Expression3	
Receiver Type	Name	Correlation From XPath Name	CorrelationFromXPathName
Receiver Value	GlobalChips	Correlation From XPath Value	
Sender	Acme	Correlation From XPath Expression	
Receiver	GlobalChips	Correlation To XPath Name	CorrelationToXPathName
Agreement Id	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Correlation To XPath Value	
Agreement	Acme_GlobalChips_Custom_1.0_ORDERS_GenericFile	Correlation To XPath Expression	
Document Type	ORDERS	Wire Message	Wire Message
Document Protocol	Custom	Application Message	Application Message
Document Version	1.0	Payload Storage	Payload Storage
Message Type	REQ	Attachment	Attachment
Direction	OUTBOUND	Label	soa_b2b_ - Wed Aug 14 10:00:36 UTC 2024 - 5
State	MSG_COMPLETE	Collaboration Id	646869E51915058DABE000002C1A1535
Acknowledgement Mode	NONE	Collaboration Name	
Response Mode	NONE		

Message State Definition

Table 17-1 provides message state definitions.

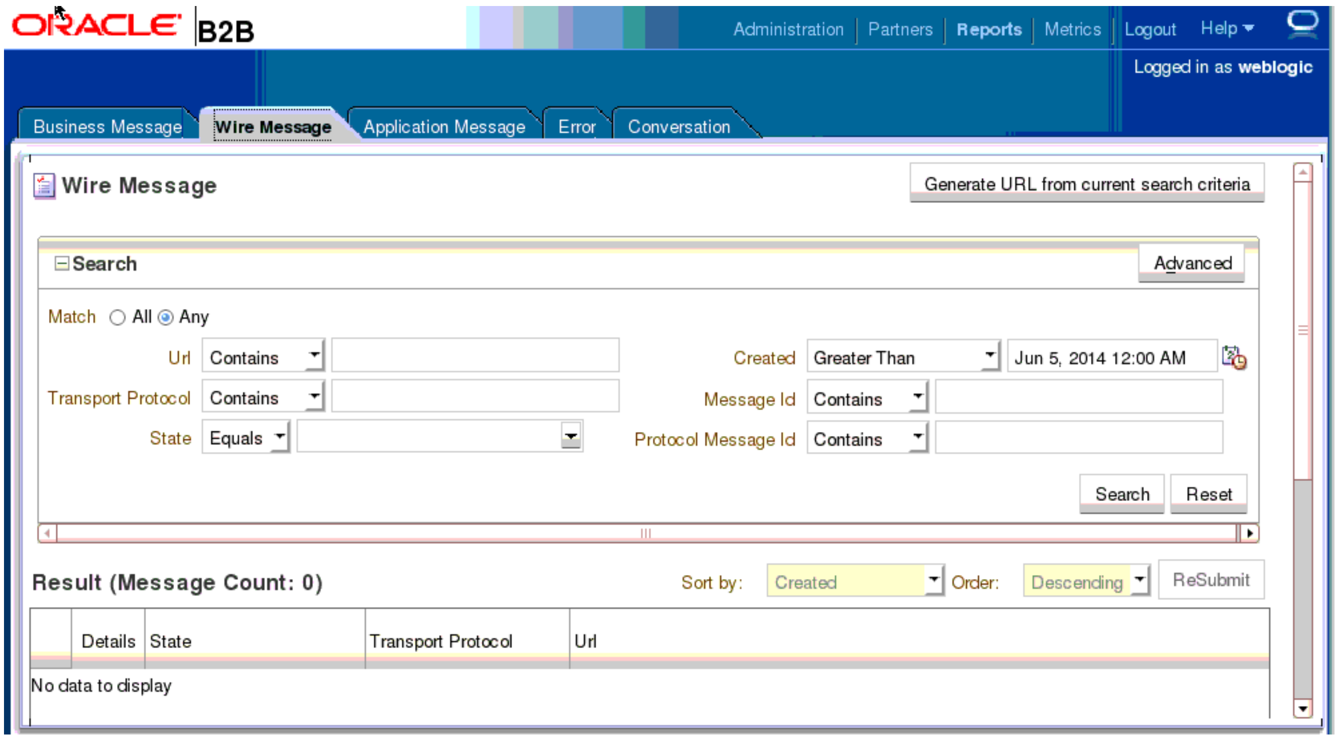
Table 17-1 Message State Definitions

Message	Descriptions
Stable status messages	-
MSG_COMPLETE	Business message state after completion of message transfer (and after receiving acknowledgment of the transmission if there is Ack/FA). The wire message state is moved to MSG_COMPLETE state as well.
MSG_WAIT_ACK	Business message state when an outbound message is sent to trading partner and B2B waits for Ack to be received.
MSG_ERROR	Business message state after a problem occurs in B2B or a negative acknowledgement is received from the trading partner. Wire message state is moved to MSG_ERROR state as well.
Intermittent status messages	-
MSG_WAIT_TRANSMIT	Business message state while B2B is sending message to trading partner. This state is also observed for the messages when they are queued in case of sequencing.
MSG_WAIT_FA	Business message state when an outbound message is sent to trading partner and B2B waits for Functional Ack to be received.
MSG_SEND_FA	Business message state when B2B is sending Functional Ack to trading partner.
MSG_WAIT_BATCH	Business message state while messages are batched up during the interval before batch expiration. After the batch expires, the entire batch of messages are sent out, and business and wire message states move to MSG_COMPLETE (or MSG_ERROR if a problem occurs).
MSG_WAIT_STACK	Business message state when there is any transport error in case of sequencing.
MSG_WAIT_TA1	Business message state while waiting for TA1 Message for EDI-X12.
MSG_SEND_TA1	Business message state while sending TA1 Message for EDI-X12.
MSG_CONTINUE_PROCESS	Business message state when message is being processed in B2B (engine).
MSG_COLLAB_WAIT	Business message state when message is waiting for Collaboration.
MSG_PROCESS_ACK	Business message state while processing an Acknowledgement.
MSG_SEND_ACK	Business message state when an inbound message is received from trading partner and B2B is sending an Acknowledgement.
MSG_SEND_EXP	Business message state while sending an Exception message.
MSG_PROCESS_EXP	Business message state while Processing an Exception Message
MSG_INVALD	This state is the default/first state when the message processing begins in B2B. This state should not be encountered while monitoring the Message states in B2B.
MSG_ABORTED	The message has been aborted.

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 17-3 shows a wire message report.

Figure 17-3 Wire Message Report



See Table 17-1 for a list of applicable message states.

To create a wire message report:

1. Click **Reports**, and then **Wire Message**.
2. Provide search parameters.

Field	Value
Generate URL from current search criteria	A URL is generated for the current search criteria. This URL can then be used for repeatable searches using the same search criteria without setting up the search criteria again.
URL	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the URL.

Field	Value
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:
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** button.
Message ID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of a message ID.
Protocol message id	The Protocol message id is the message id specific to the protocol if it is defined in the exchange/transport protocol. For example, in the AS2 exchange protocol, the Protocol message id is the Message-ID field in the HTTP header (http://www.ietf.org/rfc/rfc4130.txt page 12). You can use this Message-ID to identify the message exchanged between 2 B2B systems without specifying the details of the payload such as purchase order number or control numbers. If there is no message id specified in the protocol such as the custom exchange and file protocol, the protocol message id is auto-generated by B2B. For an outbound file transport protocol, the file name is used as the protocol message id. For the inbound file transport protocol, the file name is appended with a UUID (for example, `prodatTXNUM.dat@0AF29CCD14586BE31190000075F71D16` comprised of `prodatTXNUM.dat` (filename) appended with `@0AF29CCD14586BE31190000075F71D16`

3. To add more search fields, click **Advanced** and select from **Add Fields**.

Field	Description
Document Protocol Name	Select available document protocol such as Custom, OAG, RosettaNet and UCCNet. (Equals is the only operator.)
Document Type	Select from a previously created document type. (**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.)

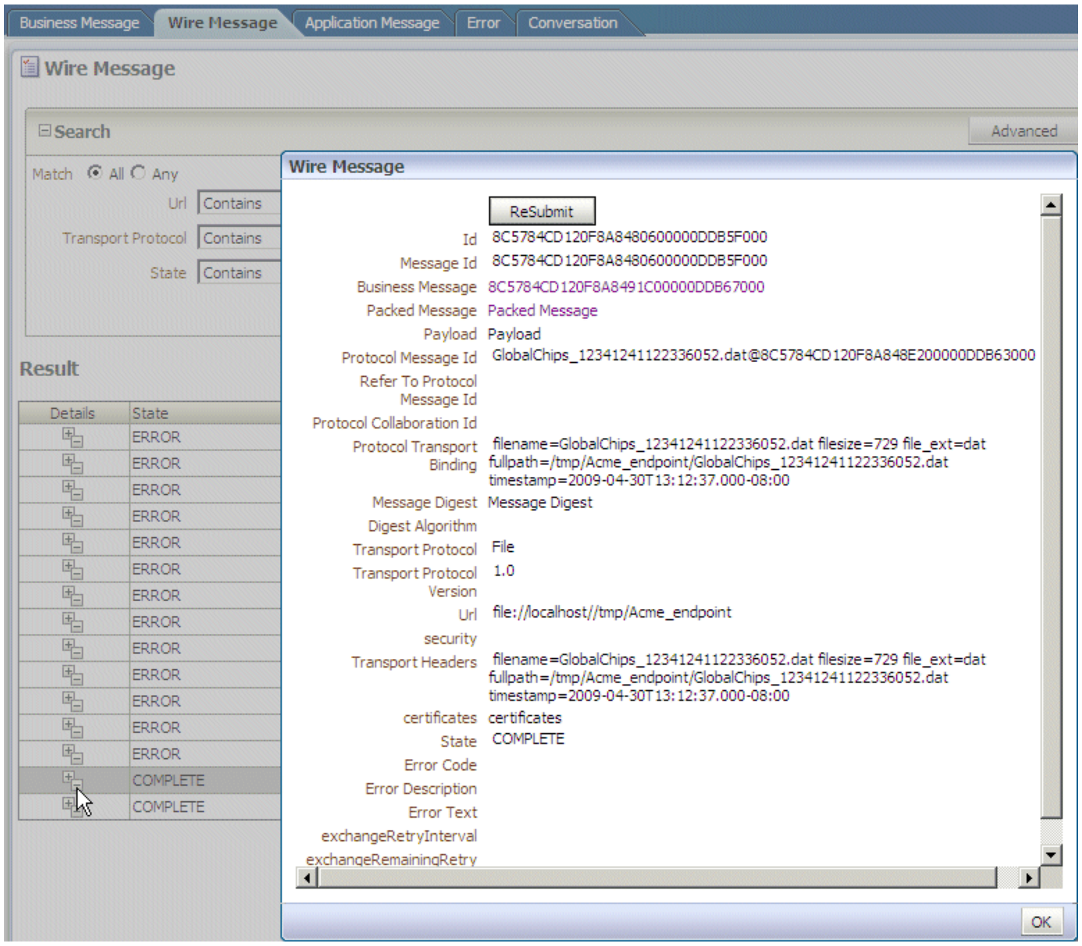
4. Click **Search**.

   View the results, as shown in Figure 17-3.

5. In the **Details** column of the **Results** area, click the button to see report details.

   Figure 17-4 shows wire message details.

Figure 17-4 Wire Message Details



Details	State
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	ERROR
	COMPLETE
	COMPLETE

Wire Message

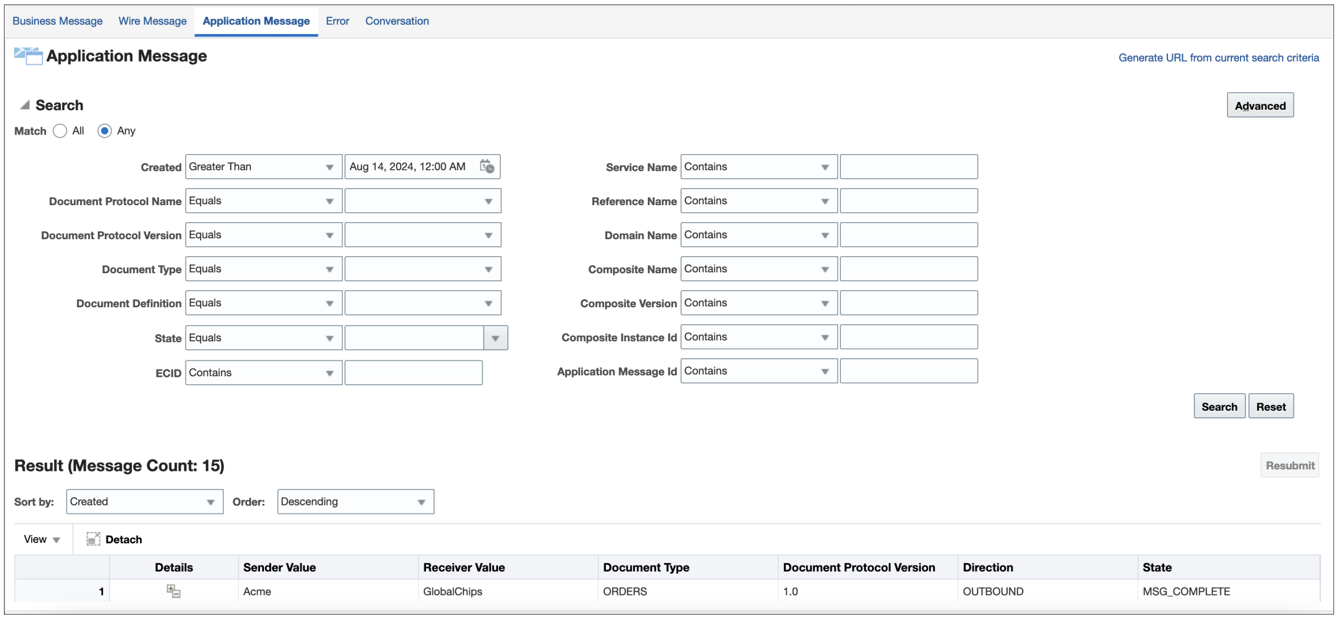
Id	8C5784CD120F8A8480600000DDB5F000
Message Id	8C5784CD120F8A8480600000DDB5F000
Business Message	8C5784CD120F8A8491C00000DDB67000
Packed Message	Packed Message
Payload	Payload
Protocol Message Id	GlobalChips_12341241122336052.dat@8C5784CD120F8A848E200000DDB63000
Refer To Protocol Message Id	
Protocol Collaboration Id	
Protocol Transport Binding	filename=GlobalChips_12341241122336052.dat filesize=729 file_ext=dat fullpath=/tmp/Acme_endpoint/GlobalChips_12341241122336052.dat timestamp=2009-04-30T13:12:37.000-08:00
Message Digest	Message Digest
Digest Algorithm	
Transport Protocol	File
Transport Protocol Version	1.0
Url	file://localhost//tmp/Acme_endpoint
security	
Transport Headers	filename=GlobalChips_12341241122336052.dat filesize=729 file_ext=dat fullpath=/tmp/Acme_endpoint/GlobalChips_12341241122336052.dat timestamp=2009-04-30T13:12:37.000-08:00
certificates	certificates
State	COMPLETE
Error Code	
Error Description	
Error Text	
exchangeRetryInterval	
exchangeRemainingRetry	

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 17-5 shows an application message report.

Figure 17-5 Application Message Report



Application Message

Generate URL from current search criteria

Search

Advanced

Match ○ All [✓ Any]

Created	Greater Than	Aug 14, 2024, 12:00 AM	Service Name	Contains	
Document Protocol Name	Equals		Reference Name	Contains	
Document Protocol Version	Equals		Domain Name	Contains	
Document Type	Equals		Composite Name	Contains	
Document Definition	Equals		Composite Version	Contains	
State	Equals		Composite Instance Id	Contains	
ECID	Contains		Application Message Id	Contains	

Search Reset

Result (Message Count: 15)

Resubmit

Sort by: Created Order: Descending

View Detach

	Details	Sender Value	Receiver Value	Document Type	Document Protocol Version	Direction	State
1		Acme	GlobalChips	ORDERS	1.0	OUTBOUND	MSG_COMPLETE

See Table 17-1 for a list of applicable message states.

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** button.
Document Protocol Name	Select from Custom, OAG, RosettaNet, or UCCNet. (**Equals** is the only operator.)
Document Protocol Version	Select from a previously created document protocol version. (**Equals** is the only operator.)
Document Type	Select from a previously created document type. (**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.
ECID	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide an instance ID.
Fabric CompositeDN	Do not use.
Service Name	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide the name of the B2B service binding component.

Field	Description
Reference Name	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide the name of the B2B reference binding component.
Domain Name	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide the name of the. basic administration unit that includes a special WebLogic Server instance called the Administration Server.
Composite Name	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide all or part of the SOA composite application name.
Composite Version	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide the version of the SOA composite application in Oracle JDeveloper.
Composite Instance Id	Select from **Starts With**, **Equals**, **Contains**, or **Ends With**. Provide the id of the composite application.

Note:

The application message pop-up in the B2B UI reports displays the following fields only when there is in-memory integration with fabric:

- Domain Name
- Composite Name
- Composite Version
- Service Name
- Reference 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	Do not use.

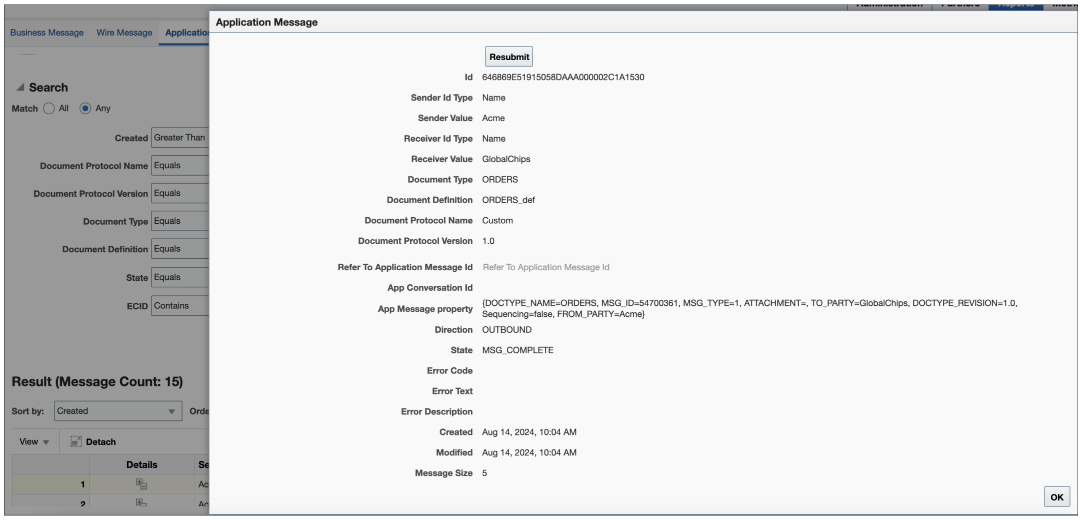
4. Click **Search**.

   View the results, as shown in Figure 17-5.

5. In the **Details** column of the **Results** area, click the button to see report details.

   Figure 17-6 shows application message details.

Figure 17-6 Application Message Details



Application Message

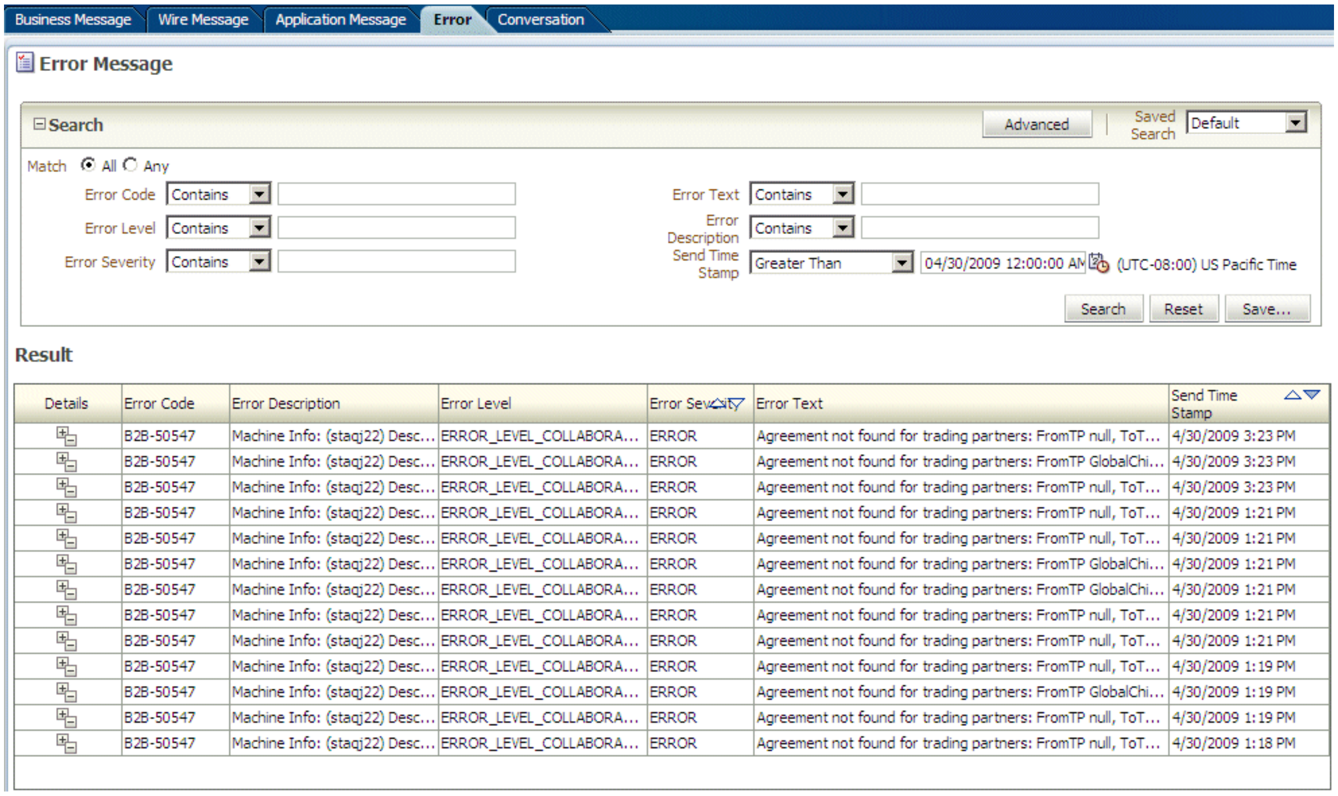
Id	646869E51915058DAAA000002C1A1530
Sender Id Type	Name
Sender Value	Acme
Receiver Id Type	Name
Receiver Value	GlobalChips
Document Type	ORDERS
Document Definition	ORDERS_def
Document Protocol Name	Custom
Document Protocol Version	1.0
Refer To Application Message Id	Refer To Application Message Id
App Conversation Id	
App Message property	{DOCTYPE_NAME=ORDERS, MSG_ID=54700361, MSG_TYPE=1, ATTACHMENT=, TO_PARTY=GlobalChips, DOCTYPE_REVISION=1.0, Sequencing=false, FROM_PARTY=Acme}
Direction	OUTBOUND
State	MSG_COMPLETE
Error Code	
Error Text	
Error Description	
Created	Aug 14, 2024, 10:04 AM
Modified	Aug 14, 2024, 10:04 AM
Message Size	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 17-7 shows an error report.

Figure 17-7 Error Report



Error Message

Search

Saved Search: Default

Match: [✓ All] Any

Error Code	Contains		Error Text	Contains	
Error Level	Contains		Error Description	Contains	
Error Severity	Contains		Send Time Stamp	Greater Than	04/30/2009 12:00:00 AM (UTC-08:00) US Pacific Time

Result

Details	Error Code	Error Description	Error Level	Error Severity	Error Text	Send Time Stamp
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 3:23 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP GlobalChi...	4/30/2009 3:23 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 3:23 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP GlobalChi...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP GlobalChi...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:21 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:19 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP GlobalChi...	4/30/2009 1:19 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:19 PM
	B2B-50547	Machine Info: (stagj22) Desc...	ERROR_LEVEL_COLLABORA...	ERROR	Agreement not found for trading partners: FromTP null, ToT...	4/30/2009 1:18 PM

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.
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** button.

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. (**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, OAG, RosettaNet, or UCCNet. (**Equals** is the only operator.)

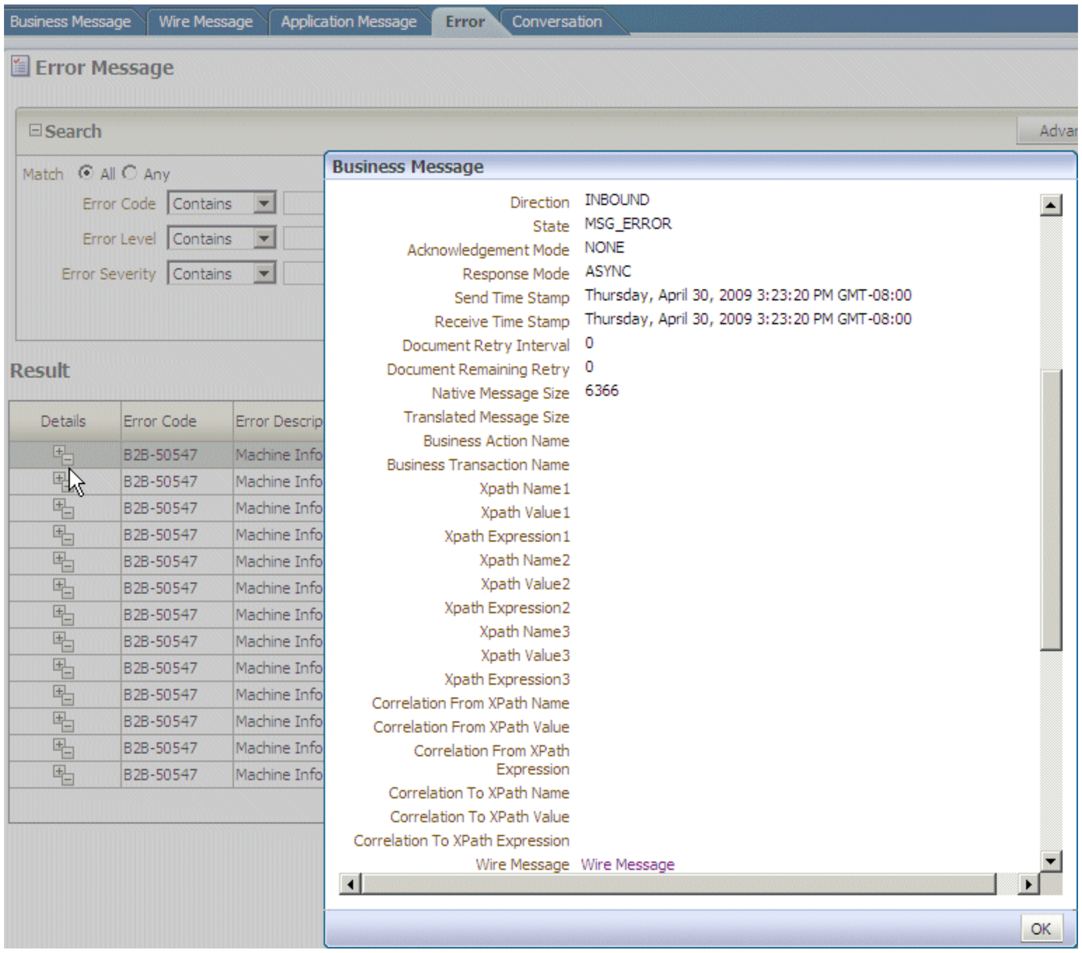
4. Click **Search**.

   View the results, as shown in Figure 17-7.

5. In the **Details** column of the **Results** area, click the button to see report details.

   Figure 17-8 shows error report details.

Figure 17-8 Error Report Details



Details	Error Code	Error Descrip
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info
⊞	B2B-50547	Machine Info

Business Message

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

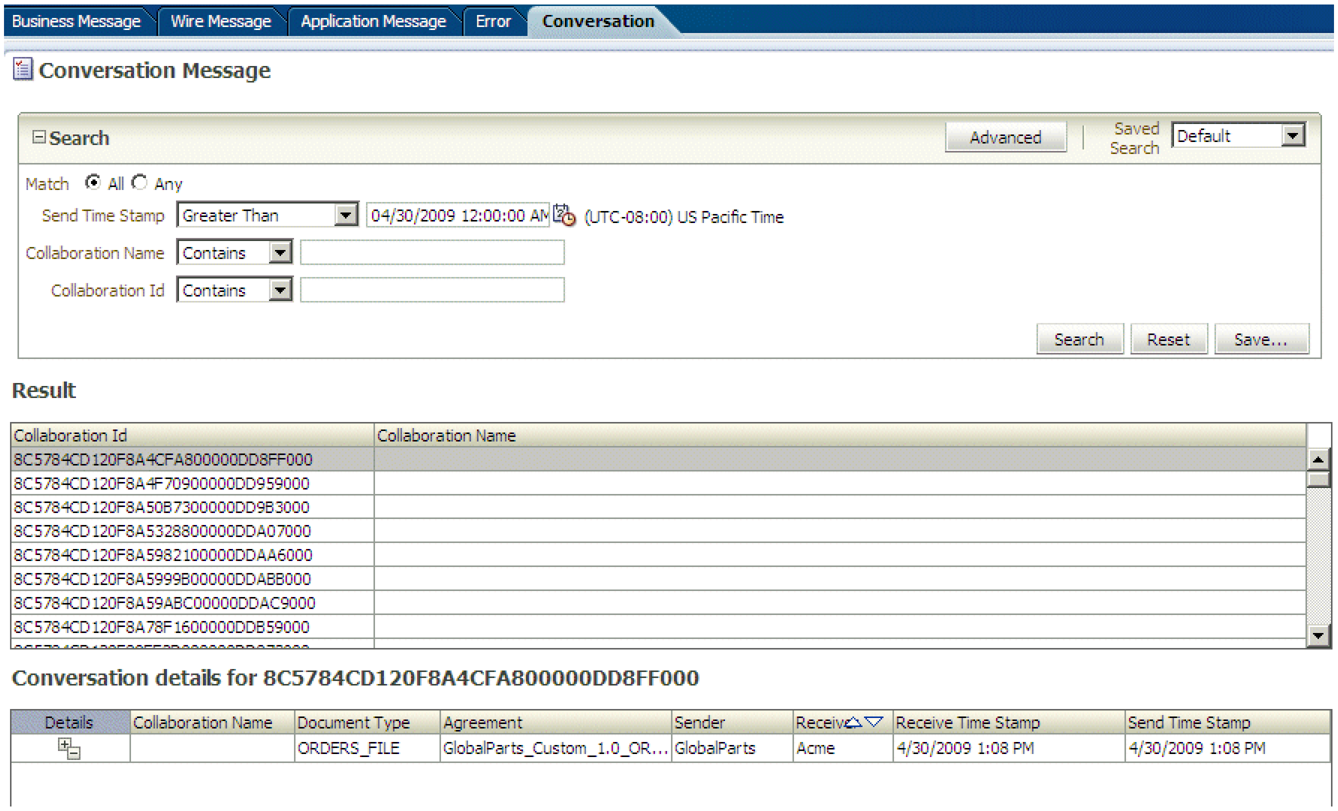
Creating Conversation Reports

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 17-9 shows a conversation report.

Figure 17-9 Conversation Report



Match [✓ All] Any

Send Time Stamp	Greater Than	04/30/2009 12:00:00 AM	(UTC-08:00) US Pacific Time
Collaboration Name	Contains		
Collaboration Id	Contains		

Result

Collaboration Id	Collaboration Name
8C5784CD120F8A4CFA800000DD8FF000	
8C5784CD120F8A4F70900000DD959000	
8C5784CD120F8A50B7300000DD9B3000	
8C5784CD120F8A5328800000DDA07000	
8C5784CD120F8A5982100000DDAA6000	
8C5784CD120F8A5999B00000DDABB000	
8C5784CD120F8A59ABC00000DDAC9000	
8C5784CD120F8A78F1600000DDB59000	

Conversation details for 8C5784CD120F8A4CFA800000DD8FF000

Details	Collaboration Name	Document Type	Agreement	Sender	Receiv	Receive Time Stamp	Send Time Stamp
⊞		ORDERS_FILE	GlobalParts_Custom_1.0_OR...	GlobalParts	Acme	4/30/2009 1:08 PM	4/30/2009 1:08 PM

To create a conversation report:

1. Click **Reports**, and then **Conversation**.
2. Provide search parameters.

Field	Description
Match	Select **All** or **Any**.

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** button.
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.

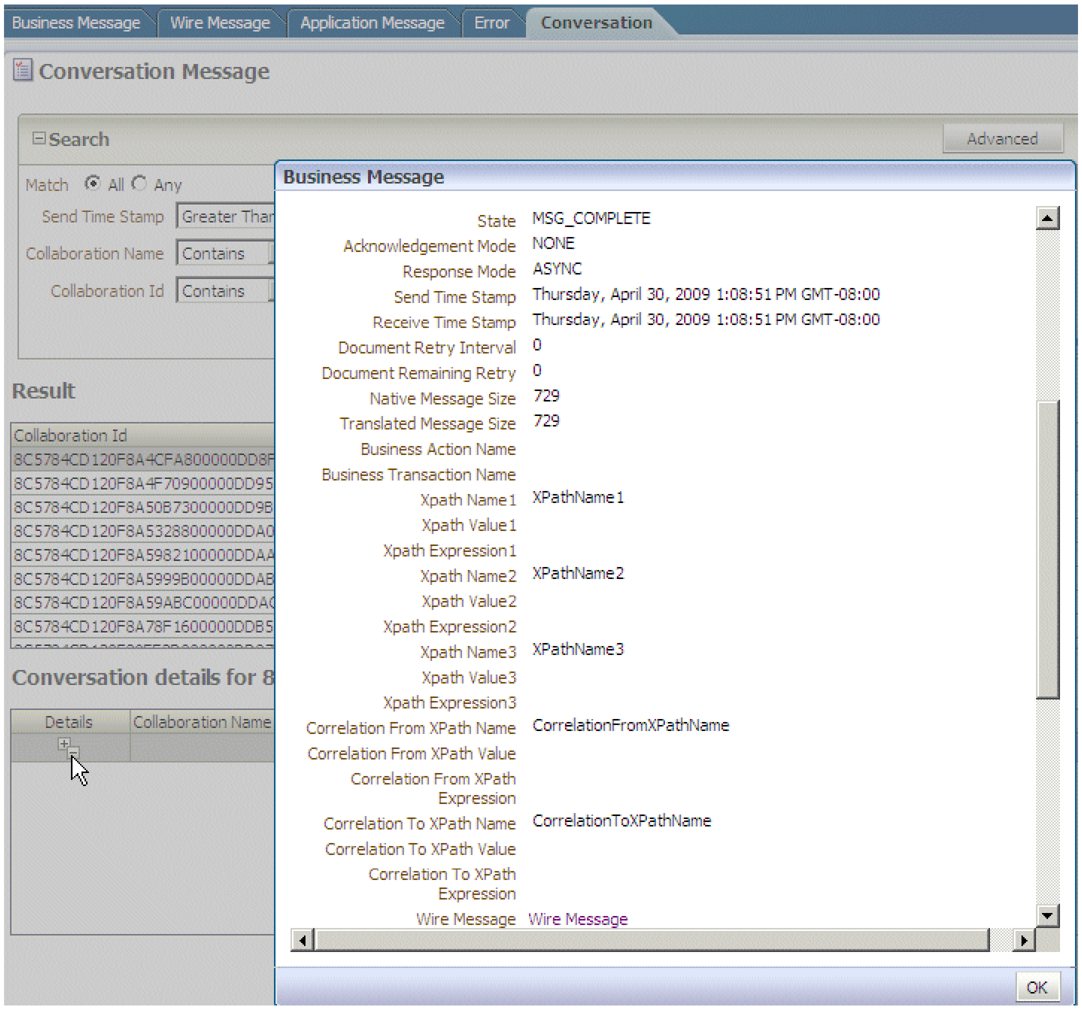
3. Click **Search**.

   View the results, as shown in Figure 17-9.

4. In the **Details** column of the **Results** area, click the button to see report details.

   Figure 17-10 shows conversation report details.

Figure 17-10 Conversation Report Details



Send Time Stamp	Greater Than
Collaboration Name	Contains
Collaboration Id	Contains

Result

Collaboration Id
8C5784CD120F8A4CFA800000DD8F
8C5784CD120F8A4F70900000DD95
8C5784CD120F8A50B7300000DD9B
8C5784CD120F8A5328800000DDA0
8C5784CD120F8A5982100000DDAA
8C5784CD120F8A5999B00000DDAB
8C5784CD120F8A59ABC00000DDAC
8C5784CD120F8A78F1600000DDB5

Conversation details for 8

Details	Collaboration Name

Business Message

State	MSG_COMPLETE
Acknowledgement Mode	NONE
Response Mode	ASYNC
Send Time Stamp	Thursday, April 30, 2009 1:08:51 PM GMT-08:00
Receive Time Stamp	Thursday, April 30, 2009 1:08:51 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

18

Using B2B Metrics

This chapter describes how Oracle B2B metrics provide system-level and partner-level status on B2B runtime data. It also discusses status on messages and errors, message counts, active document types and trading partners, and error messages.

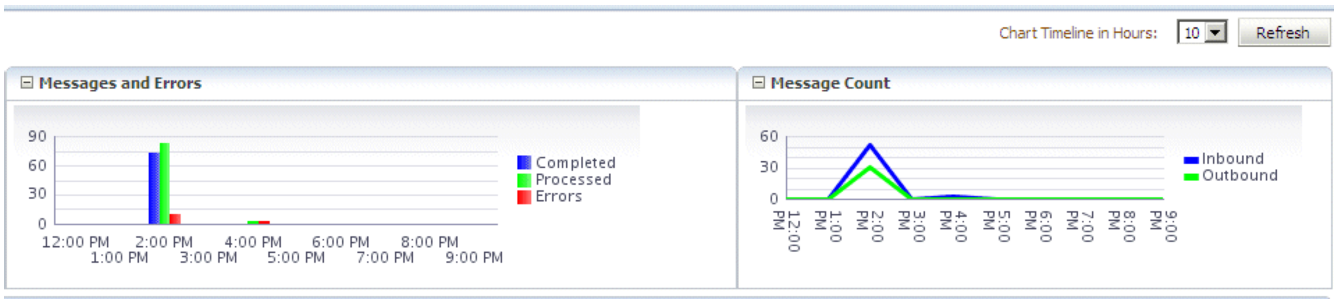
This chapter includes the following sections:

Introduction to B2B Metrics

Use the **Metrics** tab to view current runtime data in the repository. The **Metrics** tab reflects changes that occur in the runtime repository (for example, purging the runtime instance data).

Metrics data shown in the **Messages and Errors** chart and the **Message Count** chart, shown in Figure 18-1, display data for the last 10 hours or the last 20 hours.

