# Summary

This document details the process of installing the NREL mediator service onto an OpenESB server. The service, as currently written, supports two methods: create mainenance order, and get maintenance orders. The service performs the following functions. It receives a request and splits it, sending it to two remote SOAP servers. It also outputs the requests to logfiles.

# Installation of OpenESB

## Downloading OpenESB:

1. Download OpenESB (<http://www.open-esb.net/>). My version 3.0.5
2. Unzip the file
3. Move the directory to /usr/local

## Get the mysql connector:

* Copy mysql-connector-java-5.1.38-bin.jar (or latest version) to OpenESB-Se-3.0.5/OE-Instance/lib/ext
* This will require an openesb restart if done later in the process

## Starting the OpeneSB Server:

1. Start the server :   
   sh> /usr/local/OpenESB-SE-3.0.5/OE-Instance/bin/openesb.sh start &
2. Server Console :  
   <http://localhost:4848/plugin/webui>
3. Login : admin/admin

## Component Installation

OpenESB supports plugins. The mediator requires installation of the following components :

* BPEL SE – scalable orchestrator (Component) bpelse.jar
* FILE BC – provides interacting with files (Component) filebc.jar
* HTTP BC FULL – provides messaging over HTTP (Component) httpbc-full.jar
* Wsdlextlib.jar (Shared lib)
* Encoderlib.jar (Shared lib)

### Installing Shared Libraries

* Access web console : <http://localhost:4848/plugin/webui>
* Click on Shared Libraries (on left menu)
* Click Install
* Select file : /usr/local/OpenESB-SE-3.0.5/OE-Components/**encoderlib.jar**
* Click ‘Start upload’

Repeat process for **wsdlextlib.jar**

### Installing the 3 components

* Access web console : <http://localhost:4848/plugin/webui>
* Click on Components (on left menu)
* Click Install
* Click ‘Choose File’
* Select file : /usr/local/OpenESB-SE-3.0.5/OE-Components/**bpelse.jar**
* Click Install

Repeat process for **filebc.jar** and **httpbcfull.jar**

## Setup the database context file :

Set up the database context file.

* Edit file : /usr/local/OpenESB-SE-3.0.5/OE-Instance/config/context.xml
* See Appendix B for example file
* The jndi-name must be : interop (as in example)

**Congratulations. The instance is now configured.**

# Installation of Mediator Application

* Access web console : <http://localhost:4848/plugin/webui>
* Click ‘Service Assemblies’ (left side menu)
* Click on ‘+ Deploy’ button
* Choose file : git/nrel/NRELApp/dist/NRELApp.zip
* Click ‘Start upload’
* Click ‘Start’ button on the service assembly page

Note: the start will partially fail at this point, but it will also create the endpoint variables (initially set to null) for the next step.

# Endpoint Configuration

The mediator endpoint are configurable. There are six endpoints, three for Get, and three for Create:

* NREL\_CREATE\_SERVICE – This is the port that the mediator SOAP service listens on for create maint order requests. It logs the requests and sends them to the two remote services for execution
* NREL\_CREATE\_R1 –SOAP service for create maint order remote server 1
* NREL\_CREATE\_R2 –SOAP service for create maint order remote server 2
* NREL\_GET\_SERVICE – This is the port that the mediator SOAP service listens on for get maint order requests. It logs the requests and sends them to the two remote services for execution
* NREL\_GET\_R1 – EPRI SOAP service for get maint order remote server 1
* NREL\_GET\_R2 – EPRI SOAP service for get maint order remote server 2

The urls for these three addresses must be configured within the OpenESB web console.

Do the following in order for each of the six variables :

* Access web console : <http://localhost:4848/plugin/webui>
* Click on Components (left menu)
* Click on sun-http-binding item
* Click on ‘Application Variables’
* Do the following steps for the each variable (example below):
  + Click ‘+ Add’
  + The vars should already be created, but null. Fill in with your own values which should resemble those below
  + Application Configuration Name : NREL\_CREATE\_SERVICE <http://localhost:8081/epriConnect/MaintOrderServiceCreate>

|  |  |
| --- | --- |
| **Name** | **Value** |
| NREL\_GET\_SERVICE | <http://localhost:8081/epriConnect/MaintOrderServiceGet> |
| NREL\_GET\_R1 | <http://localhost:8080/epriConnect/MaintOrderServiceGet> |
| NREL\_GET\_R2 | <http://localhost:8082/epriConnect/MaintOrderServiceGet> |
| NREL\_CREATE\_SERVICE | <http://localhost:8081/epriConnect/MaintOrderServiceCreate> |
| NREL\_CREATE\_R1 | <http://localhost:8080/epriConnect/MaintOrderServiceCreate> |
| NREL\_CREATE\_R2 | <http://localhost:8082/epriConnect/MaintOrderServiceCreate> |

