WS-Guard

Enhancing UDDI Registries with Testing Capabilities*

PLASTIC consortium†

July 24, 2007

WS-Guard (WS-Guaranteeing Uddi Audition at Registration and Discovery) provides an implementation of the Audition Framework previously described in the deliverable 4.1. Scope of this chapter is to give technical detail on how the current implementation of the framework can be installed and used. However for a detailed description of the framework please refer to deliverable 4.1.

This short installation and usage manual assumes that the reader has a good knowledge on the basic standards enabling the WS vision. In particular to use the tool the reader has to be acquainted with technologies and standards such such as, XML, SOAP, WSDL and UDDI concepts.

1 Overview

The audition framework has been described in detail in the deliverable 4.1 and in a couple of scintific papers [6, 7]. Furthermore in deliverable 4.1 it has been illustrated the architecture of a real implementation of the framework with reduced capabilities with respect to the one presented in [6, 7]. The simplification concerns checks carried on by services receiving invocation by the service under evaluation.

Implementing the framework one of the main objective has been to reuse as much as possible available technologies and to refer to affirmed standards in the WS domain. To set up and use WS-Guard it is then necessary to have a basic knowledge and experience with the following technologies, in addition to the basic ones mentioned above:

- **Tomcat** [3]: Apache Tomcat is a web container developed at the Apache Software Foundation (ASF). Tomcat implements the servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. Tomcat also includes its own internal HTTP server.
- **jUDDI** [2]: this is an Apache open source Java implementation of the UDDI 2.0 specification. The standard UDDI functionality are exposed

*part of this work has been developed by Federica Ciotti working at her Master Thesis [8]

†for any inquiry please report to andrea.polini@unicam.it

using servlet based technologies requiring to set up a servlet container at priori. To store all the data structure foreseen by the UDDI specification jUDDI relies on the availability of a DBMS. In principle any DB for which JDBC drivers are available can be used for such purpose. Nevertheless the WS-Guard implementation has been tested only using a MySQL server.

- **UDDI4J** [4]: this is a Java class library that provides an API to interact with a UDDI registry. Using such a library it becomes much mode easier to interact with a UDDI server to publish and discover services.

- **Axis2** [1]: Axis2 is the Apache implementation of a SOAP container. Using Axis the developer will not be overwhelmed with the generation and parsing of SOAP messages. Development and deployment of a Web Service is extremely simplified using Axis. In particular service can be directly derived by a java class through the generation of a corresponding WSDL description (using an enclosed tool called Java2WSDL) and the definition of a WSDD (Web Service Deployment Descriptor) that provides directives to Axis on how to expose the defined interface and to invoke the implemented methods. At run time Axis is mainly composed of a servlet that then have to be deployed within a servlet container such as Tomcat.

WS-Guard applies known methodologies for model based testing derivation and execution. In particular it assumes that a Service State Machine is available for the service under registration. Such SSM will be provided to a service encapsulating the JAmbition library and that at the same time will be able to make invocations on the servive under evaluation.

2 Technical description

In order to set up the environment to run WS-Guard all the technologies referred above have to be downloaded and correctly installed. In particular WS-Guard assumes the availability of a correct installation of the Apache Tomcat servlet container, on which jUDDI and Axis2 have been previously deployed. At the same time, in order to correctly derive test cases and execute them against the service under evaluation, it is assumed that an instance of a tester service, encapsulating JAmbition, is available at the following address:

`http://pc-dispo5.isti.cnr.it:8080/Ambition2/AmbitionService`

As described in deliverable 4.1 WS-Guard relies on an augmented description model for registered services providing, in particular, behavioural definitions of the service. In the current implementation the model is specified using Service State Machine (SSM). Such models are stored in the WS-Guard registry through the use of tModel data structures as defined in the UDDI 2.0 specification. For detail on SSM specification see [5].