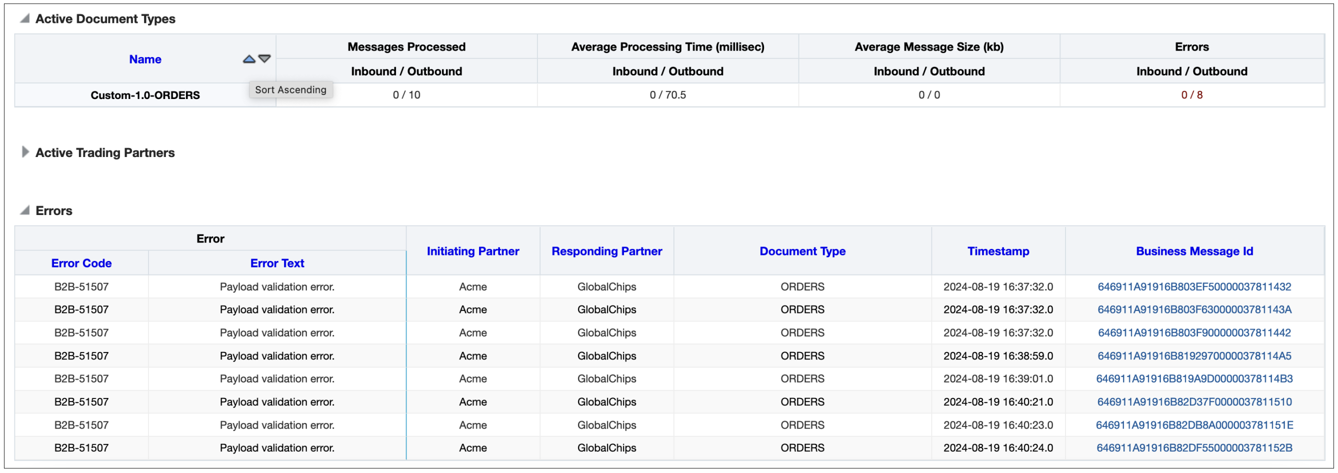
Figure 18-1 The Messages and Errors Chart and Message Count Chart



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 Monitoring Oracle 2B in *Administering 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 18-2.

Figure 18-2 Sorting Columns



Active Document Types

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound
Custom-1.0-ORDERS	0 / 10	0 / 70.5	0 / 0	0 / 8

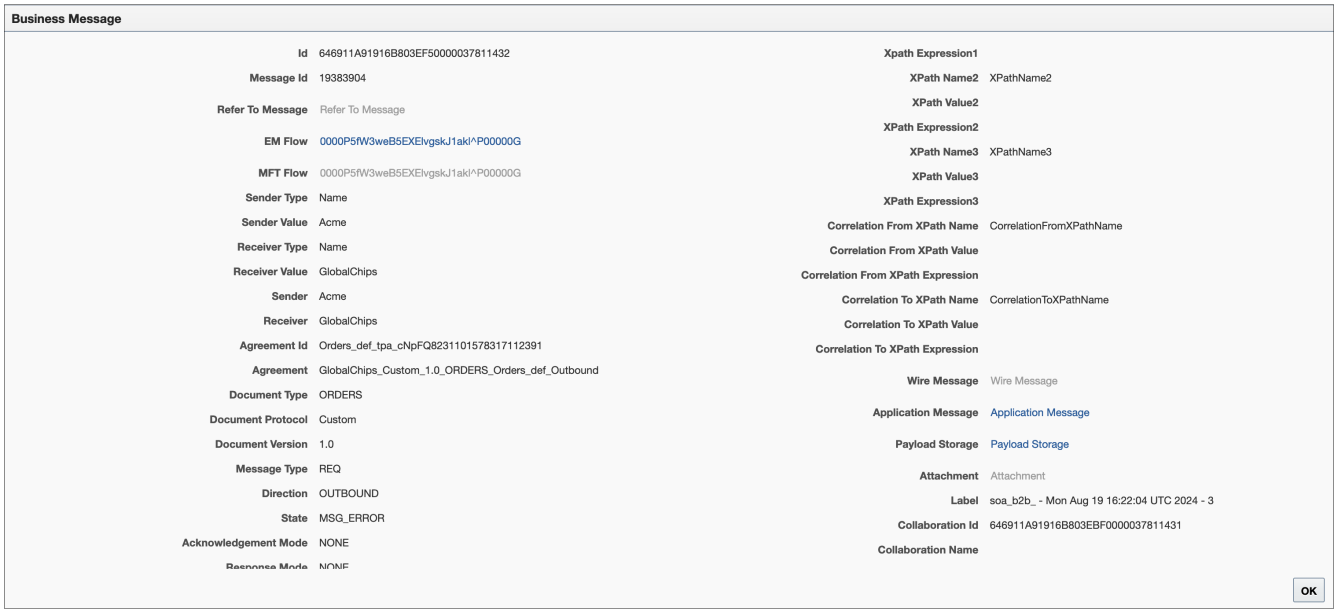
Active Trading Partners

Errors

Error		Initiating Partner	Responding Partner	Document Type	Timestamp	Business Message Id
Error Code	Error Text					
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:37:32.0	646911A91916B803EF50000037811432
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:37:32.0	646911A91916B803F63000003781143A
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:37:32.0	646911A91916B803F90000037811442
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:38:59.0	646911A91916B81929700000378114A5
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:39:01.0	646911A91916B819A9D00000378114B3
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:40:21.0	646911A91916B82D37F0000037811510
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:40:23.0	646911A91916B82DB8A000003781151E
B2B-51507	Payload validation error.	Acme	GlobalChips	ORDERS	2024-08-19 16:40:24.0	646911A91916B82DF55000003781152B

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 18-3.

Figure 18-3 Business Message Details



Business Message

Id	646911A91916B803EF50000037811432
Message Id	19383904
Refer To Message	Refer To Message
EM Flow	0000P5fW3weB5EXElvgskJ1akl^P00000G
MFT Flow	0000P5fW3weB5EXElvgskJ1akl^P00000G
Sender Type	Name
Sender Value	Acme
Receiver Type	Name
Receiver Value	GlobalChips
Sender	Acme
Receiver	GlobalChips
Agreement Id	Orders_def_tpa_cNpFQ8231101578317112391
Agreement	GlobalChips_Custom_1.0_ORDERS_Orders_def_Outbound
Document Type	ORDERS
Document Protocol	Custom
Document Version	1.0
Message Type	REQ
Direction	OUTBOUND
State	MSG_ERROR
Acknowledgement Mode	NONE
Response Mode	NONE

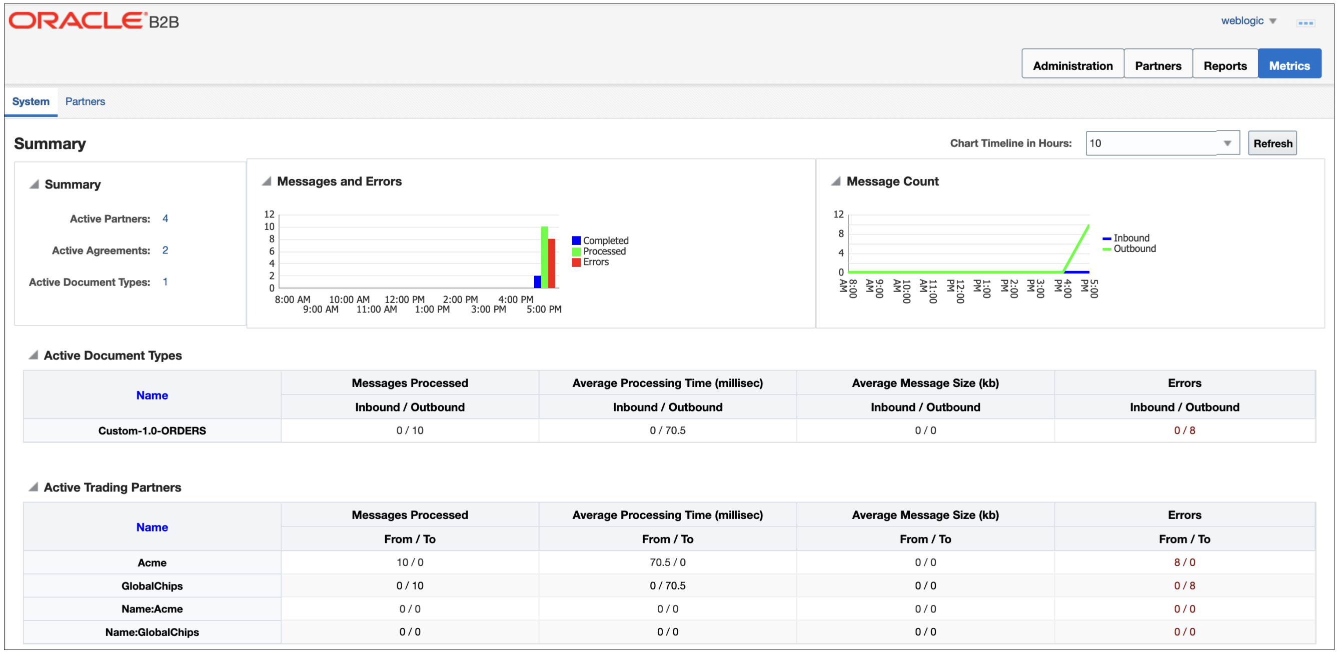
Xpath Expression1	
XPath Name2	XPathName2
XPath Value2	
XPath Expression2	
XPath Name3	XPathName3
XPath Value3	
XPath Expression3	
Correlation From XPath Name	CorrelationFromXPathName
Correlation From XPath Value	
Correlation From XPath Expression	
Correlation To XPath Name	CorrelationToXPathName
Correlation To XPath Value	
Correlation To XPath Expression	
Wire Message	Wire Message
Application Message	Application Message
Payload Storage	Payload Storage
Attachment	Attachment
Label	soa_b2b_ - Mon Aug 19 16:22:04 UTC 2024 - 3
Collaboration Id	646911A91916B803EBF0000037811431
Collaboration Name	

B2B System Metrics

System metrics summary data includes messages and errors, message counts, and statistics on active document types and active trading partners.

Figure 18-4 shows system metrics summary data.

Figure 18-4 System Metrics



Summary

Chart Timeline in Hours: 10

Summary

Active Partners: 4

Active Agreements: 2

Active Document Types: 1

Active Document Types

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound	Inbound / Outbound
Custom-1.0-ORDERS	0 / 10	0 / 70.5	0 / 0	0 / 8

Active Trading Partners

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	From / To	From / To	From / To	From / To
Acme	10 / 0	70.5 / 0	0 / 0	8 / 0
GlobalChips	0 / 10	0 / 70.5	0 / 0	0 / 8
Name:Acme	0 / 0	0 / 0	0 / 0	0 / 0
Name:GlobalChips	0 / 0	0 / 0	0 / 0	0 / 0

Table 18-1 describes the information on the **System** metrics tab.

Table 18-1 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.

Table 18-1 (Cont.) B2B System Metrics

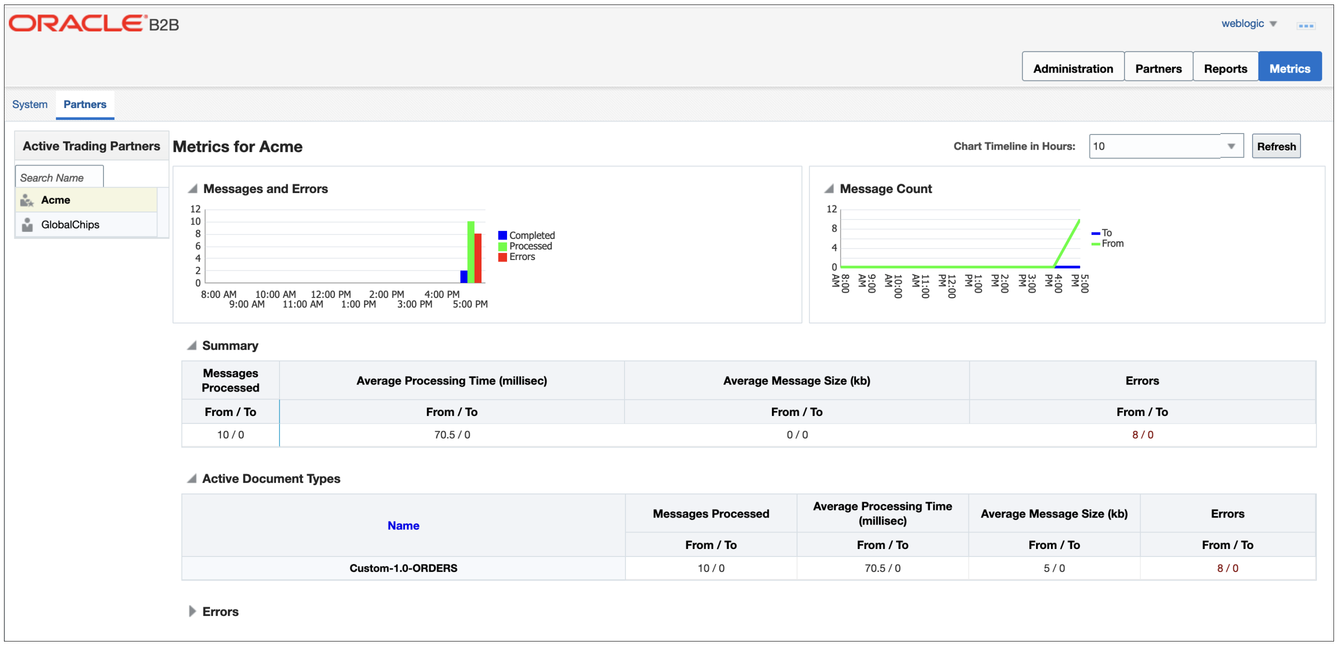
Area/Field	Description
Active Trading Partners	Active trading partners are partners for which an agreement has been deployed and is in an active state. The host trading partner is included in the list. Messages processed include completed plus errored messages, that is, active messages.
Name	Name of 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.

B2B Partner Metrics

Metrics summary data for a trading partner includes messages and errors, message counts, statistics for active document types, and errors.

Figure 18-5 shows metrics summary data for a selected trading partner.

Figure 18-5 Partner Metrics



Metrics for Acme

Chart Timeline in Hours: 10

Summary

Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
From / To	From / To	From / To	From / To
10 / 0	70.5 / 0	0 / 0	8 / 0

Active Document Types

Name	Messages Processed	Average Processing Time (millisec)	Average Message Size (kb)	Errors
	From / To	From / To	From / To	From / To
Custom-1.0-ORDERS	10 / 0	70.5 / 0	5 / 0	8 / 0

Note:

Discrepancies between the Summary counts and the Active Document Types detail counts are due to a variety of factors, such as:

- Messages for which there in no agreement show in the Summary but not in the Active Document Types list.
- Acknowledgement messages for agreements show in the Summary but not in the Active Document Types list.
- Agreements that are deactivated prior to being sent to the Metrics page show in the Summary but not in the Active Document Types list.
- The set of messages used for the Summary may be different than the set of messages shown in Average Processing Times.

Table 18-2 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:

19 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, deploying agreements, and so on.

This chapter includes the following sections:

Prerequisites for Running the Command-line Tools

There are several things you should do before you run 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 correct values for each of the properties present in the file.

Note:

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. In summary, if you are executing the B2B command utility and a user session is opened in the UI simultaneously, then it is advisable for the user to log out of the B2B UI and log back in again to see the candidate changes done by the command utility.

4. Most of the command-line tools cannot be run without providing the JNDI credentials. To restrict the command-line tools from anonymous use, provide the following information in the `jndi.properties` file:

```
java.naming.provider.url=t3://localhost:8001
java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory
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`.

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. This is not a permanent directory. Once the archive procedure is complete, the archived files can be moved to another location, and this directory can be deleted, if necessary for security purposes. For example:

   ```
   mkdir /archive
   ```

2. Use the `chmod` command to grant permissions to this directory so that the database process can write to it for this operation.
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 '/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 19-1 lists the options for this utility.

Table 19-1 Options for B2B_ARCHIVE_INSTANCE_MGS Utility

Options	Type	Description
`startDate`	`varchar2`	Starting date for archival.
`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`	The sender `ID_TYPE`, or identifier type, which uniquely identifies a trading partner and defines how to exchange documents.
`idVal`	`varchar2`	The sender `ID_VALUE`, or identifier value. The value associated with the identifier type.
`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.

Purging Data

Before purging runtime data, you must set up the configuration properly. If not done properly, then the archive will fail with a misleading error.

Note:

The configuration setup isdescribed in Archiving Data.

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 runtime data and resets the environment to the installation time.

```
ant -f ant-b2b-util.xml b2bpurge
```

Table 19-2 lists the options for this utility.

Table 19-2 Options for b2bpurge Utility

Option	Description	Domain	Required
`mode`	Specifies purging design-time or runtime data. (see Note below)	`DT` `RT`	Yes

`msgState`	Deletes messages with the specified message state. Used for runtime 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 runtime data. When set to true, all the runtime data is truncated from B2B tables. This option not available when data is deleted for a specific date range. Archiving will only occur is this option is set to false.	`true` `false` (default)	No
`fromdate`	Deletes all messages created on or after this date.	Date format `dd/mm/yyyy hh:mm AM/PM`	No
`todate`	Deletes all messages created on or before this date.	Date format `dd/mm/yyyy hh:mm AM/PM`	No
`tp`	Trading partner name.	-	No
`direction`	Direction of the message	-	No

Table 19-2 (Cont.) Options for b2bpurge Utility

Option	Description	Domain	Required
`msgtype`	Type of the message	-	No
`agreement`	Name of the agreement	-	No
`idtype`	The sender `ID_TYPE`, or identifier type, which uniquely identifies a trading partner and defines how to exchange documents.	-	No
`idvalue`	The sender `ID_VALUE`, the value associated with the identifier type.	-	No
`archive`	Specifes whether or not to archive.	`true`(default) `false`	No
`commitfrequency`	Specifies the number of records after which the database commit takes place	Value of the frequency; default is `5000`	No
`archivename`	File name of archived file	-	No
`partitioned`	Indicates if the database is partitioned. Based on the value of this parameter, Oracle B2B allows you to purge data normally or based on database partitions.	`true` `false`	No
`partitioncleanmode`	Specifies the mode of cleaning for a partition that is identified for cleanup. In case of absence of partitions, or non-date parameters are provided, this parameter is ineffective. **Note:** This parameter is effective only when the `partitioned` flag is set to `true`.	`DROP`: Indicates that the clean up mechanism is to drop the partition (Default). `TRUNCATE`: Cleans up the partition by truncating it rather than deleting it. `STATEMENT`: Statement mode does not clean up the tables, but only creates a `drop` statement and writes the statement to a file in the SOA_PURGE_DIR directory. In this mode, no change is made to the database.	No

`logmode`	Indicates the log level. There are two variations, `CONSOLE_*` that uses the `dbms_output` mode of logging (applicable only when the procedure is invoked directly through a client) and the other modes that log to a file in a SOA_PURGE_DIR directory.	`ERROR`
		`INFO`
		`DEBUG`
		`CONSOLE_ERROR`
		`CONSOLE_INFO`
		`CONSOLE_DEBUG`
			No

`refreshmw`	Controls whether to completely refresh the `b2b_system` materialized view.	`true` (default; when any truncate or drop operation is performed) `false`	No

Table 19-2 (Cont.) Options for b2bpurge Utility

Option	Description	Domain	Required
`rowlimit`	Limits the number of records to be deleted; this is a nonimportation-specific feature	-	No
`timelimit`	Limits the time the purge job runs. The accuracy depends on `commitfreq` because the check is performed after each commit cycle; this is a nonpartition-specific feature	-	No
`cascadedelete`	Deletes all the artifacts related to a specific document ID or CPA ID.	`true` `false`	No

Example 19-1 Removes Design-Time Data

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=DT
```

Example 19-2 Purges Runtime Data

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT
```

Example 19-3 Purges Runtime Data, Including Control Numbers

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT -Dpurgecontrolnumber=true
```

Example 19-4 Purges Messages with the Specified State Between the Specified Dates

```
ant -f ant-b2b-util.xml b2bpurge -Dmode=RT -Dfromdate="05/10/2012 11:51 PM" -
Dtodate="06/10/2012 12:01 AM" -Dmsgstate=MSG_COMPLETE -Darchive=false
```

Example 19-5 Cascade Deletes the Robot143 Trading Partner

```
ant -f ant-b2b-util.xml b2bpurge -Dtp=Robot143 -Dmode=DT -Dcascadedelete=true
```

This is useful when there is one document or CPA attached with the trading partner.

Example 19-6 Cascade Deletes the Robot131 to Robot139 Trading Partners

```
ant -f ant-b2b-util.xml b2bpurge -
Dtp=Robot131,Robot132,Robot133,Robot134,Robot135,Robot136,Robot137,Robot138,Robot139 -
Dmode=DT -Dcascadedelete=true
```

This is useful when there is one document or CPA attached with the trading partners.

Example 19-7 Cascade Deletes All the Artifacts of an AgreementID or CPA

```
ant -f ant-b2b-util.xml b2bpurge -Dtp=Robot142 -Dmode=DT -Dcascadedelete=true -
Dagreementid=Openreach_Robot142_WholesaleProvider_ServiceProvider_R1800.0.5_cpa
```

This is useful when multiple documents or CPA are attached to a trading partner.

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 Prerequisites for Running the Command-line Tools and the initial steps listed in 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.

Note:

When performing purging of data based on the trading partner name:

- If the direction is specified, then direction is also considered for the purge.
- If the direction is not specified, then purge happens in both direction (inbound and outbound).
- If the hostname is provided as the value for `-Dtp`, an error message is displayed.

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.

Note:

No security or permission checks are performed to prevent the logged-in user from importing data.

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 19-3 lists the options for this utility.

Table 19-3 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

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 19-4 lists the options for this utility.

Table 19-4 Options for b2bexport Utility

Option	Description	Domain	Required
`tpanames`	One or more trading partner 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 name of the trading partner to be exported.	-	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

Table 19-4 (Cont.) Options for b2bexport Utility

Option	Description	Domain	Required
`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 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 19-8 Export Entire Repository with Policy Details to /tmp/export.zip

```
ant -f ant-b2b-util.xml b2bexport -Dexportfile="/tmp/export.zip" -Dpolicies=true
```

Example 19-9 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 19-10 Exports the Trading Partner Acme to /tmp/Acme.zip

```
ant -f ant-b2b-util.xml b2bexport -Dtpname="Acme" -Dexportfile="/tmp/Acme.zip"
```

Example 19-11 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 19-12 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.

Resetting Channel Passwords

This utility sets or resets a channel password.

```
ant -f ant-b2b-util.xml resetchannelpassword
```

Table 19-5 lists the options for this utility.

Table 19-5 Options for resetchannelpassword Utility

Options	Description	Required
`channelname`	Existing channel name.	Yes

Table 19-5 (Cont.) Options for resetchannelpassword Utility

Options	Description	Required
`tp`	Trading partner name.	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 19-13 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 19-14 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.

Activating or Deactivating Listening Channels

This utility activates or deactivates a listening channel.

```
ant -f ant-b2b-util.xml updatechannel
```

Table 19-6 lists the options for this utility.

Table 19-6 Options for updatechannel Utility

Options	Description	Value	Required
`channelname`	Name of the existing listening channel. You can use the wildcard character * to specify all the listening channels in the system.	-	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 19-15   Starting a Listening Channel

```
ant -f ant-b2b-util.xml updatechannel -Dchannelname="AcmeInboundListening" -Dstate=active
```

The preceding example starts the `AcmeInboundListening` channel.

Example 19-16   Starting All Listening Channels

```
ant -f ant-b2b-util.xml updatechannel -Dchannelname="*" -Dstate=active
```

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 19-7 lists the options for this utility.

Table 19-7   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 19-17   Deploys the Agreements Acme_GC_Agreement1 and GC_Acme_Agreement1

```
ant -f ant-b2b-util.xml b2bdeploy -Dtpanames="Acme_GC_Agreement1,GC_Acme_Agreement1"
```

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 [-Dmdsreference="comma_separated_argumants"]
```

Table 19-8 lists the options for this utility.

Table 19-8   Options for b2bvalidate Utility

Options	Description	Domain	Required
`mdsreference`	File names of the trading partner, agreement, or document protocol.	File names must be separated by a comma.	No

Table 19-8 (Cont.) Options for b2bvalidate Utility

Options	Description	Domain	Required
`agreementid`	Validates agreements based on the IDs provided. It also throws warning message if any agreementID does not exist and validate the others.	Agreement IDs must be separated with commas.	No

Example 19-18 Validates All Agreements

```
ant -f ant-b2b-util.xml b2bvalidate
```

Example 19-19 Validates Agreement tpa_ID1234.xml

```
ant -f ant-b2b-util.xml b2bvalidate -Dmdsreference="tpa_ID1234.xml"
```

Example 19-20 Validates Trading Partner tp_MyCompany.xml and Agreement tpa_ID1234.xml

```
ant -f ant-b2b-util.xml b2bvalidate -Dmdsreference="tp_MyCompany.xml,tpa_ID1234.xml"
```

Example 19-21 Validates Agreements Having IDs as agreementID1 and agreementID2

```
ant -f ant-b2b-util.xml b2bvalidate -Dagreementid="agreementID1,agreementID2"
```

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.

Note that the CPP/CPA Filename should follow below naming convention while passing through the command-line:

```
JavaLetterOrDigit | '-' | '.' | ' '
```

`JavaLetterOrDigit` returns true from the `java.lang.Character.isJavaIdentifierPart` method.

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 19-9 lists the options for this utility.

Table 19-9 Options for b2bcreate-cpaprop Utility

Option	Description	Domain	Required
`propfile`	Property file that stores configuration details for `b2bcpaimport` and `b2bcpaexport`	-	Yes

Example 19-22 Creates a Property File Template That Is Used in the propfile Option

```
ant -f ant-b2b-util.xml b2bcreate-cpaprop
```

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

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.

```
oracle.tip.b2b.ebms.allDocumentParameter (Optional Property)
```

This property is used to improve the import performance. Set this property to false to stop generating the unwanted or unset parameters in soa.zip.

```
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 import from the remote trading partner profile.

After using the `b2bcpaimport` utility to obtain the OracleB2B zip file, when you import the zip file into the Oracle B2B console:

- Certificate alias will be available under **Administration** > **Types**
- Certificate alias <*value from b2bcpaimport profiles*> will be available under **Partners** > **Remote Trading Partner** > **Profile** tab > **Identifiers**

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.

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.

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 19-10 lists the options for this utility.

Table 19-10 Options for b2bcpaimport Utility

Option	Description	Domain	Required
`propfile`	Property file that stores configuration details for `b2bcpaimport` and `b2bcpaexport`	-	Yes

Example 19-23 Converts CPA-Formatted XML to an Oracle B2B ZIP File

```
ant -f ant-b2b-util.xml b2bcpaimport -Dpropfile="/tmp/cpp_cpa.properties"
```

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 19-11 lists the options for this utility.

Table 19-11 Options for b2bcpaimport Utility

Option	Description	Domain	Required
`propfile`	Property file that stores configuration details for `b2bcpaimport` and `b2bcpaexport`	-	Yes

Example 19-24 Converts an Oracle B2B ZIP File to a CPA-Formatted XML File

```
ant -f ant-b2b-util.xml b2bcpaexport -Dpropfile="/tmp/cpp_cpa.properties"
```

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 19-12 lists the options for this utility.

Table 19-12 Options for b2bcheckcpaid Utility

Option	Description	Domain	Required
`exportfile`	Oracle B2B metadata file	file name	Yes

Example 19-25 b2bcheckcpaid Utility

```
ant -f ant-b2b-util.xml b2bcheckcpaid -Dexportfile="soa_file.zip"
```

Creating Oracle B2B Metadata Based on selfservice.xsd

This utility creates the `selfservice.xsd` file, which is used to understand or create self service 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 Self Service Utility Protocols, Identifications, Security Specifications, and Parameters for information about self service protocols, identifications, and security specifications.

Table 19-13 lists the options for these utilities.

Table 19-13 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 19-26 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 19-27 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"
```

Using Self Service to Batch Create Document Protocols, Trading Partners, and Agreements

When using a multiple file approach, you must make sure that the names of the self service XML files are in following order, otherwise, an error claiming that the referenced object does not exist might be thrown.

1. Document Protocols self service XML files.
2. Trading Partner self service XML files (`HOST` must be first in this list).
3. Trading Partner Agreement self service 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 19-28 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"
```

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. Also, when running the command line `selfservice` with the input `selfservice.xml` that does not exist, the command line `selfservice` will be able to run successfully and an empty B2B zip file created. You should ensure the existence of a valid `selfservice.xml` file before running the command line `selfservice` utility. This note is specific to your running `selfservice` with the folder option `-Dinput="/folder"` (rather than running the utility against the XML file itself outside of the context of a folder.). That is, the valid input XML file is expected within the `/folder`. See Example 19-28.

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"
```

Resubmitting a Message

This utility resubmits an application message or a wire message for a selected business message.

```
ant -f ant-b2b-util.xml b2bresubmit
```

The resubmit count gets reflected in the Oracle B2B console reports for the application message or wire message.

Note:

The resubmit functionality for payload rectification will only work for outbound messages for application messages. 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
```

Note:

It is mandatory that you provide at least one message searching or matching criteria for resubmit (excluding `maxcount`, `exclresubmit`, and `mode`)

Table 19-14 lists the options for this utility.

Table 19-14 Options for b2bresubmit Utility

Option	Description	Domain	Required
`direction`	The direction of the message.	`INBOUND` `OUTBOUND`	-
`msgsource`	The message source.	`APPMSG` `WIREMSG`	-

Table 19-14 (Cont.) Options for b2bresubmit Utility

Option	Description	Domain	Required
`msgid`	The message ID.	-	-
`doctype`	Document type.	-	-
`msgstate`	Message state.	-	-
`fromdate`	The `sendTimestamp` of the start date of the message.	Date format must be provided within double quotes. `"dd/mm/yyyy hh:mm AM/PM"` **Note:** This cannot be a future date.	Yes, when `todate` is used.
`todate`	The `sendTimestamp` of the end date of the message	Date format must be provided within double quotes. `"dd/mm/yyyy hh:mm AM/PM"` **Note:** `todate` should be greater/later than `fromdate`. You can provide both dates.	Yes, when `fromdate` is used
`agreement`	Agreement name.	-	-
`payloadpath`	This option is applicable for outbound application message resubmission, by providing the rectified file path.	-	-
`tp`	The trading partner name. The trading partner name cannot be the hostname.	-	-
`action`	-	-	-
`service`	-	-	-
`idtype / idvalue`	The sender `ID_TYPE`, or identifier type, which uniquely identifies a trading partner and defines how to exchange documents. The sender `ID_VALUE` is the value associated with the identifier type.	-	-
`mode`	Simulates the command without actual execution by either retrieving the affected rows or their count. It lists and shows the count of messages for the selected criteria. It does not resubmit the messages, but just lists the information.	`list`, `count`	-

Table 19-14 (Cont.) Options for b2bresubmit Utility

Option	Description	Domain	Required
`exclresubmit`	Excludes the already resubmitted message.	Default value: `false` Allowed values: `true`/ `false`	-
`exclmsgid`	Excludes a specific message ID.	One or more message IDs (comma-separated)	-
`excldoctype`	-	-	-
`maxcount`	Limits the maximum number of messages to be resubmitted for the selected criteria and enables resubmission in batch. When `maxcount` is specified, the acknowledgment messages in complete state are not considered for resubmission.	Positive integer	-
`oldcpaid` / `newcpaid`	Resubmission for a changed `cpaID` case. This is achieved by providing both `oldcpaID` and `newcpaID`.	-	-
`filepath`	Used in the case of file-based MessageID List, Conversation ID-based resubmit.	-	-
`sourceid`	Works in tandem with – `Dfilepath`	-	-
`protocolmsgid`	-	-	-
`protocolcollaborationid`	-	Comma-separated	-

Example 19-29 Resubmits an Outbound Message with Message ID 12345

```
ant -f ant-b2b-util.xml b2bresubmit -Ddirection=OUTBOUND -Dmsgsource=APPMSG -
Dmsgid=12345\
```

Example 19-30 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"

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dmsgid=39950852 -
Dexclresubmit=true

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=WIREMSG -Dmsgid=39950852 -
Dexclstate=MSG_COMPLETE

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -
Dagreement="Acme_GlobalChips_X12_4010_850_File" -Dexclmsgid=33774513

ant -f ant-b2b-util.xml b2bresubmit -Dmsgsource=APPMSG -Dfromdate="29/05/2010 5:40 AM" -
Dtodate="30/05/2010 7:39 AM" -Dmode=list
```

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 19-31 Specifying a single protocol message ID

```
ant -f ant-b2b-util.xml b2bresubmit -
Dprotocolmessageid="@0AE4A13C13049AF1C940000011B831E8"
```

Example 19-32 Specifying a comma-separated list of protocol message IDs

```
ant -f ant-b2b-util.xml b2bresubmit -
Dprotocolmessageid="@0AE4A13C13049AF1C940000011B831E8,@0AE56A6F13049AF21500000016729990"
```

Example 19-33 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.

Scheduling Trading Partner Downtime

This utility schedules downtime for a trading partner.

```
ant -f ant-b2b-util.xml b2bschedule
```

Table 19-15 lists the options for this utility.

Table 19-15 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)
`fromdate`	The date and time at which to begin the downtime.	Date format must 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 must 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 19-39.	`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 19-34 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 19-35 Schedule a particular channel for downtime

```
ant -f ant-b2b-util.xml b2bschedule
-mode=schedule
-Dtp="MarketInc"
-Dchannelname="MarketInc_ebMS_DC"
-Dschedulename= "Maintenance"
```

Example 19-36 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 19-38, brings the trading partner back to an active state.

```
ant -f ant-b2b-util.xml b2bschedule -mode=schedule -Dtp="MarketInc" -Dschedulename=
"Maintenance"
```

Example 19-37 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 19-38 Unscheduling the scheduled event

```
ant -f ant-b2b-util.xml b2bschedule -Dmode=unschedule -Dschedulename="Maintenance"
```

Example 19-39 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, 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
```

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 prerequisite, 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_days
]
```

Table 19-16 lists the options for this utility.

Table 19-16   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 19-40   Get a List of Certificate Aliases

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Dmode=list
```

Example 19-41   Get the Status of a Certificate

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Dalias="MarketInc_Cert"
```

Example 19-42   Get a List of Certificates Expiring at a Future Date

```
ant -f ant-b2b-util.xml b2bmanagekeystore -Ddays=15
```

Note:

When using Farm Keystore Service (FKS), it is recommended to use the Oracle Enterprise Manager Control console for any kind of keystore manageability.

Updating the Keystore

This utility updates the keystore location and password.

```
ant -f ant-b2b-util.xml b2bkeystoreupdate
```

Table 19-17 lists the mandatory parameters for this utility.

Table 19-17   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

Example:

```
ant -f ant-b2b-util.xml b2bkeystoreupdate -Dkeystorelocation="/tmp/acme.jks" -
Dkeystorepassword="welcome"
```

Errors During Import

This utility resolves a broken pipe error.

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
```

Moving B2B Agreement from a Test to a Production Environment

Moving data from test to production is one of the important aspects in the software production life cycle. There is always a test system to test the required functionality and then configuration/data is moved to the production by changing some artifacts. This process is called a test to production (T2P). The difficulty starts when there is a T2P required in the existing system. Incremental T2P is the process by which the data is moved from test to production one piece at a time. In Oracle B2B, incremental T2P is achieved by migrating an agreement from test to production.

Note:

Agreement T2P promotion is for Oracle B2B only.

Procedures for moving a B2B agreement from a source to a target environment include:

- Export a single agreement as a zip file
- Generate a config plan in XML format&nbsp; using the command line
- Edit the config plan with details for the production environment
- Update the export metadata with the config plan changes
- Import the changes into the production environment

Export a Single Agreement as a ZIP File

You can export a single agreement using the command line or the B2B UI. User can export an agreement using agreement page or deployment page in admin from test system.
To export a single agreement from the test system:

- From the B2B user interface, use the Agreement page or the Administration area Deployment page to select and export the agreement.
- From the command line, use the following command:

  ```
  ant -f ant-b2b-util.xml b2bexport -Dtpanames="Agreement1" -Dactive=true
  ```

Generate a Config Plan in XML Format Using the Command Line

Once you have exported the agreement as a file named `agreementexport.zip`, you can use the `b2bconfig` command to generate the configplan.

Enter the following command on the command line to generate the configplan:

```
ant -f ant-b2b-util.xml b2bconfig -Dmode="generate" -Dexportfile="path/
filename.zip" -Dconfigplan="path/configplan.xml"
```

For example:

```
ant -f ant-b2b-util.xml b2bconfig -Dmode="generate" -Dexportfile="/server/it2p/
Test1_Outbound/OB_MDS_EXPORT_OB.zip" -Dconfigplan="/server/it2p/Test1_Outbound/
OB_MDS_EXPORT_OB.xml" -DaddAllDCParameters=true - DaddAllDocParameters=true -
Ddebug=true
```

Parameter	Description
`b2bgeneratepasswordkey`	Generates the key for password.
`mode`	Set to one of the following: • "`generate`": to generate configplan from agreementexport.zip • "`publish`": to update agreementexport.zip as per configplan updates. • "`publishandimport`": after publish, import generated export file in production system
`exportfile`	The single agreement export file, exported from the deployment page or agreement page in UI or command line agreement export.
`configplan`	The path and filename where the configplan exists or will be generated.
`generatedexportfile`	The new file the is generated after the `publish` command.
`addAlDCParameters`	By default, all channel parameters are not available in the configplan. To extract all parameter set `addAlDCParameters` to `true`. This will generate non-changeable parameters in read only mode.
`addAllDocParameters`	By default, all override doc parameters are not available in the configplan. To extract all parameter set `addAllDocParameters` to `true`. This will generate non-changeable parameters in read only mode.
`debug`	Set to `true` if debug details are required
`passkey`	The key which is stored in CSF against the password.
`password`	The password for metadata.

Edit the Config Plan for the Production Environment

You can edit the configplan metadata to reflect values in the production environment.

Note:

Do not change the attribute and readonly values in the `configplan.xml`, or the structure of the XML. This could cause the metadata to be corrupted. You should only change the `readwrite` value in the `value` element.

All of the parameters in the configplan cannot be updated. Only parameters with `ReadWrite` scope can be updated. Parameters with `readonly` scope are for reference only.

