An Oracle White Paper
May 2011

Oracle BEA WebLogic – Oracle Enterprise Gateway Integration Guide

Disclaimer

1. Introduction 4
Purpose.. 4
JMS Architecture.. 4
Prerequisites 5
Configuration Steps 5
2. WebLogic Details 5
WebLogic Connection Settings.. 5
Creating Jar File for Client Application.................................. 5
3. Setting up the OEG Environment.. 6
Upload WebLogic Class Provider to OEG Gateway 6
4. Configure the Gateway to Place Messages on WebLogic Queue 6
Creating a JMS Session: 6
Create a 'WebLogic' Policy 7
Ensure policies are updated on the Gateway 11
Using OEG Service Explorer to Test the Integration 12
5. Configuring the Gateway to read from the WebLogic Queue 12
Create Policy that will be invoked with message read from WebLogic 12
Creating a JMS Session: 14
Ensure policies are updated on the Gateway 15
Testing to read messages from a queue 15
6. Conclusion 18

1. Introduction

Purpose

This document describes how to configure the Gateway to perform protocol translation. This will be demonstrated by the following:

1. The Gateway will listen for messages on a HTTP interface. Messages read from this interface will be placed on a message queue.
2. The Gateway will listen for messages on a message queue. Messages read from the queue will be sent to an email account via SMTP.

This guide applies to OEG software products, from version 6.x upwards.
In this guide the message system that will be used is **Oracle BEA WebLogic 10.3.**

JMS Architecture

The Gateway utilises JMS (Java Message Service) for sending and receiving messages from messaging systems. JMS API which was developed by Sun defines a common set of interfaces and associated semantics that allow the Gateway to communicate with various messaging applications in a standard way.

Messaging system products (IBM WebSphere MQ, JBossMQ, SonicMQ, Fiorano, WebLogic and OpenJMS) provide implementations of JMS which can be plugged into the Gateway.

The Gateway has been designed to allow 3rd party JMS providers to be "plugged in". To plug in new JMS providers, you must install the JMS provider on the Gateway machine. The messaging system vendors can provide an implementation of the JMS provider which is normally in the form of jar files and configuration settings to be entered in the OEG Policy Studio.

Prerequisites

- Oracle BEA WebLogic available from http://www.oracle.com
- OEG Gateway Software

Configuration Steps

1. Install OEG Gateway.
2. Configure Gateway to send messages to WebLogic.
3. Configure Gateway to listen for messages from WebLogic queue.
4. Test the configuration.

2. WebLogic Details

The WebLogic instance used for purpose of this tutorial is using default installation and configuration options. Please refer to WebLogic documentation for any WebLogic specific settings and configuration options.

WebLogic Connection Settings

Default Queue: jms/testQueue (configuration could differ depending on custom configuration)
Initial Context Factory: weblogic.jndi.WLInitialContextFactory
Connection Factory: weblogic.jms.ConnectionFactory
Provider URL: t3://WebLogic_host_ip:7001

Creating Jar File for Client Application

The Gateway requires the WebLogic jar file to provide the necessary classes at runtime. The necessary jar file is created from the WebLogic server with the following commands.
Creating a wlfullclient.jar for JDK 1.6 client applications

1. Change directories to the server/lib directory.

```
cd WebLogic_Home/server/lib
```

2. Use the following command to create wlfullclient.jar in the server/lib directory:
   ```
   java -jar wljarbuilder.jar
   ```
3. The file can now be copied and bundled the with client applications.
4. Add the wlfullclient.jar to the client application's classpath. See section 3

3. Setting up the OEG Environment

Upload WebLogic Class Provider to OEG Gateway

Oracle BEA WebLogic provides a particular JMS provider that the Gateway will use to connect to WebLogic. The JMS provider takes the form of Java archive files (i.e. JAR files). Once WebLogic is installed it is a simple matter to drop the JMS provider JAR files onto the OEG Gateway.

