Oracle® Service Bus

Using the Oracle Service Bus Plug-ins for Workshop for WebLogic

10*g* Release 3 Maintenance Pack 1 (10.3.1)

June 2009

ORACLE

Oracle Service Bus Using the Oracle Service Bus Plug-ins for Workshop for WebLogic, 10*g* Release 3 Maintenance Pack 1 (10.3.1)

Contents

3. User Interface Reference

4. Transport Configuration

CHAPTER 1

Introduction to Oracle Service Bus

Oracle Service Bus is a configuration-based, policy-driven Enterprise Service Bus (ESB). It provides highly scalable and reliable service-oriented integration, service management, and traditional message brokering across heterogeneous IT environments. It combines intelligent message brokering with routing and transformation of messages, along with service monitoring and administration in a unified software product. For more information, see *Oracle Service Bus Concepts and Architecture*.

This documentation tells how to use the Oracle Service Bus Plug-ins for Workshop for WebLogic to configure services and polices. These are design-time activities, so you do not need a running WebLogic Server instance. The Web-based Oracle Service Bus Console, also provided with Oracle Service Bus, provides the same design tools as the Oracle Service Bus plug-ins, and it also provides run-time activities such as monitoring. You need a running WebLogic Server instance to use the Oracle Service Bus Console. See *Using the Oracle Service Bus Console*.

CHAPTER 2

Tasks

This section tells how to perform tasks in the Oracle Service Bus Plug-ins for Workshop for WebLogic:

Working with Projects, Folders, Resources, and Configurations

This section tells how to perform the following tasks:

- “Creating Oracle Service Bus Configuration Projects” on page 2-3
- “Creating Oracle Service Bus Projects” on page 2-4
- “Creating Custom Resources” on page 2-4
- “Creating JNDI Provider Resources” on page 2-4
- “Creating Message Format Files” on page 2-4
- “Editing JNDI Provider Resources” on page 2-4
- “Editing JNDI Provider Resources” on page 2-4
- “Exporting Resources” on page 2-5
- “Generating an Effective WSDL” on page 2-13
- “Modifying JAR Dependencies” on page 2-13
- “Importing Resources” on page 2-14
- “Creating Servers” on page 2-22
- “Creating Service Account Resources” on page 2-22
- “Creating Service Key Provider Resources” on page 2-22
- “Creating SMTP Server Resources” on page 2-23
- “Creating XQuery Transformations” on page 2-23
- “Creating XSL Transformations” on page 2-23

See also the following information about working with other kinds of resources:

- “Working with Business Services” on page 2-23
- “Working with Proxy Services” on page 2-24
- “Working with Alert Destinations” on page 2-26
- “Working with MQ Connections” on page 2-46
- “Working with UDDI Registries” on page 2-47

Editing Resources

Edit resources using the built-in editors. For example, edit a proxy service by double-clicking its name in the Project Explorer.

Do not manually edit resource files as text or XML files. This can result in unpredictable behavior. Do not manually edit these resource types:

- Alert Destination
- Business Service
- Custom Resources
- Proxy Service
- Service Account
- Service Key Provider
- Split-Join
- JNDI Provider
- SMTP Server
- Proxy Server
- UDDI Registry

Cloning Oracle Service Bus Projects and Folders

1. In the Project Explorer, right-click the Oracle Service Bus project or folder you want to clone.
2. From the menu, select **Oracle Service Bus > Clone** to display the Select Clone Target dialog. Enter information as appropriate.

Creating Oracle Service Bus Configuration Projects

In the Oracle Service Bus perspective, select **File > New >** Oracle Service Bus **Configuration Project** to display the New Oracle Service Bus Configuration Project wizard. Enter information as appropriate.

Creating Oracle Service Bus Projects

In the Oracle Service Bus perspective, select **File > New > Oracle Service Bus Project** to display the New Oracle Service Bus Project wizard. Enter information as appropriate.

Note: You can create an Oracle Service Bus project in an Oracle Service Bus configuration project only.

Creating Custom Resources

In the Oracle Service Bus perspective, select **File > New > Custom Resource** to display New Custom Resource wizard. Enter information as appropriate.

Note: You can create a custom resource in an Oracle Service Bus project only.

Creating JNDI Provider Resources

In the Oracle Service Bus perspective, select **File > New > JNDI Provider** to display the New JNDI Provider Resource wizard. Enter information as appropriate.

Note: You can create a JNDI provider resource in an Oracle Service Bus configuration project only.

Creating Proxy Server Resources

In the Oracle Service Bus perspective, select **File > New > Proxy Server** to display the New Proxy Server Resource wizard. Enter the information described in Proxy Servers as appropriate.

Note: You can create a proxy server resource in an Oracle Service Bus configuration project only.

Creating Message Format Files

In the Oracle Service Bus perspective, select **File > New > MFL** to display the New Message Format File wizard. Enter information as appropriate.

Note: You can create a message format file in an Oracle Service Bus project only.

Editing JNDI Provider Resources

1. In the Project Explorer, find the Oracle Service Bus configuration project containing the JNDI provider resource you want to edit.

2. Double-click the name of the JNDI provider to display the JNDI Provider Editor. Edit as appropriate.

Editing Proxy Server Resources

1. In the Project Explorer, find the Oracle Service Bus configuration project containing the proxy server resource you want to edit.
2. Double-click the name of the proxy server to display the Proxy Server Editor. Edit the information described in Proxy Servers as appropriate.

Exporting Resources

In the Oracle Service Bus perspective, select **File > Export** to display the Export wizard. See:

- Export wizard - Oracle Service Bus Configuration JAR Export page
- Export wizard - Export to Server - Select Resources page

Using the Command Line or a Script to Export an Oracle Service Bus Configuration

This section describes scripting and command-line options for exporting an Oracle Service Bus configuration:

- Exporting a Configuration Using Ant
- Exporting a Configuration Using WLST
- Exporting a Configuration Using the Command Line

Before You Begin

Refer to the following prerequisites and guidance before you begin.

- Java 1.6 is required.
- Oracle Service Bus 10gR3 MP1 or later and Eclipse must be installed.
- You may see exception stack traces in out the output or the workspace log file if workspace files are read-only.
- An exit value of 0 means the export succeeded.

Exporting a Configuration Using Ant

You can export an Oracle Service Bus configuration using an Ant buildfile. Following is a sample buildfile with accompanying properties file.

Ant Buildfile Example

```
<project name="ConfigExport">
    <property file="./build.properties"/>
    <property name="eclipse.home"
     value="${bea.home}/tools/eclipse_pkgs/2.0/eclipse_3.3.2/eclipse"/>
    <property name="weblogic.home" value= "${bea.home}/wlserver_10.3"/>
    <property name="metadata.dir" value="${workspace.dir}/.metadata"/>

    <target name="export">
        <available file="${metadata.dir}" type="dir"
         property="metadata.dir.exists"/>

        <java dir="${eclipse.home}"
jar="${eclipse.home}/plugins/org.eclipse.equinox.launcher_1.0.1.R33x_v2008
0118.jar"
          fork="true"
          failonerror="true"
          maxmemory="768m">
           <arg line="-data ${workspace.dir}"/>
           <arg line="-application com.bea.alsb.core.ConfigExport"/>
           <arg line="-configProject ${config.project}"/>
           <arg line="-configJar ${config.jar}"/>
           <sysproperty key="weblogic.home" value="${weblogic.home}"/>
        </java>

             <antcall target="deleteMetadata"/>
    </target>

       <target name="deleteMetadata" unless="metadata.dir.exists">
         <delete failonerror="false" includeemptydirs="true"
          dir="${metadata.dir}"/>
       </target>
</project>
```

build.properties Example

```
bea.home=c:/bea
workspace.dir=c:/bea/user_projects/workspaces/default
config.project="OSB Configuration"
config.jar=c:/sbconfig.jar
```

Running “ant export” results in exporting the project “OSB Configuration” from the default workspace to c:\sbconfig.jar.

Exporting a Configuration Using WLST

You can export an Oracle Service Bus configuration using the WebLogic Scripting Tool (WLST). Following are sample files involved in a WLST export. The Ant buildfile and Python script use a properties file and a customization file.

Ant Buildfile Example

You can also use the following Ant buildfile for importing an Oracle Service Bus configuration.

```
<project default="export">

  <property environment="env"/>
  <property name="domain.export.script" value="export.py"/>
  <property name="domain.import.script" value="import.py"/>
  <property name="export.config.file" value="export.properties"/>
  <property name="import.config.file" value="import.properties"/>
  <property name="build" value="build"/>
  <property name="dist" value="dist"/>
  <property name="bea.home" value="${env.BEA_HOME}"/>

    <path id="class.path">
      <pathelement
        path="${bea.home}/wlserver_10.3/server/lib/weblogic.jar"/>
      <pathelement path="${bea.home}/osb_10.3/lib/sb-kernel-api.jar"/>
    <pathelement
        path="${bea.home}/modules/com.bea.common.configfwk_1.2.0.0.jar"/>
    </path>

    <taskdef name="wlst"
        classname="weblogic.ant.taskdefs.management.WLSTTask"/>

    <target name="export">
    <echo message="exportscript: ${domain.export.script}"/>
```

```
    <java classname="weblogic.WLST" fork="true">
        <arg line="${domain.export.script} ${export.config.file}"/>
        <classpath refid="class.path"/>
    </java>
  </target>

  <target name="import">
    <echo message="importscript: ${domain.import.script}"/>
    <java classname="weblogic.WLST" fork="true">
        <arg line="${domain.import.script} ${import.config.file}"/>
        <classpath refid="class.path"/>
    </java>
  </target>

<target name="clean">
    <delete dir="${dist}"/>
    <delete dir="${build}"/>
    <mkdir dir="${dist}"/>
    <mkdir dir="${build}"/>
  </target>
</project>
```

export.py Python Example

```
from java.io import FileInputStream
from java.io import FileOutputStream
from java.util import ArrayList
from java.util import Collections
from com.bea.wli.sb.util import EnvValueTypes
from com.bea.wli.config.env import EnvValueQuery;
from com.bea.wli.config import Ref
from com.bea.wli.config.customization import Customization
from com.bea.wli.config.customization import FindAndReplaceCustomization

import sys

#===================================================================
# Utility function to load properties from a config file
#===================================================================

def exportAll(exportConfigFile):
    try:
```

```
        print "Loading export config from :", exportConfigFile
        exportConfigProp = loadProps(exportConfigFile)
        adminUrl = exportConfigProp.get("adminUrl")
        exportUser = exportConfigProp.get("exportUser")
        exportPasswd = exportConfigProp.get("exportPassword")

        exportJar = exportConfigProp.get("exportJar")
        customFile = exportConfigProp.get("customizationFile")

        passphrase = exportConfigProp.get("passphrase")
        project = exportConfigProp.get("project")

        connectToServer(exportUser, exportPasswd, adminUrl)

        ALSBConfigurationMBean = findService("ALSBConfiguration",
"com.bea.wli.sb.management.configuration.ALSBConfigurationMBean")
        print "ALSBConfiguration MBean found"

        print project
        if project == None :
            ref = Ref.DOMAIN
            collection = Collections.singleton(ref)
            if passphrase == None :
                print "Export the config"
                theBytes = ALSBConfigurationMBean.export(collection, true,
                 None)
            else :
                print "Export and encrypt the config"
                theBytes = ALSBConfigurationMBean.export(collection, true,
                 passphrase)
        else :
            ref = Ref.makeProjectRef(project);
            print "Export the project", project
            collection = Collections.singleton(ref)
            theBytes = ALSBConfigurationMBean.exportProjects(collection,
             passphrase)

        aFile = File(exportJar)
        out = FileOutputStream(aFile)
        out.write(theBytes)
```

```
            out.close()
            print "ALSB Configuration file: "+ exportJar + " has been exported"

            if customFile != None:
                print collection
                query = EnvValueQuery(None,
                    Collections.singleton(EnvValueTypes.WORK_MANAGER),
                    collection, false, None, false)
                customEnv = FindAndReplaceCustomization('Set the right Work
                     Manager', query, 'Production System Work Manager')
                print 'EnvValueCustomization created'
                customList = ArrayList()
                customList.add(customEnv)
                print customList
                aFile = File(customFile)
                out = FileOutputStream(aFile)
                Customization.toXML(customList, out)
                out.close()

            print "ALSB Dummy Customization file: "+ customFile + " has been
            created"
        except:
            raise

    #===================================================================
    # Utility function to load properties from a config file
    #===================================================================

    def loadProps(configPropFile):
        propInputStream = FileInputStream(configPropFile)
        configProps = Properties()
        configProps.load(propInputStream)
        return configProps

    #===================================================================
    # Connect to the Admin Server
    #===================================================================

    def connectToServer(username, password, url):
        connect(username, password, url)
        domainRuntime()
```

```
# EXPORT script init
try:
    exportAll(sys.argv[1])

except:
    print "Unexpected error: ", sys.exc_info()[0]
    dumpStack()
    raise
```

export.properties Example

```
########################################################################
# OSB Admin Security Configuration                                     #
########################################################################
adminUrl=t3://localhost:7021
exportUser=weblogic
exportPassword=weblogic

########################################################################
# OSB Jar to be exported                                               #
########################################################################
exportJar=export.jar

########################################################################
# Optional passphrase and project name.                                #
# If you specify a project, the script exports all the                 #
# resources contained in the project. If you do not specify a          #
# project, the script exports the entire configuration.                #
########################################################################
passphrase=osb
project=default

########################################################################
# Optional, create a dummy customization file                          #
########################################################################
customizationFile=customize.xml
```

customize.xml Example Customization File

```
<?xml version="1.0" encoding="UTF-8"?>
<cus:Customizations
xmlns:cus="http://www.bea.com/wli/config/customizations"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:xt="http://www.bea.com/wli/config/xmltypes">
  <cus:customization xsi:type="cus:FindAndReplaceCustomizationType">
    <cus:description>Set the right Work Manager</cus:description>
    <cus:query>
      <xt:envValueTypes>Work Manager</xt:envValueTypes>
      <xt:refsToSearch xsi:type="xt:LocationRefType">
        <xt:type>Project</xt:type>
        <xt:path>default</xt:path>
      </xt:refsToSearch>
  <xt:includeOnlyModifiedResources>false</xt:includeOnlyModifiedResources>
      <xt:searchString xsi:nil="true"/>
      <xt:isCompleteMatch>false</xt:isCompleteMatch>
    </cus:query>
    <cus:replacement>Production System Work Manager</cus:replacement>
  </cus:customization>
</cus:Customizations>
```

Exporting a Configuration Using the Command Line

Oracle Service Bus provides a ConfigExport class that you can configure and launch using the following command line arguments. Command line export is for more advanced users who need more flexibility.

```
java -Xms384m -Xmx768m -Dweblogic.home=<weblogic.dir>
-jar
<eclipse.dir>/plugins/org.eclipse.equinox.launcher_<launcher.version>.jar
-data <workspace.dir> -application com.bea.alsb.core.ConfigExport
>-configProject <project.name> -configJar <config.jar>
```

where

- <weblogic.dir> is the location of the installed Oracle WebLogic Server. In a default installation, the location is c:\bea\wlserver_10.3.
- <eclipse.dir> is the location of the installed Eclipse that is linked to the Oracle Service Bus IDE plugins. In a default installation of Oracle Service Bus 10gR3 MP1, the location is c:\bea\tools\eclipse_pkgs\2.0\eclipse_3.3.2\eclipse.
- <launcher.version> is the version of the Eclipse launcher JAR. The version can be found by looking in the <eclipse.dir>\plugins directory for the .jar file that starts with the name

org.eclipse.equinox.launcher. In the default installation of Oracle Service Bus 10gR3 MP1, the version is 1.0.1.R33x_v20080118.

- <workspace.dir> is the location that contains Oraclce Service Bus artifacts to be exported. For example, c:\bea\user_projects\workspaces\default. If this location contains an Eclipse workspace, the workspace is used and the configuration jar is exported from the workspace projects. If this location does not contain a workspace, but instead contains only Eclipse Oracle Service Bus projects, the utility imports the projects into a temporary workspace for the configuration JAR export.
- <project.name> is the name of the Oracle Service Bus Configuration project to be exported. For example, “OSB Configuration.” If you do not specify this artument, the first Oracle Service Bus Configuration Project found in the workspace is exported.
- <config.jar> is the name and location of the Oracle Service Bus Configuration JAR to be exported. For example, c:\sbconfig.jar.

Following is an example of exporting an Oracle Service Bus Configuration from the command line.

```
java -Xms384m -Xmx768m -Dweblogic.home=c:/bea/wlserver_10.3
-jar
c:/bea/tools/eclipse_pkgs/2.0/eclipse_3.3.2/eclipse/plugins/org.eclipse.
equinox.launcher_1.0.1.R33x_v20080118.jar
-data c:/bea/user_projects/workspaces/default
-application com.bea.alsb.core.ConfigExport -configProject “OSB
Configuration” -configJar c:/sbconfig.jar
```

Generating an Effective WSDL

1. In the Project Explorer, find the proxy service or business service from which you want to generate the effective WSDL.
2. Right-click the name of the service and select **Oracle Service Bus > Generate Effective WSDL** from the menu.
3. Select a location and save the file.

Modifying JAR Dependencies

1. In the Project Explorer, find the JAR file whose dependencies you want to modify.

2. Right-click the name of the file and select **Oracle Service Bus > Modify JAR Dependencies** from the menu.
3. Make modifications in the Modify JAR Dependencies dialog.

Importing Resources

In the Oracle Service Bus perspective, select **File > Import** to display the Import wizard. See:

- Import wizard - Config JAR Import - Load Resources page
- Import wizard - Config JAR Import - Review Resources page
- Import wizard - URL Import - Review Resources page
- Import wizard - Config ZIP Import - Load Resources page
- Import wizard - Config ZIP Import - Review Resources page

Using the Command Line or a Script to Import an Oracle Service Bus Configuration

You can use scripting and command-line options for importing an Oracle Service Bus configuration. Use the examples in Exporting a Configuration Using Ant and Exporting a Configuration Using the Command Line for guidance on importing.

The following section provides examples for importing an Oracle Service Bus configuration using the WebLogic Scripting Tool (WLST).

Importing a Configuration Using WLST

You can import an Oracle Service Bus configuration using the WebLogic Scripting Tool (WLST). Following are sample files involved in a WLST import. The Ant buildfile and Python script use a properties file and a customization file.

Ant Buildfile Example

Use the same “Ant Buildfile Example” shown in Exporting a Configuration Using WLST.

import.py Python Example

```
from java.util import HashMap
from java.util import HashSet
from java.util import ArrayList
from java.io import FileInputStream

from com.bea.wli.sb.util import Refs
```

```
from com.bea.wli.config.customization import Customization
from com.bea.wli.sb.management.importexport import ALSBImportOperation

import sys

#=====================================================================
# Entry function to deploy project configuration and resources
#        into a ALSB domain
#=====================================================================

def importToALSBDomain(importConfigFile):
    try:
        SessionMBean = None
        print 'Loading Deployment config from :', importConfigFile
        exportConfigProp = loadProps(importConfigFile)
        adminUrl = exportConfigProp.get("adminUrl")
        importUser = exportConfigProp.get("importUser")
        importPassword = exportConfigProp.get("importPassword")

        importJar = exportConfigProp.get("importJar")
        customFile = exportConfigProp.get("customizationFile")

        passphrase = exportConfigProp.get("passphrase")
        project = exportConfigProp.get("project")

        connectToServer(importUser, importPassword, adminUrl)

        print 'Attempting to import :', importJar, "on ALSB Admin Server
listening on :", adminUrl

        theBytes = readBinaryFile(importJar)
        print 'Read file', importJar
        sessionName = createSessionName()
        print 'Created session', sessionName
        SessionMBean = getSessionManagementMBean(sessionName)
        print 'SessionMBean started session'
        ALSBConfigurationMBean =
findService(String("ALSBConfiguration.").concat(sessionName),
```

```
"com.bea.wli.sb.management.configuration.ALSBConfigurationMBean")
        print "ALSBConfiguration MBean found", ALSBConfigurationMBean
        ALSBConfigurationMBean.uploadJarFile(theBytes)
        print 'Jar Uploaded'

        if project == None:
            print 'No project specified, additive deployment performed'
            alsbJarInfo = ALSBConfigurationMBean.getImportJarInfo()
            alsbImportPlan = alsbJarInfo.getDefaultImportPlan()
            alsbImportPlan.setPassphrase(passphrase)
            alsbImportPlan.setPreserveExistingEnvValues(true)
            importResult =
ALSBConfigurationMBean.importUploaded(alsbImportPlan)
          SessionMBean.activateSession(sessionName, "Complete test import
with customization using wlst")
        else:
            print 'ALSB project', project, 'will get overlaid'
            alsbJarInfo = ALSBConfigurationMBean.getImportJarInfo()
            alsbImportPlan = alsbJarInfo.getDefaultImportPlan()
            alsbImportPlan.setPassphrase(passphrase)
            operationMap=HashMap()
            operationMap = alsbImportPlan.getOperations()
            print
            print 'Default importPlan'
            printOpMap(operationMap)
            set = operationMap.entrySet()

            alsbImportPlan.setPreserveExistingEnvValues(true)

            #boolean
            abort = false
            #list of created ref
            createdRef = ArrayList()

            for entry in set:
                ref = entry.getKey()
                op = entry.getValue()
                #set different logic based on the resource type
```

```
                type = ref.getTypeId
                if type == Refs.SERVICE_ACCOUNT_TYPE or type ==
Refs.SERVICE_PROVIDER_TYPE:
                    if op.getOperation() ==
ALSBImportOperation.Operation.Create:
                     print 'Unable to import a service account or a service
provider on a target system', ref
                        abort = true
                elif op.getOperation() ==
ALSBImportOperation.Operation.Create:
                    #keep the list of created resources
                    createdRef.add(ref)

            if abort == true :
                print 'This jar must be imported manually to resolve the
service account and service provider dependencies'
                SessionMBean.discardSession(sessionName)
                raise

            print
            print 'Modified importPlan'
            printOpMap(operationMap)
            importResult =
ALSBConfigurationMBean.importUploaded(alsbImportPlan)

            printDiagMap(importResult.getImportDiagnostics())

            if importResult.getFailed().isEmpty() == false:
               print 'One or more resources could not be imported properly'
                raise

            #customize if a customization file is specified
            #affects only the created resources
            if customFile != None :
                print 'Loading customization File', customFile
              print 'Customization applied to the created resources only',
createdRef
                iStream = FileInputStream(customFile)
```

```
                customizationList = Customization.fromXML(iStream)
                filteredCustomizationList = ArrayList()
                setRef = HashSet(createdRef)

                # apply a filter to all the customizations to narrow the
target to the created resources
                for customization in customizationList:
                    print customization
                    newcustomization = customization.clone(setRef)
                    filteredCustomizationList.add(newcustomization)

               ALSBConfigurationMBean.customize(filteredCustomizationList)

           SessionMBean.activateSession(sessionName, "Complete test import
with customization using wlst")

        print "Deployment of : " + importJar + " successful"
    except:
        print "Unexpected error:", sys.exc_info()[0]
        if SessionMBean != None:
            SessionMBean.discardSession(sessionName)
        raise

#=====================================================================
# Utility function to print the list of operations
#=====================================================================
def printOpMap(map):
    set = map.entrySet()
    for entry in set:
        op = entry.getValue()
        print op.getOperation(),
        ref = entry.getKey()
        print ref
    print

#=====================================================================
# Utility function to print the diagnostics
#=====================================================================
```

```
def printDiagMap(map):
    set = map.entrySet()
    for entry in set:
        diag = entry.getValue().toString()
        print diag
    print

#=====================================================================
# Utility function to load properties from a config file
#=====================================================================

def loadProps(configPropFile):
    propInputStream = FileInputStream(configPropFile)
    configProps = Properties()
    configProps.load(propInputStream)
    return configProps

#=====================================================================
# Connect to the Admin Server
#=====================================================================

def connectToServer(username, password, url):
    connect(username, password, url)
    domainRuntime()

#=====================================================================
# Utility function to read a binary file
#=====================================================================
def readBinaryFile(fileName):
    file = open(fileName, 'rb')
    bytes = file.read()
    return bytes

#=====================================================================
# Utility function to create an arbitrary session name
#=====================================================================
def createSessionName():
    sessionName =
```

```
String("SessionScript"+Long(System.currentTimeMillis()).toString())
    return sessionName

#=====================================================================
# Utility function to load a session MBeans
#=====================================================================
def getSessionManagementMBean(sessionName):
    SessionMBean = findService("SessionManagement",
"com.bea.wli.sb.management.configuration.SessionManagementMBean")
    SessionMBean.createSession(sessionName)
    return SessionMBean


# IMPORT script init
try:
    # import the service bus configuration
    # argv[1] is the export config properties file
    importToALSBDomain(sys.argv[1])

except:
    print "Unexpected error: ", sys.exc_info()[0]
    dumpStack()
    raise
```

import.properties Example

```
########################################################################
# OSB Admin Security Configuration                                     #
########################################################################
adminUrl=t3://localhost:7021
importUser=weblogic
importPassword=weblogic

########################################################################
# OSB Jar to be exported, optional customization file                  #
########################################################################
importJar=export.jar
customizationFile=OSBCustomizationFile.xml
```

```
########################################################################
# Optional passphrase and project name.                                #
# If you specify a project, the script overlays the existing     #
# project with appropriate updates, creations, and deletions.    #
# If you do not specify a project, the script performs an        #
# additive update with resources created and updated.            #
########################################################################
passphrase=osb
project=default
```

OSBCustomizationFile.xml Example

```
<?xml version="1.0" encoding="UTF-8"?>
<cus:Customizations
xmlns:cus="http://www.bea.com/wli/config/customizations"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xt="http://www.bea.com/wli/config/xmltypes">
  <cus:customization xsi:type="cus:EnvValueCustomizationType">
    <cus:description/>
    <cus:envValueAssignments>
      <xt:envValueType>UDDI Auto Publish</xt:envValueType>
      <xt:location xsi:nil="true"/>
      <xt:owner>
        <xt:type>ProxyService</xt:type>
        <xt:path>default/foo</xt:path>
      </xt:owner>
      <xt:value xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">false</xt:value>
    </cus:envValueAssignments>
    <cus:envValueAssignments>
      <xt:envValueType>Service URI</xt:envValueType>
      <xt:location xsi:nil="true"/>
      <xt:owner>
        <xt:type>ProxyService</xt:type>
        <xt:path>default/foo</xt:path>
      </xt:owner>
      <xt:value xsi:type="xs:string"
xmlns:xs="http://www.w3.org/2001/XMLSchema">/foo/updated</xt:value>
    </cus:envValueAssignments>
```

```
  </cus:customization>
  <cus:customization xsi:type="cus:FindAndReplaceCustomizationType">
    <cus:description/>
    <cus:query>
      <xt:resourceTypes>ProxyService</xt:resourceTypes>
      <xt:envValueTypes>UDDI Auto Publish</xt:envValueTypes>
      <xt:envValueTypes>Service URI</xt:envValueTypes>
      <xt:refsToSearch xsi:type="xt:ResourceRefType">
        <xt:type>ProxyService</xt:type>
        <xt:path>default/foo</xt:path>
      </xt:refsToSearch>

<xt:includeOnlyModifiedResources>false</xt:includeOnlyModifiedResources>
      <xt:searchString>Search String</xt:searchString>
      <xt:isCompleteMatch>false</xt:isCompleteMatch>
    </cus:query>
    <cus:replacement>Replacement String</cus:replacement>
  </cus:customization>
  <cus:customization xsi:type="cus:ReferenceCustomizationType">
    <cus:description/>
  </cus:customization>
</cus:Customizations>
```

Creating Servers

In the Oracle Service Bus perspective, select **File** > **New** > **Server** to display the New Server wizard.

Creating Service Account Resources

In the Oracle Service Bus perspective, select **File** > **New** > **Service Account** to display the New Service Account Resource wizard.

Note: You can create a service account resource in an Oracle Service Bus project only.

Creating Service Key Provider Resources

In the Oracle Service Bus perspective, select **File** > **New** > **Service Key Provider** to display the New Service Key Provider Resource wizard.

Note: You can create a service key provider resource in an Oracle Service Bus project only.

Creating SMTP Server Resources

4. In the Oracle Service Bus perspective, select **File > New > SMTP Server** to display the New SMTP Server Resource wizard. Enter information as appropriate.

Note: You can create an SMTP server resource in an Oracle Service Bus configuration project only.

Creating XQuery Transformations

In the Oracle Service Bus perspective, select **File > New > XQuery Transformation** to display the XQuery/XSLT Expression Editor. Enter information as appropriate.

Note: You can create an XQuery transformation resource in an Oracle Service Bus project only.

Creating XSL Transformations

In the Oracle Service Bus perspective, select **File > New > XSL Transformation** to display the XPath Expression Editor.

Note: You can create an XSL transformation resource in an Oracle Service Bus project only.

Working with Business Services

The following topics describe how to create and work with business services in the Oracle Service Bus plug-ins.

- Creating Business Services
- Editing Business Services

Creating Business Services

In the Oracle Service Bus perspective, select **File > New > Business Service** to display the New Business Service wizard. See:

- Business Service General Configuration page
- Business Service Message Type Configuration page
- Business Service - Service Policy Configuration page

- Business Service SOAP Binding Configuration page
- Business Service Transport Configuration page

Note: You can create a business service in an Oracle Service Bus project only.

Generating a Business Service from an Existing Service

To generate a business service from an existing proxy or business service:

1. In the Project Explorer, right-click the existing service and select **Oracle Service Bus** > **Generate Business Service**.
2. Name and configure the service.

Editing Business Services

1. In the Project Explorer, find the business service you want to edit.
2. Double-click the name of the service.
3. Select the page containing the options you want to edit, and make changes as appropriate. See:
   - Business Service General Configuration page
   - Business Service Message Type Configuration page
   - Business Service - Service Policy Configuration page
   - Business Service SOAP Binding Configuration page
   - Business Service Transport Configuration page

Working with Proxy Services

The following topics describe how to create and work with proxy services in the Oracle Service Bus plug-ins.

- Creating Proxy Services
- Editing Proxy Services

Creating Proxy Services

In the Oracle Service Bus perspective, select **File > New > Proxy Service** to display the New Proxy Service wizard. See:

- Proxy Service General Configuration page
- Proxy Service Message Level Security Configuration page
- Proxy Service Message Type Configuration page
- Proxy Service Operation Selection Configuration page
- Proxy Service - Service Policy Configuration page
- Proxy Service SOAP Binding Configuration page
- Proxy Service Transport Configuration page

Note: You can create a proxy service in an Oracle Service Bus project only.

Generating a Proxy Service from an Existing Service

To generate a proxy service from an existing business or proxy service:

1. In the Project Explorer, right-click the existing service and select **Oracle Service Bus** > **Generate Proxy Service**.
2. Name and configure the service.

For a proxy services generated from a business service, the message flow automatically includes a route node to the business service.

Editing Proxy Services

1. In the Project Explorer, find the proxy service you want to edit.
2. Double-click the name of the service.
3. Select the page containing the options you want to edit, and make changes as appropriate. See:
   - Proxy Service General Configuration page
   - Proxy Service Message Level Security Configuration page
   - Proxy Service Message Type Configuration page
   - Proxy Service Operation Selection Configuration page
   - Proxy Service - Service Policy Configuration page
   - Proxy Service SOAP Binding Configuration page
   - Proxy Service Transport Configuration page

Working with Alert Destinations

The following topics describe how to create and work with alert destinations in the Oracle Service Bus plug-ins.

- Creating Alert Destinations
- Editing Alert Destinations
- Adding E-mail Recipients to Alert Destinations
- Adding JMS Destinations to Alert Destinations

Creating Alert Destinations

1. In the Project Explorer in the Oracle Service Bus perspective, right-click a project or folder in which you want to create an alert destination.
2. From the menu, select **File > New > Alert Destination** to display the New Alert Destination Resource wizard. Enter information as appropriate. See Alert Destination editor.

Note: You can create an alert destination in an Oracle Service Bus project only.

Editing Alert Destinations

1. In the Project Explorer, find the project folder containing the alert destination you want to edit.
2. Double-click the name of the alert destination to display the Alert Destination editor. Edit information, as desired.

Adding E-mail Recipients to Alert Destinations

1. Create or edit an alert destination, as described in Creating Alert Destinations and Editing Alert Destinations.
2. In the **E-mail Recipients** field of the Alert Destination editor, click **Add** to display the Edit E-mail Recipient page. Add information, as appropriate.

Adding JMS Destinations to Alert Destinations

1. Create or edit an alert destination, as described in Creating Alert Destinations and Editing Alert Destinations.

2. In the **JMS Destinations** field of the Alert Destination editor, click **Add** to display the Edit JMS Destination page. Add information, as appropriate.

Working with Proxy Service Message Flows

The following topics describe how to add and configure nodes and actions to proxy service message flows.

- Constructing Proxy Service Message Flows
- Adding and Configuring Alert Actions in Message Flows
- Adding and Configuring Assign Actions in Message Flows
- Adding and Configuring Conditional Branch Nodes in Message Flows
- Adding and Configuring Dynamic Publish Actions in Message Flows
- Adding and Configuring Dynamic Routing Actions in Message Flows
- Adding and Configuring Error Handlers in Message Flows
- Adding and Configuring For-Each Actions in Message Flows
- Adding and Configuring If-Then Actions in Message Flows
- Adding and Configuring Insert Actions in Message Flows
- Adding and Configuring Java Callout Actions in Message Flows
- Adding and Configuring Log Actions in Message Flows
- Adding and Configuring MFL Transform Actions in Message Flows
- Adding and Configuring Operational Branch Nodes in Message Flows
- Adding and Configuring Pipeline Pair Nodes in Message Flows
- Adding and Configuring Publish Actions in Message Flows
- Adding and Configuring Publish Table Actions in Message Flows
- Adding and Configuring Raise Error Actions in Message Flows
- Adding and Configuring Rename Actions in Message Flows
- Adding and Configuring Replace Actions in Message Flows

- Adding and Configuring Report Actions in Message Flows
- Adding and Configuring Replace Actions in Message Flows
- Adding and Configuring Reply Actions in Message Flows
- Adding and Configuring Resume Actions in Message Flows
- Adding and Configuring Route Nodes in Message Flows
- Adding and Configuring Routing Actions in Message Flows
- Adding and Configuring Routing Options Actions in Message Flows
- Adding and Configuring Routing Table Actions in Message Flows
- Adding and Configuring Service Callout Actions in Message Flows
- Adding and Configuring Skip Actions in Message Flows
- Adding and Configuring Stages in Message Flows
- Adding and Configuring Transport Headers Actions in Message Flows
- Adding and Configuring Validate Actions in Message Flows

Constructing Proxy Service Message Flows

When you create a proxy service, a message flow is created by default, with an empty starting node. The process for constructing the message flow follows this general pattern:

1. Open the Message Flow Editor for the proxy service. To open the proxy service, double-click its name in Project Explorer. The Message Flow Editor appears as a tab in the proxy service view.
2. Open the Message Flow Design Palette. To open the palette, in the Oracle Service Bus perspective, select **Window > Show View > Design Palette.**
3. Open the Properties view, if it is not already open:
   a. In the Oracle Service Bus perspective, select **Window > Show View > Other**.
   b. In the Show View dialog, select **General > Properties**.
4. Drag nodes and actions from the Message Flow Design Palette to the Message Flow Editor.

Alternatively, you can right-click a node or action in the Message Flow Editor to display menus of nodes and actions that can be inserted in that location. The menu contains one or more the following:

- **Insert > (list of nodes and actions)**
- **Insert Into > (list of nodes and actions)**
- **Insert After > (list of nodes and actions)**
- **Add Error Handler**

5. Configure nodes and actions:

   a. In the Proxy Service Editor, select the node or action by clicking it.

      Alternatively, you can select a node or an action from the Outline view. To open the Outline view, in the Oracle Service Bus perspective, select **Window > Show View > Outline.**

   b. In the Properties view, set the properties, as appropriate for the selected node or action. For instructions on how to configure the nodes and actions, click the Properties view for a node or action, and press F1 for help.

Adding and Configuring Alert Actions in Message Flows

Use the alert action to generate alerts based on message context in a pipeline, to send to an alert destination.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an alert action

1. In the Message Flow Design Palette, open the **Stage Actions > Reporting** list, if it is not already open.
2. Drag the alert action to the desired location in the message flow.

To configure the alert action

1. In the Message Flow Editor, click the alert action, if it is not already selected.
2. On the Alert Action Properties page, edit properties as appropriate.

Adding and Configuring Assign Actions in Message Flows

Use an assign action to assign the result of an XQuery expression to a context variable.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an assign action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the assign action to the desired location in the message flow.

To configure the assign action

1. In the Message Flow Editor, click the assign action, if it is not already selected.
2. On the Assign Action Properties page, edit properties as appropriate.

Adding and Configuring Conditional Branch Nodes in Message Flows

Use a conditional branch node to specify that message processing is to proceed along exactly one of several possible paths, based on a result returned by an XPath condition.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a conditional branch node

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the conditional branch node to the desired location in the message flow.

To configure the conditional branch node

1. In the Message Flow Editor, click the conditional branch node, if it is not already selected.
2. On the Conditional Branch Node Properties page, edit properties as appropriate.x

Adding and Configuring Delete Actions in Message Flows

Use a delete action to delete a context variable or a set of nodes specified by an XPath expression.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a delete action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the delete action to the desired location in a stage action in the message flow.

To configure the delete action

1. In the Message Flow Editor, click the delete action, if it is not already selected.
2. On the Delete Action Properties page, edit properties as appropriate.

Adding and Configuring Dynamic Publish Actions in Message Flows

Use a dynamic publish action to publish a message to a service specified by an XQuery expression.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a dynamic publish action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the dynamic publish action to the desired location in the message flow.

To configure the dynamic publish action

1. In the Message Flow Editor, click the dynamic publish action, if it is not already selected.
2. On the Dynamic Publish Action Properties page, edit properties as appropriate.

Adding and Configuring Dynamic Routing Actions in Message Flows

Use a dynamic routing action to assign a route for a message based on routing information available in an XQuery resource.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a dynamic routing action

1. In the Message Flow Design Palette, open the **Route Actions > Communication** list, if it is not already open.
2. Drag the dynamic routing action to the route action in the message flow.

To configure the dynamic routing action

1. In the Message Flow Editor, click the dynamic routing action, if it is not already selected.
2. On the Dynamic Routing Action Properties page, edit properties as appropriate.

Adding and Configuring Error Handlers in Message Flows

Use an error handler to specify what should happen if an error occurs in a specific location in the message flow.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an error handler

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the error handler to the desired location in the message flow.
3. Drag a stage node to the error handler.
4. Add actions to the stage, as appropriate, to define the error handler.