The following are the metadata that you can update:

- Identifiers (except Name):

  ```
  <Identifiers>
      <Parameter type="Generic Identifier" id="id1">
          <value scope="ReadWrite">genId</value>
      </Parameter>
  </Identifiers>
  ```

- Contacts

  ```
  <Contacts>
      <Parameter type="Contact Name" id="ct1">
          <value scope="ReadWrite">name</value>
      </Parameter>
      <Parameter type="Email" id="ct1">
          <value scope="ReadWrite">name@name.com</value>
      </Parameter>
  </Contacts>
  ```

- Trading Partners

  ```
  <TradingpartnerParameters>
      <Parameter name="TPParam1" id="tpp1">
          <value scope="ReadWrite">"TPParam1Value"</value>
      </Parameter>
      <Parameter name="TPParam2" id="tpp2">
          <value scope="ReadWrite">TPParam2Value</value>
      </Parameter>
  </TradingpartnerParameters>
  ```

- Channels

  ```
  <Channels>
      <Channel name="channel1" protocol="Generic File" version="1.0"
              internal="true" listening="false" id="ch1">
      <Parameters category="Transport">
          <Parameter name="file-param-folder" >
  ```

```
                <value scope="ReadWrite">/tmp12/</value>
            </Parameter>
        </Parameters>
        </Channel>
</Channels>
```

Parameter	Description
File	`file-param-folder`
FTP	`ftp-param-host, ftp-param-folder, ftp-param-user, ftp-param-password`
SFTP	`sftp-param-host, sftp-param-port, sftp-param-folder, sftp-param-user, sftp-param-password`
Email	`email-param-host, email-param-user, email-param-email-id, email-param-password`
HTTP	`http-param-url, http-param-use_proxy, http-param-additional_headers, http-param-password`
AS2	`as2-param-Receipt-Delivery-Option`
TCP	`tcp-param-host, tcp-param-port, tcp-param-PermanentConnectionType, tcp-param-timeout`
JMS	`jms-param-queue_name, jms-param-jndi_connection_factory_location, jms-param-DestinationProviderProperties, jms-param-is_topic, jms-param-password, jms-param-Flow_Trace_EM_URL`
AQ	`aq-param-datasource, aq-param-recipient, aq-param-queue_name, aq-param-password`
WS-HTTP	`wshttp-param-url, wshttp-param-service, wshttp-param-port, wshttp-param-action`
MFT	`mft-param-source, mft-param-url, mft-param-user, mft-param-password`

Update the Export Metadata with the Config Plan Changes

You must update the `agreementexport.zip` file with the changes to the configplan.

Enter the following command on the command line to apply the configplan.xml to the export file:

```
ant -f ant-b2b-util.xml b2bconfig -Dmode="publishandimport" -Dexportfile="path/
filename.zip" -Dconfigplan="path/configplan.xml" -Dgeneratedexportfile="path/
filename.zip" -Ddebug=true
```

For example:

```
ant -f ant-b2b-util.xml b2bconfig -Dmode="publishandimport" -Dexportfile="/
server/it2p/MDS_EXPORT.zip" -Dconfigplan="/server/it2p/MDS_EXPORT.xml" –
Dgeneratedexportfile="/server/it2p/MDS_EXPORT_new.zip” -Ddebug=true
```

Parameter	Description
`b2bgeneratepasswordkey`	Generates the key for password.

Parameter	Description
`mode`	Set to one of the following: • "`generate`": to generate configplan from agreementexport.zip • "`publish`": to update agreementexport.zip as per configplan updates. • "`publishandimport`": after publish, import generated export file in production system
`exportfile`	The single agreement export file, exported from the deployment page or agreement page in UI or command line agreement export.
`configplan`	The path and filename where the configplan exists or will be generated.
`generatedexportfile`	The new file the is generated after the `publish` command.
`addAlDCParameters`	By default, all channel parameters are not available in the configplan. To extract all parameter set `addAlDCParameters` to `true`. This will generate non-changeable parameters in read only mode.
`addAllDocParameters`	By default, all override doc parameters are not available in the configplan. To extract all parameter set `addAllDocParameters` to `true`. This will generate non-changeable parameters in read only mode.
`debug`	Set to `true` if debug details are required
`passkey`	The key which is stored in CSF against the password.
`password`	The password for metadata.

Import the Changes into the Production Environment

You can import the B2B configuration zip file using the B2B UI or B2B command line tool.

To import the configuration zip file, do one of the following:

- Use the B2B console admin Import/Export page to import the configuration zip file.
- Use the following command to import the data from thetmp/export.zip location:

  ```
  ant -f ant-b2b-util.xml b2bimport -Dlocalfile=true -Dexportfile="/tmp/
  export.zip"
  ```

20

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 includes the following sections:

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 Using the Outbound Web Service for more information.
- **Query API** helps to retrieve the configured details in Oracle B2B. See 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 runtime 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.

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.

The following SOAP headers are used to pass information required to find a trading partner agreement in the case of B2B web services:

```
<To xmlns="http://com.oracle.b2b/soap">GlobalChips</To>
<DocVersion xmlns="http://com.oracle.b2b/soap">4010</DocVersion>
<From xmlns="http://com.oracle.b2b/soap">Acme</From>
<DocType xmlns="http://com.oracle.b2b/soap">850</DocType>
```

Table 20-1, Table 20-2, and Table 20-3 describe the Outbound Web Service request, response and fault notification message parameters.

Table 20-1 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*
documentTypeName	String Example: `850`	Document type name	Yes*
action	String Example: `PurchaseOrder`	ebMS action name	Yes *
service	String Example: `OrderProcessing`	ebMS service name	Yes *
serviceType	String Example: `string` Default: `string`	ebMS service type	Yes *
messageId	String Example: `123456789`	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

Table 20-1 (Cont.) Outbound Web Service Request Parameters

Header	Data Type	Description	Required
messageType	String Examples: INVALID (0) REQ (1) RESP (2) ACK (3) IN_BAND_EXCEPTION (4) OUT_OF_BAND_EXCEPTION (5 STATUS_REQ (6) STATUS_RESP (7) PULL (8) FUNCTIONAL_ACK (9) BATCH_REQ (10) OTHER (99) PING (11) PONG (12) TA1(13)	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

*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 20-2 Outbound Web Service Response Parameters

Header	Data Type	Description	Required
isTransmitted	Boolean	If `true`, the message was successfully transmitted to Oracle B2B; otherwise `false`.	Yes

Table 20-3 Outbound Web Service Fault Message Parameters

Header	Data Type	Description
ExceptionMessage	String	If a fault is found, the Exception Stack Trace is transmitted.

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 Is Trading Partner Agreement Setup Parameters for parameter details.
- `Get Trading Partner Agreement Information` returns agreement details. See Get Trading Partner Agreement Information Parameters for parameter details.

Is Trading Partner Agreement Setup Parameters

Table 20-4, Table 20-5, and Table 20-6 describe the Trading Partner Agreement Setup message parameters.

Table 20-4 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

Table 20-5 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 20-6 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.

Get Trading Partner Agreement Information Parameters

Table 20-7, Table 20-8, and Table 20-9 describe the Trading Partner Agreement Information message parameters.

Table 20-7 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 20-8 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	Document Protocol 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
DCName	String	Delivery Channel associated with the Trading Partner Agreement	Yes
ExchangeProtocol	String	Type of transport protocol to exchange messages with trading partner	Yes
SyncMode	True/False	True is Request Response; false is one-way	Optional for FTP and SMTP; required for HTTP/WS.

Table 20-9 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.

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 life cycle of Web services and their policies.

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.

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 "Which OWSM Policies Are Supported for Java EE Web Services?" 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.

Example 20-1 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>
```

21 Enabling Web-Service-Based Message Exchange

This chapter describes how to enable Web-service-based message (typically SOAP-based) exchange between trading partners in Oracle B2B.

The chapter contains the following sections:

Introduction to Web-Service-Based Message Exchange

Oracle B2B allows you to exchange Web service (SOAP) based messages between trading partners. You can exchange messages in both inbound and outbound direction. However, currently, this support is limited to SOAP 1.1 messages over HTTP only.

The Web service feature not only enables partners to receive or send messages, but also it is layered as a protocol implementation and supports other general features such as reporting, tracking, and auditing.

Many enterprises are increasingly having a requirement to integrate their trading partners file transfer and/or message exchange using Web service (in addition to B2B-specific protocols.)

Oracle B2B support for Web services is based on the following specifications, however, implementation may differ at times:

- SOAP 1.1
- WS Addressing 1.0
- Attachment

Exchanging SOAP-Based Service Messages with Custom WSDL File

The support for SOAP-based messages is available for both inbound and outbound directions.

You need to create or upload a Web Service Definition Language (WSDL) file that you can customize according to your requirement.

Exchanging Outbound SOAP-Based Messages

To enable exchange of outbound SOAP-based messages, you need to perform the following tasks:

- Uploading the WSDL
- Creating a document
- Adding the document to as a part of partner documents
- Creating a Trading Partner Delivery Channel
- Creating and deploying an agreement

Uploading the WSDL

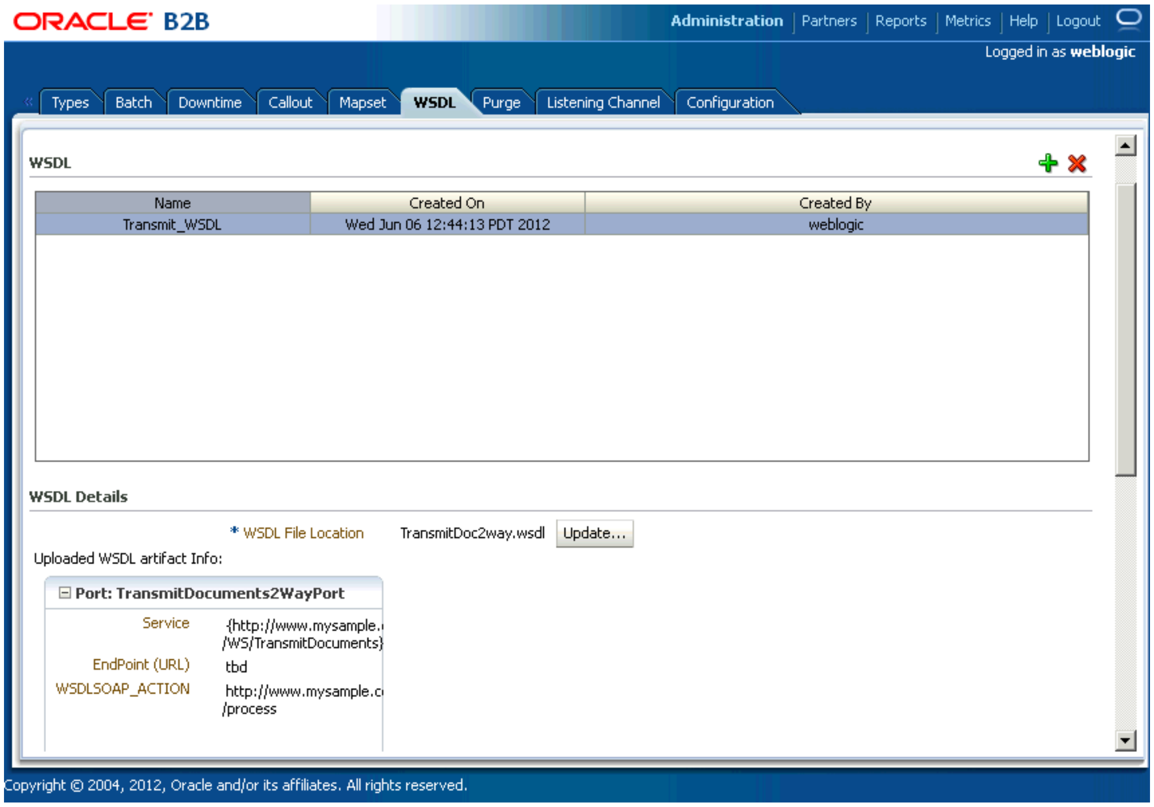
As the first task, you need to upload the WSDL file that is required to register a Web service to exchange messages. You can upload the WSDL in the either of the following ways:

- Inline WSDL - a normal WSDL file where the XSD information is defined within the WSDL itself
- A ZIP file containing a WSDL file and an XSD file
- A ZIP file containing multiple WSDL and XSD files

To upload a WSDL:

1. Log on to the Oracle B2B console (http://<*hostname*>:<*port*>/b2bconsole), where *hostname* is the name of the computer where Oracle SOA Suite is hosted and *port* is typically `8001` (in the case of a non-SSL connection).
2. Click the **Administration** link and then click the **WSDL** tab.
3. Click the **+** button (Add WSDL). An entry for the new WSDL gets created in the WSDL table.
4. Specify a name for the WSDL such as Transmit_WSDL. In case of multiple WSDLs that have been zipped and uploaded, the start WSDL or the root WSDL has to be selected.
5. In the WSDL section, click the Browse button to select the WSDL to be uploaded. For this example, the WSDL selected is TransmitDoc2way.wsdl. You can see the details of the WSDL is listed under the Uploaded WSDL artifact Info section as shown in Figure 21-1.

Figure 21-1 WSDL Artifacts


ORACLE B2B
Administration | Partners | Reports | Metrics | Help | Logout
Logged in as weblogic
Types | Batch | Downtime | Callout | Mapset | WSDL | Purge | Listening Channel | Configuration

WSDL

Name	Created On	Created By
Transmit_WSDL	Wed Jun 06 12:44:13 PDT 2012	weblogic

WSDL Details

* WSDL File Location TransmitDoc2way.wsdl Update...

Uploaded WSDL artifact Info:

Port: TransmitDocuments2WayPort

Service	{http://www.mysample. /WS/TransmitDocuments}
EndPoint (URL)	tbd
WSDLSOAP_ACTION	http://www.mysample.c /process

Copyright © 2004, 2012, Oracle and/or its affiliates. All rights reserved.

6. Click **Save** and then click **OK** in the confirmation dialog.

> **Note:**
>
> After the uploaded WSDL file is referenced by any other metadata such as channels and they are deployed as a part of agreements, any update to the WSDL is disabled. After these references are inactivated, only then you can update the uploaded WSDL.

Creating a document

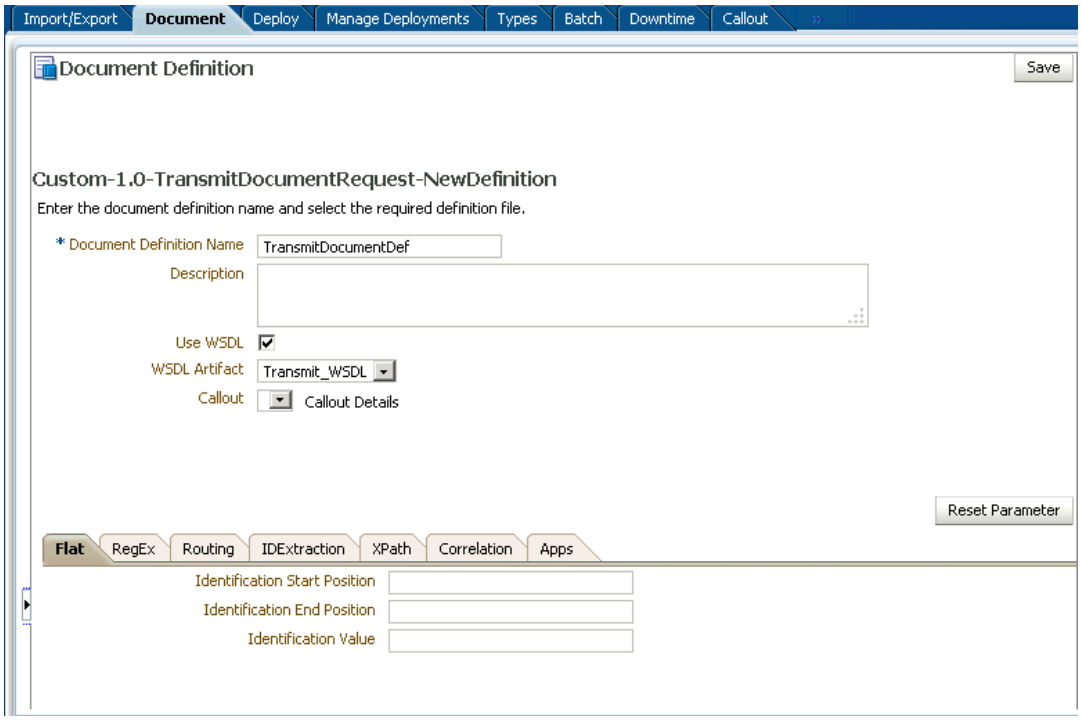
The next task is to create a document for the outbound flow. Please note that currently, the only supported document type is custom.

To create a custom document:

1. Create document definitions as specified in Creating Document Definitions. Specify:
   - Document Version - 1.0
   - Document Type - TransmitDocumentRequest
   - Document Definition - TransmitDocumentDef
2. Select **Use WSDL**.
3. Select the relevant WSDL Artifact, which in this case is **Transmit_WSDL**.

4. Select the required WSDL Message, which in this case is the **TransmitDocumentsRequestMessage** as shown in Figure 21-2.

Figure 21-2 Creating Documents



Document Definition

Custom-1.0-TransmitDocumentRequest-NewDefinition

Enter the document definition name and select the required definition file.

* Document Definition Name: TransmitDocumentDef

Description:

Use WSDL: ✓

WSDL Artifact: Transmit_WSDL

Callout: Callout Details

Identification Start Position:

Identification End Position:

Identification Value:

5. Select any required callout from the **Callout** list.
6. Click **Save**.

Adding the document to as a part of partner documents

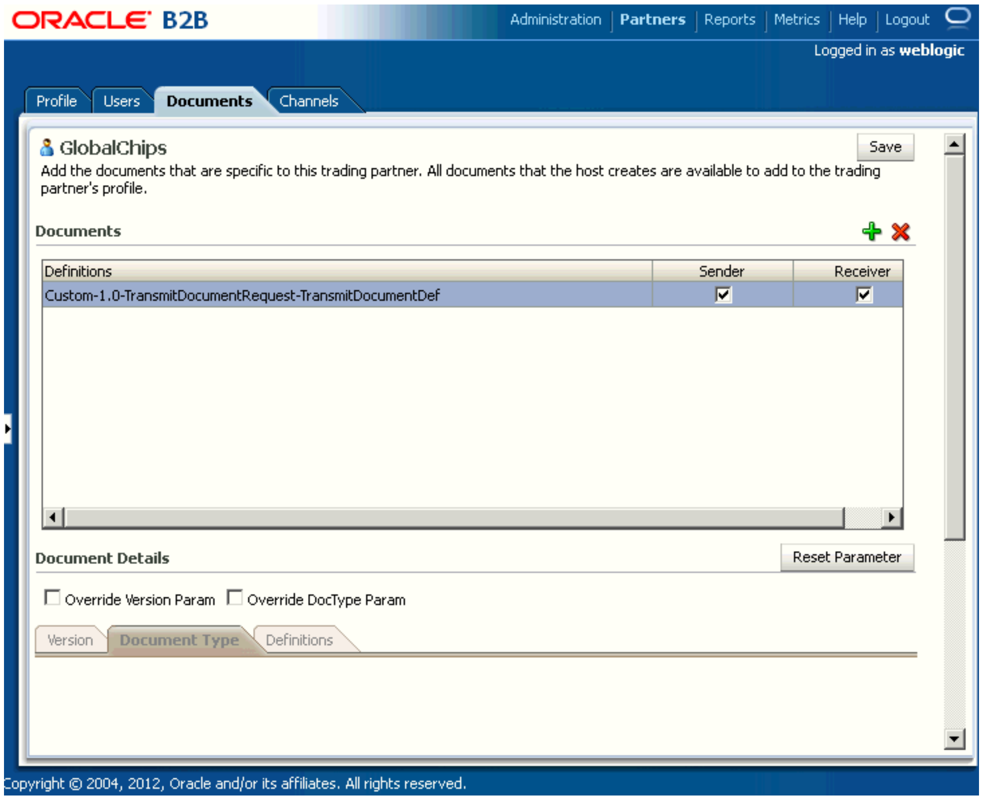
The third task is to add the newly created custom document as a part of the Partner documents.

To add the document:

1. Click the **Partners** link and then click the **Documents** tab.
2. Click to select the remote trading partner name (GlobalChips) under the partner section.
3. Under the Documents section on the right-hand pane, click the **+** button (Add Document Definition) to display the Select Document Definition window.
4. Navigate to the required document definition and click **Add** to add the document definition.
5. Click **Save** in the Documents tab.

   The document gets added to the list of partner documents as shown in Figure 21-3.

Figure 21-3 Adding a Document


ORACLE B2B
Administration | Partners | Reports | Metrics | Help | Logout
Logged in as weblogic
Profile
Users
Documents
Channels
Save
Reset Parameter
Version
Document Type
Definitions
Copyright © 2004, 2012, Oracle and/or its affiliates. All rights reserved.

GlobalChips

Add the documents that are specific to this trading partner. All documents that the host creates are available to add to the trading partner's profile.

Documents

Definitions	Sender	Receiver
Custom-1.0-TransmitDocumentRequest-TransmitDocumentDef	✓	✓

Document Details

☐ Override Version Param ☐ Override DocType Param

Creating a Trading Partner Delivery Channel

After adding the documents, you need to create a trading partner delivery channel to send messages.

To add a trading partner delivery channel:

1. Click to select the trading partner (GlobalChips) under **Partner** on the left.
2. Click the **Channels** tab.
3. Click the **+** (Add Channel to Trading Partner) button.
4. In the Channel section, specify a name for the channel (such as GlobalChips_Channel).
5. Select **Generic WS-1.0** from the Protocol list.
6. In the Channel details section, in the Web Service Parameters tab:
   - Select **Transmit_WSDL**, the WSDL that you uploaded from WSDL Artifact from the WSDL Artifact list.
   - Select the available service (**TransmitsDocumentService** in this case).

Note:

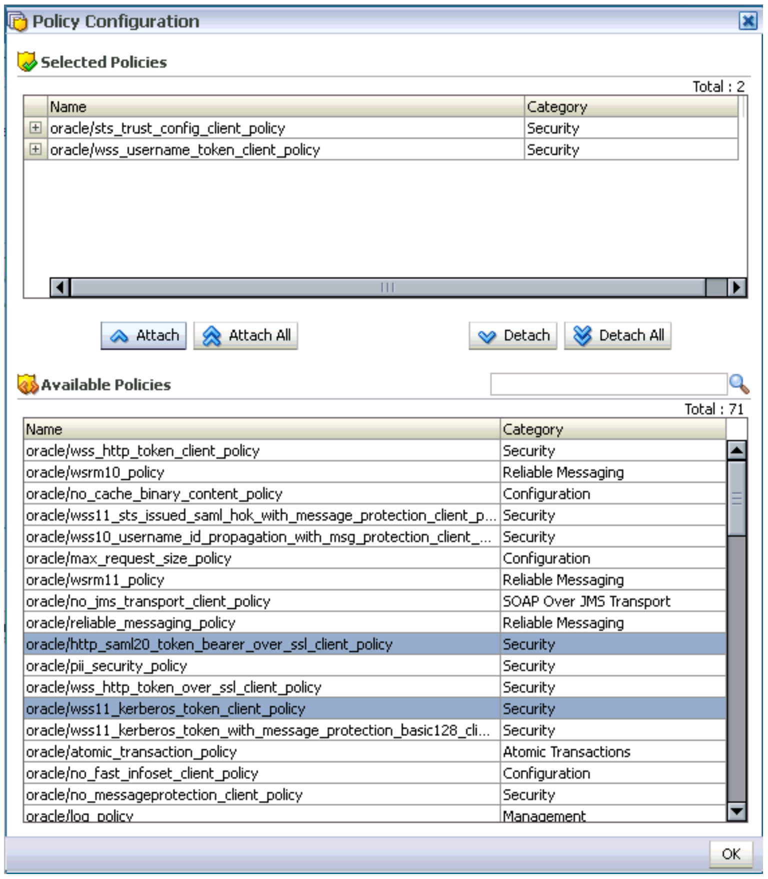
Oracle B2B allows you to specify services, ports, and actions manually. This is useful when you want to specify additional service, port, or action beyond and over what is available as part of the selected WSDL. You can also provide string values for the preceding parameters manually. In the case of manually providing the service name, it should be in the following format, else there would be a validation error:

{*namespace*}*ServiceName*

The namespace *must* match the target namespace of the WSDL, else, the Security Policy Configuration will not be applied.

- Select the available port (**TransmitDocuments2WayPort**).
- Select the SOAP action (available from the WSDL).
- Enter the URL where the server is listening as the Endpoint (URL), which is the URL of the GlobalChips trading partner.
- Enter any additional HTTP headers, if required.
- Select **Omit WS Addressing Headers** if you want to discard WS Addressing headers as a part of the message to be exchanged.
- Select **Omit Oracle Default SOAP Headers** if you do not want Oracle B2B Web services outbound channel to send the default SOAP headers such as From, To, Doctype, and DocRevision as a part of the message.
- Click the **Policy Configuration Click here** link to display the Policy Configuration window. The policy Configuration window lists all the available Web services security policies that you can attach to the channel locally. Attaching a policy locally to a channel provides granular control over the security of the channel as opposed to setting global policies at the Oracle Fusion Middleware Enterprise Manager Control console level.
- In the Policy Configuration window, from the list of available Web services policies under Available Policies, select the policy that you want to attach to the channel. You can select multiple policies by pressing the Control key and clicking the policy names.
- Click **Attach** to attach the selected policy or policies to the channel. You can click the **Attach All** button to attach all the available policies to the channel and click **OK** as shown in Figure 21-4.

Figure 21-4 Attaching Web Services Policies



Selected Policies

Total : 2

Name	Category
oracle/sts_trust_config_client_policy	Security
oracle/wss_username_token_client_policy	Security

Available Policies

Total : 71

Name	Category
oracle/wss_http_token_client_policy	Security
oracle/wsrm10_policy	Reliable Messaging
oracle/no_cache_binary_content_policy	Configuration
oracle/wss11_sts_issued_saml_hok_with_message_protection_client_p...	Security
oracle/wss10_username_id_propagation_with_msg_protection_client_...	Security
oracle/max_request_size_policy	Configuration
oracle/wsrm11_policy	Reliable Messaging
oracle/no_jms_transport_client_policy	SOAP Over JMS Transport
oracle/reliable_messaging_policy	Reliable Messaging
oracle/http_saml20_token_bearer_over_ssl_client_policy	Security
oracle/pii_security_policy	Security
oracle/wss_http_token_over_ssl_client_policy	Security
oracle/wss11_kerberos_token_client_policy	Security
oracle/wss11_kerberos_token_with_message_protection_basic128_cli...	Security
oracle/atomic_transaction_policy	Atomic Transactions
oracle/no_fast_infoset_client_policy	Configuration
oracle/no_messageprotection_client_policy	Security
oracle/log_policy	Management

Note:

Some policies are contradictory to each other. If you try to attach such contradictory policies together to an endpoint, you will get an error.

For more information, see Overview of Global Policy Attachments Using Policy Sets in *Understanding Oracle Web Services Manager*.

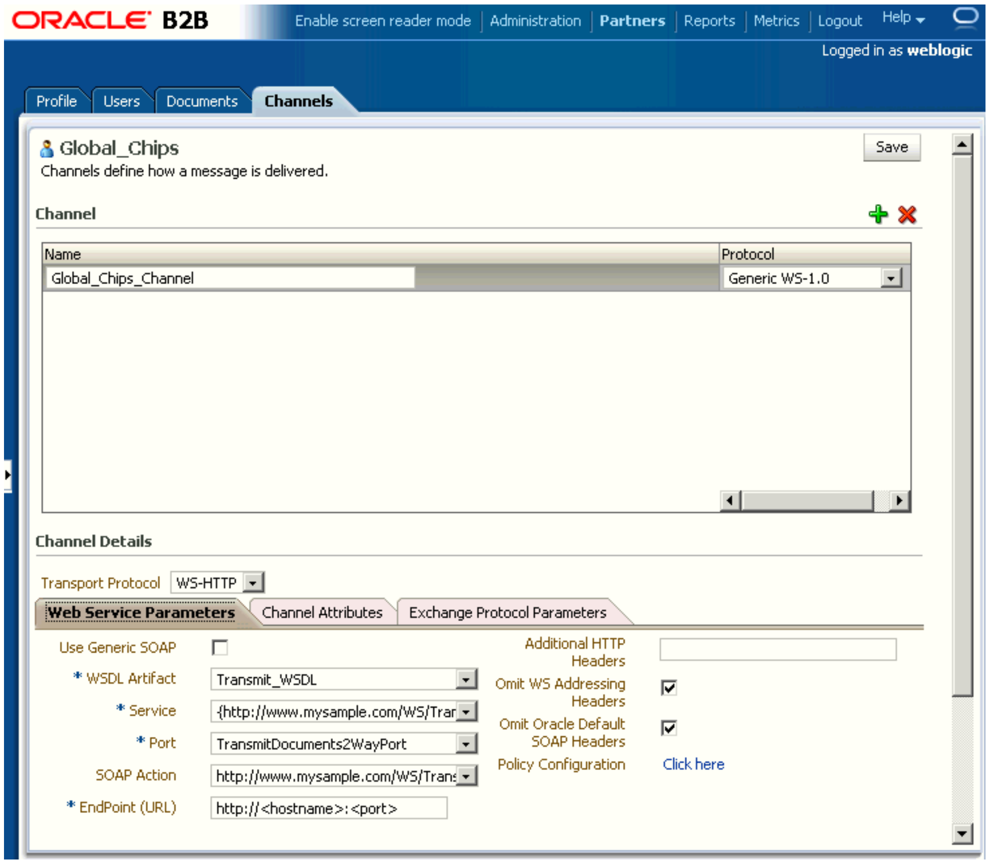
Note:

You can also set the parameters, such as **Additional SOAP Headers** and **Payload XPath** in the **Exchange Protocol Parameters** tab.

Select the **Additional SOAP Headers** option to specify any additional SOAP headers that you want to pass along with the message. The **Payload XPath** option helps to select the payload based on the XPath; rest of the SOAP body is ignored.

7. Click **Save** as shown in Figure 21-5.

Figure 21-5 Web Service Channel Details



Global_Chips
Channels define how a message is delivered.

Channel

Name	Protocol
Global_Chips_Channel	Generic WS-1.0

Channel Details

Transport Protocol WS-HTTP

Web Service Parameters

Use Generic SOAP	☐	Additional HTTP Headers	
* WSDL Artifact	Transmit_WSDL	Omit WS Addressing Headers	✓
* Service	{http://www.mysample.com/WS/Trar	Omit Oracle Default SOAP Headers	✓
* Port	TransmitDocuments2WayPort	Policy Configuration	Click here
SOAP Action	http://www.mysample.com/WS/Trans		
* EndPoint (URL)	http://<hostname>:<port>		

Note:

If you do not want to upload and use a custom WSDL, you can select the **Use Generic SOAP** check box. This should enable you to send any XML document over SOAP and you do not need to associate any WSDL in Channel/Document page. Oracle B2B provides a preseeded generic WSDL that is used when you select **Use Generic SOAP**.

Creating and deploying an agreement

The last task is to create and deploy a trading partner agreement.

Refer to Creating and Deploying Trading Partner Agreements to learn more about creating and deploying an agreement.

Exchanging Inbound SOAP-Based Messages

To exchange inbound SOAP-based messages, you need to perform the following tasks:

- Uploading the WSDL
- Creating a document
- Adding the document to as a part of partner documents

- Creating a Trading Partner Delivery Channel

Note:

After you create a listening channel and enable it, you cannot edit the channel to use a different WSDL. You need to delete the channel and create a new one.

Uploading the WSDL

This task is the same as Uploading the WSDL of the outbound case.

Creating a document for the inbound flow

This task is the same as Creating a document of the outbound case.

Adding the document to as a part of partner documents

This task is the same as Adding the document to as a part of partner documents of the outbound case.

Creating a listening channel

The next task is to create a listening channel to receive the incoming messages.

To create a listening channel:

1. Click the **Administration** link and then click the **Listening Channel** tab.
2. Click the **+** (Add Channel to Trading Partner) button.
3. In the Listening Channel section, specify a name for the channel (such as Acme_ListeningChannel).
4. Select **Generic WS-1.0** from the Protocol list.
5. In the Channel details section, in the Web Service Parameters tab:
   - Select **Transmit_WSDL**, the WSDL that you uploaded from WSDL Artifact from the WSDL Artifact list.
   - Select the available service (**TransmitsDocumentService** in this case).
   - Select the available port (**TransmitDocuments2WayPort**).
   - Enter the URL where the server is listening as the Endpoint (URL), which is the URL of the GlobalChips trading partner.

     Note that the Endpoint (URL) gets populated automatically with the URL of the partner where the server is listening.
   - Click the **Policy Configuration Click here** link to display the Policy Configuration window. The policy Configuration window lists all the available Web services security policies that you can attach to the channel locally. Attaching a policy locally to a channel provides granular control over the security of the channel as opposed to setting global policies at the Oracle Fusion Middleware Enterprise Manager Control console level.

- In the Policy Configuration window, from the list of available Web services policies under Available Policies, select the policy that you want to attach to the channel. You can select multiple policies by pressing the Control key and clicking the policy names.
- Click **Attach** to attach the selected policy or policies to the channel. You can click the **Attach All** button to attach all the available policies to the channel and click **OK**.

6. Click the **Channel Attributes** tab and ensure that **Enable Channel** is selected.
7. Click **Save**.

Note:

To ensure that the that the Web services have been registered for listening messages, access the following URL, log on by providing the user name and password, and check if the Web service is listed in the list of registered Web services:

http:<*host*>:<*port*>/b2b/services

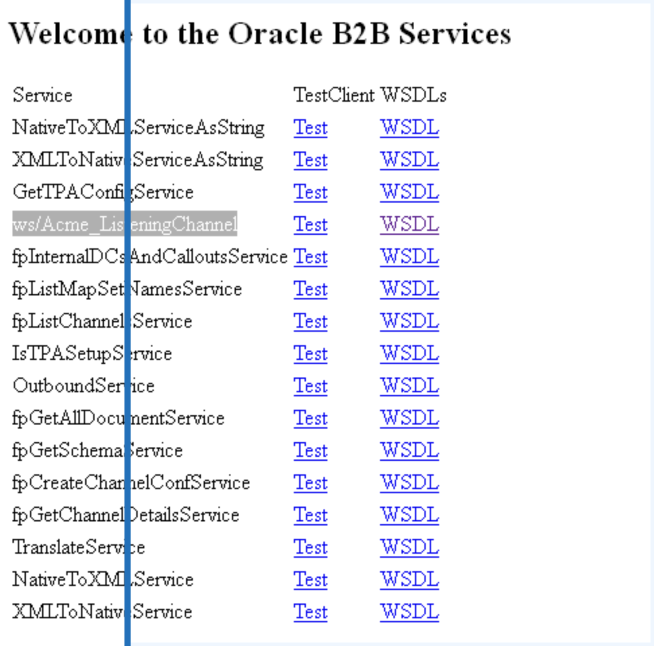
You need to create users (along with their passwords) in the Oracle Weblogic Server console. Access the following link to know more about creating users in the Oracle Weblogic Server console:

`http://docs.oracle.com/cd/E23943_01/apirefs.1111/e13952/taskhelp/security/ManageUsersAndGroups.html`

The Web service is listed in the following format:

ws/*Listening_Channel_Name*

You can download the WSDL from the specific service by clicking the respective WSDL link as shown in the following graphic.



Welcome to the Oracle B2B Services

Service	TestClient	WSDLs
NativeToXMLServiceAsString	Test	WSDL
XMLToNativeServiceAsString	Test	WSDL
GetTPAConfigService	Test	WSDL
ws/Acme_ListeningChannel	Test	WSDL
fpInternalDCsAndCalloutsService	Test	WSDL
fpListMapSetNamesService	Test	WSDL
fpListChannelsService	Test	WSDL
IsTPASetupService	Test	WSDL
OutboundService	Test	WSDL
fpGetAllDocumentService	Test	WSDL
fpGetSchemaService	Test	WSDL
fpCreateChannelConfService	Test	WSDL
fpGetChannelDetailsService	Test	WSDL
TranslateService	Test	WSDL
NativeToXMLService	Test	WSDL
XMLToNativeService	Test	WSDL

Note:

After you create a listening channel and enable it, you cannot edit the channel to use a different WSDL. You need to delete the channel and create a new one.

Configuring wsa.action

There are various way to configure the `wsa.action`.

- `wsa.action` in the case of fabric or `WSA_ACTION` in the case of JMS can be sent from the back end and the same will be used as `wsa.action` while sending the message.
- The SOAP action, `eventName=ACTION:test`, can also be used as `wsa.action` in case `wsa.action` is not provided from the back end.
- The action configured as a part of the delivery channel will be used as `wsa.action` in case `wsa.action` is not specified from the back end application.

Sending Custom SOAP Headers

Oracle B2B allows you to send custom SOAP headers as a part of your outbound Web-service-based messages.

In the case of an outbound JMS channel, the sender trading partner can send multi-level custom SOAP headers as the following sample:

```
<CustomSOAPHeader xmlns="http://schemas.xmlsoap.org/soap/envelope/">
   <hello xmlns="http://xmlns.oracle1.com/soa1/generic/soap">
       <name xmlns="http://MY_NAME_SPACE">
         <firstname>John</firstname>
         <lastname>Doe</lastname>
       </name>
    </hello>
</CustomSOAPHeader>
```

In the case of an outbound fabric, the sender can send multi-level custom headers as the following sample:

```
<bpelx:inputProperty name="b2b.customSOAPHeaders" expression="'<CustomSOAPHeader
xmlns="http://schemas.xmlsoap.org/soap/envelope/">
   <hello xmlns="http://xmlns.oracle1.com/soa1/generic/soap">
       <name xmlns="http://MY_NAME_SPACE">
         <firstname>John</firstname>
         <lastname>Doe</lastname>
       </name>
    </hello>
</CustomSOAPHeader>
```

Attaching Policies to Web Services Using Enterprise Management

This section discusses how to attach policy sets to Web services by using the Oracle Fusion Middleware Enterprise Management Control console.

To create the UserName token policy and required credentials, you need to perform the following tasks:

- Creating Default Credentials Map and Key
- Creating and attaching outbound policy sets
- Creating and attaching inbound policy sets

Creating Default Credentials Map and Key

The first task is to create credentials.

To create credentials:

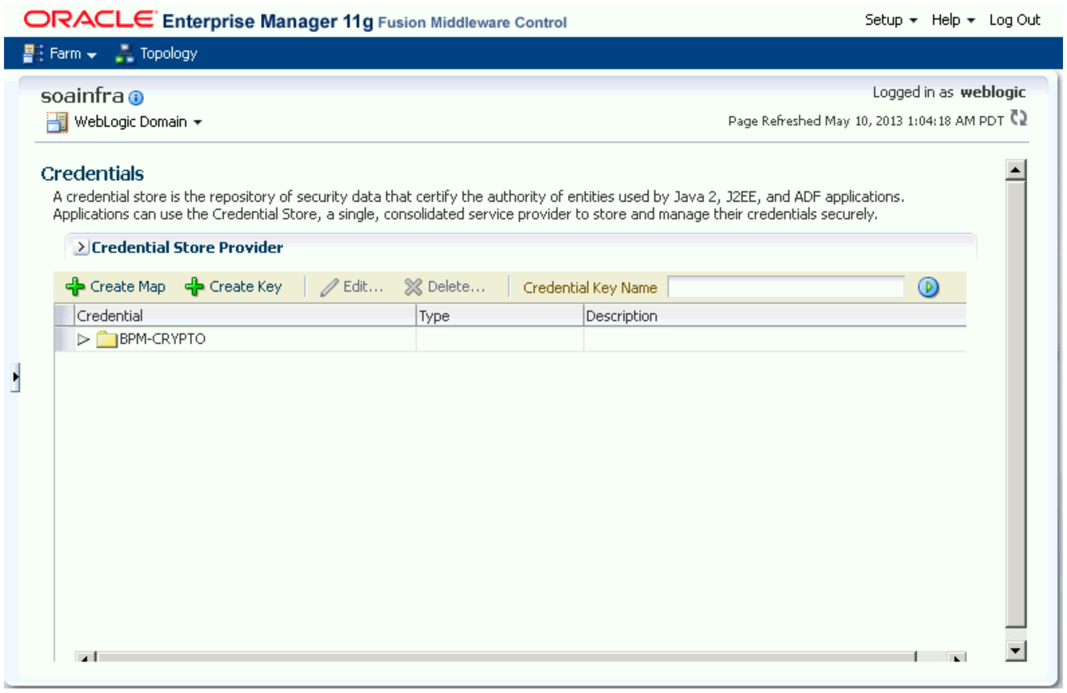
1. Log on to Oracle Fusion Middleware Enterprise Management Control console by accessing the following URL:

   http://<*hostname*>:<*port*>/em

   Where:

   - *hostname*: The name of the computer running the Oracle SOA Suite
   - *port*: The port number of the Admin server; typically `7001`.
2. Expand the domain name, such as **soainfra**, under Weblogic Domain in the Target Navigation pane.
3. Right-click **soainfra**, and select **Security** > **Credentials** to display the Credentials page as displayed in Figure 21-6.

Figure 21-6 Accessing the Credentials Page

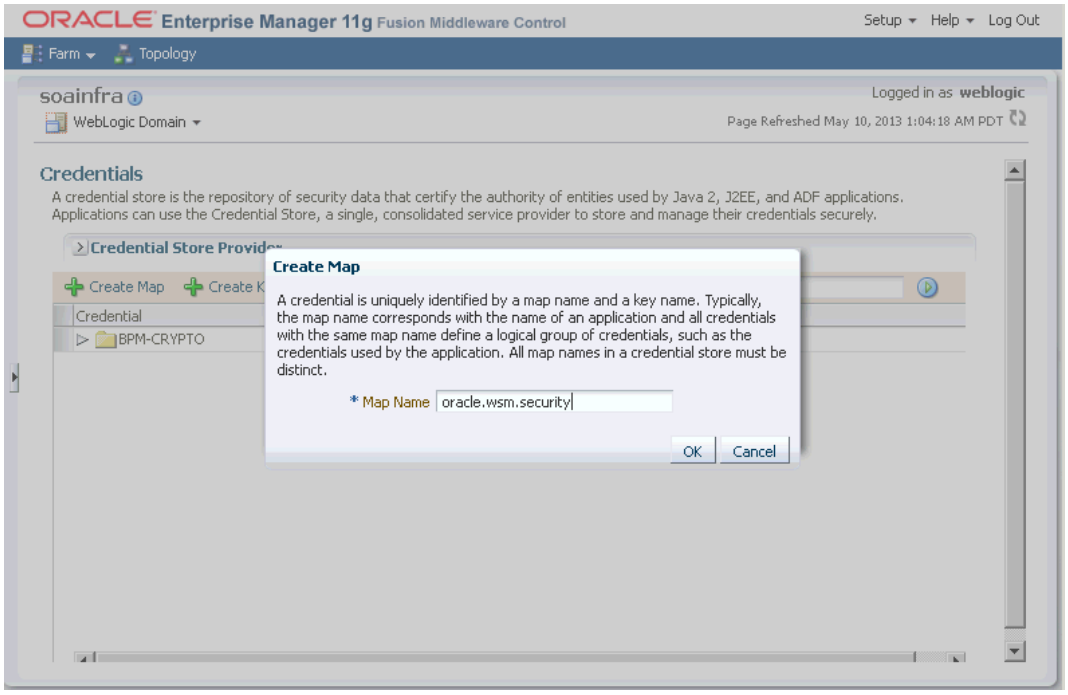


Credentials

A credential store is the repository of security data that certify the authority of entities used by Java 2, J2EE, and ADF applications. Applications can use the Credential Store, a single, consolidated service provider to store and manage their credentials securely.

4. In the Credentials page, click **Create Map**.
5. In the Create Map dialog box, enter `oracle.wsm.security` as the map name as displayed in Figure 21-7 and click **OK**.

Figure 21-7 Create Map Dialog Box



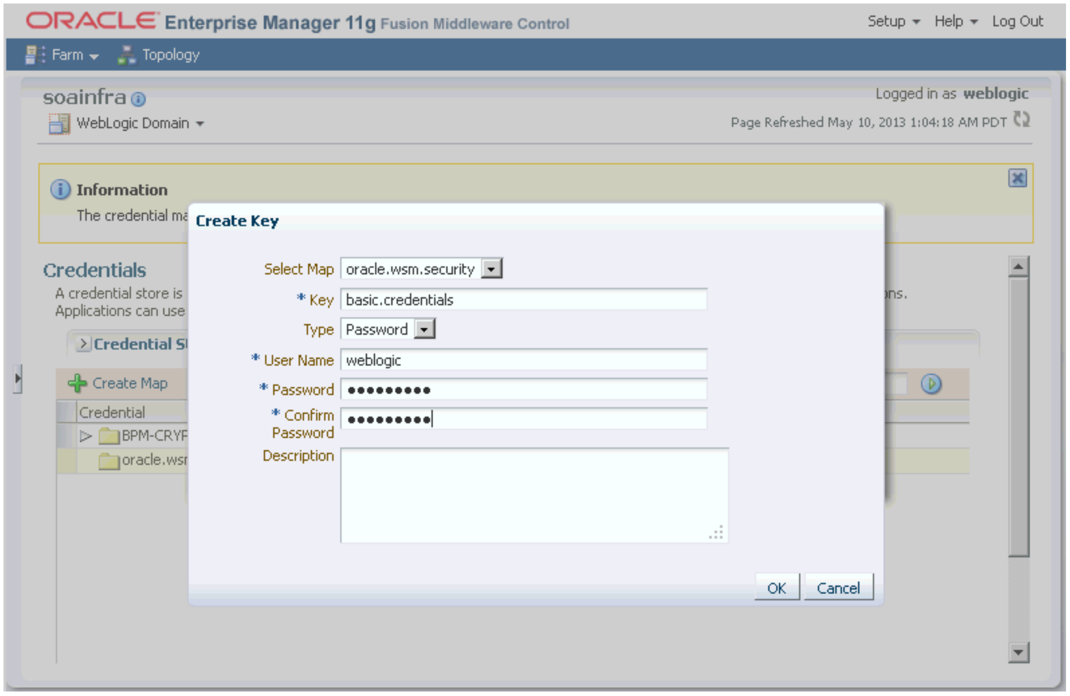
A credential is uniquely identified by a map name and a key name. Typically, the map name corresponds with the name of an application and all credentials with the same map name define a logical group of credentials, such as the credentials used by the application. All map names in a credential store must be distinct.

* Map Name: oracle.wsm.security

6. Select **oracle.wsm.security** (the map that you created) and click **Create Key** to display the Create Key dialog box.

7. In the Create Key dialog box, do the following as displayed in Figure 21-8:
   a. Enter the key name as basic.credentials in the Key field.
   b. Select **Password** as Type.
   c. Enter the Admin user name, which in this case is `weblogic`.
   d. Enter a password in the Password field.
   e. Confirm the password that you have entered in the last step.
   f. Click **OK**.

Figure 21-8 Create Key Dialog Box



Select Map: oracle.wsm.security
* Key: basic.credentials
Type: Password
* User Name: weblogic
* Password: ●●●●●●●●●
* Confirm Password: ●●●●●●●●●
Description:

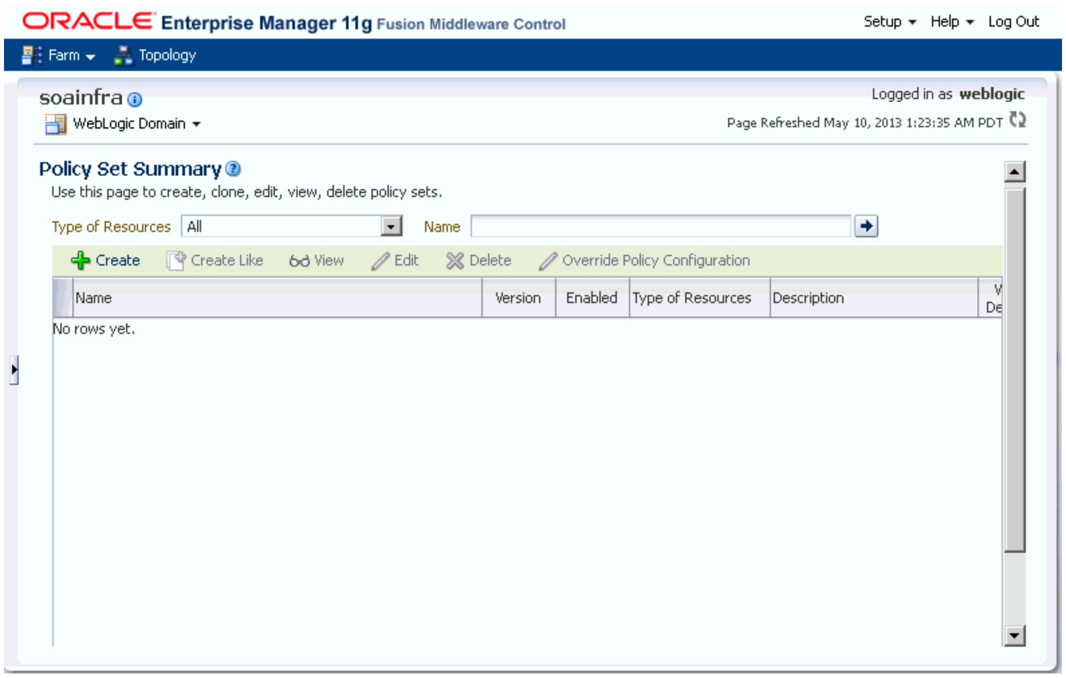
Creating and attaching outbound policy sets

The next task is to create and attach the outbound policy sets to the Web service.

To create an outbound policy set and attach it to Web services:

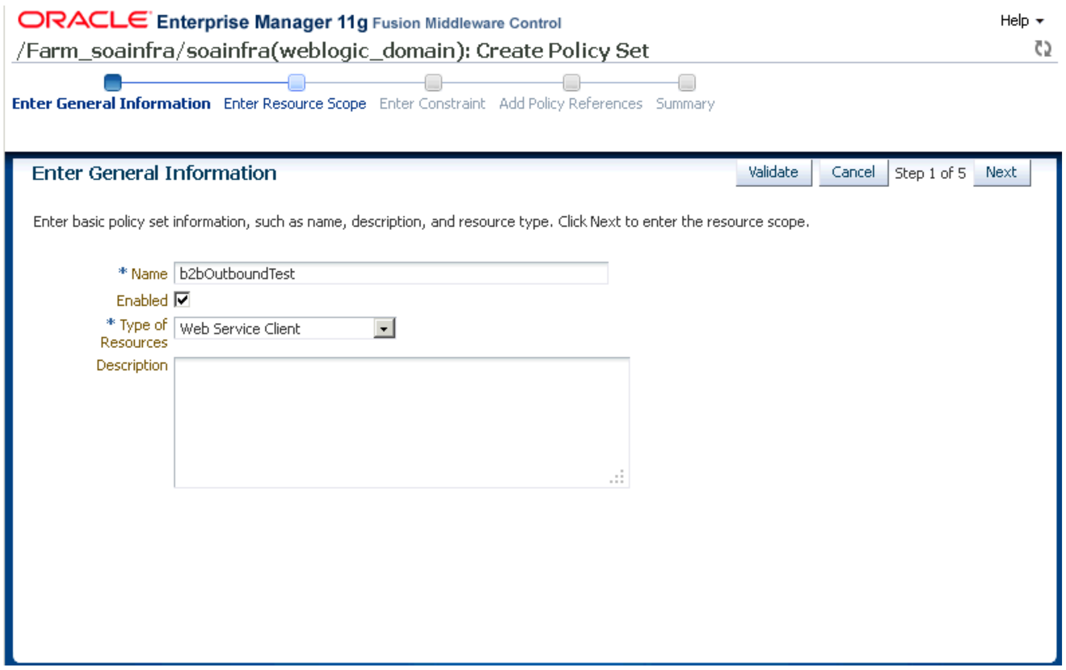
1. Click the **Weblogic Domain** list on the domain name (soainfra) page and select **Web Services** > **Policy Sets** to display the Policy Set Summary page as displayed in Figure 21-9.

Figure 21-9 Policy Set Summary Page



2. In the Policy Set Summary page, click **Create** to display the Create Policy Set wizard.
3. Enter the name as `b2bOutboundTest`, select **Enabled**, select **Web Service Client** from the Type of Resources list, and click **Next** as displayed in Figure 21-10.

Figure 21-10 Create Policy Set - Enter General Information Page


ORACLE Enterprise Manager 11g Fusion Middleware Control
Help
/Farm_soainfra/soainfra(weblogic_domain): Create Policy Set
Enter General Information
Enter Resource Scope
Enter Constraint
Add Policy References
Summary
Enter General Information
Validate
Cancel
Step 1 of 5
Next

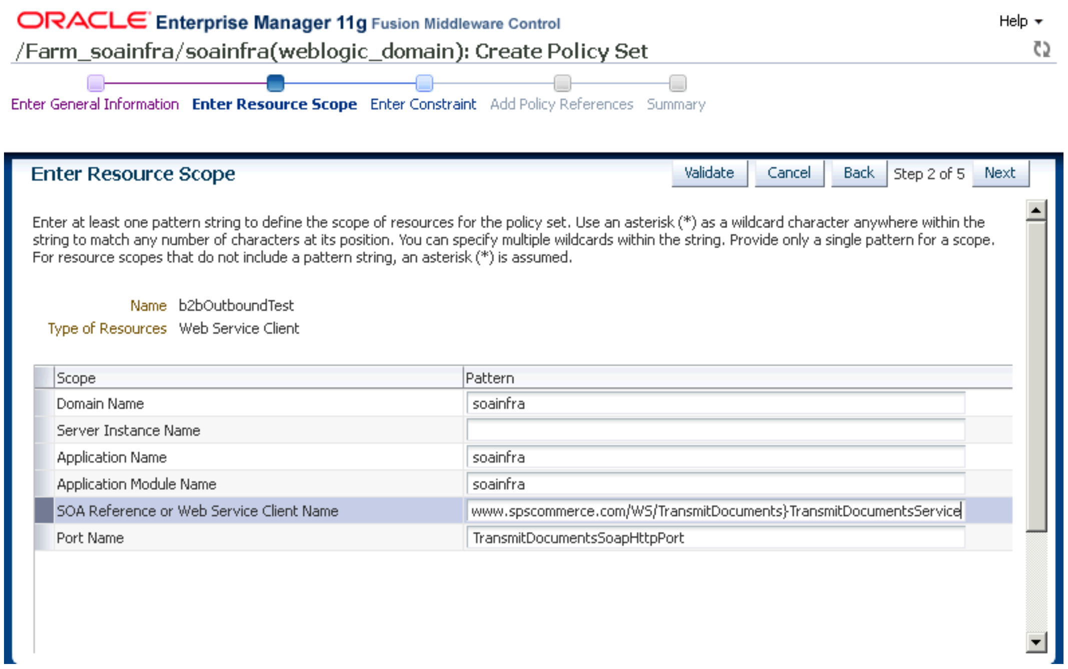
Enter basic policy set information, such as name, description, and resource type. Click Next to enter the resource scope.

* Name	b2bOutboundTest
Enabled	✓
* Type of Resources	Web Service Client
Description	

4. In the Enter Resource Scope page, enter the following as displayed in Figure 21-11 and click **Next**:
   - The domain name, such as `soainfra`

- The application name, such as `soainfra`
- The application module name, such as `soainfra`
- The SOA reference, such as `{http://xmlns.example.com/soa/generic/soap}test`
- The port name, such as `TransmitDocumentsSoapHttpPort`

Figure 21-11 Create Policy Set - Enter Resource Scope Page


ORACLE Enterprise Manager 11g Fusion Middleware Control
Help
/Farm_soainfra/soainfra(weblogic_domain): Create Policy Set
Enter General Information
Enter Resource Scope
Enter Constraint
Add Policy References
Summary
Enter Resource Scope
Validate
Cancel
Back
Step 2 of 5
Next

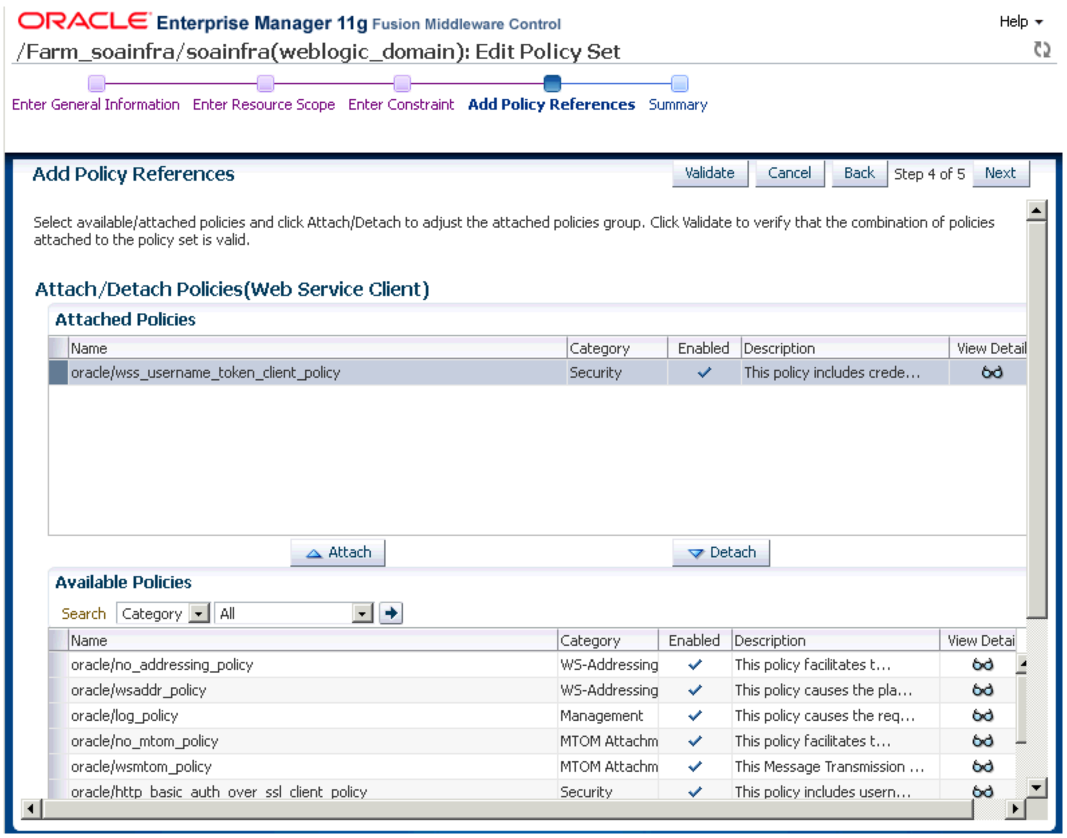
Enter at least one pattern string to define the scope of resources for the policy set. Use an asterisk (*) as a wildcard character anywhere within the string to match any number of characters at its position. You can specify multiple wildcards within the string. Provide only a single pattern for a scope. For resource scopes that do not include a pattern string, an asterisk (*) is assumed.

Name	b2bOutboundTest
Type of Resources	Web Service Client

Scope	Pattern
Domain Name	soainfra
Server Instance Name	
Application Name	soainfra
Application Module Name	soainfra
SOA Reference or Web Service Client Name	www.spscommerce.com/WS/TransmitDocuments}TransmitDocumentsService
Port Name	TransmitDocumentsSoapHttpPort

5. In the Enter Constraint page, click **Next**.
6. In the Add Policy References page, select the **oracle/wss_username_token_client_policy** from the Available Policies section, click **Attach** as displayed in Figure 21-12, and then click **Next**.

Figure 21-12 Create Policy Set - Add Policy References Page


ORACLE Enterprise Manager 11g Fusion Middleware Control
Help
/Farm_soainfra/soainfra(weblogic_domain): Edit Policy Set
Enter General Information
Enter Resource Scope
Enter Constraint
Add Policy References
Summary
Add Policy References
Validate
Cancel
Back
Step 4 of 5
Next

Select available/attached policies and click Attach/Detach to adjust the attached policies group. Click Validate to verify that the combination of policies attached to the policy set is valid.

Attach/Detach Policies(Web Service Client)

Attached Policies

Name	Category	Enabled	Description	View Detail
oracle/wss_username_token_client_policy	Security	✓	This policy includes crede...	👓

Name	Category	Enabled	Description	View Detai
oracle/no_addressing_policy	WS-Addressing	✓	This policy facilitates t...	👓
oracle/wsaddr_policy	WS-Addressing	✓	This policy causes the pla...	👓
oracle/log_policy	Management	✓	This policy causes the req...	👓
oracle/no_mtom_policy	MTOM Attachm	✓	This policy facilitates t...	👓
oracle/wsmtom_policy	MTOM Attachm	✓	This Message Transmission ...	👓
oracle/http_basic_auth_over_ssl_client_policy	Security	✓	This policy includes usern...	👓

7. In the Summary page, click **Save** to return to the Policy Set Summary page.

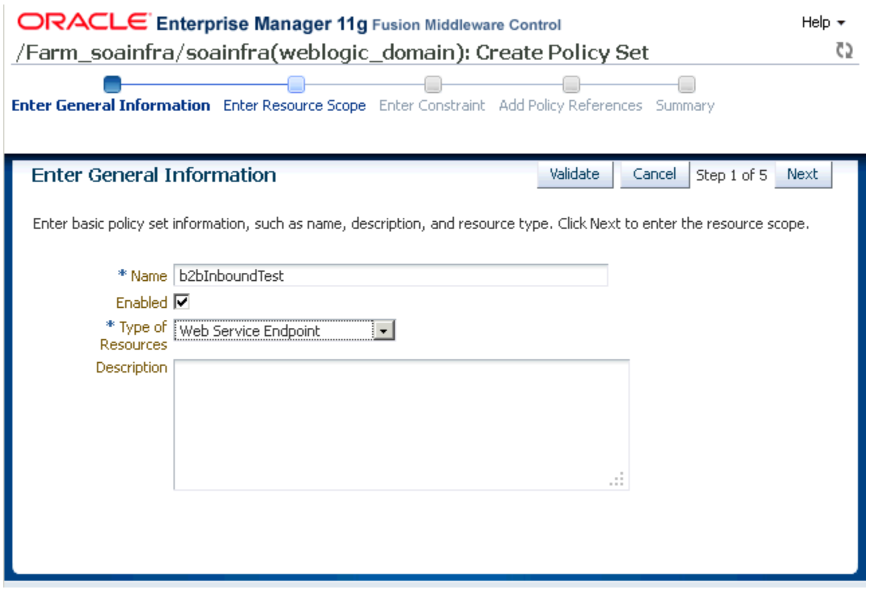
Creating and attaching inbound policy sets

The final task is to create and attach inbound policy sets.

To create an inbound policy set and attach it to Web services:

1. Repeat steps 1 and 2 from Creating and attaching outbound policy sets.
2. Enter the name as `b2bIutboundTest`, select **Enabled**, select **Web Service Endpoint** from the Type of Resources list, and click **Next** as displayed in Figure 21-13.

Figure 21-13 Create Policy Set - Enter General Information Page


ORACLE Enterprise Manager 11g Fusion Middleware Control
Help
/Farm_soainfra/soainfra(weblogic_domain): Create Policy Set
Enter General Information
Enter Resource Scope
Enter Constraint
Add Policy References
Summary
Validate
Cancel
Step 1 of 5
Next

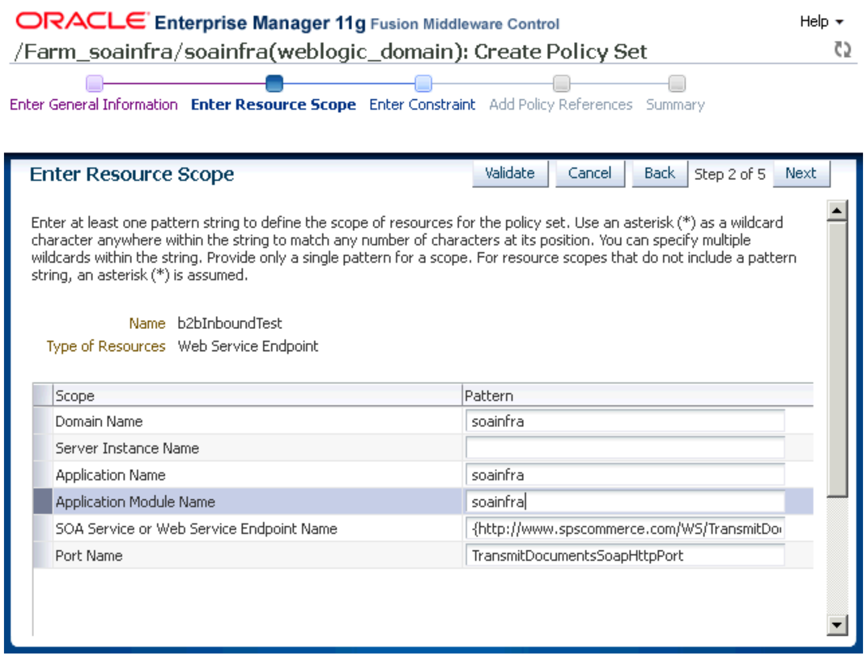
Enter General Information

Enter basic policy set information, such as name, description, and resource type. Click Next to enter the resource scope.

* Name	b2bInboundTest
Enabled	✓
* Type of Resources	Web Service Endpoint
Description	

3. In the Enter Resource Scope page, enter the following as displayed in Figure 21-14 and click **Next**:
   - The domain name, such as `soainfra`
   - The application name, such as `soainfra`
   - The application module name, such as `soainfra`
   - The SOA reference or Web service client name, such as `{http://www.spscommerce.com/WS/TransmitDocuments}TransmitDocumentsService`
   - The port name, such as `TransmitDocumentsSoapHttpPort`

Figure 21-14 Create Policy Set - Enter Resource Scope Page


ORACLE Enterprise Manager 11g Fusion Middleware Control
Help
/Farm_soainfra/soainfra(weblogic_domain): Create Policy Set
Enter General Information
Enter Resource Scope
Enter Constraint
Add Policy References
Summary
Validate
Cancel
Back
Step 2 of 5
Next

Enter Resource Scope

Enter at least one pattern string to define the scope of resources for the policy set. Use an asterisk (*) as a wildcard character anywhere within the string to match any number of characters at its position. You can specify multiple wildcards within the string. Provide only a single pattern for a scope. For resource scopes that do not include a pattern string, an asterisk (*) is assumed.

Name b2bInboundTest
Type of Resources Web Service Endpoint

Scope	Pattern
Domain Name	soainfra
Server Instance Name	
Application Name	soainfra
Application Module Name	soainfra
SOA Service or Web Service Endpoint Name	{http://www.spscommerce.com/WS/TransmitDo
Port Name	TransmitDocumentsSoapHttpPort

4. In the Enter Constraint page, click **Next**.
5. In the Add Policy References page, select the **oracle/ wss_username_token_service_policy** from the Available Policies section, click **Attach**, and then click **Next**.
6. In the Summary page, click **Save** to return to the Policy Set Summary page.

Identifying the Trading Partner at Runtime

At runtime, you can identify the trading partner from which the message is received on the basis of various transport headers.

You can identify a trading partner by using the following information (listed in order of priority):

- Matching the value of HTTP (From) Header Override with a Generic Identifier of the trading partner
- Using XPath to identify `FROM_TP` found in the SOAP Header
- Matching the value of HTTP (From) Header with a Generic Identifier of the trading partner
- Matching the value of SOAP (From) Header with a Generic Identifier of the trading partner

  Note: In case you cannot use the predefined SOAP headers, you can override the same and use XPath to identify trading partner.
- Matching the value of WS Security Authenticated User with a Generic Identifier of the trading partner
- Matching the value of Remote HostName with a Generic Identifier of the trading partner
- Matching the value of SOAP envelope header “PartyInfo” with the Generic Identifier of the trading partner.

Sample Request-Reply Scenarios

These scenarios are for synchronized request-reply when using a Web service or the Generic SOAP.

The section contains:

- Outbound Synchronization: Composite
- Inbound Synchronization: Composite
- Outbound Synchronization: JMS Queues
- Inbound Synchronization: JMS Queues

Outbound Synchronization: Composite

In the case of outbound synchronization with a composite:

- **Web Service:** Relies on the composite to decide whether the synchronization is one-way, or if the request/reply pattern is used. If `soapAction` is provided, it is used only to overwrite the HTTP `soapAction` headers.

  Resubmission of a synchronized request/reply message needs to be done from the composite. If any such message is resubmitted from the Oracle B2B console, it is be treated as a one-way message.

- **Generic SOAP:** Relies on the composite to decide one-way or request/reply operation. The channel configuration (**None**/**Sync**) does not take effect.

Inbound Synchronization: Composite

In the case of inbound synchronization with a composite:

- **Web Service:** Uses `soapAction` from the HTTP header to decide whether it is a one-way or request/reply operation specified by the WSDL. If `soapAction` is not defined in the WSDL, then Oracle B2B falls back to the Channel configuration. In the case of a non-responsive payload, an error is reported for request/reply.
- **Generic SOAP:** Depends on the Channel configuration (**None**/**Sync**). In the case of a non-responsive payload, an error is reported for request/reply.

Outbound Synchronization: JMS Queues

In the case of outbound synchronization with a JMS queue:

- **Web Service:** Uses `soapAction` (provided from the back end or configured in the listening channel) to decide whether it is a one-way or request/reply operation specified by the WSDL. If `soapAction` cannot be located in the WSDL, Oracle B2B defaults to the Channel configuration. Reply is sent to the back end by using the inbound agreement configuration.
- **Generic SOAP:** Depends on the Channel configuration (**None**/**Sync**). If no reply comes back for **Sync** case, Oracle B2B reports an error. Reply is sent to the back end by using inbound agreement configuration.

Inbound Synchronization: JMS Queues

- **Web Service:** Uses `soapAction` from the HTTP header to decide whether it is a one-way or request/reply operation specified by the WSDL. If `soapAction` is not defined in the WSDL, Oracle B2B falls back to the Channel configuration (**None**/**Sync**). In the case of **Sync**, a reply needs to be generated in a Oracle B2B callout to be sent as response back to the trading partner.

  If the incoming request has WS-Addressing enabled, then the response message sends the ws-addressing mandatory headers.
- **Generic SOAP:** Depends on the Channel configuration (**None**/**Sync**). In case of **Sync**, a reply needs to be generated in a Oracle B2B callout to be sent as response back to the trading partner.

22 Enabling AS4-Based Message Exchange

This chapter describes how to enable AS4-based message (typically SOAP-based) exchange between trading partners in Oracle B2B.

This chapter contains the following sections:

Introduction to AS4–Based Message Exchange

Applicability Statement4 (AS4) is a leading standard that aggregates the web service standards to provide necessary transactional features. It standardizes exception handling by defining acknowledgement receipts and error messages, and supports message choreographies. Oracle B2B supports the AS4 protocol in its exchange protocol stack for secure document-agnostic exchange of payloads using web services.

AS4 provides various features such as one-way push, one-way pull, reliability and security over web services. AS4 secures data with authentication, message integrity, non-repudiation of origin, and privacy features.

Exchanging AS4-Based Service Messages with Custom WSDL File

The support for AS4-based messages is available for both inbound and outbound directions. You need to create or use Generic WebService or upload a Web Service Definition Language (WSDL) file that you can customize according to your requirement.

AS4 ebHandler mandates SOAP 1.2, the requirements from BSP 1.1 that depend on SOAP 1.1 would not apply. Similarly, none of the requirements for DESCRIPTION (WSDL) or REGDATA (UDDI) apply here, as these are not used. In the case of SOAP 1.1, the SOAP fault will be received from the service.

It is not necessary to provide a WSDL, because AS4 eliminates WSDL complexity by avoiding the pitfalls of mapping document types and business processes to SOAP operations and actions. The existing Generic WSDL can be used with SOAP 1.2 support.

Exchanging Outbound Messages

From the backend Application (Fabric/AQ/JMS/File/FTP/SFTP) the message will be sent to B2B like other messages. In B2B the Agreement identification will happen based on the From/To/Action and Service or From/To/Doctype and Doc Revision.

Based on the Agreement Configuration, the message (any type of document) will be processed using AS4 Exchange Plugin, SOAP Packing and send through AS4 Transport as configured in the trading partner delivery channel. Based on the Configured delivery channel SYNC/None and Async mode of transport can be supported.

Exchanging Inbound Messages

Based on the registered AS4 WS listening channel the trading partner can post the message with appropriate AS4 headers. The user can register an AS4 endpoint using administration -> Listening Channel. The document will be identified based on the user message soap header or normal document identification flow. The message will be processed by AS4 exchange plugin and deliver to the configured backend application.

The AS4 Exchange can be identified by the following url:

```
http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/
```

AS4 exchange plugin also uses WS-HTTP as transport layer. The generic SOAP exchange will use this for identify the exchange. The AS4 Exchange plugin is added much before Generic WS, so if the exchange is identified as AS4 then Generic Exchange will skip the identification layer.

The document identification will be based on the SOAP Header “CollaborationInfo” -> Action/ Service combination. The trading partner Identification will be always based on the ebMS 3.0 specification, PartyInfo ->From/To partyId and type combination. There is no specific identifier for AS4, the user can configure custom identifier as part of the administration->types and use the same in partner profile configuration. Default identifier is name and any other Identified like duns. But there is no specific identifier for AS4 (no AS4 Identifier)

Setting up Trading Partners and Hosts

Oracle B2B provides support for AS4 web-service based message exchange with trading partners. This is achieved by having a composite with B2B binding, or AQ, JMS, File, FTP, or SFTP to initiate the payload from the backend application. The same can be sent to the trading partner using AS4.

On the inbound side, Oracle B2B receives the AS4 message and delivers it to the backend application using a SOA composite or through AQ, JMS, File, FTP, or SFTP. Oracle B2B can identify the incoming trading partner based on the SOAP envelope header “PartyInfo”.

To add the initiating trading partner and configure the trading partner delivery channel:

1. In **Partner Configuration**, click **Partners**.
2. Rename the host to your company name.
3. Click **+** to add a trading partner and enter the name of the partner.
4. Click the trading partner you just created.
5. Click the **Documents** tab to add the supported documents and click **+** to add the appropriate document.

6. Click the **Channels** tab and click **+** to add the exchange/transport.
7. In the **Protocol** list, select `AS4-1.0`.
8. If you have defined a custom WSDL file in admin, select it here. If you do not have a custom WSDL file, select the **Use Generic SOAP** option.
9. From the **Service** drop-down list, select `Generic SOAPService`.
10. From the **Port** drop-down list, select `Generic Port`.
11. From the **SOAP Action** drop-down list, select `generic/soap/process`.
12. Enter the URL of the endpoint.

    An example URL: `http://myhost.com:7878/b2b/services/ws/tradingpartnername_listeningchannel`
13. Click **Save** to save the details.

> **Note:**
>
> Follow the steps above to set up the responder as well, using the name of the responder in step 2.

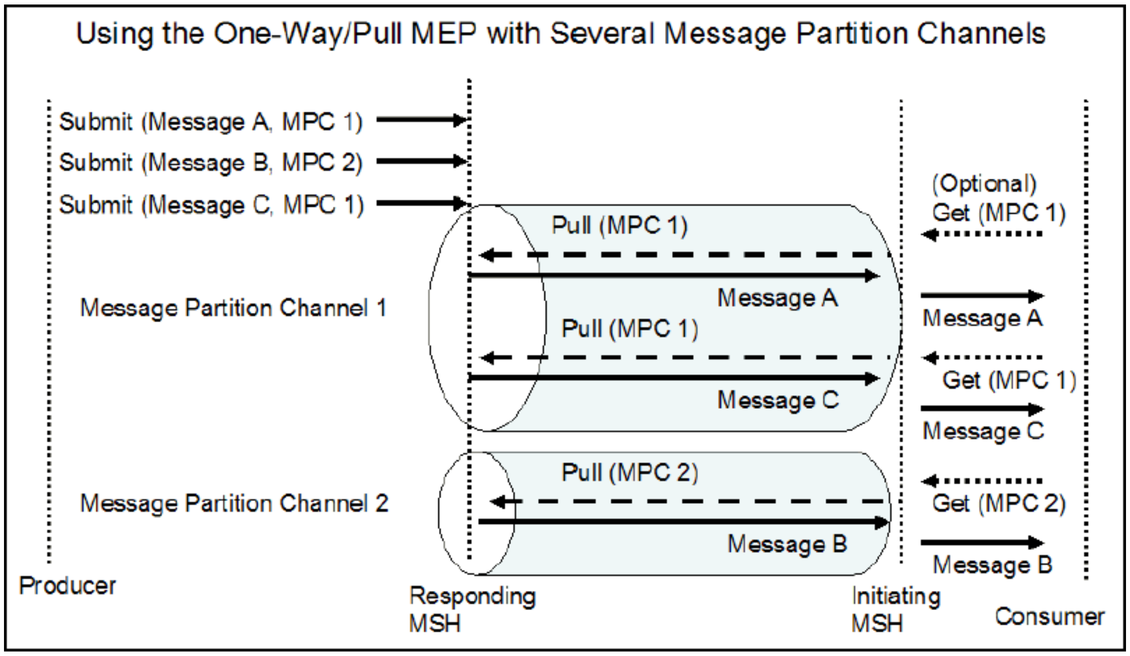
Message Partition Channels

The Message partition channels (MPC) allows for partitioning the flow of the message from sender to receiver into several flows that can be controlled separately and consumed differently.

MPC allows:

- **Setting the transfer priorities:** Some messages may be transferred with higher priority than others regardless of the order which they have been submitted.
- **Organizing the inflow of messages on the receiver side:** Enables the receiver to dictate each flow in a distinct way.

The following figure shows an example of the MPC work flow for one-way pull transactions.


Using the One-Way/Pull MEP with Several Message Partition Channels
Submit (Message A, MPC 1)
Submit (Message B, MPC 2)
Submit (Message C, MPC 1)
Message Partition Channel 1
Message Partition Channel 2
Producer
Pull (MPC 1)
Message A
Pull (MPC 1)
Message C
Pull (MPC 2)
Message B
Responding MSH
Initiating MSH
(Optional) Get (MPC 1)
Message A
Get (MPC 1)
Message C
Get (MPC 2)
Message B
Consumer

This can be achieved in Oracle B2B's existing sequencing mechanism. From the backend application, if the messages are sent with the "MPC" property, then MPC is used as a sequence message target. It is inserted into the appmessage table and sequence manager table. The messages cannot be processed by B2B until the pull message request is received.

Based on the pull message request the corresponding message is picked from the sequence manger table and enqueue into event queue. The message thenpasses through the agreement identification based on the From/To/Action/Service or From/To/Doctype/Docrevision. Based on the agreement configuration, the message is processed and sent to the trading partner as per the delivery channel configuration.

The message prioritization can be handled as part of pull request and response.

Duplicate Message Detection

The duplicate message detection feature provides the ability to detect the message duplication based on the message id. The existing duplicate detection feature is re-used for AS4.

It can be configured as part of the delivery channel. In case of inbound message flow, if the duplicate detection is enabled then the message duplication is identified and a fault message is sent to the message sender.

Example 22-1 Example — Duplicate Message Detection

The following message id is used to detect the message is duplicated or not.

eb:MessageInfo/eb:MessageId.

P-Mode Parameters

B2B supports P-Mode parameters. The parameters relevant to AS4 features are listed here.

Table 22-1 P-Mode Parameters and Descriptions

P-Mode Parameter	Description
PMode.ID	Trading Partner profile, the identifier can be configured.
PMode.Agreement	The Agreement id will be used to provide th agreement Ref.
PMode.MEP	The message exchange pattern(MEP) will be based on the eventName enqueued from backend application.
PMode.MEPbinding	The message exchange pattern (MEP) will be based on the eventName enqueued from backend application. The trading partner delivery channel configuration is available.
PMode.Initiator.Party	While Enqueue set the From Party will be always a initiator
PMode.Initiator.Role	While Enqueue set the ActionName/event Name to set the FromRole
PMode.Initiator.Authorization.username	Set the Auth user in OWSM credentials, incase of Basic Authentication
PMode.Responder.Party	While Enqueue set the To Party will be always a Responder
PMode.Responder.Role	While Enqueue set the ActionName/event Name to set the ToRole
PMode.Responder.Authorization.username	Set the Auth user in OWSM credentials, incase of Basic Authentication
Protocol.Address	There is no need to configure anything specific to this, as part of delivery channel http will be consider always.
Protocol.SOAPVersion	AS4 delivery channel configuration is enough, it will always uses SOAP 1.2
BusinessInfo.Service:	As part of enqueue, set the ActionName/event Name with Service or other wise set as part of Custom Document protocol in case of static service
BusinessInfo.Action	As part of enqueue, set the ActionName/event Name with Action or other wise set as part of Custom Document protocol in case of static Action
BusinessInfo.Properties[]:	Not Supported to add Properties
ErrorHandling.Report.ReceiverErrorsTo	Incase of error/exception B2B will send a exception message to sender
ErrorHandling.Report.AsResponse	Incase of error/exception B2B will send a exception message to sender
[1].ErrorHandling.Report.ProcessErrorNotifyProducer	The errors will be processed and it will also update request message state to Error.
ErrorHandling.Report.DeliveryFailuresNotifyProducer	The errors will not be delivered then the error receipt message will be change the state to Error.
Security.WSSVersion	OWSM configuration handles
Security.X509.Sign	OWSM configuration handles
Security.X509.Signature.Certificate	OWSM configuration handles
Security.X509.Signature.HashFunction	OWSM configuration handles
Security.X509.Signature.Algorithm	OWSM configuration handles
Security. X509.Encryption.Encrypt	OWSM configuration handles
Security.X509.Encryption.Certificate	OWSM configuration handles

Table 22-1 (Cont.) P-Mode Parameters and Descriptions

P-Mode Parameter	Description
Security.X509.Encryption.Algorithm	OWSM configuration handles
Security.UsernameToken.username	OWSM configuration handles
Security.UsernameToken.password	OWSM configuration handles
Security.UsernameToken.Digest	OWSM configuration handles
Security.UsernameToken.Created	OWSM configuration handles
Security.PModeAuthorize	OWSM configuration handles
Security.SendReceipt	OWSM configuration handles
Security.SendReceipt.ReplyPattern	OWSM configuration handles

Local Policy Attachment

You can attach multiple policies to a delivery channel using LPA (Local Policy Attachment). As part of the delivery channel configuration, you select the OWSM policies and attach to the particular endpoint.

This is used to handle the policies attached for the webservice endpoint. The list of policies can be selected from the drop -down list and then provide the necessary information. For example, message protection policy will be used to protect the messages (sign+encryption) based on the certificate alias configured in the policy.

Use-Case Scenarios

Scenarios about exchanging documents between two partners.

In the following scenarios, we are assuming that two partners, Acme and GlobalChips, want to exchange documents using AS4 Protocol. For this purpose, we assume that Oracle B2B AS4 is installed on two separate servers (one for each partner). One of the servers, called Acme, acts as the initiator and sends an XML document using the AS4 protocol to the responder, called GlobalChips.

Inbound Messaging

The sending and receiving MSH are configured to exchange a particular message type using reliable messaging. The sending MSH sends a signed message of this type.

The expected result is that the receiving MSH returns a synchronous AS4 non-repudiation receipt. The content of the NonRepudiationInformation element in the returned receipt MUST match the Signature of the received message. This can be determined using Message logs, message trackers and/or TCP monitoring tools; the latter requires the to not use TLS.

Setting Up the Responder

To set up the responder, in this case, GlobalChips:

1. Login to Oracle B2B on the receiver side,
2. Click **Administration** to create a document.
3. In the **Document** tab, select **Custom** and add the Document Version, for example 1.0.
4. Select the Document Version created and add **Document Type**, for example, ORDERS.

5. Select the Document Type created and add **Document Definition**, for example, Ord_def.
6. Click **Definition** to provide the schema.
7. Select the **Identification Type** as xml.
8. Provide the **Identification Expression** (XPath) to identify the incoming document.

   Example: `/*/*[local-name()='shipto']/*[local-name()='country']`
9. Click **Save** to save the details.
10. On the Receiver side, as part of setting up the listening channel, click on the **Policy Configuration**.
11. From the policy list, select the **oracle/wss10_message_protection_service_policy** and attach.
12. Click **+** on the above selected policy to provide the certificate alias
13. Provide **keystore.sig.csf.key** as the private key alias of the host to verify the signature of the messages.
14. Provide **keystore.enc.csf.key** as private key alias of the host to decrypt the message.

23

Using Scripts for Archiving and Restoring Data

This chapter describes how to archive and restore B2B business messages using SQL scripts.

This chapter includes the following sections:

- Introduction to Archiving and Restoring B2B Business Messages
- Archiving B2B Business Messages
- Restoring B2B Business Messages

See Importing and Exporting Data for information on importing and exporting design-time data.

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 runtime instance data *in Oracle databases*.

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 runtime 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 runtime data into Oracle B2B (Oracle Metadata Service repository) and accessing it through B2B reports.

For non-Oracle databases, external database archiving tools can be used to export and import runtime data.

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
   ```

