# Framework for Testing Web Services Through SOA(*Service Oriented Architecture)*

Y.Prasanth[1],V.Sarika[2], D.Santhosh Anuhya[3] , Y.Vineela[4] , A. Ajay Babu[5]

*[#]Dept. of Information Science & Technology , Koneru lakshmaiah college of Engineeering*
*Green Fields -522502,Guntur, Andhra Pradesh, INDIA*

*Abstract*—**In today's connected e-world Web Services is arguably the most popular and powerful software technology. This paper focuses on web services and also the various standards that have evolved over the years, such as XML, SOAP, WSDL, and UDDI. This paper advocates the need for testing web services as testing web services poses a big challenge to testing professionals because of its inherently complex and distributed nature. Web Service Automated Testing Utility will enable us to locate and invoke web service methods directly. It supports all of the core Web service technologies like WDSL, SOAP, and it is an ideal Web service tool for testing Web services, inspecting WSDL files, automating or accelerating verification of each component when developing Web service enabled applications.**

*Keywords-**component web service; WSDL; SOAP; functional testing.***

## I.INTRODUCTION

The web has been embraced by millions of businesses as an inexpensive channel to communicate and exchange information with prospects and transactions with customers. It is a highly programmable environment which allows mass customization through the immediate deployment of a large and diverse range of applications, to millions of global users. But a web application can be enhanced using web services. Simply, web services make application functionality available over the internet in a standardized, programmatic way. Web services utilize existing IT infrastructures and allow companies to wrap legacy applications in a standardized, consistent and reusable format so every investment can be leveraged.

.

## II.WEB SERVICES

The layman's definition of web services can be, an application that provides a Web API. An API which provides communication using Extensible Markup Language(XML) and the web is called Web API. In today's connected e-world ,Web Services is arguably the most popular and powerful software technology. For integrating applications on the web , Web Services promise Internet enabled modular, standards based, platform independent and language method. Web services are a new form of middleware based on XML and the Web. The limitations/drawbacks of traditional middleware technologies are Usage difficulty, Lack of heterogeneity , Expensive, Maintenance difficulty ,Tight coupling with the solutions they work for, Lack of support for complex distributed systems etc.,

The issues mentioned above can be solved by the Web Services as they are platform and language independent. A web Service that is deployed in one language can be deployed on any platform. Once a web service is deployed, it can be accessed by any other application regardless of language or platform.

### A. *Web Service Standards*

Web services has its basis on a set of standards, though it has an independent architecture. The set of standards enable the web services to be scalable and interoperable. Apart from the most populary used XML, the web services standards that have evolved over the last few years is given an overview.

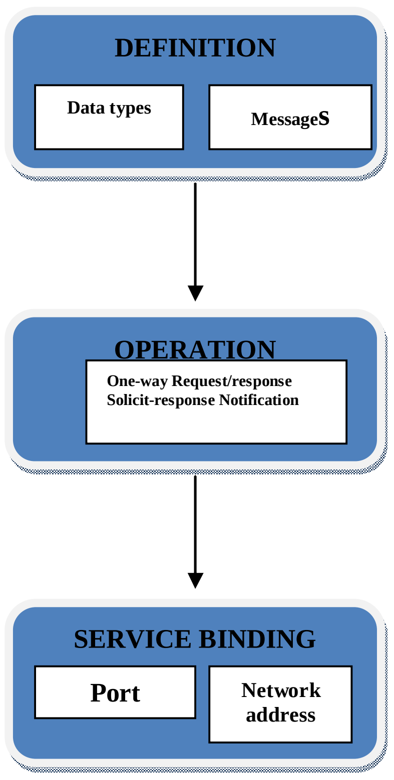
### *B. WSDL (Web Service Description Language)*

WSDL means Web Service Description language, it is an xml based language used to describe the services and access to those services. There are 3 parts of WSDL they are

1) Definition

2) Operation

3) Service binding



*Parts of WSDL*

There are four major elements to describe the WSDL document they are as follows

*Type:* The data types of the message used by the web service

*Message:* The message to be used by the web service

*Port Type:* The operations performed by the web service

*Binding:* The communication protocol and defines message format of a web service