Instructions for Software install :

Copy the wlfullclient.jar file to the vordel_product_dir/ext/lib directory

- wlfullclient.jar

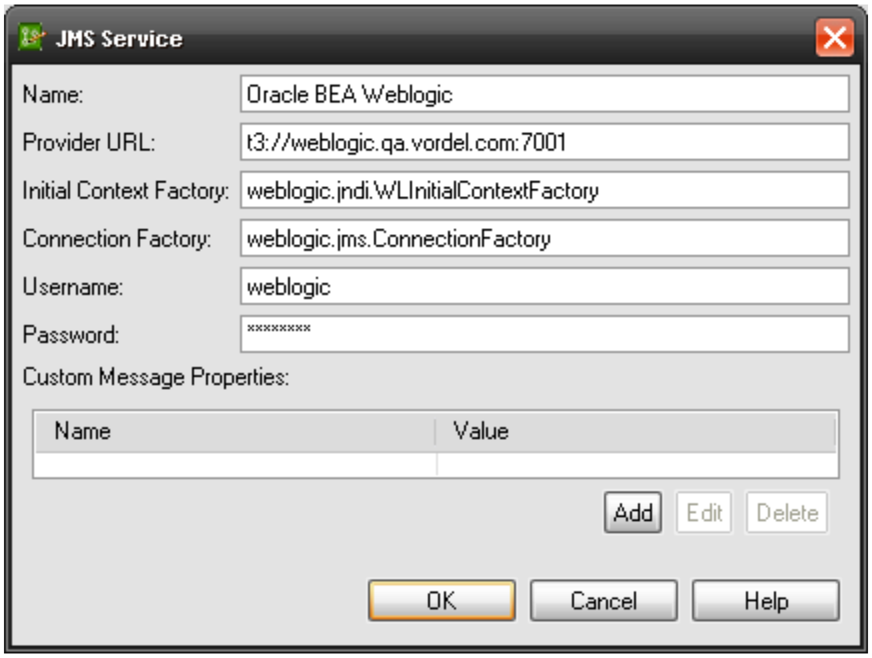
4. Configure the Gateway to Place Messages on WebLogic Queue

The gateway will be configured to place messages it receives on a queue (i.e. destination) named 'jms/testQueue' in WebLogic.

Creating a JMS Session:

1. Start the **Gateway** and **Policy Studio** (for more details refer to Getting Started in the Help Configuration Guide.)
2. Click on the **External Connections** navigation panel in Policy Studio
3. Right click on **JMS Services** and click on **Add a JMS Service**
4. Configure the following fields for the **JMS Service**:
   - **Name:** Oracle BEA WebLogic
   - **Provider URL:** t3://WebLogic_host_ip:7001(where IP address points to WebLogic has been installed)
   - **Initial Context Factory:** weblogic.jndi.WLInitialContextFactory
   - **Connection Factory:** weblogic.jms.ConnectionFactory
   - **Username:** The value here is configured during WebLogic install

• **Password:** The value here is configured during WebLogic install


JMS Service

Name:	Oracle BEA Weblogic
Provider URL:	t3://weblogic.qa.vordel.com:7001
Initial Context Factory:	weblogic.jndi.WLInitialContextFactory
Connection Factory:	weblogic.jms.ConnectionFactory
Username:	weblogic
Password:	********

Custom Message Properties:

Name	Value

Create a 'WebLogic' Policy

Create a policy to route messages on to the WebLogic queue by completing the following steps:

- Open **Policy Studio** and select **Policies** navigation pane. Create a new Policy titled "WebLogic" by right clicking on **Policies** and select **Add Policy**.
- Create a new relative path on the Gateway Process called **/jms** by selecting **Services** navigation button, then expand **OEG Gateway** and right click on **Default Services** to **Add Relative Path.**
- Map the **/jms** path to the policy called **Weblogic** with a tick. This means that when a message is received by the Gateway on the path **/jms**, it will be passed to the **Weblogic** policy, which will then process the message.