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 23-1 lists the parameters for the `b2b_archive_procedure` API.

Table 23-1 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`.

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 directory 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.

Note:

Archiving and restoring of RT data from one major release (such as 11.1.1.7.0) to another major release (12.1.3.0.0) is not supported.

24 Utilities for Enqueuing and Dequeuing

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.

This chapter includes the following sections:

- AQ Enqueue and Dequeue Utilities
- JMS Enqueue and Dequeue Utilities
- Using the attachmentsDescriptor.xsd

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.

AQ Enqueue

Table 24-1 lists the Java AQ enqueue utility (`oracle.tip.b2b.data.IPEnqueue`) properties.

Table 24-1 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
attachment_name	Attachment file name. Use this property to assign a name to the attachment file that is something other than the email subject name. For a custom outbound message over email with AQ do the following: `actionName=ATTACHMENT_NAME:Sample.txt`

Table 24-1 (Cont.) IPEnqueue Properties

Name	Description
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 email delivery channel
dynamicemail	Provides the `to` party email address as part of `actionName`. For example: `actionName=DYNAMICEMAIL:`*`email_id`*
dynamic_from_email	Provides the `from` party email 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.

AQ Dequeue

To dequeue messages, use the `IPDequeue` utility.

Table 24-2 lists the Java AQ dequeue utility (`oracle.tip.b2b.data.IPDequeue`) properties.

Table 24-2 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
subscriber	Default is b2buser

Example: `ipdequeue.properties:`

```
queue            =
count            = 1
output           = t1.trc
url              = jdbc:oracle:thin:@host:1521:sid
user             = user1
password         = password
subscribee       = b2buser
```

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.

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 24-3 lists the properties that can be configured in the file.

Table 24-3 JMS Enqueue Properties

Name	Description
destination	JNDI name of queue or topic to send message to

Table 24-3 (Cont.) JMS Enqueue Properties

Name	Description
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
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 email subject name.
	For a custom outbound message over email with JMS do the following:
	`constant : Sample.xml property :`
	`jca.jms.JMSProperty.ATTACHMENT_NAME`

dateFormat	Used to convert the date format used in email delivery channel

dynamicemail	Provides the `to` party email address. For example:
	`constant : `*`email_id`*` property :`
	`jca.jms.JMSProperty.DYNAMICEMAIL`

dynamic_from_email	Provides the `from` party email address. For example:
	`constant : `*`email_id`*` property :`
	`jca.jms.JMSProperty.DYNAMIC_FROM_EMAIL`

Example 24-1 shows the sample `jms_enqueue.properties` file.

Example 24-1 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.

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 24-4 are set as part of the `javax.jms.Message` delivered to the destination that B2B listens on.

Table 24-4 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

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 24-2 shows the sample JMS dequeue properties file.

Example 24-2   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>
```

Using the attachmentsDescriptor.xsd

Use the `attachmentsDescriptor.xsd` file for sending attachments.

Example 24-3 shows a sample attachment XML file.

Example 24-3   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>
```

25 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 includes the following sections:

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.

Creating a Data Source in Oracle WebLogic Server

You could use an existing data source if it exists or create a new data source.

Follow 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. after the test is successful, click **Next.**
11. Select the server where Oracle BAM is deployed, and click **Finish**.

Creating a Foreign JMS Server

Create a foreign JMS server to monitor messages sent to and from a BAM 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`). Typically, your Data Source should point to the Oracle B2B `SOAINFRA` schema.
   d. Check the **Default Targeting Enabled** check box.
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`). The JNDI name can be arbitrary as long as it matches the name of the created JMS queue.
   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. Create an Outbound Connection JNDI for Oracle BAM Enterprise Message Source
   a. In the left pane of the Administration Console, click **Deployments**
   b. On the Summary of Deployments page, expand **BamServer > BeamAdapter.rar**. Click **BeamAdapter.rar**.
   c. On the Settings for BeamAdapter page, click **Configuration > Outbound Connection Pools**, and then click **New**.
   d. Select **oracle.tip.adapter.jms.IJmsConnectionFactory**, and then click **Next**.
   e. In the **JNDI Name** field, enter a name that will be used by Oracle BAM Enterprise Message Source. For example, eis/bam/b2bconnection

      Message element name must be always “<row>", Type the parent element that contains column values in either its sub-elements or attributes.

      To configure Source Value Formatting, check the DateTime Specification check box and click the hyper link specified below the check box.
   f. Click **Finish**.
   g. On the Save Deployment Plan Assistant page, click **OK**
8. Add JNDI name to Oracle BAM Enterprise Message Source.
   a. On the Settings for BeamAdapter page, click **Configuration > Outbound Connection Pools**.
   b. Expand **oracle.tip.adapter.jms.IJmsConnectionFactory** and then select the JNDI name that you created in step 7.
   c. In the **Outbound Connection Properties** table, search for Property Name - **FactoryProperties** and add *"java.naming.factory.initial=weblogic.jndi.WLInitialContextFactory;java.naming.provider.url=t3://<host>:<port>;java.naming.security.principal=<user>;java.naming.security.credentials=<pwd>;"* under the **Property Value** column, and then press **Save**.
9. Restart Oracle WebLogic Server and BAM managed server.

Create a B2B Data Object using Oracle BAM Composer

The BAM Composer can create configured to create B2B data objects.

To create a B2B data object:

1. Log in to Oracle BAM at the following URL:

   `http://`*`host_name`*`:`*`port_number`*`/bam/composer`
2. Click **Administrator** tab to manage data objects and enterprise message sources.

3. In the left pane of the BAM Composer administrator page, click **Data Objects**, and click then click **+** button to add new data object.
4. Enter a name in the **Name** field. For example, *B2BDO*.

   Similarly, add details to all the fields listed in the Data Objects dialog box, and then click **Create**. The data object fields includes:

   Table 25-1 Data Objects

Property	Values and Examples
Name	Enter a name for the Data Object
Type	Select **Simple Data Object**
Continuous Query Type	Select **Relation**

5. Click the **Columns** tab of the data object to add all fields as listed in Table 25-3.

For more information about creating Oracle BAM data objects, see "Creating and Managing Oracle BAM Data Objects" in *Oracle Fusion Middleware Monitoring Business Activity with Oracle BAM*.

Define a JMS Message Source Using an Oracle BAM Enterprise Message Source

An Oracle BAM Enterprise Message source can be used to define a JMS message source in Oracle B2B.

To define a JMS message source:

1. Log in to Oracle BAM at the following URL:

   `http://`*`host_name`*`:`*`port_number`*`/OracleBAM`

2. In the left pane of the BAM Composer administrator page, click **Enterprise Message Sources**, and then click **+** button to add new Enterprise Message Source.
3. Enter a name in **Name** field of the **Enterprise Message Source** dialog box.
4. Click **Create**, and then enter the Enterprise Message Source information for AQ JMS Queue as indicated in Table 25-2, leaving unmentioned fields with default or blank entries.

   Table 25-2 Enterprise Message Source Properties for AQ JMS Queue

Property	Values and Examples
Name	Enter a name for the Enterprise Message Source (for example, `B2B_EMS`).
Display Name	Enter a display name for the Enterprise Message Source (for example, `B2B_EMS`).
Outbound Connection JNDI	Select the outbound connection JNDI created in Creating a Foreign JMS Server, step 7.
Topic/Queue Name	Enter the name of your JMS topic (or queue) (for example, jms/BAMAQQueue).
Data Object Name:	Choose the B2B data object to send the values received from AQ JMS server

Table 25-2 (Cont.) Enterprise Message Source Properties for AQ JMS Queue

Property	Values and Examples
Operation	Choose the **Insert** or **Upsert** operation, depending on the use case.
Message Element Name	Type the parent element that contains column values in either its sub-elements or attributes.

5. To configure **Source Value Formatting**, check the **DateTime Specification** check box, 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 25-3 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, expand Data Objects from the BAM Composer administrator page, and select the B2BDO object. Click **Data** tab 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 Monitoring Business Activity with Oracle BAM*.

Mapping Oracle B2B Instance Messages to Oracle BAM Data Objects

Oracle B2B instance message fields can be used to design data objects and Enterprise Message Sources in Oracle BAM to monitor Oracle B2B performance in real time.

Table 25-3 lists the Oracle B2B instance message fields.

Note that the payloads (`APP_PAYLOAD`, `PAYLOAD`, `WIREPAYLOAD`) are not included in the list below. They are not transferred for performance reasons.

Table 25-3 Oracle B2B Instance Message Fields

Field	Type
ID	NOT NULL VARCHAR2(256)
REFERTOID	VARCHAR2(256)
B2BMESSAGEID	NOT NULL VARCHAR2(256)
ACKNOWLEDGEMODE	VARCHAR2(256)
MESSAGEDATETIME	DATETIME
MESSAGETYPE	VARCHAR2(256)
STATE	VARCHAR2(256)
REMAININGRETRY	INT
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	DATETIME
MODIFIED	DATETIME
SEND_TIMESTAMP	DATETIME
RECEIVE_TIMESTAMP	DATETIME
NATIVE_MSG_SIZE	INT
TRANSLATED_MSG_SIZE	INT
BM_RESUBMIT_COUNT	INT
ERRORCODE	VARCHAR2(256)
ERRORTEXT	VARCHAR2(1024)
ERRORDESCRIPTION	VARCHAR2(2000)
PROCESSING_TIME	INT
DOCUMENTTYPE	VARCHAR2(513)
CORRELATIONID	VARCHAR2(256)
FABRIC_ECID	VARCHAR2(512)
FABRIC_COMPOSITE_NAME	VARCHAR2(512)
AM_RESUBMIT_COUNT	INT

Table 25-3 (Cont.) Oracle B2B Instance Message Fields

Field	Type
TRANSACTIONCONTROLNUMBER	VARCHAR2(256)
GROUPCONTROLNUMBER	VARCHAR2(256)
INTERCHANGECONTROLNUMBER	VARCHAR2(256)
B2BWIREMESSAGEID	VARCHAR2(256)
B2BTIMESTAMP	DATETIME
CONVERSATIONID	VARCHAR2(1024)
PROTOCOLMESSAGEID	VARCHAR2(1024)
URL	VARCHAR2(1024)
TRANSPORTPROTOCOL	VARCHAR2(513)
TRANSPORTHEADERS	VARCHAR2(2000)
WM_RESUBMIT_COUNT	INT
DOCUMENTPROTOCOL	VARCHAR2(513)
EXCHANGEPROTOCOL	VARCHAR2(513)
AGREEMENTID	VARCHAR2(256)
SENDERNAME	VARCHAR2(256)
SENDERID	VARCHAR2(513)
RECEIVERNAME	VARCHAR2(256)
RECEIVERID	VARCHAR2(513)

Creating a Dashboard to Monitor Oracle B2B

Real-time monitoring dashboards are created using Oracle BAM Active Studio.

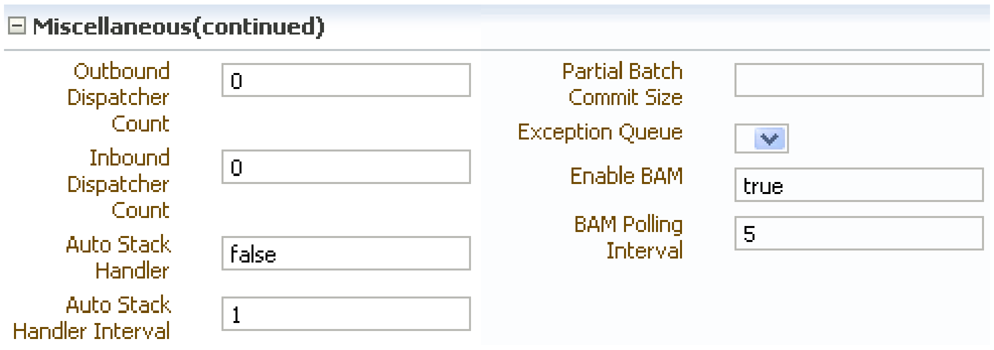
To create a dashboard to monitor Oracle B2B using the B2BDO data object, see *Oracle Fusion Middleware Monitoring Business Activity with Oracle BAM*, chapter 8 Creating Dashboards.

Enabling the Oracle BAM Integration in Oracle B2B

To enable the Oracle BAM integration in Oracle B2B, set 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.



Miscellaneous(continued)

Outbound Dispatcher Count	0	Partial Batch Commit Size	
Inbound Dispatcher Count	0	Exception Queue	
Auto Stack Handler	false	Enable BAM	true
Auto Stack Handler Interval	1	BAM Polling Interval	5

[also need to update bb_config.fm table with these parameters]

26 Programmatically Accessing Instance Message Data

This chapter provides an overview of the Instance Message Java API for programmatically accessing instance message data.

This chapter includes the following sections:

For more information about the Instance Message API, `InstanceMessageUtil` class, see *Oracle Fusion Middleware B2B Instance Message Java API Reference*.

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*.

Instance Message Java API Examples

This example gives code samples for retrieval based on trading partner name

You can write your logic based on the out put of

```
Vector messages = (Vector) imUtil.getInstanceMessagesForTP("GlobalChips")
```

as shown in Example 26-1.

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 26-1 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());
    }
}
```

Troubleshooting Instance Message Java API

Examples are given for troubleshooting handling a large number of messages and large paylods, and executing queries remotely.

See the following topics for troubleshooting information:

- Handling a Large Number of Messages
- Handing Large Payloads

- Executing Queries Remotely Using Weblogic.jar

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
```

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 Handling Large Payloads for more information about how Oracle B2B handles large payloads.

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.AbstractSubject` 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
Appendices

This part contains information about handling large payloads and email attachments, diagnosing issues, setting properties and using the back-end application interface, and various utilities.

A

Handling Large Payloads

This appendix describes how to use Oracle B2B to handle large payloads by using the SOA Infrastructure and JMS internal queues.

The appendix includes the following sections:

- Handling Large Payloads
- Using Document Streams to Handle Large Payloads

Handling Large Payloads

Oracle B2B can handle large payloads through the SOA Infrastructure and JMS internal queues. Examples of large payloads that B2b transactions can handle include a large batch file containing multiple purchase orders.

Note, however, that the definition of a large payload is user-defined and varies by customer and use case. Anything that is in the payload that is above a large payload is not stored in the database but in the file system and specified as part of the large payload configuration. This helps save database table space and optimizes runtime performance.

Additionally, with the Document Streaming feature available in 12.1.3, the document is processed through a stream rather than loading the whole payload into memory, if the document channel and type meet the streaming criteria. Please see the streaming documentation below for more details.

Note:

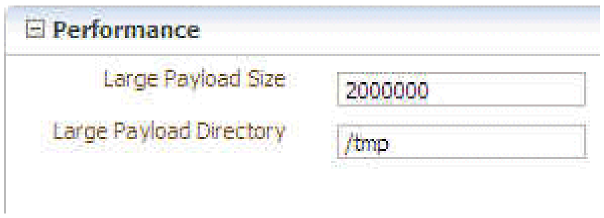
A JMS large payload is sent via the JMS Body instead of a file reference. If you want to send a file reference to the back end, set `b2b.InlineJMSLargePayLoad` to false.

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

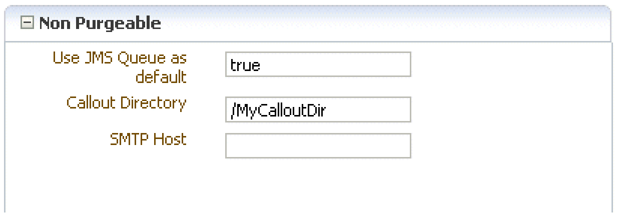


⊟ Performance

Large Payload Size	2000000
Large Payload Directory	/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



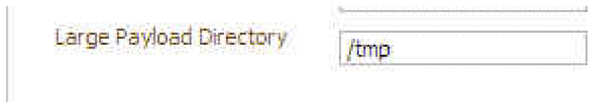
Use JMS Queue as default: true
Callout Directory: /MyCalloutDir
SMTP Host:

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



Large Payload Directory: /tmp

Using Document Streams to Handle Large Payloads

Oracle B2B enables you to handle large payloads (whose size is specified by the user under Configuration settings) by using streams instead of loading them in-memory.

Earlier, when you specified the large payload size, Oracle B2B, instead of persisting the payload in the database, the payload is placed in the large payload directory. However, prior to streams processing being available, Oracle B2B would load the payload into memory to process the payload, which, for very large payloads, can cause the JVM to run out of Heap Space.

To avoid running out of JVM memory space in the heap for large payloads, Oracle B2B now supports stream-based processing of payloads.

Currently, the document types that are supported for stream-based processing are

- EDI X12
- EDI_EDIFACT
- HL7

Currently, the transport protocols supported for stream-based processing are

- File
- FTP

- SFTP
- MFT

Stream-based processing is particularly useful in the case of batching, where you have say, 10,000 payloads zipped in a 1 Gigabyte batch. Once the payloads are debatched, they put a good deal of load on the system.

How Streaming Works

From the transport perspective, File, FTP, SFTP, and MFT are the only protocols that are enabled to process the input payloads through streams once the large payload threshold is exceeded. This means if a payload received by the system through any of the preceding transports exceeds the large payload size, then the payload is not loaded in-memory; instead, the payload is sent to the file system (stored in the large payload directory). With HTTP transport, the payload is first loaded on to the memory, but during the time of persistence, the payload size is checked, and if the size is more than the large payload size, the payload is persisted in the file system and not in the database, but the payload will not be able to be processed with streams.

When a large payload is received, the Oracle B2B engine stores an input stream reference to the file, and does not load it to the memory. In the identification layer (where the trading partners or document types are identified), Oracle B2B identifies a payload as a large payload and it pushes the payload on to a stream rather than on to memory. In the processing layer, the same check is performed to identify whether a payload is a large payload, and Oracle B2B processes the payload as a stream.

However, if the payload is a non-EDI payload, such as an OAG payload, it will be loaded on to the memory. However, for EDI payloads, the payload is read from an input stream and is passed to an output stream during processing. The output stream is stored in a directory called the Processing directory. Typically, Oracle B2B creates a processing directory inside the large payload directory itself.

For example, Oracle B2B receives a 1 GB payload containing 10,000 batched messages (payloads) over a File channel, and the large payload size is specified as 2 MB. The File listening channel reads the payload and copies it to the large payload directory and creates a wire message that references the payload.

After the processing, the payload is then inserted into the business message and the application message, and is delivered to the back end; not all the 10,000 messages will be greater than 2 MB.

When the payload is inserted into the business message, a check is performed to identify if the payload is a large payload. If the payload is a large payload, then the payload file is copied from the processing directory, and an entry is created in the large payload directory. A reference to that entry is provided to the business message with a header identifying it as a large payload (large payload = true).

When you click that link reference in the business message, the payload is loaded from the respective location. Payloads that are lesser than the large payload size are copied on to the memory and then moved to the database. However, it may so happen that a payload whose size is less than the large payload size, after getting converted to XML, the size increases to more than the large payload size. In this case, the payload is not persisted in the database, and a file reference is passed on to the business message.

When you use streaming, B2B reads one transaction at a time and process it in memory until completion before loading the next transaction into memory. This enables the B2N engine to not run out of memory when processing EDI files that have a large number of transactions that is a large payload.

Taking the preceding example of 10,000 messages of less than 2 MB each, however, it is not possible to process the entire payload of 10,000 messages in a single transaction. You have to specify a partial commit size for the batch (say 200 messages) accordingly so that every time the partial commit size is reached, those 200 messages are committed to the database.

The limitation, if you try to process all the messages in a single transaction even if the JTA is set to a high value, is Oracle B2B will have to load all the 10000 into memory during delivery and persistence because they are small payloads, eventually causing a memory overflow.

When Does Streaming Not Occur

Streaming does not occur in the following circumstances:

- With HTTP transport, the payload is first loaded on to the memory, but during the time of persistence, the payload size is checked, and if the size is more than the large payload size, then the payload is persisted in the file system and not in the database, but the payload will not be able to be processed with streams.
- In the processing layer, when the check is performed to identify whether a payload is a large payload, and Oracle B2B processes the payload as a stream. In case of non-EDI/X12/HL7 payload, such as an OAG payload, it will be loaded on to the memory.

Using Document Streams from the Back End

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.

Note:

The behavior of this part of the setup from earlier releases is different. Earlier this directory was a plain directory, but from 12.1.3.0 is a treated as a stream store. The implication is that if you change the directory mid way during the processing, the application would previously work fine because the application maintained the entire file reference. However, these internally have been abstracted and the application now only maintains a Stream store id.

You need to migrate all payloads from 1 dir to another if there needs to be a change in location. You need to restart the server for the change in stream store location to take effect.

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"/>
```

> **Note:**
>
> If BPEL is not sending a large payload to Oracle B2B, this property should not be set.
>
> after 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

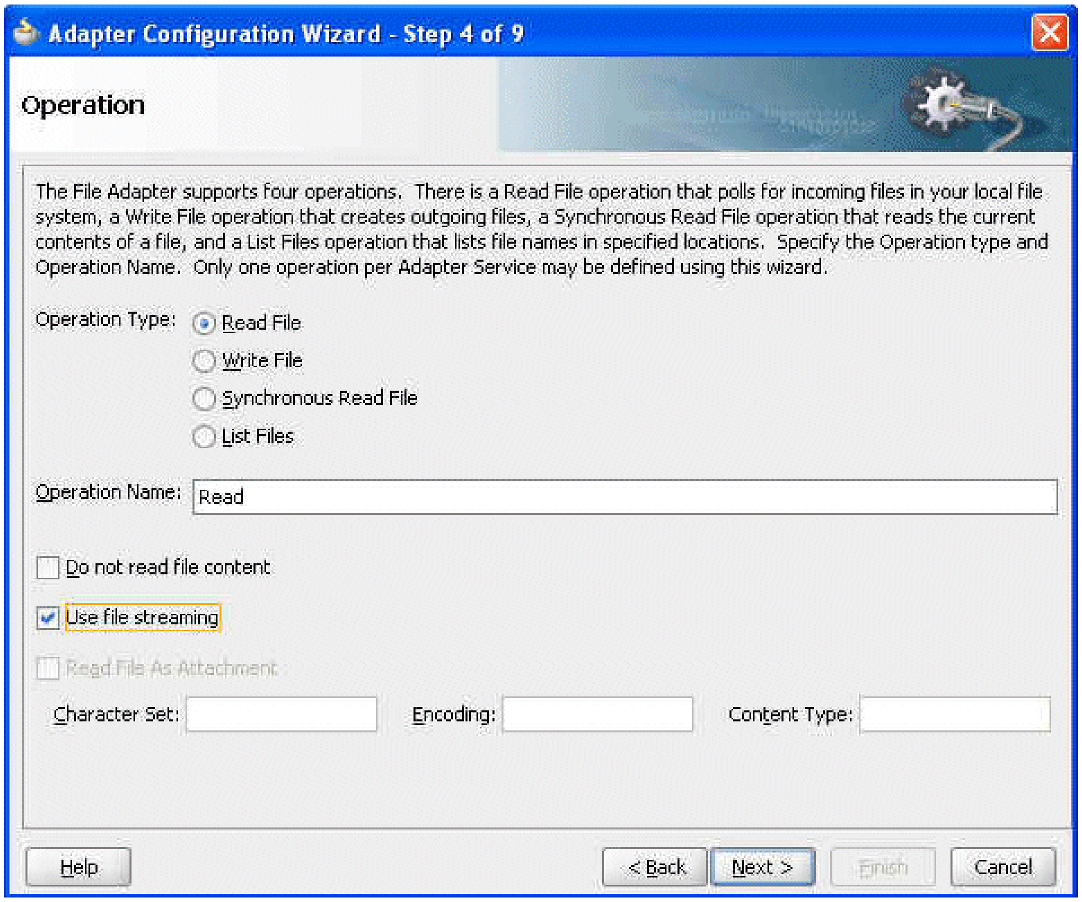
   ```
   eventName=LARGE_PAYLOAD=true
   ```

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


Adapter Configuration Wizard - Step 4 of 9

Operation

The File Adapter supports four operations. There is a Read File operation that polls for incoming files in your local file system, a Write File operation that creates outgoing files, a Synchronous Read File operation that reads the current contents of a file, and a List Files operation that lists file names in specified locations. Specify the Operation type and Operation Name. Only one operation per Adapter Service may be defined using this wizard.

Operation Type: [✓ Read File] Write File Synchronous Read File List Files

Operation Name: Read

☐ Do not read file content

☑ Use file streaming

☐ Read File As Attachment

Character Set: Encoding: Content Type:

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.

Limitations

The following limitations apply to payload handling and streaming:

- Only certain documentation types support streaming.
- Only certain transport types are supported.
- Functions such as XPatch extraction, tend to load the XML payloads into memory for extraction currently. (However, the impact tends to be less where there is batching, as the individual payloads are of lesser size and once the extraction is complete, the loaded references are cleared allowing for a immediate garbage collection,)
- Documents greater than 2 Gigabytes are not supported.
- When using HL7 custom delimiters, the native payload is loaded into memory.

B

Handling E-Mail Attachments

This appendix discusses how Oracle B2B handles payloads by using e-mail.

The appendix contains the following sections:

Sending and Receiving Payload as an E-Mail Attachment

For inbound messages, if the e-mail body is empty, and it has a part of an e-mail attachment, then the attachment part is be treated as the payload and accordingly processed For outbound messages, you can configure the delivery channel to send the payload as an e-mail attachment.To send the payload as an attachment, you configure the delivery channel by setting **Send as attachment** to `true`.
The default e-mail body ignore size is 5 characters. If the e-mail body contains more than 5 characters, then the body is treated as a payload.

The default size to ignore can be configured using the following Oracle B2B server property in Oracle Fusion Middleware Control console:

`b2b.DefaultBodySize=20` (otherwise, the default value is `5`).

Sending and Receiving Payload as E-Mail Body

In the case of outbound message processing, if you want to send the payload as the e-mail body, you can configure the delivery channel by setting **Send as attachment** to `false`.

In case of inbound, in this case, the e-mail body is be treated as the payload and is accordingly processed by Oracle B2B.

Sending and Receiving Payload as E-Mail Body Along with an Attachment

In the case of outbound message processing, if you want to send the payload as an e-mail body along with an e-mail attachment, you need to configure the delivery channel by setting **Send as attachment** to `false`, and specifying the attachment in the `Attachment` header at the back-end application.

Example B-1 shows a single attachment.

Example B-1 Single Attachment

```
<?xml version="1.0" encoding="UTF-8"?>
<Attachments xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="AttachmentDescriptor.xsd" boundary="boundary---->
  <AttachmentPart>
     <Location>file:///tmp/oralogo_small.gif</Location>
     <Content-Type>
```

```
          <Top-Level-Type>image</Top-Level-Type>
          <Sub-Type>jpeg</Sub-Type>
      </Content-Type>
     <Content-Transfer-Encoding>BASE64</Content-Transfer-Encoding>
     <Content-ID/>
     <Content-Description>A GIF file</Content-Description>
</AttachmentPart>
</Attachments>
```

In the case of inbound, in this case, the e-mail body is treated as the payload, and the attachment is stored in the configured attachment directory. The attachment directory can be configured by setting the following Oracle B2B server property in the Oracle Fusion Middleware Control console:

`b2b.attachments.dir=<`*`directory_name`*`>/`

Sending and Receiving Payload as E-Mail Body Along with Multiple Attachments

In the case of outbound message processing, if you want to send the payload as an e-mail body along with multiple e-mail attachments, you need to configure the delivery channel by setting **Send as attachment** to `false`, and specifying the attachment in the `Attachment` header at the back-end application.

Example B-2 shows multiple attachments.

Example B-2   Multiple Attachments

