**Server-side Web Services**

Mario G.C.A. Cimino

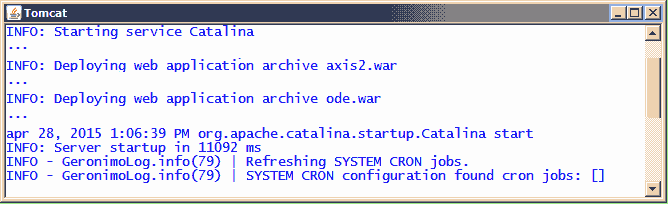
Department of Information Engineering

# Apache Axis and Apache Ode

* Download the *all-in-one* package and extract it to C:

http://www.iet.unipi.it/m.cimino/sse/res/sse\_extract\_to\_C.zip

* Apache6 Tomcat is a Java servlet[[1]](#footnote-1) container. To start/stop the web server use *c:\sse\hutdownApache* and *c:\sse\startupApache*



Periodic check on the status of services or processes

* The package contains the servlets *Apache Ode* and *Apache Axis2*, deployed as a WAR[[2]](#footnote-2)
* After startup, point the browser to *http://localhost:8080* to test the web server
* **Apache Axis2** is a Web Services / SOAP / WSDL engine

http://localhost:8080/axis2/ as a first test.

Click on *Services* to see the available ones.

* Open the c:\myapp\Calculator\Calculator.java file

**public** **class** Calculator **{**

**public int** add**(int** i1**,** **int** i2**) {**

**return i1 + i2;**

**}**

**public** b subtract**(int** i1**,** int i2**) {**

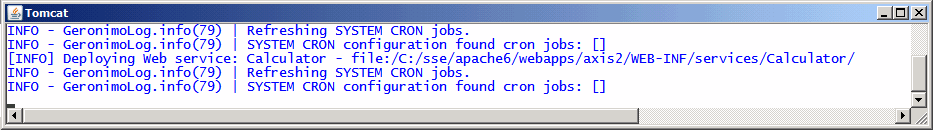
return i1 - i2;

}

}

* Click on *make.bat* to compile it. Copy the *c:\myapp\Calculator* folder to *C:\sse\apache6\webapps\axis2\WEB-INF\services*

The Tomcat console will show some deployment info



* Test the Calculator service on http://localhost:8080/axis2/ → click on *Services*.
* Click on the *Calculator* link:

[http://localhost:8080/axis2/services/Calculator?wsdl](http://localhost:8080/axis2/services/Calculator?wsdl%20)

You will get the WSDL description of the Calculator service.

*wsdl:binding*: network protocols and soap data format specification (document-centric or rpc)

*wsdl:service*: defines the endpoints (addresses) associated to the binding (wsdl:binding), with one or more logical port name

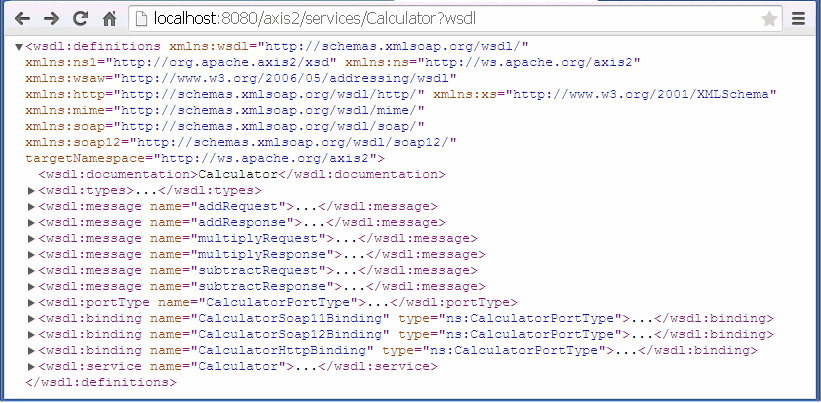
https://www.w3schools.com/xml/xml\_wsdl.asp

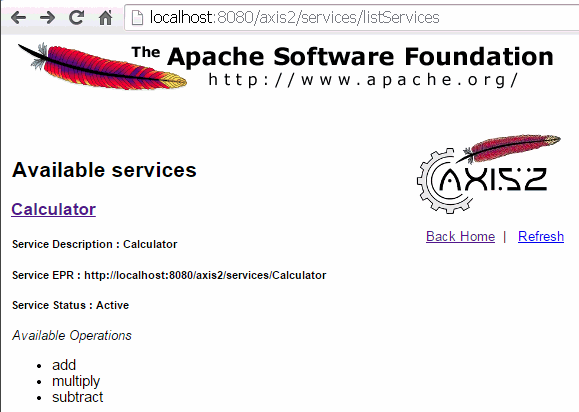
*wsdl:definition* : root element, with the target namespace and some other fundamental attributes

