How to deploy a EJB as a Web Service on Axis 1.0

John Mammen : john_mammen@yahoo.com

In this 'how to', there are four classes involved, three EJB related classes and one details class.

Lets call this bean **Rates** Bean. In brief, the functionality of this bean is to obtain rate details of a currency when an three digit currency code is passed.

The four Classes are *Rates.java*, the remote bean, *RatesBean.java*, implementation bean, *RatesHome.java* home and *RatesDetails* return class for rate details.

Below are excerpts of xml deployment descriptors and java code for Rates bean for Weblogic deployment, which can used for reference.

Note: None of the below excerpts are complete. Only required details are shown.

Rates.java (This is the EJB Remote class)

```
……
public interface Rates extends EJBObject {

public RatesDetails getRateDetails(String currencyCode) throws
RemoteException,RateException;
…
}
```

RateDetails.java (This is the class returned by remote EJB Call)

```
…

public class RatesDetails implements Serializable{

    private String isoCode;
    private String vaildFrom;
    private String vaildTo;
    private double retailBuy;
    private double retailMid;
    private double retailSell;
    private double wholesaleBuy;
    private double wholesaleMid;
    private double wholesaleSell;
    private int rateType;
    public RatesDetails()  {
    }
    // setters and getters are defined but left out here
  ……
```

ejb-jar.xml

```
    <session>
      <ejb-name>RatesBean</ejb-name>
      <home>RatesHome</home>
      <remote>Rates</remote>
      <ejb-class>RatesBean</ejb-class>
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
       <resource-ref>
        <description>Datasource for Rates DB</description>
        <res-ref-name>jdbc/NewRatesDB</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Container</res-auth>
      </resource-ref>
    </session>
```

weblogic-ejb-jar.xml

```
<weblogic-enterprise-bean>
        <ejb-name>RatesBean</ejb-name>
            <stateless-session-descriptor>
                <pool>
             <max-beans-in-free-pool>50</max-beans-in-free-pool>
           <initial-beans-in-free-pool>5</initial-beans-in-free-pool>
                </pool>
            </stateless-session-descriptor>
             <reference-descriptor>
                <resource-description>
                    <res-ref-name>jdbc/NewRatesDB</res-ref-name>
                    <jndi-name>jdbc.RatePool</jndi-name>
                    </resource-description>
            </reference-descriptor>
        <jndi-name>jndi.Rates</jndi-name>
</weblogic-enterprise-bean>
```

Assuming similar classes as described above and successfully deployed to your container, we can start the deployment of web services.

1. First make sure you have successfully deployed your EJB to your EJB Container.
2. Then install and configure Axis 1.0 with your J2EE application server.
3. Test the axis deployment by clicking on the validate of axis installation page.
4. Create a rates.wsdd in notepad as shown below. The below example corresponds to the above beans described. Please make note of the bolded text. Also make sure your class names should include the packages if in packages. For example: if your **RatesHome** was in package com.mycompany, then your home interface name is **com.mycompany. RatesHome**

```
<?xml version="1.0" encoding="UTF-8"?>
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
        xmlns:java="http://xml.apache.org/axis/wsdd/providers/java"
xmlns:xsi="http:/www.w3.org/2000/10/XMLSchema-instance">
        <service name="RateDetails" provider="java:EJB">
        <parameter name="beanJndiName" value="jndi.Rates" />
        <parameter name="homeInterfaceName" value="RatesHome" />
        <parameter name="remoteInterfaceName" value="Rates" />
```

```
        <parameter name="allowedMethods" value="getRateDetails" />
        <parameter name="className" value="RatesBean" />
        <parameter name="jndiURL" value="t3://localhost:7001" />
        <parameter name="jndiContextClass"
value="weblogic.jndi.WLInitialContextFactory" />
<typeMapping
      xmlns:ns="http://soapinterop.org/xsd"
      qname="ns:RatesDetails"
      type="java:RatesDetails"
      serializer="org.apache.axis.encoding.ser.BeanSerializerFactory"
      deserializer="org.apache.axis.encoding.ser.BeanDeserializerFactory"
      encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
    />
</service>
</deployment>
```

5. Now deploy your rates.wsdd by issuing the command:
   `java org.apache.axis.client.AdminClient -lhttp://localhost:7001/axis/services/ejbservice rates.wsdd`

6. If successfully deployed, you will not get any error message. If there are any errors, the console will display the errors. Also take a look at your EJB Container's console for any errors.
7. Now test your web service in the browser url:
   http://localhost:7001/axis/services/RateDetails?method=getRateDetails&value=USD

If deployment was successful, you should get something like this.

Axis Reply XML

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <geRateDetailsResponse
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <geRateDetailsReturn href="#id0"/>
  </geRateDetailsResponse>
  <multiRef id="id0" soapenc:root="0"
soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xsi:type="ns1:RatesDetails"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:ns1="http://soapinterop.org/xsd">
   <rateType xsi:type="xsd:int">0</rateType>
   <wholesaleSell xsi:type="xsd:double">1.0</wholesaleSell>
   <retailSell xsi:type="xsd:double">1.0</retailSell>
   <retailMid xsi:type="xsd:double">1.0</retailMid>
   <wholesaleMid xsi:type="xsd:double">1.0</wholesaleMid>
   <validTo xsi:type="xsd:string" xsi:nil="true"/>
   <validFrom xsi:type="xsd:string" xsi:nil="true"/>
   <isoCode xsi:type="xsd:string" xsi:nil="true"/>
```

```
   <retailBuy xsi:type="xsd:double">1.0</retailBuy>
   <wholesaleBuy xsi:type="xsd:double">1.0</wholesaleBuy>
  </multiRef>
 </soapenv:Body>
</soapenv:Envelope>
```

If there is any error, please post to axis-user@xml.apache.org