The WSDL document structure

<definitions>
<types>

Defines data types used by the service

</types>

<message>

Defines data elements of an operation
</message>

<portType>

Defines the operations performed by the web service

</portType>

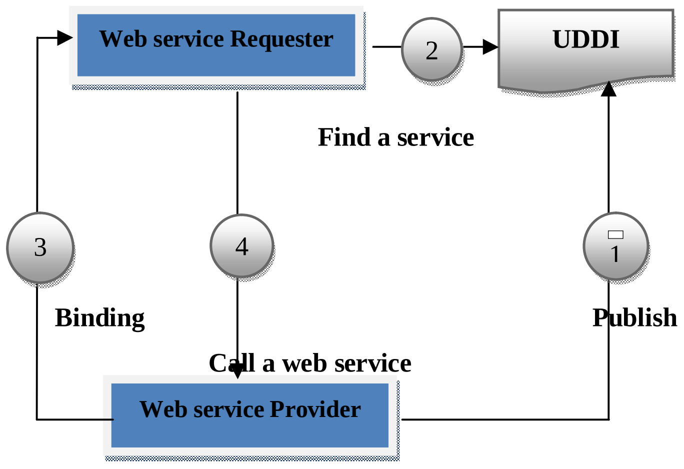
<binding>
Defines the communication protocol and defines message format of a service

</binding>

</definitions>

C. *UDDI* : (Universal Discovery, Description and Integration Language)

The UDDI registry is intended to eventually serve as a means of "discovering" Web Services described using WSDL .



*Web service provider*: The provider describes its service using WSDL. The person or an organization wants their web services to be used by others, they has to register themselves which is called UDDI registry where the consumer can search the required web service.

*Web service Requester*: The consumer who wants a required web service then he has to search the registry. If he finds the required web service then he can bind with provider of that service.

D. *SOAP:*

In a distributed environment for exchanging information a light weight protocol SOAP is used. Using XML, SOAP provides a mechanism for exchanging information between peers. . As such SOAP does not define programming model or implementation details. It just defines a simple mechanism for expressing application semantics.

### E. *XML:*

XML is Extensible Markup Language, a new language developed for web is different from other different types of scripting and programming languages. The main purpose of xml is not for processing and displaying data but to store data or information. It can be highly structured data like data stored in database or spreadsheets and loosely structured like data stored in letters or manuals.

*A Sample XML code:*

```
<? xml version="1.0"?>
<person>
 <name>
  <firstname>geetha</firstname>
  <lastname>Madhuri</lastname>
 </name>
 <job>Singer</job>
 <gender>Female</gender>
</person>
```

XML is free and it doesn't cost more since it can be written in a normal notepad or XML notepad.

*XML Documents*: An XML document is made up of following parts:

#### a) *XML Prolog:*

XML prolog consists of XML declaration, Processing instructions, Comments, Document Type Declaration(DTD).

*(i) XML Declaration:*

It is the first line of the document and it should not be preceded by any white spaces.It is represented as:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

An XML document can be recognized by the above declaration. Here the version attribute is mandatory and remaining two may or may not be present.

*(ii) Processing instructions:*

These are used to pass the parameters to an application which tells the application how to process the XML document. For example let us consider the following processing instruction which converts XML document into XSL style sheet "beatles.xsl".As shown below the processing instruction starts with "<?" and ends with "?>".

<? Xml-style sheet href="beatles.xsl" type="text/xsl"?>

*(iii)Comments:*

Comments in an XML document can appear anywhere and they are represented as :<!—and end with ---!>.

*(iv) Document Type Declaration:*

It mainly performs three roles. They are:

1. It species the name of a document element. The document element beatles of XML document can be specified can be specified by DOCTYPE declaration as:

<! DOCTYPE beatles>

2. It may point to any external Document Type Definition. If it points to an external DTD then it has to specify whether the DTD is on the system on which the XML document is present itself or in some public location. In order to differentiate between these two DTD's it uses two keywords like SYSTEM and PUBLIC. Then it points to location of DTD using Uniform Resource Indicator (URI) or an absolute URI. Here are some examples:

*Syntax when DTD is on same system:*

<!--DTD is on the same system as the XML document-->

<! DOCTYPE akiras SYSTEM "dtds/akiras.dtd">

*Syntax when DTD is on some public Location:*

<!--DTD is publicly available-->

<! DOCTYPE beatles PUBLIC "-//VENNELA//DTD akiras 1.0//EN"

"http://www.VENNELA.com/AKIRA/DTD/akiras.dtd">

In the above declaration public identifiers are divided into three parts. They are:

(i).An organization (eg: VENNELA)

(ii).A name for DTD (eg: akiras 1.0)

(iii).A Language (eg:EN for English)

#### b) *XML Element:*

Every XML document should have atleast one element, which is called Document element. Each document element contains other elements, which contains other elements, and so on. Let us consider the same example as we considered above like person is the document element is person which consists of three elements: name, job, gender.The name element consists of first name and last name elements. Elements which are nested within another element are called children.