*wsdl:types*: data types used by operations, as input, output, and fault types. Often data types are specified using **XML schema**

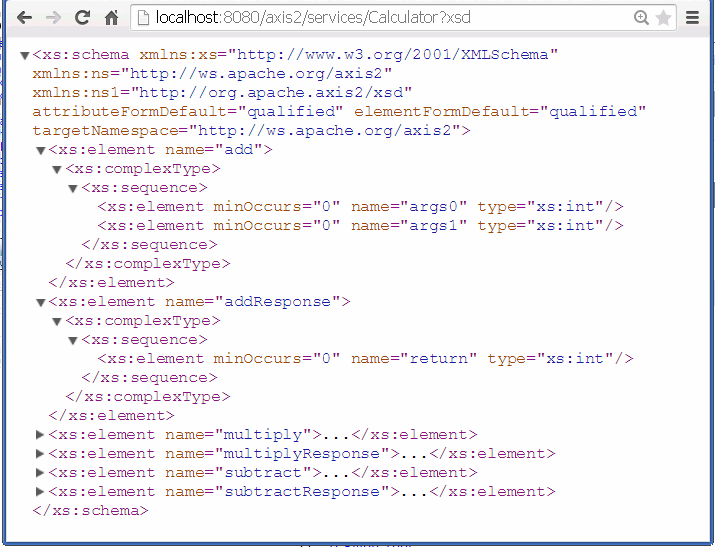
*wsdl:message*: defines the messages to exchange and the related types (wsdl:types)

*wsdl:portType* (interface) – defines operations with input and output which refer to the messages (wsdl:message)

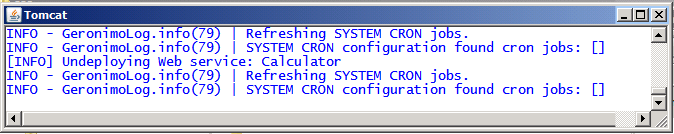




* Point the browser to *http://localhost:8080/axis2/services/Calculator?xsd* for data types



* Create a sample request by using SoapUI, as in the *Client-side WS* tutorial.
* Use it with sendsoap, as in the *Client-side WS* tutorial.
* Remove the *C:\sse\apache6\webapps\axis2\WEB-INF\services\Calculator* folder
* The Tomcat console will show some undeployment info



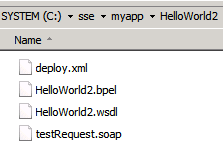
* Add a *divide* method to the *c:\myapp\Calculator\Calculator.java* file. Create and test the new web service.
* Create a web service with a new name, *DistilledWater*:

1. make a copy of the folder *c:\myapp\Calculator*
2. rename the folder *Calculator* as *DistilledWater*
3. in the file *META-INF\services.xml*, replace the two occurrences of the term *Calculator* with *DistilledWater*
4. rename the file *Calculator.java* as *DistilledWater.java*
5. update the internal java implementation of *DistilledWater.java*
6. Compile it by clicking on *make.bat*
7. Move the *c:\myapp\DistilledWater* folder to

to *C:\sse\apache6\webapps\axis2\WEB-INF\services*

* **Apache Ode** (Orchestration Director Engine) is a BPEL[[3]](#footnote-3) engine.

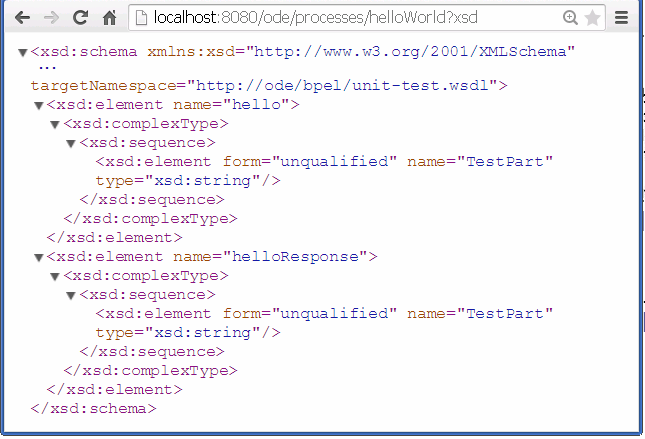
Point the browser to *http://localhost:8080/ode/* as a first test.

****

Click on *Processes* to see the available ones.

