Intelligent Framework for Web-Service Testing

L. S. Rajiv Krishna, Y. Prasanth
Dept. of Computer Science Engineering, K L University, India

Article Info

Article history:

Received Dec 14, 2014
Revised Feb 14, 2015
Accepted Feb 22, 2015

Keyword:

Service Oriented Architecture (SQA)
Test management
Web services

ABSTRACT

Web services provide a distributed computing architecture, with an emerging way of service oriented architecture (SQA). Here service oriented architecture is an interface to both computer systems and web services. Which implement an interaction with each other in new and different ways. According to service oriented architecture it virtually provides a platform for web services to communicate with each other. As it was an easy way for communicating with both clients and services. Many organizations and companies are either evaluating themselves into an enterprise information architectures, or they are in the process of getting adopt to the web services technology. As web services are platform independent it is playing a major role in the enterprise environment, and currently web services are widely accepted by many companies and organizations. So commonly web services possess some challenges to the enterprise environment. As a part of it web service must be tested before publish into a service oriented architecture. It involves large number of test cases, test scenarios that takes more time and effort. Testing management is needed so that it should control the time effort and should reduce the complexity of web service in a large software system, also in a real time world. Automation testing faces these challenges and fixes these issues. Automation testing has an ability to handle the complexities which are experiencing by the web services in a current environment. This paper present the automatic testing strategies of a web service and detect the problems between both manual and automation testing. Finally results shows the proper effective report on improving the visibility of testing process based on the web approach to enhance the critical communication among multiple testing groups.

Corresponding Author:

L. S. Rajiv Krishna,
Department of Computer Science Engineering,
K L University,
Vaddeswaram, Guntur District, India.
Email: satyarajivkrishna@outlook.com

1. INTRODUCTION

In spite of focusing on requirement verification, design reviews, coding structure, and software defects all these are inevitable. Automation testing mainly focuses and ensures on software quality. The quality of software can be measured based on how it works on critical conditions, jitter control, and the no. of users stress on the service, data management and execution algorithms etc. So proper execution frame work must be designed for resolving these critical conditions. Web services can be developed by one company, can be used by another company and also may be hosted by the third party. For example let's take a telecom company, allows its customer's for online recharge. So this company leverage's the search services provided by the search engines to implement the functionalities which were got published on this Bing web service portal. The below Figure 1 shows the web service stack so that automation testing can be made according to this criteria. '

1.1. Business Rules

1.1.1. Semantics of Business Vocabulary and Business Rules: (SBVR)

According to OMG (Object Management Group) document number 2013-11-04, defines some rules and protocols for exchanging of business vocabulary among the organizations between software tools. These SBVR specifications is applicable to domain business vocabularies and rules for all kinds of business activities of organizations.

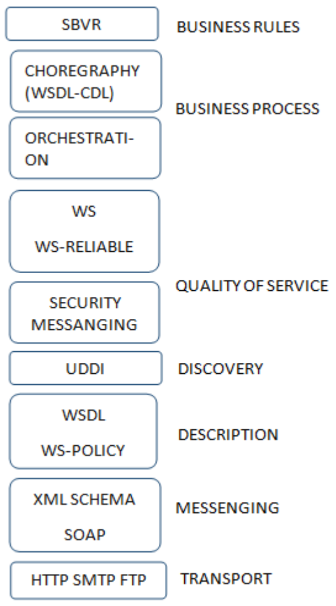

Figure 1. Web service

It provides an un-ambiguous, meaning-centric, multi-lingual, and semantics for rich capability for defining meanings of the languages used by the people in an organization, and also industries for professionally maintaining disciplines among the organizations.

1.2. Business Processes

1.2.1. WS-Choreography Definition Lang-uage (WS-CDL) and Orchestration (BPEL, OWL-S)

WS-CDL is a working group which is hosted by w3c. WS-Orchestration is a definition language. Here this working group is an articulation between Orchestration (BPEL) and Choreography. BEPL is said to be programming language which is used to specify the behaviour of the participants in a choreography, where choreography is meant for describing messages among the participants. It was mainly concerned with the internal process communication of a web service.

1.3. Quality of Service

1.3.1. WS-Reliable Messaging

This is a messaging service, which is used to deliver SOAP based messages among the distributed applications. These SOAP based messages were entirely different from the Http messages.

1.3.2. Secure Messaging and Web Service Transaction

They work under the category of WS-Reliable message. Their major role is to send and deliver messages securely and reliably. Here is a small architecture for reliable message modeling.