#### c) *XML Attributes:*

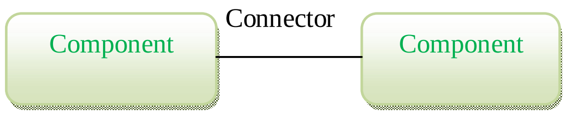
XML elements can further defined with attributes which appear in the element's open tag. For example,

```
<name title="Sir">
 <firstname>yalla</firstname>
 <lastname>prasanth</lastname>
</name>
```

*F. SOA:*

For the systems that are loosely coupled and interoperable web services create an environment. The concepts for web services come from a conceptual architecture called service oriented architecture (SOA). To maximize loose coupling and reuse, SOA configures entities like services, registers, contracts, and proxies. To better understand the SOA; first it is important to explore the concept of software architecture. Software architecture consists of the software's coarse grained structures describing the systems components and the way they interact. Though the system components are not necessarily entity beans or distributed objects but are the abstract modules of the software which are deployed as a unit onto a server with other components. The interactions taking place between these components are called connectors. The way a system is structured and behaves is described by the configuration of the components and the connectors.

Software Architecture

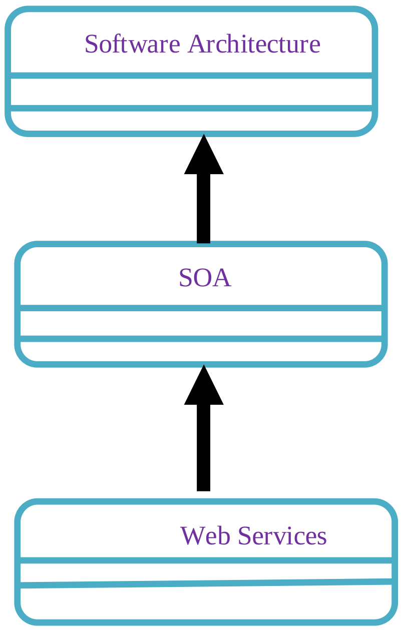


The classic definition of the software architecture is

"The software architecture of a program or computing system is the structure or the structures of the system, which comprise software components, the externally visible properties of these components, and the relationships among them."

The special kind of the software architecture is a service oriented architecture which has several unique characteristics. To make the most effective use of the Web services in their environment, it is important to understand the concepts of the SOA. Though SOA is relatively new term, but the term "service" as it relates to a software service has been around since at least the early 1990's.Becuase of the arrival of the web services more interest about the concepts behind SOA occurred lately in the software development community.

The other technologies that can be used to implement service oriented architecture are depicted in the figure.

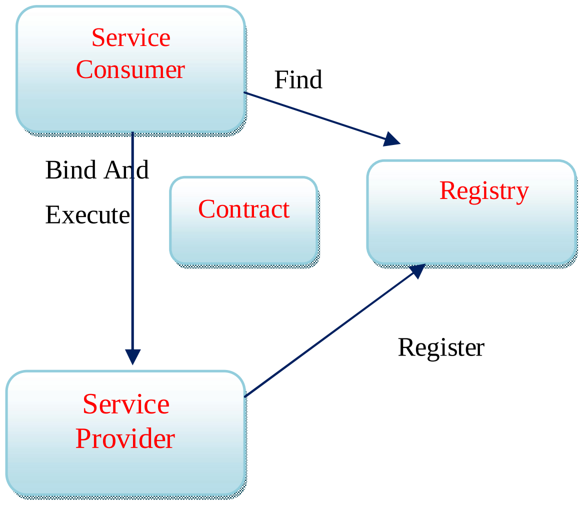


To implement the SOA successfully, web services are simply one set of technologies. The most important aspect of the service oriented architecture is that it separates the services implementation from its interface i.e., it separates the "what" from the "how". From the view of the service consumers, a service is simply an endpoint which supports a particular request format or contract.

Service consumers only expect that the service will execute their requests and they are not concerned how it does. They also expect that their interaction with the service will follow a contract, an agreed –upon interaction between two parties. When the service is given to the service consumers the way the service executes tasks is irrelevant. By executing a servlet, or a mainframe application or a visual basic application the service might fulfill request. In the agreed –upon format, is that the service sends the response back to the consumer is the only requirement.

*SOA Entities:*

Using the "find, bind and execute" paradigm, the consumer of a service can ask a third party registry for the service that matches its criteria. A contract and an endpoint address for the service is provided for the consumer if the registry has such a service. In order to support the find, bind, and execute paradigm SOA configured together the following six entities.