Configuring the Messaging System Filter

When a policy that routes to a JMS provider (such as WebLogic) is created, the policy must contain a **Messaging System** filter, which can be found under the **Routing** filter category in Policy Studio. To configure this filter, complete the following steps:

1. Go back to the "Weblogic" policy under Policies. (the TAB should still be open).
2. Drag a **Messaging System** from the **Routing** filter category onto the policy canvas
3. Under the **Request** tab select the "Oracle BEA Weblogic" **JMS Service** that has been configured above from the **JMS Session** dropdown.
4. Set the **Destination** to 'jms/testQueue', which was configured during the WebLogic destination configuration.
5. The **Message Type** should be specified. For example, in **Message Type**, **select Use content.body attribute' to create a message in the format specified in the 'SOAP over JAVA Message Service'**.
6. All other settings may be left as default.
7. Click on the **Response tab** and select an option for the Response. For this tutorial 'No Response" has been selected'. Please see Note 2 on Response settings below
8. Click on **Finish**.

Note 1 : Message Type Settings:
Here is an explanation of how the various serializations (from Vordel message to JMS message) work.

- Use content.body attribute to create a message in the format specified in the "SOAP over Java Messaging Service" proposal:
If this option is selected, messages will be formatted according to the SOAP over JMS proposal. This is the default option since, in most cases; it is the message body that is to be routed to the messaging system. This will result in a ByteMessage being sent to the queue/topic and JMS a property will contain the Content-Type (i.e. text/xml)

- Create a MapMessage from the java.lang.Map in the attribute named below:
Select this option to create a javax.jms.MapMessage from the Vordel message attribute named below that consists of name-value pairs.

- Create a ByteMessage from the attribute named below:
Select this option to create a javax.jms.ByteMessage from the Vordel message attribute named below.

- Create an ObjectMessage from the java.lang.Serializable in the attribute named below:
This option can be selected in order to create a javax.jms.ObjectMessage from the Vordel message attribute named below.

- Create a TextMessage from the attribute named below:
A javax.jms.TextMessage can be created from the message attribute named below by selecting this option from the dropdown.

Note 2: Response settings

The **Response** tab is used to configure whether the Gateway should use asynchronous or synchronous communication when talking to the messaging system. If the Gateway is to use asynchronous communication then select the "No response" radio button. If synchronous communication is required, you can select to read the response from either a temporary queue or from a named queue or topic.

When you elect to use synchronous communication, the Gateway will wait on a message from a queue/topic from the messaging system. The Gateway will set the "JMSReplyTo" property on each message that it sends. The value of the "JMSReplyTo" property will be the queue, temporary queue, topic, or temporary topic that was selected in the **Response Type** dropdown. It is up to the application that consumes the message from the queue to send the message back to the destination specified in "JMSReplyTo".

The Gateway will set the "JMSCorrelationID" property to the value of the **Correlation ID** field on the **Response** tab to correlate requests messages to their corresponding response messages. If the user has selected to use a temporary queue or temporary topic then this will be created when the Gateway is started up.

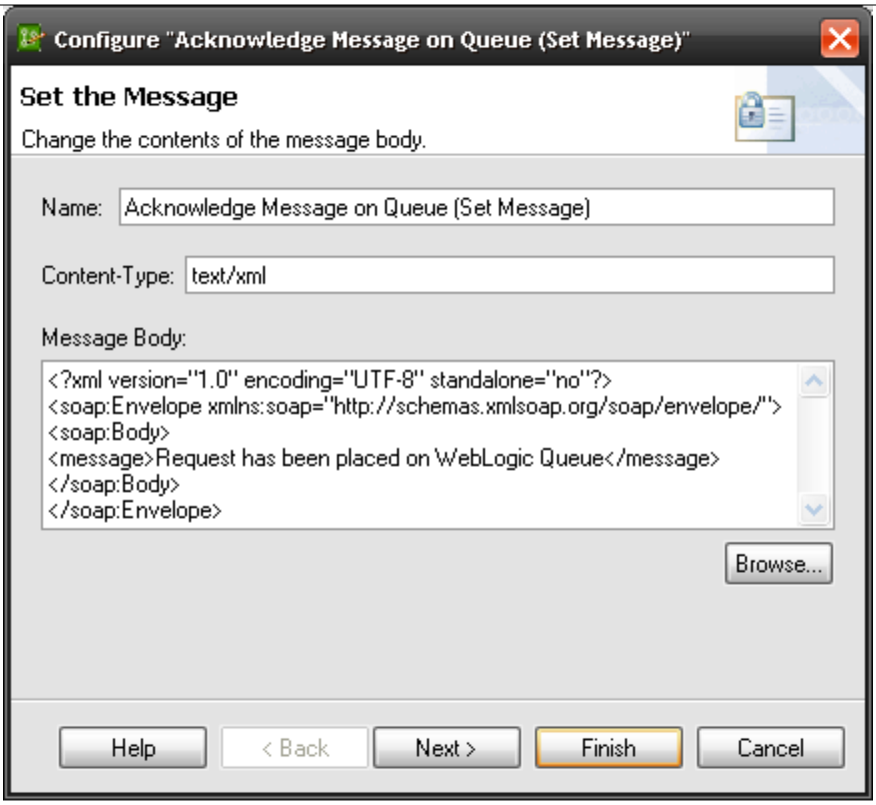
Response Type:
Select where the response message is to be placed by selecting one of the following options:

- *No Response:*
  Select this option if you do not expect or do not care about receiving a response from the JMS provider.
- *Response from Temporary Queue:*
  Select this option to instruct the JMS provider to place the response message on a temporary queue. In this case, the temporary queue will be created on start-up of the Gateway. Only the Gateway will be able to read from the temporary queue, however any application can write to it. The Gateway will use the value of the "JMSReplyTo" header to indicate the location where it will read responses from.
- *Take Response from Queue or Topic:*
  If you want the JMS provider to place response messages on a named queue or topic, select this option and enter the name of the queue or topic in the **Destination** field below.

To complete the test policy create the following flow:

1. **Messaging System Filter**: This filter should be configured as described above. This is a mandatory filter in the policy.
2. **Set Message Filter**: Used to set the content of an XML response message that can be returned to the client to acknowledge that the message has been placed on the Open JMS queue. This step is not mandatory, but is useful for informational purposes. This filter can be found in the 'Conversion' filter category.

How the Set Message filter has been configured:



Set the Message

Change the contents of the message body.

Name: Acknowledge Message on Queue (Set Message)

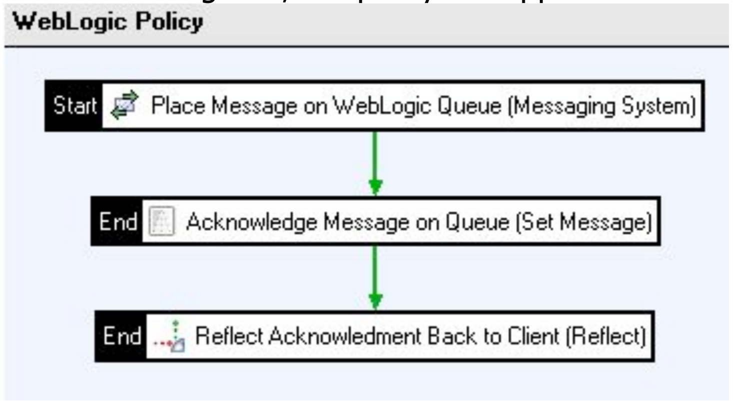
Content-Type: text/xml

Message Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
<message>Request has been placed on WebLogic Queue</message>
</soap:Body>
</soap:Envelope>
```

3. Reflect Filter: Routes the customized response back to the client . The Reflect filter can be found under the 'Utility' filter category.

Once configured, the policy will appear as follows:


WebLogic Policy
Start
Place Message on WebLogic Queue (Messaging System)
End
Acknowledge Message on Queue (Set Message)
End
Reflect Acknowledment Back to Client (Reflect)

Having configured the JMS Session and the 'WebLogic' policy, we will be able to place messages on the WebLogic 'jms/testQueue' queue.