1.4. UDDI (Universal Discovery and Description)

UDDI is publish and discovery agent for a web service, It is said to be a registry log, where whatever web service is published in UDDI. It is been discovered by a user agent and utilizes that service. UDDI is a platform independent and XML based registry. Which is used by the business world and can publish themselves on the internet, and also can list that particular web service in a registry pro log.

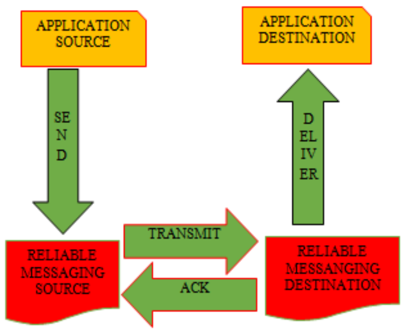

Figure 2. Reliable Messaging Model

1.5. Description

1.5.1. Web Service Policy

Web service description (WSP) is a language which has got a framework and Web Service Policy Attachment Specification (WSAPS). They used to provide a complete description of a web service, based upon some policy language protocols.

1.5.2. Messaging

XML-Schema is an interface for SOAP (Simple Object Access Protocol), which implements a reliable and platform independent messaging services between web applications. Here schema provides both structure and constraints of a soap protocol.

1.6. Transport

1.6.1. HTTP

It is a standard method called Hyper Text Transfer Protocol. Which is used to transmit Request and Response over HTML in a standard format. This protocol uses TCP/IP to manage web transmissions.

1.6.2. SMTP (Simple Mail Transfer Protocol)

It is used for configure the email clients and their addresses.

1.6.3. FTP (File Transfer Protocol)

FTP as it name says that it provides a method for copying files over network from one computer to another. It mainly handles the server side traffic.

Based upon the Figure 1 testing a web service is a critical part. In a software development life cycle, testing is very important, because it achieves the quality of software, based on correctness and robustness problems. Manual testing seems to be very difficult at this criteria, as a part of it takes more time and effort to execute or test a single test case. So tester thought that we need an automation tool for testing these composite web services, which possess the quality of software, with automation testing the main advantage is it reduces time, effort and cost more over it increases the quality of software product. Also automation testing works better than manual testing. As web services are platform independent and most probably they contain some coding part like XML, XSD's, SOAP, UDDI etc..! Here is a sample code for web service. This sample code possess the correct definition of a web service i.e. Services are legally platform independent which are derived based on user defined standard elements that can be accessed by the autonomous server. They are published themselves and discovered upon the client request, and were programmed using standard tools and protocols in order to build a distributed network applications over across and within the organizational boundaries.

1.7. Sample Code of a Web Service

Web-service.java

```
import javax.jws.WebMethod;
import javax.jws.WebService;
import javax.jws.soap.SOAPBinding;
import javax.jws.soap.SOAPBinding.Style;

@WebService
```

```
@SOAPBinding(style = Style.RPC)
public interface WebServiceInterface {
@WebMethod
String printMessage();
}
```

Web-service-impl.java

```
import javax.jws.WebService; @WebService(endpointInterface =
"com.javacodegeeks.enterprise.ws.WebServiceInterface")
public class WebServiceImpl implements WebServiceInterface{
@Override
public String printMessage() {
return "Hello from Java Code Geeks Server";
}
}
```

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun Microsystems,
Inc.//DTD Web Application 2.3//EN"
"http://java.sun.com/j2ee/dtds/web-app_2_3.dtd">
<web-app>
<listener>
<listener-class>
com.sun.xml.ws.transport.http.servlet.WSServletContextListener
</listener-class>
</listener>
<servlet>
<servlet-name>sayhello</servlet-name>
<servlet-class>
com.sun.xml.ws.transport.http.servlet.WSServlet
</servlet-class>
<load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
<servlet-name>sayhello</servlet-name>
<url-pattern>/sayhello</url-pattern>
</servlet-mapping>
<session-config>
<session-timeout>30</session-timeout>
</session-config>
</web-app>
```

Jaxws.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<endpoints xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime"
version="2.0">
<endpoint name="WebServiceImpl"
implementation="com.javacodegeeks.enterprise.ws.WebServiceImpl"
url-pattern="/sayhello" />
</endpoints>
```

2. FRAMEWORK

We had developed a frame work for testing a reliable robustness applications in order to achieve Quality of service QoS for a software product. This frame work was divided in to 4 parts as testing will be done in 4 stages and each stage has some phases to handle the complexity of web service.