SSM format contains information that put in relation a behavioural specification with a service port implementing it. Figure 1 shows the usual mapping of WSDL data to UDDI data structures. However in order to explicitly include

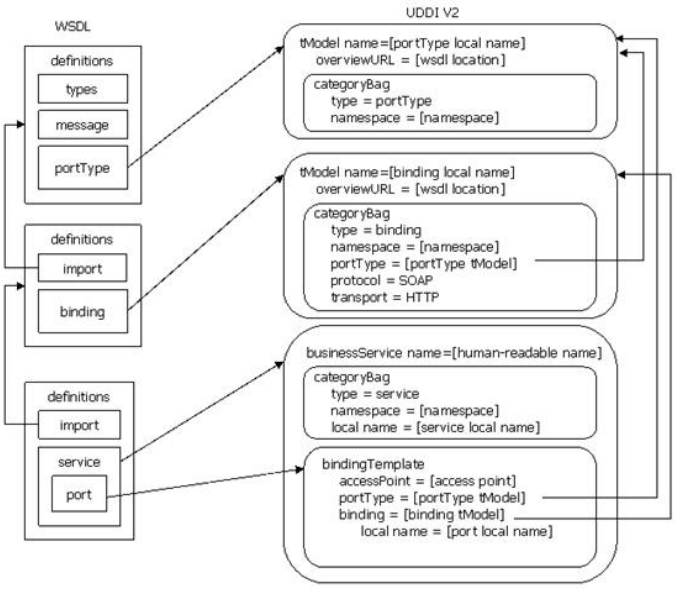

Figure 1: WSDL to UDDI mapping

support for SSM specification within a UDDI registries it has been necessary to revise the mapping that has been redefined according to what is shown in Figure 2.

The additional information and relation are managed by the WS-Guard implementation and no additional effort is required to developers of service to be registered within a WS-Guard registry.

Finally in order to recognize invocations on the inquiry interface fired by a service under evaluation WS-Guard needs to recognize the sender of the inquiry message. In general SOAP messages are sent without including this information and the sender of a message can be recognized only at the transport layer, for instnace through specific fields of the HTTP packet. Nevertheless WS-Guard require to manipulate this information at the application level. To do this it assumes that each message exchange with possible clients follow formatting rules defined by the WS-Addressing specification [9]. Figure 3 shows how, according to the "WS-Addressing - SOAP binding" W3C recommendation [10], the sender can be specified within the header of a SOAP message.

The current implementation of WS-Guard has been tested using the following configuration:

- Sun Java version jdk 1.5.0_11 (with version 1.6.* we experimented some problems with the combined use of UDDI4J)
- Apache Tomcat version 6.0.7