*a)Service Consumer*

An application, service, or some other type of software module that requires a service is called service consumer. The locating of the service in the registry, binding to the service over a transport, and executing the service function by sending a request formatted according to the contract is initiated by this entity.

*b)Service Provider*

The network addressable entity which accepts and executes requests from the consumers is the service Provider. It can be a component, a mainframe system or some other type of software system that executes the service request. For access to the service consumers the service provider publishes its contract in the registry.

*c) Service Registry*

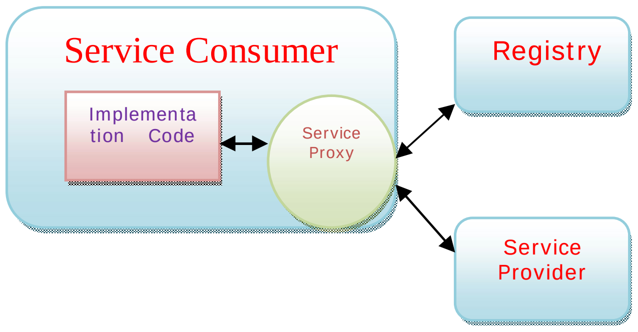
It is a network based directory which accepts and stores contracts from service providers and provides to the interested consumers.

*d) Service Contract*

The way a consumer of a service has to interact with the provider of the service is specified in the service contract. The format of the request and response from the service is also specified in the contract. A set of preconditions and post conditions which specify the state that the service must be in order to execute a particular function are required for a service contract.

*e)Service Proxy*

The proxy provided by the service provider is used by the service consumer to execute the request by calling an API function. The contract and a reference to the service provider in the registry is found by the service proxy. The service proxy then formats the request message and executes the message on behalf of the consumer. For the service consumer ., service proxy is a convenience entity.



*f) Service lease*

The service lease is granted to the service consumer by the registry. It specifies the amount of time the contract is valid: only from the time the consumer requests it from the registry to the time specified by the lease. The customer must request a new lease from the registry when the lease runs out.

For services, a lease is necessary to maintain state information about the binding between the service provider and service consumer and also the time or which the state may be maintained. By limiting the amount of time consumers and providers may be bound, a service lease reduces the coupling between the consumer and provider.

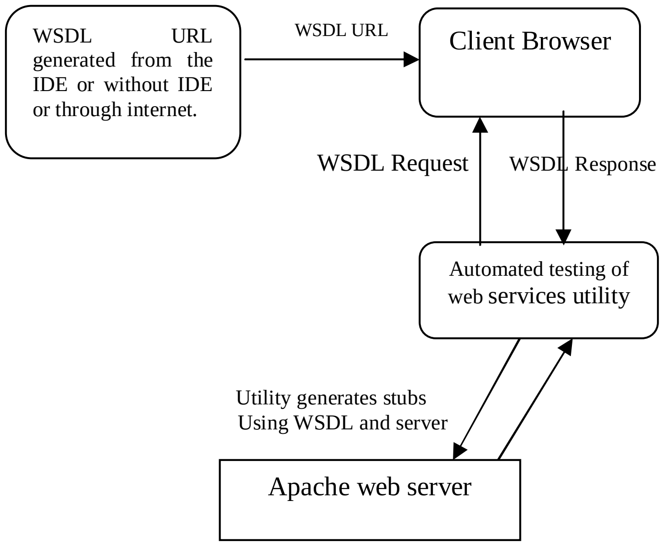
## III.PROPOSED FRAMEWORK

The framework proposed in this paper is the Automated testing of Web Services. It supports testing of Service-Oriented Architecture (SOA) consisting of any web service that is generated using IDE or without IDE or through the internet.

- No-code SOAP/XML testing and WSDL exploration and test maintenance.
- All the information need to know is the URL to capture and invoke test against a web service.
- It is a no-code automated testing, meaning no longer have to script tests.
- It runs on any client and supports Java and .NET and any other SOAP compliant web service.

It provides the way to perform several tests in very short period of time. It supports Functional Testing to ensure that the functionality of the web service is as expected. It allows to run all test cases one time for each web service. The WSDL URL

The architecture of the proposed system is


WSDL URL generated from the IDE or without IDE or through internet.
WSDL URL
Client Browser
WSDL Request
WSDL Response
Automated testing of web services utility
Utility generates stubs Using WSDL and server
Apache web server

ATU (AUTOMATED Testing Utility) can be implemented in wide area of testing in any kind of web application software in today's IT industry. According to the architecture of the web application, a user has to configure and setup the application to be ready to communicate with any compliance clients.