```
<Attachments xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
xs:noNamespaceSchemaLocation="AttachmentDescriptor.xsd" boundary="boundary----">
  <AttachmentPart>
     <Attachment>UjBsR09EbGhjZ0dTQUxNQUFBUUNBRU1tQ1p0dU1GUXhEUzhi</Attachment>
     <Content-Type>
        <Top-Level-Type>image</Top-Level-Type>
        <Sub-Type>jpeg</Sub-Type>
     </Content-Type>
     <Content-Transfer-Encoding>BASE64</Content-Transfer-Encoding>
     <Content-ID/>
     <Content-Description/>
  </AttachmentPart>
  <AttachmentPart>
      <Location>file:///tmp/SpreadSheet.xls</Location>
      <Content-Type>
        <Top-Level-Type>application</Top-Level-Type>
        <Sub-Type>vnd.ms-excel</Sub-Type>
      </Content-Type>
      <Content-Transfer-Encoding>BASE64</Content-Transfer-Encoding>
      <Content-ID>SpreadSheet-1</Content-ID>
      <Content-Description/>
  </AttachmentPart>
</Attachments>
```

In the case of inbound, in this case, the e-mail body is treated as the payload, and the attachments are stored in the configured attachment directory.

C

High Availability Architecture and Failover Considerations

This appendix provides an overview of the high availability characteristics of Oracle B2B.

This appendix contains the following sections:

- Overview
- Protection from Failures and Expected Behavior
- Cluster-Wide Configuration Changes

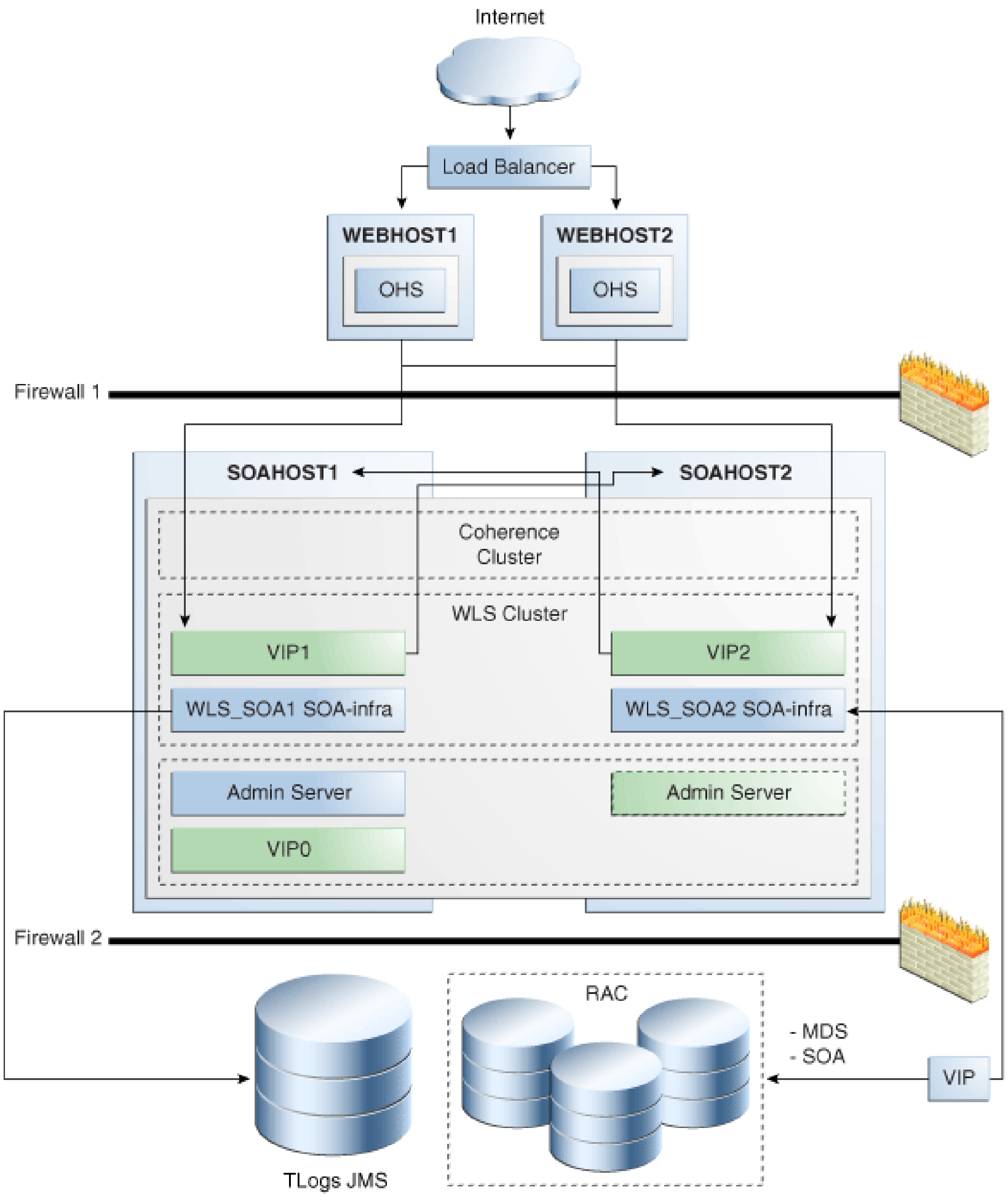
Overview

Oracle B2B is deployed as part of the Oracle SOA Service infrastructure composite application. Some high availability characteristics are specific to an Oracle B2B high availability deployment.

- The Oracle B2B user interface application runs inside each one of Oracle WebLogic Server servers, and as part of the same cluster as the Oracle SOA Service Infrastructure application.
- Oracle B2B's server maintains the partners, documents, and channels definitions in the SOA database using an Oracle RAC database.

The figure below describes a two-node Oracle SOA Service Infrastructure cluster running on two WebLogic Servers.

Figure C-1   Oracle SOA Service Infrastructure High Availability Architecture



Protection from Failures and Expected Behavior

This section describes how an Oracle B2B high availability cluster deployment protects components from failure. This section also describes expected behavior in the event of component failure.

Oracle B2B UI Failure

The Oracle B2B user interface application maintains some navigation information in memory. When a failure occurs in one of the managed servers running the Oracle B2B user interface, users' requests are redirected to another active WebLogic Server running the application. Ongoing requests from Oracle HTTP Server time out according the time out setting in Oracle HTTP Server's configuration. Failover requires those users accessing the failed instance to log in again, since the application is not enabled for serializing the session information.

The Oracle B2B user interface in-memory data is non-transactional, and some steps may need to be revisited in the case of failover.

Node Failure

If a node failure occurs, or if the local Oracle WebLogic Server Node Manager reaches the maximum restart tries on the failed managed server, whole server migration is triggered after the available server verifies the time stamp in the database leasing system. The other Oracle B2B engine remains available for processing new messages from the different channels. At the same time, the remaining Oracle B2B server should resume the pending operations for singleton channels, such as FTP, email, or file, after the default timeout period is reached in the time stamp the failed node sets in the database (two minutes by default). For Oracle B2B's User Interface application, ongoing requests from Oracle HTTP Server time out according to Oracle HTTP Server configuration.

Note:

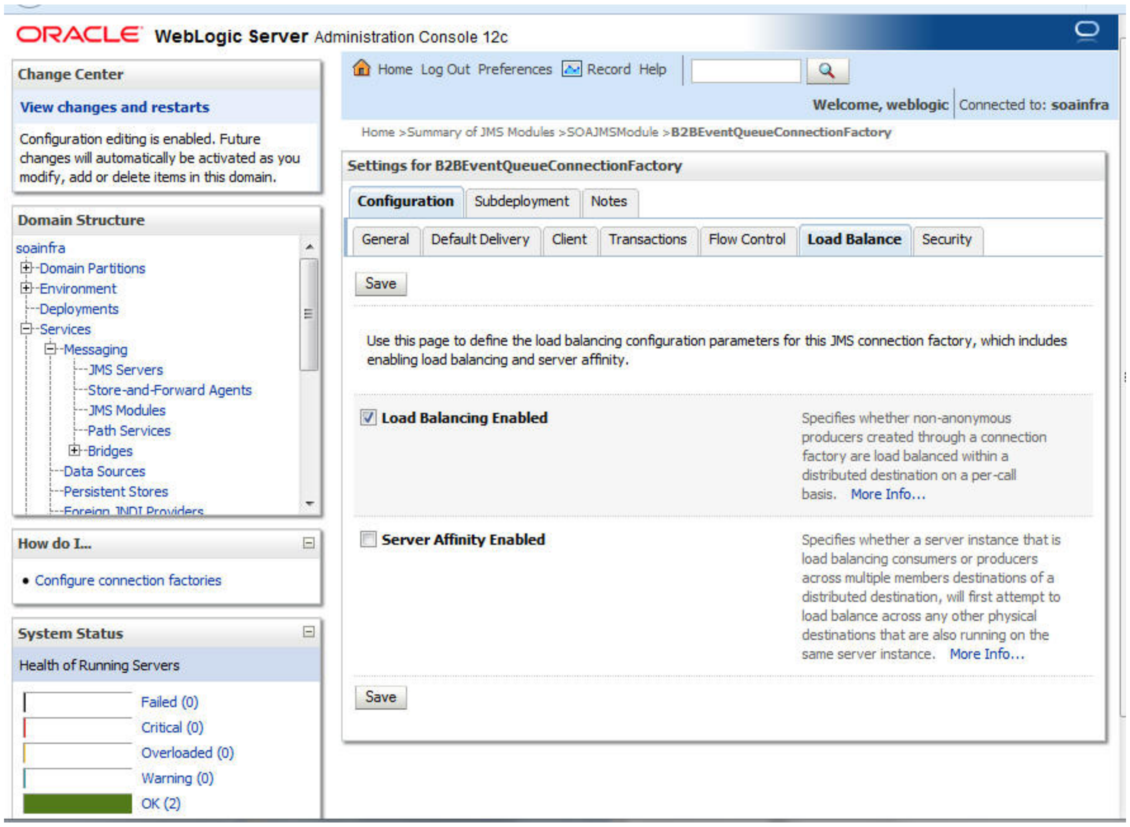
This SOA Suite feature is part of Oracle Integration Continuous Availability. Please refer to the *Oracle Fusion Middleware Licensing Information User’s Guide* for more details on Oracle SOA Suite for Middleware Options.

Cluster-Wide Configuration Changes

The standard Java EE artifacts that the Oracle B2B engine uses are configured as part of the Oracle WebLogic domain in which the Oracle SOA Service Infrastructure is installed. Oracle WebLogic Clusters provide automatic configuration synchronization for artifacts, such as data sources, persistent stores, and JMS modules across the WebLogic Server domain. At the same time, the WebLogic Server cluster controls synchronization of deployments and libraries used by Oracle B2B's engine.
All Oracle B2B server-specific configuration is maintained in the database, and configuration changes are applied to all the SOA Servers running in a WebLogic Server domain. Therefore, configuration properties, such as Payload Size, and Outbound Dispatcher Counts are applied cluster-wide, meaning they are used by all instances in the Oracle SOA cluster.

To set up File, FTP, or Email transports in a high availability environment, set b2b.HAInstance = true in FMW Control. See Setting B2B Configuration Properties in Fusion Middleware Control for more information. In addition, deselect **Server Affinity Enabled** in the **Load Balance** tab for the B2BEventQueueConnectionFactory in the SOAJMSModule.

Figure C-2   The Settings for B2BEventQueueConnectionFactory



ORACLE WebLogic Server Administration Console 12c

Home Log Out Preferences Record Help

Welcome, weblogic | Connected to: soainfra

Home >Summary of JMS Modules >SOAJMSModule >B2BEventQueueConnectionFactory

Change Center

View changes and restarts

Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Domain Structure

- soainfra
  - Domain Partitions
  - Environment
  - Deployments
  - Services
    - Messaging
      - JMS Servers
      - Store-and-Forward Agents
      - JMS Modules
      - Path Services
      - Bridges
    - Data Sources
    - Persistent Stores
    - Foreign JNDI Providers

How do I...

- Configure connection factories

System Status

Health of Running Servers

Failed (0)
Critical (0)
Overloaded (0)
Warning (0)
OK (2)

Settings for B2BEventQueueConnectionFactory

Configuration | Subdeployment | Notes

General | Default Delivery | Client | Transactions | Flow Control | Load Balance | Security

Save

Use this page to define the load balancing configuration parameters for this JMS connection factory, which includes enabling load balancing and server affinity.

✓ Load Balancing Enabled	Specifies whether non-anonymous producers created through a connection factory are load balanced within a distributed destination on a per-call basis. More Info...
☐ Server Affinity Enabled	Specifies whether a server instance that is load balancing consumers or producers across multiple members destinations of a distributed destination, will first attempt to load balance across any other physical destinations that are also running on the same server instance. More Info...

Save

D

Diagnosing Generic Issues

This appendix discusses how to diagnose and troubleshoot basic issues in Oracle B2B.

Generic Diagnostics

When you encounter any error in sending or receiving messages by using Oracle B2B, you can diagnose the root cause of the error.

- Identifying the Error in Oracle B2B Console
- Examining the Log Files and Composites in Fusion Middleware Enterprise Manager Control Console

Identifying the Error in Oracle B2B Console

Studying error details and identifying the type of errors by using Oracle B2B is the first step to diagnose issues.

Identifying the Error Type

Errors can be of the following types:

- Connection error
- Document identification error
- Document validation error
- Agreement identification error

Table D-1 lists the common error types.

Table D-1 Types of Errors

Error Type	Error Message	Possible Cause
Connection	`Transport error: [IPT_HttpSendConnectionRefused`	Outbound Case: Delivery Channels: Trading Partner is not up or Trading Partner URL is not correct.
Document Identification	`Document protocol identification error`	Inbound Case (depends on protocol): • Incorrect XPath, Positional file, Document type, or filename Outbound Case: • B2B parameters not set properly • Incorrect XML: Internal-properties
Document Validation	`General Validation Error`	Depends on the document protocol.

Table D-1 (Cont.) Types of Errors

Error Type	Error Message	Possible Cause
Agreement Identification	`Trading partner agreement not found for the given input values`	Inbound or Outbound Case: • No agreement found for Partner and Document combination • Wrong direction in agreement

Identifying the Location of the Error

After you identify the error type, you need to find the location of the error.

Errors can occur at the following levels:

- Trading Partner: If you have sent a message, but did not receive an ACK from the trading partner, something might have gone wrong at the partner side.
- B2B: At the Oracle B2B end, it could be a Control Number issue (duplicate Control Number), a validation issue, or an identification issue. Identification could be document identification or agreement identification.
- Composite: At the composite end, it can be a routing issue where the msg was received by a wrong composite. In addition, it can be a validation issue, XSLT issue, or even the endpoint may be down, so there is a communication failure with the application.
- Application: At the application level, it can be an issue with incorrect documents or authorization, or a data validation error in the application.

Checking the Message Flow

You can also check the message flow and view the error report to figure out the error details. The message flow can be from:

- From the trading partner:

  Wire > Business > Application
- From Middleware:

  Application > Business > Wire

Sample Diagnosis

Consider a situation where an agreement between two trading partners is not found leading to an error.

In this case, do the following checks:

- Check the order of identification:
  - Partner: First identify the partners between whom the message is getting exchanged
  - Document: Identify the document types and document revision of the message
  - Agreement: Identify the agreement based on the partners and the documents exchanged
  - Document Validation: Check whether the document is valid for the agreement
- Check from the trading partner side:

  - Check the identifiers. Identifier types enable Oracle B2B to identify a trading partner at runtime. 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.
  - Check whether the correct document identification method has been used.
- Check from the middleware side:
  - Check whether the correct B2B parameters related to agreements are set properly in Oracle Fusion Middleware Enterprise Manager Control console.

Examining the Log Files and Composites in Fusion Middleware Enterprise Manager Control Console

Apart from the Oracle B2B console, you can diagnose issues with message exchange by using the Oracle Fusion Middleware Enterprise Manager Control console.

Note:

All errors that are created are queued up in the B2B exception queue by default. You can also monitor that queue. If there is an error handling framework built around the error queue, then you can receive error notifications rather than checking the console for any errors that might have occurred. This might be covered in the exception handling section so you can point them there as well. For additional related information, see Using a Custom Exception Queue for Error Message Delivery and other sections in Exception Handling.

To view diagnostic logs for Oracle B2B, you need to set the log levels by using the Enterprise Manager Control console.

See Configuring "Oracle B2B Logging Mode" in *Oracle Fusion Middleware Administering Oracle SOA Suite and Oracle BPM Suite*.

Note:

For more information about log files and the level and type of logging information to write to a log file, see *Oracle Fusion Middleware Administrator's Guide*.

To view Oracle B2B related log files:

1. Open Oracle Fusion Middleware Enterprise Manager Control console.
2. Navigate to **SOA** > <*SOA Domain name*>.
3. Right-click <SOA Domain name> and select **Logs** > **View Log Messages**.
4. Click **Target Log Files** on the right section.
5. Select <Managed_Server name>-diagnostic.log and click **View Log File**.

You can also check the deployed composite and perform a flow tracing for any error that may have occurred.

To check the composite and perform flow tracing:

1. Open Oracle Fusion Middleware Enterprise Manager Control console.
2. Navigate to the deployed composite under **SOA** > <*SOA Domain name*>.
3. Click the Flow Instances tab and search for an instance.
4. Click the relevant Flow ID link and view the flow trace.

Displaying the Actual Thread ID in the Log File

Thread IDs can be used to trace the execution of each thread in the diagnostic log. You can use them to associate each trace line with its thread ID. By default, the log file contains the thread name, which can be generic, such as Workmanager.

You can display the actual thread ID in the log file by changing the configuration of the handlers in the `logging.xml` file. Use the ODLHandler properties `useThreadName` and `useRealThreadId` to control logging of thread name or ID. Setting `useThreadName` to false stops the logging of the thread name. Setting `useRealThreadId` to true adds the actual thread ID to the log.

This is a generic configuration:

```
<log_handler name='odl-handler'
class='oracle.core.ojdl.logging.ODLHandlerFactory'>
     <property name="path" value="..."/>
     ....
     <property name="useThreadName" value="false"/>
     <property name="useRealThreadId" value="true"/>
</log_handler>
```

This is an example configuration:

```
$Domain/config/fmwconfig/servers/soa_server1/logging.xml
   <property name='useThreadName' value='false'/>
   <property name='useRealThreadId' value='true'/>
```

Note that the location of the logging.xml file can vary slightly, depending on the version you are using.

- For 12.1.3: `oracle_common/modules/oracle.odl_12.1.3/server_config/logging.xml`
- For 12.2.1: `oracle_common/modules/oracle.odl/server_config/logging.xml`

E

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 includes the following sections:

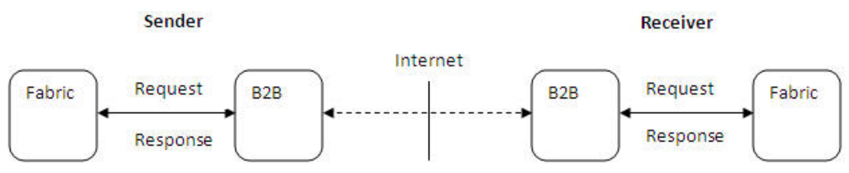
- Introduction
- Configuring Sender
- Configuring Receiver
- Resubmitting Messages

Introduction

Oracle B2B provides synchronous request/response between from Fabric to Oracle B2B. However, only HTTP transport is supported for the initial release.

Figure E-1 shows the end-to-end scenario.

Figure E-1   Synchronous Request/Reply



Configuring Sender

To configure the sender, you need to perform the following tasks.

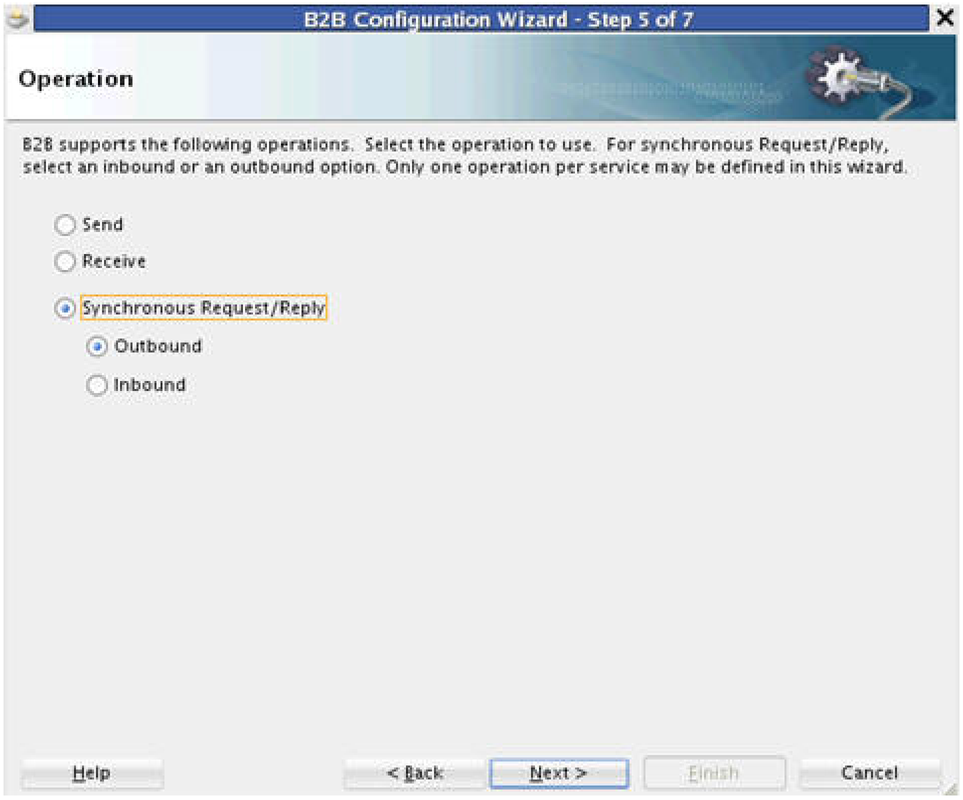
- "Configuring Oracle JDeveloper"
- "Configuring Oracle B2B"

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 E-2.

Figure E-2 B2B Configuration Wizard: Operation Page



B2B supports the following operations. Select the operation to use. For synchronous Request/Reply, select an inbound or an outbound option. Only one operation per service may be defined in this wizard.

Send

Receive

[✓ Synchronous Request/Reply]

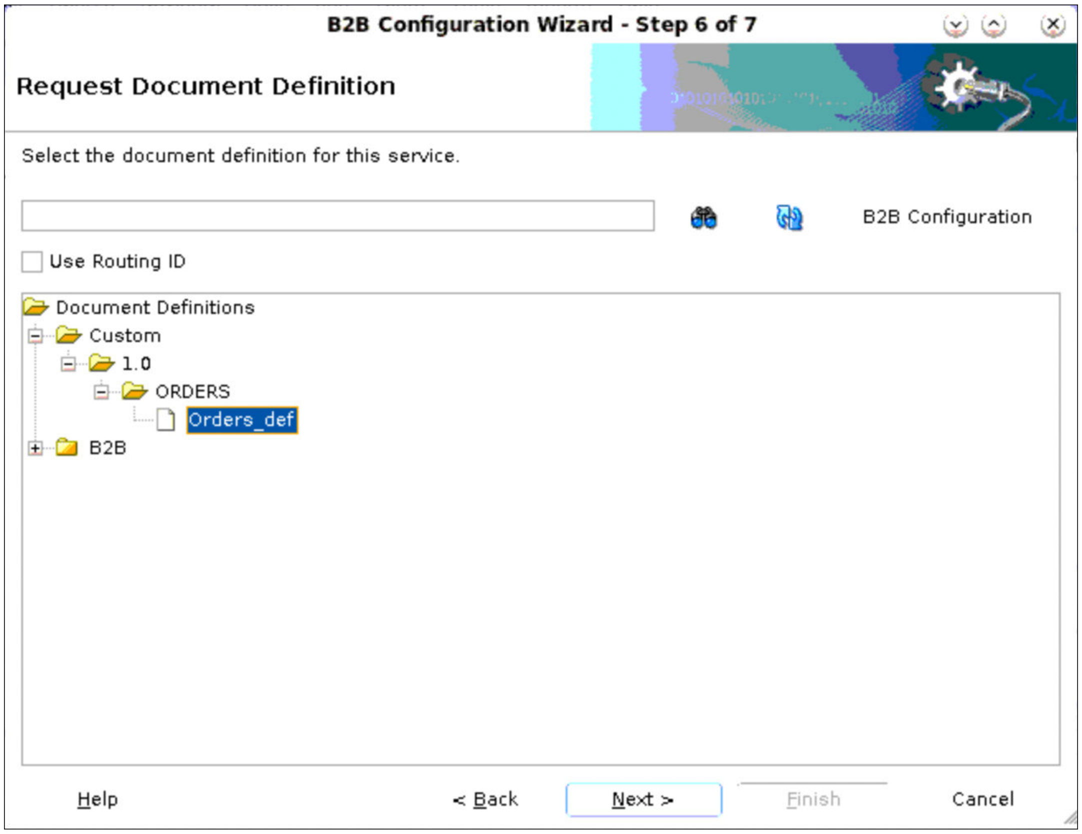
[✓ Outbound]

Inbound

- 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 E-3.

Figure E-3   B2B Configuration Wizard: Document Definition Page


B2B Configuration Wizard - Step 6 of 7

Request Document Definition

Select the document definition for this service.

B2B Configuration

Use Routing ID

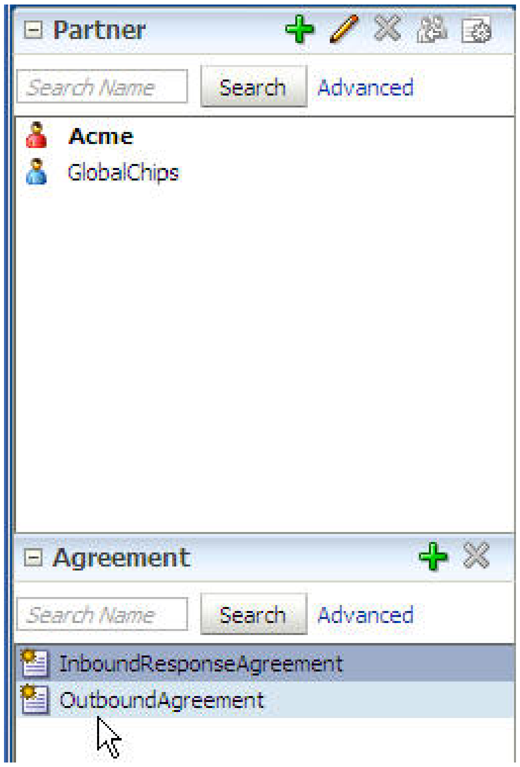
```
Document Definitions
  Custom
    1.0
      ORDERS
        Orders_def
  B2B
```

Configuring Oracle B2B

In Oracle B2B console, you need to:

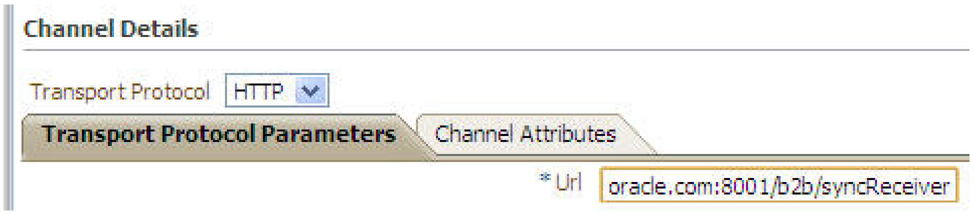
- **Set up Oracle B2B agreements:** You need to define two agreements. One agreement is required to send the Request, and the other agreement is needed to receive the Response as shown in Figure E-4. You do not need to set up any Oracle B2B property for this. Based on the call from the Fabric to Oracle B2B, the message type "sync request" is determined.

Figure E-4 Setting Up Oracle B2B Agreements



- **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 E-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 E-5 Creating a Trading Partner Channel



Channel Details

Transport Protocol: HTTP

* Url: oracle.com:8001/b2b/syncReceiver

Configuring Receiver

To configure the receiver, you need to perform the following tasks.

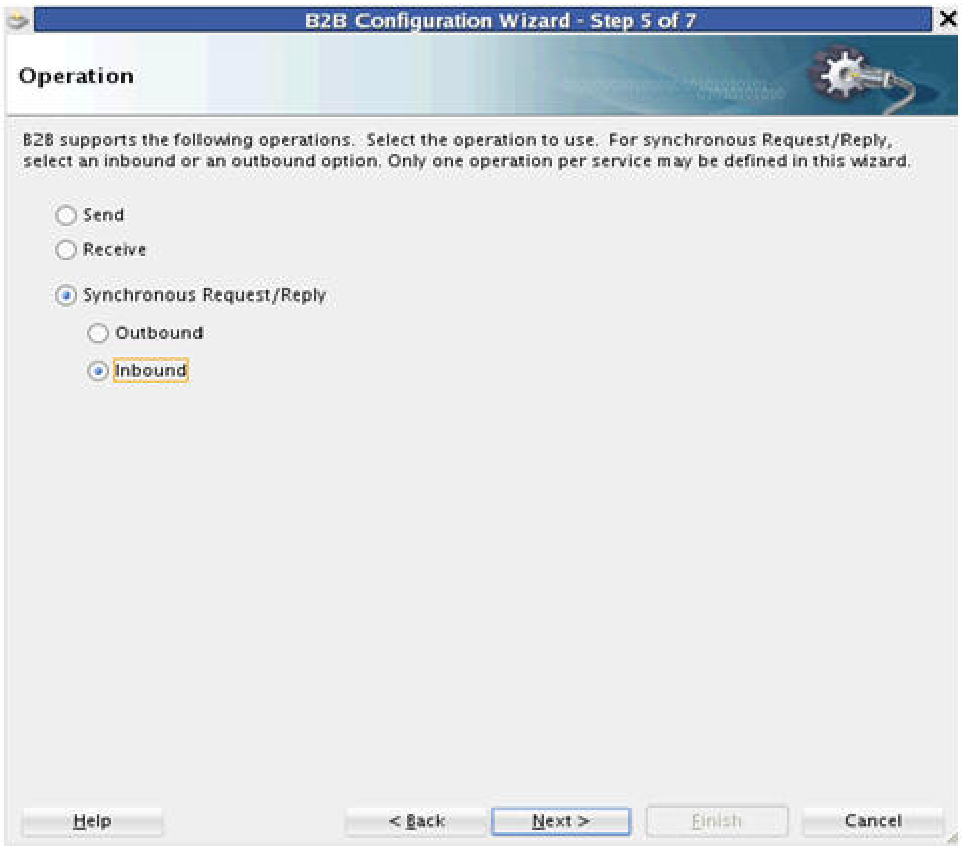
- "Configuring Oracle JDeveloper"
- "Configuring Oracle B2B"

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 E-6.

Figure E-6   B2B Configuration Wizard: Operation Page



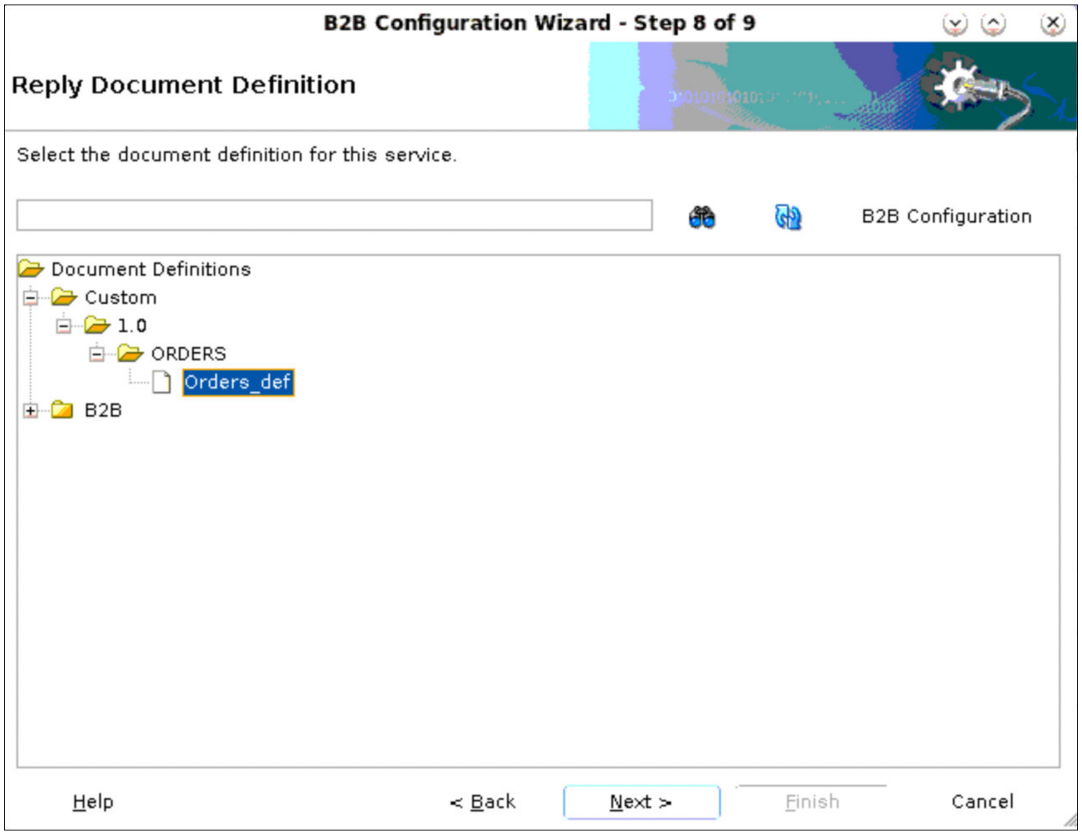
B2B supports the following operations. Select the operation to use. For synchronous Request/Reply, select an inbound or an outbound option. Only one operation per service may be defined in this wizard.

- ○ Send
- ○ Receive
- ● Synchronous Request/Reply
  - ○ Outbound
  - ● Inbound

- 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 E-7.

Figure E-7   B2B Configuration Wizard: Document Definition Page


B2B Configuration Wizard - Step 8 of 9

Reply Document Definition

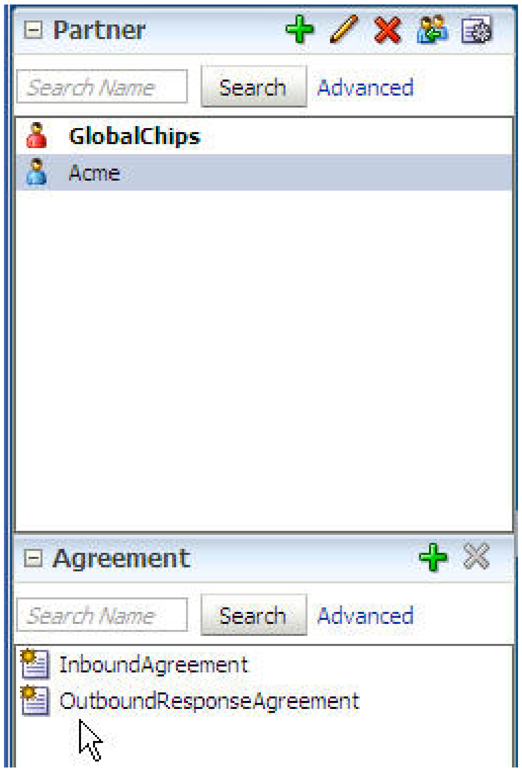
Select the document definition for this service.

Configuring Oracle B2B

In Oracle B2B console, you need to:

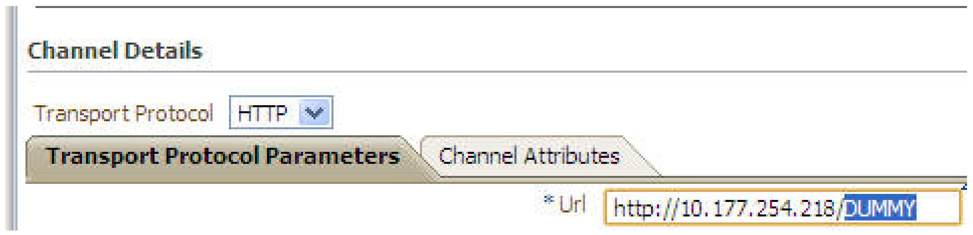
- **Set up Oracle B2B agreements:** You need to define two agreements. One agreement is required to receive the Request, and the other agreement is needed to send the Response as shown in Figure E-8. You do not need to set up any Oracle B2B property for this, but you do need to configure the optional B2B headers, such as From, To, DocVersion and DocType. The synchronous HTTP receiver decides the message type as "sync request".

Figure E-8   Setting Up Oracle B2B Agreements



- **Create a dummy Trading Partner channel:** You need to create a dummy remote Trading Partner Channel, but this channel is not used at runtime. You can specify any URL value, but the value needs to be in the format `http://`*`host`*`:`*`port`*`/xyz` as shown in Figure E-9.

Figure E-9   Creating a Dummy Trading Partner Channel



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 asynchronous 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.

F

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 Configuring B2B System Parameters for more information. It also discusses how to set B2B properties by using the `configmbeanutil` utility.
The appendix includes the following sections:

- Properties To Set in Fusion Middleware Control
- Using the configmbeanutil Utility

Properties to Set in Fusion Middleware Control

The section lists the properties that can be set in Fusion Middleware control.

The properties in Table F-1 can be set in Oracle Enterprise Manager Fusion Middleware Control. See Configuring Oracle B2B Server Properties for how to set the properties.

> **Note:**
>
> Restarting the SOA Server is required for changes to B2B properties.

Table F-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 F-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`, which is the default.
	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.
	This property needs to be set to `false` to enable processing of duplicate RosettaNet messages.
	All document protocols can use this feature. Note that this checking is based on business message ID only. An example on how to reproduce this is using the file name format in this PDF document, on page 6:
	http://www.oracle.com/technetwork/middleware/b2b-integrations/learnmore/tnb2b11g002-326857.pdf
	Inbound to Oracle B2B (Request): sending a message
	Format:
	`%TO_PARTY%_`
	`%DOCTYPE_NAME%_`
	`%DOCTYPE_REVISION%_`
	`%MSG_TYPE%_`
	`%MSG_ID%`
	Filename example: `SalesInc_850_4010_1_1234.dat`
	The business message ID is `1234`. The first file consumed is fine. If the second file with the same name is used, B2B will mark the message state with `MSG_ERROR` if `b2b.checkDuplicate` is set to true; this is for inbound messages only. The property checking is not applicable to inbound wire message resubmitted from the UI or the command line utility.

`b2b.rejectDuplicateMessage`	When this property is set to `true`, duplicate messages with same message ID are not allowed in a batch. If such messages are sent, Oracle B2B changes the message ID of messages and sets the messages to `ERROR` state. Also, in the Reports page, the following error is reported: `"Duplicate message with same message id found in a batch. Changing original message id <org_msg_id> to new message id <org_msg_id+timestamp) and rejecting the message."` The default value of this property is `false`.
`b2b.useJMSDataSourceCache`	Set this property to determine whether JMS objects or SOA datasource information is cached. The value of the property is either `true` or `false`. By default, the value is `true`. If the property is set to `true`, then Oracle B2B looks up for the JMS objects or SOA datasource in the cache. If there is a cache hit, the cached value is returned, otherwise the lookup value is returned. This property may be used for incremental performance improvement.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.additionalDebugInfo`	The value is `LogDir=<directory>`, where the directory is a location to write the value of the `b2b.batchMonitorStragglerBatchName` and `b2b.batchMonitorStragglerTimeLag` in the files. The `BatchStragglerKeySet_<unique_guid>.dat` file contains the value of the `b2b.batchMonitorStragglerBatchName` and `b2b.batchMonitorStragglerTimeLag`. The `BatchStragglerKeySet_<unique_guid>.dat` file contains the key name of the `b2b.batchMonitorStragglerBatchName` during the batch processing time. These files are created only if you specify a value for this property. To disable logging, remove this property. **Note:** In case of an incorrect directory name or a file creation failed in the given directory, then no file is written. As long as the this property is defined, the information is still logged in the server diagnostic as information.
`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.ebMS.ProtMsgId.prependHost`	When this property is set to `true`, then Protocol Message ID and Protocol Collaboration ID values are prefixed with the Host Trading Partner name. By default, this property is set to `false`. For example: When `b2b.ebMS.ProtMsgId.prependHost` is `false`, the values will be: • Protocol Message Id: `@0A261B4D13F33830161000006AB9B3F8` • Protocol Collaboration Id: `@0A261B4D13F33830161000006AB9B3F8` When `b2b.ebMS.ProtMsgId.prependHost` is `true`, the values will be: • Protocol Message Id: `INDIGO@0A261B4D13F33830161000006AB9B3F8` • Protocol Collaboration Id: `INDIGO@0A261B4D13F33830161000006AB9B3F8` wherein `INDIGO` is the Host Trading Partner name.
`b2b.mdsCache.minutesToLive`	Set this property to `0` to specify that the MDS cache would be stored in memory forever. By default, the value for this property is 5, which means that the cache would be removed from memory after 5 minutes, if the objects are not used. You can set the value of the property to be `0` or any other non-zero value.
`b2b.encoding`	This property can be used to specify encoding other than default UTF-8.
`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.transportCache`	Set this property to `true` to enable the transport cache to cache the transport message objects between the transport layer and the engine.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`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 a 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`.
`b2b.ignoreTPWithAsterisk`	In cases where the B2B UI is configured with a generic identifier as a client IP address in the partner profile and the partner has provided a range of IP addresses in the form of a mask (for example, `120.12.23.*`) from which they can send a document, then B2B cannot recognize the sender and fails with an Agreement Not Found Error. Set this boolean property to `true` to enable B2B to find the agreement. The default value for this property is `false`.
`b2b.fabricRetryCount`	This property specifies how many times retry of a failed message to fabric is attempted. For fabric, retry count is a global property and for non-fabric it is a channel retry count.
`b2b.fabricRetryInterval`	This property specifies the number of seconds between attempts to retry delivery of a message to fabric.
`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 10. • 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.showEncryptedData`	For an encrypted RosettaNet message being exchanged, set this property to `true` for viewing Packed Message contents.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description

`b2b.threads.LogStatusInterval`
`b2b.threads.LogStatusLevel`
	Use this feature to log the number of active inbound and outbound threads from the available pool of threads during a periodic interval. Oracle B2B publishes event thread information such as name, last active status at periodic intervals to DMS metrics and diagnostic logs. By default, thread information are published at a 30 minute interval and the log level is `DEBUG`.
	Users can change the interval and log level using these properties.
	Possible values for `b2b.threads.LogStatusLevel` are `DEBUG`, `INFORMATION`, `WARNING`, and `ERROR`.
	For the feature to work:
	• When `b2b.threads.LogStatusLevel = DEBUG`, the server log level needs to be set to `TRACE`.
	When `b2b.threads.LogStatusLevel = INFORMATION`, then the server log level needs to be set to `NOTIFICATION`.
	Sample Output:
	`<Mar 3, 2011 11:37:01 PM PST> <Error> <oracle.soa.b2b.engine> <BEA-000000>`
	`<B2B Thread Name :`
	`weblogic.work.j2ee.J2EEWorkManager$WorkWithListener@f6e5fd8 - Direction :`
	`Default - Status : ACTIVE>`

`b2b.addAllDocParams`	Set this property to `false` to stop adding unwanted or unset parameters in the import zip.
`b2b.useDefaultQuery`	Set this property to `false` to make the value of Receive Time Stamp blank and Send Time Stamp to be automatically filled. The default value is `true`.
`b2b.refreshCache`	Set this property to `true` indicate if there must be a retry by refreshing the cache in case of a failure (only for EBMS.) The property is required only in a cluster environment when the cache synchronization is not set up. The default value is `false`.
`b2b.commitTxnOnMsgDelivery`	Set this property to `true` to indicate whether the database transaction message should be (intermediate) committed while delivering the message to the back end. If this parameter is not set, the message is still delivered to the back end, but the database records are not committed.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description

`b2b.inboundThreadCount`
`b2b.outboundThreadCount`
`b2b.defaultThreadCount`
`b2b.inboundSleepTime`
`b2b.outboundSleepTime`
`b2b.defaultSleepTime`
	Set these properties to set the number of threads and thread sleep time to improve message processing.
	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.
	If `b2b.inboundThreadCount` and `b2b.outboundThreadCount` are set, then `b2b.defaultThreadCount` sets the number of threads to process events other than inbound messages or outbound messages. For example, events such as auto retry, manual resubmit, trading partner agreement deployment, batch messages together, and send batch out events are processed by `b2b.defaultThreadCount` threads. If `b2b.inboundThreadCount` and `b2b.outboundThreadCount` are *not* set, then `b2b.defaultThreadCount` threads will process all the events including inbound messages or outbound messages.
	The `b2b.inboundSleepTime`, `b2b.outboundSleepTime`, and `b2b.defaultSleepTime` 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.
`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.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`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. after 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.payloadObfuscation`	To turn on payload obfuscation, set this property to `true`. For more information, see Payload Obfuscation.
`b2b.setDynamicNameSpace`	To use EDI `ecs` and `xsd` files from Oracle B2B 10*g*, set this property to `true`. When using EDI `ecs` and `xsd` files in Oracle B2B 11g that 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`.
`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`	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`.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`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 back end 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.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.optimizeStorage`	Use this property to determine whether the storage of payloads would be optimized (shared) across Business, Wire, and Application instances, if applicable. Setting this property accordingly reduces the I/O load at the database layer due to persistence of payloads. You need to restart the server for applying any changes to this parameter. The valid values for the parameter are `true` and `false` (default).

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`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 property. The valid value for this parameter is `Integer`.
`b2b.fromHeaderName`	If the HTTP From message header contains a value that can be mapped to an IP Address, then Oracle B2B converts the value to an IP Address. If the remote Trading Partner identifiers, specifically, Generic Identifier, does not contain that IP Address, then Oracle B2B fails to identify that the message came from which Trading Partner. Use this property to specify a value for the header name so that when a message arrives, Oracle B2B does an exact match (no IP Address conversion) of the value extracted from the From message header with the Generic Identifier specified for the remote Trading Partner. The value of this property is case-sensitive. So, the header must match exactly with the value that is provided in the message. The value for this property takes the following format: `<header name>#<header name2>...` The value can contain multiple header names separated by `#`, such as `sender_id#receiver_id`.

`b2b.auditFileLocationb2b.maxNumAuditFile`	Use `b2b.auditFileLocation` to retrieve the EM URL for Composite Flow Trace . The property takes a string value.
	Example:
	`b2b.auditFileLocation=/tmp/audit.log`
	Use `b2b.maxNumAuditFile` to specify the maximum number of audit logs. The default value for this property is `10`.
	Format: `audit-<managed server name>.log.<n>o` where `n=0,1,2,3`
	Example:
	`b2b.maxNumAuditFile= 10`
	The audit file is stored in the CSV format.

`b2b.TreatRNIFasRNDocument`	Use this property to identify a RosettaNet document. If you set this property to `true` and the document is received through RNExchangePlugin, then the document is identified successfully. If you set this property to `false`, then the document is identified based on the XPath expression and its value (just like a custom document.)

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description

`b2b.mdnAliasesAndAlgorithm`	If an inbound AS2 message has requested the sign MDN, the outbound MDN needs to be signed. However, B2B does not have information about the private key that is needed to sign the MDN nor does it have the algorithm used. This is solved by using this property to handle signing of negative MDN for AS2 messages. The property takes its value in the following format:
	`B2BHost=<default alias to sign>:<default algorithm>;<TP1>=<alias1>:<alg1>;...<TPn>=<aliasn>:<algn>`
	Example:
	`B2BHost=b2bs0:md5;ACME=stg_2010:md5;GLOBALCHIPS=stage2014:sha1`
	This property only affects the AS2 Exchange protocol. The managed server does not need to be re-started for the property to take effect.

`b2b.fromTPXPath`	Use this property to include XPath from where the Trading Partner name has to be extracted from the payload. When a message comes, Oracle B2B uses the XPath specified in the Enterprise Management console to extract the Trading Partner name from the payload, and then Oracle B2B compares that value to the value specified as a Generic Identifier to determine which Trading Partner has sent the message. **Note:** Do *not* include `/text()` at the end of the XPath, otherwise Oracle B2B will not find the path.
`b2b.correlateResponse`	Use this property to toggle correlation in Oracle B2B. The default value for this property is `true`. If you set this property to `false`, Oracle B2B does not to correlate the outbound enqueued message with an existing message having the ID that is specified in the **replyToMsgID** field.
`b2b.protMsgIdAsBmId`	Set this property to `true` enable Oracle B2B to send the protocol `MsgID` to the back end application. The default value for this property is `false`.
`b2b.sqldumpTimeRange`	Set this property to get the time range for SQL dumps. This property is used during the Diagnostic Framework implementation. The time is specified in minutes.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.flowTraceEMURL`	When Oracle B2B is hosted in one domain and the Oracle SOA composite (Oracle B2B adapter with the JMS option) is deployed in a different domain (Oracle Enterprise Manager Fusion Middleware Control in a different domain), Oracle B2B needs a mechanism to provide the link to the domain where the composite is deployed. Use this property to provide the domain URL details of the Oracle SOA composite for tracking the instance message flow based on the ECID. The format for setting the value of this property is: `http://<host>:<port>#<domain_name>#<domain_type>` You can also specify this URL details at the Delivery Channel level by using the Oracle B2B console. This URL is available as a **Flow Trace** link for individual messages in the Oracle B2B Application Message reports. While creating the Flow Trace link, Oracle B2B first checks if the value is provided at the channel level. If the value is present, it is used to construct the URL link. If the value is not present, Oracle B2B checks whether `b2b.flowTraceEMURL` is set in Oracle Enterprise Manager Fusion Middleware Control. If it is set, then the property value is used to construct the URL link. If both values are not set, by default, Oracle B2B assumes that the composite is available in the local domain and constructs the URL link accordingly. See Configuring Oracle B2B Logging Mode in *Administering Oracle SOA Suite and Oracle Business Process Management Suite*.
`b2b.b2bReportsURL`	Set this property (along with `b2b.flowTraceEMURL`) on the domain where Oracle Enterprise Manager Fusion Middleware Control is hosted to construct the Flow Trace URL link in the reports of the Oracle B2B console. The format for setting the value of this property is: `http://<host>:<port>` **Note:** Currently, Oracle SOA Suite supports tracking of multiple SOA domains and one Oracle B2B instance. This is because this property is set at the global level.
`b2b.TPAbasedebMSIdValidateAndOverride`	This property enables: • Trading partner Identifier sent from Middleware or Trading partner is validated against the configured Identifiers in the Agreement for both inbound and outbound flow. • Override the Identifiers configured in the Agreement using the Identifier sent from back-end application or from trading partner. The default value is `false`. If `b2b.TPAbasedebMSIdValidateAndOverride= true`: • Agreement is configured with Identifier ID1, ID2 and inbound message has Identifier ID2, then ID2 would be sent to the back-end application. **Note:** This applies to outbound messages. • Agreement is configured with Identifier ID1, ID2 and inbound message has Identifier ID3, even though the trading partner is configured with Identifier ID3, there is a failure in Agreement Identification. • For any outbound message without a specific Identifier type, Name identifier is used in ebMS message header. If `b2b.TPAbasedebMSIdValidateAndOverride= false`: The Identification is done against the configured Identifier in the profile, and the configured Identifier in the Agreement is used in ebMS headers.
`b2b.resubmitOutboundAck`	Set this property to `true` to enable resubmission of OUTBOUND ACK. The default value is `false`.

Table F-1 (Cont.) Oracle B2B Properties in Oracle Enterprise Manager Fusion Middleware Control

Property	Description
`b2b.defaultCustomDocTypeVersion`	When the payload comes in as inbound over AS2 delivery channel, B2B tries to identify the document protocol which is not possible because there is no identifier in the payload to look at. The payload can be any flat file or binary file payload. Set this to binary to define the protocol.
`b2b.defaultCustomDocType`	When the payload comes in as inbound over AS2 delivery channel, B2B tries to identify the document protocol which is not possible because there is no identifier in the payload to look at. The payload can be any flat file or binary file payload. Set this to generic to define the protocol.
`b2b.httpsTrustStore`	If this property is set to true, then B2B uses the default .jks file to send for the https delivery channel.

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

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 F-1   To Print All Properties to the Console