Ensure policies are updated on the Gateway

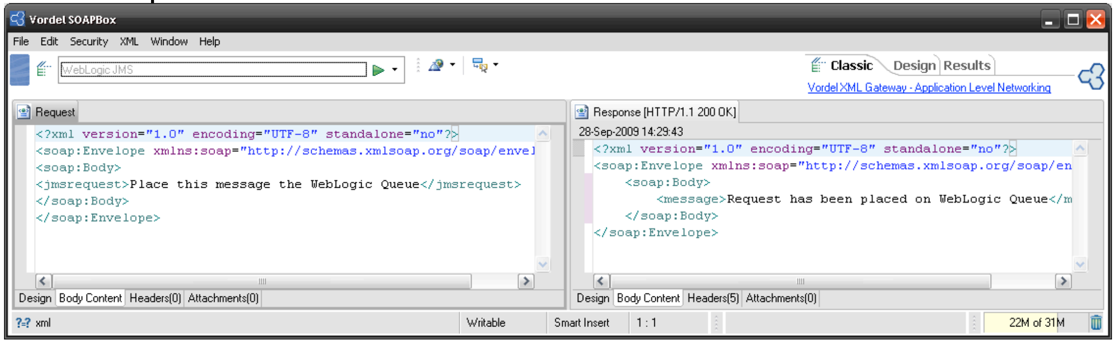
- Open the **Policy Studio**.
- Click on **Settings**.
- Select **Deploy** to ensure that the changes made are propagated to the running instance of the Gateway.
- Click **Yes** to deploy

Using OEG Service Explorer to Test the Integration

OEG Service Explorer will be used to test the configuration.

- Load a sample XML message into OEG Service Explorer.
- Ensure that the URL field in OEG Service Explorer points to the Gateway and configured path: http://xml_gateway_ip:8080/jms

The screenshot below shows a sample SOAP Request loaded in the OEG Service Explorer:


Vordel SOAPBox
File Edit Security XML Window Help
WebLogic JMS
Classic Design Results
Vordel XML Gateway - Application Level Networking
Request
Response [HTTP/1.1 200 OK]
28-Sep-2009 14:29:43
Design Body Content Headers(0) Attachments(0)
Design Body Content Headers(5) Attachments(0)
?-? xml Writable Smart Insert 1 : 1 22M of 31M

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envel
<soap:Body>
<jmsrequest>Place this message the WebLogic Queue</jmsrequest>
</soap:Body>
</soap:Envelope>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/en
    <soap:Body>
        <message>Request has been placed on WebLogic Queue</m
    </soap:Body>
</soap:Envelope>
```

5. Configuring the Gateway to read from the WebLogic Queue

The Gateway will be configured to read the messages from a WebLogic queue. When the Gateway consumes a message from a JMS queue, it can then be processed by a policy for handling of the consumed message. Below is an example of how a policy is configured to be called when the Gateway consumes the message off the WebLogic JMS queue.

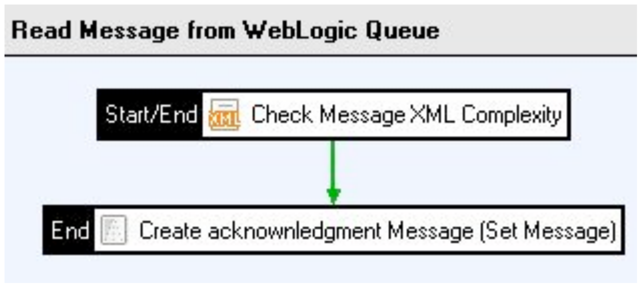
Create Policy that will be invoked with message read from WebLogic

A test policy will be created to check the message's XML complexity and to convert the message taken from the queue to an acknowledgment message that is sent back to the client. Of course the policy can be as complex as desired and can be configured with any of the filters in the filter categories.

