|  |
| --- |
| Probe Reference ApplicationInstallation Manual |
| AuthorsRobert van MölkenCreation date02 July 2013 | Version / status1.0**AMIS**Edisonbaan 15Postbus 243430 AA NieuwegeinT +31(0) 30 601 60 00E info@amis.nlI amis.nlBTW nummer NL811770400B69KvK nummer 30114159Statutair gevestigd te EnschedeModified date09 July 2013 |

Document management

modification history

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Author | Version | Notes |
| 02-07-2013 | Robert van Mölken | 0.9 | Initial Dutch documentation |
| 10-07-2013 | Robert van Mölken | 1.0 | Rewrite to English |

Copyright 2013, AMIS

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of AMIS

Inhoudsopgave

[1 Installation instructions 4](#_Toc361229309)

[1.1 Data source and Adapter resource (Weblogic) 4](#_Toc361229310)

[1.2 Installation of the JAXWS web service (uses ADF-BC) 8](#_Toc361229311)

[1.3 Installation of the Oracle Service Bus Services 9](#_Toc361229312)

[1.4 Installation of the SOA Suite Service 12](#_Toc361229313)

[2 Using/testing components 14](#_Toc361229314)

[2.1 Available components 14](#_Toc361229315)

[2.2 Service interface and operations 14](#_Toc361229316)

[2.3 Sample messages 14](#_Toc361229317)

1 Installation instructions

* 1. Data source and Adapter resource (Weblogic)

Some Probe services are, on specific moments, using a data source to connect to a physical database. The configuration is done with use of the Weblogic Console. Within Weblogic the physical connections to a database (= data source) are configured and for the different adapters (i.g. DB, AQ, JMS…) connection pools are configured based on a data source.

* + 1. Data sources

Data sources are physical connections to a database. For this sample implementation one data source is used.

1. Open a browser window and browse to the following web address: <http://host:7001/console/>
2. Log in with the username of the weblogic admin which is created during the installation.
The default username/password combination is **weblogic/welcome1**.

3. Look under *Domain Configurations -> Services* and click on the link **Data Sources**
4. On next screen, click the  button and choose Generic Data Source to open the wizard for creating a new data source.
5. Choose the name **ProbeDataSource** and choose the JNDI Name **jdbc/ProbeDataSource**. Choose default value **Oracle** for the Database Type field.

6. Click  and keep the database driver to the specified value are and click  twice.
7. Enter the details for the physical connection to the database, including the database name (SID), host, port and database (for example the **soa\_infra** database).

8. Click  to accept the settings and check that the settings are correct using the button.

9. Upon successful connection, click on  and in next screen select the target servers. Select for example soa\_server1 and osb\_server1. The Oracle VM select the Admin Server here.

	* 1. Adapter Resources

Besides a data source a connection factory is needed which manages the transactions to the database, also called an adapter resource. Adapters are used by the Service Bus and Oracle SOA Suite to communicate with underlying systems. For the database we will to create an adapter resource.

1. Log in on the Weblogic console and click under *Domain Configurations -> Your Deployed Resources* on the link **Deployments**.

2. Search through the list for the DbAdapter deployment.

3. The resources can be made / changed by clicking on the link of the adapter deployment and clicking on the tab *Configuration -> Outbound Connection Pools*.
4. To create a new connection pool click the  button.

5. In the first step select the group **javax.resource.cci.ConnectionFactory**, then click the  button and enter in the JNDI name **eis/DB/probe** and click on .
6. After creating the connection pool it should be linked to the data source created earlier. For **eis/DB/probe**, the value for the field to be xADataSourceName should be filled with value **jdbc/ProbeDatasource**. After completing first press enter and then click on the button  to save the value.


7. After changing the data, which is ultimately stored as a plan, remember to update the DbAdapter deployment. Go back to the list of deployments and check the box in front of the label DbAdapter and click on the update button at the top or bottom of the list.


8. On the next screen, select "**redeploy this application**" and click the  button to select this option and then click on the  button to redeploy.

	1. Installation of the JAXWS web service (uses ADF-BC)

Once Weblogic is configured the JAXWS web service can be deployed. For this, we still use the WebLogic console.

1. Go to the overview page of the deployments, from the home screen, follow *Domain Configurations -> Your Deployed Resources*.
2. Click the button  to deploy a new application. In the screen that opens, you can choose the deployment from an existing location on the server but for now it is assumed that the file is not present on the server.
3. Click on the link “**upload your file(s)**” written I the in the note.

4. Browse to the archive file (**/jaxws/ProbeServiceJAXWS.ear**) and click the  button to upload the file.
5. You will return to the previous screen, just select the uploaded file and then click  and then on  again to install application the file.
6. On the target screen, select the WebLogic Server were the application should be installed on, on this Weblogic server the data source **jdbc/ProbeDataSource** should be present, and click on .
7. Retain the default settings shown in the next screen and click  to install the application.

	1. Installation of the Oracle Service Bus Services

After installing the JAXWS web service the OSB projects can connect to it. There are two service, the project ProbeService and ProbeServiceOSBProxy. These services use the Oracle Service Bus. For the OSB there is a separate console to configure, manage and monitor the services.