```
java -cp $MW_HOME/soa/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME
/wlserver/server/lib/wljmxclient.jar:$MW_HOME/oracle_common/modules/
glassfish.el_1.2.0.0_2-2.jar oracle.tip.b2b.utility.ConfigMBeanUtility
```

Example F-2   To Add a Property

```
java -cp $MW_HOME/soa/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME
/wlserver/server/lib/wljmxclient.jar:$MW_HOME/oracle_common/modules/
glassfish.el_1.2.0.0_2-2.jar oracle.tip.b2b.utility.ConfigMBeanUtility
add b2b.test cool ok
```

Example F-3   To Update a Property

```
java -cp $MW_HOME/soa/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME
/wlserver/server/lib/wljmxclient.jar:$MW_HOME/oracle_common/modules/
glassfish.el_1.2.0.0_2-2.jar oracle.tip.b2b.utility.ConfigMBeanUtility
update b2b.test thru
```

Example F-4   To Remove a Property

```
java -cp $MW_HOME/soa/soa/modules/oracle.soa.b2b_11.1.1/b2b.jar:$MW_HOME
/wlserver/server/lib/wljmxclient.jar:$MW_HOME/oracle_common/modules/
glassfish.el_1.2.0.0_2-2.jar oracle.tip.b2b.utility.ConfigMBeanUtility
remove b2b.test
```

Example F-1, Example F-2, Example F-3, and Example F-4 show uses for this utility.

G

Back-End Applications Interface

This appendix lists the message properties supported by Oracle B2B.

The appendix includes the following sections:

- Mapping B2B IP_MESSAGE_TYPE to SCA Normalized Message Properties
- Normalized Message Properties

Mapping B2B IP_MESSAGE_TYPE to SCA Normalized Message Properties

This table maps the `B2B IP_MESSAGE_TYPE` to SCA normalized message properties.

Table G-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	-
OVERRIDE_URI	-	OVERRIDE_URI

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.

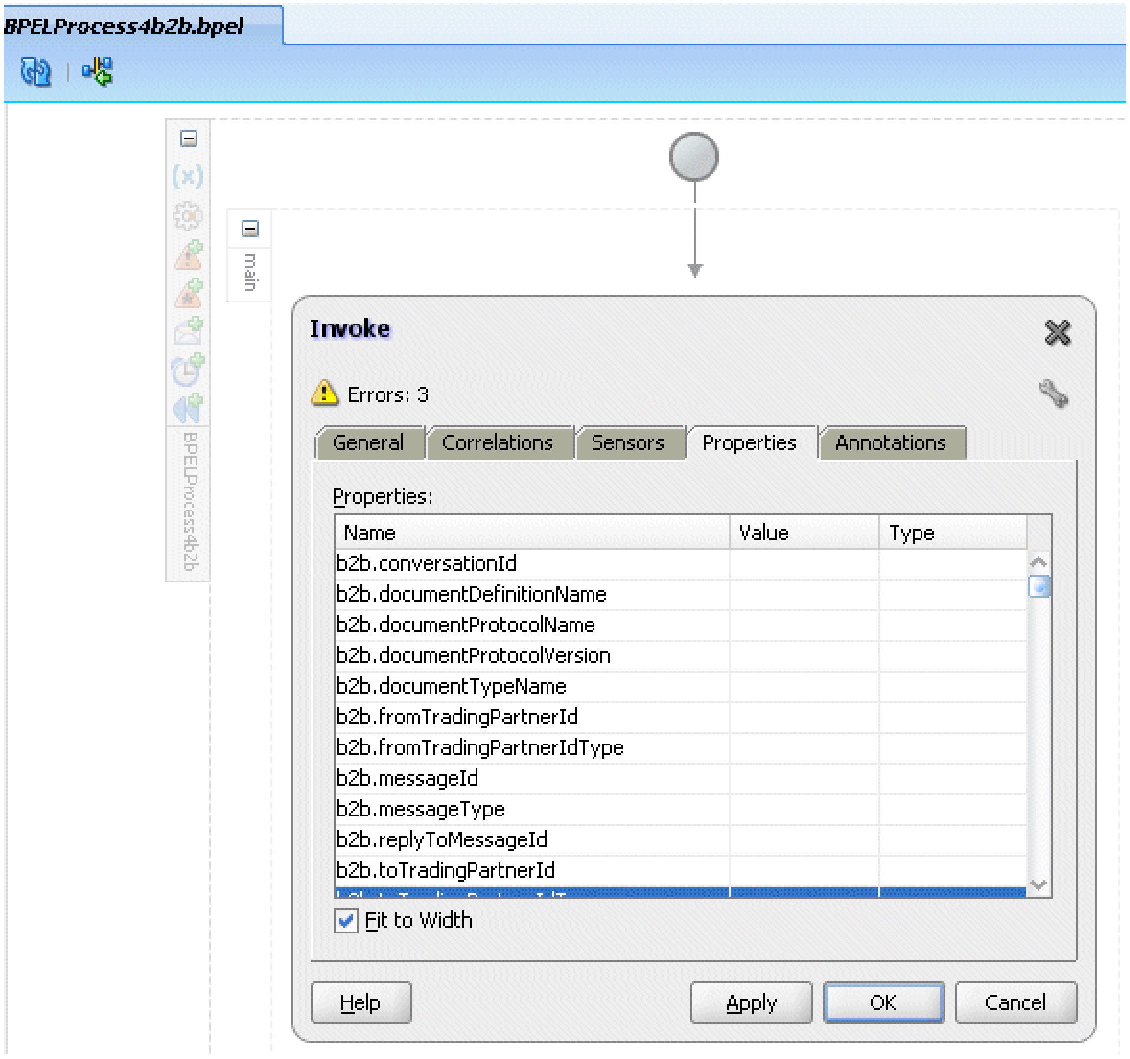
Manipulating headers in design time is simplified by using predetermined complex properties. In B2B, you can manipulate headers with reserved key words. However, some properties are dynamically generated based on your input. These definitions are not predetermined and hence cannot be accounted for in 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.

Note:

For B2B to identify the correct agreement, the properties `b2b.fromTradingPartnerId`, `b2b.toTradingPartnerId`, `b2b.documentProtocolVersion` and `b2b.documentTypeName` need to be added.

Figure G-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 G-1   Invoke Activity Showing B2B Normalized Message Properties



Properties:

Name	Value	Type
b2b.conversationId		
b2b.documentDefinitionName		
b2b.documentProtocolName		
b2b.documentProtocolVersion		
b2b.documentTypeName		
b2b.fromTradingPartnerId		
b2b.fromTradingPartnerIdType		
b2b.messageId		
b2b.messageType		
b2b.replyToMessageId		
b2b.toTradingPartnerId		

☑ Fit to Width

Table G-2 lists the predetermined properties of a normalized message for Oracle B2B.

Table G-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 X12 document
`b2b.documentProtocolName`	No	Both	String	-	The document protocol, for example, X12 for an EDI X12 document

Table G-2 (Cont.) Properties for Oracle B2B

Property Name	Propagable (Yes/No)	Direction (Inbound / Outbound)	Data Type	Range of Valid Values	Description
`b2b.documentProtocolVersion`	No	Both	String	-	The document version, for example, 4010 for an EDI X12 document
`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.
`b2b.overrideURI`			String		Tenant specific connection information for where the message is to be posted instead of the connection information defined in the Delivery Channel associated with the agreement.

H

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:

- Why Do You Need to Manage Sequenced Messages
- Ant Commands and Public APIs

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

Ant Commands and Public APIs

Oracle B2B provides Ant command line 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.String target, 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
```

I

Tracking Business Message Flow

This appendix discusses the process of tracking and managing the flow of business messages between an Oracle SOA composite and Oracle B2B.

The appendix contains the following sections:

Introduction to Business Flow Events

A business flow involves one or more messages being exchanged.

A typical business flow involves:

- A single message exchange such as a purchase order
- Multiple message exchanges such as request/response
- A separate notification flow in case of any error
- An additional flow in case of retry/resubmit

Different types of business flows, such as error, retry, and request/response, can be distinguished by an ID called flow events or FlowID.

What You Need to Know About Flow Events?

A flow event contains the following information:

- FlowID, which uniquely identifies the underlying Business Flow.
- ECID, which uniquely identifies a synchronous subflow associated with the underlying Business Flow. A Business Flow can possibly include many synchronous and asynchronous subflows, and therefore can comprise Events associated with different ECIDs. An asynchronous subflow can be created when a BPEL process is dehydrated and then re-hydrated.
- InstanceID, which uniquely identifies the service, reference, or component instance associated with the event.
- SCAEntityID, which uniquely identifies the type of the service, reference, or component instance referenced by the InstanceID. The SCAEntityID corresponds to values of the ID column stored in the `sca_entity` table. Using SCAEntityID is much less verbose than the alternative information such as:
  - Domain name associated with the service, reference, or component instance described by the event
  - Name of the composite that contains the service, reference, or component instance described by the event

  - Revision of the composite that contains the service, reference, or component instance described by the event
  - Label for the Composite that contains the service, reference, or component instance described by the event
  - Name of the service, reference, or component instance described by the event
- FlowEventID, which is a unique ID associated with the flow event. It is used to build the parent or child relationship between flow events.
- ParentFlowEventID, which uniquely identifies the parent of the flow event under consideration. This is the FlowEventID value for the parent flow event. If the flow event under consideration does not have any parent, the Parent FlowEventID is set to null.
- TransactionID, which identifies the transaction associated with flow event under consideration, or null if the flow event under consideration is not associated with any transaction.
- CorrelatedFlowID, which uniquely identifies the business flow that needs to be correlated with the current business flow.
- FaultID, which corresponds to the Fault associated with flow event under consideration. This ID is associated with a common fault that is stored in the `common_fault` table. The value is set to null if no Fault is associated with flow event under consideration.
- ExceptionTrace, which corresponds to the Exception stack trace for the Fault associated with flow event under consideration. The value is set to null if no Fault is associated with flow event under consideration, or if no stack trace could be obtained.
- FaultPolicyAction, which represents the triggered Fault policy action.
- ParentAuditEventID, which uniquely identifies the component-specific AuditEvent, which is the parent event (occurred prior) of flow event under consideration. This is used to stitch together events from different Audit Traces.

Note:

In the case of an Oracle SOA composite calling Oracle B2B, the correlationFlowId is retrieved from the Oracle SOA composite itself. However, in the case of Oracle B2B calling Oracle SOA, Oracle B2B need to generate the correlationFlowId and provide the ID to the composite for integration with the common instance tracking framework.

Tracking Oracle B2B Message Events

This section discusses how instance tracking is handled in the case of the Retry, Batching, and Resubmit message events.

Retry

In the case of channel level or document level retry, the FlowID is related or associated to the parent message's FlowID.

Batching

In the case of batching, the FlowEventID is associated with every individual message. Even though there is only one Wire Message in this case, the same will be shown for all the individual messages.

In the case of de-batching, the batch case is identified and it is updated for every individual message with a new FlowEvent ID after de-batch operation. With this, the error-related scenarios of a particular message in batch can also be updated. However, typically, during the inbound processing, the FlowID is be generated at the entry point (during Wire Message creation).

Resubmit

Resubmit is a manual process. So, a new FlowID is created and associated with the original message.

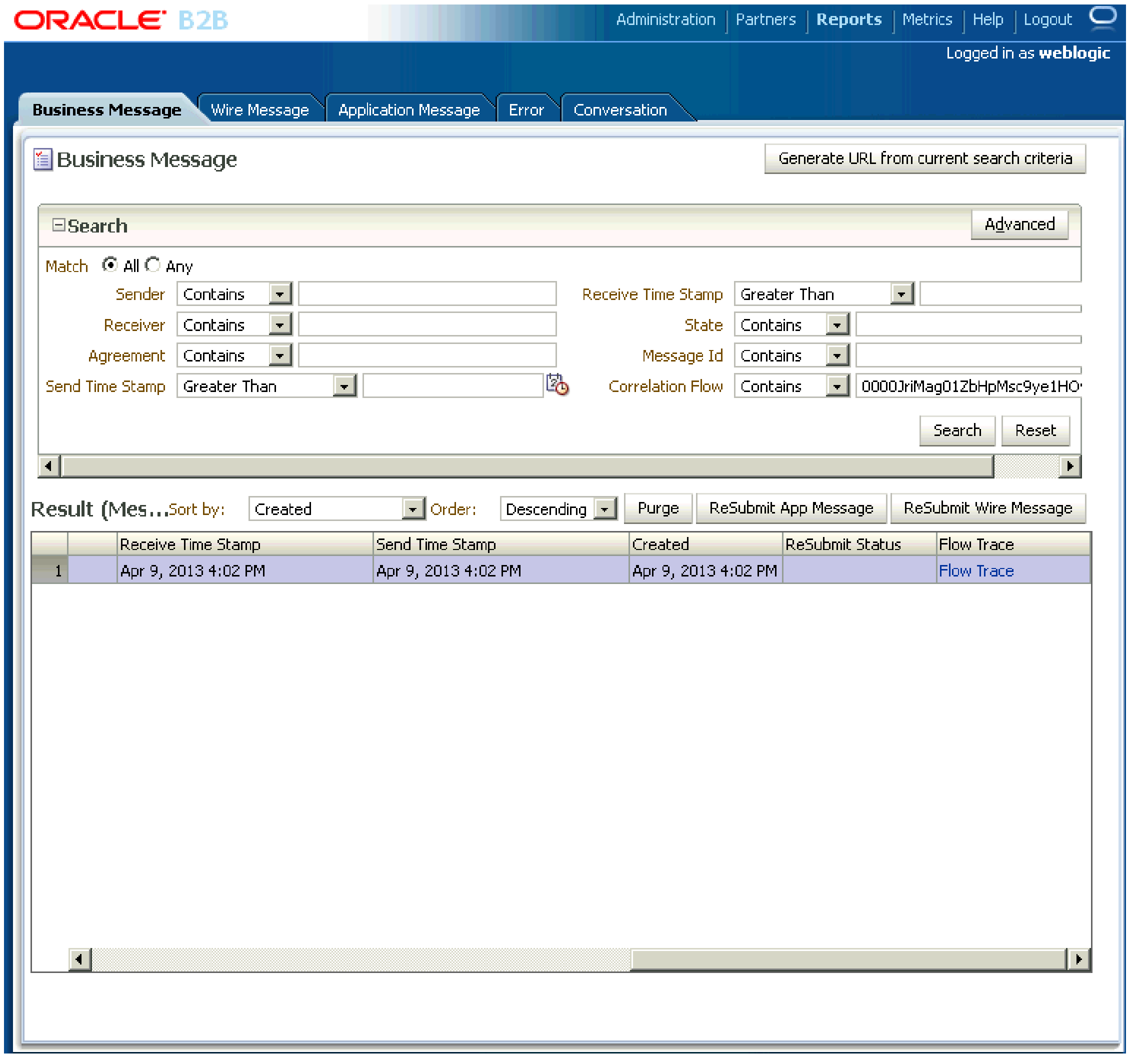
- Outbound Application Message - A new FlowID is created and stored in the `EXT_BUSINESS_MESSAGE` table.
- Inbound Application Message - A new FlowID is created and stored in the `EXT_BUSINESS_MESSAGE` table. The default approach is to track the FlowID created in Wire Message and send it all the way to back end. However, because this is a resubmit of the inbound Application Message, a new FlowID is created and the parent FlowEventID stores the value of original message.
- Outbound Wire Message - In this case, a new FlowID is created that stores the original message reference. The default approach is that a FlowID is created during the Application Message creation during outbound process. However, in this case, the new FlowID is created at the Wire Message creation level.
- Inbound Wire Message - A new FlowID is created and stored in the `B2B_WIRE_MESSAGE` table.

Tracing Flow By Using Oracle B2B Console

You can view the flow trace for a business flow event by clicking the Flow Trace link in a Business Report in the Oracle B2B console.

This is displayed in Figure I-1.

Figure I-1   The Flow Trace Link in Oracle B2B Business Message Report


ORACLE B2B
Administration | Partners | Reports | Metrics | Help | Logout
Logged in as weblogic
Business Message | Wire Message | Application Message | Error | Conversation
Business Message
Generate URL from current search criteria
Search
Advanced
Search
Reset
Purge
ReSubmit App Message
ReSubmit Wire Message

Match [✓ All] Any

Sender	Contains		Receive Time Stamp	Greater Than	
Receiver	Contains		State	Contains	
Agreement	Contains		Message Id	Contains	
Send Time Stamp	Greater Than		Correlation Flow	Contains	0000JriMag01ZbHpMsc9ye1HO

Result (Mes... Sort by: Created Order: Descending

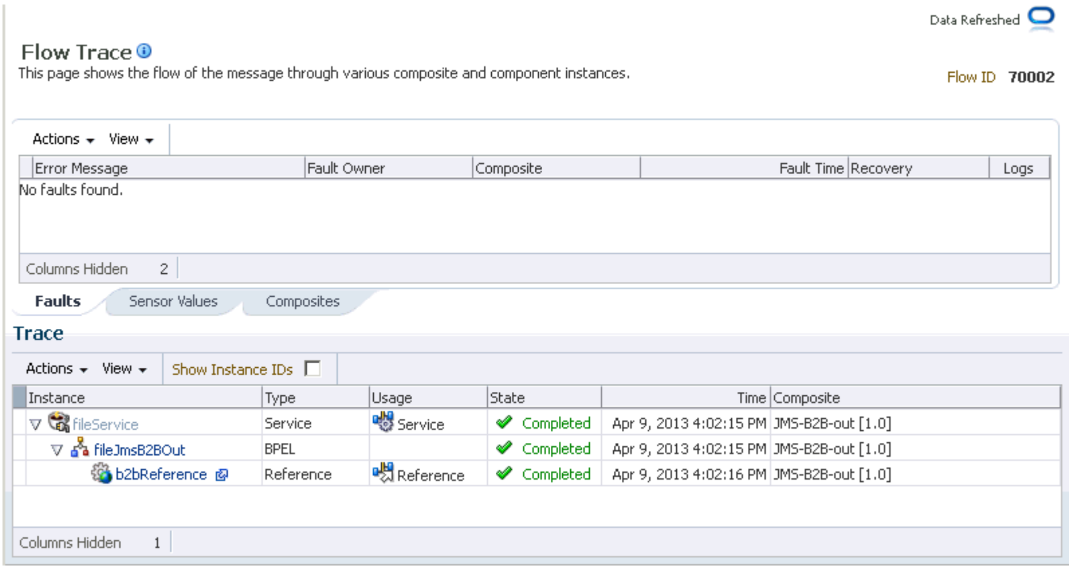
	Receive Time Stamp	Send Time Stamp	Created	ReSubmit Status	Flow Trace
1	Apr 9, 2013 4:02 PM	Apr 9, 2013 4:02 PM	Apr 9, 2013 4:02 PM		Flow Trace

When you click the link, you will be redirected to the Oracle Fusion Middleware Enterprise Management control console, where you can view the Flow Trace as displayed in Figure I-2.

Note:

In a B2B High Availability environment, the Enterprise Management SOA Composite Flow Trace link the B2B Reports does not work.

Figure I-2   The Flow Trace Window



Flow Trace

This page shows the flow of the message through various composite and component instances.

Data Refreshed

Flow ID 70002

Error Message	Fault Owner	Composite	Fault Time	Recovery	Logs
No faults found.					

Columns Hidden 2

Faults | Sensor Values | Composites

Trace

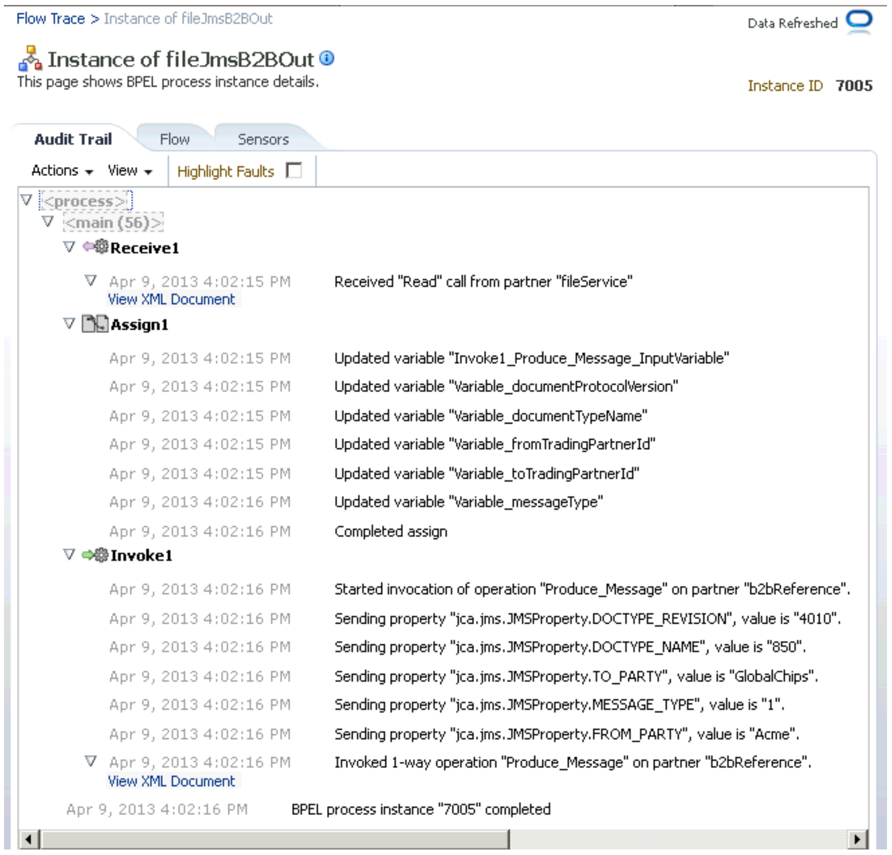
Show Instance IDs

Instance	Type	Usage	State	Time	Composite
fileService	Service	Service	Completed	Apr 9, 2013 4:02:15 PM	JMS-B2B-out [1.0]
fileJmsB2BOut	BPEL		Completed	Apr 9, 2013 4:02:15 PM	JMS-B2B-out [1.0]
b2bReference	Reference	Reference	Completed	Apr 9, 2013 4:02:16 PM	JMS-B2B-out [1.0]

Columns Hidden 1

You can click the BPEL component link in the Flow Trace window to view the detailed trace as displayed in Figure I-3.

Figure I-3   Detailed Flow Trace Report



Flow Trace > Instance of fileJmsB2BOut

Instance of fileJmsB2BOut

This page shows BPEL process instance details.

Data Refreshed

Instance ID 7005

Audit Trail | Flow | Sensors

Highlight Faults

```
<process>
  <main (56)>
    Receive1
      Apr 9, 2013 4:02:15 PM    Received "Read" call from partner "fileService"
      View XML Document
    Assign1
      Apr 9, 2013 4:02:15 PM    Updated variable "Invoke1_Produce_Message_InputVariable"
      Apr 9, 2013 4:02:15 PM    Updated variable "Variable_documentProtocolVersion"
      Apr 9, 2013 4:02:15 PM    Updated variable "Variable_documentTypeName"
      Apr 9, 2013 4:02:15 PM    Updated variable "Variable_fromTradingPartnerId"
      Apr 9, 2013 4:02:15 PM    Updated variable "Variable_toTradingPartnerId"
      Apr 9, 2013 4:02:16 PM    Updated variable "Variable_messageType"
      Apr 9, 2013 4:02:16 PM    Completed assign
    Invoke1
      Apr 9, 2013 4:02:16 PM    Started invocation of operation "Produce_Message" on partner "b2bReference".
      Apr 9, 2013 4:02:16 PM    Sending property "jca.jms.JMSProperty.DOCTYPE_REVISION", value is "4010".
      Apr 9, 2013 4:02:16 PM    Sending property "jca.jms.JMSProperty.DOCTYPE_NAME", value is "850".
      Apr 9, 2013 4:02:16 PM    Sending property "jca.jms.JMSProperty.TO_PARTY", value is "GlobalChips".
      Apr 9, 2013 4:02:16 PM    Sending property "jca.jms.JMSProperty.MESSAGE_TYPE", value is "1".
      Apr 9, 2013 4:02:16 PM    Sending property "jca.jms.JMSProperty.FROM_PARTY", value is "Acme".
      Apr 9, 2013 4:02:16 PM    Invoked 1-way operation "Produce_Message" on partner "b2bReference".
      View XML Document
  Apr 9, 2013 4:02:16 PM    BPEL process instance "7005" completed
```

Note:

Instance tracking is not supported for AQ in the current release. Instance tracking is supported for only in-memory and JMS back end.

Instance Tracking and Error Hospital Integration

Instance tracking and error hospital tracking functionality includes two parts.

They are described in sections Tracking Messages Between the Oracle Enterprise Manager Fusion Middleware Control Flow Trace and the B2B/Healthcare Console and Tracking the State of a Message from the Oracle Enterprise Manager Fusion Middleware Control Flow Trace XML.

Tracking Messages Between the Oracle Enterprise Manager Fusion Middleware Control Flow Trace and the B2B/Healthcare Console

It is possible to track messages between the Oracle Enterprise Manager Fusion Middleware Control flow trace and the B2B/Healthcare console. This functionality is available for both the JMS and in-memory integration modes.

If you intend to track instances and errors across domains for JMS integration, you must continue to use the following properties that are available for ECID-based JMS integration:

- `b2b.flowTraceEMURL` (or specify the same at the JMS channel level)

  Use this property to specify the information about the domain that Oracle Enterprise Manager Fusion Middleware Control consumes and uses to send JMS messages from the queue.
- `b2b.b2bReportsURL`

  Configure this property on the Oracle Enterprise Manager Fusion Middleware Control instance to point to the Oracle B2B URL.
- `hc.hcReportsURL`

  Configure this property to point to the SOA Suite for healthcare URL on the instance where the Oracle Enterprise Manager Fusion Middleware Control is running.

These properties are described in more detail in Setting B2B Configuration Properties in Fusion Middleware Control.

Tracking the State of a Message from the Oracle Enterprise Manager Fusion Middleware Control Flow Trace XML

It is possible to track the complete state of the message from Oracle Enterprise Manager Fusion Middleware Control through the flow trace XML. This functionality is available only for in-memory integration.

In general, the composite instance state follows the application message in B2B. As soon as the application message is marked complete, the corresponding composite instance state is updated as complete.

The existing error notification mechanism (sending an exception message to the exception composite/JMS queue) continues to function normally. Whenever possible the notification message is associated with the same flow ID of the original business message.

Inbound Messages

If an inbound document and a composite are deployed to accept the document, there are two types of failure that can be reported.

- **Failure before the document is identified**: The document itself is not identified and composite detection is not possible, therefore no fault reporting occurs.
- **Failure after the document is identified**: The composite is detected and a fault instance is reported in the composite. If an exception composite deployed, the exception composite instance is part of the same flow ID as the reported fault.

Outbound Non-Batch Messages

The following apply to outbound non-batch messages:

- The message remains in the "running" state until the message is delivered to the remote TP.
- Upon successful delivery, the composite instance is marked as being in the "complete" state.
- If an exception occurs during outbound processing, the composite is marked as being in a "faulted" state with the recovery set to `B2B_RECOVERY_REQUIRED`. The user is expected to recover the message in the B2B/Healthcare console. Note that the recovery state is same for both B2B and healthcare errors.
- The following cases are not handled properly and cause the state to be incorrect in the composite:
  - Negative `Acks` that are received on B2B mark the message as in "error". However, this does not update the composite state back to "error".
  - `Ack` time outs are not tracked. The B2B message itself is in the "error" state, but the composite remains in the "complete" state.
  - Resubmittal of a completed message results in an error. In this case the flow instance state shows up as "complete" despite B2B showing it as an error.
  - To track errors in batching, the user must rely on an exception composite to tap into the exception notifications sent to the back-end. The exception message delivered to the back-end is processed as a part of the same flow ID.

Outbound Batch Messages

The following apply to outbound batch messages:

- As soon as the batched message is staged within B2B and the message is inserted into the pending message table, the composite instance is marked "complete". Any exceptions that occur beyond this state during the batching process do not affect the state of the original composite.
- The user must rely on an exception composite to tap into the exception notifications sent to the back-end to track batching errors.

J

Setting Up B2B Communication By Using Remote JNDI Queue

This appendix discusses the process of 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:

- Configuring B2B Communication By Using Foreign JNDI

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:

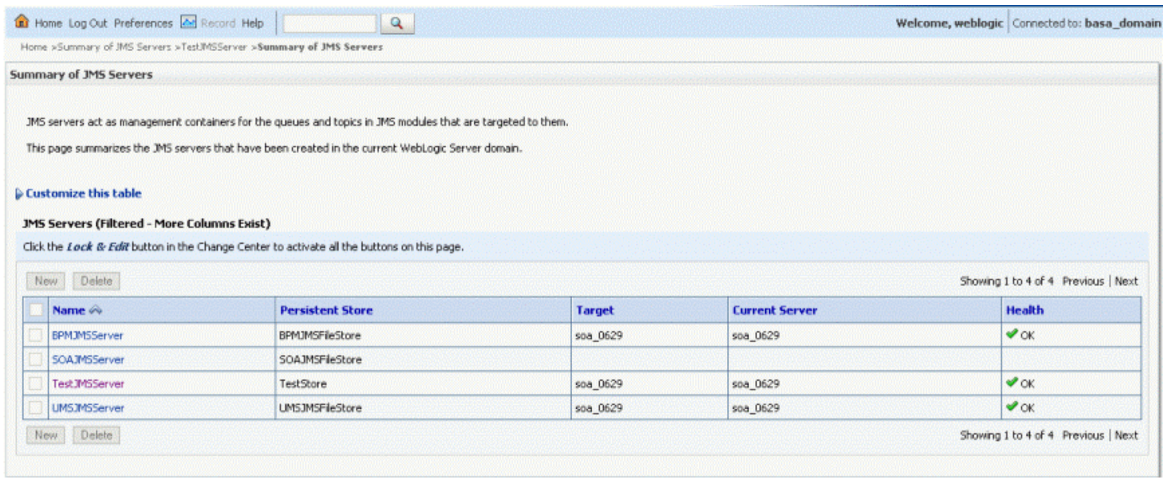
1. Creating the target JMS server, module, and queue.
2. Setting up an environment for B2B to write to the JMS queue.

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 J-1.

Figure J-1 Creating a Test JMS Server


Home Log Out Preferences Record Help
Welcome, weblogic | Connected to: basa_domain
Home >Summary of JMS Servers >TestJMSServer >Summary of JMS Servers

Summary of JMS Servers

JMS servers act as management containers for the queues and topics in JMS modules that are targeted to them.

This page summarizes the JMS servers that have been created in the current WebLogic Server domain.