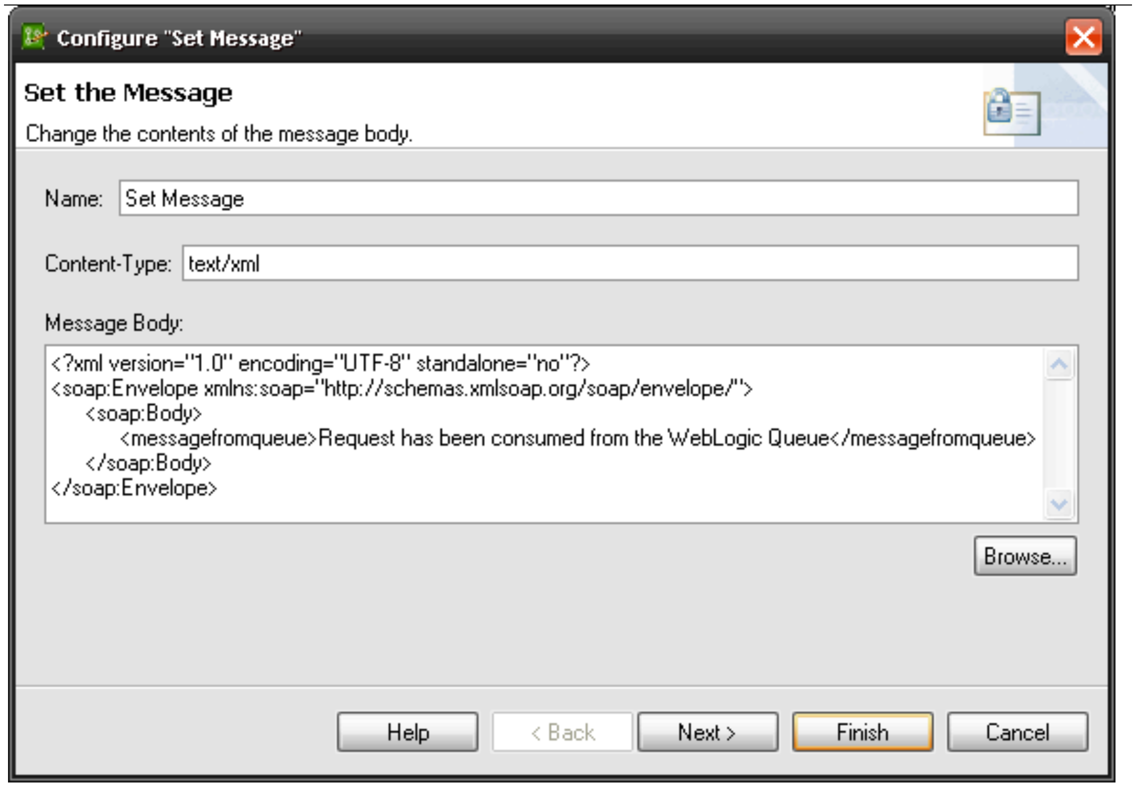
To create policy that the JMS Consumer (Part of JMS Session in Gateway) will invoke when the message has been consumed:

1. Right click on **Policies**.
2. Click **Add Policy** and create a new policy titled **Read from WebLogic Queue**.
3. Click on the newly created Policy.
4. Drag a **XML Complexity** filter from the **Content Filtering** category onto the canvas.
5. Drag a **Set Message** filter from the palette on the right of the screen. The **Set Message** filter is located under the **Conversion** group.
6. Configure the **Set Message** filter with an acknowledgment message (as shown below).
7. Connect the **XML Complexity** filter to the **Set Message** filter via a success path
8. Right click on the **XML Complexity** filter and **Set as Start**

The Policy that will be called by the JMS consumer:


Read Message from WebLogic Queue
Start/End
Check Message XML Complexity
End
Create acknownledgment Message (Set Message)

The **Set Message** filter settings used for test:



Set the Message

Change the contents of the message body.