To configure the error handler

1. In the Message Flow Editor, click the error handler, if it is not already selected.
2. On the Error Handler Node Properties page, edit the properties.
3. Click the stage node, if it is not already selected.
4. On the Stage Node Properties page, edit the properties.
5. Select and edit any actions contained by the stage, as appropriate.

Adding and Configuring For-Each Actions in Message Flows

Use the for-each action to iterate over a sequence of values and execute a block of actions.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a for-each action

1. In the Message Flow Design Palette, open the **Stage Actions > Flow Control** list, if it is not already open.
2. Drag the for-each action to the desired stage action in the message flow.

To configure the for-each action

1. In the Message Flow Editor, click the for-each action, if it is not already selected.
2. On the For-Each Action Properties page, edit properties as appropriate.

Adding and Configuring If-Then Actions in Message Flows

Use an if-then action to perform an action or a set of actions conditionally, based on the Boolean result of an XQuery expression.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an if-then action

1. In the Message Flow Design Palette, do one of the following:

  - For an if-then action in a route node, open the **Route Actions > Flow Control** list, if it is not already open.
  - For an if-then action in a stage node, open the **Stage Actions > Flow Control** list, if it is not already open.

2. Drag the if-then action to the route node or to the desired stage action in the message flow.

To configure the if-then action

1. In the Message Flow Editor, click each if condition and else-if condition contained by the if-then action, and define the conditions in the Condition Editor, as described in If-Then Action Properties.

Adding and Configuring Insert Actions in Message Flows

Use an insert action to insert the result of an XQuery expression at an identified place relative to nodes selected by an XPath expression.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an insert action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the insert action to the desired location in the message flow.

To configure the insert action

1. In the Message Flow Editor, click the insert action, if it is not already selected.
2. On the Insert Action Properties page, edit properties as appropriate.

Adding and Configuring Java Callout Actions in Message Flows

Use a Java callout action to invoke a Java method or an EJB business service from within the message flow.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an Java callout action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the Java callout action to the desired location in the message flow.

To configure the Java callout action

1. In the Message Flow Editor, click the Java callout action, if it is not already selected.
2. On the Java Callout Action Properties page, edit properties as appropriate.

Adding and Configuring Log Actions in Message Flows

Use the log action to construct a message to be logged and to define a set of attributes with which it will be logged.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a log action

1. In the Message Flow Design Palette, open the **Stage Actions > Reporting** list, if it is not already open.
2. Drag the log action to the desired location in the message flow.

To configure the log action

1. In the Message Flow Editor, click the log action, if it is not already selected.
2. On the Log Action Properties page, edit properties as appropriate.

Adding and Configuring MFL Transform Actions in Message Flows

Use a MFL (Message Format Language) transform action to convert message content from XML to non-XML, or vice versa, in the message pipeline.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a MFL transform action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the MFL transform action to the desired location in the message flow.

To configure the MFL transform action

1. In the Message Flow Editor, click the MFL transform action, if it is not already selected.
2. On the MFL Transform Action Properties page, edit properties as appropriate.

Adding and Configuring Operational Branch Nodes in Message Flows

Use an operational branch node to configure branching based on operations defined in a WSDL.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an operational branch node

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the operational branch node to the desired location in the message flow.

To configure the operational branch node

1. In the Message Flow Editor, click the operational branch node, if it is not already selected.
2. On the Operational Branch Node Properties page, edit properties as appropriate.

Adding and Configuring Pipeline Pair Nodes in Message Flows

Use a pipeline pair node to define request and response processing.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a pipeline pair node

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the pipeline pair node to the desired location in the message flow.

To configure the pipeline pair node

1. In the Message Flow Editor, click the pipeline pair node, if it is not already selected.
2. On the Pipeline Pair Node Properties page, edit properties as appropriate.

Adding and Configuring Publish Actions in Message Flows

Use a publish action to identify a statically specified target service for a message and to configure how the message is packaged and sent to that service.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a publish action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the publish action to the desired location in the message flow.

To configure the publish action

1. In the Message Flow Editor, click the publish action, if it is not already selected.
2. On the Publish Action Properties page, edit properties as appropriate.

Adding and Configuring Publish Table Actions in Message Flows

Use a publish table action to publish a message to zero or more statically specified services.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a publish table action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the publish table action to the desired location in the message flow.

To configure the publish table action

1. In the Message Flow Editor, click the publish table action, if it is not already selected.
2. On the Publish Table Action Properties page, click **<Expression>** to display the XQuery/XSLT Expression Editor. Create an XQuery expression, which at run time returns the value upon which the routing decision will be made.
3. In the Message Flow Editor, select a case action.
4. From the **Operator** list on the Publish Table Action Properties page, select a comparison operator. Then, in the **Value** field, enter a value against which the value returned from the XQuery expression will be evaluated.
5. In the Message Flow Editor, click one of the publish table's publish actions to select it.
6. On the Publish Action Properties page, click **Browse** to select a service. Select the service to which messages are to be published if the expression evaluates true for the value you specified. The **Select a Service Resource** dialog is displayed.
7. Select a service from the list, then click **OK**. This is the target service for the message.
8. If the service has operations defined, you can specify the operation to be invoked by selecting it from the **invoking** list.
9. If you want the outbound operation to be the same as the inbound operation, select the **Use inbound operation for outbound** check box.
10. In the **Request Actions** field, to configure how the message is packaged and sent to the service, click **Add an Action**, then select one or more actions that you want to associate with the service. To learn more about the type of action you want to add, see "Adding and Editing Actions in Message Flows" on page 18-1.

    Note: There is a nesting limit of four cumulative levels in the stage editor. If you attempt to add a fifth level, nesting action is not displayed. Cumulative levels include all branching actions: if... then... conditions, publish tables, and route tables. For

example, you can have 2 levels of conditionals, then a publish table with a route table inside of it, bringing the total to 4 levels. If you attempt to add another conditional (to the last publish table), the conditional is not displayed.

11. To insert a new case, click the **Case** icon, then select **Insert New Case**.
12. Repeat steps 4-8 for the new case.
13. Add additional cases as dictated by your business logic.
14. Click the **Case** icon of the last case you define in the sequence, then select **Insert Default Case** to add a default case at the end.
15. Configure the default case—the configuration of this case specifies the routing behavior in the event that none of the preceding cases is satisfied.

Adding and Configuring Raise Error Actions in Message Flows

Use the raise error action to raise an exception with a specified error code (a string) and description.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a raise error action

1. In the Message Flow Design Palette, open the **Stage Actions > Flow Control** list, if it is not already open.
2. Drag the raise error action to the desired location in the message flow.

To configure the raise error action

1. In the Message Flow Editor, click the raise error action, if it is not already selected.
2. On the Raise Error Action Properties page, edit properties as appropriate.

Adding and Configuring Rename Actions in Message Flows

Use the rename action to rename elements selected by an XPath expression without modifying the contents of the element.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add an rename action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the rename action to the desired location in the message flow.

To configure the rename action

1. In the Message Flow Editor, click the rename action, if it is not already selected.
2. On the Rename Action Properties page, edit properties as appropriate.

Adding and Configuring Replace Actions in Message Flows

Use a replace action to replace a node or the contents of a node specified by an XPath expression. The node or its contents are replaced with the value returned by an XQuery expression.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a replace action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the replace action to the desired location in the message flow.

To configure the replace action

1. In the Message Flow Editor, click the replace action, if it is not already selected.
2. On the Replace Action Properties page, edit properties as appropriate.

Adding and Configuring Reply Actions in Message Flows

Use the reply action to specify that an immediate reply be sent to the invoker.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a reply action

1. In the Message Flow Design Palette, open the **Stage Actions > Flow Control** list, if it is not already open.
2. Drag the reply action to the desired location in the message flow.

To configure the reply action

1. In the Message Flow Editor, click the reply action, if it is not already selected.
2. On the Reply Action Properties page, edit properties as appropriate.

Adding and Configuring Report Actions in Message Flows

Use the report action to enable message reporting for a proxy service.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a report action

1. In the Message Flow Design Palette, open the **Stage Actions > Reporting** list, if it is not already open.
2. Drag the report action to the desired location in the message flow.

To configure the report action

1. In the Message Flow Editor, click the report action, if it is not already selected.
2. On the Report Action Properties page, edit properties as appropriate.

Adding and Configuring Resume Actions in Message Flows

Use the resume action to resume message flow after an error is handled by an error handler. This action has no parameters and can only be used in error pipelines.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a resume action

1. In the Message Flow Design Palette, open the **Stage Actions > Flow Control** list, if it is not already open.
2. Drag the resume action to the desired location in the message flow.

To configure the resume action

1. In the Message Flow Editor, click the resume action, if it is not already selected.
2. On the Resume Action Properties page, edit properties as appropriate.

Adding and Configuring Route Nodes in Message Flows

Use the route node to handle request and response dispatching of messages to and from business services.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a route node

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the route node to the desired location in the message flow.

To configure the route node

1. In the Message Flow Editor, click the route node action, if it is not already selected.
2. On the Route Node Properties page, edit properties as appropriate.

Adding and Configuring Routing Actions in Message Flows

Use a routing action to identify a target service for the message and configure how the message is routed to that service.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a routing action

1. In the Message Flow Design Palette, open the **Route Actions > Communication** list, if it is not already open.
2. Drag the routing action to the desired location in the message flow.

To configure the routing action

1. In the Message Flow Editor, click the routing action, if it is not already selected.
2. On the Routing Action Properties page, edit properties as appropriate.

Adding and Configuring Routing Options Actions in Message Flows

Use a routing options action to modify any or all of the following properties in the outbound request: URI, Quality of Service, Mode, Retry parameters, Message Priority.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a routing options action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the routing options action to the desired location in the message flow.

To configure the routing options action

1. In the Message Flow Editor, click the routing options action, if it is not already selected.
2. On the Routing Options Action Properties page, edit properties as appropriate.

Adding and Configuring Routing Table Actions in Message Flows

Use a routing table to select different routes based upon the results of a single XQuery expression. A routing table action contains a set of routes wrapped in a switch-style condition table.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a routing table action

1. In the Message Flow Design Palette, open the **Route Actions > Communication** list, if it is not already open.
2. Drag the routing table action to the desired location in the message flow.

To configure the routing table action

1. In the Message Flow Editor, click the routing table action, if it is not already selected.
2. On the Routing Table Action Properties page, edit properties as appropriate.

Adding and Configuring Service Callout Actions in Message Flows

Use a service callout action to configure a synchronous (blocking) callout to an Oracle Service Bus-registered proxy or business service.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a service callout action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the service callout action to the desired location in the message flow.

To configure the service callout action

1. In the Message Flow Editor, click the service callout action, if it is not already selected.
2. On the Service Callout Action Properties page, edit properties as appropriate.

Adding and Configuring Skip Actions in Message Flows

Use the skip action to specify that at run time, the execution of the current stage is skipped and the processing proceeds to the next stage in the message flow.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a skip action

1. In the Message Flow Design Palette, open the **Stage Actions > Flow Control** list, if it is not already open.
2. Drag the skip action to the desired location in the message flow.

To configure the skip action

1. In the Message Flow Editor, click the skip action, if it is not already selected.
2. On the Skip Action Properties page, edit properties as appropriate.

Adding and Configuring Stages in Message Flows

Use a stage node as a container for actions in a message flow.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a stage

1. In the Message Flow Design Palette, open the **Oracle Service Bus Message Flow > Nodes** list, if it is not already open.
2. Drag the stage to the desired location in the message flow.
3. Add actions to the stage, as appropriate for your configuration.

To configure the stage

1. In the Message Flow Editor, click the stage, if it is not already selected.
2. On the Stage Node Properties page, edit properties as appropriate.

Adding and Configuring Transport Headers Actions in Message Flows

Use a transport header action to set header values in messages.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a transport headers action

1. In the Message Flow Design Palette, open the **Stage Actions > Communication** list, if it is not already open.
2. Drag the transport headers action to the desired location in the message flow.

To configure the transport headers action

1. In the Message Flow Editor, click the transport headers action, if it is not already selected.
2. On the Transport Headers Action Properties page, edit properties as appropriate.

Adding and Configuring Validate Actions in Message Flows

Use a validate action to validate elements selected by an XPath expression against an XML schema element or a WSDL resource.

Before you begin

Display the message flow for the desired proxy service. See Constructing Proxy Service Message Flows.

To add a validate action

1. In the Message Flow Design Palette, open the **Stage Actions > Message Processing** list, if it is not already open.
2. Drag the validate action to the desired location in the message flow.

To configure the validate action

1. In the Message Flow Editor, click the validate action, if it is not already selected.
2. On the Validate Action Properties page, edit properties as appropriate.

Working with MQ Connections

MQ connections are sharable resources that can be reused across multiple MQ proxy and business services. MQ proxy and business services must connect to a MQ queue manager before accessing an MQ queue. MQ Connection resources provide the connection required for connecting to an MQ queue manager.

Each MQ Connection resource has a connection pool. Every business or proxy service using a given MQ Connection resource to get a connection to a given queue manager uses the same

connection pool that was created for that resource. Thus, multiple business services and proxy services using the same queue manager share a connection pool.

To learn more about Oracle Service Bus MQ Connection resources and native MQ transports, see the *Native MQ Transport User Guide*.

To learn more about WebSphere MQ Fundamentals, see http://www.redbooks.ibm.com/redbooks/SG247128/wwhelp/wwhimpl/java/html/wwhelp.htm.

Adding MQ Connections

In Oracle Service Bus, MQ connections are created as custom resources. Therefore, to add an MQ connection, you must create it as a custom resource, as follows:

1. In the Oracle Service Bus perspective, select **File > New > Custom Resource** to display New Custom Resource wizard.
2. On the Create a New Custom Resource page, in the **Resource Type** field, select **MQ Connection**.
3. Enter configuration information, as appropriate, on the Custom MQ Resource Configuration page.

   Note: Do not include spaces in the MQ Connection resource name.

Editing MQ Connections

1. In the Project Explorer, find the Oracle Service Bus configuration folder containing the MQ connection resource you want to edit.
2. Double-click the name of the MQ connection to display the Custom MQ Resource Configuration page. Edit as appropriate.

Working with UDDI Registries

Universal Description, Discovery and Integration (UDDI) registries are used in an enterprise to share Web Services. UDDI provides a framework in which to classify your business, its services, and the technical details about the services you want to expose.

Publishing a service to a registry requires knowledge of the service type and the data structure representing that service in the registry. A registry entry has certain properties associated with it and these property types are defined when the registry is created. You can publish your service to a registry and make it available for other organizations to discover and use. Proxy services

developed in Oracle Service Bus can be published to a UDDI registry. Oracle Service Bus can interact with any UDDI version 3.0-compliant registry.

See also UDDI in the *Oracle Service Bus User Guide*.

Adding UDDI Registries

1. In the Oracle Service Bus perspective, select **File > New > UDDI Registry**. to display the New UDDI Resource wizard.

   Note: You can add a UDDI registry only to an Oracle Service Bus configuration project only.

2. After choosing a location and entering a name, configure the registry on the UDDI Registry Configuration page.

Note: You can add a UDDI registry in an Oracle Service Bus configuration project only.

Configuring UDDI Registries

1. In the Project Explorer, find the UDDI registry you want to configure.
2. Double-click the name of the registry.
3. Set configure options on the UDDI Registry Configuration page.

Importing Business Services From a UDDI Registry

1. Create a business service, as described in “Creating Business Services” on page 2-23.
2. In the New Business Service wizard Business Service General Configuration page, select **WSDL Web Service**. then click **Browse**.
3. In the Select a WSDL dialog, click **Consume**.
4. In the Service Consumption dialog, select **UDDI**.

   For help using UDDI Register Access in the Service Consumption dialog, press F1, or search in the help system for “Service Consumption dialog.“

Working with Split-Join

This section provides instructions for creating and configuring Split-Joins. Following are the primary topics in this section:

Introduction to Split-Join

The Split-Join is a mediation pattern that can be used by a transport typed business service in an Oracle Service Bus message flow. Split-Join allows you to send message requests to multiple services concurrently, thus enhancing performance in comparison to sending them sequentially. Split-Join achieves this task by splitting an input message into individual messages, routing them concurrently to their destinations, and then aggregating the responses into one overall message.

You design a Split-Join in the Workshop for WebLogic Split-Join editor, then export it to the Oracle Service Bus console for testing and production.

Note: In the Oracle Service Bus console, a Split-Join is associated with a business service using the Flow transport protocol. Therefore, the Split-Join has a .flow file name extension in Workshop for WebLogic even though it is always referred to simply as a "Split-Join" in this document.

There are two types of Split-Join pattern: static Split-Join and dynamic Split-Join, as described in Designing a Split-Join.

For more information on invoking a business service from an Oracle Service Bus message flow, see Proxy Services: Message Flow in the Oracle Service Bus console help system.

Using Split-Join with Content in SOAP Headers

You can use Split-Join to enhance the performance of services that put message content in the SOAP header. Proxy services can pass SOAP headers into Split-Joins, and Split-Joins can pass SOAP headers to proxy and business services as an invocation or response.

To enable this capability, you must declare the header parts along with the body parts in a single request/response message in the Split-Join WSDL and in the WSDL of the proxy or business services invoked by the Split-Join. With the message parts declared in the WSDLs, SOAP header content is available to Split-Joins in the request/response message variables.

Following is an example of the message and binding definitions in the WSDL.

Message

```
<wsdl:message name="retrieveCustomerOverviewByIdRequestMessage">
    <wsdl:part name="retrieveCustomerOverviewByIdRequest"
     element="co:retrieveCustomerOverviewByIdRequest"/>
    <wsdl:part name="serviceContext" element="sc:serviceContext"/>
</wsdl:message>
```

Binding

```
<wsdl:input>
<soap:body use="literal" parts="retrieveCustomerOverviewByIdRequest"/>
    <soap:header message="tns:retrieveCustomerOverviewByIdRequestMessage"
     part="serviceContext" use="literal"/>
</wsdl:input>
```

Designing a Split-Join

There are two Split-Join patterns, the Static Split-Join and the Dynamic Split Join.

The Static Split-Join can be used to create a fixed number of message requests (as opposed to an unknown number). For instance, a customer places an order for a cable package that includes three separate services: internet service, TV service, and telephone service. In the Static use case, you could execute all three requests in separate parallel branches to improve performance time over the standard sequential execution.

The Dynamic Split-Join can be used to create a variable number of message requests. For instance, a retailer places a batch order containing a variable number of individual purchase orders. In the Dynamic use case, you could parse the batch order and create a separate message request for each purchase. Like the Static use case, these messages can then be executed in parallel for improved performance.

Initial Setup

Split-Joins potentially include the following tasks as part of their initial setup:

Creating/Importing a WSDL Containing the Base Operation

Every Split-Join is based upon a WSDL operation. When you first create a Split-Join, you will be asked to browse to the appropriate WSDL file and to select this operation as part of the creation process. You can create this WSDL file in Workshop for WebLogic.

Creating/Importing a Business Service to Use the Split-Join

Every Split-Join will be used by a transport typed business service, which, in turn, is invoked by a proxy service. You cannot export or test your Split-Join until you have created this business service. If it already exists, you can import it into Workshop for WebLogic, or, if it does not exist, you can create it in Workshop for WebLogic or the Oracle Service Bus console. If you want to get started on your Split-Join before you create the business service, you can generate the business service automatically after you create the Split-Join.

Designing a Static Split-Join

Suppose you want to design a new Split-Join called *Service Availability* that handles orders for a telco's cable service package including TV, phone, and internet service. The idea is for the Split-Join to receive an incoming package order and to reply with an order acknowledgement for each type of service. In this case, *Service Availability* is designed as a *Static* Split-Join because there are three message requests per order, one for each type of service. In this particular example the customer requests only TV and DSL service.

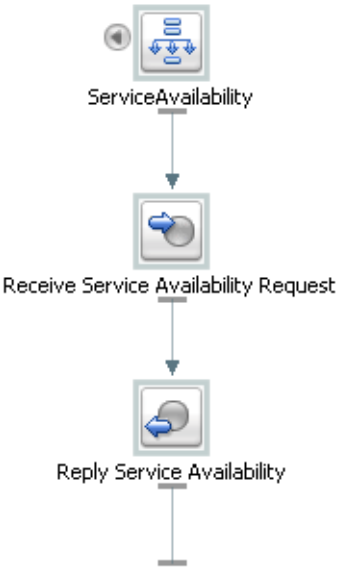
Creating the *Service Availability* Split-Join may include the following tasks:

1. Creating a New Split-Join
2. Adding an Assign
3. Adding a Parallel Node
4. Adding an Assign for Each Branch
5. Adding an Invoke Service
6. Adding an Assign for Each Branch
7. Exporting and Testing the Split-Join

1. Creating a New Split-Join

Create a new Split Join based on the WSDL operation you want to use for placing the order. In this case the WSDL operation we want is called "telecom."

After you select the WSDL operation, a skeleton of the newly created Split-Join appears in the Split-Join editor, as shown in the following figure. It consists of a Start Node, a Receive, a Reply. The labels are then edited in the general properties tab to better reflect the specific function of each node in this particular Split-Join.



The Start Node contains both a Request Variable and a Response Variable that have been determined by the WSDL operation initially selected. The Receive receives the incoming request message (in this case for the three or fewer different kinds of cable service), and the Reply generates a response message and sends it back to the client.

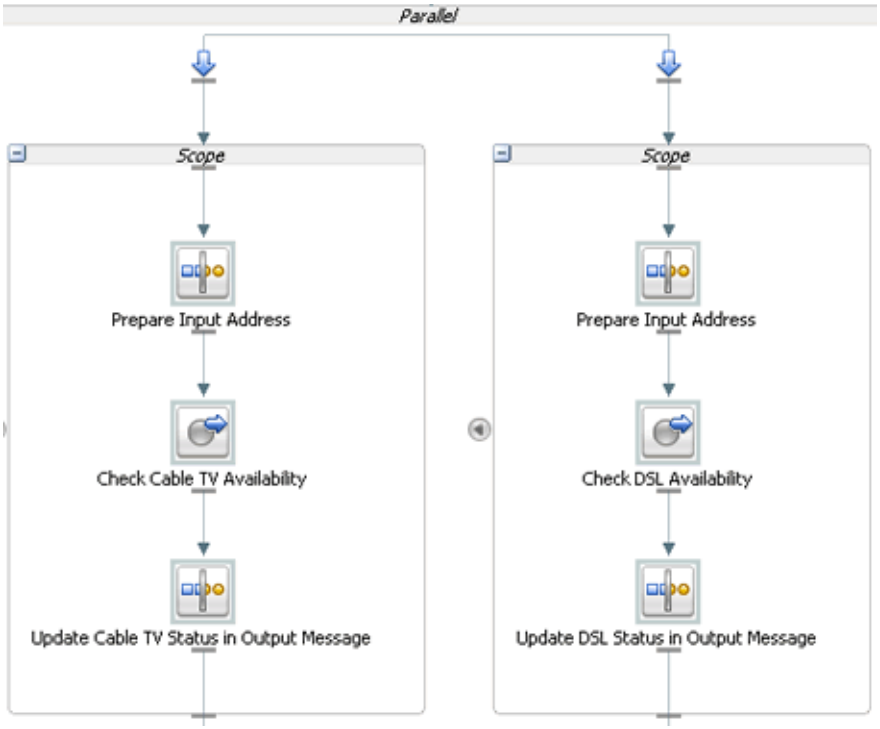
Note: The Receive node requires no further configuration. Similarly the Reply requires no further configuration, unless to generate an error fault--which is not the case in this scenario (see Configuring a Reply for more information on generating faults).

2. Adding an Assign

The first Assign, *Prepare Output Message*, contains an Assign operation that prepares the Response Variable in a form such that the later nodes can work on the data within it (that is, Copy/Insert/Assign/Replace/Delete/Java Callout/Log the data). This output message is relayed to the final Reply node in the Split-Join and, in turn, returned to the original client.

3. Adding a Parallel Node

The Parallel Node contains two main branches, one to check cable TV availability and one to check DSL availability. Each branch is composed of a number of actions, the sequence and general configuration being the same for both branches.



4. Adding an Assign for Each Branch

The first Assign in each Parallel Branch, Prepare Input Address, copies the incoming customer address data into a Variable that is referenced to check the availability of the service at that location. The Assigns are the same for each branch and would be for additional branches as well.

5. Adding an Invoke Service

An External Service is then invoked to check whether the requested service type is available at the customer's location. Each branch contains one Invoke Service, *Check Cable TV Availability* and *Check DSL Availability*. Each invocation calls an External Service, which compares the customer's address (stored in the Variable initialized in the previous step) to the availability of

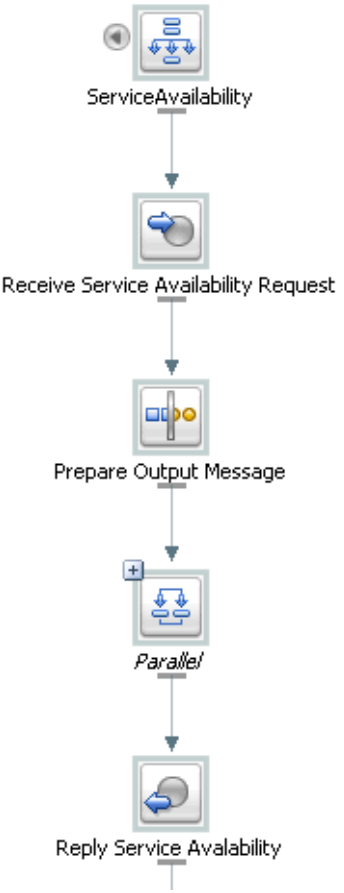
the service at that location. The result is then stored in an output Variable that is passed on to the final Assign in the Branch below.

6. Adding an Assign for Each Branch

The final two Assigns, *Update Cable TV Status in Output Message* and *Update DSL Status in Output Message*, take the results of the external service invocations and put them into the output message using a Replace operation. The aggregated response are then sent to the original client in the final Reply node, which requires no further configuration.

7. Exporting and Testing the Split-Join

After you design the Split-Join, you can export it to the Oracle Service Bus console for testing and production.



Related Topics

- Creating a New Split-Join
- Configuring the Start Node
- Creating an Assign
- Invoking a Service
- Creating a Parallel
- Exporting and Testing a Split-Join

Designing a Dynamic Split-Join

Suppose that you want to design a Split-Join that handles a batch order from a retailer containing a variable number of individual purchase orders (as opposed to a fixed number of orders). The idea is for the Split-Join to receive the batch order and to reply with an order acknowledgement for each order within. This would be a *Dynamic* Split-Join because the number of individual purchase order requests is variable and unknown at design time.

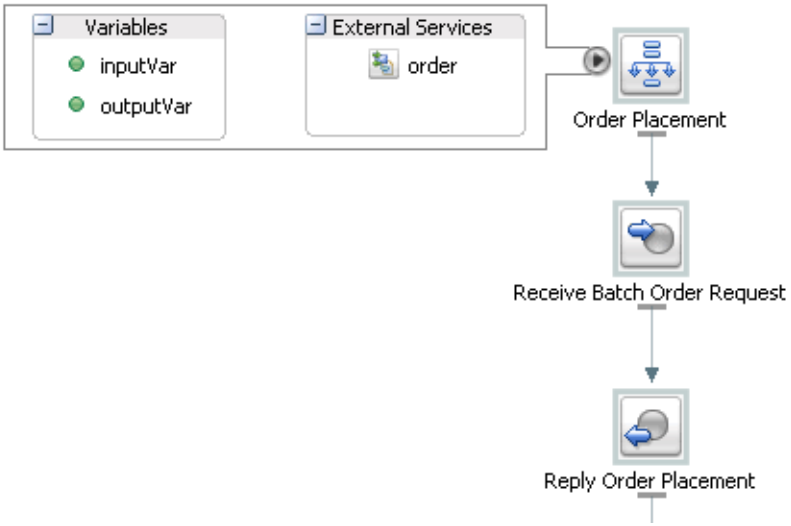
Creating this Split-Join may include the following tasks:

1. Creating a New Split-Join
2. Adding an Assign
3. Adding a For Each
4. Adding an Assign
5. Adding an Invoke Service
6. Adding an Assign
7. Adding an Error Handler
8. Exporting and Testing the Split-Join

1. Creating a New Split-Join

Create a new Split Join based off of the WSDL operation you want to use for placing the order. In this case the WSDL operation we want is called “batchOrders.”

After the operation is selected, a skeleton of the newly created Split-Join appears in the Split-Join editor consisting of a Start Node, a Receive, a Reply. The labels are then edited in the general properties tab to better reflect the specific function of each node in this particular Split-Join.



The Start Node**,** *Order Placement*, contains both a request variable, inputVar, and an response variable, outputVar. The Receive**,** *Receive Batch Order Request***,** will initialize the contents of the Request Variable (in this case purchase orders), and the Reply**,** *Reply Order Placement***,** will send a response, based on the order acknowledgements aggregated into the Response Variable, back to the client. In this example *Order Placement* also contains a callout to an External Service, "Order" that will be invoked to approve individual orders.

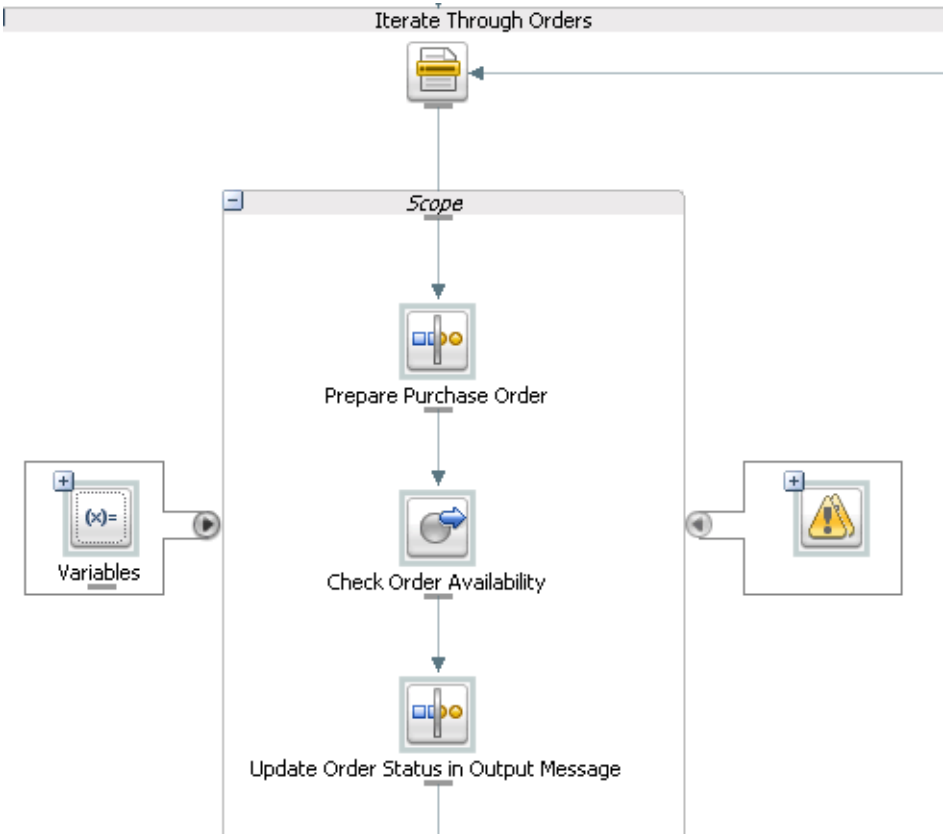
Note: The Receive node requires no further configuration. Similarly, the Reply requires no further configuration, unless you would like to generate an error fault—which is not the case in this scenario (see Configuring a Reply for more information on generating faults).

2. Adding an Assign

The first Assign, *Prepare Output Message*, contains an Assign operation that prepares the response variable (here labeled an "Output Message" for readability) in a form such that the later nodes can work on the data captured within it (that is, Copy/Insert/Assign/Replace/Delete into the Variable). In this case, that data would consist of order acknowledgments and/or errors.This Output Message is relayed to the final Reply node in the Split-Join and, in turn, returned to the original client.

3. Adding a For Each

The For Each, *Iterate Through Orders*, contains logic that will parse through each order in the batch, send it to an external proxy for approval, and capture an order acknowledgment in response. If there is a problem with an order, an error is sent from the invoked proxy and captured in the Error Handler. The following figure depicts the entire scope of the For Each logic.



4. Adding an Assign

The Assign, *Prepare Purchase Order*, copies the incoming purchase order requests into a variable that is referenced to check approval of the order in the next step.

5. Adding an Invoke Service

An external service, *Check Order Availability*, is then invoked to approve each individual purchase order. If the order is accepted, the service responds with an order acknowledgment. If the order is not accepted, the service responds with an error.The result is then stored in an output variable that is passed on to the final Assign in the next step.

6. Adding an Assign

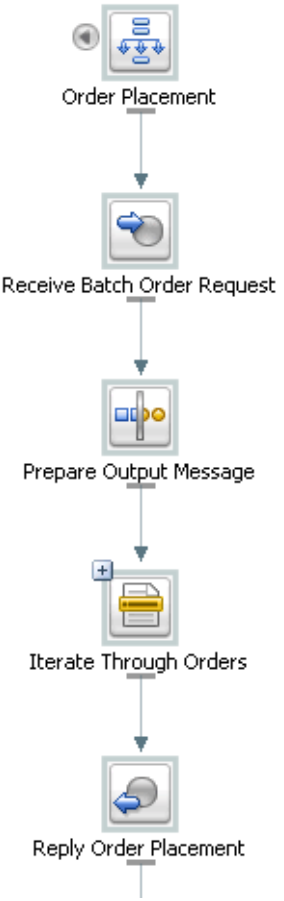
The final Assign, *Update Order Status in Output Message*, takes the results of the external service invocation and copies them into the output message using an Insert operation. The aggregated response is then sent to the original client in the final Reply node, which requires no further configuration.

7. Adding an Error Handler

The Error Handler captures any Errors returned by the invoked service. It takes these errors and inserts them into the output message using an Assign operation.

8. Exporting and Testing the Split-Join

After you design the Split-Join, you can export it to the Oracle Service Bus console for testing and production.



Related Topics

- Creating a New Split-Join
- Configuring the Start Node
- Creating an Error Handler
- Creating a Parallel
- Configuring a Reply
- Exporting and Testing a Split-Join

Creating a New Split-Join

In order to create a new Split/Join, you must have access to a WSDL containing the operation upon which to base the Split-Join. The Split Join must be created in an existing Oracle Service Bus project within an existing Oracle Service Bus configuration project.

To create a new Split-Join:

1. In the Oracle Service Bus perspective, select **File > New > Split-Join**. This opens the New Split-Join Wizard.
2. In the New Split-Join Wizard, type or select an Oracle Service Bus project location and enter a file name for the new Split-Join. When you have finished, click Next.
3. In the next screen, you must select a binding and then an operation on which to implement the Split-Join. There are two ways to make your selection:
   a. Choose your operation from one of the WSDLs displayed in the Select Operation tree. All of the WSDLs in your current Oracle Service Bus configuration project are available.
   b. Import your WSDL into the Oracle Service Bus configuration project using the Consume button. Consumption imports a new WSDL into your configuration from an outside source, as described in the following step.
4. If you choose to consume the base WSDL, go through the following steps:
   a. Click **Consume**.
   b. Browse for the location, or "Artifact Folder," wherein you wish to generate the consumed WSDL. The default artifact folder is your current Oracle Service Bus project.
   c. If you want to overwrite existing local files, select the checkbox.
   d. Choose the service resource in which the WSDL to be consumed resides: Enterprise Repository, File System, UDDI, URI, or Workspace.
   e. Select The WSDL that you want to consume from that Service Resource. After you have made your selection the workspace will rebuild momentarily and the Service Consumption Status dialog will appear depicting the status of your consume. If it was successful, click **OK** to close the dialog.
   f. The consumed WSDL is now in your Oracle Service Bus configuration project, and you can select an operation from it upon which to base your Split-Join.
5. Click **Finish**.

A basic Split-Join is created and visually represented as a diagram in the Design View. By default, it consists of a Start Node, a Receive, and a Reply. The Start Node contains the Request and Response Variables introspected from the WSDL operation. The Receive is used to receive incoming request messages. The Reply is used to send response messages.

Related Topics

- Configuring the Start Node
- Configuring a Receive
- Creating a Parallel
- Split-Join Wizard - New Split Join

Configuring the Start Node

The Start Node is generated automatically whenever you create a new Split-Join. It is the starting point from which all the other nodes proceed. Configuring a Start Node can include the following tasks:

- Add General Information
- Define Global Variables
- View External Services

Related Topics

- Invoking a Service
- Start Node Properties

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Defining Global Variables

Variables in the Start Node store data that can be referenced globally, that is by any node in the Split-Join. By default, every Start Node is assigned both a request and a response variable when the Split-Join is initially created. From the Start Node, you can either create a new global variable or edit an existing global variable.

For information on the relationship between global and local variables, see Scope and Variables.

To create a new global variable:

1. Right-click the Start Node and select **Create Variable** to open the Create Variable Dialog.
2. Enter a name for the variable.
3. Select the Variable Type (Builtin, Schema, or Message).
4. Choose a Variable.

   Note: You may need to drill down into the hierarchy to select a Schema or Message Type variable.

5. Click **OK**.

If it is not already open, expand the content area to the left by clicking the arrow to the left of the Start Node icon. The newly created variable appears in the Variables field along with any other global variables. To view the details of any variable, simply select it and its structure will appear in the Properties view.

To edit an existing global Variable:

1. Open the Edit Variable Dialog in one of the following ways:
   - Right-click the selected variable and select **Edit Variable** from the context menu.
   - Select a variable and click **Edit** in the variable Properties view.
2. Enter a name for the Variable.
3. Select the Variable Type: Builtin, Schema, or Message.
4. Choose a variable.

   Note: You may need to drill down into the hierarchy to select a Schema or Message Type variable.

5. Click **OK**.

If it is not already open, expand the content area to the left of the Start Node icon. The newly created variable appears in the Variables field along with any other global variables. To view the details of any variable, simply select it and its structure will appear in the Properties view.

Viewing External Services

The External Services listed in the Start Node are those invoked outside of the context of the Split-Join. They are specified in an Invoke Service but listed here for convenience.

To view External Services, expand the content area to the left of the Start Node by clicking the arrow to the left of the Start Node icon. When an External Service is selected, a dashed blue arrow appears pointing to the Invoke Service associated with the service, and the service's location appears in the Properties view.

Configuring a Receive

A Receive is generated automatically whenever you create a new Split-Join. The purpose of the Receive is to place incoming request data in a variable and make the contents available for later nodes to use. Configuring a Receive can include the following tasks:

- View the Operation
- Define the Receive Variable
- Add General Information

Related Topics

- Receive Properties

Viewing the Operation

The Operation is based upon the initial WSDL selection for the overall Split-Join. It is displayed in the Properties View for reference.

Defining the Receive Variable

You must define the Incoming Message Variable the Receive will initialize.