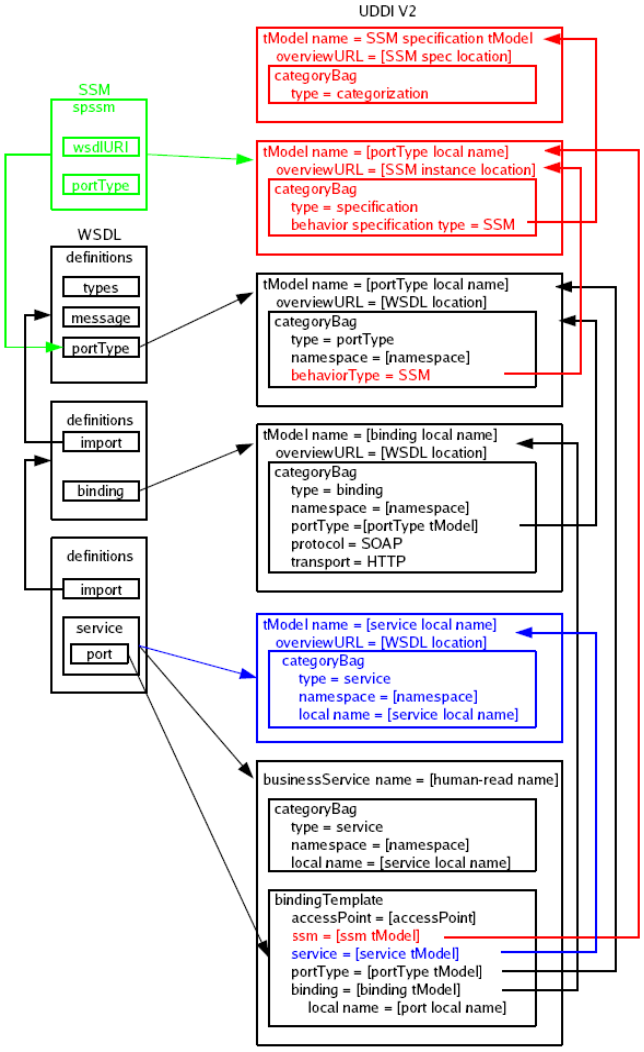

Figure 2: SSM/WSDL to UDDI mapping

- Apache Axis2 version 1.1.1
- Apache jUDDI version 0.9rc4
- UDDI4J version 2.0.5
- MySQL version 5.0

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
            xmlns:wsa="http://www.w3.org/2005/08/addressing">
   <S:Header>
    <wsa:To>http://firm1.com/ValutationService</wsa:To>
    <wsa:Action>http://firm1.com/ValutationService</wsa:Action>
    <wsa:MessageID>6B29FC40-CA47-1067-B31D-00DD010662DA</wsa:MessageID>
    <wsa:ReplyTo>
      <wsa:Address>http://holdingCompany.com/clientService</wsa:Address>
    </wsa:ReplyTo>
   </S:Header>
   <S:Body>     ...
   </S:Body>
</S:Envelope>
```

Figure 3: Sender reference specification within a SOAP header

2.1 Installation

Given the pre-requisites specified in the previous sections, concerning the availability of a running tomcat server on which required libraries have been deployed, the installation of WS-Guard is extremely easy. WS-Guard has been implemented as a standard Web Service, therefore using it, will only require to deploy the file containing the service on the correct tomcat directory. In particular to run WS-Guard you have to perform the following steps.

1. deploy the service into Tomcat copying the `UDDIProxyService.aar` in the folder:
   `apache-tomcat-6.0.7/webapps/axis2/WEB-INF/services`

2. Check that the deployment has been correctly performed. If your Tomcat configuration allows hot-deployment just open the browser at the address:
   `http://localhost:8080/axis2/services/listServices`
   In case hot-deployment is not supported restart the server and access the page previously indicated.

3. Since `UDDIProxyService.aar` is an Axis2 service group containing both the Publish and the Inquiry UDDI Web service, the two services should be present in the service list as two separate services, as show in the screenshots of Figure 4 and 5.

The content of the `UDDIProxyService.aar` can be explored using any program suitable for compressing files. The content is structured in 5 main directories. The `lib` directory contains the library required by the WS-Guard service to correctly run. The directories `org`, `uddi_org`, and `utils` contain the implementation of the service. The structure of the file follows the guidelines defined for the deployment within the Axis2 container. The deployment into another WS technology will probably require to revise the file structure.

Finally in order to create, within the DB, the necessary data structure to store and manipulate SSM information, it is necessary to run the script:
`ssmTmodels.sql`

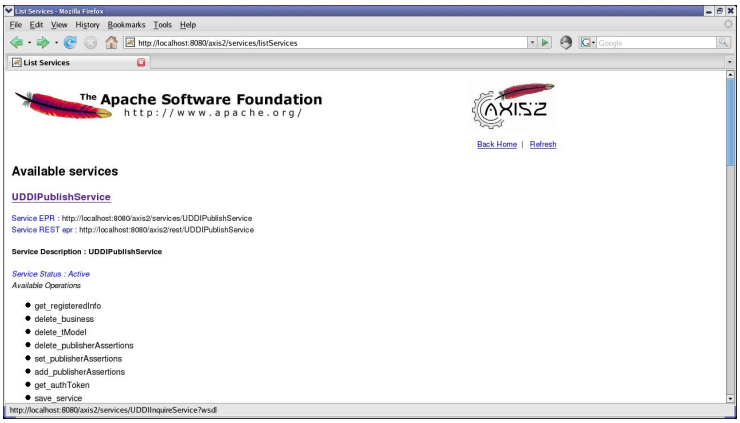

Figure 4: WS-Guard publish interface

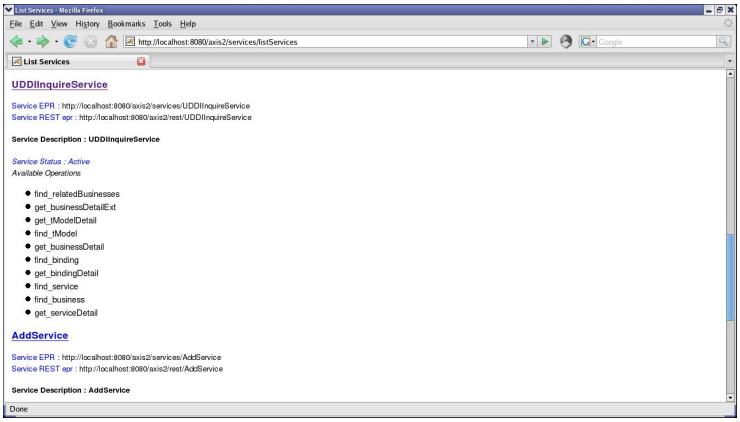

Figure 5: WS-Guard inquiry interface

2.2 Usage

WS-Guard provides a ready to use UDDI registries with testing capabilities. Differently from a jUDDI registries functionality are exposed directly with web service interfaces. The interaction with WS-Guard have to follow the rules defined above in particular SOAP messages must contains the sender address in the header section.

The directory `client` contains the code for a generic simple client that can be used to acquire a better understanding on the usage of the WS-GUard registry. Under Linux the client can be launched using the command:

```
sh startClient.sh
```

available in the directory `client`

3 Planned improvements

WS-Guard will not be further developed within PLASTIC. Currently among the various case studies under development within PLASTIC, no one seems to show the necessity for having a directory service with testing capabilities. This chapter has been enclosed mainly for completeness with respect to the testing phases illustrated in the deliverable 4.1. Nevertheless there are several possibilities to extend the current implementation of the framework in particular in order to include and improve mechanisms for usage based checking. Such mechanisms refer to the possibility that services invoked by the service under audition check if it make correct invocation with respect to the protocol (order of messages) and/or expected pre-conditions. Exptensions/improvements to the current implementation of the framework will have to be planned in case WS-Guard will be adopted in one or more case studies.

References

[1] Apache axis2. on-line at: http://ws.apache.org/axis/.

[2] Apache jUDDI. on-line at: http://ws.apache.org/juddi/.

[3] Apache tomcat. on-line at: http://tomcat.apache.org/.

[4] UDDI4J. on-line at: http://uddi4j.sourceforge.net/.

[5] AA.VV. Plastic deliverable 4.2. Technical report, ISTI-CNR, July, 31st 2007.

[6] A. Bertolino, L. Frantzen, A. Polini, and J. Tretmans. Audition of Web Services for Testing Conformance to Open Specified Protocols. In R. Reussner, J. Stafford, and C. Szyperski, editors, *Architecting Systems with Trustworthy Components*, LNCS 3938, 2004.

[7] A. Bertolino and A. Polini. The Audition Framework for Testing Web Services Interoperability. In *Proc. 31st EUROMICRO Conf. on Sw Eng. and Advanced Applications (EUROMICRO–SEAA 2005)*, pages 134–142. IEEE Computer Society, 2005.

[8] Federica Ciotti. WS-Guard —Enhancing UDDI Registries with on-line Testing Capabilities. Master's thesis, Department of Computer Science, University of Pisa, June, 8th 2007.

[9] W3C. Web service addressing (ws-addressing) specfication. on-line at: http://www.w3.org/Submission/ws-addressing/, 2004.

[10] W3C. Web Service Addressing 1.0 - SOAP Binding. on-line at: http://www.w3.org/TR/ws-addr-soap/, 2006. W3C Recommendation.