Name: Set Message

Content-Type: text/xml

Message Body:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <messagefromqueue>Request has been consumed from the WebLogic Queue</messagefromqueue>
    </soap:Body>
</soap:Envelope>
```

Creating a JMS Session:

NOTE: If a JMS Session has already been created as per section 4.1, skip to number 5 below to add a JMS consumer to the existing JMS Session.

1. Click on **the External Connections** navigation panel in Policy Studio
2. Right click on **JMS Services** and click on **Add a JMS Service**
3. Configure the following fields for the JMS Service:
   - **Name:** Oracle BEA WebLogic
   - **Provider URL:** t3://WebLogic_host_ip:7001(where IP address points to WebLogic has been installed)
   - **Initial Context Factory:** weblogic.jndi.WLInitialContextFactory
   - **Connection Factory:** weblogic.jms.ConnectionFactory
   - **Username:** The value here is configured during WebLogic install

- **Password:** The value here is configured during WebLogic install

4. Select the **Services** navigation button then right click on the **OEG Gateway** process and select **Messaging System->Add JMS Session**
5. In the popup window by the JMS Service drop down select the "Oracle BEA Weblogic" service that was created earlier.
6. Do not select the **Allow Duplicates**. Click **OK**
7. Right click on the **JMS Session** and select **Add JMS Consumer**.
8. Configure as follows:
   - **Source**: jms/testQueue (for this tutorial the same queue is used as where message was place on before. Of course any queue can be used)
   - **Selector:** Can be left blank by default
   - **Extraction Method:** Create a content.body attribute based on the SOAP Over JMS Draft specification.
9. Point the JMS Consumer to the **Read from WebLogic Queue** policy

Ensure policies are updated on the Gateway

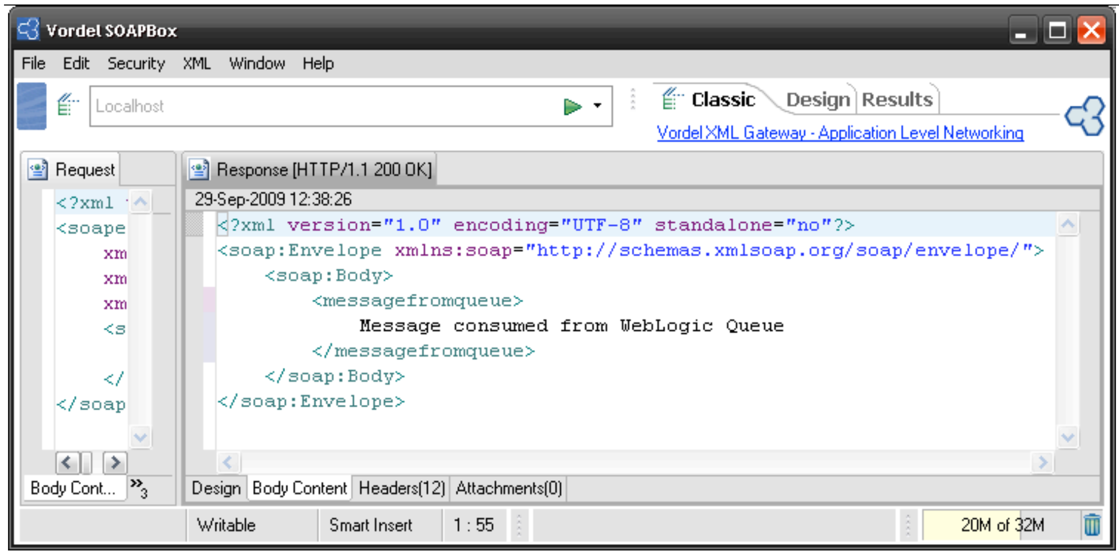
Complete the following steps to refresh the policies:

1. Open the **Policy Studio** and click on **Settings**.
2. Select **Deploy** and **Yes** to the popup window to ensure that the changes made are propagated to the running Gateway.

Testing to read messages from a queue

The Gateway has also been configured to let the JMS service consume the message on the queue and to set a customized response when the message has been processed by the 'Read from WebLogic Queue' policy.
By creating the JMS consumer and the policy that it pointed to (i.e. 'Read from WebLogic Queue') that contains a **Set Message** filter, the messages have been converted to a customized response message.

Below is the response from the Gateway after having consumed and processed a message from the WebLogic queue.


Vordel SOAPBox
File Edit Security XML Window Help
Localhost
Classic Design Results
Vordel XML Gateway - Application Level Networking
Request
Response [HTTP/1.1 200 OK]
Body Cont...
Design Body Content Headers(12) Attachments(0)
Writable Smart Insert 1 : 55 20M of 32M

29-Sep-2009 12:38:26

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
        <messagefromqueue>
            Message consumed from WebLogic Queue
        </messagefromqueue>
    </soap:Body>
</soap:Envelope>
```