1. Select the Receive operation.
2. Create a new Message Variable (following steps).

Note: If there are no available Message Variables associated with the previously chosen Operation, you must create a new Message Variable.

To create a new Message Variable, select **Create Message Variable** from the Message Variable menu, enter a variable name in the Create Variable dialog, and click **OK**.

Note that Message Type Namespace and Message Type are displayed automatically on the Properties page once the variable is defined.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Creating an Assign

The Assign is used for data-manipulation including initializing and updating a variable. It is composed of a set of one or more operations that can be added from the Assign toolbar. Configuring an Assign can include the following tasks:

- Add and Configure Assign Operations
- Add General Information

Related Topics

- Assign Properties

Adding and Configuring Assign Operations

Assign operations include Assign, Copy, Delete, Insert, Java Callout, Log, and Replace. Every Assign is composed of one or more of these operations, which you can add to the Assign using the Design Palette view.

Note: The Assign operations in the Split-Join editor are essentially the same as the corresponding actions in the Workshop for WebLogic Message Flow editor. However, one important difference is that when you are using the XQuery\XSLT or XPath Editors

to edit expressions in the Split-Join context, only variables and namespaces internal to the Split-Join are available.

A brief explanation of each operation follows:

- **Assign** the result of an XQuery Expression to a Variable.

  For more information on configuring the Assign, see Assign Properties.

- **Copy** the information specified by an XPath 1.0 expression from a source document to a destination document. (This operation is unique to Split-Join and has no corresponding Action in the Workshop for WebLogic Message Flow editor, as described in Adding a Copy Operation).

- **Delete** a set of nodes specified by an XPath expression.

  Note: Unlike the Oracle Service Bus delete, only an XPath expression may be deleted in a Split-Join, not the entire variable.

  For more information on configuring the Delete operation, see Delete Properties.

- **Insert** the result of an XQuery Expression at an identified place relative to nodes selected by an XPath expression.

  For more information on configuring the Insert operation, see Insert Properties.

- **Java Callout** lets you invoke a Java method for processing such as validation, transformation, and logging.

  For more information on configuring the Java Callout operation, see Java Callout Properties.

- **Log** lets you log Split-Join data at a specified severity to the server log file.

  For more information on configuring the Log operation, see Log Properties.

- **Replace** a node or the contents of a node specified by an XPath expression

  For more information on configuring the Replace operation, see Replace Properties.

Adding an Operation to an Assign

Adding an operation to the Assign involves the following steps:

1. Drag the operation from the Design Palette into the Assign node in the Split-Join editor.
2. Configure the operation in the Properties view.

3. Save the Split-Join.

You can edit an operation by selecting it and modifying the properties in the Properties view.

Adding a Copy Operation

The Copy operation lets you copy the information specified by an XPath 1.0 expression from a source document to a destination document. It is an operation unique to the Split-Join editor. Adding a Copy operation to the Assign involves the following steps:

1. Add a Copy to the Assign from the Design Palette.
2. In the Properties view, select a **Copy From** type and a **Copy To** type.
3. If the type is an expression, enter the expression manually or click browse to launch the XQuery Expression Builder.
4. In the type is a variable, drill down to and select the desired element. The resulting query will be displayed in the Query field below.
5. If the Copy From type is a Literal, enter the literal in the text field.
6. If the Copy From type is an XML fragment, enter [or paste] the fragment in the text field.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Invoking a Service

The Invoke Service is used to invoke external, WSDL-based business services, WSDL-based proxy services, and Split-Joins. Configuring an Invoke Service can include the following tasks:

- Select an Operation
- Define Input and Output Variables

- Add General Information

Related Topics

- Configuring the Start Node
- About Scope
- Invoke Service

Selecting an Operation

An operation must be selected upon which to base the Invoke Service. You must select this operation before you can configure Input and Output variables. To select an operation:

1. Add an Invoke Service operation to the Split-Join from the Design Palette.
2. From the Operations tab in the Properties view, click **Browse** to launch the Service Browser.
3. In the Service Browser, drill down to the desired service and select a Binding, then an operation.
4. Click **OK**. The selected operation and its Service Location appear in the Properties view.

   Note: Clicking a Service Location in the Properties view will open the external service file.

Defining Input and Output Variables

An Invoke Service requires both an Input Variable and an Output Variable, unless it is a one-way invocation. The procedure to configure these variables is essentially the same. Either type of variable can be global (that is, available within the entire Split-Join) or local (that is, available within a particular context Scope.) To define either an Input or Output variable:

1. In the Input Variable and Output Variable tabs in the Properties view, define the Message Variable for each. This can be done in two ways:

   a. Select a pre-existing variable from the Message Variable menu.

   b. Create a new Message Variable (following steps).

   Note: If there are no available Message Variables associated with the previously chosen operation, you must create a new Message Variable.

To create a new Message Variable:

1. Select **Create Message Variable** from the Message Variable menu. The Create Message Variable Dialog appears.

2. Provide a name for the variable.
3. Make the variable either global or local. Global variables are accessible within the entire Split-Join, whereas local variables are restricted to the current Scope.

Message Type Namespace and Message Type are displayed automatically on the Properties view once a variable is defined.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Creating a Parallel

The Parallel creates a fixed number of configured parallel branches. Each branch has its own Scope which in turn can contain any number of nodes. Configuring a Parallel can include the following tasks:

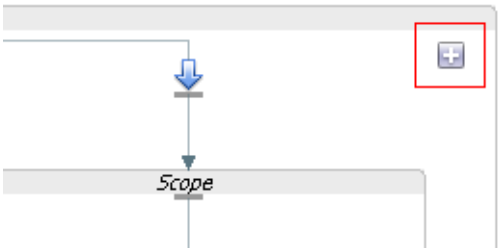
- Add Nodes
- Add General Information

Related Topics

- About Scope
-
- Parallel Properties

Adding Nodes

The Parallel is essentially a placeholder for a fixed number of processing branches, each with its own scope. Two branches are automatically generated by default. An individual scope may contain unique processing logic according to your construction; simply drag the appropriate nodes into the Scope. You may add additional branches with the Add Scope button.



Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Creating a For Each

The For Each is used to create conditional logic for iterating through a variable number of requests. It is primarily used to create dynamic Split-Joins. Configuring a For Each can include the following tasks:

- Define the For Each Logic
- Add General Information

Related Topics

- Invoking a Service
- For Each Properties

Defining the For Each Logic

To define the For Each logic:

1. Add a For Each node to the Split-Join, and select it.

2. In the Properties view, select the **Counter Variables** tab, and set the Parallel property to yes or no. If you choose yes, individual branches will be processed in parallel. If you select no, they are processed sequentially.
3. Define the Counter Variable Name by clicking the **Counter Name** link.
4. Enter the **Start Counter Value**. If necessary, use the browse button to create a new value in the XPath Expression Builder.

   Note: The lowest possible starting counter value is "1."
5. Enter the **Final Counter Value**. If necessary, use the browse button to create a new value in the XPath Expression Builder.

   Note: The lowest possible starting counter value is "1."

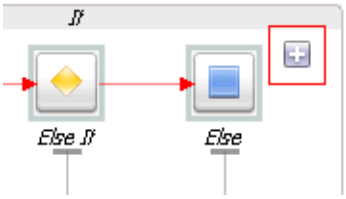
Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Creating an If Activity

The If Activity is used to provide conditional logic within a Split-Join. It is composed of a number of nodes that determine the behavior for the overall If activity. Each node must be individually configured. When you create an If activity, an If and an Else are automatically generated within it. You can add an unlimited number of Else If nodes with the Add Else If button.



Configuring an If Activity can include the following tasks:

- Configure the If
- Add and Configure Else If
- Configure the Else

Related Topics

- About Scope
- Designing a Dynamic Split-Join
- If Properties
- Repeat Until Properties
- While Properties

Configuring the If

The If provides a unit of conditional logic within the overall If activity. It is automatically generated when you create an If activity. Configuring an If can include the following tasks:

- Write the logic of the condition
- Add resulting nodes
- Add General Information

Related Topics

- If and Else If Properties
- Repeat Until Properties
- While Properties

Writing the logic of the condition

The If Activity executes conditional logic defined by an XPath 1.0 expression. Enter this condition in the Condition text field of the Condition tab, or click the browse button to launch and write the expression in the expression builder.

Adding resulting nodes

If the condition in the If logic is met, a subsequent node or string of nodes will result. Add and configure any resulting nodes by dragging them in sequential order to a drop point beneath the If icon.

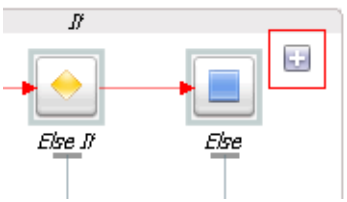
Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Adding and Configuring Else If

The Else If is used to provide additional logic within the context of an overall If. You can add an Else If every time you press the “Add Else If” button.



Configuring an Else If can include the following tasks:

- Write the Logic of the Condition
- Add Resulting Nodes
- Add General Information

Related Topics

- If and Else If Properties
- Repeat Until Properties

- While Properties

Writing the Logic of the Condition

The Else If uses conditional logic defined by an XPath 1.0 expression. Enter this condition in the Condition text field of the Condition tab or click the browse button to launch and write the expression in the expression builder.

Adding Resulting Nodes

If the condition in the Else If logic is met, a subsequent node or string of nodes will result. Add and configure any resulting nodes by dragging them in sequential order to a drop point beneath the Else If icon.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Configuring the Else

The Else provides a final case of logic within the context of an overall If. It is automatically generated when an If is created. Configuring an Else can include the following tasks:

- Add Resulting Nodes
- Add General Information

Related Topics

- If and Else If Properties
- Repeat Until Properties
- While Properties

Adding Resulting Nodes

As the final case in the If's logic, the Else requires no conditions to be met in order to execute. It will automatically execute resulting activities when invoked. Add and configure any resulting nodes by dragging them in sequential order to a drop point beneath the Else icon.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.
2. In the Documentation field enter any notes that you think are important.

Creating an Error Handler

The Error Handler receives and handles errors. If it is attached to a Start Node, it is a "global" Error Handler and serves as a catch-all for the output of all local Raise Error nodes. If it is attached to a Scope, it only handles errors raised locally. To create an Error Handler:

1. Select the start node or Scope node to which the Error Handler will be added.
2. Right-click the selected icon and select **Add Catch** or **Add CatchAll**.
3. If you are invoking a SOAP Fault, in the Catch All tab of the Properties View, select **SOAP Fault Variable Name** to define a SOAP Fault variable associated with the Error Handler.

The basic Error Handler is now configured, but you may need to add additional Assign, If, and/or Reply nodes to it depending on whether you wish to execute logic upon the received faults before sending a response.

Related Topics

- Creating an Assign
- Creating an If Activity
- Creating a Parallel
- About Scope

- Error Handler Properties

Creating a Raise Error

The Raise Error generates an error that causes the Split-Join to stop normal processing. If the error is not handled using an Error Handler, the Split-Join will terminate and a Fault will be sent to the Oracle Service Bus message flow. Configuring a Raise Error can optionally include documenting the nature of the error in the General Information tab.

You can also add a Re-Raise Error operation to an Error Handler. Configuration involves modifying Label and adding Documentation.

Related Topics

- Raise Error Properties

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with developer notes, or Documentation. General information is optional.

1. To add a Label to an node, open the Properties view and enter a unique, identifying string in the **Label** field. The Label that you enter appears underneath the node Icon in the Canvas area.
2. In the Documentation field enter any notes that you think are important.

Configuring a Reply

A global Reply is generated automatically whenever you create a new Split-Join. The purpose of the global Reply is to send a response back to the invoking Oracle Service Bus message flow. However, you may also create a Reply elsewhere in the Split-Join, including within Error Handlers. Configuring the Reply can include the following tasks:

- View the Operation
- Define the Reply Variable
- Add General Information

Related Topics

- Reply Properties

Viewing the Operation

The operation is based upon the initial WSDL selection for the overall Split-Join. It is displayed in the Properties view for reference.

Defining the Reply Variable

The Reply can either send a Response or a Fault back to the client depending on how you configure the variable. The Fault options available vary depending upon whether the Reply is global or local.

- A global Reply (that is, a Reply in a Split-Join outside of an Error Handler) can never have a SOAP Fault but can have a WSDL Fault. This is why the SOAP Fault option is always disabled.
- A local Reply (that is, a Reply attached to an Error Handler) can have either a WSDL Fault or a SOAP Fault. WSDL Faults will be available only if they were defined in the WSDL upon which the Split-Join is based. The SOAP Fault option will always be available provided one has been previously defined in the Error Handler.

Note: Switching back and forth between the Response and Fault buttons will clear either configuration. For instance, if you have previously selected “Propagate SOAP Fault” for Fault configuration and you then switch to the “Response” configuration, “Propagate SOAP Fault” will be deselected.

Given the available options as outlined above, select either a Response or a Fault for your Reply Variable.

If you select Response, you must define the Message Variable the Response will be assigned to. This can be done in two ways:

1. Select the Reply, and in the Properties view, select the Variable tab.

1. Select a pre-existing variable from the Message Variable menu.

2. Create a new Message Variable (following steps).

   Note: If there are no available Message Variables associated with the previously chosen operation, you must create a new Message Variable.

To create a new Message Variable, select **Create Message Variable** from the Message Variable menu. The Create Message Variable Dialog appears:

1. Provide a name for the variable.

2. Click **OK**.

Note that Message Type Namespace and Message Type are displayed automatically in the Properties view once the variable is defined.

If you select **Fault**, you must specify either a **WSDL Fault** or propagate a **SOAP Fault**.

Note: In some circumstances, no Faults or only a SOAP Fault will be available. See previous notes.

If you select a WSDL Fault, you must specify the Fault by name and define the Message Variable that it will be assigned to.

1. Select **WSDL Fault Name** and choose a name from those available.

   Note: There may only be one name available in which case no choice is necessary.

2. Define a Message Variable. This can be done in two ways:

   a. Select a pre-existing variable from the Message Variable menu.

   b. Create a new Message Variable (following steps).

   Note: If there are no available Message Variables associated with the previously chosen operation, you *must* create a new Message Variable.

To create a new Message Variable, select **Create Message Variable** from the Message Variable menu. The Create Message Variable Dialog opens:

1. Provide a name for the variable.
2. Click **OK**.

Message Type Namespace and Message Type are displayed automatically on the properties page once a variable is defined.

If you select Propagate SOAP Fault, the SOAP Fault specified in the parent Error Handler will automatically be propagated in the Reply. There is nothing else to configure.

Adding General Information

General information is useful for making a node more legible. It includes the ability to add a unique identifier, or Label, to the node and to supplement it with notes, or Documentation. General information is optional.

1. To add a Label to a node, select the General tab in the Properties view and enter a unique, identifying string in the Label field. The Label that you enter appears underneath the node in the Split-Join editor.

2. In the Documentation field enter any notes that you think are important.

About Scope

A Scope is a container that groups various elements together. The container creates a context that influences the behavior of its enclosed elements. Local Variables and the Error Handler defined within the Scope are restricted to this context. However, some nodes within the scope may operate both locally (that is, within the Scope) and globally (that is, outside of the Scope.) For instance, an Invoke Service within a certain Scope might call upon an service external to the Scope's context.

Scope and Variables

Although variables are visible in the scope in which they are defined and in all scopes nested within that scope, a variable declared in an outer scope is hidden when you declare a variable with an identical name in an inner scope. For example, if you define variable myVar in an outer scope (So) and then define variable myVar again in an inner scope (Si) which is contained by scope So, then you can only access the myVar you defined in the inner scope Si. This myVar overrides the myVar you defined in scope So.

Related Topics

- Scope Properties

Exporting and Testing a Split-Join

You can export and test a Split-Join on an Oracle Service Bus server provided that it is associated with a transport typed business service. Exporting and testing a Split-Join can include the following tasks:

- Creating a Transport Typed Business Service
- Exporting the Split-Join Files
- Testing the Split-Join in the Test Console

Creating a Transport Typed Business Service

A Split-Join is used by a particular transport typed business service. If you do not have an appropriate business service, you must create one before you can export or test your Split-Join. There are two ways to create a business service:

1. Create the business service manually in Workshop for WebLogic or the Oracle Service Bus console.
2. Generate the business service automatically from the Split-Join (.flow) menu:
   a. Right click on the Split-Join (.flow) file in the Project Explorer to open the Split-Join menu.
   b. Select **Oracle Service Bus**.
   c. Select **Generate Business Service**.
   d. Name and save the new service in a project.

After you create the business service, you can export the Split-Join provided that it has no errors.

Note: It is a helpful practice to place the associated business service in the same Oracle Service Bus project as the Split-Join. It can also be useful to give the business service the same name as the Split-Join so that they are easily correlated.

Exporting the Split-Join Files

Split-Joins without errors can be exported to an Oracle Service Bus server.

Note: Errors appear in the Problems view of the Split-Join editor. If you try to export a Split-Join with errors, the export fails.

There are three ways to export a Split-Join:

1. Export from the Business Service Menu
2. Auto-export
3. Manual export

Exporting from the Business Service Menu

It is possible to export a Split-Join directly from the Business Service menu. However, because exporting by this method automatically launches the Oracle Service Bus Test Console, it is useful if you want to both export and test. Exporting from the Business Service menu involves the following steps:

1. In the Project Explorer, right click on the Business Service (.biz file) to be exported/tested.
2. Select **Run as**.
3. Select **Run on server**. The Run on Server Dialog opens.

4. Select an existing server or define a new one and go to the next step.
5. In the Add and Remove Projects window, the Oracle Service Bus project containing the business service and any other dependent files have been pre-selected for configuration/export. They can not be removed because the business service can not be tested without its dependent files. The entire Split-Join will therefore be exported.
6. Select **Finish**, and the Oracle Service Bus Test Console will launch. You can now test the business service.

Auto-export

A Split-Join can be auto-exported to an Oracle Service Bus server. If you use this method, you must manually launch the Oracle Service Bus console in order to test the exported files. Auto-exporting involves the following steps:

1. Select **File > Export**.
2. Select **Oracle Service Bus**.
3. Select **Resources to Oracle Service Bus Server**. This will export the resources to the Oracle Service Bus server, but it will not launch the Oracle Service Bus Test Console. You must launch the Test Console manually within the Oracle Service Bus console application.

Manual export

A Split-Join can be manually exported to an Oracle Service Bus server. If you use this method, you must manually launch the Oracle Service Bus console to test the exported files. Manually exporting involves the following steps:

1. Select **File > Export**.
2. Select **Oracle Service Bus**.
3. Select **Resources as Configuration JAR** and go to the next step.
4. In the Oracle Service Bus Configuration JAR Export window, configure the following options:
    a. Select the Oracle Service Bus Configuration file containing the files to be exported.
    b. Set the Export Level to **Project** or **Resource** depending upon whether you wish to export entire projects or individual files. The selection available in the tree below will change based upon the Export Level.

c. Select the projects and/or resources to be exported in the configuration JAR.

d. Select **Include Dependencies** if you want to export any file dependencies associated with the selected files.

e. Browse to a destination for the exported JAR file.

f. Click **Finish** to export the JAR file.

5. Import the JAR file via the Oracle Service Bus console.

Note: A quick way to access the Oracle Service Bus console from Workshop for WebLogic is to right-click the server and select **Launch Service Bus Console.**

Testing the Split-Join in the Test Console

You can test a Split-Join by executing the business service that uses it in the Oracle Service Bus Test Console. This can either be done within the Split-Join editor or by exporting the Split-Join to an Oracle Service Bus server. To test the Split-Join within the IDE, you need to export the files using the menu for the business service that uses the Split-Join.

Exporting from the Business Service Menu

You can export and test a Split-Join directly from the Business Service menu. If you use this method, the export happens in the background while the Oracle Service Bus Test Console launches. Exporting from the Business Service menu involves the following steps:

1. In the Project Explorer, right click on the Business Service (.biz file) to be exported/tested.
2. Select **Run as**.
3. Select **Run on server**. The Run on Server Dialog appears.
4. Select an existing server or define a new one and go to the next step.
5. In the Add and Remove Projects window, the Oracle Service Bus project containing the business service and any other dependent files have been pre-selected for configuration/export. The dependent files cannot be removed because the business service cannot be tested without its dependent files.

Click **Finish**, and the Oracle Service Bus Test Console will launch. You can now test the business service.

Note: Although only the Oracle Service Bus Test Console is displayed at this point, the entire Split-Join has been exported to the Oracle Service Bus server.

Using the Oracle Service Bus Debugger

Oracle Service Bus extends the Eclipse debugging framework to provide debugging functionality for proxy service message flows and Split-Joins.

The Oracle Service Bus Debugger can handle Java callouts and supports multi-threaded debugging on Split-Joins that use parallel processing. You can also perform debugging on remote servers.

You can use the Oracle Service Bus Debugger in two different ways:

- Using Standard Debugging – Provides automated service debugging features for debugging on a local machine.
- Using the Oracle Service Bus Debugger Launch Configuration – Provides more manual control of the Oracle Service Bus Debugger environment.

Enabling Debugging

Oracle Service Bus debugging is enabled automatically on a server running in development mode. When you create a new domain, the following entries are included in the <domain>/bin/setDomainEnv script:

set ALSB_DEBUG_FLAG=true

set ALSB_DEBUG_PORT=7453

If you want to turn Oracle Service Bus debugging functionality off, set ALSB_DEBUG_FLAG=false.

If you start the server in production mode, Oracle Service Bus debugging is automatically disabled.

Using Standard Debugging

To debug a proxy service or a Split-Join, you must execute it while in debug mode (unless you are using the debugger launch configuration described in Using the Oracle Service Bus Debugger Launch Configuration.)

To debug a proxy service or Split-Join:

1. Set breakpoints in your message flow or Split-Join to automatically pause the test execution at those points. Right-click an action in the flow editor and choose **Toggle Breakpoint**.

2. Start the server in debug mode. On server startup, Workshop for WebLogic automatically switches to the Debug perspective, shown in Figure 2-1.
3. With the proxy or business service file open and active, select **Run > Debug As > Debug on Server** in Workshop for WebLogic.
   a. In the Debug on Server window, select the server on which your Oracle Service Bus configuration is deployed, or is to be deployed, and complete the steps in the wizard.

   The Debug As option automatically enables the Java debugger in order to handle Java callouts in services.
4. The Oracle Service Bus Test Console appears. Execute your test, looking at and interacting with the execution threads in the Debug view. As you move through the test, the debugger highlights the current stage of the test in the service's flow view, as shown in Figure 2-1.

You are not required to use the Test Console to execute a service test and use the debugger. You can also execute your service in other ways, such as posting a JMS message or dropping a file in the directory of a file proxy service.

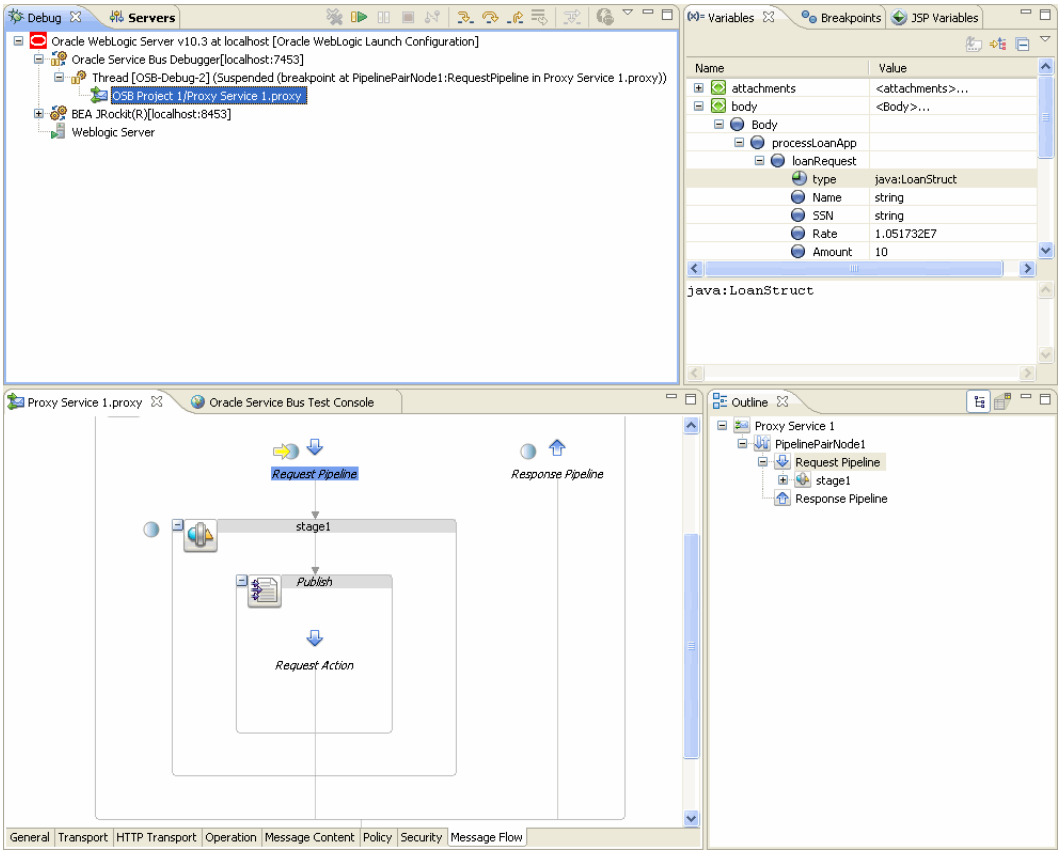
For debugging on remote servers, execute the service on the remote server and step through the test in Debug view. If the remote server is running in normal mode rather than debug mode, use the debugger launch configuration, as described in Using the Oracle Service Bus Debugger Launch Configuration.

See the Debug Views section for descriptions of different debug views for Oracle Service Bus Debugger.

Debug Views

This section describes the different views you can use while debugging a service, illustrated in Figure 2-1.

Figure 2-1 Debugging a proxy service



The Debug perspective provides the following key views for debugging a service:

- **Debug view** – The Debug view shows the Oracle Service Bus call stack, displaying the current execution thread. The Debug view also provides a toolbar that lets you resume the test from its current location or perform step actions (Step Into, Step Over, Step Return). As you step through a test, if you are testing a local service, the current operation is highlighted in the service's flow view. Errors appear in the Console view.
- **Console view** – The Console view, which displays server messages, shows any run-time errors that occur as your service runs.

- **Variables view** – The Variables view shows the variable names and values in your service at each stage of the test. Variables are read-only while debugging. Select the call stack at a specific breakpoint to see the current variables and values.
- **Breakpoints view** – The Breakpoints view lists the breakpoints you have manually inserted into your service's flow. The test stops at each breakpoint, letting you view the state of the service at that point. You can enable and disable breakpoints in the Breakpoints view. Disabling a breakpoint by deselecting it keeps the breakpoint in place, but the debugger ignores it.
- **Service editor, flow view** – As the test runs, if you are testing a local service, the test progress is graphically highlighted in the service's flow view. The flow view also shows the breakpoints you have set (by right-clicking actions in the service's flow view editor and choosing Toggle Breakpoint).
- **Oracle Service Bus Test Console** – The Test Console lets you execute local proxy and business services. For more detailed information on using the Test Console, see Using the Test Console in the *Oracle Service Bus User Guide*.

Step Actions and Breakpoints

The Eclipse debugging framework lets you debug incrementally by performing step actions to debug code. See "Execution Control Commands" in the Eclipse help system (*Java Development User Guide*).

If you use step actions to debug, the Oracle Service Bus Debugger still stops at each breakpoint you have set, ignoring the current step action being performed.

Using the Oracle Service Bus Debugger Launch Configuration

If you want more manual control of the Oracle Service Bus Debugger environment, use the Oracle Service Bus Debugger launch configuration. Launch configurations are an Eclipse feature. The Oracle Service Bus Debugger launch configuration removes the automation of running the debugger with the Debug As option (Debug perspective launching, server restarting in debug mode, Java debugger starting), letting you connect and disconnect the debugger as needed on a server running in either normal or debug mode.

If you want to use the Java debugger in conjunction with the Oracle Service Bus Debugger to handle Java callouts, the server must be running in debug mode.

To use the Oracle Service Bus Debugger launch configuration:

1. Set breakpoints in your proxy service message flow or Split-Join.
2. Start the server in normal or debug mode. However, if you want to use the Java debugger to handle Java callouts from your flow, start the server in debug mode.
3. In the Workshop for WebLogic menu, select **Run > Open Debug Dialog**.
4. Double-click **Oracle Service Bus Debugger**. A "New_configuration" sub-item appears. You need only perform this step once for each launch configuration you want to create.

   The right pane displays the default server and port. Make sure the port matches the value of the ALSB_DEBUG_PORT in your domain's setDomainEnv script.

   You can also rename the launch configuration in the right pane, as well as add the launch configuration to the Debug toolbar item as a shortcut.

   a. If your services use Java callouts, enable the Java debugger by double-clicking **Remote Java Application** and configuring the "New_configuration."
5. With your launch configuration selected, click **Debug** in the Debug dialog. The debugger is connected. If you are using the Java debugger as well, select it and click Debug.
6. Open the Debug perspective or the debug views you want, such as Debug, Breakpoints, and Variables.
7. Run the service.
   - To execute the service in the Test Console, use **Run As > Run on Server**.
   - You can also execute your service in other ways, such as through a custom test client.
8. Step through the test.

To disconnect the debugger, click the Disconnect icon in the Debug view. Reconnect the debugger in the Debug dialog by selecting the launch configuration and clicking Debug, or by creating a shortcut in the Debug toolbar item as described in the previous steps.

Remote Debugging

If you are debugging remote services, you can select a remote server in the server configuration window (for Debug As) or set the remote server and port in the launch configuration. After you connect the debugger to the remote server, excute the services on the remote server and step through the test in your local Debug perspective.

Server Sharing

If multiple users share a single server instance for debugging, only one user at a time can have the debugger connected. Other users trying to connect a debugger will get a connection refused error.

CHAPTER 3

User Interface Reference

This section describes all the views, dialogs, wizards, and other user interface objects in the Oracle Service Bus plug-ins.

- Alerts
- Business Service Configuration
- Proxy Service Configuration
- Oracle Service Bus Configurations and Projects
- Custom Resources
- Export wizard
- Import wizard
- JNDI Providers
- Proxy Servers
- Message Flow Design Palette
- Message Flow Editor
- Modify JAR Dependencies dialog
- SMTP Servers
- UDDI Registry Configuration page

- Outline view - Oracle Service Bus
- Resource Management
- New Service Key Provider Resource wizard
- New WS-Policy wizard
- Service Accounts
- Expression Editors
- New XSL Transformation wizard
- Split-Join User Interface Reference

Alerts

The following pages are provided for managing alerts:

- Alert Destination editor
- Edit E-mail Recipient page
- Edit JMS Destination page

Alert Destination editor

An alert destination is a destination address for alert notifications in Oracle Service Bus. Use this page to configure an alert destination resource. For more information, see Alert Destinations in *Using the Oracle Service Bus Console*.

Option	Description
Description	Enter a description for this alert destination.
SNMP Trap	If you specify **SNMP Trap**, alerts are sent as SNMP traps, and can be processed by any third-party enterprise monitoring systems (ESM).
Reporting	If you specify **Reporting**, alerts are sent to the Oracle Service Bus Reporting module and can be captured using a custom reporting provider that can developed using the reporting APIs provider by Oracle Service Bus. This allows third-parties to receive and process alerts in custom Java code.

Continue in the Edit Alert Destination page. Click **Add** to add e-mail and JMS recipients to an alert destination. See:

- "Edit E-mail Recipient page" on page 3-3
- "Edit JMS Destination page" on page 3-4

For more information, see Adding E-Mail and JMS Recipients in *Using the Oracle Service Bus Console*.

Edit E-mail Recipient page

Use this page to configure the destination target for an alert sent via e-mail (that is, using the e-mail transport). For more information, see Adding E-Mail Recipients in *Using the Oracle Service Bus Console*.

Option	Description
Mail Recipients	Enter an e-mail recipient in the format: `mailto:username@hostname` You can specify multiple e-mail recipients by entering the user names and hostnames in a comma-separated list. For example, `mailto:username@hostname [,username_1@hostname_1]...[,username_n@hostname_n ]` Only the first mail recipient needs to be prefixed with the text "`mailto:`". However doing so is optional; the code will add it if it is missing.
SMTP Server	Select the name of the SMTP server for the outgoing e-mail. This field is not required if **Mail Session** is selected in the next field. These fields are mutually exclusive; it is an error to configure both.
Mail Session	Select an available mail session. This field is not required if an **SMTP Server** is selected in the previous field. These fields are mutually exclusive; it is an error to configure both.
From Name	Provide a sender's name for the alert notification. This field is optional.
From Address	Provide a valid e-mail address. This field is required if a value for the **From Name** field is specified.

Option	Description
Reply To Name	Provide a name to which a reply may be addressed. This field is optional.
Reply To Address	Provide an e-mail address to which a reply may be sent. This field is required if a value for the **Reply To Name** field is specified.
Connection Timeout	Enter the number of seconds a connection must wait for a response from the server before timing out. The default value is `0`.
Request Encoding	Enter a character set encoding value. The default encoding value is `iso-8859-1`.

Edit JMS Destination page

Use this page to configure the destination target for an alert sent via JMS (that is, using the JMS transport). For more information, see Adding JMS Recipients in *Using the Oracle Service Bus Console*.

Option	Description
Destination URI	Enter a JMS destination URI in the format: `jms://host:port/factoryJndiName/destJndiName`
Destination Type	Select **Queue** or **Topic**.
Message Type	Select **Bytes** or **Text**.
Request Encoding	Enter a character set encoding value. The default encoding value is `iso-8859-1`.

Business Service Configuration

You configure new business services while creating them in the New Business Service wizard. You can view and modify those settings in the Business Service editor: With a few exceptions, configuration options are identical in the wizard and the editor and are therefore documented in one place. The wizard and editor business service configuration pages are:

- "Business Service General Configuration page" on page 3-5

Business Service General Configuration page

Use the Business Service General Configuration page to specify general configuration settings for a business service. This page appears both in the New Business Service wizard and in the Business Service editor. Options vary, depending on whether you are using the wizard or the editor, as described below.

New Business Service Wizard Options

The following table describes the options in the wizard:

Option	Description
Description	Enter a description for this service.
Service Type - Create a New Service (wizard only) A service type defines the types and packaging of the messages exchanged by the service. Select the type of business service to create:	
WSDL Web Service	Select this option to create a business service based on a WSDL. Then, enter the WSDL name, qualified by its path (for example, `myProject/myFolder/myWSDL`). Alternatively, click **Browse** to select a WSDL resource. **(port or binding)** - Enter the name of a port (defined in the WSDL) to describe an actual transport address, or enter the name of a binding (defined in the WSDL) to map to a transport address. If you use **Browse** to select a WSDL, as described above, the **Select a WSDL Definition** dialog lists any ports and bindings defined in the WSDL. When you choose a port or a binding on that page, the **(port or binding)** field is filled with the selected name.
Transport Typed Service	Select this option to create a service that uses EJB transport.

Option	Description
Messaging Service	Select this option to create a service that exchanges messages of different content-types. These exchanges can be either request/response or one-way. They can also have a response with no request when used with the HTTP 'GET' option for the HTTP transport. Unlike Web Services, the content-type of the request and response need not be the same.
Any SOAP Service	Select this option to create a SOAP service that does not have an explicitly defined, concrete interface. Select **SOAP 1.1** or **SOAP 1.2** from the drop-down list to specify the SOAP version to be used.
Any XML Service	Select this option to create an XML service that does not have an explicitly defined, concrete interface. HTTP GET is only supported for messaging services and this service type.
Service Type - Create From Existing Service Choose one of these options to create a service based on another service.	
Business Service	Select this option to clone an existing business service. Enter the path (`project/folder`) and the name of the business service; or click **Browse** to select a service. Since Oracle Service Bus does not accept the same URI for multiple services, you must change the URI for the cloned service.
Proxy Service	Select this option to create a business service based on an existing proxy service. Enter the path (`project/folder`) and the name of the proxy service; or click **Browse** to select a service. Since Oracle Service Bus does not accept the same URI for multiple services, you must change the URI for the cloned service.

Business Service Editor Options

The following table describes the options in the editor:

Option	Description
Description	Enter or modify a description for this service.
Service Type (editor only)	This option shows the service type of the business service. You can change only some of the properties of some of the service types: • **WSDL Web Service** - You can enter (or click **Browse** to select) a different port or binding from the same WSDL. You can also specify a different WSDL, but by doing so, you are effectively creating a new service and you will have to configure it as if it were a new service. • **Transport Typed Service** - This option cannot be modified. • **Messaging Service** - This option cannot be modified. • **Any SOAP Service** - You can change the SOAP version (SOAP 1.1 or SOAP 1.2) • **Any XML Service** - This option cannot be modified.

Business Service Message Type Configuration page

Use the Business Service Message Type Configuration page to configure message types for a business service whose type is Messaging Service. This page appears both in the New Business Service wizard and in the Business Service editor:

The binding definition for messaging services consists of configuring the content-types of the messages that are exchanged. The content-type for the response does not need to be the same as for the request; therefore, the response is configured separately (for example, the service could accept an MFL message and return an XML acknowledgment receipt).

Note: E-mail, File, FTP, or SFTP transport business services whose type is Messaging Service support one-way messaging *only*; the **Response Message Type** should be **None**. If you select an option other than **None**, the file, ftp, or sftp protocol will not be available on the Transport Configuration page.

Option	Description
Request Message Type	Select a message type for the request message: • **None** - Select this option if there is no request message (HTTP GET example) • **Binary** - Select this option if the content-type of the message is unknown or not important. • **Text** - Select this option if the message can be restricted to text. • **MFL** - Select this option if the message is a binary document conforming to an MFL definition. Enter the MFL file name (qualified by its path), or click **Browse** to select a file. You can configure only one MFL file. **Note:** To support multiple MFL files, define the content as binary or text and use the MFL action in the message flow to convert to XML. • **XML** - Select this option if the message is an XML document. Enter the XML file name (qualified by its path), or click **Browse** to select a file. Optionally provide some type information by declaring (in the **element or type** field) the XML schema type of the XML document exchanged.
Response Message Type	Select a message type for the response message: • **None** - Select this option if there is no response message. • **Binary** - Select this option if the content-type of the message is unknown or not important. • **Text** - Select this option if the message can be restricted to text. • **MFL** - Select this option if the message is a binary document conforming to an MFL definition. Enter the MFL file name (qualified by its path), or click **Browse** to select a file. You can configure only one MFL file. **Note:** To support multiple MFL files, define the content as binary or text and use the MFL action in the message flow to convert to XML. • **XML** - Select this option if the message is an XML document. Enter the XML file name (qualified by its path), or click **Browse** to select a file. Optionally provide some type information by declaring (in the **element or type** field) the XML schema type of the XML document exchanged.

