

# Exercise 10

*Using a Registry to publish endpoint metadata and retrieving that endpoint data  
Looking at wider Registry Governance issues*

## Prior Knowledge

Using the ESB, creating services, deploying services  
WSDL

## Objectives

*Deploy a governance registry, connect the services server to the governance registry, publish services into it. Find those services from the ESB.*

## Software Requirements

- Previous installations of ESB and AS from Exercises 8 and 9
- Java Development Kit 7
- WSO2 Governance Registry 4.5.2

1. Start the Governance Registry

```
cd ~/servers/wso2greg-4.6.0  
bin/wso2server.sh
```

2. Go to the administrators console: <https://localhost:9445/>

3. Login with admin/admin



© Paul Fremantle 2012. Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

4. You should see a screen like this:

The screenshot shows the WSO2 Governance Registry Management Console. The left sidebar has sections for Home, Manage, Resources, and Metadata. Under Home, there are 'List' and 'Add' sections. 'List' contains icons for APIs, Documents, Endpoints, Policies, Providers, Proxies, Sequences, Services, URIs, WADLs, and WSDLs. 'Add' contains icons for API, Document, Endpoint, Policy, Provider, Proxy, and Schema. The main content area displays the 'WSO2 Governance Registry Home' page with a welcome message and server configuration details. The server configuration table includes:

Server	
Host	172.16.40.136
Server URL	local/services/
Server Start Time	2013-11-26 20:41:40
System Up Time	0 day(s) 0 hr(s) 1 min(s) 8 sec(s)
Version	4.6.0
Repository Location	file:/home/ox-soa/servers/wso2greg-4.6.0/repository/deployment/server/

Below the server configuration is another table for the 'Operating System' and 'Operating System User'.

Operating System	
OS Name	Linux
OS Version	3.11.0-13-generic

Operating System User	
Country	GB
Home	/home/ox-soa
Name	ox-soa
Timezone	Europe/London

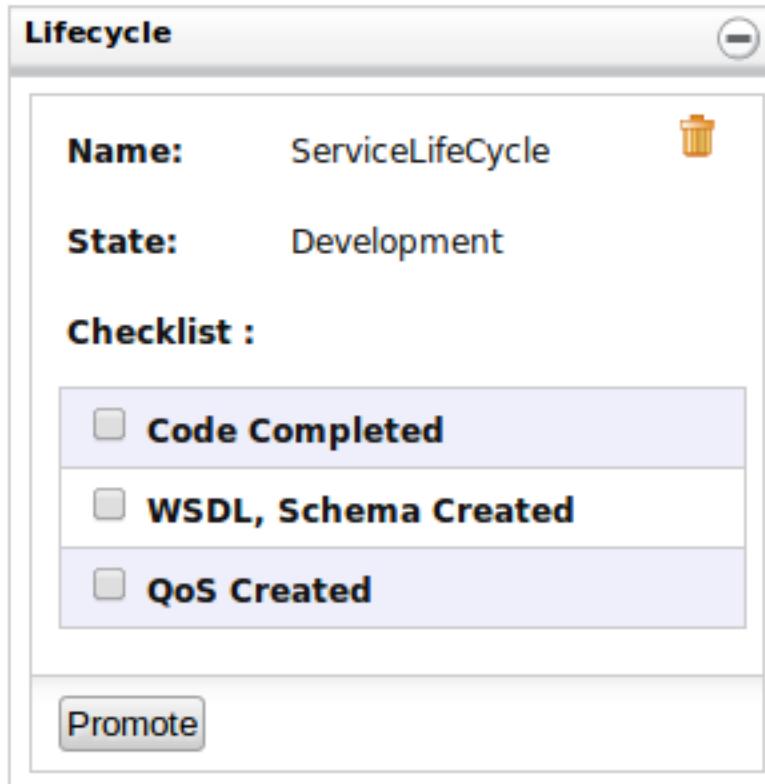
At the bottom, there is a 'Java VM' section with a single entry:

Java Home	javac/jdk/jdk-7-openjdk-i386-free
-----------	-----------------------------------

5. If you click on Services you should see an empty list (since we haven't registered any services with the Registry)
6. In a moment we are going to automatically add some services, but let's first do one the "hard way" – by hand. To make it easy we are going to be quite lazy.
7. Go to the **Add /Service** button on the left hand side.
8. This is a "top-down" way of defining a service. Fill in some information. Use the following information / leave other fields empty:  
Name: OrderService  
Namespace: me.freo.po  
Version: 0.0.1  
Description: Ordering Service.  
Save it.