And these four stages are: 1) Modeller agent, 2) Dash board agent, 3) Test execution agent, and 4) Response analyser agent. This tool takes a wsdl file as input and generates necessary test cases, these test cases will be executed individually and checks for errors. If so it results a runtime error, otherwise test cases results successful. Mainly this frame work rectifies the hard coded part and searches for un-integral parameters between functions.

2.1. Modeller Agent

Modeller agent is an initial stage of the framework. It takes wsdl file as input and facilitates it to code generator component. The results of the web service was based upon the code generator component as it generates entire code for a given web services. As it is said to be a core component of this frame work. As code generator component takes wsdl file as input it generates both client code and necessary test classes also known as test cases. Here client code is said to be the code which is been executed and accessed by the client, which said to be browser browsing part. And classes are said be a blue print architecture of a web service which decides the flow of execution of a web service.

Here client code contains only requests and response part. Whereas test classes contain entire classes that are listed in the web service.

2.2. Dashboard Agent

It is a main stage for testing a web services because dashboard agent will generate a test parser by taking input as generated test classes. Whereas test parser will play a major role in test execution agent. Here test parser contains the data flow of a web service, so that the test execution agent will execute based upon these test parser. Test parser is said to be a test suite which contains Junit test cases. Junit contains some different classes that can run all together using annotations i.e. @Run and @Suit.

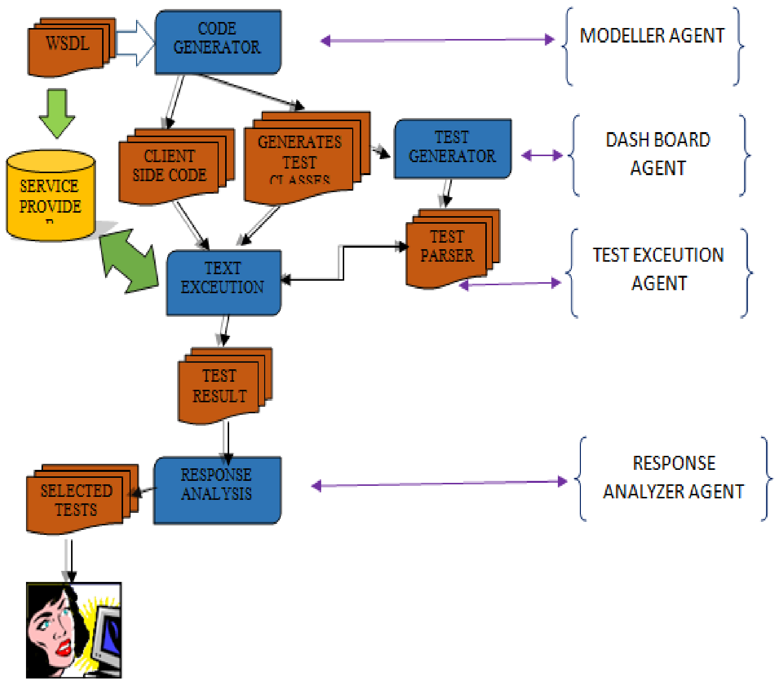

Figure 3. Overview of Automation Testing Framework

As we know Junit is the unit testing frame work for JAVA language, so it seems, to make task very easy and simple. Some of the features of Junit are:

1. It provides a @annotation based classes to identify the test methods.
2. It try to assert test for expected results.
3. Junit will reduces the complexity of test classes.

2.3. Test Execution Agent

Test Execution agent is the middleware of this framework. Where test execution agent acts as an interface between dash board agent and response analyser agent. This execution agent contains a test execution component, a service provide which is connected to this component which serves input as wsdl file to it, and also both inputs and outputs of the dash board agent. And these all will facilitate input to test execution component and generates output as test results. It mainly acts as a core part of this frame work. Here test execution component contains some test execution methods, protocols, and standards. Some of the test execution methods are:

1. Functional and Non-functional testing
2. Regression Testing
3. System testing
4. Software testing
5. Integration testing
6. Unit testing
7. Database testing and
8. Security testing

These all testing methods makes software as validation. In order to validate the software product it typically six steps:

1. It identifies the function (@ annotations) that a particular web service wants to perform.
2. It identifies the boundaries of function and parameters, based upon the client's inputs.
3. It identifies weather it (web service) was generating a valid output or not based upon the clients request.
4. It determines the execution flow of test case.
5. It compare's both actual and expected output of the web service.
6. It checks whether the application works as per the customer needs or not.