JMS Servers (Filtered - More Columns Exist)

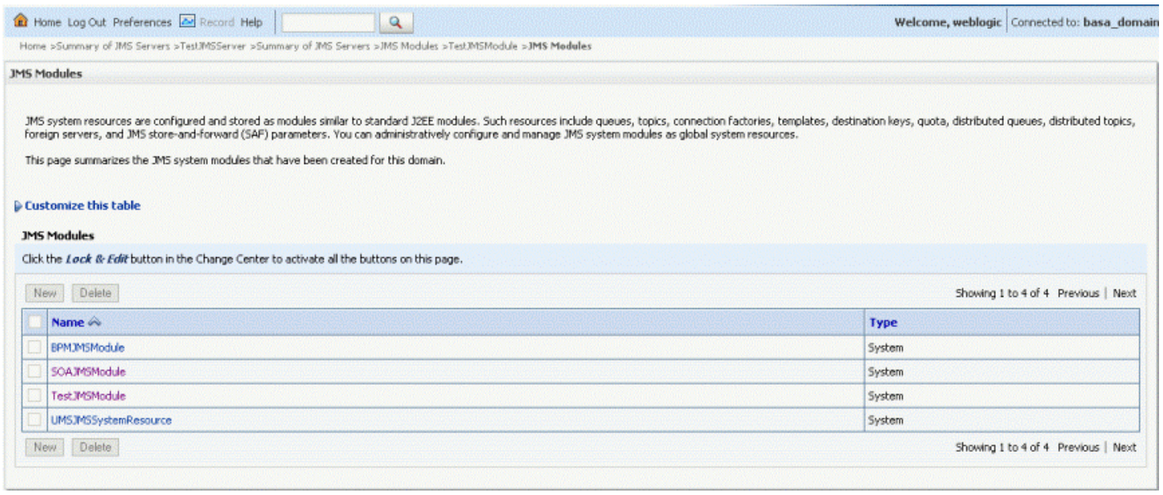
Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

New Delete
Showing 1 to 4 of 4 Previous | Next

Name	Persistent Store	Target	Current Server	Health
BPMJMSServer	BPMJMSFileStore	soa_0629	soa_0629	✓ OK
SOAJMSServer	SOAJMSFileStore			
TestJMSServer	TestStore	soa_0629	soa_0629	✓ OK
UMSJMSServer	UMSJMSFileStore	soa_0629	soa_0629	✓ OK

4. Create a test JMS module as shown in Figure J-2.

Figure J-2 Creating a Test JMS Module



Home Log Out Preferences Record Help | Welcome, weblogic | Connected to: basa_domain

Home >Summary of JMS Servers >TestJMSServer >Summary of JMS Servers >JMS Modules >TestJMSModule >**JMS Modules**

JMS Modules

JMS system resources are configured and stored as modules similar to standard J2EE modules. Such resources include queues, topics, connection factories, templates, destination keys, quota, distributed queues, distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters. You can administratively configure and manage JMS system modules as global system resources.

This page summarizes the JMS system modules that have been created for this domain.

Customize this table

JMS Modules

Click the ***Lock & Edit*** button in the Change Center to activate all the buttons on this page.

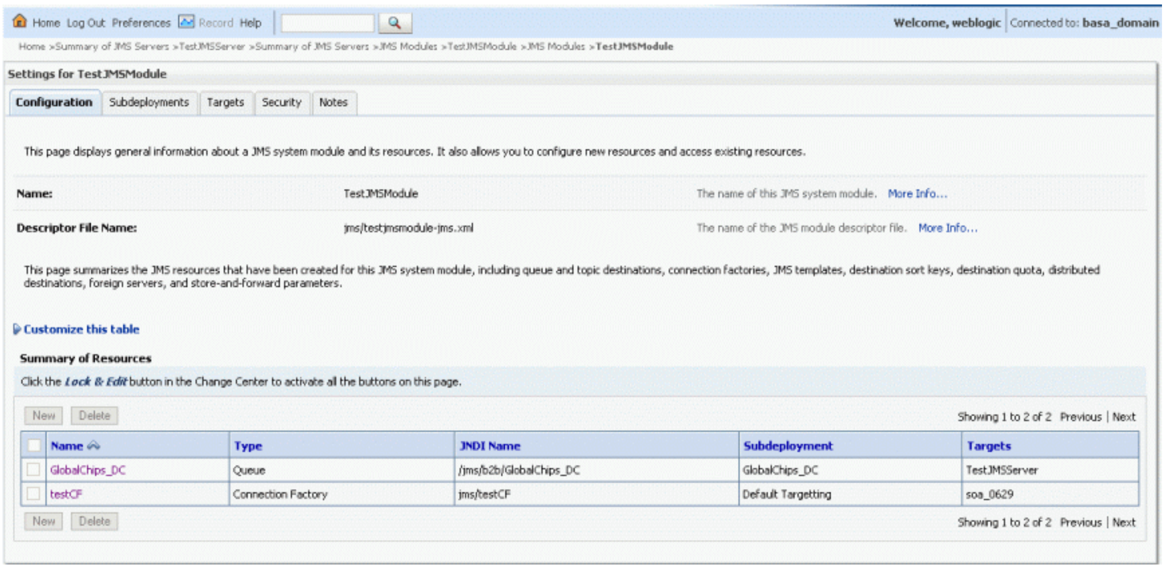
New Delete Showing 1 to 4 of 4 Previous | Next

Name	Type
BPMJMSModule	System
SOAJMSModule	System
TestJMSModule	System
UMSJMSSystemResource	System

New Delete Showing 1 to 4 of 4 Previous | Next

5. Launch the test JMS module that you created and create a connection factory with target as SOA server. Create a JMS Queue and with the target as the created test JMS server as shown in Figure J-3.

Figure J-3 Creating a Test JMS Queue



Home Log Out Preferences Record Help | Welcome, weblogic | Connected to: basa_domain

Home >Summary of JMS Servers >TestJMSServer >Summary of JMS Servers >JMS Modules >TestJMSModule >JMS Modules >**TestJMSModule**

Settings for TestJMSModule

Configuration | Subdeployments | Targets | Security | Notes

This page displays general information about a JMS system module and its resources. It also allows you to configure new resources and access existing resources.

Name:	TestJMSModule	The name of this JMS system module. More Info...
Descriptor File Name:	jms/testjmsmodule-jms.xml	The name of the JMS module descriptor file. More Info...

This page summarizes the JMS resources that have been created for this JMS system module, including queue and topic destinations, connection factories, JMS templates, destination sort keys, destination quota, distributed destinations, foreign servers, and store-and-forward parameters.

Customize this table

Summary of Resources

Click the ***Lock & Edit*** button in the Change Center to activate all the buttons on this page.

New Delete Showing 1 to 2 of 2 Previous | Next

Name	Type	JNDI Name	Subdeployment	Targets
GlobalChips_DC	Queue	/jms/b2b/GlobalChips_DC	GlobalChips_DC	TestJMSServer
testCF	Connection Factory	jms/testCF	Default Targetting	soa_0629

New Delete Showing 1 to 2 of 2 Previous | Next

Setting Up an Environment to Connect to the JMS queue

After 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:

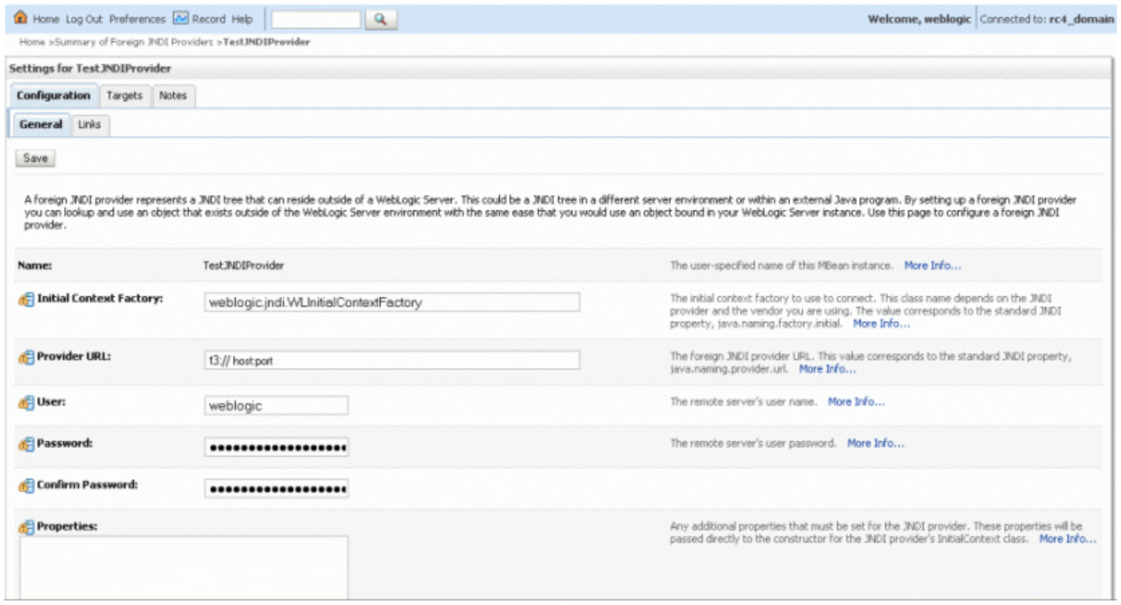
1. Install Weblogic server and SOA server.

> **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 J-4.

Figure J-4 Configuring Connection to Remote Server



Home Log Out Preferences Record Help
Welcome, weblogic
Connected to: rc4_domain
Home >Summary of Foreign JNDI Providers >TestJNDIProvider
Settings for TestJNDIProvider
Configuration
Targets
Notes
General
Links
Save

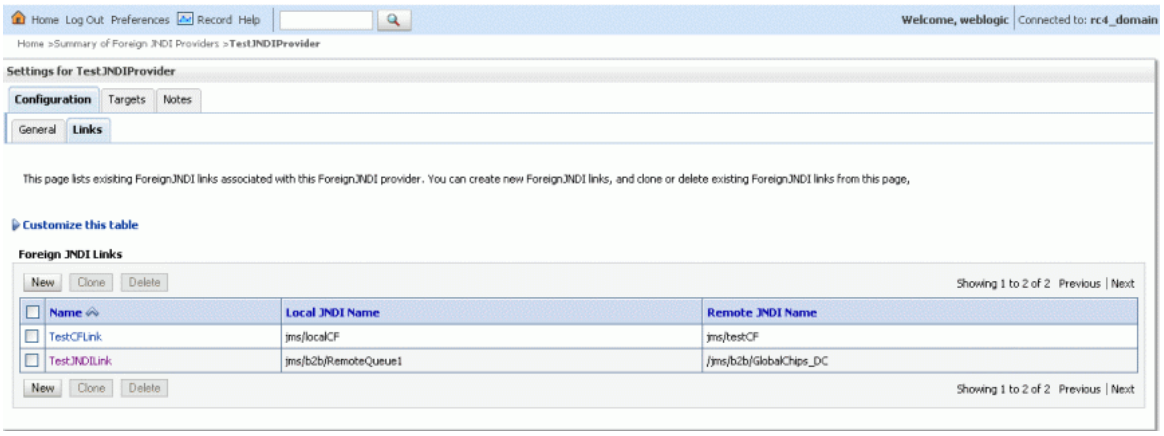
A foreign JNDI provider represents a JNDI tree that can reside outside of a WebLogic Server. This could be a JNDI tree in a different server environment or within an external Java program. By setting up a foreign JNDI provider you can lookup and use an object that exists outside of the WebLogic Server environment with the same ease that you would use an object bound in your WebLogic Server instance. Use this page to configure a foreign JNDI provider.

Name:	TestJNDIProvider	The user-specified name of this MBean instance. More Info...
Initial Context Factory:	weblogic.jndi.WLInitialContextFactory	The initial context factory to use to connect. This class name depends on the JNDI provider and the vendor you are using. The value corresponds to the standard JNDI property, java.naming.factory.initial. More Info...
Provider URL:	t3:// host:port	The foreign JNDI provider URL. This value corresponds to the standard JNDI property, java.naming.provider.url. More Info...
User:	weblogic	The remote server's user name. More Info...
Password:	•••••••••••••••••••	The remote server's user password. More Info...
Confirm Password:	•••••••••••••••••••	
Properties:		Any additional properties that must be set for the JNDI provider. These properties will be passed directly to the constructor for the JNDI provider's InitialContext class. More Info...

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 J-5 shows the Foreign JNDI Links table displaying a list of foreign JNDI links.

Figure J-5 Foreign JNDI Links


Home Log Out Preferences Record Help
Welcome, weblogic Connected to: rc4_domain
Home >Summary of Foreign JNDI Providers >TestJNDIProvider
Configuration Targets Notes
General Links
Customize this table
New Clone Delete
Showing 1 to 2 of 2 Previous | Next

Settings for TestJNDIProvider

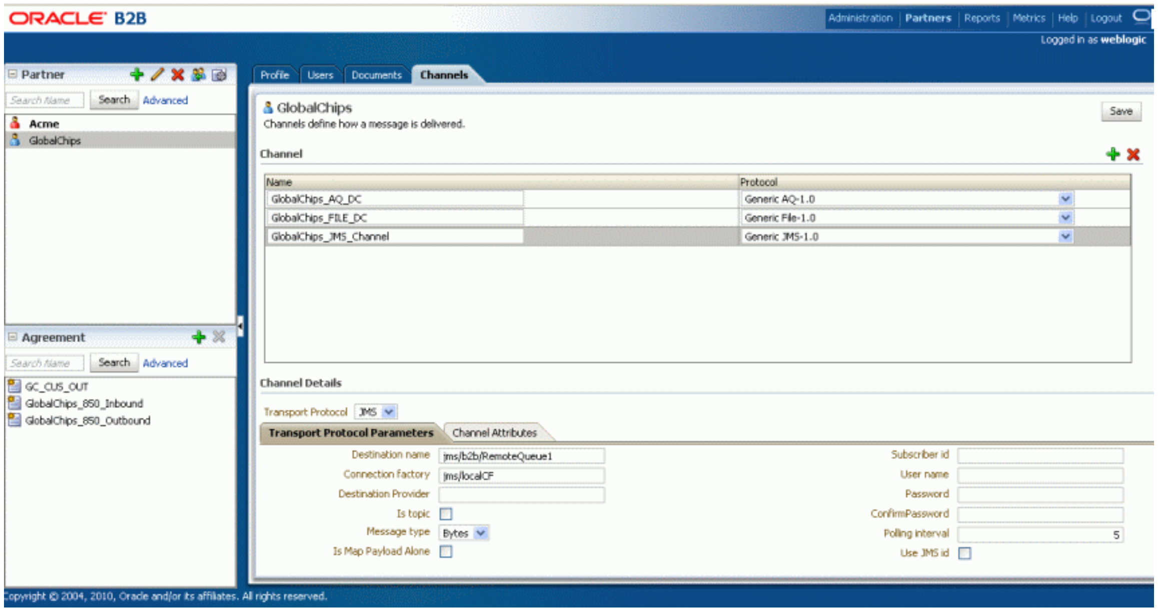
This page lists existing ForeignJNDI links associated with this ForeignJNDI provider. You can create new ForeignJNDI links, and clone or delete existing ForeignJNDI links from this page.

Foreign JNDI Links

Name	Local JNDI Name	Remote JNDI Name
TestCFLink	jms/localCF	jms/testCF
TestJNDILink	jms/b2b/RemoteQueue1	/jms/b2b/GlobalChips_DC

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 J-6.

Figure J-6 Creating a JMS Channel in the B2B User Interface



GlobalChips

Channels define how a message is delivered.

Channel

Name	Protocol
GlobalChips_AQ_DC	Generic AQ-1.0
GlobalChips_FILE_DC	Generic File-1.0
GlobalChips_JMS_Channel	Generic JMS-1.0

Channel Details

Transport Protocol: JMS

Destination name	jms/b2b/RemoteQueue1	Subscriber id	
Connection factory	jms/localCF	User name	
Destination Provider		Password	
Is topic	☐	ConfirmPassword	
Message type	Bytes	Polling interval	5
Is Map Payload Alone	☐	Use JMS id	☐

Copyright © 2004, 2010, Oracle and/or its affiliates. All rights reserved.

K

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 includes the following sections:

- Inbound Messages
- Outbound Messages
- Using a JMS Queue for Error Message Delivery
- Using a Custom Exception Queue for Error Message Delivery
- Inbound Exception Handling Scenarios
- Exception Payload Definition
- AS4-Based Message Errors

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

Inbound Messages

This section describes inbound message types.

- Request or Response Messages
- Acknowledgment Messages
- Exception Messages

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.

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.

Inbound Message Content-Type Handling for AS1 Messages

Inbound message content-type must be one of the following for AS1 messages to be retrieved from the email server and processed properly:

- application/xml
- enveloped
- pkcs7-mime
- ipart/report
- multipart/signed
- application/edi

For example, given this scenario:

```
Sender: Acme admin@slc05hzo.us.mydomain.com
Receiver: GlobalChips joe@slc05hzo.us.mydomain.com
Outbound channel: joe@slc05hzo.us.mydomain.com
Listening channel is listening on admin@slc05hzo.us.mydomain.com
```

If an inbound payload is sent by sending email to admin@slc05hzo.us.mydomain.com with the payload as attachment, the reports show the following error message.

```
The error info was as follows, see "Inbound failed.jpg":
Error Code: B2B-51566
Error Description: Machine Info: (slc05hzo) Description: Parse stream
error
```

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.

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 Using a Custom Exception Queue for Error Message Delivery for more information.

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.
- The data inside the `<b2bMessageId>` tag of the exception message contains non-parsable characters and so, it is provided in the CDATA section so that it does not get parsed removing the chances of getting XML parsing errors.
- 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.

Using a JMS Queue for Error Message Delivery

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 default settings, as described in Inbound Messages and Outbound Messages, use an AQ queue, `IP_IN_QUEUE`, as the exception queue. 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 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 15-1 for more information on the **Use JMS Queue as default** parameter.

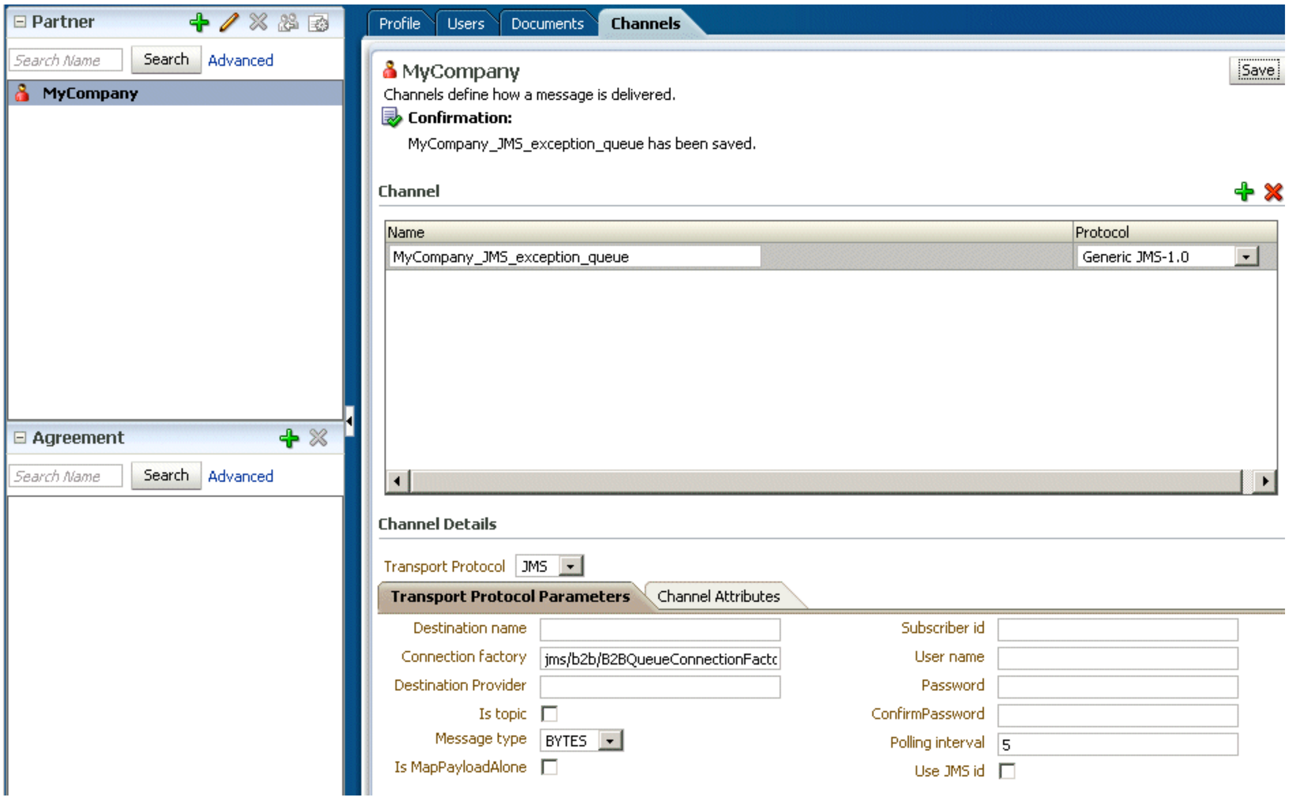
Using a Custom Exception Queue for Error Message Delivery

This section contains the steps to create a custom JMS exception queue.

To create custom JMS exception queues, perform these steps:

1. 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 K-1.

Figure K-1 Creating a Custom Exception Queue



MyCompany

Channels define how a message is delivered.

Confirmation:

MyCompany_JMS_exception_queue has been saved.

Channel

Name	Protocol
MyCompany_JMS_exception_queue	Generic JMS-1.0

Channel Details

Transport Protocol: JMS

Transport Protocol Parameters | Channel Attributes

Destination name		Subscriber id	
Connection factory	jms/b2b/B2BQueueConnectionFactc	User name	
Destination Provider		Password	
Is topic	☐	ConfirmPassword	
Message type	BYTES	Polling interval	5
Is MapPayloadAlone	☐	Use JMS id	☐

2. Select the queue from the **Exception Queue** parameter on the **Configuration** tab. The **Exception Queue** drop down 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 15-1 for more information on the **Exception Queue** parameter.

Inbound Exception Handling Scenarios

This section describes inbound exception handling scenarios.

Table K-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

Table K-1 (Cont.) Inbound Exception Handling Scenarios

If an exception occurs because. . .	Then Oracle B2B does. . .
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
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.

Exception Payload Definition

This section shows the definition for the exception payload, `ipException.xsd`.

Example K-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>
```

AS4-Based Message Errors

This section describes AS4–based message errors, including SOAP faults and ebMS errors.

SOAP Fault

The receiving partner reports errors from message processing as SOAP faults to the sender. The error message is sent to the sending MSH using the same connection. Also, the backend application is notified of the errors.

ebMS Error

The error codes shown in Table K-2extend the set of ebMS V3 error codes to support the AS4 additional features.

Table K-2   ebMS V3 Error Codes

Error Code	Short Description	Description or Semantics
EBMS:0301	MissingReceipt	A receipt has not been received for a message that was previously sent by the MSH generating this error.
EBMS:0302	InvalidReceipt	A receipt has been received for a message that was previously sent by the MSH generating this error, but the content does not match the message content; for example, some part has not been acknowledged, or the digest associated does not match the signature digest, for NRR.
EBMS:0303	Decompression-Failure	An error occurred during the decompression.

L

Database Partitioning

This appendix discusses the partitioning of the Oracle B2B instance tables.

The appendix contains the following sections:

- Introduction
- Partitioning Requirements

Introduction

Oracle B2B allows you to partition the Oracle B2B instance tables based on the `CPST_INST_CREATED_TIME` column.

The following tables can be partitioned:

- `B2B_APP_MESSAGE`
- `B2B_BUSINESS_MESSAGE`
- `B2B_DATA_STORAGE`
- `B2B_EXT_BUSINESS_MESSAGE`
- `B2B_WIRE_MESSAGE`

Partitioning Requirements

In this section are the requirements that you need for successful partitioning of the Oracle B2B tables.

- Use range or interval partitioning to partition the tables.
- Ensure that the date ranges specified across all the Oracle B2B tables are the same. This is a prerequisite for purge to function properly.
- When using range partitioning, it is recommended to use a default partition to avoid runtime failures.
- Re-creation of tables:
  - The Oracle B2B system materialized view has dependencies on the Oracle B2B instance tables (`B2B_BUSINESS_MESSAGE`, `B2B_EXT_BUSINESS_MESSAGE`) and materialized view logs are created on these tables. During the creation of the partitions, the materialized view logs need to be re-created and the system materialized view needs to be fully refreshed.
  - Post database table partitioning, the materialized view `B2B_SYSTEM_MV` is in STALE. This would have happened after partitioning the DB as any DDL operation on one of the materialized view's dependencies (DROP, ALTER...) will invalidate it. Even though the corresponding master tables (`B2B_BUSINESS_MESSAGE`, `B2B_EXT_BUSINESS_MESSAGE`) now after partitioning are FRESH, this view has to be updated.

- After database table partitioning, you need to recreate the B2B indexes, otherwise the throughput of the system can be drastically reduced.
- At the time of upgrade:
  - It is recommended to perform the partitioning only after the application instances are upgraded to the required version of Oracle SOA Suite, because upgrading the Oracle SOA Suite schema needs to be completed before partitioning. This is done to avoid NULL values being introduced in the `CPST_INST_CREATED_TIME` column of the database from the older versions of the runtime.
  - In case there are existing data in these tables that contains NULL values for the `CPST_INST_CREATED_TIME` columns, you need to update the records to non-NULL values for appropriate partitioning.

Setting Up Partitioning

To support purge based on partitioning the instance tables of B2B need to be partitioned.

You need to partition the following tables:

- B2B_APP_MESSAGE
- B2B_BUSINESS_MESSAGE
- B2B_WIRE_MESSAGE
- B2B_EXT_BUSINESS_MESSAGE
- B2B_DATA_STORAGE

By default these tables will be set up in the schema with referential constraints. The DBA's should disable the constraints in the environment where the partitions need to be setup.

These tables will have to be partitioned based on the CPST_INST_CREATED_TIME column present in these tables.

The DBAs can use either range or interval partitioning to set up the partitions.

The ranges of the partitions should be the same across all the B2B instance tables.

The current purge supports an all or none partition mechanism where it expects all the B2B tables to be partitioned for the partition based purge to take effect.

Migrating Data

For migrating data from the existing tables to the partitioned tables, you can choose a strategy that is applicable based on the volume of data which needs to be migrated. Some of the more popular strategies employed include

- Create Table as Select (CTAS)
  - Creating a new partitioned table by copying over the required data from the existing tables using the CTAS statement
  - Once the tables are created, the DBAs can create indexes on the new tables and rename them to the original table name.
- Exchange partition
  - Exchange partition achieves data movement, by exchanging data segments between the existing table and the partitioned table.

Note:

Prior versions of B2B messages (pre-11.1.1.7 releases) might have the `CPST_INST_CREATED_TIME` field as null in some cases (due to resubmits).

You can clean these up by populating these fields with values from the `CREATED_TIME` column of the `B2B_BUSINESS_MESSAGE` table.

Purge Process

To accomplish the purge process:

- You can use the command line purge -Dpartitioned=true parameter to invoke the partition based purge.
- The partition-based purge is triggered only if date range is the only criteria provided. If any other date range is provided, the purge falls back to the normal purge mechanism.

M

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 includes the following sections:

- Protocols
- Identifications
- Security Specifications
- Exchange Protocols Parameter Values
- Transport Protocols Parameter Values
- Document Protocol Parameter Values

Protocols

Document protocols, exchange protocols, and transport protocols are listed here.

Document Protocols

- OAG
- UCCNET
- RosettaNet
- Custom

Exchange Protocols

- AS2
- MLLP
- ebMS2
- ebMS1
- RNIF20
- RNIF11
- AS1
- Generic-File
- Generic-AQ
- Generic-FTP
- Generic-SFTP
- Generic-JMS

- Generic-HTTP
- Generic-Email

Transport Protocols

- HTTP
- File
- AQ
- JMS
- FTP
- SFTP
- AS1
- TCP
- Email

Identifications

Any of the these values can be used as Identification name in Self-Service. Name Identifier will be created by Self-Service using the Trading Partner name.

Table M-1 Identifications

Identification Name	Identification ID
Generic Identifier	Generic
DUNS	DUNS
ebMS Identifier	ebMS
AS2 Identifier	AS2
MLLP ID	MLLP
AS1 Identifier	AS1

Security Specifications

Any of these values can be used as Identification name in Self-Service.

Table M-2 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 3.0 with SHA256 - RSA	SMIME-3_0-SHA256-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

Table M-2 (Cont.) Security Specifications

Security Specifications Name	Security Specifications ID
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
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

Exchange Protocols Parameter Values

This table lists the valid exchange protocol parameter values.

Table M-3 Exchange Protocols Parameter Values

Exchange	Parameter Value Name	Domain	Required
AS2	Receipt-Delivery-Option	String, any URL	No
AS2	Signed-And-Compressed	Boolean, false (default), true	No
MLLP	ImmediateACK	String, any of these values - None (default), Default, Simple, Custom	No
MLLP	ImmediateACK-Custom-File	Absolute File Path	No
MLLP	ImmediateACK-mapAckControlID	Boolean, false (default), true	No
MLLP	ImmediateACK-MapImmTriggerEvt	Boolean, false (default), true	No
MLLP	DiscardHL7ACK	String, any of these values -None (default), AA, AE, AR, CA, CE, CR	No
MLLP	Start-Block-Char	Hexadecimal, 0x0B(default)	No
MLLP	End-Block-Char	Hexadecimal, 0x1C(default)	No
MLLP	Carriage-Return-Char	Hexadecimal, 0x0D(default)	No
MLLP	Identify-TP-by-delivery-channel	Boolean, false (default), true	No
ebMS2.0	Duplicate-Elimination	Boolean, false (default), true	No
ebMS2.0	messageOrderSemantics	String	No
ebMS2.0	PersistDuration	String	No
ebMS2.0	SendPartyTypeAndValue	Boolean, false (default), true	No

Table M-3 (Cont.) Exchange Protocols Parameter Values

Exchange	Parameter Value Name	Domain	Required
ebMS1.0	Duplicate-Elimination	Boolean, false (default), true	No
ebMS1.0	SendPartyTypeAndValue	Boolean, false (default), true	No
AS1	Signed-And-Compressed	Boolean, false (default), true	No

Transport Protocols Parameter Values

This table lists the valid transport protocol parameter values.

Table M-4 Transport protocols parameter Values

Transport	Parameter Name Value	Domain	Required
HTTP	url	String, any URL	Yes
HTTP	user	String	No
HTTP	password	String	No
HTTP	additional_headers	String	No
HTTP	use_proxy	Boolean, false (default), true	No
File	polling_interval	Integer, 5(default)	No
File	folder	String	Yes
File	filename_format	String	No
AQ	sid	String, orcl (default)	No
AQ	port	Integer, 1521(default)	No
AQ	schema	String	No
AQ	queue_name	String	No
AQ	password	String	No
AQ	host	String	No
AQ	polling_interval	Integer	No
AQ	recipient	String	No
AQ	consumer	String	No
AQ	datasource	String, either `datasource` or `jdbc(host,sid,port,schema)`	No
JMS	queue_name	String	No
JMS	jndi_connection_factory_location	String	No
JMS	is_topic	Boolean, false (default), true	No
JMS	is_map_message	String, Any of these values BYTES,TEXT,MAP	No
JMS	is_map_payload_alone	Boolean, false (default), true	No
JMS	Subscriber_ID	String	No
JMS	user	String	No
JMS	password	String	No

Table M-4   (Cont.) Transport protocols parameter Values

Transport	Parameter Name Value	Domain	Required
JMS	polling_interval	Integer, 5(default)	No
FTP	host	String	Yes
FTP	polling_interval	Integer, 5(default)	No
FTP	folder	String	Yes
FTP	user	String	Yes
FTP	password	String	No
FTP	channel_mask	String, None (default)	No
FTP	cipher_suites	String	No
FTP	control_port	Integer	No
FTP	data_port	Integer	No
FTP	use_proxy	Boolean, false (default), true	No
FTP	filename_format	String	No
FTP	sourcefile_encoding	String	No
SFTP	host	String	Yes
SFTP	port	Integer	No
SFTP	polling_interval	Integer, 5(default)	No
SFTP	folder	String	Yes
SFTP	user	String	Yes
SFTP	password	String	No
SFTP	private_key	String	No
SFTP	pass_phrase	String	No
SFTP	use_proxy	Boolean, false (default), true	No
SFTP	filename_format	String	No
TCP	sockettype	String, Either of Server, Client (default)	No
TCP	host	String	No
TCP	port	Integer	No
TCP	PermanentConnectionType	Boolean, false (default), true	No
TCP	Sequencing	Boolean, false (default), true	No
TCP	pollinterval	Integer, 10(default)	No
TCP	timeout	Integer, 300(default)	No
Email	host	String	Yes
Email	password	String	No
Email	user	String	No
Email	polling_interval	Integer	No
Email	content-type	String	No
Email	send_as_attachment	String	No
Email	folder	String	No

Table M-4 (Cont.) Transport protocols parameter Values

Transport	Parameter Name Value	Domain	Required
Email	email-id	String	Yes
Email	subject	String	No
Email	server	String, either of IMAP (default), PoP3	Yes

Document Protocol Parameter Values

The valid document protocol parameter values are listed.

This section contains the following tables:

- Table M-5 RosettaNet Document Protocol Parameter Values
- Table M-6 Custom Document Protocol Parameter Values
- Table M-7 Common Parameter Values
- Table M-8 AgrDocType Parameter Values

Table M-5 RosettaNet Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Type	FromRole	String	Yes
Type	ToRole	String	Yes
Type	ServiceHeader	String	Yes
Type	ToService	String	Yes
Type	BusinessTransaction	String	Yes
Type	ServiceHeader	String	Yes
Type	CollaborationTimeToPerform	String	Yes
Type	CollaborationName	String	Yes
Type	CollaborationCode	String	Yes
Definition	DocumentRoutingID	String	No
Definition	DTDXSDNamespaceConversion	String, any of these values - Both, Inbound, Outbound, None	No
Definition	Common Parameter Values (see Table M-7)	String	No

Table M-6 Custom Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Type	ActionName	String	No
Type	Service	String	No
Type	ServiceType	String	No
Type	FromRole	String	No

Table M-6 (Cont.) Custom Document Protocol Parameter Values

Document	Parameter Value Name	Domain	Required
Type	ToRole	String	No
Definition	IdentificationExpression	String	No
Definition	IdentificationExpressionValue	String	No
Definition	DTDXSDNamespaceConversion	String, any of these values - None, Both, Inbound, Outbound	No
Definition	IdentificationStartPosition	String	No
Definition	IdentificationEndPosition	String	No
Definition	FlatIdentificationExpressionValue	String	No
Definition	DocumentRoutingID	String	No
Definition	Common Parameter Values (see Table M-7)	String	No

Table M-7 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
CorrelationToXPathName	String, CorrelationToXPathName (default)	No
CorrelationToXPathXPR	String	No

Table M-8 AgrDocType Parameter Values

Parameter Value Name	Domain	Required	Applicable Documents
validate	boolean	No	Depends on document protocol
translate	boolean	No	X12, EDIFACT, HL7
fa	boolean	No	X12, EDIFACT, HL7

N
Exchanging Messages By Using IBM Websphere MQ

This appendix discusses the communication of Oracle B2B with IBM Websphere MQ.

This appendix contains the following sections:

Overview

Oracle B2B provides the "out-of-the box" feature to connect to IBM MQ to exchange messages. This is support is provided by using the JMS offering of Oracle B2B.

This feature is in addition to the stack of existing communication capabilities of Oracle B2B with trading partners.

Prerequisites

These libraries must be specified in the classpath to connect to IBM MQ.

- com.ibm.mqjms.jar
- dhbcore.jar
- com.ibm.mq.jar
- com.ibm.mq.jmqi.jar
- mqcontext.jar
- com.ibm.mq.pcf.jar
- com.ibm.mq.commonservices.jar
- com.ibm.mq.headers.jar
- fscontext.jar
- jms.jar

Add the preceding jars in the domain library directory. The directory is typically located at *$DOMAIN_DIR*/lib.

The jars located in the *$DOMAIN_DIR*/lib directory are picked and added dynamically to the end of the server classpath at server startup.

For example, <*$FMW_Home*>/user_projects/domains//lib/

Alternatively, the jars can also be added as a part of the setDomainEnv.sh file.

Connecting to IBM MQ

There are two ways of connecting to IBM MQ using Oracle B2B.

- Configuring Credential-based Connectivity
- Configuring Bindings-based Connectivity

Configuring Credential-based Connectivity

In the case of credential-based connectivity, you have to configure both the outbound and inbound channels.

For outbound, you need to configure the trading partner delivery channel for using the Generic JMS protocol.

For inbound, you need to configure the internal delivery channel for using "Generic JMS" protocol by using the following parameters:

- Destination Name: The MQ Queue name
- Connection Factory: The MQ Queue Manager name
- Destination Provider:
  `java.naming.factory.initial=com.ibm.mq.jms.context.WMQInitialContextFactory`

  `java.naming.provider.url=`*<host>:<QM Listen port>*/*<MQ Channel Name>*;
- User Name: The MQ user name
- Password: The MQ password

Configuring Bindings-based Connectivity

As a prerequisite, you need to obtain or generate the .bindings file in the MQServer. This can be done by an MQ Administrator.

Set the following values in the respective delivery channel for the outbound or inbound case:

- Destination Name: The MQ Queue name
- Connection Factory: The MQ Queue Manager name
- Destination Provider:
  `java.naming.factory.initial=com.ibm.mq.jms.context.WMQInitialContextFactory`

  `java.naming.provider.url=`*file:///<location of .bindings file>*

O

Running Oracle B2B as a Hub

Oracle B2B provides an out-of-box solution that allows Oracle B2B to run as a hub.

The appendix includes the following sections:

Introduction

To run Oracle B2B as a hub, you need to configure three trading partners.

For example:

- OracleServices: The host trading partner that acts as a hub.
- MarketInc: The trading partner that sends CUSTOM PurchaseOrder_A01 documents to another trading partner, RetailInc by using OracleServices.
- RetailInc: The trading partner that receives the CUSTOM PurchaseOrder_A01 documents sent by MarketInc through OracleServices.

So, the message flow as per the preceding example will be:

```
MarketInc  <--- PurchaseOrder_A01 ----> OracleServices (HUB) <--- PurchaseOrder_A01 ----
> RetailInc
```

The preceding flow demonstrates that MarketInc is sending the PurchaseOrder_A01 messages to OracleServices, but actually in the technical implementation, MarketInc assumes the following flow:

```
MarketInc <----------------> RetailInc
```

So, at the Oracle B2B server of MarketInc, the agreement(s) are configured between MarketInc and RetailInc, but the channel location is the one on which OracleServices' Oracle B2B server listens for messages from partners.

Configuring Oracle B2B as a Hub

The overall steps to configure Oracle B2B hub are as shown below.

To configure Oracle B2B as a hub:

1. Create two CUSTOM document definition as the following:
   - PurChaseOrder_A01_Def
   - PurChaseOrder_A01-Def

Note:

There is no specific configuration required in creating the document definition. It is the same as creating documents for one-to-one communication.

2. Configure the trading partners:
   - Configured MarketInc as sender of PurchaseOrder and receiver of ACK.
   - Configure RetailInc as receiver of PurchaseOrder and sender of ACK.
3. Configure the channels and add identifiers in the trading partner profile.
4. Configure the agreements:
   - MarketInc: Configure two agreements with MarketInc, one each for PurchaseOrder (Inbound) and ACK (Outbound).
   - RetailInc: Configure two agreements with RetailInc, one each for PurchaseOrder (Outbound) and ACK (Inbound).
5. Save, validate, and deploy the agreements.

Testing the Configuration

To start and test the message flow, place a PurchaseOrder_A01 message in the external listening channel directory of OracleServices.

Once the message gets removed from the directory, navigate to Oracle B2B reports and verify:

- The incoming PurchaseOrder from MarketInc has been processed successfully.
- An ACK for incoming PurchaseOrder has been sent to MarketInc.
- The same PurchaseOrder has been passed to RetailInc.