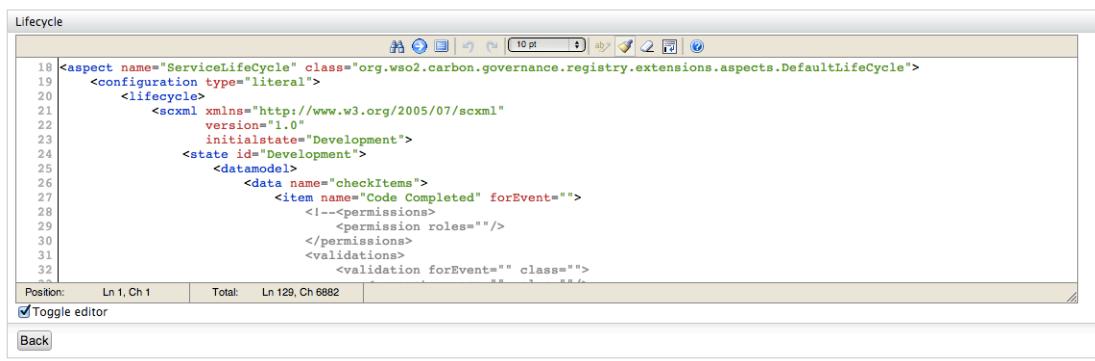
9. Find the Lifecycle section on the right hand side and expand it, if necessary. The service will have been given the “default service lifecycle”, and be in the first stage:



10. Now in order to move this Service out of Development we need to complete some tasks:
  - \* Code Completed
  - \* WSDL, Schema Created
  - \* QoS Created
11. Pretend you've done all that. Select the checkboxes. Now click Promote. You will be prompted to enter a new Version Number. Type in 1.0.0
12. Now look at your Services list again. You will have another version of this service available, now in the Testing Lifecycle. Take a look at it.
13. If you want to understand the gory details of how to configure your own lifecycles and lifecycle actions, go to the left hand **Extensions** tab, click Configure/Lifecycles, then **View** the ServiceLifecycle configuration. You



will see an XML based on the SCXML standard.



The screenshot shows a code editor window titled "Lifecycle". The code is an SCXML configuration file. It defines an aspect named "ServiceLifeCycle" with a class of "org.wso2.carbon.governance.registry.extensions.aspects.DefaultLifeCycle". Inside this aspect, there is a "lifecycle" section with an "scxml" namespace declaration. The "scxml" tag has attributes "version='1.0'" and "initialstate='Development'". It contains a "state" element with id="Development" and a "datamodel" section. The "datamodel" section includes a "data" element named "checkItems" which contains an "item" element named "Code Completed" with a "forEvent" attribute. There are also sections for "permissions", "roles", and "validations". The code editor interface includes a toolbar at the top, status bars at the bottom, and a "Back" button at the bottom left.

```
18 <aspect name="ServiceLifeCycle" class="org.wso2.carbon.governance.registry.extensions.aspects.DefaultLifeCycle">
19   <configuration type="literal">
20     <lifecycle>
21       <scxml xmlns="http://www.w3.org/2005/07/scxml"
22         version="1.0"
23         initialstate="Development">
24         <state id="Development">
25           <datamodel>
26             <data name="checkItems">
27               <item name="Code Completed" forEvent="">
28                 <!--<permissions>
29                   <permission roles="" />
30                 </permissions>
31                 <validations>
32                   <validation forEvent="" class="">
33                     <!--<permissions>
34                       <permission roles="" />
35                     </permissions>
36                   </validation>
37                 </validations>
38               </item>
39             </data>
40           </datamodel>
41         </state>
42       </lifecycle>
43     </configuration>
44   </aspect>
```

Position: Ln 1, Ch 1    Total: Ln 129, Ch 6882

Toggle editor

Back

14. Now lets get our AppServer Services published in the Registry. Our services are hosted in WSO2 Application Server, and this can automatically publish into the Registry using WS-Dynamic Discovery. To do this, we need to tell the AppServer where the Registry is.  
[Please note this only currently works for Axis2 services deployed in AS as there is an incompatibility between CXF's WSDD and G-Reg's WSDD which we are looking into.]