Business Service - Service Policy Configuration page

Use the Business Service - Service Policy Configuration page to configure service policy settings for a proxy service. This page appears both in the New Business Service wizard and in the Business Service editor.

Option	Description
WSDL-Based Policy	Select this option if the service policy is associated with the WSDL upon which the service is based.
Custom Policy Bindings	Select this option to add service-level policies, operation-level policies (in which case the policy applies to both the request and response messages), request policies, and response policies directly. For more information about configuring service policies, see Configuring Message Level Security for Web Services in the *Oracle Service Bus Security Guide*.

Business Service SOAP Binding Configuration page

Use Business Service SOAP Binding Configuration page to configure the SOAP Binding for a business service based on a WSDL. This page appears both in the New Business Service wizard and in the Business Service editor:

Select or deselect **Enforce WS-I Compliance** to specify whether or not the service is to conform to the Basic Profile defined by the Web Services Interoperability Organization. This option is available for or SOAP 1.1 services only

When a service is marked WS-I compliant, checks are performed against the messages sent to and from that service.

Business Service Transport Configuration page

Use the Business Service Transport Configuration page to select, review, or change the service's transport protocol and to set, review, or change general transport configuration settings. This page appears both in the New Business Service wizard and in the Business Service editor:

Outbound transport-level security applies to the connections between Oracle Service Bus proxy services and business services. For more information about transport-level security, see Configuring Transport-Level Security in the *Oracle Service Bus Security Guide*.

Option	Description
Protocol	Select a transport protocol from the list. The protocols available differ, depending on the service type: • **WSDL Web Service**: bpel-10g, dsp, http, jca, jms, jpd, sb, ws • **Transport-Typed Service**: ejb, flow • **Messaging Service:** email, file, ftp, http, jms, mq (if available), sftp, tuxedo • **Any SOAP Service:** dsp, http, jms, jpd, sb • **Any XML Service:** dsp, email, file, ftp, http, jms, jpd, mq (if available), sb, sftp, tuxedo
Load Balancing Algorithm	Select one of these load-balancing algorithms: • **Round-robin** - This algorithm dynamically orders the URLs that you enter in the **Endpoint URI** field for this business service. If the first one fails, it tries the next one, and so on until the retry count is exhausted. For every new message, there is a new order of URLs. • **Random** - This algorithm randomly orders the list of URLs that you enter in the **Endpoint URI** field for this business service. If the first one fails, it tries the next one, and so on until the retry count is exhausted. • **Random-weighted** - This algorithm randomly orders the list of URLs that you enter in the **Endpoint URI** field for this business service, but some are retried more than others based on the value you enter in the **Weight** field. • **None** - This algorithm orders the list of URLs that you enter in the **Endpoint URI** field for this business service from top to bottom.

Option	Description
Endpoint URI	Enter an endpoint URL in the format based on the transport protocol you selected in the **Protocol** field, above: The formats are: • **bpel-10g** – `protocol://host[:port][/protocol-path]/domain/process[/version[/partnerlink/role]]` Optional URI elements are shown in square brackets. For more information, see the Oracle BPEL Process Manager User Guide. • **dsp** - `t3://dsp-ip-address:port/dsp-app-name` • **ejb** - `ejb:provider:jndiname` In the URI, `provider` is the name of the JNDI provider resource, and `JNDIname` is the JNDI name in the JNDI server for the EJB. If the JNDI provider is located on the same server, the JNDI provider need not be specified. The URI then would be `ejb::jndiname` • **email** - `mailto:foo@bar.com` • **file** - `file:///root-dir/dir1` • **ftp** - `ftp://hostname:port/directory` • **http** - `http://host:port/someService` The HTTP transport supports both `HTTP` and `HTTPS` endpoints. • **jca** - `jca://<resource_adapter_jndi>` • **jms** - `jms://host:port[,host:port]*/factoryJndiName/destJndiName` To target a JMS destination to multiple servers, use the following URI format: `jms://host1:port,host2:port/QueueConnectionFactory/DestName` **In a cluster**: The host names in the JMS URI must exactly match the host names of the cluster servers as they are configured in WebLogic Server. • **jpd** - `jpd:[<provider>]:<jpd_uri>` `provider` (optional) is the name of the JNDI provider which corresponds to the WLI JNDI provider resource. When omitted, the JNDI provider on the local server is used. `<jpd uri>` is the relative URL of the JPD on the WLI server. For example, if `processes.Process.jpd` is in the SampleApp Web project, then the relative URL of the JPD is `/SampleApp/processes/Process.jpd`.

Option	Description
Endpoint URI (continued)	• **mq** - `mq://local-queue-name?conn=mq-connection-resource-ref` `local-queue-name` is the name of the MQ queue from which the business service reads messages. `mq-connection-resource-ref` is the path (project/folder) and name of the MQ connection resource; for example, `default/my_MQconnection`. **Note:** The Endpoint URI cannot contain spaces, so do not create MQ Connection resources or projects/folders with spaces in the names. To make the MQ transport available in Oracle Service Bus, see the MQ Transport User Guide. • **sb** - `sb://<jndi_provider_name/>service_name` `jndi_provider_name` (optional) is the name of the Oracle Service Bus JNDI provider resource. When omitted, the default context is used. `service_name` is a target service and corresponds to the remote proxy service URI. • **sftp** - `sftp://hostname:port/directory` • **tuxedo** - `tuxedo:resourcename/remotename` `tuxedo-queue:sendQSpace/sendQName[/[rcvQspace:]rvcQname][/failureQname]` In the URI, `resourcename` corresponds to a WTC Import name and `remotename` corresponds to the service name exported by the remote Tuxedo domain. The URI `resourcename` is required, and the `remotename` is optional. If more than one URI is specified, you must have unique resource names for the endpoints. If no remote name is specified, its value is the value of the resource name. If no remote name is entered or if remote and resource name are the same, only one URI is allowed. In this case resource name and remote name have the same value. This allows already defined WTC Imports to make use of WTC load-balancing and failover. For more information, see *Oracle Service Bus Interoperability Solution for Tuxedo*. • **ws** - `http://host:port/someService` **Note:** Oracle Service Bus no longer supports duplicate endpoint URIs within the same business service. Click **Add** to add one or more additional URIs. At run time, the URLs are selected based on the load balancing algorithm you selected in the **Load Balancing Algorithm** field. If you selected **Random-weighted** in the **Load Balancing Algorithm** field, you can also enter a weight in the **Endpoint URI** field. The default is `1`. If you have multiple endpoint defined, and you selected **None** in the **Load Balancing Algorithm** field, the order of endpoints is significant. You can reorder the endpoints using the **Up** and **Down** buttons. Oracle Service Bus does not support duplicate endpoint URIs within the same business service.

Option	Description
Retry count	In case of delivery failure when sending outbound requests, specify the number of times to retry individual URL endpoints; in other words, the number of failover attempts. For example, a business service has one configured URI (U1) and the number of retries is set to 3. If U1 fails on the first attempt, the system retries the U1 endpoint 3 more times. If a business service has 2 configured URIs (U1 and U2) and a retry count of 3, if the first attempt (for example, to U1) fails, the system tries (fails over to) the next URI (U2). If that also fails, the system makes two more attempts, once to U1 and once to U2.
Retry Iteration Interval	Specify the number of seconds the system pauses before iterating over all the endpoint URIs in the list again. For example, a business service has two configured URIs (U1 and U2) and a retry count of 2 with a retry iteration interval of 5 seconds. If the first attempt (to U1) fails, the system tries U2 right away. But if U2 also fails, the system waits for 5 seconds and retries U1 once more.
Retry Application Errors	Select **Yes** or **No**. In case of delivery failure when sending outbound requests, specify whether or not to retry endpoint URIs based on application errors (for example, a SOAP fault).

Business Service Message Content Handling Configuration page

Use the Business Service Message Content Handling Configuration page to specify how Oracle Service Bus is to encode outbound messages sent by business services and whether Oracle Service Bus should stream attachments in outbound response messages instead of buffering the attachment contents in memory.

Using this page, you can enable the business service to encode outbound messages in MTOM/XOP format. SOAP Message Transmission Optimization Mechanism (MTOM) is a method of sending binary data to and from Web services. MTOM uses XML-binary Optimized Packaging (XOP) to transfer the binary data.

Using this page, you can also enable the business service to store attachments in outbound response messages to a disk file and then process the data in a streaming fashion without buffering the attachment contents in memory. This enables the business service to process large attachments robustly and efficiently.

Option	Description
XOP/MTOM Support	Oracle Service Bus supports XOP/MTOM using the following transports: • HTTP/S • Local • SB Select the **Enabled** check box to enable the business service to encode outbound messages in MTOM/XOP format. Note that this option is disabled for imported business services that are based on previous release configurations. If XOP/MTOM Support is enabled, select how to handle binary data in the $header and $body message context variables from among the following options: • **Include Binary Data by Reference**: (Default) In an outbound response message, replace xop:Include elements with ctx:binary-content elements when setting up the $body message context variable. • **Include Binary Data by Value**: In an outbound response message, replace xop:Include elements with base64-encoded text versions of corresponding binary data when setting up the $body message context variable. Note that if XOP/MTOM Support is enabled for a business service, it is not required that every outbound message be in the MTOM format. Instead, this setting specifies that the business service is capable of handling an MTOM payload. Since Oracle Service Bus does not support a combination of MTOM and SwA, the system issues a runtime error when Oracle Service Bus attempts to dispatch an outbound request to a business service and the business service is both MTOM/XOP-enabled and the $attachments message context variable is not null.

Option	Description
Attachments	Oracle Service Bus supports streaming MIME attachments using the HTTP/S transport. Select the **Page Attachments to Disk** check box to enable the business service to stream attachments in outbound response messages. Note that if you enable XOP/MTOM Support, the Attachments option is only available if you choose the **Include Binary Data by Reference** option under XOP/MTOM Support. Note also that payloads that contain attachments must conform to RFC 822. Specifically, lines containing Internet headers need to be terminated with CRLF (carriage return line feed).

Proxy Service Configuration

You configure new proxy services while creating them in the New Proxy Service wizard. You can view and modify those settings in the Proxy Service editor: With a few exceptions, configuration options are identical in the wizard and the editor and are therefore documented in one place. The wizard and editor business service configuration pages are:

- “Proxy Service General Configuration page” on page 3-15
- “Proxy Service Message Level Security Configuration page” on page 3-18
- “Proxy Service Message Type Configuration page” on page 3-21
- “Proxy Service Operation Selection Configuration page” on page 3-24
- “Proxy Service Message Content Handling Configuration page” on page 3-27
- “Proxy Service - Service Policy Configuration page” on page 3-30
- “Proxy Service SOAP Binding Configuration page” on page 3-31
- “Proxy Service Transport Configuration page” on page 3-32

Proxy Service General Configuration page

Use the Proxy Service General Configuration page to set or modify general configuration properties for a proxy service. This page appears both in the New Proxy Service wizard and in the Proxy Service editor: Options vary, depending on whether you are using the wizard or the editor, as described below.

New Proxy Service Wizard Options

The following table describes the options in the wizard:

Option	Description
Description	Enter a description for this proxy service.
Service Type - Create a New Service	Select the type of proxy service to create, as described below.
WSDL Web Service	Select this option to create a proxy service based on a WSDL. Then enter the WSDL name, qualified by its path (for example, `myProject/myFolder/myWSDL`). Alternatively, click **Browse** to select a WSDL resource. **(port or binding)** - Enter the name of a port (defined in the WSDL) to describe an actual transport address, or enter the name of a binding (defined in the WSDL) to map to a transport address. If you use **Browse** to select a WSDL, the **Select a WSDL Definition** dialog lists any defined ports and bindings. **Note:** If you are going to use the SOAP Body Type for operations, ensure that the WSDL does not have two operations with the same input message. The SOAP Body Type operation cannot be uniquely identified by inspecting the input message.
Messaging Service	Select this option to create a service that can receive messages of one data type and respond with messages of a different data type. These exchanges can be either request/response or one-way. (HTTP GET is supported only in the Any XML Service and Messaging Service service types.)
Any SOAP Service	Select this option to create a SOAP service that does not have an explicitly defined, concrete interface. Select **SOAP 1.1** or **SOAP 1.2** from the drop-down list to specify the SOAP version to be used.
Any XML Service	Select this option to create an XML service that does not have an explicitly defined, concrete interface. (HTTP GET is supported only in the Any XML Service and Messaging Service service types.)

Option	Description
Service Type - Create From Existing Service	Choose one of these options to create a service based on another service. **Note:** When a proxy service is created from another service, alert rules are maintained in the following way: – When a proxy service is created from a business service or a business service is created from a proxy service, the alert rules, if any, are removed. – When a proxy service is created from another proxy service or a business service is created from another business service, the alert rules, if any, are retained.
Business Service	Select this option to create a proxy service with a route node that routes to the business service you select. You cannot create a proxy service from a transport typed business service. If you create a proxy service from a DSP transport business service, Oracle Service Bus switches the transport type of the proxy service to HTTP, because the DSP transport cannot be used for proxy services. You can change the transport type of the proxy service to any other available transport. Enter the path (`project/folder`) and the name of the business service; or click **Browse** to select a service.
Proxy Service	Select this option to clone an existing proxy service. Enter the path (`project/folder`) and the name of the proxy service, or click **Browse** to select a service. Since Oracle Service Bus does not accept the same URI for multiple services, you must change the URI for the cloned service.

Proxy Service Editor Options

The following table describes the options in the editor:

Option	Description
Description	Enter a description for this proxy service.
Service Type (editor only)	This option shows the service type of the proxy service. You can change only some of the properties of some of the service types: • **WSDL Web Service** - You can enter (or click **Browse** to select) a different port or binding from the same WSDL. You can also specify a different WSDL, but by doing so, you are effectively creating a new service and you will have to configure it as if it were a new service. • **Transport Typed Service** - This option cannot be modified. • **Messaging Service** - This option cannot be modified. • **Any SOAP Service** - You can change the SOAP version (SOAP 1.1 or SOAP 1.2) • **Any XML Service** - This option cannot be modified.

Proxy Service Message Level Security Configuration page

Use the Proxy Service Message Level Security Configuration page to configure message-level security for the proxy service. This page appears both in the New Proxy Service wizard and in the Proxy Service editor:

Message-level custom tokens and message-level user name and password are supported on proxy services of the following binding types:

- WSDL-SOAP
- WSDL-XML
- Abstract SOAP
- Abstract XML
- Mixed - XML (in the request)
- Mixed - MFL (in the request)

The configuration for both custom user name/password and custom token is similar. In both cases, you specify XPath expressions that enable Oracle Service Bus to locate the necessary information. The root of these XPath expressions is as follows:

- Use `soap-env:Envelope/soap-env:Header` if the service binding is AnySOAP or WSDL-SOAP.
- Use `soap-env:Body` if the service binding is not SOAP based.

All XPath expressions must be in a valid XPath 2.0 format. The XPath expressions must use the XPath "declare namespace" syntax to declare any namespaces used, as follows:

```
declare namespace
ns='http://webservices.mycompany.com/MyExampleService';)
```

Option	Description
Service Key Provider	The name of a service key provider to be used by the service. You can enter the path (`project/folder`) and name of a service key provider, or click **Browse** to select one. A service key provider is only required in certain cases: • Outbound two-way TLS/SSL, where the proxy service routes messages to HTTPS services that require client-certificate authentication. • In some Web Service security scenarios, for example, if the proxy service requires messages to be encrypted. To add a Web service security-enabled proxy service, you must create the proxy service from a WSDL (port or binding) with WS-policy attachments.
Custom Authentication Settings	Select one of the following: • **None** - if the service will not use custom authentication. • **Custom User Name and Password** - if the service will use a custom name and password, specified as XPath expressions • **Custom Token** - if the service will use a custom token
Custom User Name and Password - User Name XPath	The user name, specified as an XPath expression. The XPath expression is evaluated against the message headers or payload, as appropriate, which allows Oracle Service Bus to obtain the user name and for custom authentication.

Option	Description
Custom User Name and Password - User Password XPath	The password, specified as an XPath expression. The XPath expression is evaluated against the message headers or payload, as appropriate, which allows Oracle Service Bus to obtain the password values for custom authentication.
Custom Token - Token Type	Enter the type for the custom token type. Only the active token types configured for a WebLogic Server Identity Assertion provider can be used. See Configuring Identity Assertion Providers for Custom Tokens for more information.

Option	Description
Custom Token - Token XPath	An XPath expression that specifies a path to the custom token. Oracle Service Bus evaluates the Token XPath expression against the message headers or payload, as appropriate, to obtain the token for custom authentication. To create or edit an expression, click **<XPath>** (or the ***`expression_fragment`***, if one is already defined) to display the XPath Expression Editor.
Custom User Name and Password - Context Properties *or* Custom Token - Context Properties	Optionally, specify one or more context properties to pass additional context information to the Authentication (Custom User Name and Password) or Identity Assertion (Custom Token) security provider. Context Properties provide a way (the `ContextHandler` interface) to pass additional information to the WebLogic Security Framework so that a security provider can obtain contextual information. See Additional Context Properties for Message Level Authentication for more information. Enter the **Property Name** as a literal string, and the **Value Selector** as a valid XPath expression. (XPath expressions can also be literal strings.) The XPath expressions are evaluated against the same message-part that is used for the custom token or custom user name/password. That is, the **Value Selector** XPath expressions for SOAP-based proxy services evaluate against the header and against the payload for non-SOAP-based proxy services. The XPath expression is evaluated at runtime to produce the property's value. A ContextHandler is essentially a name/value list and, as such, it requires that a security provider know what names to look for. Therefore, the XPath expressions are evaluated only if a security provider asks for the value of one of these user-defined properties. Click **Add Property** to add this context property. You can add multiple context properties.

Proxy Service Message Type Configuration page

Use Proxy Service Message Type Configuration page to configure message types for a proxy service whose type is Messaging Service. This page appears both in the New Proxy Service wizard and in the Proxy Service editor:

The binding definition for messaging services consists of configuring the content-types of the messages that are exchanged. The content-type for the response does not have to be the same as

for the request; therefore, the response is configured separately (for example, the service could accept an MFL message and return an XML acknowledgment receipt).

Note: E-mail, File, FTP, or SFTP transport proxy services whose type is Messaging Service support one-way messaging *only*; the **Response Message Type** should be **None**. If you select an option other than **None**, the E-mail, File, FTP, or SFTP protocols will not be available on the Transport Configuration page.

Option	Description
Request Message Type	Select a message type for the request message: • **None** - Select this option if there is no request message. • **Binary** - Select this option if the content-type of the message is unknown or not important. • **Text** - Select this option if the message can be restricted to text. • **MFL** - Select this option if the message is a binary document conforming to an MFL definition. Enter the MFL file name (qualified by its path), or click **Browse** to select a file. You can configure only one MFL file. **Note:** To support multiple MFL files, define the content as binary or text and use the MFL action in the message flow to convert to XML. • **XML** - Select this option if the message is an XML document. Enter the XML file name (qualified by its path), or click **Browse** to select a file. Optionally provide some type information by declaring (in the **element or type** field) the XML schema type of the XML document exchanged.
Response Message Type	Select a message type for the response message: • **None** - Select this option if there is no response message. • **Binary** - Select this option if the content-type of the message is unknown or not important. • **Text** - Select this option if the message can be restricted to text. • **MFL** - Select this option if the message is a binary document conforming to an MFL definition. Enter the MFL file name (qualified by its path), or click **Browse** to select a file. You can configure only one MFL file. **Note:** To support multiple MFL files, define the content as binary or text and use the MFL action in the message flow to convert to XML. • **XML** - Select this option if the message is an XML document. Enter the XML file name (qualified by its path), or click **Browse** to select a file. Optionally provide some type information by declaring (in the **element or type** field) the XML schema type of the XML document exchanged.

Proxy Service Operation Selection Configuration page

Use Proxy Service Operation Selection Configuration page to enforce WS-I compliance (for SOAP 1.1 services only) and select the selection algorithm to use to determine the operation called by this proxy service. This option is only available for SOAP or XML services defined from a WSDL.

The WSDL specification defines a default algorithm to compute which operation is called based on the type of the SOAP message received. However, there are cases (for example, performance issues, signature/encryption issues, or the default algorithm is not applicable) when you may need to select the operation based on other means.

Oracle Service Bus provides additional algorithms. Each of them follows the same pattern and are based on the evaluation of an expression to get a value that is then used to lookup the corresponding operation in a static table.

Oracle Service Bus is generally very forgiving if an inbound message is either missing data such that the operation cannot be determined, or has data that does not correspond to a valid operation. Both of these conditions result in `$operation` being empty. Rather than reject all such messages, Oracle Service Bus does not initialize the operation variable in the context but otherwise continues to process the message.

However, security requirements are enforced if the proxy service is WSDL-based and at least one of the following conditions is true:

- The WSDL has a WS-Security policy and the proxy is an active intermediary.
- The proxy has message-level custom authentication (either custom token or username/password).

If these conditions are met, then there is a runtime check to make sure the operation selection algorithm returns a valid operation name. If the operation selection returns null or an operation that is not in the WSDL, then the message is rejected and an error is raised.

Option	Description
Enforce WS-I Compliance	Select or deselect this check box to specify whether or not the service is to conform to the Basic Profile defined by the Web Services Interoperability Organization. This option is available for or SOAP 1.1 services only When a service is marked WS-I compliant, checks are performed against the messages sent to and from that service. For proxies, checks are performed against request messages received by the proxy. For invoked services (i.e. services invoked by a proxy via service callout action or route node), checks are performed against the response messages received from those services. Note that it is the WS-I compliance property of the invoked service and not the proxy that determines whether or not checks are performed against messages received from the invoked service. If you specify WS-I compliance testing for an invoked service, the message flow generates a fault for response errors.

Option	Description
Selection Algorithm	Select one of the following and perform any required additional steps: • **Transport Header** - Select this algorithm to define the transport header that contains the lookup value. Then: – In the **Header Name** field, enter the transport header that extracts the value used as a key to select the operation being invoked. – Under the **Operation Mapping** field, specify the value for each operation in the **Value** field. The value is used as the key of the operation. • **SOAPAction Header** - Select this algorithm to specify that operation mapping be done automatically from the WSDL associated with this proxy service. • **WS-Addressing** - Select this algorithm to specify that the lookup value is contained by the WS-Addressing `Action` tag located in the SOAP headers of the SOAP message. Then, under the **Operation Mapping** field, specify the value for each operation in the **Value** field. The value is used as the key of the operation. • **SOAP Header** - Select this algorithm to define an XPath expression to be evaluated against the SOAP headers. This allows you to get the lookup value. Then: – In the **XPath Expression** field, specify the XPath expression that extracts the value used as a key to select the operation being invoked. – In the **Operation Mapping** field, specify the value for each operation in the **Value** field. The value is used as the key of the operation. • **SOAP Body Type** - This is the default algorithm defined by the WSDL specification to compute which operation is called based on the type of the SOAP message received. If the proxy service is configured for a Web Service security pass-through scenario with an encrypted body, you cannot select this algorithm. A similar caveat applies to pass-through encrypted SOAP headers. If you have a WSDL that has two operations with the same input message, do not select this algorithm for operations, because the operation cannot be uniquely identified by inspecting the input message. • **Payload Type** - Available only for XML services based on a WSDL port or WSDL binding.

Option	Description
Header Name	This option is available only when the **Selection Algorithm** option is set to **Transport Header**. Enter the transport header that extracts the value used as a key to select the operation being invoked.
XPath Expression	This option is available only when the **Selection Algorithm** option is set to **SOAPHeader**. Specify the XPath expression that extracts the value used as a key to select the operation being invoked.
Operation Mapping	This option is available only when the **Selection Algorithm** option is set to **Transport Header**, **WS-Addressing,** or **SOAP Body Type**. Specify the value for each operation in the **Value** field. The value is used as the key of the operation.

Proxy Service Message Content Handling Configuration page

Use the Proxy Service Message Content Handling Configuration page to specify whether the proxy service should stream message content, how the service is to decode request messages received and encode response messages sent, and whether the proxy service should stream MIME attachments instead of buffering the attachment contents in memory.

Using this page, you can enable the proxy service to stream message content rather than storing it in memory. You can also enable the proxy service to decode and parse inbound messages in MTOM/XOP format and to send responses using the MTOM/XOP format, when appropriate. SOAP Message Transmission Optimization Mechanism (MTOM) is a method of sending binary data to and from Web services. MTOM uses XML-binary Optimized Packaging (XOP) to transfer the binary data.

Using this page, you can also enable the proxy service to store MIME attachment content to a disk file and then process the data in a streaming fashion without buffering the attachment contents in memory. This enables the proxy service to process large attachments robustly and efficiently.

Option	Description
Content Streaming	Select this option to stream message content rather than store it in memory. Select the **Enabled** check box and choose the following: • Whether to buffer the intermediate content in memory (**Memory Buffer**) or to a disk file (**Disk Buffer**) • Whether to enable **Compression** For more information, see Streaming body Content in the *Oracle Service Bus User Guide*.

Option	Description
XOP/MTOM Support	Oracle Service Bus supports XOP/MTOM using the following transports: • HTTP/S • Local • SB Select the **Enabled** check box to enable the proxy service to decode and parse inbound messages in MTOM/XOP format and to send responses using the MTOM/XOP format, when appropriate. Note that this option is disabled for imported proxy services that are based on previous release configurations. If XOP/MTOM Support is enabled, select how to handle binary data in the $header and $body message context variables from among the following options: • **Include Binary Data by Reference**: (Default) In an inbound request message, replace xop:Include elements with ctx:binary-content elements when setting up the $header and $body message context variables. • **Include Binary Data by Value**: In an inbound request message, replace xop:Include elements with base64-encoded text versions of corresponding binary data when setting up the $header and $body message context variables. Use Include Binary Data by Reference when you need direct access to binary data, for example to pass data to a Java callout or Message Format Language (MFL) transform. Use Include Binary Data by Value in the following cases: • To bridge between MTOM and non-MTOM services. For example, consider an MTOM-enabled proxy service that receives a request that is then routed to a non-MTOM-enabled service. You could use this option to comply with existing standards for sending binary data in XML in base64-encoded form. • To validate the contents of the message against an XML schema that requires a base64binary element to be used in place of binary data Note that if XOP/MTOM Support is enabled for a proxy service, it is not required that every inbound message be in the MTOM format. Instead, this setting specifies that when an MTOM-formatted message arrives, the proxy service should handle it accordingly. Note also that when proxy services not enabled for XOP/MTOM Support receive an MTOM-formatted message, the service rejects the message and issues a runtime error.

Option	Description
Attachments	Oracle Service Bus supports streaming MIME attachments using the following transports: • HTTP/S • Local (when chained through an HTTP proxy with streaming attachments enabled) Select the **Page Attachments to Disk** check box to enable the proxy service to stream MIME attachments. When enabled for HTTP proxy services, the option applies to proxy service inbound request messages. Note that if you select XOP/MTOM Support, the Attachments option is only available if you choose the **Include Binary Data by Reference** option under XOP/MTOM Support. Note also that payloads that contain attachments must conform to RFC 822. Specifically, lines containing Internet headers need to be terminated with CRLF (carriage return line feed).

Proxy Service - Service Policy Configuration page

Use Proxy Service - Service Policy Configuration page to configure service policies for a proxy service. This page appears both in the New Proxy Service wizard and in the Proxy Service editor:

Option	Description
Option	**Description**
WSDL-Based Policy	Select this option if the service policy is associated with the WSDL upon which the service is based.
Custom Policy Bindings	Select this option to add service-level policies, operation-level policies (in which case the policy applies to both the request and response messages), request policies, and response policies directly. For more information about configuring service policies, see Configuring Message Level Security for Web Services in the *Oracle Service Bus Security Guide*.

Proxy Service SOAP Binding Configuration page

This page is displayed only if the service you are creating has operations. This page appears both in the New Proxy Service wizard and in the Proxy Service editor:

Use the Proxy Service SOAP Binding Configuration page to enforce WS-I compliance (for SOAP 1.1 services only) and select the selection algorithm to use to determine the operation called by this proxy service. This option is only available for SOAP or XML services defined from a WSDL.

The WSDL specification defines a default algorithm to compute which operation is called based on the type of the SOAP message received. However, there are cases (for example, performance issues, signature/encryption issues, or the default algorithm is not applicable) when you may need to select the operation based on other means.

Oracle Service Bus provides additional algorithms. Each of them follows the same pattern and are based on the evaluation of an expression to get a value that is then used to lookup the corresponding operation in a static table.

Oracle Service Bus is generally very forgiving if an inbound message is either missing data such that the operation cannot be determined, or has data that does not correspond to a valid operation. Both of these conditions result in `$operation` being empty. Rather than reject all such messages, Oracle Service Bus does not initialize the operation variable in the context but otherwise continues to process the message.

However, security requirements are enforced if the proxy service is WSDL-based and at least one of the following conditions is true:

- The WSDL has a WS-Security policy and the proxy is an active intermediary.
- The proxy has message-level custom authentication (either custom token or user name/password).

If these conditions are met, then there is a runtime check to make sure the operation selection algorithm returns a valid operation name. If the operation selection returns null or an operation that is not in the WSDL, then the message is rejected and an error is raised.

Option	Description
Enforce WS-I Compliance	For SOAP 1.1 services only: Select or deselect this check box if you want to specify whether or not the service is to conform to the Basic Profile defined by the Web Services Interoperability Organization. When a service is marked WS-I compliant, checks are performed against the messages sent to and from that service. For proxies, checks are performed against request messages received by the proxy. For invoked services (i.e. services invoked by a proxy via service callout action or route node), checks are performed against the response messages received from those services. Note that it is the WS-I compliance property of the invoked service and not the proxy that determines whether or not checks are performed against messages received from the invoked service. If you specify WS-I compliance testing for an invoked service, the message flow generates a fault for response errors.

Proxy Service Transport Configuration page

Use the Proxy Service Transport Configuration page to select a transport protocol for the proxy service and to set other general transport configuration settings.

This page appears both in the New Proxy Service wizard and in the Proxy Service editor:

Note: Inbound transport-level security applies to the client applications and Oracle Service Bus proxy services. Outbound transport-level security applies to the connections between Oracle Service Bus proxy services and business services. To learn more about transport-level security, see Configuring Transport-Level Security in the *Oracle Service Bus Security Guide*.

Option	Description
Protocol	Select a transport protocol from the list. The protocols available differ, depending on the service type you are creating: • **WSDL Web Service:** http, jca, jms, local, sb, ws • **Messaging Service:** email, file, ftp, http, jms, local, mq (if available), sftp, tuxedo • **Any SOAP Service:** http, jms, local, sb • **Any XML Service:** email, file, ftp, http, jms, local, mq (if available), sb, sftp, tuxedo

Option	Description
Endpoint URI	Enter an endpoint URI in the format based on the transport protocol you selected in the **Protocol** field, above: The formats are: • **email** - `mailfrom:mail-server-host:port` • **file** - `file:///root-dir/dir1` • **ftp** - `ftp://hostname:port/directory` • **http** - /`someName` The HTTP transport supports both `HTTP` and `HTTPS` endpoints. • **jca** - `jca://<resource_adapter_jndi>` • **jms** - `jms://host:port[,host:port]*/factoryJndiName/destJndiName` To target a target a JMS destination to multiple servers, use the following URI format: `jms://host1:port,host2:port/QueueConnectionFactory/DestName` Note that when you create a proxy service, you can configure a JMS endpoint URI even if the server at that endpoint if not available. However, in the case of JMS, when you activate the session, the endpoint must be available. **In a cluster**: The host names in the JMS URI must exactly match the host names of the cluster servers as they are configured in WebLogic Server. • **local** This transport does not require an endpoint URI. • **mq** - `mq://local-queue-name?conn=mq-connection-resource-ref` `local-queue-name` is the name of the MQ queue from which the proxy service reads messages. `mq-connection-resource-ref` is the path (project/folder) and name of the MQ connection resource; for example, `default/my_MQconnection`. **Note:** The Endpoint URI cannot contain spaces, so do not create MQ Connection resources or projects/folders with spaces in the names. To make the MQ transport available in Oracle Service Bus, see the MQ Transport User Guide. • **sb** - `service_name` `service_name` is the unique identifier for the proxy service. By default, this name will be the proxy service name. `service_name` must only contain characters permitted in URIs (as described in RFC2396), except it cannot contain forward slash (/) or colon (:) characters. • **sftp** - `sftp://hostname:port/directory`

Option	Description
Endpoint URI (continued)	• **tuxedo** - `servicename` The URI `servicename` corresponds to a WTC Export that the remote Tuxedo domain identifies as a Tuxedo service. If more than one URI is specified, you must have unique resource names for the endpoints. If no remote name is specified, its value is the value of the resource name. If no remote name is entered or if remote and resource name are the same, only one URI is allowed. In this case resource name and remote name will have the same value. This allows users using already defined WTC Imports to make use of WTC load-balancing and failover. **Note:** If you configure two identical URIs, an error indicates that the service name already exists. • **ws** - `/contextPath` `contextPath` must be unique for proxy services that use either HTTP or WS transport.
Get All Headers	Select **Yes** to retrieve all the headers from the transport. Select **No** to retrieve a defined set of headers. If you select **No**, enter a set of headers in the **Header** field, then click **Add**. (This step does not apply to Local transport.) **Note:** Oracle Service Bus does not pass the HTTP Authorization header from the request to the pipeline because it opens a security vulnerability. You could inadvertently create a log action that writes the user name and unencrypted password to a log file. If your design pattern requires the HTTP Authorization header to be in the pipeline, do the following: a. In the startup command for Oracle Service Bus, set the following system property to true: `com.bea.wli.sb.transports.http.GetHttpAuthorizationHeaderAllowed` b. In Eclipse, on the Transport Configuration page, select **Yes** for **Get All Headers** or select **No** and specify Authorization. c. Restart Oracle Service Bus. Oracle Service Bus will pass the Authorization header to the pipeline.

Oracle Service Bus Configurations and Projects

The following are provided for working with Oracle Service Bus configurations and projects.

- Oracle Service Bus Configurations view
- New Oracle Service Bus Configuration Project wizard
- New Oracle Service Bus Project wizard

Oracle Service Bus Configurations view

This view displays a hierarchical list of all your Oracle Service Bus configurations and the projects they contain. See also "New Oracle Service Bus Configuration Project wizard" on page 3-35.

You can do the following in the Configurations view:

- Edit the configuration:
  a. Right click the configuration you want to edit.
  b. Select **Properties** from the menu.
  c. Modify information as needed in the Oracle Service Bus Configuration page.
- Create a new configuration:
  a. Right-click anywhere in the view.
  b. Select **New Oracle Service Bus Configuration** from the menu.
  c. Enter information as needed in the Oracle Service Bus Configuration page.
- Move a project from one configuration to another by dragging it from the source configuration to the target configuration. You can also drag a project from the Project Explorer.
- Delete a configuration project:
  a. Right-click the configuration you want to delete.
  b. Select **Delete** from the menu.

New Oracle Service Bus Configuration Project wizard

Use this wizard to create an Oracle Service Bus configuration project. For configuration options, see Oracle Service Bus Configuration page.

Oracle Service Bus Configuration page

An Oracle Service Bus configuration project is a grouping of Oracle Service Bus projects and resources destined for a server, a location for system resources (SMTP, UDDI, and such), and a container for validation; for example, a resource in a project associated with one Oracle Service Bus configuration cannot refer to a resource in a project associated to another Oracle Service Bus configuration.

Use this page to create a new Oracle Service Bus configuration project (in the New Oracle Service Bus Configuration Project wizard) or to configure an existing one (in the Properties for Oracle Service Bus Configuration editor).

The preserve, session, and customization file settings are used when publishing an Oracle Service Bus configuration to the server, and used as defaults when configuring the Export wizard - Export to Server - Select Resources page.

Option	Description
Configuration name	Enter a name for this Oracle Service Bus configuration or keep the default.
Advanced Options	
Preserve environment variable values	Select this check box when you are re-importing a resource but want to preserve environment variable values in the existing resource.
Preserve security and policy settings	Select this check box to preserve the security configuration (excluding access control policies) and the references to the WS-policies bound directly to the service (instead of bound to the WSDL).
Preserve credentials (user name and password)	Select this check box to preserve PKI credentials in service key providers, user name and passwords in service accounts, and user name and password credentials in SMTP servers, JNDI providers, and UDDI registries.
Discard session if activation fails	Select this check box to discard the session if the activation fails (for example, due to conflicts). For more information, see Activating Sessions in *Using the Oracle Service Bus Console*.
Session Name	The session name.
Description	The session description.
Deployment customization file	Specify a *customization*.xml file or click **Browse**, locate the file, then click **Open**. For information on customization, see Creating Customization Files in *Using the Oracle Service Bus Console*.

Option	Description
Keystore file	Specify a *keystore*.jks file or click **Browse**, locate the file, then click **Open**. The key store settings are used when configuring a service key provider.
Password	Enter the password that you use to secure access to the key store.
Server	The name of the server associated with this Oracle Service Bus configuration. This setting is automatically configured unless there is more than one server from which to choose. When multiple servers are associated with the same Oracle Service Bus configuration, use the drop-down list to select the server you want to associate with this Oracle Service Bus configuration. The server setting is only used for transport specific configuration, when the transport benefits from being connected to a server (for example, when configuring the dispatch policy setting in the HTTP transport).

New Oracle Service Bus Project wizard

Use this page to create a new Oracle Service Bus project.

Option	Description
Project name	Enter a unique name for the project. For more information, see Naming Projects and Folders in *Using the Oracle Service Bus Console*.
Project contents	Select the **Use default** check box to keep the project in the default location or click **Browse** and select a folder.
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration or click **New** to open the New Oracle Service Bus Configuration Project wizard, where you can create a new configuration.

Custom Resources

You can define custom resources for use by Oracle Service Bus using the New Custom Resource wizard and the Custom Resource editor, as described in the following topics:

New Custom Resource wizard

Use this wizard to create a new custom resource.

New Custom Resource Editor

Use this editor to modify the configuration of a custom resource. The General page identifies the type of custom resource. The Custom page provides options for editing the configuration for that type of custom resource.

New Custom Resource - Resource Type page

Use this page to select the type of custom resource to create. Select the type of custom resource to create from the list of available resource type.