Below is a snippet from the trace file showing how the message was processes by the policy being called by the JMS consumer:

```
--------------------------------------------------------------------------------------------------------------
DEBUG   12:40:19:908 [0b08] New message read from JMS queue, Vordel
message ID is: Id-000001240599dc44-00000000009ad840-2
DEBUG   12:40:19:908 [0b08] JMS header: name=contentType,
value=text/xml;
        charset="utf-8"
DEBUG   12:40:19:908 [0b08] JMS header: name=JMSXDeliveryCount,
value=1
DEBUG   12:40:19:908 [0b08] JMS Message ID:
ID:<390902.1254224476532.0>
DEBUG   12:40:19:908 [0b08] JMS Correlation ID: Id-000001240599dc25-
000000000133f6dd-3
DEBUG   12:40:19:908 [0b08] JMS Type: Unknown JMS Type: Tue Sep 29
12:40:19 BST 2009
DEBUG   12:40:19:908 [0b08] JMS Destination: SystemModule-0!Queue-0
DEBUG   12:40:19:908 [0b08] Extracting mesage with automime
DEBUG   12:40:19:908 [0b08] handle type text/xml with factory class
com.vordel.mime.XMLBody$Factory
DEBUG   12:40:19:908 [0b08] relay from a class
com.vordel.dwe.jms.BytesMessageInputStream to a class
```

```
 com.vordel.dwe.BufferingContentSource$FeedStream
DEBUG   12:40:19:908 [0b08] run circuit "Read Message from WebLogic
Queue"
DEBUG   12:40:19:908 [0b08] run filter [Check Message XML Complexity] {
DEBUG   12:40:19:908 [0b08]     read 290 from native stream
DEBUG   12:40:19:908 [0b08]     read EOF from native stream
DEBUG   12:40:19:908 [0b08]     close content stream
DEBUG   12:40:19:908 [0b08]     close content stream
DEBUG   12:40:19:908 [0b08] } = 1, in 0 milliseconds
DEBUG   12:40:19:908 [0b08] run filter [Create acknownledgment Message
(Set Message)] {
DEBUG   12:40:19:908 [0b08]
ConversionProcessor.setConvertedMessage:The contentype of the converted
message is text/xml
DEBUG   12:40:19:908 [0b08]     handle type text/xml with factory class
com.vordel.mime.XMLBody$Factory
DEBUG   12:40:19:908 [0b08]     ConversionProcessor.setConvertedMessage:
coverted message is added to the the pipeline
DEBUG   12:40:19:908 [0b08]     ChangeMessageProcessor.convert: finished
DEBUG   12:40:19:908 [0b08] } = 1, in 0 milliseconds
DEBUG   12:40:19:908 [0b08] Read Message from WebLogic Queue
DEBUG   12:40:19:908 [0b08] Sending the response message to the reply-to:
JMSServer-0!JMSServer-0.TemporaryQueue10
DEBUG   12:40:19:908 [0b08] Creating JMS byte message using automime
DEBUG   12:40:19:908 [0b08] relay from a class java.io.ByteArrayInputStream
to a class com.vordel.dw
-----------------------------------------------------------------------------------------
----------------
```

6. Conclusion

This document is a demonstration on how to setup the connection from a OEG Gateway to the WebLogic provider using the JMS Service and filter options in the Gateway. It demonstrates how the Gateway can place messages on a WebLogic JMS queue, and also the consumption of messages from the WebLogic JMS queue and processing the consumed messages via custom policy.

This configuration can be part of a larger policy, including features such as XML threat detection and conditional routing, features which are out of the scope of this document but are covered in other documents which can be obtained from Oracle at http://www.oracle.com.

Oracle is committed to developing practices and products that help protect the environment

Oracle Enterprise Gateway
May 2011
Author:

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

SOFTWARE. HARDWARE. COMPLETE.