Note : If you are lazy (or don’t have two epriConnect servers set up), you can set R1 and R2 to point to the same service, ie: port 8080/epriConnect/MaintOrderGet|Create

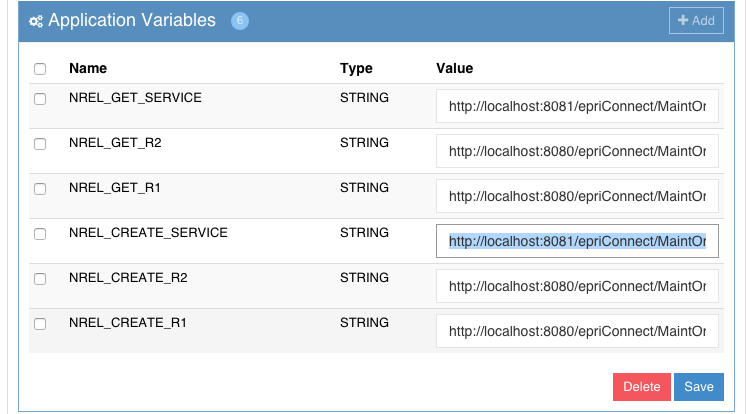


Figure : Configuration Variables

# Start Mediator Service

* Click on ‘Service Assemblies’ on left menu
* Click on NRELApp
* Click on Shutdown
* Click on ‘Start’

At this point the service should be up and ready to receive connections.

# Testing Mediator Application

### Set up (open) SOAP UI Project to send create maintenance order

* Open Soapui – git/nrel/soapui/NREL-CREATE-soapui-project.xml
* Select CreateMaintenanceOrders Test Case/Test Steps/CreateMaintenanceOrders
* Set url : <http://localhost:8081/epriConnect/MaintOrderServiceCreate> (This should be the url set for NREL\_CREATE\_SERVICE – See EndPointConfiguration)

### Set up (open) SOAP UI Project to get maintenance order

* Open Soapui – git/nrel/soapui/NREL-GET-soapui-project.xml
* Select CreateMaintenanceOrders Test Case/Test Steps/CreateMaintenanceOrders
* Set url : <http://localhost:8081/epriConnect/MaintOrderServiceGet> (This should be the url set for NREL\_GET\_SERVICE – See EndPointConfiguration)

### Start both epricConnect services

* Start two copies of tomcat (instructions for running two versions of tomcat on the same server are included in Appendix A)
* Drop the epriConnect.war file to your tomcat/webapps directory
* Verify that the your SOAP services are by doing the following :
  + Browse to the url you entered as REMOTE\_1, appending ‘?wsdl’ to it, eg: <http://localhost:8080/epriConnect/MaintOrderServiceCreate?wsdl>
  + Browse to the url for REMOTE\_2, appending ‘?wsdl’ to it,  
    eg: <http://localhost:8085/epriConnect/MaintOrderServiceCreate?wsdl>

### Start Mediator Service

* Access web console : <http://localhost:4848/plugin/webui>
* Click Service Assemblies
* Start NRELApp
* Browse to the url you set during configuration, appending ‘?wsdl’ to the url, eg: <http://localhost:8081/epriConnect/MaintOrderServiceGet?wsdl> (you should see the wsdl for the get service)
* Browse to the url you set during configuration, appending ‘?wsdl’ to the url, eg: <http://localhost:8081/epriConnect/MaintOrderServiceCreate?wsdl> (you should see the wsdl for the create service)

### Executing Basic Test

#### Create Test

This test will do the following :

* Send a create maint order request from SOAPUI to the mediator service
* The mediator will then
  + Send the SOAP request to R1 (soap service)
  + Send the SOAP request to R2 (soap service)
  + Log the request message to /tmp/CreateMaintOrderRequest.xml
* Mediator Part 2
  + R1 reply is written to /tmp/CreateMaintOrderResponse.xml
  + R2 reply is written to /tmp/CreateMaintOrderResponse.xml
  + R1 reply is relayed to SOAPUI as the response

Execute the Test :

* Open soapui file : **NREL-CREATE-soapui-project.xml**
* Open test : **TestSuite/TestCase/Test Steps/CreateMaintenanceOrder**
* Set the endpoint to point to point to the mediator : <http://localhost:8081/epriConnect/MaintOrderServiceCreate>
* Set the End Point in SOAP ui to the URL you set for SERVICE\_ENDPOINT (eg: <http://localhost:8081/epriConnect/MaintOrderServiceCreate>)