* Open the folder *c:\myapp\HelloWorld2*
* In the Apache ODE, business processes are written with BPEL standard.
* Copy the *c:\myapp\HelloWorld2* folder to *C:\sse\apache6\webapps\ode\WEB-INF\processes*

The Tomcat console will show some deployment info



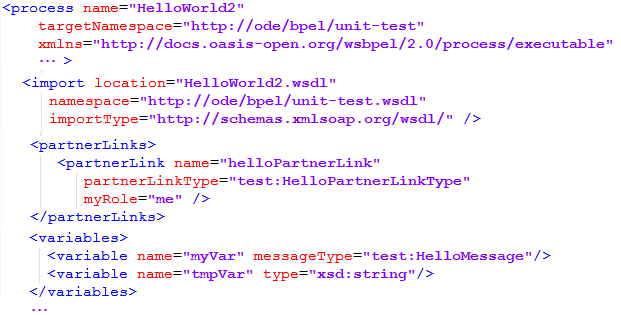
* Test the HelloWorld service on http://localhost:8080/ode/processes/helloWorld?wsdl

and

http://localhost:8080/ode/processes/helloWorld?xsd

* The service takes an input string (hello) and returns an output string (helloResponse)

**Open the .bpel file**

****

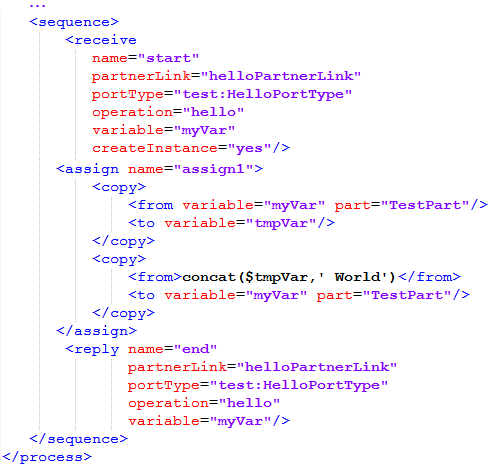
* BPEL has three basic components:

- Input/Output (WSDL)

- Data types (XSD)

- Programming logic (BPEL)

* Import the WSDL od XSD

****

* A PartnerLink is a web service portType
* Variables declaration (programming)
* Message Exchange: receive
* Variable assignment (programming)
* XPath Expression (programming)

http://www.w3schools.com/xpath/xpath\_functions.asp

* Message exchange: reply

http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html#\_Toc164738506

* The service concatenates the string ‘ World’ to the input message. Modify the service:

1. Remove the *C:\sse\apache6\webapps\ode\WEB-INF\processes\HelloWorld2*

folder. The Tomcat console will show some undeployment info

1. Replace the string ‘World’ with the string ‘Universe’ in

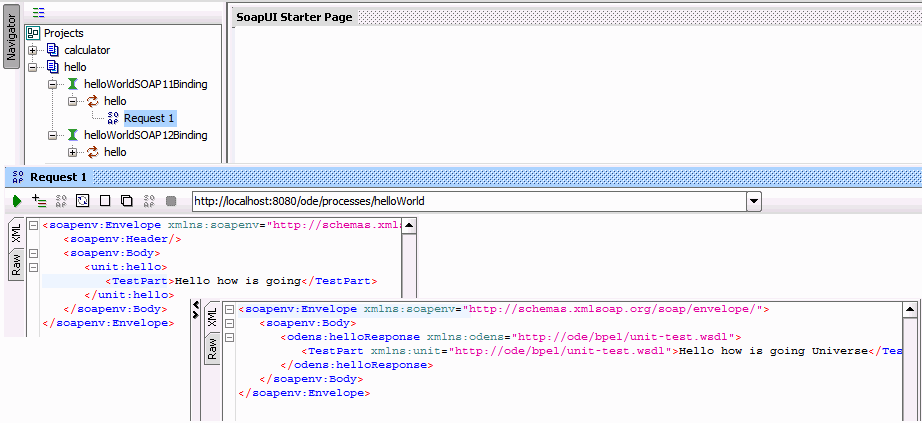
*c:\myapp\HelloWorld2\ HelloWorld2.bpel*

1. Copy the *c:\myapp\HelloWorld2 folder* to

*C:\sse\apache6\webapps\ode\WEB-INF\processes*

The Tomcat console will show some deployment info

* Create a sample request by using SoapUI.



1. Java server side components to extend the capabilities of an application server. [↑](#footnote-ref-1)
2. A WAR file (or Web application ARchive) is a JAR file used to distribute Java-based web applications. In general, to deploy a war, put it into the webapps folder of tomcat. [↑](#footnote-ref-2)
3. BPEL or WS-BPEL (Business Process Execution Language) is an XML standard to **orchestrate** web services. [↑](#footnote-ref-3)