Custom MQ Resource Configuration page

Use this page to configure a custom MQ connection resource. For information on MQ connection resources and MQ transports, see the following:

- MQ Connections in *Using the Oracle Service Bus Console*
- *MQ Transport User Guide*

Option	Description
Connection Type	Select one of the following modes for connecting to the MQ queue manager: • `tcp mode`—Use TCP/IP to connect to a queue manager that does not reside on the same machine as Oracle Service Bus. • `binding mode`—Use the bindings mode to connect to a queue manager that is located on the same machine as Oracle Service Bus.
MQ Host Name	For `tcp mode` connections only: Enter the host name of the MQ queue manager.
MQ Port Number	For `tcp mode` connections only: Enter the port number of the MQ queue manager listener.
MQ Queue Manager Name	Enter the name of the MQ queue manager to which to connect.
Queue Manager CCSID	For `tcp mode` connections only: The coded character set identifier (CCSID) to be used when establishing a connection. The CCSID is used mainly for internationalization support. To learn more, see IBM's WebSphere MQ Fundamentals.
MQ Queue Manager Channel Name	For `tcp mode` connections only: Enter the queue manager server connection channel name.
SSL Required	For `tcp mode` connections: Select the check box to use SSL for sending messages. Only server-side SSL will be enabled when the **2-way SSL Required** option is *not* selected.
Cipher Suite	This option is available only when the **SSL Required** check box is selected. Select the Cipher Suite algorithm to be used by SSL. The Cipher Suite algorithm is used to encrypt and decrypt message communications between the WebSphere MQ server and the WebSphere MQ client. Thus a Cipher Suite algorithm must be specified when using SSL to communicate with a WebSphere MQ server.

Option	Description
2-way SSL Required	This option is available only when the **SSL Required** check box is selected. Select the check box to enable both client-side and server-side SSL authentication.
Reference to the Service Key Provider	If you selected **2-way SSL Required**, you must provide a reference to the service key provider for obtaining the appropriate key store and trust store information. Enter the path (`project/folder`) and name of a service key provider, or click **Browse** to select one from the Select Service Key Provider page.
Reference to the Static Service Account	For `tcp mode` connections only: Required for user name and password authentication. Enter the path (`project/folder`) and name of a static service account, or click **Browse** to select service accounts from a browser.
WebSphere MQ Version	Select the WebSphere MQ version: • 5.3 • 6.0
MQ Connection Pool Size	Enter the size of the MQ connection pool.
MQ Connection Timeout	Enter the time interval in seconds after which unused connections are destroyed. The default is 1800 seconds.
MQ Connection Max Wait	Enter the Max Wait in seconds for the amount of time to wait for a connection to become available. If a connection is not made within that time interval, Oracle Service Bus throws an exception. The default is 3 seconds.

Export wizard

The Export wizard contains the following pages:

- "Export wizard - Oracle Service Bus Configuration JAR Export page" on page 3-41
- "Export wizard - Export to Server - Select Resources page" on page 3-41
- "Export wizard - Export to Server - Review Resources page" on page 3-42

Export wizard - Oracle Service Bus Configuration JAR Export page

Use this page to export Oracle Service Bus resources to a configuration JAR file. For more information, see *Exporting Resources* in *Using the Oracle Service Bus Console*.

Option	Description
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration and resources to export.
Export Level	Select whether to export projects or resources. **Note:** A **System** project cannot be exported at the project level. **Note:** Exporting projects might cause resource deletion when you import the full project JAR file. For more information, see *Importing Resources* in *Using the Oracle Service Bus Console*.
Include Dependencies	If you selected to export resources, select or clear this check box. Use the **Include Dependencies** option to export any other resources that this resource references
Export Destination	Enter the fully qualified name of a JAR file to export, or click **Browse** to select it.

Export wizard - Export to Server - Select Resources page

Use this page to select the projects or resources you want to export. Clear the check boxes next to any resources that you do not want to include in this export.

Option	Description
Resource	The name of the project and resource.
Operation	Create or update. The operation that will be performed on the resource.
Include Dependencies	Select this check box if you are exporting resources (not projects) and want to ensure that all the associated resources are exported.

Option	Description
Preserve Environment Settings	
Preserve environment variable values	Select this check box when you want to preserve (protect against overwriting) the environment variables values in the resource you are exporting.
Preserve security and policy settings	Select this check box to preserve the security configuration (excluding access control policies) and the references to the WS-policies bound directly to the service (instead of bound to the WSDL).
Preserve credentials (user name and password)	Select this check box to preserve PKI credentials in service key providers, user name and passwords in service accounts, and user name and password credentials in SMTP servers, JNDI providers, and UDDI registries.
Session Settings	
Activate session after publish	Select this check box to create and activate a session in the Oracle Service Bus console. For more information, see Activating Sessions in *Using the Oracle Service Bus Console*.
Discard session if activation fails	Select this check box to discard the session if the activation fails (for example, due to conflicts).
Description	The session description.
Customization File	Specify a customization file or click **Browse**, locate the file, then click **Open**. For information on customization, see Creating Customization Files in *Using the Oracle Service Bus Console*.

Export wizard - Export to Server - Review Resources page

Use this page to select the projects or resources you want to export.

Option	Description
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration and resources to export.
Export Level	Select whether to export projects or resources.

Option	Description
Server	From the drop-down list, select an Oracle Service Bus destination server.
Session	The session name.

Import wizard

The Import wizard has the following pages:

- "Import wizard - Config JAR Import - Load Resources page" on page 3-43
- "Import wizard - Config JAR Import - Review Resources page" on page 3-43
- "Import wizard - Config ZIP Import - Load Resources page" on page 3-44
- "Import wizard - Config ZIP Import - Review Resources page" on page 3-45
- "Import wizard - URL Import - Load Resources page" on page 3-45
- "Import wizard - URL Import - Review Resources page" on page 3-46

Import wizard - Config JAR Import - Load Resources page

Use this page to select a configuration JAR to import and an import destination. For more information, see *Importing Resources* in *Using the Oracle Service Bus Console*.

Option	Description
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration or click **New** to open the New Oracle Service Bus Configuration Project wizard, where you can create a new configuration.
Jar	Select a full project JAR file or a resource JAR file that has been previously exported from another Oracle Service Bus domain.

Import wizard - Config JAR Import - Review Resources page

Use this page to select the objects on which you want to operate (create, update, or delete). Resources are only scheduled for deletion when the JAR being imported is a full project JAR and there are project resources in the importing system that are not present in the imported JAR file. To prevent resources

from being deleted, deselect them. For more information, see *Importing Resources* in *Using the Oracle Service Bus Console*.

Option	Description
Resources	The name of the project and resource.
Operation	Expand the Project folder to display the operations (create, update, delete) that will be performed on the resources. To prevent resources from being deleted, deselect them. Resources are only scheduled for deletion when the JAR being imported is a full project JAR and there are project resources in the importing system that are not present in the imported JAR file.
Include Dependencies	Select this check box if you are importing a resource JAR file and want to ensure that all its associated resources are imported.
Passphrase	Enter the password that was used to encrypt the data.
Preserve Environment Settings	
Preserve environment variable values	Select this check box when you are re-importing a resource but want to preserve environment variable values in the existing resource.
Preserve security and policy settings	Select this check box to preserve the security configuration (excluding access control policies) and the references to the WS-Policies bound directly to the service (instead of bound to the WSDL).
Preserve credentials (user name and password)	Select this check box to preserve PKI credentials in service key providers, user name and passwords in service accounts, and user name and password credentials in SMTP servers, JNDI providers, and UDDI registries.

Import wizard - Config ZIP Import - Load Resources page

Use this page to select a ZIP file with resources to import and an import destination. For more information, see Loading Resources from a Zip File in *Using the Oracle Service Bus Console*.

Option	Description
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration and an import destination (project or folder).
File Name	Enter the fully qualified name of a ZIP file to import, or click **Browse** to select it.

Import wizard - Config ZIP Import - Review Resources page

Use this page to select the objects you want to import. Clear the check boxes next to any resources that you do not want to include in this import. For more information, see Loading Resources from a Zip File in *Using the Oracle Service Bus Console*.

Option	Description
Resource	The name of the project and resource.
Operation	Create or update. The operation that will be performed on the resource.
File Name	The file name of the resource, including the file extension.

Import wizard - URL Import - Load Resources page

Use this page to import resources such as WSDLs or XML schemas that are available on the Web. You can import them, along with all their dependents, by specifying the URL of the root resource. For more information, see Loading Resources from a URL in *Using the Oracle Service Bus Console*.

Option	Description
Oracle Service Bus Configuration	Select an existing Oracle Service Bus configuration and an import destination (project or folder).
URL	Enter the URL where the file is located. To specify a local resource, you can use the `file` protocol with a file name fully qualified by its path, for example, `file:///c:/alsbresources/ForeachAction.jar`.

Option	Description
Resource Name	Enter a name for the resource.
Resource Type	Select the type of resource from the list.

Import wizard - URL Import - Review Resources page

Use this page to select the objects you want to import. Clear the check boxes next to any resources that you do not want to include in this import. For more information, see Loading Resources from a URL in *Using the Oracle Service Bus Console*.

Option	Description
Resource	The name of the project and resource.
Operation	Create or update. The operation that will be performed on the resource.
URL	Enter the URL where the file is located. To specify a local resource, you can use the `file` protocol with a file name fully qualified by its path, for example, `file:///c:/alsbresources/ForeachAction.jar`.

JNDI Providers

The following are provided for working with JNDI providers:

- JNDI Provider Editor
- New JNDI Provider Resource wizard

JNDI Provider Editor

Use this editor to modify an existing JNDI provider configuration. For descriptions of the fields, see the New JNDI Provider Resource wizard. For more information, see Editing JNDI Providers in *Using the Oracle Service Bus Console*.

New JNDI Provider Resource wizard

Use this page to configure a new JNDI provider resource. For more information, see Adding JNDI Providers in *Using the Oracle Service Bus Console*.

Option	Description
Description	Enter a description for the JNDI provider.
JNDI Cache	Keep the default **Enabled** option or select **Disabled**. When enabled, the JNDI context and JNDI objects are cached locally which improves performance when doing an object lookup. Oracle recommends that you keep the JNDI cache enabled.
Provider URL	Enter the URL for the JNDI provider in the format: `protocol://host:port` You can use any protocol, for example, `http, https, t3, t3s, iiop, iiops`.
User Name	If access to the target JNDI provider requires a user name and password, enter a user name in the **User Name** field, and the associated password in the **Password** and **Confirm Password** fields. These fields are optional, and required only if the JNDI tree is secured.
Password	Enter the associated password.
Confirm Password	Enter the same password you entered for the **Password** field.

Proxy Servers

Use this page to configure and edit a proxy server resource. For more information, see Adding Proxy Servers in *Using the Oracle Service Bus Console*.

Option	Description
Description	Enter a description of the proxy server resource.
User Name	Enter the user name used for proxy authentication.
Password	Enter the password associated with the user name.

Option	Description
Confirm Password	Enter the same password you entered for the **Password** field.
Host-Port Parameters	The list of proxy server hosts. You can configure multiple proxy servers for each proxy server resource. This enables Oracle Service Bus to perform load balancing and offer fault tolerance features for the proxy server resource. The following options are available: • Click **Add** and specify the following information to configure a new proxy server for the resource: – Server Host: The host name or IP address of the proxy server. The Server Host name for the Oracle Service Bus proxy server must be identical to the server host name of the actual proxy server. – Clear Text Port: The proxy server clear-text port number. – SSL Port: The proxy server SSL port number. • Select a proxy server in the list and edit the **Host-Port Parameters**. • Select a proxy server in the list and click **Delete** to remove it from the resource configuration.

Message Flow Design Palette

The Message Flow Design Palette lists all the nodes and actions you can use to construct a message flow for a proxy service. To insert a node or an action into a message flow, drag the item from the palette to the Message Flow Editor. When you drag an item to the palette, one or more targets appear () to indicate that you can drop the dragged item in that position on the palette. When you drag the item onto a target, the target is highlighted () to show that you can drop the item there.

You can also add nodes and actions to message flows by right-clicking an object in the Message Flow Editor and choosing a node or action from the **Insert**, **Insert Into**, or **Insert After** submenus. You can also add an error handler by selecting the **Add Error Handler** command. These submenus and commands are context sensitive. That is, they appear only when it is legal to add the item into the selected context in the message flow.

The Message Flow Design Palette is organized into the following categories:

Message Flow Nodes

The following table describes the nodes you can add to a message flow:

Node	Description
Conditional branch	A branch node allows processing to proceed down exactly one of several possible paths. For configuration properties, see Conditional Branch Node Properties.
Operational branch	An operational branch node determines what branch to follow based on specified operations. For configuration properties, see Operational Branch Node Properties.
Pipeline pair	A pipeline pair node consists of a request pipeline and a response pipeline. For configuration properties, see Pipeline Pair Node Properties.
Route	Route node actions define the handling of messages as they flow through the route node. For configuration properties, see Route Node Properties.
Stage	A stage node is a container of actions. For configuration properties, see Stage Node Properties.
Error handler	An error handler provides the logic for resending errors in the message flow. For configuration properties, see Error Handler Node Properties.

Message Flow Route Actions - Communication Actions

The following table describes the communication actions you can add to a route in a message flow:

Action	Description
Dynamic routing	A dynamic routing action assigns a route for a message based on routing information available in an XQuery resource. For configuration properties, see Dynamic Routing Action Properties.
Routing	A routing action identifies a target service for the message and configures how the message is routed to that service: For configuration properties, see Routing Action Properties.
Routing table	A routing table action assigns a set of routes wrapped in a switch-style condition table. Different routes are selected based upon the results of a single XQuery expression. For configuration properties, see Routing Table Action Properties.

Message Flow Route Actions - Flow Control Actions

The following table describes the flow control action you can add to a route in a message flow:

Action	Description
If-then	An if-then action performs an action or a set of actions conditionally, based on the Boolean result of an XQuery expression. For configuration properties, see If-Then Action Properties.

Message Flow Stage Actions - Communication Actions

The following table describes the communication actions you can add to a stage in a message flow:

Action	Description
Dynamic Publish	A dynamic publish action publishes a message to a service identified by an XQuery expression For configuration properties, see Dynamic Publish Action Properties.
Publish	A publish action publishes a message to a statically specified service. For configuration properties, see Publish Action Properties.
Publish Table	A publish table action publishes a message to zero or more statically specified services. Switch-style condition logic is used to determine at run time which services will be used for the publish. For configuration properties, see Publish Table Action Properties.
Routing Options	A routing options action modifies any or all of the following properties in the outbound request: URI, Quality of Service, Mode, Retry parameters, Message Priority. For configuration properties, see Routing Options Action Properties.
Service Callout	A service callout action configures a synchronous (blocking) callout to an Oracle Service Bus-registered proxy or business service. For configuration properties, see Service Callout Action Properties.
Transport Headers	A transport header action sets the transport header values in messages For configuration properties, see Transport Headers Action Properties.

Message Flow Stage Actions - Flow Control Actions

The following table describes the flow control actions you can add to a stage in a message flow:

Action	Description
For-Each	A for-each action iterates over a sequence of values and executes a block of actions For configuration properties, see For-Each Action Properties.

If-Then	An if-then action performs an action or set of actions conditionally, based on the Boolean result of an XQuery expression. For configuration properties, see If Action and Else-If Action Properties.
Raise error	A raise-error action raises an exception with a specified error code (a string) and description. For configuration properties, see Raise Error Action Properties.
Reply	A reply action specifies that an immediate reply be sent to the invoker. For configuration properties, see Reply Action Properties.
Skip	A skip action specifies that at run time, the execution of the current stage is skipped and the processing proceeds to the next stage in the message flow. For configuration properties, see Skip Action Properties.
Resume	A resume action resumes message flow after an error is handled by an error handler. For configuration properties, see Resume Action Properties.

Message Flow Stage Actions - Message Processing Actions

The following table describes the message processing actions you can add to a stage in a message flow:

Action	Description
Assign	An assign action assigns the result of an XQuery expression to a context variable. For configuration properties, see Assign Action Properties.
Delete	A delete action deletes a context variable or a set of nodes specified by an XPath expression. For configuration properties, see Delete Action Properties.
Insert	An insert action inserts the result of an XQuery expression at an identified place relative to nodes selected by an XPath expression. For configuration properties, see Insert Action Properties.

Java callout	A Java callout action invokes a Java method from the pipeline. For configuration properties, see Java Callout Action Properties.
MFL transform	A MFL transform action converts non-XML to XML or XML to non-XML in the pipeline. For configuration properties, see MFL Transform Action Properties.
Rename	A rename action renames elements selected by an XPath expression without modifying the contents of the element. For configuration properties, see Rename Action Properties.
Replace	A replace action replaces a node or the contents of a node specified by an XPath expression. For configuration properties, see Replace Action Properties.
Validate	A validate action validates elements selected by an XPath expression against an XML schema element or a WSDL resource. For configuration properties, see Validate Action Properties.

Message Flow Stage Actions - Reporting Actions

The following table describes the reporting actions you can add to a stage in a message flow:

Action	Description
Alert	An alert action ends an alert notification based on pipeline message context. For configuration properties, see Alert Action Properties.
Log	A log action constructs a message to be logged. For configuration properties, see Log Action Properties.
Report	A report action enables message reporting for a proxy service. For configuration properties, see Report Action Properties.

Message Flow Editor

Use this editor to construct a proxy service message flow. To insert a node or an action into a message flow, drag the item from the Message Flow Design Palette to the editor. When you drag an item to the editor, one or more targets appear () to indicate that you can drop the dragged item in that position on the palette. When you drag the item onto a target, the target is highlighted () to show that you can drop the item there.

You can also add nodes and actions to message flows by right-clicking an object in the Message Flow Editor and choosing a node or action from the **Insert**, **Insert Into**, or **Insert After** submenus. You can also add an error handler by selecting the **Add Error Handler** command. These submenus and commands are context sensitive. That is, they appear only when it is legal to add the item into the selected context in the message flow.

When you select a node or action in the editor, a configuration page for that item is displayed in the Properties view. You can add or modify configuration properties in those pages. The message flow node and action properties configuration pages are:

- “Alert Action Properties” on page 3-56
- “Message Flow Properties- Comment” on page 3-70
- “Assign Action Properties” on page 3-57
- “Conditional Branch Node Properties” on page 3-58
- “Conditional Branch Node Properties” on page 3-58
- “Delete Action Properties” on page 3-59
- “Dynamic Publish Action Properties” on page 3-60
- “Dynamic Routing Action Properties” on page 3-61
- “Error Handler Node Properties” on page 3-62
- “Error Handler Node Properties” on page 3-62
- “For-Each Action Properties” on page 3-63
- “If Action and Else-If Action Properties” on page 3-65
- “If-Then Action Properties” on page 3-64

Alert Action Properties

In a message flow, use the alert action to generate alerts based on message context in a pipeline, to send to an alert destination. Unlike SLA alerts, notifications generated by the alert action are primarily intended for business purposes, or to report errors, and not for monitoring system health. Alert destinations should be configured and chosen with this in mind. To learn more about alert destinations, see "Alert Destination editor" on page 3-2.

If pipeline alerting is not enabled for the service or at the domain level, the configured alert action is bypassed during message processing.

In the Message Flow Editor, click an alert action to display its properties in the Properties view. Use these properties pages to configure the selected alert action. The pages are:

- Alert
- Comment
- Namespaces
- Variables

The Alert page has the following options:

Option	Description
Expression	An XQuery expression that specifies the message context to be added to the alert message. To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Summary	A short description of the alert. This will be the subject line in the case of an e-mail notification and can contain no more than 80 characters. If no description is provided, a predefined subject line that reads, "ALSB Alert," will be used instead.
Severity	The severity level for this alert. Select a level from the list.
Destination	The destination for the alert. To specify a destination, click **Browse** to select an appropriate resource.

Tip: To prevent exceptions from aborting the message being processed when generating a pipeline alert, it is recommended that an error handler for the alert action be defined to handle and contain such exceptions locally, rather than having them bubble up to a global error handler.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Assign Action Properties

In a message flow, use an assign action to assign the result of an XQuery expression to a context variable.

In the Message Flow Editor, click an assign action to display its properties in the Properties view. Use these properties pages to configure the selected assign action. The pages are:

- Assign
- Comment
- Namespaces
- Variables

The Assign page has the following options:

Option	Description
Expression	An expression that creates the data that is assigned to variable named in the **Variable** field, described below. To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Variable	The variable to which the value created in the XQuery expression described above is assigned.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Conditional Branch Node Properties

In a message flow, use a conditional branch node to specify that message processing is to proceed along exactly one of several possible paths, based on a result returned by an XPath condition.

Conditional branching is driven by a lookup table with each branch tagged with a simple, but unique, string value. A variable in the message context is designated as the lookup variable for that node, and at run time, its value is used to determine which branch to follow. If no branch matches the value of the lookup variable, the default branch is followed. You should design the proxy service in such a way that the value of the lookup variable is set before reaching the branch node.

In the Message Flow Editor, click a conditional branch node to display its properties in the Properties view. Use these properties pages to configure the selected conditional branch node. The pages are:

- Flow
- Conditional Branch

The Flow page has the following options:

Option	Description
Name	Enter a name for the conditional branch node.
Description	Enter a description for the conditional branch node.

The Conditional Branch page has the following options:

Option	Description
XPath	The XPath expression that defines the condition to be evaluated for determining the branch to follow. To create or edit the XPath expression, click **<XPath>** (or the ***`XPath_fragment`***, if one is already defined) to display the XPath Expression Editor.
Variable	Enter a context variable.

A conditional branch node also always contains one or more branches, which are configured as part of the node.

Click a **Branch** icon in a conditional branch node to display the Branch properties page in the Properties view. Use these properties pages to configure the selected branch node in a message flow. This page has the following options.

Option	Description
Label	A label for the branch. This label appears as the label for the branch in the Message Flow Editor.
Operator	Select an operator from the list to be used with the value in the next field, for creating the condition on which the branch is based.
Value	Enter a value to be used with the operator in the previous field, for creating the condition

Delete Action Properties

In a message flow, use a delete action to delete a context variable or a set of nodes specified by an XPath expression.

In the Message Flow Editor, click a delete action to display its properties in the Properties view. Use these properties pages to configure the selected delete action. The pages are:

- Delete
- Comment

- Namespaces
- Variables

The Delete page has the following options. Select **Variable** to delete a variable, or select **XPath** to delete an XPath expression.

Option	Description
Variable	Select this radio button to delete a context variable. Enter the name of the context variable to delete in the adjacent text field.
XPath	Select this radio button to delete all nodes selected by an XPath expression. Then: • To create or edit the XPath expression, click **<XPath>** (or the ***`XPath_fragment`***, if one is already defined) to display the XPath Expression Editor. • After saving the expression, in the **In Variable** field, enter the context variable containing the nodes specified in the XPath expression.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Dynamic Publish Action Properties

In a message flow, use a dynamic publish action to publish a message to a service specified by an XQuery expression.

In the Message Flow Editor, click a dynamic publish action to display its properties in the Properties view. Use these properties pages to configure the selected dynamic publish action. The pages are:

- Dynamic Publish
- Comment
- Namespaces
- Variables

The Dynamic Publish page has the following options:

Option	Description

Service	An XQuery expression that defines the service to which a message is to be published.
	To create or edit the expression, click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor.
	In the editor, enter an Xquery expression or select an XQuery resource that provides a result similar to:
	`<ctx:route isProxy="false">`
	`<ctx:service>project/folder/businessservicename</ctx:service>`
	`<ctx:operation>foo</ctx:operation>`
	`</ctx:route>`
	Note: The element `operation` is optional.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Dynamic Routing Action Properties

In a message flow, use a dynamic routing action to assign a route for a message based on routing information available in an XQuery resource.

This is a terminal action, which means you cannot add another action after this one. However, this action can contain request and response actions.

In the Message Flow Editor, click a dynamic routing action to display its properties in the Properties view. Use these properties pages to configure the selected dynamic routing action. The pages are:

- Dynamic Routing
- Comment

- Namespaces
- Variables

The Dynamic Routing page has the following option:

Option	Description

Service	An XQuery expression that defines the route to be taken by a message.
	To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
	In the XQuery/XSLT Expression editor, enter an Xquery expression, the result of which is similar to:
	`<ctx:route>`
	`<ctx:service isProxy='true'>{$service}</ctx:service>`
	`<ctx:operation>{$operation}</ctx:operation>`
	`</ctx:route>`
	Note: If a proxy service is being invoked, `isProxy` attribute should be set to true.
	– The service name is the fully qualified service name.
	– The operation element is optional

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Error Handler Node Properties

Use an error handler to specify what should happen if an error occurs in a specific location in the message flow.

All configuration for an error handler is in one or more stages contained by the error handler. See Stage Node Properties.

For-Each Action Properties

In a message flow, use the for-each action to iterate over a sequence of values and execute a block of actions.

In the Message Flow Editor, click a for-each action to display its properties in the Properties view. Use these properties pages to configure the selected for-each action. The pages are:

- For Each
- Comment
- Namespaces
- Variables

The For Each page has the following options:

Option	Description
For Each Variable	Enter the name of the variable on whose contents the for-each action(s) will be executed.
XPath	An XPath expression that specifies where in the structure of the containing context variable the variable specified in the **For Each Variable** field is located. To create or edit the XPath expression, click **<XPath>** (or the ***`XPath_fragment`***, if one is already defined) to display the XPath Expression Editor.
In Variable	The context variable containing the variable on whose contents the for-each action(s) will be executed.
Index Variable	A variable containing the current number of iterations in the loop.
Count Variable	A variable containing the total number of iterations.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

If-Then Action Properties

In a message flow, use an if-then action to perform an action or a set of actions conditionally, based on the Boolean result of an XQuery expression.

In the Message Flow Editor, click an if-then action to display its properties in the Properties view. Use this page to review and configure general properties of the if-then action. The pages are:

- Comment
- Namespaces
- Variables

An if-then action always contains an if condition plus zero or more else-if condition, where you define the conditions for the if-then action. An if-then action also always contains an else condition, which defines the default path when no other condition is met.

Click an **If:condition** icon or an **Else-If:condition** icon in an if-then action to display their properties in the Properties view. Use these properties pages to configure the selected condition. This **If** condition properties page has the following option.

If Action and Else-If Action Properties

If actions and else-if actions are always contained by an if-then action. The message follows the path associated with the first if action or else-id action that returns true.

Option	Description
Condition	A condition in an if action or an if-else action that contains one of the tests for following alternative paths in the flow. To add or edit a condition, 1. Click <**Condition**> (or ***`condition_fragment`***, if one is already defined) to display the Condition Editor. Define a condition to be evaluated in the if-then action. 2. When you finish editing the condition, add one or more action that will be executed when the condition evaluates to true. To add an action, drag it from the **Design Palette** to associate with the condition. In the route node, you can select only the routing, dynamic routing, or routing table actions. However, these actions can contain request and response actions. 3. To add an else-if condition, click **add condition**. Then add one or more actions to this else-if condition, as described above for the If condition. Continue with as many else-if conditions as your logic requires. Condition actions can be nested.

Else Action Properties

The else action does not have associate properties. When all if and else-if conditions return false, the message proceeds down the path associated with the else action.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Insert Action Properties

In a message flow, use an insert action to insert the result of an XQuery expression at an identified place relative to nodes selected by an XPath expression.

In the Message Flow Editor, click an insert action to display its properties in the Properties view. Use these properties pages to configure the selected insert action. The pages are:

- Insert
- Comment
- Namespaces
- Variables

The Insert page has the following options:

Option	Description
Expression	The XQuery expression used to create the data that will be inserted at a specified location in a named variable. To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Location	The location where the insert is performed, relative to the result of the XPath expression: Options are: • **Before**—as sibling before each element or attribute selected by the XPath expression • **After**—as sibling after each element or attribute selected by the XPath expression • **As first child of**—as first child of each element identified by the XPath expression. An error occurs if the result of the XPath returns attributes. • **As last child of**—as last child of each element identified by the XPath expression. An error occurs if the XPath returns attributes.

Option	Description
XPath	Valid configurations include those in which: • XQuery and XPath expressions both return elements. • The XQuery and XPath expressions both return attributes—in which case, the XQuery expression must return attributes. To create or edit the XPath expression, click **<XPath>** (or the ***`XPath_fragment`***, if one is already defined) to display the XPath Expression Editor.
Variable	The context variable whose contents are evaluated by the XPath variable defined above. Enter the name of the variable in the text field.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Java Callout Action Properties

In a message flow, use a Java callout action to invoke a Java method or an EJB business service from within the message flow.

In the Message Flow Editor, click a Java callout action to display its properties in the Properties view. Use these properties pages to configure the selected Java callout action. The pages are:

- Java Callout
- Comment
- Namespaces
- Variables

The Java Callout page has the following options:

Option	Description
Method	An external Java method to be called from the message flow. Click **Browse** to select a class and a static method from an archived resource. Once you have selected the class and method, a table appears on the **Java Callout Properties** page: The **Name** column lists all the method's arguments. The **Action** column provides an **<Expression>** or *`expression_fragment`* link to the XQuery/XSLT Expression Editor, where you can create an expression to retrieve a value for the argument. Data types for the arguments must be one of the following: • `Java.lang.String` • Primitive types, and their corresponding class types (e.g., `int` vs. `java.lang.Integer`) • `java.lang.BigDecimal`, and `java.lang.BigInteger` (these types are used in financial calculations where round-off errors or overflows are not tolerable) • only `org.apache.xbeans.XmlObject` and no typed xml beans. • `byte[]` • `java.lang.String[]` (INPUT ONLY) • `XmlObject [ ]` (INPUT ONLY) If the type of the input value you enter does not match the declared input argument type, Oracle Service Bus tries to automatically typecast input values to the declared type of the input argument. For example a string value of `"123"` will be converted to integer `123` if the declared type of the input argument is java primitive `int`.

Option	Description
Result Type	The variable to which the result is assigned. The label for the field indicates the data type of the result. If the result is a byte array (the only possible array returned), the binary-content XML element is returned.
Service Account	An optional Service Account, which can be specified if there is a security context for this Java method. Click **Browse** to select a service account. In the case of fixed and mapped service accounts, the userid/password from the service account is authenticated in the local system and the security context propagated to the Java callout. In the case of `passthru`, the security context is propagated to the Java callout. This context is the message level context if defined (with WS-Security). Otherwise, it is the transport level context.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Log Action Properties

In a message flow, use the log action to construct a message to be logged and to define a set of attributes with which it will be logged.

In the Message Flow Editor, click a log action to display its properties in the Properties view. Use these properties pages to configure the selected log action. The pages are:

- Log
- Comment
- Namespaces
- Variables

The Log page has the following options:

Option	Description
Expression	The message context to be logged through XQuery expressions on context variables. To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Annotation	Notes for this log action. These notes are logged along with the result of the previously defined expression.
Severity	The severity of the log message. Options are: **Debug** - While your application is under development, you might find it useful to create and use messages that provide verbose descriptions of low-level activity within the application. **Info** -Used for reporting normal operations; a low-level informational message. **Warning** - A suspicious operation or configuration has occurred but it might not affect normal operation. **Error** - A user error has occurred. The system or application can handle the error with no interruption and limited degradation of service.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Message Flow Properties- Comment

In the Message Flow Editor, click a node or an action to display its properties in the Properties view. For most nodes and actions, one of the properties pages displayed (as a tab) is the Comment page. Use this page to add an optional comment about the item selected in the message flow.

Message Flow Properties - Namespaces

In the Message Flow Editor, click a node or an action to display its properties in the Properties view. For most nodes and actions, one of the properties pages displayed (as a tab) is the Namespaces page. Use this page to see a list of defined namespaces or to create a new namespace.

Message Flow Properties - Variables

In the Message Flow Editor, click a node or an action to display its properties in the Properties view. For many nodes and actions, one of the properties pages displayed (as a tab) is the Variables page. Use this page to see a list of all the predefined and user-defined context variables. Click **Add** to define a new variable.

For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

MFL Transform Action Properties

In a message flow, use a MFL (Message Format Language) transform action to convert message content from XML to non-XML, or vice versa, in the message pipeline. An MFL is a specialized XML document used to describe the layout of binary data. It is a Oracle proprietary language used to define rules to transform formatted binary data into XML data, or vice versa.

In the Message Flow Editor, click a MFL transform action to display its properties in the Properties view. Use these properties pages to configure the selected MFL transform action. The pages are:

- MFL Transform
- Comment
- Namespaces
- Variables

The MFL Transform page has the following options:

Option	Description
Apply Transform	The type of transform to be applied. Select **XML to Non-XML** or **Non-XML to XML**, according to your requirement.
on <Expression>	The variable on which the MFL transformation action is to be performed. Click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor, where you can specify the variable. This input must be text or binary when transforming to XML, and must be XML when transforming to non-XML. Binary content in the message context is represented by the binary-content XML element. This XML should be the result of the Xquery expression when the input needs to be binary.
Resource	Select this option to perform an MFL transform action using a static MFL resource. Click **Browse** to select the resource.
Resource From	Select this option to specify an MFL resource that will perform the transform action. Click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor, where you can specify the MFL resource that will perform the action, in the format `project/folder/MFLresourcename`.
Variable	The name of the variable to which the result of this transform action is to be assigned. The result will be a binary-content XML element.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

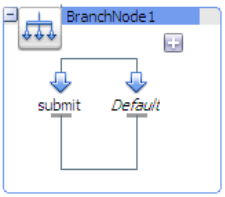
Operational Branch Node Properties

In a message flow, use an operational branch node to configure branching based on operations defined in a WSDL.

When message flows define WSDL-based proxy services, operation-specific processing is required. Instead of configuring a branching node based on operations manually, Oracle Service Bus provides a branching node that automatically branches based on operations. In other words, when you create an operational branch node in a message flow, you can quickly build your branching logic based on the operations defined in the WSDL, because the Oracle Service Bus plug-ins present those operations in the operational branch node configuration page.

A branch node allows processing to proceed along exactly one of several possible paths. Branching is driven by an XPath-based switch table. Each branch in the table specifies a condition (for example, `<500`) that is evaluated in order down the message flow against a single XPath expression (for example, `./ns: PurchaseOrder/ns:totalCost` on `$body`). Whichever condition is satisfied first determines which branch is followed. If no branch condition is satisfied, then the default branch is followed. A branch node may have several descendants in the message flow: one for each branch, including the default branch.

When you add an operational branch node to a message flow in the Message Flow Editor, the node contains an initial conditional branches based on the first operation defined in the WSDL, as shown below, plus a default branch:



Click the node itself (the **Operational Branch Node** icon or the bounding box connected to the icon) to display the Flow properties page in the Properties view. You can provide a name and a description for the node on that page.

Click an operational branch icon to display the properties for that operational branch on the Operational Branch page. That page contains a drop-down list of all the operations defined in the WSDL. You can select a different operation for that branch. You can also click the plus sign in the operational branch node, in the Message Flow Editor, to add another operation branch.

After you have added all the branches, add nodes and stages to them to define the processing for each branch.

Pipeline Pair Node Properties

In a message flow, use a pipeline pair node to define request and response processing.

Message flows can include zero or more pipeline pair nodes: request and response pipelines for the proxy service (or for the operations on the service), and error handler pipelines that can be defined for stages, pipelines, and proxy services. Pipelines can include one or more stages, which in turn include actions.

A pipeline pair always contains a request pipeline and a response pipeline. Add stages and actions to those pipelines, as needed.

In the Message Flow Editor, click a pipeline pair node to display its properties in the Properties view. Use this properties page to configure the selected pipeline pair node. The options are:

Option	Description
Name	Enter a name for the pipeline pair node.
Description	Enter a description for the pipeline pair node.

Publish Action Properties

In a message flow, use a publish action to identify a statically specified target service for a message and to configure how the message is packaged and sent to that service.

In the Message Flow Editor, click a publish action to display its properties in the Properties view. Use these properties pages to configure the selected publish action. The pages are:

- Publish
- Comment
- Namespaces
- Variables

The Publish page has the following options:

Option	Description
Service	The target service for the publish action. Click **Browse** to select a proxy service or business service from a list.
Invoking	The operation to be invoked on the target service. This option appears only if the selected service defines any operations. To configure how the message is packaged and sent to the service, in the **Request Actions** field, click **Add an Action**. Then select an action to associate with the service. You can add more than one action.
Use inbound operation for outbound	Select this option to make the outbound operation the same as the inbound operation. This option appears only if the selected service defines any operations.

Use the Comment page to add a comment, if desired:

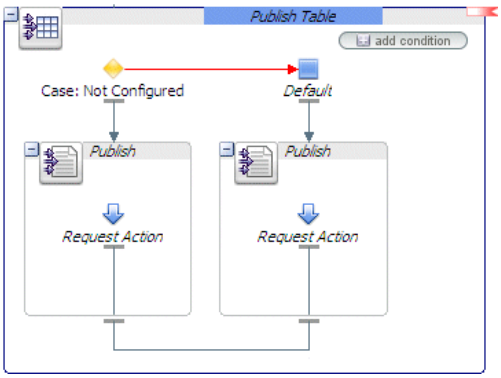
Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Publish Table Action Properties

In a message flow, use a publish table action to publish a message to zero or more statically specified services. Switch-style condition logic is used to determine at run time which services will be used for the publish. A publish table action contains one or more case actions to define conditions for following alternative paths (by default, via publish actions), as well as a default path to a default publish action. To fully define a publish table action, you must configure the publish table action, its case actions, and its publish actions (and any other actions you may add).

When you add a publish table action to a message flow in the Message Flow Editor, the publish table action contains a case action with a path to a publish action plus a default case action with a path to a default publish action, as shown below. Click **add condition** to add another condition to the publish table.



Publish Table Properties

In the Message Flow Editor, click the publish table action itself (the **Publish Table** icon or the bounding box connected to the icon) to display the Publish Tables properties page in the Properties view. Use these properties pages to configure the selected publish table action. The pages are:

- Publish Table
- Comment
- Namespaces
- Variables

The Publish Table properties page has the following option:

Option	Description
Expression	An XQuery expression, which at run time returns the value upon which the routing decision will be made. To create or edit an XQuery expression, click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Case Action Properties

In the Message Flow Editor, click a case action to display its properties in the Properties view. Use this properties page to configure the selected case action, as described below:

Option	Description
• Operator • Value	Select a comparison operator from the list. Then enter a value against which the value returned from the XQuery expression defined for the publish table action will be evaluate

Publish Action Properties

See Publish Action Properties.

Raise Error Action Properties

In a message flow, use the raise error action to raise an exception with a specified error code (a string) and description.

In the Message Flow Editor, click a raise error action to display its properties in the Properties view. Use these properties pages to configure the selected raise error action. The pages are:

- Raise Error
- Comment
- Namespaces
- Variables

The Raise Error page has the following options:

Option	Description
Code	The error code to be raised.
Message	Description of the error code

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Transactions

If a service is transactional, a triggered Raise Error action aborts the transaction in the request (asynchronous) or in either the request or response (synchronous). For example, you may introspect messages and determine conditions under which a Raise Error action should occur even if no SOAP fault occurs, and Raise Error causes the transaction to be aborted.

Rename Action Properties

In a message flow, use the rename action to rename elements selected by an XPath expression without modifying the contents of the element.

In the Message Flow Editor, click a rename action to display its properties in the Properties view. Use these properties pages to configure the selected rename action. The pages are:

- Rename
- Comment
- Namespaces
- Variables

The Rename page has the following options:

Option	Description
XPath	An XPath expression used to specify the data (in the named variable) that will be renamed. To create or edit the XPath expression, click **<XPath>** (or the ***`XPath_fragment`***, if one is already defined) to display the XPath Expression Editor.
In Variable	The context variable that holds the element you want to rename. Enter the name of the variable in this field.