### *d) Obtain WSDL for deployed services*

When a service is made available through Axis, there is typically a unique URL associated with that service. If the service is accessed through that URL in a browser, a message can be seen, indicating that the endpoint is an Axis service that should be usually accessed using SOAP. However, if we tack on "? wsdl" to the end of the URL, Server will automatically generate a service description for the deployed service and return it as XML in the browser. The resulting description may be saved or used as input to proxy-generation.

### *e) Generate Stubs/Drivers*

The required steps to generate Stubs or Driver are

1. Extract the WSDL for the required webservices.
2. Use the AXIS WSDL2Java utility to generate the Axis stubs specific to Java.
3. Use the stubs to make calls to the web services.

For example:

| | |
|---|---|
| Web Service | : Subtract API |
| Method | : subtract |
| Input Params | : int a1, int a2 |
| Output Params | : a1-a2 |

Step1: extract the WSDL for the web service

Enter URL in the browser to obtain the WSDL for the web service.

Step2: Use Axis WSDL2Java tool for generating the stubs

Step3: Using the Stub to make calls to the web services.

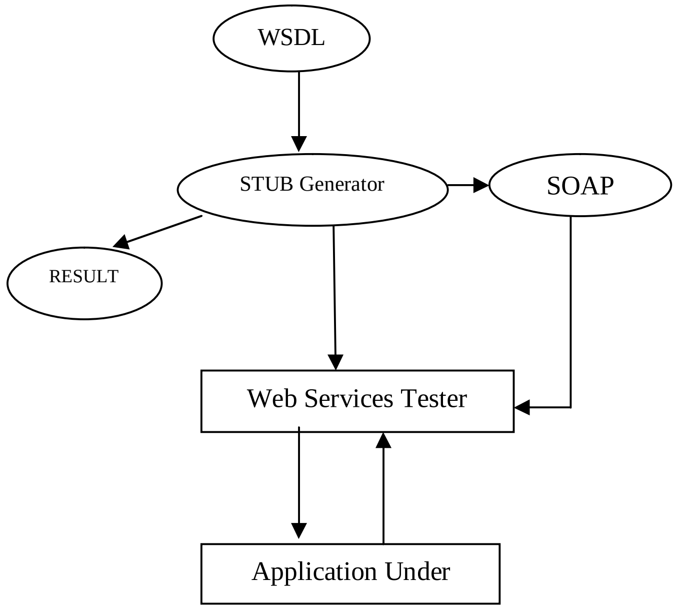
Verify the bean classes before using them as input and output parameters.

### *f) Invoke Services*

Using the generated stubs and drivers, all the available services should be able to get invoked using java client program which could be deployed into the web server. This ATU would provide a user friendly GUI to invoke all these services.

## IV. TESTING STRATEGY: FUNCTIONAL TESTING

This type of testing is done to ensure that the functionality of the web service is as expected. Functional testing also takes care of bounds testing and error checking. Essentially, functional testing is carried out to test the behavior of the system as a whole. When performing functional testing for Web Services, it becomes important that all the methods exposed by a WSDL are tested properly with proper input values. Ideally, the functional testing tool should fit into the framework as depicted in Figure.



A STUB generator is a program which reads the WSDL file and gives the tester all the exposed methods by the web service. The WSDL file also contains other vital information about the exposed methods, like the input parameters and output parameters. It also contains information about complex data types if they are required for some methods. While testing Web Services, it becomes necessary to have a STUB generator which provides the necessary SOAP stubs for calling the exposed methods in WSDL. Once the SOAP stubs are created, the next big task is to create the input parameters for a method under test.

The popular tool Axis is used widely to generate SOAP stubs from the WSDL specifications. It generates code after reading the WSDL file. Once the code is generated it is normally consumed by a client code which is language specific to get the functional testing done.

## *V.CONCLUSION*

Due to its inherent complex nature, testing Web Services poses various challenges to test engineers. ATU is developed after thorough study of Web Service Architecture and its related technologies. It is an excellent web service Functional Testing Tool. ATU is very fast and execute with great speed because it uses Sax Parsing. ATU runs on any client and supports Java and .NET and any other SOAP complaint web service.

## VI.REFERENCES

[1].The Java™ Web Services Tutorial

[2]. http://publib.boulder.ibm.com/

[3].Web Services Handbook for Web Sphere Application

Server Version 6.1

[4].Developing Web Services by Sandeep Chatterjee

[5].http://dlc.sun.com.edgesuite.net/javaee5/screencast
[6].http://RoseIndia.net
[7].https://developer.mozilla.org/en/How_to_create_a DOM_tree

[8].T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, S. Thatte, and S. Weerawarana, "Business process execution language for webservices version 1.1," May 2003.