2.4. Response Analyser Agent

It is an end agent of this frame work. This agent contains a response analysis component where it takes input as test results which was generated by test execution component and automatically generates a required selected tests of that web service. Based upon the selected tests the response ratings will be decided by the tester. The response analysis results was based on the response time of requests that are generated, and the amount of time taken to respond a request after submitting the process. Based upon the test results it (frame work) automatically categorizes the operation into lower, higher, medium, complex and complex high. Let us consider if the response time for 4 attributes is recorded as 896ms, then the category of the web service was based upon the business logic. Here the number of attributes were decided based on the combination of both inputs and outputs of a particular web service. Now let's talk about the response analysis time for getting an Assembly bhavan as response upon the request. So input is given as longitude and latitude of the assembly bhavan i.e. 28lat and 32lon. As soon as the input was given to the web service it must provide the address of the assembly bhavan based on longitude and latitude i.e. $28^{o}32^{|}$N Assembly Bhavan jubilee Hills Telangana. It is called as assembly response.

Our frame work does not expect that tester must have some knowledge regarding technology. It automatically generates right responses when the right requested was submitted. This frame work accepts both black-box and white-box testing. So the automatic response time i.e. minimum, maximum and average was based on TPS and BPS values.

3. CONCLUSION

This paper provides flexible test for test frame work for java based web services using the features of testing and test control notations. This frame works for the web services which are developed on SOAP and XML. This paper contains a new and variety of new and interesting challenges for testing the web services. It is an hierarchy based frame work were testing can be done in four stages i.e. very effectively and efficiently. There are some other issues that will be handle in future. They are:

1. Testing must be performed based upon the frequency of testing.
2. Test must be done only based upon the particular operation that is to be tested.
3. Recursive testing need to be implemented at every stage in test execution component

This survey has focused on mainly brief introduction on web service stack, difference between manual and automation testing, and a frame work for automation testing of a web service which was developed on JAVA, XML and SOAP.

REFERENCES

[1] Apache web service project – Apache Axis, http://ws.-apache.org/axis/.
[2] Web services protocol stack - Wikipedia, the free encyclopaedia, http://en.wikipedia.org/wiki/Web_services_protocol_stack.

[3] Simple and fast .NET and Mono Open Source web services framework, https://servicestack.net/.
[4] Functional Tests of Web Services, http://www.soapui.org/Getting-Started/web-service-sample-project/3-Functional-Tests-of-Web-Services.html.
[5] Web Services Testing, http://www.-cs.colorado.edu/~kena/classes/7818/f06/lectures/WebTest.pdf.
[6] Using Web services Reliable Mess-enging, https://docs.oracle.com/cd/E21-764_01/web.1111/e13734/rm.htm#WSADV275.
[7] Web services Messaging, http://-docs.oasis-open.org/ws-rx/wsrm-/200702/wsrm-1.1-spec-os-01-e1.pdf.
[8] Container Authentication with JAX-WS, http://www.tuicool.com/artic-les/ZnQJze.
[9] Antonia Bertolino, "Software Tes-ting Research: Achievements, Challen-ges, Dreams," Future of Software Engineering (FOSE'07), IEEE Computer Society. 2007.
[10] Mike P. Papazoglou, Willem-Jan van den Heuvel, "Service oriented architectures: approaches, Technolo-gies and research issues", The VLDB Journal- Springer, pp. 389-415, 2007.
[11] Chunyan Ma*, et al.*, "*WSDL-Based Automated Test Data Generation for Web Service*", 2008 International Conference on Computer Science and Software Engineering, 2008.
[12] X. Bai, *et al,*, "*Ontology-based test modeling and partition testing of web services*", In ICWS '08: Proceedings of the 2008 IEEE International Conference onWeb Services, Beijing, China, pp. 465-472, Sept 2008.
[13] B. Banieqbal, *et al.*, "*Temporal Logic in Specif-ication*", Proceedings of Lecture Notes in Computer Science, Springer, Altrincham, UK, Vol. 398, 1989.
[14] C. Benedetto, "SOA and integration testing: The end-to-end view", September 2006.
[15] vCPNTOOLS, http://w-iki.daimi.au.dk/cpntools/cpn-tools.wiki.