Option	Description
Localname	A local name to use to rename the selected elements. Enter the local name in this field.
Namespace	A namespace to use when renaming the selected elements. Enter the namespace in this field.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Replace Action Properties

In a message flow, use a replace action to replace a node or the contents of a node specified by an XPath expression. The node or its contents are replaced with the value returned by an XQuery expression.

A replace action can be used to replace simple values, elements and even attributes. An XQuery expression that returns nothing is equivalent to deleting the identified nodes or making them empty, depending upon whether the action is replacing entire nodes or just node contents.

In the Message Flow Editor, click a replace action to display its properties in the Properties view. Use these properties pages to configure the selected replace action. The pages are:

- Replace
- Comment
- Namespaces
- Variables

The Replace page has the following options:

Option	Description
XPath	The XPath expression used to specify the data (in the named variable) that will be replaced. To create or edit the XPath expression, click **<XPath>** (or the ***XPath_fragment***, if one is already defined) to display the XPath Expression Editor.
Variable	Enter a context variable.
Expression	The XQuery expression used to create the data that replaces the data specified by the XPath in the named variable. To create or edit an expression, click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Replace entire node or Replace node contents	When you finish editing the XQuery expression, select one of the options: • **Replace entire node**—to specify that the nodes selected by the XPath expression you defined are replaced along with all of its contents • **Replace node contents**—to specify that the node is not replaced; only the contents are replaced. **Note:** Selecting the **Replace node contents** option and leaving the **XPath** field blank is more efficient than selecting the **Replace entire node** option and setting the XPath to `./*`

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Reply Action Properties

In a message flow, use the reply action to specify that an immediate reply be sent to the invoker.

The reply action can be used in the request, response or error pipeline. You can configure it to result in a reply with success or failure. In the case of reply with failure where the inbound transport is HTTP, the reply action specifies that an immediate reply is sent to the invoker.

In the Message Flow Editor, click a reply action to display its properties in the Properties view. Use these properties pages to configure the selected reply action. The pages are:

- Assign
- Comment
- Namespaces
- Variables

The Reply page has the following options:

Option	Description
With Success or With Failure	Select **With Success** to reply that the message was successful, or select **With Failure** to reply that the message has a fault.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Report Action Properties

In a message flow, use the report action to enable message reporting for a proxy service.

In the Message Flow Editor, click a report action to display its properties in the Properties view. Use these properties pages to configure the selected report action. The pages are:

- Report
- Comment
- Namespaces
- Variables

The Report page has the following option:

Option	Description
Expression	The XQuery expression used to create the data that will be reported. To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor.
Search Keys	When you finish editing the XQuery expression, click **Add a Key** to add one or more key value pairs to be used to extract key identifiers from any message context variable or message payload. (The rest of the message is ignored.) The keys are a convenient way to identify a message. 1. In the **Key Name: Name** field enter a name for the key. 2. In the **Key Value** column, click **<XPath>** to create the XPath expression in the XPath Expression Editor. 3. In the **In variable** field, enter the name of the variable on which the expression will be executed.

For example, consider a report action configured on an error handler in a stage. The action reports the contents of the fault context variable in the event of an error. The report action is configured as follows:

- Key name = `errorCode`
- Key value = `./ctx:errorCode` in variable `fault`

Each time this action is executed at run time, a message is reported via the Reporting Data Stream. The following table shows the results after the report action is executed twice.

Report Index	DB TimeStamp	Inbound Service	Error Code
errorCode=BEA-382505	04/26/07 9:45 AM	MortgageBroker/ProxySvcs/loanGateway3	BEA-382505
errorCode=BEA-382505	04/26/07 9:45 AM		BEA-382505

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Resume Action Properties

In a message flow, use the resume action to resume message flow after an error is handled by an error handler. This action has no parameters and can only be used in error pipelines.

In the Message Flow Editor, click a resume action to display its properties in the Properties view. Use these properties pages to configure the selected resume action. The pages are:

- Comment
- Namespaces
- Variables

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Route Node Properties

In a message flow, use the route node to handle request and response dispatching of messages to and from business services. No other nodes can follow a route node.

In the Message Flow Editor, click a route node to display its properties in the Properties view. Use these properties pages to configure the selected route node. The pages are:

- Route
- Namespaces
- Variables

The Route page has the following option:

Option	Description
Name	Enter a name for the route node.
Description	Enter a description for the route node.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Routing Action Properties

In a message flow, use a routing action to identify a target service for the message and configure how the message is routed to that service.

This is a terminal action, which means you cannot add another action after this one. However, this action can contain request and response actions.

In the Message Flow Editor, click a routing action to display its properties in the Properties view. Use these properties pages to configure the selected routing action. The pages are:

- Routing
- Comment
- Namespaces
- Variables

The Routing page has the following option:

Option	Description
Service	The target service for the routing action. Click **Browse** to select a proxy service or business service from a list.
Invoking	The operation to be invoked on the target service. This option appears only if the selected service defines any operations. To configure how the message is packaged and sent to the service, in the **Request Actions** field, click **Add an Action**. Then select an action to associate with the service. You can add more than one action.
Use inbound operation for outbound	Select this option to make the outbound operation the same as the inbound operation. This option appears only if the selected service defines any operations.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Routing Options Action Properties

In a message flow, use a routing options action to modify any or all of the following properties in the outbound request: URI, Quality of Service, Mode, Retry parameters, Message Priority.

In the Message Flow Editor, click a routing options action to display its properties in the Properties view. Use these properties pages to configure the selected routing options action. The pages are:

- Routing Options
- Comment
- Namespaces
- Variables

The **Routing Options** page has the following option:

Option	Description
URI	Select this option to set the URI for the outbound message: To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor. Enter an expression that returns a URI. This overrides the URI for the invoked service.
QoS	Select this option to set the Quality of Service element: Select a QoS option from the drop-down list. This overrides the default.
Mode	Select this option to set the mode to Request or Request-Response. Select the mode from the drop-down list. **Note:** This is normally already automatically set, based on the interface of the service invoked. However, in some cases like Any Soap or Any XML services, this is not so.

Option	Description
Retry Interval	Select this option to set the retry interval. Specify the number of seconds between retries. This overrides the default configured with the invoked service.
Retry Count	Select this option to set the retry count. Specify the number of retries the system must attempt before discontinuing the action. This overrides the default configured with the invoked service.
Priority	Select this option to set the message priority: To create or edit an expression, click **<Expression>** (or the ***`expression_fragment`***, if one is already defined) to display the XQuery/XSLT Expression Editor. Enter an expression that returns a positive integer.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

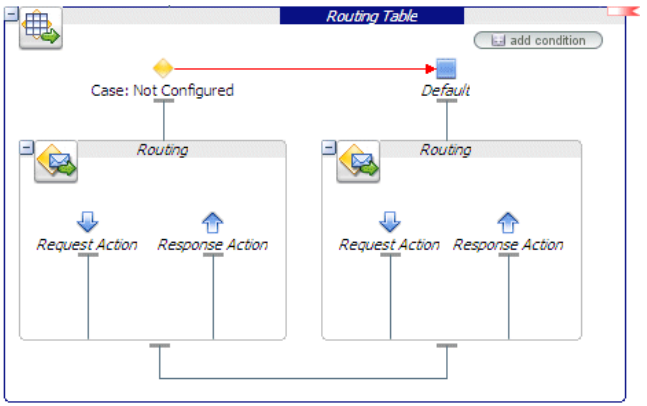
Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Routing Table Action Properties

In a message flow, use a routing table to select different routes based upon the results of a single XQuery expression. A routing table action contains a set of routes wrapped in a switch-style condition table.

This is a terminal action, which means you cannot add another action after this one. However, this action can contain request and response actions.

When you add a routing table action to a message flow in the Message Flow Editor, the routing table action contains a case action with a path to a routing action plus a default case action with a path to a default routing action, as shown below. Click **add condition** to add another condition to the routing table.



Routing Table Properties

In the Message Flow Editor, click the publish table action itself (the **Publish Table** icon or the bounding box connected to the icon) to display the Publish Tables properties page in the Properties view. Use these properties pages to configure the selected publish table action. The pages are:

- Routing Table
- Comment
- Namespaces
- Variables

The Routing Table properties page has the following option:

Option	Description
Expression	An XQuery expression, which at run time returns the value upon which the routing decision will be made. To create or edit an XQuery expression, click **<Expression>** (or the ***expression_fragment***, if one is already defined) to display the XQuery/XSLT Expression Editor.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Case Action Properties

In the Message Flow Editor, click a case action to display its properties in the Properties view. Use this properties page to configure the selected case action, as described below:

Option	Description
• Operator • Value	Select a comparison operator from the list. Then enter a value against which the value returned from the XQuery expression defined for the routing table action will be evaluate

Routing Action Properties

See "Routing Action Properties" on page 3-84.

Service Callout Action Properties

In a message flow, use a service callout action to configure a synchronous (blocking) callout to an Oracle Service Bus-registered proxy or business service.

In the Message Flow Editor, click a service callout action to display its properties in the Properties view. Use these properties pages to configure the selected service callout action. The pages are:

- Service Callout
- Comment
- Namespaces
- Variables

The Service Callout page has the following options:

Option	Description
Service	The target service for the service callout action. Click **Browse** to select a proxy service or business service from a list.
Invoking	The operation to be invoked on the target service. This option appears only if the selected service is WSDL-based and has operations that can be invoked on the service.
Configure Soap Body or Configure Payload Document	Specify how you want to configure the request and response messages by selecting one of the following options: • Select **Configure SOAP Body** to configure the SOAP Body. Selecting this option allows you to use `$body` directly. This option supports SOAP-RPC encoded, which is not supported when configuring payload parameters or document. • Select **Configure Payload Parameters** or **Configure Payload Document** to configure the payload.

Subsequent configuration options depend on the kind of service you selected in step 3 and on the kind of configuration options you chose for that service in step 5. Table 3-1 shows the options available for each service type.

Selected Service Type	"Configure SOAP Body" Options	"Configure Payload Parameters" Options or "Configure Payload Document" Options
• SOAP RPC	• SOAP Request Body and SOAP Response Body • SOAP Request Header and SOAP Response Header (optional)	• Request Parameters and Response Parameters • SOAP Request Header and SOAP Response Header (optional)

Selected Service Type	"Configure SOAP Body" Options	"Configure Payload Parameters" Options or "Configure Payload Document" Options
• SOAP Document • Any SOAP	• SOAP Request Body and SOAP Response Body • SOAP Request Header and SOAP Response Header (optional)	• Request Document and Response Document • SOAP Request Header and SOAP Response Header (optional)
• XML • Any XML • Messaging	• SOAP Request Body and SOAP Response Body	• Request Document and Response Document

The following table provides instructions for each of the options listed in the table above.

For These Options...	Follow These Steps...
SOAP Request Body and SOAP Response Body	To configure these options, • In the **SOAP Request Body** field, enter the name of a variable to hold the XML of the SOAP Body element for the callout request. • In the **SOAP Response Body** field, enter the name of a variable to which the XML of the SOAP Body element on the response will be bound.
SOAP Request Header and SOAP Response Header	To configure these options, • In the **SOAP Request Header** field, enter the name of a variable to hold the XML of the SOAP Header element for the callout request You must wrap the input document for the SOAP Request Header with `<soap-env:Header>...</soap-env:Header>`. • In the **SOAP Response Header** field, enter the name of a variable to which the XML of the SOAP Headers on the response, if any, will be bound.

For These Options...	Follow These Steps...
Request Parameters and Response Parameters	To configure options, • In the **Request Parameters** fields, enter names for the variables that will be evaluated at run time to provide values for the request parameters. You must provide only the core payload documents in the input variable—the SOAP package is created for you by Oracle Service Bus. In other words, do not wrap the input document with `<soap-env:Body>...</soap-env:Body>`. For example, when creating a body input variable that is used for this request parameter, you would define that variable's contents using the XPath statement `body/*` (to remove the wrapper `soap-env:Body`), not `$body` (which results in keeping the `soap-env:Body` wrapper). • In the **Response Parameters** fields, enter the names of the variables to which the responses will be assigned at run time.

For These Options...	Follow These Steps...
Request Document and Response Document	To configure these options, • In the **Request Document Variable** field, enter the name of a variable to assign a request document to. For *SOAP Document-type* services, the variable is evaluated at runtime to form the body of the SOAP message sent to the service. For *Any XML* services, the variable is evaluated at runtime to form the body of the XML message sent to the service. For SOAP Document-type services and for Any XML services, you provide only the core payload documents in the input variable—the SOAP package is created for you by Oracle Service Bus. In other words, do not wrap the input document with `<soap-env:Body>...</soap-env:Body>`. For example, when creating a body input variable that is used for this request parameter, you would define that variable's contents using the XPath statement `body/*` (to remove the wrapper `soap-env:Body`), not `$body` (which results in keeping the soap-env:Body wrapper). For *Messaging* services, the variable is evaluated to form the body of the message, based on the type of data expected by the service. The following restrictions apply to variables used with Messaging services: – For services that expect binary data, the variables must have a `ctx:binary-content` element. – For services that expect MFL data, the variable must have the XML equivalent. – For services that expect text data, the variable is a string. • In the **Response Document Variable** field, enter the name of the variable to which a response document will be assigned at run time.

Optionally, add one or more transport header actions. For more information about transport header actions, see "Transport Headers Action Properties" on page 3-94

Note: In addition to the transport headers you specify, headers are added by the Oracle Service Bus binding layer.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Skip Action Properties

In a message flow, use the skip action to specify that at run time, the execution of the current stage is skipped and the processing proceeds to the next stage in the message flow. This action has no parameters and can be used in the request, response or error pipelines.

In the Message Flow Editor, click a skip action to display its properties in the Properties view. Use these properties pages to configure the selected skip action. The pages are:

- Comment
- Namespaces
- Variables

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Stage Node Properties

In a message flow, use a stage node as a container for actions in a message flow. You can string multiple stages together, to compartmentalize processing logic.

In the Message Flow Editor, click a stage node to display its properties in the Properties view. Use these properties pages to configure the selected stage node. The pages are:

- Stage
- Namespaces
- Variables

The Stage page has the following options:

Option	Description
Name	Enter a name for the stage node.
Description	Enter a description of the stage node.

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Transport Headers Action Properties

In a message flow, use a transport header action to set header values in messages.

In the Message Flow Editor, click a transport headers action to display its properties in the Properties view. Use these properties pages to configure the selected transport headers action. The pages are:

- Transport Headers
- Comment
- Namespaces
- Variables

The Transport Header page has the following options:

Option	Description

Direction	4. From the **Set Transport Headers for** list, select one of the following, to specify to the run time which of the message context locations are to be modified:
	– **Outbound Request** - Select this option to set header values for outbound requests (the messages sent out by a proxy service in route, publish, or service callout actions). This header element is located in the message context as follows:
	`$outbound/ctx:transport/ctx:request/tp:headers`
	– **Inbound Response** - Select this option to set header values for inbound responses (the response messages a proxy service sends back to clients). This header element is located in the message context as follows:
	`$inbound/ctx:transport/ctx:response/tp:headers`

Pass All Headers	Select this option to pass all headers through from the inbound message to the outbound message or vice versa. Every header in the source set of headers will be copied to the target header set, overwriting any existing values in the target header set.
Headers	Click **Add Header** to add a header to the **Headers** table. Then configure each header as described in the following rows. You can add as many headers as necessary to this table. You do not have to order the headers in the table, because the run time declares namespaces and places header elements in their proper order when generating the corresponding XML.

Option	Description
Headers: Name:	Specify a header by doing either of the following: • From the drop-down list in the **Name** column, select a header name. The list contains all of the predefined header names for the target transport (for example, **Content-Type** for HTTP transports, **JMSCorrelationID** for JMS transports, etc.). • Enter a header name in the **Other** field. If that header name is not one of the predefined headers for this service's transport, it becomes a user-header, as defined by the transport specification.

Option	Description
Headers: Action	Select one of the options in this column to specify how to set the header's value: • **Set Header to Expression** Selecting this option allows you to use an XQuery or XSLT expression to set the value of the header. The expression can be simple (for example, "text/xml") or a complex XQuery or XSLT expression. Because the Oracle Service Bus transport layer defines the XML representation of all headers as string values, the result of any expression is converted to a string before the header value is set. Expressions that return nothing result in the header value being set to the empty string. You cannot delete a header using an expression. **Warning**: Not all of the header settings you can specify in this action are honored at run time. • **Delete Header** Specifies that the header is removed from the request or response metadata. **Copy Header from Inbound Request** (if you are setting transport headers for the Outbound Request) or **Copy Header from Outbound Response** (if you are setting transport headers for the Inbound Response) Specifies that this header is copied directly from the corresponding header of the same name from the inbound message to the outbound message and vice versa. For example, if you want to set the SOAPAction header for an outbound request, selecting **Copy Header from Inbound Request** causes the run time to copy the value from the SOAPAction request header of `$inbound`. In the case of inbound response headers, the source of the header to copy is the response headers of `$outbound`. If the **Copy Header...** option is selected for a header that does not exist in the source, this option is ignored and no action is performed on the target for this header.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Validate Action Properties

In a message flow, use a validate action to validate elements selected by an XPath expression against an XML schema element or a WSDL resource. You can validate global elements only; Oracle Service Bus does not support validation against local elements.

In the Message Flow Editor, click a validate action to display its properties in the Properties view. Use these properties pages to configure the selected validate action. The pages are:

- Validate
- Comment
- Namespaces
- Variables

The Validate page has the following options:

Option	Description
XPath	An XPath expression that specifies the elements to be validated. To create or edit the XPath expression, click **<XPath>** (or the **XPath_fragment**, if one is already defined) to display the XPath Expression Editor.
In Variable	The name of the variable to hold the element to be validated. Enter the name of the variable, or, if Content Assist is on, press **Ctrl-Space** to display a list of declared variables.
Against Resource	An XML schema element or a WSDL resource against which the elements selected by the XPath expression (in the **XPath** field described above) are validated.
Save Variable or Raise Error	To save the result of this validation (a boolean result), select **Save Variable** and enter the name of the variable in which you want to save the result. Alternatively, to raise an error if the element fails validation against the WSDL or XML schema element, select **Raise Error**.

Use the Comment page to add a comment, if desired:

Use the Namespaces page to see a list of defined namespaces or to create a new one.

Use the Variables page to see a list of defined context variables or to create a new one. For more information about context variables, see Message Context in the *Oracle Service Bus User Guide*.

Modify JAR Dependencies dialog

In Oracle Service Bus, JAR resources can contain classes that depend on other classes in different JARs. Use the Modify JAR Dependencies page to add or remove referenced JAR resources. Use the **Up** and **Down** buttons to reorder the hierarchy (order of precedence) of referenced JARs. For more information, see JARs in *Using the Oracle Service Bus Console*.

Option	Description
Available JARs	The project name and the name of the folder, if applicable, in which the JAR resources reside. This column shows all available JARs in the current workspace.
JAR references	The name and path of the referenced JAR resources. This column shows all the JARs currently configured as dependencies.

SMTP Servers

The following editor and wizard are provided for working with SMTP servers:

- Edit SMTP Server page
- New SMTP Server Resource wizard

Edit SMTP Server page

Use this page to view the details of an SMTP server and edit the configuration, if required. For descriptions of the fields, see the New SMTP Server Resource wizard. For more information, see Editing SMTP Servers in *Using the Oracle Service Bus Console*.

New SMTP Server Resource wizard

Use this page to configure a new SMTP server resource. For more information, see Adding SMTP Servers in *Using the Oracle Service Bus Console*.

Option	Description
Description	Enter a description for this SMTP server resource.
Server URL	Enter the URL that points to the SMTP server. This is a required field.
Port Number	Enter a port number for the SMTP server (the default port is 25). This is a required field.
User Name	If access to the target SMTP server requires a user name and password, enter a user name in the User Name field, and the associated password in the **Password** and **Confirm Password** fields. These fields are optional, and required only if the SMTP server is secured.
Password	Enter the associated password.
Confirm Password	Enter the same password you entered for the **Password** field.

UDDI Registry Configuration page

Use this page to configure a UDDI registry resource. This page is displayed in the New UDDI Registry Resource wizard and in the UDDI Configuration editor.

For more information, see UDDI in *Using the Oracle Service Bus Console* and UDDI in the *Oracle Service Bus User Guide*.

Option	Description
Description	Enter a description of the registry.
Inquiry URL	The URL of the Inquiry API endpoint used for locating and importing services. Enter an inquiry URL in the format: `http://host:port/APPLICATION_SERVER_CONTEXT/uddi/inquiry`
Publish URL	The URL of the Publish API endpoint used for publishing services. Enter a publish URL in the format: `http://host:port/APPLICATION_SERVER_CONTEXT/uddi/publishing`

Option	Description
Security URL	The URL of the Security API endpoint used for getting an authentication token so that you can publish to the registry. Enter a security URL in the format: `http://host:port/APPLICATION_SERVER_CONTEXT/uddi/security`
Subscription URL	The URL of the Subscription API endpoint used for subscribing to registry changes, creating a registry subscription, and listening for changes to imported services. Enter a subscription URL in the format: `http://host:port/APPLICATION_SERVER_CONTEXT/uddi/subscription` By default, the value for *`APPLICATION_SERVER_CONTEXT`* in the installer is `registry`.
User Name	Enter the user name to log into the registry console.
Password (Confirm Password)	Enter the password to log into the registry console.
Validate	Click **Validate** to validate that a connection can be made to the configured registry.

Outline view - Oracle Service Bus

This view displays a hierarchical view, or outline, of a structured file that is currently open in the editor area. The contents of this view are editor-specific.

In Oracle Service Bus, when the proxy service is editor is open and the Message Flow Editor is selected, the Outline view displays a hierarchical view of the nodes and actions in the message flow. When the Message Flow Editor is displayed, you can switch between the hierarchical view and a thumbnail view of the service, as described in the table below:

Button	Description
	Click this button to display a hierarchical view of the contents of the current editor.
	Click this button to display a graphical thumbnail view of the contents of the Message Flow Editor. When the entire message flow does not fit in the editor, a blue mask appears in the outline view to show what portion of the flow is visible in the editor. To display a different portion of the flow, you can drag the mask to the portion of the flow you want to display in the editor.

Resource Management

The following are provided for managing resources:

- Preferences dialog - Oracle Service Bus - Type Associations page
- References view
- Select Clone Target dialog
- Select Clone Target dialog

Preferences dialog - Oracle Service Bus - Type Associations page

Use this page to associate one or more file extensions with an Oracle Service Bus resource type.

Option	Description
Resource Types	This pane lists all the available types of Oracle Service Bus resources. Select a type to display its assigned file extension associated with the type.

References view

Use this page to view resource dependencies. This view shows your current selection whether it is in the Project Explorer or the active editor.

Option	Description
Referenced By	A list of the resources outside of this project, folder, or resource that are dependent on resources inside this project or folder, or this resource.
References	A list of the resources outside of this project, folder, or resource on which resources inside this project or folder, or this resource depends.

Select Clone Target dialog

Use this dialog to specify where to clone a project, as a new project or as a folder under a different project.

Option	Description
Name	Enter a name for the new (cloned) project.
As project or As folder in location	Select **As project** to clone the project as another project, that is, as a peer to the other projects. Select **As folder in location** to convert a project (and its contents) into a folder under a project. When you select this option, you must also select the new location.

Select a Resource dialog

Use this dialog to select a resource, appropriate to the context, that has been created in or imported into the Oracle Service Bus plug-ins. When you click **Browse** from an editor or a wizard, this dialog displays the available resources that are appropriate for the context. The dialog has different names, depending on the type of resource displayed, for example, **Select a WSDL**, **Select a MFL**, **Select an XML Schema**, etc. In some cases, the dialog displays a hierarchical list of items contained in each resource. For example, the **Select an XML Schema** dialog lists elements and types defined in the schemas. The **Select a WSDL** dialog displays ports and bindings defined in the WSDLs.

In some cases, for example, the **Select a WSDL** dialog, you can click the **Consume** button to open the **Service Consumption** dialog, where you can consume services from the following resource types: Oracle Enterprise Repository, file system, UDDI, URI, workspace: consume a service residing in the current workspace.

New Service Key Provider Resource wizard

Use this page to configure a service key provider resource. For more information, see Service Key Providers in *Using the Oracle Service Bus Console*.

Option	Description
Description	Enter a description for the service key provider.
Encryption Key	1. Next to **Encryption Key**, select the **Key** check box. 2. Enter a key alias that maps to an X.509 certificate and that supports encryption, or click **Browse** to display the key aliases from the key store that your realm's PKI credential mapper is using. 3. Enter the password that you use to secure access to the key store. (You set the password when you create the key store.) When you associate this service key provider with a proxy service, Oracle Service Bus embeds the X.509 certificate into the proxy service's WSDL. The proxy service then uses this certificate to encrypt the messages that it sends to its endpoint. The proxy service uses the private key in the PKI credential to decrypt the messages that the endpoint returns.
Digital Signature Key	1. Next to **Digital Signature Key**, select the **Key** check box. 2. Enter a key alias, or click **Browse** to display the key aliases from the key store that your realm's PKI credential mapper is using. 3. Enter the password that you use to secure access to the key store. (You set the password when you create the key store.)
SSL Client Authentication Key	1. Next to **SSL Client Authentication Key**, select the **Key** check box. 2. Enter a key alias, or click **Browse** to display the key aliases from the key store that your realm's PKI credential mapper is using. 3. Enter the password that you use to secure access to the key store. (You set the password when you create the key store.)

New WS-Policy wizard

Web Services Policy Framework (WS-policy) is an extensible XML-based framework that extends the configuration of a Web Service with domain specific security assertions and specifies the security requirements, expectations, and capabilities of the Web Service. In Oracle Service Bus, one of the primary uses of WS-policy is configuring message-level security in proxy

services and business services. For more information, see Custom WS-Policies in *Using the Oracle Service Bus Console*.

Use this page to select a name and location for a new WS-policy resource.

Option	Description
Enter or select the parent folder	Enter or select the name of the project or folder in which to locate a new WS-policy resource.
File Name	Enter a name for this WS-policy.

Service Accounts

The following are provided for working with service accounts:

- New Service Account Resource wizard
- Service Account editor - General Configuration page
- Service Account editor - Static User Configuration page
- Service Account editor - User Mappings Configuration page

New Service Account Resource wizard

Use this wizard to create a new service account resource, as described in the following table.

Option	Description
Description	Enter a description for the service account.
Resource Type	• To create a service account that provides the user names and passwords that it receives from incoming client requests, select **Pass Through** and click **Finish**. • To create a service account that provides a user name and password that you save with the service account configuration, select **Static**. Continue as described in Service Account editor - Static User Configuration page. • To create a service account that maps the user name from one or more authenticated clients to user names and passwords that you specify, select **Mapping**. Continue as described in Service Account editor - User Mappings Configuration page.

For more information, see Service Accounts in *Using the Oracle Service Bus Console*.

Service Account editor - General Configuration page

Use this page to configure a service account resource, as described in the following table.

Option	Description
Description	Enter a description for the service account.
Resource Type	Leave the resource type as is, or select a different resource type: • A **Pass Through** type provides the user names and passwords that it receives from incoming client requests. • A **Static** type provides a user name and password that you save with the service account configuration. Configure this type on the Service Account editor - Static User Configuration page. • A **Mapping** type maps the user name from one or more authenticated clients to user names and passwords that you specify. Configure this type on the Service Account editor - User Mappings Configuration page.

For more information, see Service Accounts in *Using the Oracle Service Bus Console*.

Service Account editor - Static User Configuration page

Use this to page to save a user name and password with the service account configuration. The service account encodes this user name and password in the outbound request. For more information, see Service Accounts in *Using the Oracle Service Bus Console*.

Option	Description
User Name	Enter a user name for this service account.
Password	Enter the associated password.
Confirm Password	Enter the same password you entered for the **Password** field.

Service Account editor - User Mappings Configuration page

Use this to page to create a service account that maps the user name from one or more clients to user names and passwords that you specify. For more information, see Service Accounts in *Using the Oracle Service Bus Console*.

Option	Description
Remote Users	In the **Remote User Name** and **Remote Password** fields, enter the user name and password that you want to send in outbound requests. (Optional) Add additional remote users in the Remote Users table.

Option	Description
Local User Mappings	To map *authorized* clients to remote user names and passwords, in the **Local User Name** field, enter the name that identifies a client that has been authenticated on its inbound request. If you have not already added this user in the Security Configuration module of the Oracle Service Bus Console, do so before you use this mapping in a runtime environment. See Adding Users in *Using the Oracle Service Bus Console*. Oracle Service Bus lets you create a mapping for a non-existent local user, but the mapping will never match an authenticated user and will never be used. From the **Remote User Name** list, select the user name that you want to send in outbound requests for the authenticated user you specified in the **Local User Name** field.
Map Anonymous Requests	To map *anonymous* clients to remote user names, select the **Map Anonymous Requests** check box. From the **Select Remote User** list, select the user name that you want to send in outbound requests for all anonymous users.

Expression Editors

The following editors help you to write expressions for use in services.

- XQuery/XSLT Expression Editor
- XPath Expression Editor
- Condition Editor

XQuery/XSLT Expression Editor

Use the XQuery/XSLT Expression editor to create variable structures, define user namespaces, use predefined message context variables to build inline XQuery expressions, build inline XQuery expressions manually, and select XQuery or XSLT resources for execution as inline XQueries.

For more information about using XQueries, see Working with Variable Structures in the *Oracle Service Bus User Guide*

The XQuery/XSLT Expression editor contains two panels, each containing three tabs.

Use the panel on the left to write or construct the expression. The panel contains these pages (tabs):

- Expression page
- XQuery Resource page
- XSLT Resource page
- Dynamic XQuery page

Use the tabs in the panel on the right to manage the variables, namespaces, and XQuery functions you can use to construct an XQuery or XSLT. The panel contains these pages (tabs):

- Variable Structures page
- Namespace Definitions page
- XQuery Functions page

XPath Expression Editor

Use the XPath Expression Editor to create an XPath expression, which is then inserted into the location from which you launched the editor.

The XPath Expression Editor contains these pages:

- Expression page
- Variable Structures page
- Namespace Definitions page
- XQuery Functions page

You can write an expression directly in the **Expression** text field, or you can drag variables from the Variable Structure page and drag XQuery functions from the XQuery Functions page to construct a valid XPath expression.

When you are finished constructing the expression, you can click **Test** to test the expression on a running server. Or click **OK** to insert the expression without testing it.

Condition Editor

Use the Condition Editor to create an inline XQuery condition.

Use the panel on the left to write or construct the condition. The panel contains these tabs, each of which is described below:

- Expression page
- Condition Builder page

Use the tabs in the panel on the right to manage the variables, namespaces, and XQuery functions you can use to construct a condition. The panel contains the following tabs. They are the same as the tabs in the right-hand panel of the XQuery/XSLT Expression Editor, and the following links display the help for those tabs in the XQuery/XSLT Expression Editor.

- Variable Structures page
- Namespace Definitions page
- XQuery Functions page

Condition Builder page

Use the Condition Builder page to build an inline condition. This page appears only in the Condition Editor.

Build an expression in the Condition Builder as follows:

1. Select **Comparison Expression** to build a comparison expression or select **Unary Expression** to build a unary expression.
2. If you chose **Comparison Expression**, do the following:
   a. In the **Operand** field, enter the name of the operand you want to compare to a value. Alternatively, drag an item from the **Variable Structures** tab on the right to the **Operand** field.
   b. In the **Value** field, enter the value against which to compare the operand. Alternatively, drag item from the **Variable Structures** tab on the right to the **Value** field.
   c. Select a comparison operator from the **Operator** list.
   d. Go to step 4.
3. If you chose **Unary Expression**, do the following:
   a. Select or deselect the **Not** check box to indicate whether to evaluate the expression as true or not true, that is, to specify whether the expression should be enclosed by not().

b. Enter an expression in the text field, or construct it by dragging items from the **Variable Structures** tab on the right.

4. Do either of the following:
   - Click **Add** to add a new statement to the expression in the **Condition Expression** field.
   - Select a statement in the **Condition Expression** field, then click **Update** to modify the statement.
5. Click **Test** to test the expression in a running server, or click **OK** to insert the condition in the message flow.

Expression page

This page appears on the XQuery/XSLT Expression Editor, the XPath Expression Editor, and the Condition Editor.

The Expression page contains a text field where you can build expressions by typing directly into the field or by dragging items from the Variable Structures page or the XQuery Functions page on the right side of the editor.

Click **Test** to test the expression in an Oracle Service Bus domain on a running server. WebLogic Server is packaged with Oracle Service Bus, so an installation of Oracle Service Bus includes an embedded server you can use for testing purposes.

XQuery Resource page

This page appears in the XQuery/XSLT Expression Editor only.

In the message flow of a proxy service, you can assign XQuery expressions to message context variables, assign if-then-else actions based on the Boolean result of an XQuery expression, insert the result of an XQuery expression at an identified place relative to an XPath expression, specify the message context that you want to log through XQuery expressions on context variables, and so on.

Use the XQuery Resource page to configure an XQuery transformation to be executed in the message flow of a proxy service.

Field	Description
XQuery	The XQuery resource to be executed. Click **Browse** to find and open an XQuery resource that has been registered in Oracle Service Bus.
Bind Variables	When you select a resource, each input parameter of the transformation is displayed. Each label corresponds to the name of a parameter, and each text box is for defining an XQuery expression to be mapped to the parameter. You must define a mapping for each parameter. Enter the expression directly, or drag variables and structures from the right panel.

XSLT Resource page

This page appears in the XQuery/XSLT Expression Editor only.

Use the **XSLT Resource** tab to configure an XSLT transformation to be executed in the message flow of a proxy service.

Field	Description
XSLT	The XSLT resource to execute. Click **Browse** to find and open a resource registered with Oracle Service Bus.

Field	Description
Input Document	An XQuery expression for the input document to the transformation, for example `$body`.

Bind Variables	A label and a corresponding text box is displayed for each input parameter of the transformation. Each label corresponds to the name of a parameter, and each text box is for defining an XQuery expression to be bound to the parameter. You must define a binding for each parameter. For example, if an XSL transformation has two input parameters named **one** and **two**, the **Variable Name** field has two labels **one** and **two—** with a text box associated with each into which the XQuery expression is entered.
	Enter the expression directly, or drag variables and structures from the right panel.
	The following XQuery expressions are examples of valid input to this field:
	`$body/*[1]`
	`$body/po:PurchaseOrder`
	Note: The following variable name is not a valid entry for this field and results in an exception:
	`body`

Dynamic XQuery page

This page appears in the XQuery/XSLT Expression Editor only.

You can specify a dynamic XQuery expression that evaluates at runtime to the name of a pre-registered XQuery resource. Oracle Service Bus executes this XQuery resource, with optional variable bindings, against the message context to produce the required transformation.

Use the XQuery Resource page to configure a dynamic XQuery transformation to be executed in the message flow of a proxy service.

Field	Description
Expression	The XQuery expression that will evaluate at runtime to the name of a pre-registered XQuery resource. The following shows the syntax for the XQuery resource (representing the full name of the resource): `Project/folder1/folder2/XQueryResourceName`
Select XQuery Template	The resource to serve as a template for the shape of the query (the number and names of the variables). Click **Browse** to select an existing registered resource. After selecting a template, the variables appear in the Bind Variables area. Note that the template is not persisted with the configuration. Instead, the template serves as a quick start to help you specify the variables for the query.
Add Custom Variable	An input parameter of the transformation. Type a variable name in the Add Custom Variable field, and click **Add**.
Bind Variables	When you add a custom variable, it appears in the Bind Variables area. Similarly, when you select an XQuery template, each input parameter of the transformation is displayed. Each label corresponds to the name of a parameter, and each text box is for defining an XQuery expression to be mapped to the parameter. You must define a mapping for each parameter. Enter the expression directly, or drag variables and structures from the right panel.

Variable Structures page

This page appears in the XQuery/XSLT Expression Editor, the XPath Expression Editor, and the Condition Editor.

The Variable Structures page displays variables and their contents as trees. It includes the built-in message context variables `attachments`, `body`, `header`, `outbound`, and `inbound`, as well as any user-defined variables. It includes `fault` if the context of the expression is appropriate (that is, in an error handler). The `outbound` variable is always listed; even though it is not valid in every context. If `outbound` is used in invalid contexts, it will be reported when validating.

Each variable structure mapping entry has a label and maps a variable or variable path to one or more structures. The scope of these mappings is a stage or a route node.

You can drag items from this page to the editor's Expression page to insert them into the current expression. They are inserted as XPath expressions.

To define a new variable and add it to the variable structure tree, click **Add** to open the Add Variable Structure dialog.

For more information about working with variable structures, see Working with Variable Structures in the *Oracle Service Bus User Guide*

Namespace Definitions page

This page appears in the XQuery/XSLT Expression Editor, the XPath Expression Editor, and the Condition Editor.

The Namespace Definitions page lists default Oracle Service Bus namespaces, variable namespaces, and user-defined namespaces. Click **Add** to define a new namespace, which is then added to the list of user-defined namespaces. To modify a user-defined namespace, select it in the list, then click **Edit**.

XQuery Functions page

This page appears in the XQuery/XSLT Expression Editor, the XPath Expression Editor, and the Condition Editor.

The XQuery Functions page lists a set of standard XQuery functions, organized alphabetically and by type. You can drag functions from this page to the editor's Expression page to insert them into the current expression. When you insert a function into an expression, placeholders are used for parameter values you must supply.

Add Variable Structure dialog

Use this dialog to define a variable and add it to the tree of variable structures in the Variable Structures page of the Expression Editors.

You create variable structures in this dialog to define the structure of a variable for design purposes. For example, it is easier to browse the XPath variable in the structure view rather than viewing the XML schema of the XPath variable. Variable structures do not create variables. Variables are created at runtime as the target of the assign action in the stage.

You can declare your own variable structures based on:

- XML types, including

  - Schema elements
  - WSDL elements
  - Schema types
  - WSDL types
- MFLs
- Service interfaces
- Simple types (string or any XML)

You can use this feature directly for all user-defined variables, as well as `$inbound`, `$outbound`, and `$fault`. However, you cannot use it directly to access XML attachments in `$attachments`, headers in `$header`, or documents and RPC parameters in `$body`, with one exception— you can use it directly to access documents and parameters in `$body` for request messages received by a WSDL proxy service.

When you create a variable structure based on XML types, MFLs, or service interfaces, you must enter the following information into the fields at the top of the page:

Option	Description
Structure Label	A display name for the variable you want to create. This display name enables you to give a meaningful name to the structure so you can recognize it at design time but it has no impact at run time.
Structure Path	The path of the variable structure at run time

For more information about working with variable structures, see Working with Variable Structures in the *Oracle Service Bus User Guide*

New XSL Transformation wizard

Transformation maps describe the mapping between two data types. eXtensible Stylesheet Language Transformation (XSLT) maps describe XML-to-XML mappings. For more information, see XSL Transformations in *Using the Oracle Service Bus Console*.