1. Open a new browser window and browse to the URL of the EM: <http://host:7001/sbconsole/>
2. On the bottom left of the menu click on the item System Administration. Under this heading you can import and export configurations. By default you see the import page.

3. Before you can change anything in the Oracle Service Bus, you must open a Change Session. A Change Session contains all changes made ​​and can therefore be easily rolled back. You can find the Change Center on the top left of the screen. Click on the  button to start a session.


4. After creating a session, it is possible to make a changes. Click the Browse button to select the file **/OSB/ProbeService\_sbconfig.jar** to import and click on the  button to start the import.


5. On the next page click the button  to import the ProbeService project.


6. The import shows that is completed successfully.

7. Perform the last three steps also for the second service. Click the button  and browse to the file **ProbeServiceOSBProxy\_sbconfig.jar**.
8. Click in the upper left within the Change Center on the button  to activate the changes.

9. When activating the session give it a good description of the changes so it is easy to identify and possibly reverse the changes.



After importing the services the environment specific settings can be adjusted, e.g. the endpoint of the SOA Suite service used in the ProbeService and the endpoint of the JAXWS web service used in ProbeServiceOSBProxy. For this there are two sample files included with the OSB deployments. The main setting that can be changed are these endpoints. Adjust them if necessary, so that it points to the correct endpoint address of the service to call.

For the ProbeService the file **ProbeService\_dev\_config.xml** can be used which has the following value:



Do this also for the envValueType **Service URI Table**, which has a field with the name URI.

For the ProbeServiceOSBProxy the file ProbeServiceOSBProxy\_dev\_config.xmlcan be used. Change the value of the Service URI and Service URI Table were necessary, so that they point to the correct endpoint of the service to call.

1. Open a new session and click on the left menu under the header Customization on the link **Execute Customization File**.

2. On the page that is displayed you can now import the Customization file in the same way as the import of a service (do this for both files).


3. After executing the Customization file the session needs to be activated to commit the changes.
	1. Installation of the SOA Suite Service

After installing the JAXWS web service and the OSB components the SOA Suite composite application can be deployed. For the installation we are using the Enterprise Manager (EM). Using the EM we can deploy, start & stop en manage & monitor SOA composite services and it’s runtime instances.

1. Open a new browser window and brwose to the URL of the EM: http://host:7001/em/
2. On the left side menu open up the item SOA -> soa-infra and click on the partition were you want to deploy it in. For example the **default** partition.

3. After clicking on the partition you end up on the overview page of the partition.

4. On this page click on the button Deployment and select the option Deploy To This Partition ... to deploy the integration services on this partition.

5. A wizard will now start. In the first step, select the supplied JAR file /**SCA/sca\_ProbeService\_rev1.0.jar** by browsing to the file on the local disk.



We also select a Configuration Plan for the specific environment. This plan includes the endpoint of the (next) external component / service that can be called. There is a sample plan included with the SOA deployments. For multiple environments a copy can be created with the setting of that environment.

In the example plan probeservice\_dev\_cfgplan.xml the location (service WSDL) of the OSB proxy that is invoked can be adjusted.



1. Select the appropriate configuration plan by browsing to the file. Click on the button  to accept archive and configuration plan.


2. In the next step, the details are shown of the service that will be created. Click on the button  to create the new service.
3. The service is now created and can be invoked.

4. The key for a integration test is that all endpoints are correct. Make sure that the configuration files contain the appropriate endpoints and re-run the OSB customization files where necessary with the correct SOA Composite service endpoint.
5. Using/testing components
	1. Available components

As indicated during installation there are four components, two OSB components, one SOA component and on JAXWS component. These four components are linked in accordance with the lower figure.



* 1. Service interface and operations

Each component uses the same service interface and therefore also has the same operations. The ProbeService interface defines three operations; ping, deepPing and checkProcess.

* + 1. The ping operation

This operation is the simplest of the three operations, each component individually implements this. The only functionality it contains is an echo back with or without the DateTime from the request, depends if value is given, and the servers current DateTime.

* + 1. The deepPing operation

This operation calls all subsequent components to inventory the route that the request travels and to report that route. There is a parameter in the request that determines how many components/levels deep that requests needs to call. For example if you give this parameter the value “1” and call the deepPing operation on the OSB service endpoint, the proxy will call the SOA Suite service which will reply back to the OSB service without calling any deeper services. The result is always a stack trace of the called components. If an error arises in any of the components an error message is added to the component in the stack trace, no error means the component is responding correctly.

* + 1. The checkProcess operation

This operation is similar to the deepPing operation, only this operation call all subsequent components and it checks if a component is also working at a functional level. This operation processes the given request values and concatenates all the values to a single string. Each component also adds a unique value/identifier to the output string on the way back (response). Looking at the final response value an administrator should determine if all components processed the input values of that an component raised an error. It is also easy to recognize that the service calls are not cached.

* 1. Sample messages

There is a sample SoapUI project available, at the same location you found this document, that can be used to test the different components and its operations.