**Exercise 3**

*Create and consume a SOAP Service using Apache CXF and JAX-WS*

**Prior Knowledge**

Played with SOAP and SOAPUI

Some Java coding skill

**Objectives**

Understand what it takes to create SOAP services. Compare the generated WSDL to the original SOAP.

Interact with the SOAP service using SOAP UI

Generate a SOAP client for your service

See how Maven and Tomcat can be used.

**Software Requirements**

(see separate document for installation of these)

* Java Development Kit 7
* Apache Maven 3.0.4 or later
* Eclipse Juno (4.2 SR1) or later – Java Development IDE
* Tomcat 7.0.30 or later
* SOAP UI 4.5.1 or later

Step 1. **Create a new project using Maven**

Maven is a very powerful (and somewhat arcane) build tool. We are going to use Maven to create and build our project.

Maven has the ability to create new projects using “archetypes”.

a. Go to your $oxsoa directory (see Exercise 1)

b. Open a command line and change to that directory

c. Test that you have maven properly installed. Execute

mvn -v

You should see something similar to this (dependent on your machine, JVM, etc)

Apache Maven 3.0.4 (r1232337; 2012-01-17 08:44:56+0000)

Maven home: /Users/paul/Apps/apache-maven-3.0.4

Java version: 1.6.0\_35, vendor: Apple Inc.

Java home: /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home

Default locale: en\_US, platform encoding: MacRoman

OS name: "mac os x", version: "10.8.1", arch: "x86\_64", family: "mac"

d. Use Maven to create a sample project:

Execute

mvn archetype:generate -Dfilter=cxf-jaxws-javafirst

This will prompt you for some choices:

Choose archetype:

1: remote -> org.apache.cxf.archetype:cxf-jaxws-javafirst (Creates a project for developing a Web service starting from Java code)

Select 1 (*type ‘1’ and hit enter!*)

Now it will ask which CXF **version** to use. The default is the latest (at the time of writing 2.7.0). *Just hit enter*.

Now it asks for a default **groupid**.

Define value for property 'groupId': :

This is a namespace, so choose something meaningful (e.g. com.mycompany.rest, uk.ac.ox.comlab.rest, etc). When creating this lab I chose *me.freo,* so that is what you will see in screenshots, etc.

Define value for property 'artifactId': :

This will be the name of the WAR and the overall maven artifact created. We are going to create a Birthday service, so type *BirthdayService***.**

Define value for property 'version': 1.0-SNAPSHOT: :

Make this 1.0

Define value for property 'package': com.mycom.restservice: :

This will default to the same namespace you chose for the groupid. That should be fine, so hit *Enter* to accept.

It will then ask you to confirm these settings. Hit *Enter* and it will go and generate the code.

You should see plenty of output explaining what is happening, and also a line showing where the resulting code was placed, e.g.:

[INFO] project created from Archetype in dir: /Users/paul/oxsoa/BirthdayService

This will have created a set of code and a tree structure for you. If you are on Linux you can use the nice **tree** command to show this:

$ tree

.

├── pom.xml

└── src

├── main

│   ├── java

│   │   └── freo

│   │   └── me

│   │   ├── HelloWorld.java

│   │   └── HelloWorldImpl.java

│   └── webapp

│   └── WEB-INF

│   ├── beans.xml

│   └── web.xml

└── test

└── java

└── freo

└── me

11 directories, 5 files

e. You can now build this code:

cd BirthdayService

mvn clean install

The first time this is run this will download a lot of stuff from the central maven repositories on the web. Depending how fast the network is, maybe a coffee is in order. You will need an active internet connection for this to work.

This will build **and test** the sample code. Its pretty cool. It actually starts an embedded Tomcat to run the service and call unit tests against it.

f. You can also build the Eclipse project for this too:

mvn eclipse:eclipse

This creates a project file that you can import into Eclipse with the right classpath, settings, etc.

g. Before you import the project, you do need to let Eclipse know where your Maven is installed.

You can do this manually in Eclipse by adding the M2\_REPO variable, but there is also a command line tool for this:

mvn -Declipse.workspace={path to eclipse workspace}   
 eclipse:add-maven-repo

On my Mac, the path to my eclipse workspace is /Users/paul/Documents/workspace

h. Once you have done this start (or restart) Eclipse, and then you can import the new project.

To do this, in Eclipse:

File -> Import -> General/Existing Projects Into Workspace->[Choose the directory where BirthdayService is]->Finish