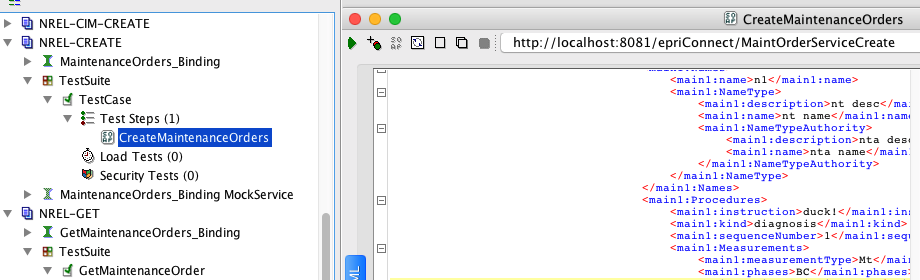


Figure : Soap UI Test Case

* Execute!

Verify results :

* Response window in SOAP UI
* File /tmp/CreateMaintOrderRequest.xml –request soap message
* File /tmp/CreateMaintOrderResponse.xml – contains response from NREL\_CREATE\_R1, NREL\_CREATE\_R2

#### Get Test

This test will do the following :

* Send a create maint order request from SOAPUI to the mediator service
* The mediator will then
  + Send the SOAP request to R1 (soap service)
  + Send the SOAP request to R2 (soap service)
  + Log the request message to /tmp/GetMaintOrderRequest.xml
* Mediator Part 2
  + R1 reply is written to /tmp/GetMaintOrderResponse.xml
  + R2 reply is written to /tmp/GetMaintOrderResponse.xml
  + R1 reply is relayed to SOAPUI as the response

Execute the Test :

* Open soapui file : **NREL-GET-soapui-project.xml**
* Open test : **TestSuit/TestCase/Test Steps/GetMaintenanceOrder**
* Set the endpoint to point to point to the mediator : <http://localhost:8081/epriConnect/MaintOrderServiceGet>
* Set the End Point in SOAP ui to the URL you set for SERVICE\_ENDPOINT (eg: <http://localhost:8081/epriConnect/MaintOrderServiceGet>)

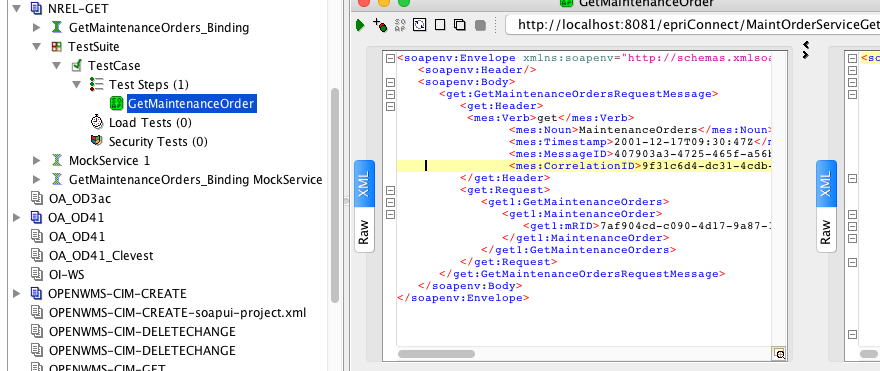


Figure : Soap UI Test Case

* Execute!

Verify results :

* Response window in SOAP UI
* File /tmp/GetMaintOrderRequest.xml –request soap message
* File /tmp/GetMaintOrderResponse.xml – contains response from NREL\_GET\_R1, NREL\_GET\_R2

## Mediator Flow

The following represents the flow of the mediator.