Use this page to select a name and location for a new XSL transformation. This wizard creates a skeleton for the XSLT. Add details in the XSLT editor.

Option	Description
Enter or select the parent folder	Enter or select the name of the project or folder in which to locate a new XSL transformation.
File Name	Enter a name for this new XSL transformation.

Split-Join User Interface Reference

The following sections describe the fields and user interface components involved with creating and configuring Split-Joins in Oracle Service Bus.

- Split-Join Design Palette
- Global / Start Node Properties
- Variable Properties
- Error Handler Properties
- Invoke Service Properties
- Reply Properties
- For Each Properties
- If Properties
- If and Else If Properties
- Parallel Properties
- Raise Error Properties
- Repeat Until Properties
- Re-Raise Error Properties
- Scope Properties
- While Properties
- Assign Properties

- Copy Properties
- Delete Properties
- Insert Properties
- Java Callout Properties
- Log Properties
- Replace Properties
- Receive Properties
- Counter Variable Dialog
- Create/Edit Variable Dialog
- Create Message Variable Dialog
- Service Browser
- SOAP Fault Variable Dialog
- WSDL Browser
- Split-Join Wizard - New Split-Join
- Split-Join Wizard - Specify Operations

Split-Join Design Palette

The Split-Join Design Palette lists all the operations you can use to construct a Split-Join. To insert a control into a Split-Join, drag the icon from the palette to the Split-Join editor. When you drag an item to the editor, one or more circles (drag-points) appear () to indicate that you can drop the dragged item in that position on the editor. When you drag the item into such a circle, it is highlighted () to show that you can drop the item there.

Operations

The Split-Join Design palette is organized into the following categories:

- Communication
- Flow Control

- Assign Operations

The following table describes the operations you can add to a Split-Join:

Operation	Description
Communication	
Invoke Service	Invoke Service invokes a WSDL-based, non-transport-typed Business Service, a WSDL-based Proxy Service, or a Split-Join. For configuration properties, see Invoke Service Properties.
Reply	Reply sends a response or fault back to the Oracle Service Bus Message Flow. For configuration properties, see Reply Properties.
Flow Control	
For Each	For Each executes logic configured within its Scope a specified number of times. For configuration properties, see For Each Properties.
If	If provides conditional behavior within a Split-Join. For configuration properties, see If Properties.
Parallel	Parallel creates a fixed number of configured parallel branches. For configuration properties, see Parallel Properties.
Raise Error	Raise Error generates an error that causes the Split-Join to stop normal processing. If the error is not handled using an Error Handler, the Split-Join will terminate and a Fault will be sent to the Oracle Service Bus Message Flow. For configuration properties, see Raise Error Properties.
Repeat Until	Repeat Until lets you repeat operations until a condition evaluates to true within a Split-Join. The condition is evaluated after each loop finishes. For configuration properties, see Repeat Until Properties.

Operation	Description
Re-Raise Error	Re-Raise Error lets you re-raise an error caught by an Error Handler Catch or CatchAll. For configuration properties, see Re-Raise Error Properties.
Scope	Scope creates a context which influences the behavior of its enclosed operations. For configuration properties, see Scope Properties.
While	While lets you repeat operations until a condition evaluates to false within a Split-Join. The condition is evaluated before each loop commences. For configuration properties, see While Properties.
Assign Operations	
Assign	Lets you assigns the result of an XQuery expression to a Variable. For configuration properties, see Assign Operation Properties.
Copy	Lets you copy the information specified by an XPath expression from a source document to a destination document. For configuration properties, see Copy Properties.
Delete	Lets you delete a set of nodes specified by an XPath Expression. For configuration properties, see Delete Properties.
Insert	Lets you insert the result of an XQuery expression at an identified place relative to nodes selected by an XPath Expression. For configuration properties, see Insert Properties.
Java Callout	Lets you invoke a static Java method from a Split-Join for custom actions such to be handled in Java such as validation, transformation, and logging. For configuration properties, see Java Callout Properties.

Operation	Description
Log	Lets you log data at a specified severity so that administrators can take appropriate action. For configuration properties, see Log Properties.
Replace	Lets you replace a node or the contents of a node specified by an XPath expression. For configuration properties, see Replace Properties.

Operations have a General properties tab for changing the node's label and providing comments.

Global / Start Node Properties

The start node in a Split-Join specifies its global properties. Among these properties, global variables and associated External Services can be reviewed and configured by expanding the left-side arrow. The global Error Handler can be reviewed and configured by expanding the right-side arrow.

Use the Properties view to review and configure the Global Properties of the selected Split-Join.

This page has two tabs:

- Imports
- General

The Imports tab has the following options:

Option	Description
WSDL Imports	Displays WSDL Imports used by the Split-Join. Select a WSDL in the list and right-click to delete it.
Schema Imports	Displays Schema Imports used by the Split-Join. Select a Schema in the list and right-click to delete it.

The General tab has the following options:

Option	Description
Label	Enter a label for the file defining the Process Node.
Documentation	Enter a description and/or comments.

Variable Properties

Use the Properties view to review and configure Variables in the selected Split-Join.

Option	Description
(tree)	Depicts the hierarchical structure of the Variable's type.
Edit	Click to display the Create/Edit Variable Dialog, where you can modify the Variable's name and type.

Error Handler Properties

The Error Handler receives and handles all of the errors that are raised in a Split-Join.

An Error Handler lets you add Catch and CatchAll operations.

Use the Properties view to review and configure the selected Error Handler in a Split-Join.

The Catch tab has the following options.

Option	Description
SOAP Fault Variable Name	Defines a variable to contain SOAP (1.1 or 1.2) faults. If the Error Handler is executed due to a SOAP fault received from invoked external services, this variable is populated with the received SOAP fault.
Fault Name – Define Fault	Lets you define a custom local fault. Click **Define Fault** to enter a Namespace and Fault Name.
Fault Name – Predefined	Lets you select an existing WSDL, Application, or Standard fault. Click **Pick Fault** to select an existing fault.

The catchAll tab has the following options.

Option	Description
SOAP Fault Variable Name	Defines a variable to contain SOAP (1.1 or 1.2) faults. If the Error Handler is executed due to a SOAP fault received from invoked external services, this variable is populated with the received SOAP fault.

Invoke Service Properties

The Invoke Service invokes a WSDL-based, non-transport-typed Business Service, a WSDL-based Proxy Service, or another Split-Join.

Use the Properties view to review and configure the selected Invoke Service in a Split-Join.

The Operation tab has the following options:

Option	Description
Operation	The operation to be invoked by the Service. Click **Browse** to select the operation you want to invoke. When you select an operation, a dashed blue line appears pointing to the external service in the Split-Join editor.
Service Location	The location of the invoked Service. Click the location path to open the service file.

The Input Variable tab has the following options:

Option	Description
Message Variable	A list of message type variables with the type matching the operation's input message type. Select **Create Message Variable** to define a new message variable. **Note:** If message type variables with the type matching the operation's input message type do not exist, you must define a new message type variable with the required type.
Message Type Namespace	The namespace of the operation's input message type.
Message Type	The operation's input message type.

The Output Variable tab has the following options:

Option	Description
Message Variable	A list of message type variables with the type matching the operation's output message type. Select **Create Message Variable** to define a new message variable. **Note:** If message type variables with the type matching the operation's output message type do not exist, you *must* define a new message type variable with the required type.
Message Type Namespace	The namespace of the operation's output message type.
Message Type	The operation's output message type.

Invoking Another Split-Join

A Split-Join can invoke another Split-Join in the same Oracle Service Bus configuration. This functionality provides more flexibility in service design, letting you split up complex Split-Join functionality into multiple Split-Joins, allowing for componentization and re-use of Split-Join functionality. Performance is maintained, because there is no marshalling and unmarshalling of data between the Split-Joins.

You must ensure that you do not create circular Split-Join references. Oracle Service Bus does not check for circular references.

Reply Properties

Reply sends a response or fault back to the Oracle Service Bus message flow.

Use the Properties view to review and configure the selected Reply in a Split-Join.

The Operation tab has the following options:

Option	Description
Operation	The operation to be invoked by the Reply.

The Variable tab has the following Select options:

Response Options	Description
Message Variable	A list of the message variables whose type matches the operation's output message type. Select **Create Message Variable** to define a new message variable. **Note:** If message type variables with the type matching the operation's output message-type do not exist, you must define a new message type variable with the required type.
Message Type Namespace	The namespace of the operation's output message type.
Message Type	The operation's output message type.
Fault Options	
Pick WSDL Fault/SOAP Fault	Determine whether the fault reply is a fault message defined in the operation of the WSDL or an explicit SOAP fault message.
WSDL Fault Name	Select a fault name from the list of faults defined in the operation of the WSDL.
Message Variable	A list of the Message Variables whose type matches the operation's output message type. Select **Create Message Variable** to define a new message variable. **Note:** If message type variables with the type matching the operation's output message-type do not exist, you must define a new message type variable with the required type.

Response Options	Description
Message Type Namespace	The namespace of the operation's output message type.
Message Type	The operation's output message type.
SOAP Fault	Select SOAP Fault See SOAP Fault Variable Dialog.
Propagate SOAP Fault	Propagate the SOAP fault in the SOAP fault variable defined in the Error Handler. See Error Handler Properties.

Reply automatically includes an implicit Exit operation to end that instance of the flow without triggering a fault. The Exit operation is not visible in the development environment.

For Each Properties

For Each executes logic configured within its Scope a specified number of times.

Use the Properties view to review and configure the selected For Each in a Split-Join.

The Counter Variables tab has the following options:

Option	Description
Parallel	Select one of the following options: • If **yes** is selected, each iteration of For Each is executed in parallel. • If **no**, each iteration of For Each is executed sequentially.
Counter Variable Name	Defines an implicit variable within the Scope of the For Each. Each iteration of the For Each contains an isolated instance of this variable. In turn, every instance is set to an iteration number; for example, the first iteration has its value set to **Start Counter Value**, the second iteration to **Start Counter Value+1**, etc.

Option	Description
Start Counter Value	The value of the Counter Variable for the first iteration of For Each. Determined as the result of an XPath 1.0 expression. The result must be “1” or more. (“0” is not a valid Start Counter Value.) The browse button launches the expression builder. The expression should generate an integer for the initial Start Count Value.
Final Counter Value	The value of the Counter Variable for the final iteration of For Each. Determined as the result of an XPath 1.0 expression. The result must be “1” or more. (“0” is not a valid Final Counter Value.) The browse button launches the expression builder. The expression should generate an integer for the Final Count Value.

The Completion Condition tab has the following options:

Option	Description
Number of Finished Branches	An optional expression that determines when to stop creating branches. Depending on the context in the Split-Join, the expression prevents some of the children from executing or forces early termination of some children. The browse button launches the expression builder.
Successful Branches Only?	Select this option if you want only successfully completed branches to be counted when determining if the completion condition has been met.

If Properties

If nodes contain If, Else If, and Else operations. See If and Else If Properties.

Use the Properties view to review and configure the selected If in a Split-Join.

If and Else If Properties

The If and Else If operations provide conditional behavior within a Split-Join.

Use the Properties view to review and configure the selected If or Else If in a Split-Join.

If and Else If have the following options:

Option	Description
Condition	Define an XPath 1.0 expression that evaluates to true or false. If the condition evaluates to true, the associated If/Else If branch is executed. Click the browse icon to launch the expression builder.

Parallel Properties

Parallel lets you create a static Split-Join that handles fixed number of message requests. Parallels contain one or more Scope branches.

Use the Properties view to review and configure the selected Parallel in a Split-Join.

Raise Error Properties

Raise Error establishes an error condition under which the execution of the process will halt.

Use the Properties view to review and configure the selected Raise Error in a Split-Join.

The Raise Error tab has the following options:

Option	Description
Define Fault	Lets you specify a custom local fault not contained in the WSDL. Click the Fault link to specify the fault Namespace and Fault Name.
Pick Fault From WSDL	Lets you select an existing fault specified in the WSDL. Click the **Pick Fault** link to select the WSDL fault.

Repeat Until Properties

Repeat Until lets you repeat operations until a condition evaluates to true within a Split-Join. The condition is evaluated after each loop finishes.

Use the Properties view to review and configure the selected Repeat Until in a Split-Join.

The Condition tab has the following options:

Option	Description
Condition	Define an XPath 1.0 expression that evaluates to true or false. The operation(s) in the Repeat Until are executed until the condition evaluates to true. Click the browse icon to launch the expression builder.

Re-Raise Error Properties

Use Re-Raise Error within an Error Handler to re-raise an error caught by a Catch or a CatchAll. In the Properties view you can rename and enter comments about the Re-Raise Error node.

Use the Properties view to review and configure the selected Re-Raise Error in a Split-Join.

Scope Properties

The Scope creates a context which influences the behavior of its enclosed operations. Local variables and the Error Handler defined within the Scope are restricted to this context.

Use the Properties view to review and configure the selected Scope in a Split-Join.

Related Topics

Scope and Variables

While Properties

While lets you repeat operations until a condition evaluates to false within a Split-Join. The condition is evaluated before each loop commences.

Use the Properties view to review and configure the selected While in a Split-Join.

The Condition tab has the following options:

Option	Description
Condition	Define an XPath 1.0 expression that evaluates to true or false. If the condition evaluates to true, the associated operations is executed. Click the browse icon to launch the expression builder.

Assign Properties

Assign lets you perform data manipulation, including initializing and updating a Variable. You can perform the following operations in an Assign node: Assign, Copy, Delete, Insert, Java Callout, Log, and Replace.

Use the Properties view to review and configure the selected Assign in a Split-Join.

Assign Operation Properties

The Assign tab has the following options:

Option	Description
Expression	An XQuery expression used to create the data that will be assigned to the Variable. Clicking the expression launches the XQuery editor.
Variable	A variable to which the result of the XQuery expression is assigned. Only previously defined Variables, the Counter Variable, and SOAP Fault Variables (for Error Handlers) are available.

Oracle Service Bus's Assign functionality in Split-Joins conforms to the WS-BPEL specification for resolution of XPath/XQuery expressions to simple type variables. Supported simple types for binding XPath/XQuery expressions to variables in Split-Joins are String, Boolean, and Float. The Assign converts the value you provide the type with which the variable is defined.

For example:

- If you assign <foo><bar>4</bar></foo> to a response variable defined as a String ($response.result), Oracle Service Bus returns <bar>4</bar> as a String in the <result> through a simple copy of the child element and value.
- If you map <foo><bar>4</bar></foo> to a String variable (such as myStr), then assign $myStr to $response.result, Oracle Service Bus returns <result>4</result>, because it first converts the value in $myStr to a String before it makes the assignment to the $response.result String variable.

Copy Properties

The Copy operation copies the information specified by an XPath expression from a source document to a destination document.

Use the Properties view to configure a Copy Operation in an Assign. Configure Select From and Select To using the following guidance.

The Copy tab has the following options:

Option	Description
Keep Source Element	Lets you determine which element name (source or destination) is used when values are copied from a source to a destination. If you do not select this option, the existing element name in the destination is used to hold the copied value. If you select this option, the name of the source element is used in the destination to hold the copied value. For example, if you are copying a zip code value from <zipCode>80303</zipCode> in the source, selecting Keep Source Element uses the <zipCode> element in the destination. If you do not select the option, the zip code value is copied to the existing destination element, such as <pinCode>80303</pinCode>.
Choose Type	Allows the user to select the desired type: Variable, Expression, Literal, or XML Fragment. Literals and XML fragments are available only in the Select From menu.
Choose Type - Variable	Select an XPath on a Variable. Use this option when simple node selection on a Variable is required. When a node is selected by expanding the Variable tree, an Xpath 1.0 expression is automatically generated.
Choose Type - Expression	Create an Xpath 1.0 expression to select a node from a Variable. Use this option when more complex Xpaths (potentially with predicates) are required to select a node on a Variable. Click the **Expression** link to use the XPath 1.0 Expression Builder. **Note:** The entered Xpath 1.0 expression should only copy to one XML node/element, or there will be run-time errors.

Option	Description
Choose Type - Literal (Select From only)	A Literal string entered by the user. Click the **Literal** link to enter the string.
Choose Type - XML Fragment (Select From only)	An XML fragment entered by the user. Click the **XML Fragment** link to enter the fragment.

Delete Properties

The Delete operation deletes a set of nodes specified by an XPath Expression.

Use the Properties view to configure a Delete Operation in an Assign.

Note: Unlike a Delete operation in a proxy service message flow, a Delete operation in a Split-Join does not allow deleting a variable directly.

The Delete tab has the following options:

Option	Description
XPath	An XPath Expression that selects the nodes to be deleted. Click the **XPath** link to launch the expression editor**.**
In Variable	Variable on which the XPath expression is executed to select the nodes to be deleted.

Insert Properties

The Insert operation inserts the result of an XQuery expression at an identified place relative to nodes selected by an XPath Expression.

Use the Properties view to configure an Insert Action in an Assign.

The Insert tab has the following options:

Option	Description
Expression	An XQuery 1.0 expression used to create the data that will be inserted at a specified location in a Variable. Click the **Expression** link to launch the expression editor.
Location	The location used to control where the Insert operation is performed relative to the result of the XPath 1.0 Expression. Options include: • **before**: Immediately before the element specified by the result of the Xpath 1.0 Expression. • **after**: Immediately after the element specified by the result of the Xpath 1.0 Expression. • **as first child of**: The first child element of the element specified by the result of the Xpath 1.0 Expression. • **as last child of**: The last child element belonging to the element specified by the result of the Xpath 1.0 Expression.
XPath	An XPath that determines the nodes to be selected. Click the **Xpath** link to launch the expression editor.
Variable	A Variable to be evaluated by the XPath.

Java Callout Properties

A Java Callout operation lets you invoke a static Java method from a Split-Join for custom actions such to be handled in Java such as validation, transformation, and logging.

Use the Properties view to configure a Java Callout operation in an Assign.

The Java Callout tab has the following options:

Option	Description
Method	Package your Java class in a JAR file in your Oracle Service Bus project. Click **Browse** to first select the JAR, then the method you want to invoke. Using the following guidelines for the Java callout method: • The method must be static. Only the following Java types are supported for input parameters: • boolean, byte, char, double, float, int, long, short and arrays of these types • java.lang.[Boolean | Byte | Character | Double | Float | Integer | Long | Short | String] and arrays of these types • java.math.[BigInteger | BigDecimal] and arrays of these types • org.apache.xmlbeans.XmlObject and arrays of this type Only the following Java types are supported for method return: • All types supported for input parameters except their array equivalent • void
Expression	An XQuery 1.0 expression used to map data to the input parameters of the static Java method. Click the **Expression** link to launch the expression editor.
Result Value	Select the variable to contain the result value for the Java method.
Service Account	You can use Service Account to put the appropriate subject on the thread when executing the Java callout. Click **Browse** to select a service account.

Oracle Service Bus provides the following errors for Java callouts:

- 2031350 – Received more than one element when only one is expected during conversion to Java
- 2031351 – Found simple type instead of XmlObject when converting to Java
- 2031352 – Error converting simple type to its corresponding Java type
- 2031353 – Received exception during invocation of the Java method
- 2031354 – Errors setting the security context

Log Properties

The Log operation lets you log Split-Join data at a specified severity to the server log file. Administrators can use log information to take appropriate action based on the severity of the data logged.

Use the Properties view to configure a Log operation in an Assign.

The Log tab has the following options:

Option	Description
Expression	An XQuery 1.0 expression used to select the data to be logged. Click the **Expression** link to launch the expression editor.
Annotation	Optionally specify a note for the log. The annotation is logged along with the data selected by the expression.
Severity	Select one of the following the severity levels for the log: • Debug • Info • Warning • Error

Replace Properties

The Replace operation lets you replace a node or the contents of a node specified by an XPath expression.

Use the Properties view to configure a Replace operation in an Assign.

The Replace tab has the following options:

Option	Description
XPath	An XPath 1.0 Expression used to specify the data (in the Variable) that will be replaced. Click the **XPath** link to launch the expression editor.
Variable	The Variable that contains the data to be replaced.

Option	Description
Expression	An XQuery 1.0 expression used to create the data that replaces the data specified by the XPath in the named Variable. Click the **Expression** link to launch the expression editor.
Replace entire node	Specifies that the nodes selected by the expression are replaced along with all of its contents.
Replace node contents	Specifies that the node is not replaced. Only the contents are replaced.

Receive Properties

Use the Properties view to review and configure the selected Receive in a Split-Join.

The Operation tab has the following options:

Option	Description
Operation	The operation to be invoked by the Receive. Browse to select an operation from the WSDL Browser.

The Variable tab has the following options:

Option	Description
Message Variable	A list of the message variables whose type matches the operation's output message type Select **Create New Variable** to define a new message variable. **Note:** If message type variables with the type matching the operation's output message-type do not exist, you *must* define a new message type variable with the required type.
Message Type Namespace	The namespace of the operation's output message type.
Message Type	The operation's output message type.

Counter Variable Dialog

Use this dialog to create a Counter Variable.

Option	Description
Counter Variable Name	The name of the Counter Variable

Create/Edit Variable Dialog

Use this dialog to review and/or configure Variables in the Split-Join.

Scope and Variables

Although variables are visible in the scope in which they are defined and in all scopes nested within that scope, a variable declared in an outer scope is hidden when you declare a variable with an identical name in an inner scope. For example, if you define variable myVar in an outer scope (So) and then define variable myVar again in an inner scope (Si) which is contained by scope So, then you can only access the myVar you defined in the inner scope Si. This myVar overrides the myVar you defined in scope So.

Option	Description
Name	Enter a name for the Variable that is unique within the Scope.
Select Variable Type	Select one of the following Variable Types: • Built-in Types • Schema Types • Message Types
Select Variable Type - Built-in Types	When selected, only Built-in Type Variables are displayed.
Select Variable Type- Schema Types	When selected, all Schema Types in the current Oracle Service Bus Configuration are displayed. If “Show only applicable schema types/elements for this Split-Join” is selected, only Schema Types directly applicable to the current Split-Join are displayed. This is checked by default.

Option	Description
Select Variable Type - Message Types	When selected, only message types are displayed. If “Show only applicable schema types/elements for this Split-Join” is selected, only Message Types directly applicable to the current Split-Join are displayed. This is checked by default.
Type	The Variable type.
Namespace	The namespace of the Variable’s type.

Create Message Variable Dialog

Use this dialog to create a new Variable.

Option	Description
Name	Enter a unique name for the Variable.
Type	One of the following Variable types: WSDL message, XSD element, or XSD type (simple, complex, built-in).
Namespace	The namespace of the Variable’s type.

Service Browser

Use this dialog to browse for and select an operation from a business service, a proxy service, or a Split-Join in the tree.

SOAP Fault Variable Dialog

Use this dialog to create a SOAP fault variable.

Option	Description
SOAP Fault Variable Name	The name of the SOAP fault variable.

WSDL Browser

Use this dialog to browse for and select an operation from a WSDL in the depicted tree.

Split-Join Wizard - New Split-Join

Use this page to locate and name the new Split-Join

Option	Description
Enter or select the parent folder	Enter the name of the folder to contain this Split-Join, or select a folder from the list.
File Name	Enter a name for the file defining the Split-Join.

Split-Join Wizard - Specify Operations

Use this page to select an operation for the new Split-Join.

Option	Description
Select Operation	Select the operation from those available in the tree.
Consume	Click this button to import WSDLs from outside your current Oracle Service Bus configuration that can subsequently be used to select an operation.

CHAPTER 4

Transport Configuration

When you configure a business service or a proxy service, you must configure the transport used by the service. Each transport is configured on its own configuration page.

Protocol-Specific Transport Configuration Pages

Each transport available for business services and proxy services has its own configuration page. Those built into Oracle Service Bus are:

BPEL-10g Transport Configuration Page (Business Services)

Use this page to configure transport settings for a business service using the BPEL-10g (Oracle BPEL Process Manager) transport protocol. For more information on using Oracle Service Bus with Oracle BPEL Process Manager, see the Oracle BPEL Process Manager Transport User Guide.

Property	Description
Role	The BPEL transport is used to send request messages from Oracle Service Bus to Oracle BPEL Process Manager. The business service can serve one of the following roles: • **Synchronous Client** For synchronous communication with an Oracle Service Bus client, the only location information that is required is the BPEL address. This address is captured statically as the endpoint URI and/or dynamically through URI rewriting. • **Asynchronous Client** For asynchronous communication with an Oracle Service Bus client, a callback from Oracle BPEL Process Manager to Oracle Service Bus is sent on a different connection than the request, and you must configure Oracle Service Bus to provide the correct callback address. For more information, see "Creating a Callback Proxy for Asynchronous Communication" in the Oracle BPEL Process Manager Transport User Guide. • **Service Callback** If the business service is designed to be a service callback to Oracle BPEL Process Manager (where Oracle BPEL Process Manager is calling an external service through Oracle Service Bus), the callback address is known only at run time. Use an Endpoint URI of bpel://callback. If you configure the business service with the marker URI, configure your pipeline logic to dynamically set the URI on $outbound; for example, using the TransportHeader action. **Note:** A Service Callback business service does not support load balancing or failover.
Callback Proxy	This optional field is available only for the Asynchronous Client role. This field lets you select the proxy service (must be either an SB or HTTP proxy of type Any SOAP) that will be used to route callbacks to the Oracle Service Bus client that made the request. For more information, see "Creating a Callback Proxy for Asynchronous Communication" in the Oracle BPEL Process Manager Transport User Guide.

Property	Description
Service Account	For JNDI context security, used to access the Oracle BPEL Process Manager delivery service. Click Browse and select a service account. If no service account is specified, an anonymous subject is used. There is no restriction on the type of service account that can be configured, such as static or pass-through, but the run time must be able to access a user name and password.
Suspend Transaction	Selecting Suspend Transaction makes the business service non-transactional even if the business service is invoked by a transaction. If you do not select Suspend Transaction: • If the protocol indicates a WebLogic Server-supported protocol (t3, iiop, http), the transaction is propagated. • If the protocol indicates an OC4J server (ormi, opmn), the BPEL transport throws an exception, since OC4J does not support transaction propagation. The BPEL transport ignores the Suspend Transaction option in the following situations: • The business service is called with quality of service (QoS) "best-effort." The BPEL transport automatically suspends any transaction that does not support QoS. • The business service is called with QoS set to "exactly-once" and there is no transaction. For a description of transaction propagation, see "Transaction Propagation" in the Oracle BPEL Process Manager Transport User Guide.
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. For information about Work Managers, see the following WebLogic Server Administration Console Online Help topics: • "Using Work Managers to Optimize Scheduled Work" at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/config_wls/self_tuned.html • "Create Global Work Managers" at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/ConsoleHelp/taskhelp/work/CreateGlobalWorkManager.html

DSP Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the DSP (Oracle Data Service Integrator) transport protocol. For more information on accessing Oracle Data Service Integrator from Oracle Service Bus, see Accessing Data Services Through Oracle Service Bus.

Option	To create or edit...
Debug Level	Specify one of the following • `0` - for no debug information • `1` - to output information on the request message • `3` - to output information on the request and the response message
Service Account	• Click **Browse** and select a service account from the list displayed. If no service account is specified, an anonymous subject is used.
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. The Work Manager is used to post the reply message for response processing. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.

EJB Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the EJB transport protocol.

Option	To create or edit...
Pass Caller's Subject	Select this check box to have Oracle Service Bus pass the authenticated subject from the proxy service when invoking the EJB and no service accounts are configured. Note that the **Service Account** field is disabled when this option is selected.
Service Account	• Click **Browse** and select a service account from the list displayed. If no service account is specified, an anonymous subject is used.

Option	To create or edit...
Supports Transaction	Select this check box to specify transaction support.
Client Jar	Click **Browse** and select an EJB client JAR resource from the list displayed.
Converter Jar	Click **Browse** and select an EJB converter class JAR resource from the list displayed. To learn more about EJB client JAR resources and converter classes, see EJB Transport in the *Oracle Service Bus User Guide*.
Home Interface	Select the required EJBHome interface from the options populated by the JAR. The JNDI name in this URI sample must be associated with the EJBHome interface you select here. If the EJB is not of the required type or an EJBHome interface is not specified in the client-jar, a warning will be displayed.
Remote Interface	This field is automatically populated depending on the configuration of the Home Interface.
Target Namespace	This field is populated by information picked up from the JAR.
Style	Select **Document Wrapped** or **RPC** according to your requirements. If two or more methods of your stateless session EJB have the same number and data type of parameters, and you want the operations to be document-oriented, you must specify that they be document-wrapped. The style is important because when routing or publishing to the EJB, `$body` must have content that matches the style. Also when calling out to an EJB, the style affects the parameter contents, especially for document wrapped. Secondly one usage pattern is to define an EJB business service and then create a proxy service with the same WSDL that routes to the EJB. In this case care must be taken on the style of the WSDL because the client tool used to invoke the proxy might have limitations on the style of the WSDL.
Encoding	Select **Encoded** or **Literal**.

Option	To create or edit...
Methods	Select the required methods (you can select multiple methods). Click + to expand the method and: edit the default parameter values and select a converter if provided (or required).
Exceptions	This field appears if a method throws a business exception. If an EJB method throws an exception that has data types not supported by Java Web Services (JWS), such as an ArrayList, use the Exceptions field to select a converter class that converts the exception to a type supported by JWS. Your converter class must implement com.bea.wli.sb.transports.ejb.ITypeConverter. Converter classes can only be configured for checked exceptions and not for run-time exceptions. Package the converter class and the converted exception class in the client or converter JAR. For more information, see the Interoperability with EJB Transport guide.

E-Mail Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the e-mail transport protocol.

Option	To create or edit...
SMTP Server	Select an SMTP server. This is a required field if you do not select a JNDI name in the **Mail Session** field.
Mail Session	Select the JNDI name of a configured session. This is a required field if you do not enter an SMTP server name in the **Mail Server** address field.
From Name	Enter a display name for the originating e-mail account for this service.
From Address	Enter the originating e-mail account for this service.
Reply To Name	Enter a display name for the reply to e-mail account.
Reply To Address	Enter an e-mail address to reply to.

Option	To create or edit...
Connection Timeout	Enter the timeout interval, in seconds, before the connection is dropped. If you enter `0`, there is no timeout.
Request Encoding	Accept the default `ISO-8859-1` as the character set encoding for requests in e-mail transports, or enter a different character set encoding.

E-Mail Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the e-mail transport protocol.

Option	To create or edit...
Service Account	Enter a service account name, or click **Browse** to select service accounts from a browser.
Managed Server	Select the Managed Server for polling in a clustered domain. This field is available *only* in a clustered domain.
Polling Interval	Enter a polling interval, in seconds.
E-Mail Protocol	Select POP3 or IMAP as the server type for the e-mail account.
Read Limit	Specify the maximum number of messages to read per polling sweep. Enter 0 to specify no limit.
Pass By Reference	Select this check box to stage the file in the archive directory and pass it as a reference in the headers. By default when you create a new service, the **Pass By Reference** option is selected and you must specify the archive directory location.
Pass Attachments by Reference	Select this check box to stage the attachments in the archive directory and pass them as a reference in the headers. By default, when the **Pass By Reference** option is selected, the **Pass Attachments By Reference** option is implicitly true and you must specify the archive directory location.

Option	To create or edit...
Post Read Action	Select what happens to a message after it has been read: • **Archive** - The message is archived. • **Delete** - The message is deleted. • **Move** - The message is moved. Move is only available with the IMAP protocol.
Attachments	Select how attachments are handled: • **Archive** - Attachments are saved to the archive directory. • **Ignore** - Attachments are ignored.
IMAP Move Folder	Enter the folder to which the message is moved if the **Post Read Action** field is set to **Move**.
Download Directory	Enter a temporary location for downloading e-mails.
Archive Directory	Specify the path to the archive location if the **Post Read Action** field is set to **Archive**. This field is required if the **Pass By Reference** or **Pass Attachments By Reference** option is selected.
Error Directory	Enter the file system directory path to write the message and any attachments if there is a problem.
Request Encoding	Accept the default `ISO-8859-1` as the character set encoding for requests in E-mail transports, or enter a different character set encoding.

File Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the file transport protocol.

Option	To create or edit...
Prefix	Enter a prefix to be prepended to the file name. Do not enter * in this field. This character causes a run-time exception.

Option	To create or edit...
Suffix	Enter a suffix to be appended to the file name. This is a required field. Do not enter * in this field. This character causes a run-time exception.
Request Encoding	Accept the default `utf-8` as the character set encoding for requests in File transports, or enter a different character set encoding.

File Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the file transport protocol.

Option	To create or edit...
File Mask	Enter the regular expression for the files to be picked. The default is `*.*`.
Managed Server	Select the Managed Server for polling in a clustered domain. This field is available *only* in a clustered domain.
Polling Interval	Enter a polling interval, in seconds. The default is `60`.
Read Limit	Specify the maximum number of messages to read per polling sweep. Enter `0` to specify no limit. The default is 10.
Sort By Arrival	Select this check box to specify that events are delivered in the order of arrival. Note that when this option is selected for a proxy service that is executed in a clustered environment, messages are always sent to the same server. In other words, load balancing across servers is ignored when this option is selected.
Scan Subdirectories	Select this check box to recursively scan all the directories.
Pass By Reference	Select this check box to stage the file in the archive directory and pass it as a reference in the headers.
Post Read Action	Select what happens to a message after it has been read: • **Archive** - The message is archived. • **Delete** - The message is deleted.

Option	To create or edit...
Stage Directory	Enter an intermediate directory to temporarily stage the files while processing them. Do not put the stage directory inside of the polling directory (the directory identified in the URL of the file transport proxy service; for example, file:///*c:/dir1/dir2*).
Archive Directory	Specify the path to the archive location if the **Post Read Action** option is set to **Archive**. The Archive Directory field is also a required field if you have selected the Pass By Reference field. Do not put the archive directory inside of the polling directory.
Error Directory	Enter the location where messages and attachments are posted if there is a problem. Do not put the error directory inside of the polling directory.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in file transports, or enter a different character set encoding.

FTP Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the ftp transport protocol.

Option	To create or edit...
User Authentication	Select **anonymous** if the user of the FTP server is anonymous, or select **external user** if the user of the FTP server is an externally configured account.
Identity (E-mail id)	This field is available only if the **User Authentication** option is set to **anonymous**. Enter the mail ID for the anonymous user.
Service Account	This field is available only if the **User Authentication** option is set to **external user**. Enter the service account for the external user.
Timeout	Enter the socket timeout interval, in seconds, before the connection is dropped. The default is 60 seconds.
Prefix for destination File Name	Enter a prefix for the file name under which the file is stored on the remote server. Do not enter * in this field. This character causes a run-time exception.

Option	To create or edit...
Suffix for destination File Name	Enter a suffix for the file name under which the file is stored on the remote server. Do not enter * in this field. This character causes a run-time exception.
Transfer Mode	Select **ASCII** or **binary** as the transfer mode.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in ftp transports, or enter a different character set encoding.

FTP Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the ftp transport protocol.

Option	To create or edit...
User Authentication	Select **anonymous** if the user of the FTP server is anonymous, or select **external user** if the user of the FTP server is an externally configured account.
Identity (E-Mail ID)	This field is available only if the **User Authentication** option is set to **anonymous**. Enter the mail ID for the anonymous user.
Service Account	This field is available only if the **User Authentication** option is set to **external user**. Enter the service account for the user. This is a required field when the **User Authentication** option is set to **external user**.
Pass By Reference	Select this check box to stage the file in the archive directory and pass it as a reference in the headers.
Remote Streaming	Select this check box to stream the FTP files directly from the remote server at the time of processing. When you select this option, the archive directory is the remote directory on the remote FTP server machine. Therefore, you should specify the archive directory as relative to the FTP user directory.
File Mask	Enter the regular expression for the files to be picked. The default is `*.*`.
Managed Server	Select the Managed Server for polling in a clustered domain. This field is available *only* in a clustered domain.
Polling Interval	Enter a polling interval, in seconds. The default is `60`.

Option	To create or edit...
Read Limit	Specify the maximum number of messages to read per polling sweep. Enter `0` to specify no limit. The default is `10`.
Post Read Action	Select what happens to a message after it has been read. • **Archive** - The message is archived. • **Delete** - The message is deleted.
Transfer Mode	Select **ASCII** or **binary** as the transfer mode.
Archive Directory	Specify the path to the archive location if the **Post Read Action** option is set to **Archive**. This field is required if the **Pass By Reference** option is selected. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Download Directory	Enter the directory on your local machine where files are downloaded during the file transfer. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Error Directory	Enter the location where messages are posted if there is a problem. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in FTP transports.
Advanced Settings	
Scan Subdirectories	Select this check box to recursively scan all directories
Sort By Arrival	Select this check box to deliver events in the order of arrival.
Timeout	Enter the socket timeout interval, in seconds, before the connection is dropped. If you enter `0`, there is no timeout.
Retry Count	Specify the number of retries for FTP connection failures.

HTTP Transport Configuration Page (Business Services)

Use this page to configure transport settings for a business service using the HTTP transport protocol. The HTTP transport supports both HTTP and HTTPS endpoints.

Option	To create or edit...
Timeout	Enter the response timeout interval in seconds. If you enter 0, there is no timeout.
HTTP Request Method	This parameter lets you to use one of the following HTTP methods in a request: • **POST** – Passes all its data, of unlimited length, directly over the socket connection as part of its HTTP request body. The exchange is invisible to the client, and the URL does not change. For REST-based requests, POST tells the transport to perform a create/replace operation or perform an action with the request. • **GET** – You can include as part of the request some of its own information that better describes what to get. This information is passed as a sequence of characters appended to the request URL in a query string. You can use GET in a business service with a Service Type of "Any XML Service," or with a Service Type of "Messaging Service" when the Request Message Type is set to "None." For REST-based requests, GET retrieves a representation of a remote resource. • **PUT** – You can use PUT in a business service with a Service Type of "Any XML Service" or "Messaging Service." PUT tells the transport to perform a create/replace operation with a REST-based request, such as uploading a file to a known location. • **HEAD** – You can use HEAD in a business service with a Service Type of "Any XML Service," or with a Service Type of "Messaging Service" when the Response Message Type is set to "None." HEAD tells the transport to get header information for a remote resource rather than getting a full representation of the resource in a REST-based request. • **DELETE** – You can use PUT in a business service with a Service Type of "Any XML Service" or "Messaging Service." DELETE tells the transport to perform a delete operation with a REST-based request. **Note:** If a method is already set in the $outbound/transport/request/http:http-method variable, that value takes precedence over any method you select for HTTP Request Method.