15. To do this, edit the ~/wso2as-5.2.0/repository/conf/axis2/axis2.xml

16. Add the following (bold) line in the correct place:

```
<!-- ===== -->
<!-- Parameters -->
<!-- ===== -->
<parameter
name="DiscoveryProxy">https://localhost:9445/services/DiscoveryProxy</parameter>
<parameter name="hotdeployment">true</parameter>
```

17. Restart (or Start) the Application Server. You can do this from the Web Admin Console or from the command line.



18. Wait a bit and then check the Registry Services list:

Service Name	Service Version	Service Namespace	Actions
admin-RESTSample	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	
echo	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	
HelloService	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	
StarbucksOutletService	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	
Version	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	
wso2carbon-sts	1.0.0-SNAPSHOT	http://docs.oasis-open.org/ws-dd/ns/discovery/2009/01	

19. You can also find the WSDLs from these services in the WSDL list.

20. We can now “Link” the ESB to the Registry:

21. Start the ESB if it isn’t running.

22. Go to the ESB Admin Console (<https://localhost:9444>)

23. Change the left hand menu tab to Configure

24. Select WS-Discovery

25. Click Add Discovery Proxy



26. Fill in the details as follows:

Proxy Name: WSO2Registry

URL: <https://localhost:9445/services/DiscoveryProxy>

Security Policy: [Leave empty]

Home > Configure > WS-Discovery > Discovery Proxy

### Configure WS-Discovery Proxy

Enter the name and the URL of a remote WS-Discovery proxy

Discovery Proxy Settings	
Proxy Name *	<input type="text" value="WSO2Registry"/>
URL *	<input type="text" value="https://localhost:9445/services/DiscoveryProxy"/>
Security Policy	<input type="text"/> <a href="#">Browse</a>
<input type="button" value="Save"/> <input type="button" value="Cancel"/>	

27. Click Save

28. It should now say that the server is Online

Proxy Name	URL	Server Status	Actions
WSO2Registry	https://localhost:9445/services/DiscoveryProxy	On-line	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

29. Click View

You will see the various endpoints from the services that are published in the Registry. Find the service with the “Starbucks endpoints” e.g.

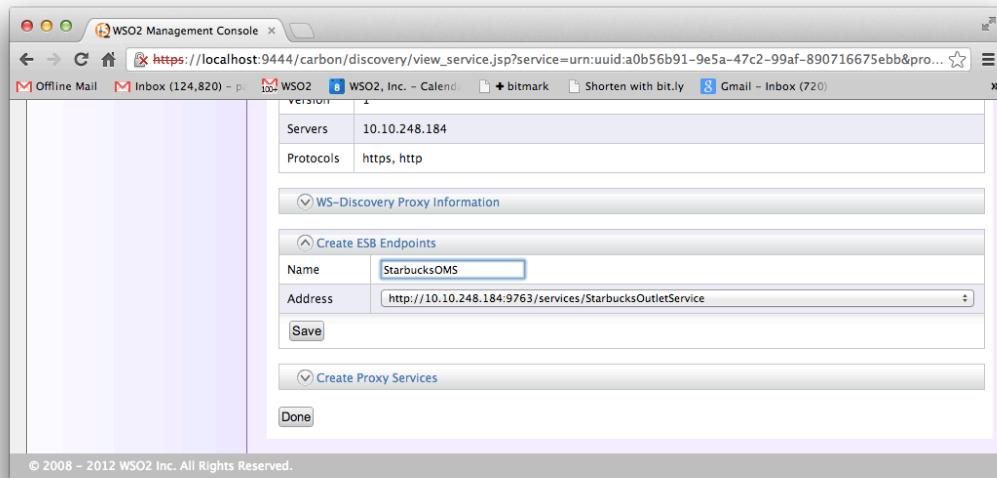
<http://10.10.248.184:9763/services/StarbucksOutletService>

30. Click on the UUID link

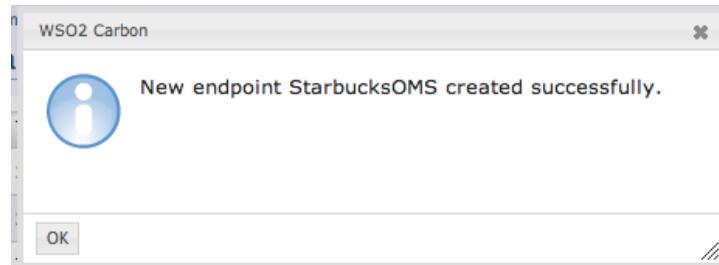


© Paul Fremantle 2012. Licensed under the Creative Commons 3.0 BY-SA (Attribution-Sharealike) license.  
See <http://creativecommons.org/licenses/by-sa/3.0/>

31. Go down to the section “Create ESB Endpoints” and expand it.



32. Give the name StarbucksOMS and hit Save. You should see:



33. If you switch back to the Manage tab and look at the endpoints, you should now find it is there.

34. That's all folks!