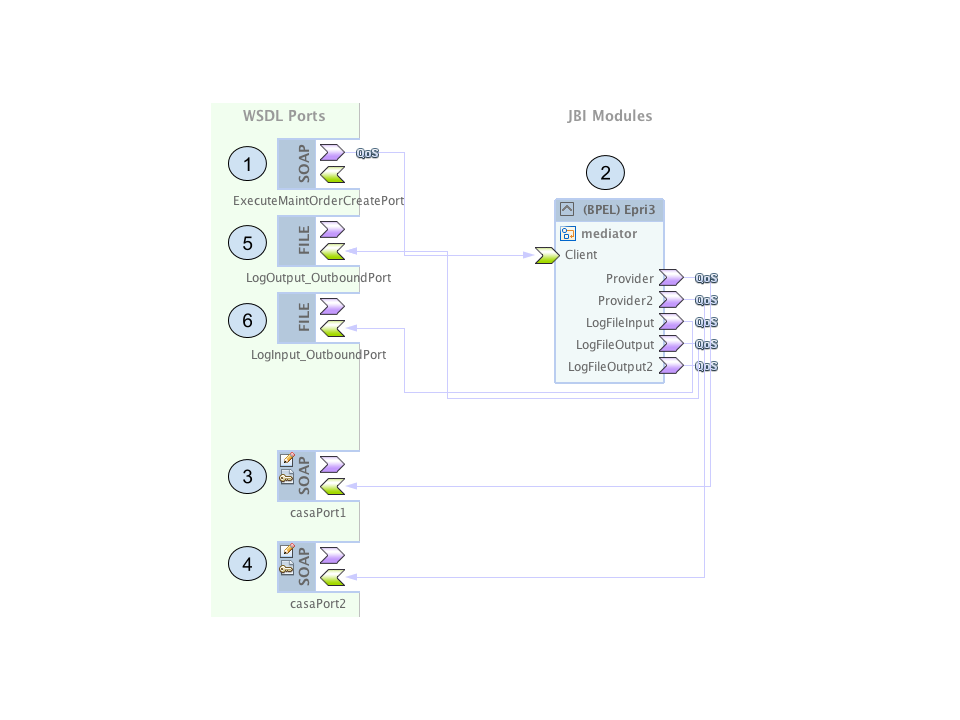


Figure : Mediator Flow

1. Mediator Service Port – Port defined by SERVICE\_ENDPOINT application variable
2. Mediator Service – Routes soap messages
3. EpriConnect service endpoint – REMOTE\_1
4. EpriConnect service endpoint 2 – REMOTE\_2
5. Log output file for SOAP Message sent, and message received from REMOTE\_1
6. Log output file for SOAP Message received back from REMOTE\_2

# Appendix A – Creating a Second Tomcat Instance on a Server

Process for creating a second tomcat instance on a unix box

* sh> cp –rf /usr/local/tomcat /usr/local/tomcat2
* sh> vi /usr/local/tomcat2/conf/server.xml

<Server port="8006" shutdown="SHUTDOWN"> <!— bump it up 1 from first instance —>

<Connector port="8085" protocol="HTTP/1.1"

connectionTimeout="20000"

redirectPort="8444" /> <!— again, bump ports up by one —>

<Connector port="8010" protocol="AJP/1.3" redirectPort="8444" />

# Appendix B – OpenESB context.xml example

**<?xml version="1.0" encoding="UTF-8"?>**

**<context xmlns="http://www.open-esb.net/standalone/jndi/">**

**<!--**

**WARNING:Be aware that you have to add JDBC driver in the classpath**

**(e.g. lib/ext of you OpenESB installation dir)**

**before being able to use this example !**

**-->**

**<!-- Oracle config sample -->**

**<dataSource-pool-properties>**

**<dbConnector-name>interop connector</dbConnector-name>**

**<datasource-classname>com.mysql.jdbc.jdbc2.optional.MysqlDataSource</datasource-classname>**

**<resource-type>Datasource</resource-type>**

**<database-name>MYSQL</database-name>**

**<database-vendor>Oracle</database-vendor>**

**<database-version>5.6</database-version>**

**<dbconnector-description>MYSQL Connector for WMS</dbconnector-description>**

**<dataSource-properties>**

**<property>**

**<name>user</name>**

**<value>interop\_dev</value>**

**<description></description>**

**</property>**

**<property>**

**<name>password</name>**

**<value>interop\_dev</value>**

**<description></description>**

**</property>**

**<property>**

**<name>hostName</name>**

**<value>localhost</value>**

**<description></description>**

**</property>**

**<property>**

**<name>port</name>**

**<value>3306</value>**

**<description></description>**

**</property>**

**<property>**

**<name>databaseName</name>**

**<value>interop\_dev</value>**

**<description></description>**

**</property>**

**</dataSource-properties>**

**<pool-properties>**

**<property>**

**<name>initialSize</name>**

**<value>4</value>**

**<description></description>**

**</property>**

**<property>**

**<name>maxActive</name>**

**<value>4</value>**

**<description></description>**

**</property>**

**<property>**

**<name>maxIdle</name>**

**<value>4</value>**

**<description></description>**

**</property>**

**<property>**

**<name>minIdle</name>**

**<value>2</value>**

**<description></description>**

**</property>**

**</pool-properties>**

**</dataSource-pool-properties>**

**<jdbc-resources>**

**<dbConnector-name>interop connector</dbConnector-name>**

**<jndi-name>interop</jndi-name>**

**<description>Datasource connection to MYSQL WMS</description>**

**</jdbc-resources>**

**</context>**