Option	To create or edit...
Authentication	Select one of the following: • **None** - Specifies that authentication is not required to access this service. • **Basic** - Specifies that basic authentication is required to access this service. Basic authentication instructs WebLogic Server to authenticate the client using a user name and password against the authentication providers configured in the security realm, such as a Lightweight Directory Access Protocol (LDAP) directory service and Windows Active Directory. The client must send its user name and password on the HTTP request header. Basic authentication is strongly discouraged over HTTP because the password is sent in clear text. However, it is safe to send passwords over HTTPS because HTTPS provides an encrypted channel. **Warning:** By default, all users (authorized and anonymous) can access a business service. To limit the users who can access a business service, create a transport-level authorization policy. can access a proxy service, create a transport-level authorization policy. See "Editing Transport-Level Access Policies" under Security Configuration in the *Using the Oracle Service Bus Console*. **Client Certificate** - Specifies encrypted communication and strong client authentication (two-way SSL). To learn more, see Configuring Transport-Level Security in the *Oracle Service Bus Security Guide*.
Service Account	• Enter a service account. A service account is an alias resource for a user name and password. This is a required field if you selected the **Basic Authentication Required** field.
Follow HTTP redirects	Select this check box to specify that HTTP redirects (which are requests with a response code `3xx`) should be automatically followed. A re-direct occurs when you send an outbound request to the URL of a business service, and that service returns a response code (for example, `302`) that says the URL is no longer valid and this request needs to be sent to another URL. If the **Follow HTTP Redirects** check box is selected, Oracle Service Bus automatically re-sends the request to the new URL without any action on your part. Deselect this check box if you do not want the HTTP redirects to be automatically followed.
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.

Option	To create or edit...
Request Encoding	Accept the default `iso-8859-1` as the character set encoding for requests in HTTP transports, or enter a different character set encoding.
Response Encoding	Accept the default `iso-8859-1` as the character set encoding for responses in HTTP transports, or enter a different character set encoding.
Proxy Server	Enter a proxy server resource or click Browse to choose an entry from the list of configured proxy server resources.

HTTP Transport Configuration Page (Proxy Services)

Use this page to configure transport settings for a proxy service using the HTTP transport protocol. The HTTP transport supports both `HTTP` and `HTTPS` endpoints.

Option	To create or edit...
HTTPS required	Select this check box for inbound HTTPS endpoints. To learn more, see Configuring Transport-Level Security in the *Oracle Service Bus Security Guide*.
Authentication	Select one of the following: • **None** - Specifies that authentication is not required. • **Basic** - Specifies that basic authentication is required to access this service. Basic authentication instructs WebLogic Server to authenticate the client using a user name and password against the authentication providers configured in the security realm, such as a Lightweight Directory Access Protocol (LDAP) directory service and Windows Active Directory. The client must send its user name and password on the HTTP request header. Basic authentication is strongly discouraged over HTTP because the password is sent in clear text. However, it is safe to send passwords over HTTPS because HTTPS provides an encrypted channel. **Warning:** By default, all users (authorized and anonymous) can access a proxy service. To limit the users who can access a proxy service, create a transport-level authorization policy. See "Editing Transport-Level Access Policies" under Security Configuration in the *Using the Oracle Service Bus Console*. • **Client Certificate** - Specifies encrypted communication and strong client authentication (two-way SSL). To learn more, see Configuring Transport-Level Security in the *Oracle Service Bus Security Guide*. • **Custom Authentication** - Specifies that an authentication token is contained in an HTTP header. The client's identity is established through the use of this client-supplied token. You must configure an Identity Assertion provider that maps the token to an Oracle Service Bus user. The custom authentication token can be of any active token type supported by a configured WebLogic Server Identity Assertion provider.

Option	To create or edit...
Dispatch Policy	Select a dispatch policy for this endpoint. Leave blank to use the default dispatch policy. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Request Encoding	• For HTTP inbound transports: If the character set encoding parameter of the `Content-Type` header is not specified in Client Request, enter a character set encoding parameter. If you do not enter a value, the field defaults to `ISO-8859-1`. • For HTTP outbound transports: If you have not configured a request encoding, the Oracle Service Bus run time decides the most appropriate encoding while it makes a request to the business service. In the case of a non-passthrough scenario, the default character encoding is `UTF-8` at run time. However if it is a passthrough scenario, the run time will pass through the encoding received with the outbound response.
Response Encoding	Accept the default `ISO-8859-1` as the character set encoding for responses in HTTP transports, or enter a different character set encoding.
Advanced Settings	
Authentication Header	Enter the HTTP header (any except `Authorization`) from which Oracle Service Bus is to extract the token. This field is available only if you selected the **Custom Authentication** check box. For example, `client-xyz-token`.
Authentication Token Type	Select an authentication token type. Only the active token types configured for an Identity Assertion provider are available. (See "Configuring Identity Assertion Providers" under Configuring Custom Authentication in the *Oracle Service Bus Security Guide* for more information.) This field is available only if you selected the **Custom Authentication** check box.

JCA Transport Configuration Page (Proxy and Business Services)

Use this page to configure transport settings using the JCA transport protocol. For more information on using the JCA transport, see the JCA Transport User Guide.

Option	Description
Adapter Name	A read-only value showing the name of the adapter that the JCA service will use.
Adapter Type	A read-only value showing the adapter type.
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. For information about Work Managers, see the following *WebLogic Server Administration Console Online Help* topics: • Using Work Managers to Optimize Scheduled Work • Create Global Work Managers
JNDI Service Account	JNDI Service Account is for JNDI context security, used to access the EIS adapter managed connection factory. Click Browse and select a service account. If no service account is specified, an anonymous subject is used. For JCA business services, there is no restriction on the type of JNDI service account that can be configured, such as static or pass-through, but the run time must be able to access a user name and password. JCA proxy services can use only static JNDI service accounts. For more information on JNDI service accounts, see “Security” in the JCA Transport User Guide.

Option	Description
Always use configuration from JCA WSDL	This option determines whether or not connection factory properties, activation spec properties (proxy services), and interaction spec properties (business services) are always used from the WSDL. If this option is selected (default), the JCA transport interacts with the JCA framework in the following ways, depending on whether you are using managed or non-managed mode: • Managed mode – The activation/interaction spec defined in the JCA WSDL. The Activation/Interaction Spec Properties field is read-only; and if Connection Mode is set to Managed, the Connection Factory Properties field is read-only. • Non-managed mode – Connection factory properties defined in the JCA WSDL. The Activation/Interaction Spec Properties field is read-only. If this option is deselected, you can override the Activation/Interaction Spec Properties of the WSDL; and in non-managed mode you can override the Connection Factory Properties of the WSDL. At run time the JCA transport interacts with the JCA framework using the JCA WSDL and any activation/interaction spec or connection factory overrides you have provided. For more information on using this option, see "Endpoint Redeployment" in the JCA Transport User Guide.
Connection Mode	This option determines how the service connects to the associated JCA adapter for the EIS system. **Managed** – Recommended for production. The JCA transport connects to the JCA adapter through the JCA adapter-managed connection factory configured in WebLogic Server. For authentication, specify a JNDI service account. If no JNDI service account is specified, an anonymous subject is used. In managed mode, Connection Factory Authentication Properties are read-only. **Non-Managed** – The JCA transport connects to the JCA adapter through the JCA adapter framework, which acts as a container for the JCA adapter. For authentication, specify a JNDI service account. If no JNDI service account is specified, an anonymous subject is used. In non- managed mode, Connection Factory Authentication Properties are editable for overrides. **Note:** If you want to change from Non-Managed mode to Managed mode, be sure to deselect the Overwrite Connection Authentication Properties option before changing modes.

Option	Description
Overwrite Connection Authentication Properties	This option indicates whether or not the user name/password in the adapter connection factory is overwritten by the Connection Authentication Service Account credentials. If no JNDI Service Account is specified, an anonymous subject is used. This option, which is displayed only if the connection factory properties contain user name/password properties, is available in non-managed mode with "Always use configuration from JCA WSDL" deselected.
Connection Authentication Service Account	This field is enabled if "Overwrite Connection Authentication Properties" is enabled and selected. This service account browser displays available service accounts. If the "Overwrite Connection Authentication Properties" option is selected, you must select a service account. For proxy services, only static service accounts are available. This field if displayed only if the connection factory properties contain user name password properties.
Connection Factory Authentication Properties	This field is displayed only if the Connection Factory Properties contain a user name and password. This field is always read-only.
Connection Factory Properties	This field shows the connection factory properties from the WSDL's <jca:address> element. **Development and Testing**: You can override this value if you deselect "Always use configuration from JCA WSDL" and make the Connection Mode Non-Managed. In production environments, use managed mode, which makes the JCA transport connect to the adapter connection factory configured in WebLogic Server.
Operation Name	Displays a read-only name of the selected WSDL operation. An operation can have its own activation/interaction spec properties, shown in the Activation/Interaction Spec Properties field.

Option	Description
Activation/Interaction Spec Properties	"Activation Spec Properties" is the field name for proxy services; "Interaction Spec Properties" is the field name for business services. If this service is an inbound service invoked by an EIS application, this field displays the activation spec properties for the JCA inbound operation shown in Operation Name field. You can override the activation/interaction spec properties if you deselect "Always use configuration from JCA WSDL."
TopLink File Browser	For Oracle Database or Oracle Applications adapters that depend on a database, use this field to select a TopLink mapping file from the file system. The content of the TopLink mapping file is displayed in an editable text box. Generate a TopLink mapping file in Oracle JDeveloper and import it into Oracle Service Bus.

JMS Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the JMS transport protocol. For more information, see the *Oracle Service Bus Interoperability Solutions for JMS*.

Option	To create or edit...
Destination Type	Select a type for the JMS bridge destination: • **Queue** (for point-to-point) • **Topic** (for publish/subscribe)
Message Type	Select one of the following: • **Bytes** (for a stream of uninterpreted bytes) • **Text** (for text messages)
Is Response Required	This option is available only when **Queue** is selected for the **Destination Type**. Select this option to specify that a response is expected after an outbound message is sent.

Option	To create or edit...
Response Correlation Pattern	This option is available only when the **Is Response Required** check box is selected. Select one of the following: • Select **JMSMessageID** for JAX-RPC services running on WebLogic Server 9.2. • Select **JMSCorrelationID** for all other services. When you select this option, you must also enter a **Response URI**, below.

Response URI	This option is available only when **JMSCorrelationID** is selected for the **Response Correlation Pattern**.
	Enter a response URI in the format:
	`jms://host:port/factoryJndiName/destJndiName`
	To target multiple servers, use the following format:
	`jms://host1:port,host2:port/QueueConnectionFactory/DestName`

Response Connection Factory	This option is available only when **JMSMessageID** is selected for the **Response Correlation Pattern**. Enter a response connection factory URI. If a connection factory is not specified, the connection factory for the request is used for the response.
Response JNDI Names	This option is available only when **JMSMessageID** is selected for the **Response Correlation Pattern**. Enter a list of JNDI destination names
Response Timeout	Enter the amount of time to wait for the response, in seconds. This field is required if you selected **Is Response Required**.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in JMS transports, or enter a different character set encoding.
Response Encoding	This option is available only when the **Is Response Required** check box is selected. Accept the default `UTF-8` as the character set encoding for responses in JMS transports, or enter a different character set encoding.

Option	To create or edit...
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. For example, if the business service has a JMS transport protocol, the business service endpoint is an MDB (message-driven bean) JAR file that you can associate with the specific dispatch policy.
Advanced Settings	
Use SSL	Select only if the requests are made over a TLS/SSL connection. TLS/SSL (Secure Sockets Layer) provides secure connections by allowing two applications connecting over a network to authenticate the other's identity and by encrypting the data exchanged between the applications. Authentication allows a server, and optionally a client, to verify the identity of the application on the other end of a network connection. Additionally, if the administrator has restricted access to individual JMS destinations (queues or topics) by setting access control on the JNDI entry for the destination, the Business Service must authenticate when looking up the entry in the JNDI tree with a user name and password.
Expiration	The time interval in milliseconds after which the message will expire. Default value is 0, which means that the message never expires.
Unit of Order	Enter a message unit-of-order. Message unit-of-order enables message producers to group messages into a single unit with respect to the processing order. This single unit-of-order requires that all messages in that unit be processed sequentially in the order they were created.

Option	To create or edit...
Pass Caller's Subject	Select this check box to have Oracle Service Bus pass the authenticated subject when sending a message. When you enable this field and the business service targets JMS resources in a different domain, enable global trust on both domains. See Configuring Security for a WebLogic Domain in *Securing WebLogic Server*.
JMS Service Account	Select a service account to use for the JMS resource managed by the JMS server. A service account is an alias resource for a User ID and its associated password. The same service account is used for both JNDI and JMS purposes. For more information: • "Service Accounts" on page 14-1 • "Adding Service Accounts" on page 14-4

JMS Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the JMS transport protocol. For more information, see the *Oracle Service Bus Interoperability Solutions for JMS*.

Option	To create or edit...
Destination Type	Select one of the following: • **Queue** (for a point-to-point destination type) • **Topic** (for a publish/subscribe destination)
Is Response Required	This option is available only when **Queue** is selected for the **Destination Type**. Select this option to specify that a response is expected after an outbound message is sent.

Option	To create or edit...
Response Correlation Pattern	This option is available only when the **Is Response Required** check box is selected. Select one of the following: • Select **JMSMessageID** for JAX-RPC services running on WebLogic Server 9.2. • Select **JMSCorrelationID** for all other services. When you select this option, you must also enter a **Response URI**, below.

Response URI	This option is available only when **JMSCorrelationID** is selected for the **Response Correlation Pattern**.
	Enter a response URI in the format:
	`jms://host:port/factoryJndiName/destJndiName`
	To target multiple servers, use the following format:
	`jms://host1:port,host2:port/QueueConnectionFactory/DestName`

Response Connection Factory	This option is available only when **JMSMessageID** is selected for the **Response Correlation Pattern**. Enter a response connection factory URI. If a connection factory is not specified, the connection factory for the request is used for the response.
Response Message Type	This option is available only when the **Is Response Required** check box is selected. Select one of the following: • **Bytes** (for a stream of uninterpreted bytes) • **Text** (for text messages)
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in JMS transports, or enter a different character set encoding.
Response Encoding	This option is available only when the **Is Response Required** check box is selected. Accept the default `UTF-8` as the character set encoding for responses in JMS transports, or enter a different character set encoding.

Option	To create or edit...
Client Response Timeout	This option is available only when the **Is Response Required** check box is selected. Enter the number of seconds to wait for a server response before dropping the connection. This only applies if the client is another proxy service in the same domain.
Dispatch Policy	Select a dispatch policy for this endpoint. **Default** signifies the default dispatch policy. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint to process the request. For example, if the proxy service has a JMS transport protocol, the proxy service endpoint is an MDB (message-driven bean) JAR file that you can associate with the specific dispatch policy.
Advanced Settings	
Use SSL	Select only if the requests are made over a TLS/SSL connection. TLS/SSL (Secure Sockets Layer) provides secure connections by allowing two applications connecting over a network to authenticate the other's identity and by encrypting the data exchanged between the applications. Authentication allows a server, and optionally a client, to verify the identity of the application on the other end of a network connection. Additionally, if the administrator has restricted access to individual JMS destinations (queues or topics) by setting access control on the JNDI entry for the destination, the Business Service must authenticate when looking up the entry in the JNDI tree with a user name and password.
Message Selector	Enter a message selector expression. Only messages with properties matching the expression are processed
Durable Subscription	This option is available only if **Topic** is selected for the **Destination Type**. Select this check box if the subscription is durable or leave it blank if the subscription is not durable
Retry Count	Enter the number of delivery retries a message can have before it is moved to the error destination. This field only applies to WebLogic Server JMS destinations.

Option	To create or edit...
Retry Interval	Enter the amount of time, in milliseconds, before rolled back or recovered messages are redelivered. This field only applies to WebLogic Server JMS destinations.
Error Destination	Enter the name of the target destination for messages that have reached their redelivery limit. This field only applies to WebLogic Server JMS destinations.
Expiration Policy	Select an Expiration Policy to use when an expired message is encountered on a destination. This field only applies to WebLogic Server JMS destinations.
Is XA Required	Select this check box if your connection factory is XA. This value is into account when the remote connection factory is unavailable. If your connection factory is available and this value is true, make sure that the connection factory is defined as transactional.
Synchronous Transactional	Select this check box to have the response pipeline inherit the transaction from the request pipeline. This means that the transaction will not commit until both pipelines have completed. This field is enabled only when the **Is Response Required** field is checked. Synchronous Transactional consumes a request thread until the response thread is completed.
JMS Service Account	Select a service account to use for the JMS resource managed by the JMS server. A service account is an alias resource for a User ID and its associated password. The same service account is used for both JNDI and JMS purposes. To learn more about service accounts, see "Service Accounts" on page 14-1.

JPD Transport Configuration page (Business Services)

Use this page to configure transport settings for a business service using the JPD transport protocol. For more information about the JPD transport, see the *JPD Transport User Guide*.

Option	To create or edit...
System Service Account	Enter the service account for the user, or click **Browse** to select one from the list displayed. The system service account specifies the user credentials for the invocation of the WLI stateless session bean that the JPD transport uses to send incoming messages. If no service account is specified, the inbound request subject is used. If there is no inbound request subject, an anonymous subject is used.
Process Service Account	Enter the service account for the user, or click **Browse** to select one from the list displayed. The process service account specifies the user credentials for the invocation of the JPD. If no service account is specified, the inbound request subject is used. If there is no inbound request subject, an anonymous subject is used
Callback Proxy Location	Enter the location of the proxy service for receiving callbacks from the JPD, specified in the following format: `jms://[host:port[,host:port]*]/MyFactory/MyQueue` Where: • `host`: (optional) Host name of the Oracle Service Bus server where the callback proxy service is configured • `port`: (optional) Port number on which Oracle Service Bus is listening for incoming requests • `MyFactory`: JNDI name of the Queue connection factory • `MyQueue`: JNDI name of the Queue where the Oracle Service Bus proxy service is waiting for callback requests In a clustered environment, specify the host name and port numbers of the Managed Servers, separated by a comma (,). The Callback Proxy Location will be ignored if the JPD being invoked is a synchronous JPD or an asynchronous JPD without callbacks.

Option	To create or edit...
Dispatch Policy	Select the default dispatch policy or a configured Work Manager instance for the outbound response. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Propagate Transaction	Select the check box to propagate the existing transaction within Oracle Service Bus to the JPD.

MQ Transport Configuration page (Business Services)

Before you use the MQ transport, you must configure a MQ Connection resource. See Working with MQ Connections. For more information about the MQ transport, see the *MQ Transport User Guide*.

Use this page to configure transport settings for a proxy service using the native MQ transport protocol.

Option	To create or edit...
Message Type	This option is available only when the **Is Response Required** check box is selected. Select one of the following: • **Bytes** (for a stream of uninterpreted bytes) • **Text** (for text messages)
Is Response Required	Select this option to specify that a response is expected after an outbound message is sent.
Polling Interval	This option is available only when the **Is Response Required** check box is selected. Enter a polling interval, in milliseconds. The default is `1000`.
Response Correlation Pattern	This option is available only when the **Is Response Required** check box is selected. Specify whether the response correlation pattern should be based on **MessageID** or **CorrelationID**.

Option	To create or edit...
Auto-generate Correlation Value	Select this check box to automatically generate a CorrelationID or MessageID.
MQ Response URI	This option is available only when the **Is Response Required** check box is selected. The destination to which the response should be published. Enter a response URI in the same format as the endpoint URI: `mq://<local-queue-name>?conn=<mq-connection-resource-ref>` For more detailed information, see the *MQ Transport User Guide*.
Response Timeout	This option is available only when the **Is Response Required** check box is selected. Enter the number of seconds to wait for a response before dropping the connection.
Dispatch Policy	This option is available only when the **Is Response Required** check box is selected. Select a dispatch policy for this endpoint. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint. For information about work managers, see Using Work Managers to Optimize Scheduled Work and Create Work Manager in the WebLogic Server Administration Console Online Help.

Option	To create or edit...
Endpoint URI 'PUT' options	Enter the MQ PUT message options from among the following: • MQC.MQPMO_ALTERNATE_USER_AUTHORITY • MQC.MQPMO_DEFAULT_CONTEXT • MQC.MQPMO_FAIL_IF_QUIESCING • MQC.MQPMO_LOGICAL_ORDER • MQC.MQPMO_NEW_CORREL_ID • MQC.MQPMO_NEW_MSG_ID • MQC.MQPMO_NO_CONTEXT • MQC.MQPMO_NO_SYNCPOINT • MQC.MQPMO_NONE • MQC.MQPMO_PASS_ALL_CONTEXT • MQC.MQPMO_PASS_IDENTITY_CONTEXT • MQC.MQPMO_RESOLVE_LOCAL_Q • MQC.MQPMO_SET_ALL_CONTEXT • MQC.MQPMO_SET_IDENTITY_CONTEXT • MQC.MQPMO_SYNCPOINT • MQC.MQPMO_VERSION_1 • MQC.MQPMO_VERSION_2 You can use either "|" or "+" to separate multiple options. For example, you can specify the following: MQC.MQPMO_LOGICAL_ORDER | MQC.MQPMO_NEW_MSG_ID The MQ PUT message options are applied when the message is placed in the outbound queue.
MQ Unrecognized Response URI	Enter the URI representing the queue to which unrecognized response messages should be sent. Note that this setting is enabled only when the **Auto-generate Correlation Value** check box is selected. If you do not specify a value for this field, unrecognized response messages are deleted.
Process RFH2 Headers	Select this option to parse WebSphere MQ RFH2 headers from a message payload and automatically generate an RFH2Headers transport header containing the RFH2 data. If you do not select this option, the payload is passed through as received.

MQ Transport Configuration page (Proxy Services)

Before you use the MQ transport, you must configure a MQ Connection resource. See Working with MQ Connections. For more information about the MQ transport, see the *MQ Transport User Guide*.

Use this page to configure transport settings for a proxy service using the native MQ transport protocol.

Option	To create or edit...
Polling Interval	Enter a polling interval, in milliseconds. The default is `1000`.
Is Response Required	Select this option to specify that a response is expected after an outbound message is sent.
Response Correlation Pattern	This option is available only when the **Is Response Required** check box is selected. Specify whether the response correlation pattern should be based on **MessageID** or **CorrelationID**.
MQ Response URI	This option is available only when the **Is Response Required** check box is selected. The destination to which the response should be published. Enter a response URI in the same format as the endpoint URI: `mq://<local-queue-name>?conn=<mq-connection-resource-ref>` For more detailed information, see the *MQ Transport User Guide*.
Response Message Type	This option is available only when the **Is Response Required** check box is selected. Select one of the following: • **Bytes** (for a stream of uninterpreted bytes) • **Text** (for text messages)
Client Response Timeout	This option is available only when the **Is Response Required** check box is selected. Enter the number of seconds to wait for a response before dropping the connection.

Option	To create or edit...
Dispatch Policy	This option is available only when the **Is Response Required** check box is selected. Select a dispatch policy for this endpoint. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint. For information about work managers, see Using Work Managers to Optimize Scheduled Work and Create Work Manager in the WebLogic Server Administration Console Online Help.
Backout Threshold	Enter a value representing the number of times the pipeline should retry a message before redirecting the message to the queue specified in the **Dead Letter URI** field. If you do not specify a value for this field, the message is redirected to the dead letter queue without attempting any retries.
MQ Dead Letter URI	Enter the URI of the dead letter queue to which request messages should be redirected after attempting the number of retries specified in the **Backout Threshold** field. If you do not specify a value for this field, the message is discarded after retrying the number of times specified in the **Backout Threshold** field. The Dead Letter URI uses the same format as the EndPoint URI.

Option	To create or edit...
Endpoint URI 'GET' options	Enter the MQ GET message options from among the following: • MQC.MQGMO_ACCEPT_TRUNCATED_MSG • MQC.MQGMO_ALL_MSGS_AVAILABLE • MQC.MQGMO_BROWSE_FIRST • MQC.MQGMO_BROWSE_NEXT • MQC.MQGMO_COMPLETE_MSG • MQC.MQGMO_CONVERT • MQC.MQGMO_FAIL_IF_QUIESCING • MQC.MQGMO_LOCK • MQC.MQGMO_LOGICAL_ORDER • MQC.MQGMO_MARK_BROWSE_CO_OP • MQC.MQGMO_MARK_SKIP_BACKOUT • MQC.MQGMO_NO_SYNCPOINT • MQC.MQGMO_NONE • MQC.MQGMO_NO_WAIT • MQC.MQGMO_SYNCPOINT • MQC.MQGMO_SYNCPOINT_IF_PERSISTENT • MQC.MQGMO_UNLOCK • MQC.MQGMO_UNMARK_BROWSE_CO_OP • MQC.MQGMO_UNMARK_BROWSE_HANDLE • MQC.MQGMO_UNMARKED_BROWSE_MSG • MQC.MQGMO_VERSION_1 • MQC.MQGMO_VERSION_2 • MQC.MQGMO_VERSION_3 • MQC.MQGMO_WAIT You can use either "|" or "+" to separate multiple options. For example, you can specify the following: MQC.MQGMO_ACCEPT_TRUNCATED_MSG | MQC.MQGMO_LOCK The MQ GET message options are applied when reading a message from the inbound queue.

Option	To create or edit...
Process RFH2 Headers	Select this option to parse WebSphere MQ RFH2 headers from a message payload and automatically generate an RFH2Headers transport header containing the RFH2 data. If you do not select this option, the payload is passed through as received.

SB Transport Configuration page (Business Services)

Use this page to configure transport settings for a proxy service using the SB (Service Bus) transport protocol. For more information about the SB transport, see the *SB Transport User Guide*.

Option	To create or edit...
Dispatch policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Timeout	The amount of time in seconds it takes the service to time out. **Note:** The timeout will be ignored when the quality of service is Exactly-Once.
Service Account	• Click **Browse** and select a service account from the list displayed. If no service account is specified, an anonymous subject is used.

SB Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the SB (Service Bus) transport protocol. For more information about the SB transport, see the *SB Transport User Guide*.

Option	To create or edit...
Dispatch Policy	Select a dispatch policy for this endpoint or use the default dispatch policy. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint to process the request. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Use SSL	When specified, requests must be sent over an SSL connection. However, unsecured connections are not forbidden. The administrator must close all unsecured protocols on the server (for example, t3 or http) to strictly enforce secured client connections.

SFTP Transport Configuration page (Business Services)

Use this page to configure transport settings for a proxy service using the sftp transport protocol. For more information about the SFTP transport, see the *SFTP Transport User Guide*.

Option	To create or edit...
User Authentication	Select one of the following: • **Username Password Authentication** - Specifies that a static service account is associated with this authentication method and the client is authenticated using the provided credentials. • **Host Based Authentication** - Specifies that a user name and service key provider is required to use this authentication method. Any user connecting from a known host is authenticated using the private key of the host. • **Public Key Authentication** - Specifies that a user name and service key provider is required to use this authentication method. Every user has their own private key.
Service Account	Enter the service account for the user, or click **Browse** to select service accounts from a browser.

Option	To create or edit...
Service Key Provider	This option is available only when **Host Based** or **Public Key Authentication** is selected. Enter a service key provider in the **Service Key Provider** field. You can click **Browse** to select service key providers from a browser. This is a required field.
Username	This option is available only when **Host Based** or **Public Key Authentication** is selected. Enter the user name.
Timeout	Enter the socket timeout interval, in seconds, before the connection is dropped. If you enter `0`, there is no timeout. The default value is 60.
Prefix for destination File Name	Enter the prefix for the file name under which the file is stored on the remote server.
Suffix for destination File Name	Enter the suffix for the file name under which the file is stored on the remote server.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in SFTP transports.

SFTP Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the sftp transport protocol. For more information about the SFTP transport, see the *SFTP Transport User Guide*.

Option	To create or edit...
User Authentication	Select one of the following: • **Username Password Authentication** - Specifies that a static service account is associated with this authentication method and the client is authenticated using the provided credentials. • **Host Based Authentication** - Specifies that a user name and service key provider is required to use this authentication method. Any user connecting from a known host is authenticated using the private key of the host. • **Public Key Authentication** - Specifies that a user name and service key provider is required to use this authentication method. Every user has their own private key.
Service Account	Enter the service account for the user, or click **Browse** to select service accounts from a browser.
Service Key Provider	This option is available only when **Host Based** or **Public Key Authentication** is selected. Enter a service key provider in the **Service Key Provider** field. You can click **Browse** to select service key providers from a browser. This is a required field.
Username	This option is available only when **Host Based** or **Public Key Authentication** is selected. Enter the user name.
Pass By Reference	Select this check box to stage the file in the archive directory and pass it as a reference in the headers.
Remote Streaming	Select this check box to stream the SFTP files directly from the remote server at the time of processing. When you select this option, the archive directory is the remote directory on the remote SFTP server machine. Therefore, you should specify the archive directory as relative to the SFTP user directory.
File Mask	Enter the regular expression for the files to be picked. The default is `*.*`.
Managed Server	This field is available only in a clustered domain. Select the Managed Server to act as the polling server. All of the Managed Servers can process the message, but only one can poll for the message.
Polling Interval	Enter the interval in seconds at which the file is polled from the specified location. The default is `60`.

Option	To create or edit...
Read Limit	Specify the maximum number of messages to read per polling sweep. Enter `0` to specify no limit. The default is `10`.
Post Read Action	Select what happens to a message after it has been read. • **Archive** - The message is archived. • **Delete** - The message is deleted.
Archive Directory	Specify the path to the archive location if the **Post Read Action** option is set to **Archive**. This field is required if the **Pass By Reference** option is selected. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Download Directory	Enter the directory on your local machine where files are downloaded during the file transfer. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Error Directory	Enter the location where messages are posted if there is a problem. **Note:** The Archive, Download, and Error directories are absolute paths, and they are automatically created. If you specify a relative path, the files are created relative to the Java process that starts the WebLogic Server.
Request Encoding	Accept the default `UTF-8` as the character set encoding for requests in SFTP transports.
Advanced Settings	
Scan Subdirectories	Select this check box to recursively scan all directories
Sort By Arrival	Select this check box to deliver events in the order of arrival.
Timeout	Enter the socket timeout interval, in seconds, before the connection is dropped. If you enter `0`, there is no timeout. The default value is 60.
Retry Count	Specify the number of retries for SFTP connection failures.

Tuxedo Transport Configuration page (Business Services)

Use this page to configure transport settings for a proxy service using the Tuxedo transport protocol. For more information about the Tuxedo transport, see *Oracle Service Bus Interoperability Solution for Tuxedo*.

Option	To create or edit...
Field Table Classes	Enter the name of the class or classes describing the `FML`/`FML32` buffer received. These are used for the `FML`/`FML32-to-XML` conversion routines to map field names to element names. This is a space separated list of fully qualified class names.
View Classes	Enter the name of the class or classes describing the `VIEW`/`VIEW32` buffer received or sent. These are used for the `VIEW`-to-`XML` or `VIEW32`-to-`XML` conversion routines to map field names to element names. This is a space separated list of fully qualified class names.
Classes Jar	Select a JAR Resource that contains a JAR file with the `FML`/`FML32` or `VIEW`/`VIEW32` classes necessary for this endpoint operation.
Remote Access Point(s)	Select a remote access point from the drop down list that is associated with the Import. The list contains remote access points configured in WTC. A business service cannot be created if there is no associated remote access point. If no remote access points exist or to create a new one, select **New**. Enter the corresponding **Access Point Name** and **Network Address** in the adjacent fields. Upon validation of the endpoint, the access point is added to the WTC configuration for each WTC server. If no WTC server exists, one is created. If more than one URI has been specified, there will be one remote access point field per URI and the URI displays for informative purposes. If more than one URI exists, each requires a different remote access point. If the URI specified already corresponds to an existing WTC resource, the corresponding remote access point displays, but cannot be modified.
Local Access Point(s)	This field appears only when you select **New** in the **Remote Access Point** field. From the drop-down list, select a local access point to be associated with the newly created remote access point. If none exist or to create a new one, select **New**. Enter the corresponding **Local Access Point Name** and **Local Network Address** in the adjacent fields.

Option	To create or edit...
Request Buffer Type	Select the type of buffer that the remote Tuxedo service will receive.
Request Buffer Subtype	This option is enabled if the previous **Request Buffer Type** value is VIEW or VIEW32. Enter the buffer subtype with which to associate the request buffer.
Response Required?	Select this check box to indicate a bidirectional call. If not checked, the underlying `tpcall` is invoked with `TPNOREPLY` flag, and a null response is posted asynchronously.
Suspend Transaction?	Select this check box to suspend the transaction, if it exists. This is useful when the remote service does not support transactions.
Dispatch Policy	Select the instance of WebLogic Server Work Manager that you want to use for the dispatch policy for this endpoint. The default Work Manager is used if no other Work Manager exists. This Work Manager is used to asynchronously post a null reply in the case of a one-way invocation. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Request Encoding	Specify a character set encoding for requests in Tuxedo transports.
Response Encoding	Specify a character set encoding for responses in Tuxedo transports.
Timeout	Specify the maximum amount of time (in seconds) that the transport run time waits for replies; an integer value that is greater than or equal to 0. If not specified or set to zero (default), replies will time out at BLOCKTIME, the maximum number of seconds that the local WTC access point allows for a blocking call.
Transformation Style	Select one of the following: • **None** - (default) The order of fields may not be respected. • **Ordered** - The fields are presented with all their occurrences in the correct order. • **Ordered and Grouped** - If the fields are logically structured as records, the fields are ordered by occurrence and grouped by record.

Tuxedo Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the Tuxedo transport protocol. For more information about the Tuxedo transport, see *Oracle Service Bus Interoperability Solution for Tuxedo*.

Option	To create or edit...
Field Table Classes	Enter the name of the class or classes describing the `FML/FML32` buffer received. These are used for the `FML/FML32-to-XML` conversion routines to map field names to element names. This is a space separated list of fully qualified class names.
View Classes	Enter the name of the class or classes describing the `VIEW/VIEW32` buffer received or sent. These are used for the `VIEW-to-XML` or `VIEW32-to-XML` conversion routines to map field names to element names. This is a space separated list of fully qualified class names. `X_C_TYPE` and `X_COMMON` Tuxedo buffer types are handled in the same manner as `VIEW/VIEW32` buffers. If an incoming request contains a `VIEW`, then the corresponding `VIEW` class should be specified in the Oracle Service Bus `CLASSPATH`.
Classes Jar	Select a JAR resource that contains a JAR file with the `FML/FML32` or `VIEW/VIEW32` classes necessary for this endpoint operation.
Local Access Point	Select a local access point from the drop-down list that is associated with the export. The drop-down list contains local access points configured in WTC. A proxy service cannot be created if there is not an associated local access point. If no local access points exist or to create a new one, select **New**. Enter the corresponding **Local Access Point Name** and **Local Network Address** in the adjacent fields. Upon validation of the endpoint, the access point is added to the WTC configuration for each WTC server. If no WTC server exists, one is created. You can enter an existing access point name after selecting the **New** option. This causes the existing information to be updated with the new parameters. You can change only the host name and port number.

Option	To create or edit...
Remote Access Point	This field appears only when you select **New** in the **Local Access Point** field. From the drop-down list, select a remote access point to be associated with the newly created local access point. If none exist or to create a new one, select **New**. Enter the corresponding **Access Point Name** and **Network Address** in the adjacent fields. You can enter an existing access point name after selecting the **New** option. This causes the existing information to be updated with the new parameters. You can change only the host name and port number. The remote access point will also be the authentication principal for the WTC connection for inbound requests. Optionally, you can create a user with the same access point ID in the default security realm to allow incoming calls. To do so, select Yes from the **Create User?** drop-down list. The password will be randomly generated using a temporary variable to avoid security issues.
Reply Buffer Type	This option is available only if the **Response Required?** field is selected. Select the type of buffer that the remote Tuxedo client will receive.
Reply Buffer Subtype	This option is available only when the **Response Required?** option is selected and the **Reply Buffer Type** value is **VIEW** or **VIEW32**. Enter the buffer subtype with which to associate the reply buffer.
Response Required?	Select this check box if this service is expected to send a response. The default status is that this option is selected. This option is cleared and the unavailable if the service type is **Messaging Service** and the response message type is **None**.
Request Encoding	Specify a character set encoding for requests in Tuxedo transports.
Response Encoding	Specify a character set encoding for responses in Tuxedo transports.
Transformation Style	Select one of the following: • **None** - (default) The order of fields may not be respected. • **Ordered** - The fields are presented with all their occurrences in the correct order. • **Ordered and Grouped** - If the fields are logically structured as records, the fields are ordered by occurrence and grouped by record.

WS Transport Configuration page (Business Services)

Use this page to configure transport settings for a proxy service using the WS transport protocol. For more information about the WS transport, see the *WS Transport User Guide*.

Option	To create or edit...
Response Timeout	Enter the number of seconds to wait for a response. Leaving this field blank indicates that there is no response timeout. The business service will wait for the duration of the sequence timeout configured in the RM policy. If you enter a zero (`0`) value, there is no timeout; as such, it will never time out.
Service Account	Enter a service account or click **Browse** to select one from the list displayed. The service account specifies the credentials to use when there is an HTTP basic authentication policy on this service.
Advanced Settings	
Queue Error Messages	Select the check box to enable sending error messages to the configured error queue.
Error Queue URI	This option is available only when the **Queue Error Messages** check box is selected. Enter the URI of JMS queue for storing error messages, in the following format: `jms://host:port/connFactoryJndiName/queueJndiName`
JMS Error Queue Service Account	This option is available only when the **Queue Error Messages** check box is selected. Enter a service account or click **Browse** to select one from the list displayed. The service account specifies the credentials to use for JNDI lookups and sending error messages to the error queue.
Use SSL for Error Queue	This option is available only when the **Queue Error Messages** check box is selected. Select the check box to use SSL for connecting to the error queue.

WS Transport Configuration page (Proxy Services)

Use this page to configure transport settings for a proxy service using the WS transport protocol. For more information about the WS transport, see the *WS Transport User Guide*.

Option	To create or edit...
Dispatch Policy	Select a dispatch policy for this endpoint or use the default dispatch policy. Dispatch policy refers to the instance of WLS Work Manager that you want to use for the service endpoint. For information about Work Managers, see Using Work Managers to Optimize Scheduled Work and Create Global Work Managers in the WebLogic Server *Administration Console Online Help*.
Advanced Settings	
Retry Count	The number of times to retry delivery of a message to the pipeline. If an unhandled exception occurs in the request pipeline of a proxy service, the incoming WS transport message will be redelivered to the pipeline up to the number of times specified by the retry count. This value is important for reliably processing WS transport messages.
Retry Delay	The number of seconds the system pauses before retrying to send a message to the pipeline after an error.
Synchronous	Select this check box to have the response pipeline inherit the transaction from the request pipeline. This means that the transaction will not commit until both pipelines have completed. Synchronous consumes a request thread until the response thread is completed.