

A WEB SERVICES TESTING TOOL

Erdogan Dogdu Raj Sunderraman
Georgia State University
Computer Science Department
Atlanta, Georgia 30303 USA
{edogdu,raj}@cs.gsu.edu

Abstract

Web Services technology is quickly becoming the choice of middleware for application integration to provide interoperability among distributed heterogeneous applications and components. This is due to the fact Web Services are based on a few open, simple, and standard protocols (e.g., XML, SOAP, WSDL, and UDDI) and the ubiquitous Internet protocols, such as HTTP, for data and message transmission.

There are many tools and packages being developed and delivered for web services development, yet most tools require client programming to test remote web services. In this paper, we present a unique tool that we have developed for web services testing. It is called the "Web Services Testing Tool" (WSTT). WSTT is a Java servlets-based web application that allows users to interact, test, and invoke any web service available on the web using the service's web services description document (WSDL document) as the only input. WSTT provides an easy to use user interface and has the potential to be extended with more features for future web services testing purposes.

Future versions of WSTT will include features like handling complex types, UDDI browsing and automatic client stub generation in several languages.

Key Words

Web Services, XML, SOAP, WSDL

1. Introduction

Recently a new technology framework has emerged as the next phase of computing, called "Web Services". The idea is not entirely new, but based on distributed computing concepts. Simply, in this framework applications exchange messages and data, and get services from each other using standard communication protocols and data formats [1][2].

Web Services technology is currently in development stages and it is rapidly becoming the basis for distributed application development and the choice of middleware for application integration over the Internet. The reasons for fast adoption of this technology lie in the fact that Web Services protocols are based on open standards and ubiquitous Internet protocols.

Current Web Services technology framework consists of a number of open protocols for description, usage, and discovery of Web Services. These are XML, SOAP,

WSDL, and UDDI. Extensible Markup Language (XML) is the standard data and message formatting language and all other Web Services protocols are based on XML. Simple Object Access Protocol (SOAP) specifies the messaging mechanism and how Web Services are used [3]. Web Services Description Language (WSDL) is an XML-based language that is used to describe Web Services. Universal Description, Discovery, and Integration Service (UDDI) is a directory service that is used to list available Web Services and can be used to automatically search and locate Web Services.

Web Services technology provides a number advantages over other distributed computing middleware technologies [1]:

- (1) *Cost effective application integration*: Since Web Services protocols are based on open standards and since they are easy to implement, there are many open-source implementations of Web Services protocols. Early distributed middleware solutions, such as Java/RMI or CORBA, were either platform dependent or were language or vendor specific. Therefore, they were not widely accepted.
- (2) *Rapid application development*: An application can be built rapidly by using a number of available Web Services as its components; or applications can be extended with new Web Services components. The component-based approach results in robust applications as we only need to test the integration of the components. The individual components are already tested and made available as services. This can be called application reuse.
- (3) *Interoperability*: Web Services components can be built and serviced in any programming language, operating system, or hardware platform. Web Services from different platforms will still be interoperable due to standard message exchange mechanisms (SOAP/XML) and standard network protocols (Internet/HTTP/SMTP/FTP). Therefore, service implementation is completely hidden from the consumers of the web services.
- (4) *Open standards*: Open communication and data formatting protocols is the true advantage of web services platform. Just like the Internet revolutionized the computing methodology via its open and simple protocols (HTTP and simple document formatting language HTML), SOAP/XML based web services

will be widely accepted since all protocols underlying web services are open standards provided by World Wide Web Consortium (www.w3.org) and other standardization organizations.

- (5) *Strong industry support*: All major information technology companies have joined the effort in building the infrastructure and tools for building Web Services, including Microsoft, Oracle, IBM, Sun Microsystems, and many more. Therefore, it is very likely that this technology will be embraced quickly.

A web-based Web Services Testing Tool (WSTT) for testing Web Services is presented in this paper. Being one of the first few of its kind available on the web, WSTT is a web-application that uses the above-mentioned technologies XML, SOAP, and WSDL. Its main goal is to connect and use any web service that is available on the web.

The rest of the paper is organized as follows: Section 2 gives the motivation for this work. Section 3 presents WSTT in detail. Section 4 discusses the implementation of WSTT. Section 5 presents the features of WSTT and its shortcomings. Section 6 discusses related work. Finally, Section 7 presents conclusions and a discussion of future work.

2. Motivation

Web services technology is touted as the next big thing in application development and it will be seen whether it will hold its promises. Many commercial and open-source tools are being developed and made available for the developer community. Most packages are far from being complete, and most are difficult to use and master, requiring a great deal of intensive configuration, installation and setup time. Commercial tools such as XMLSpy and Visual Studio .NET, are much better in terms of ease of use and short development time, but they require more investment.

Currently, business level use of web services is limited since the technology is not mature enough. But, there are many web services developed for trial purposes. Some are listed in www.xmethods.net web site. In the future, businesses will be using web services extensively, either as consumers or providers, or for both purposes.

There is certainly a need for more tools and extensions to the technology so that web services framework can be widely used and made available. One of the needs in this area is a tool to test existing web services. The automated tool should allow software developers to access any web service, view the service details, and invoke the services interactively without writing a program. Most widely used web services development packages, such as Apache Axis, require the developers to write client programs to test and utilize remote web services.

This work presents one such tool we have developed in our Database Systems Laboratory in Computer Science Department at Georgia State University. It is called the

Web Services Testing Tool (WSTT) and it is publicly available at <http://db.gsu.edu/project/wstt>.

3. Web Services Testing Tool

We have designed Web Services Testing Tool (WSTT) as a web application. WSTT can be accessible from anywhere on the web using a standard web browser. Browser needs to be directed to the application currently served at:

<http://db.gsu.edu/project/wstt>

WSTT is a web application developed using Java Servlets [4]. Figure 1 depicts the overall framework of WSTT. The tool is deployed over a web server with Java servlet hosting capabilities. Current implementation of WSTT is deployed over Apache Tomcat servlet engine [5]. In this web application, user accesses the application via a web browser. Application displays the user interface that is presented in Figure 2.

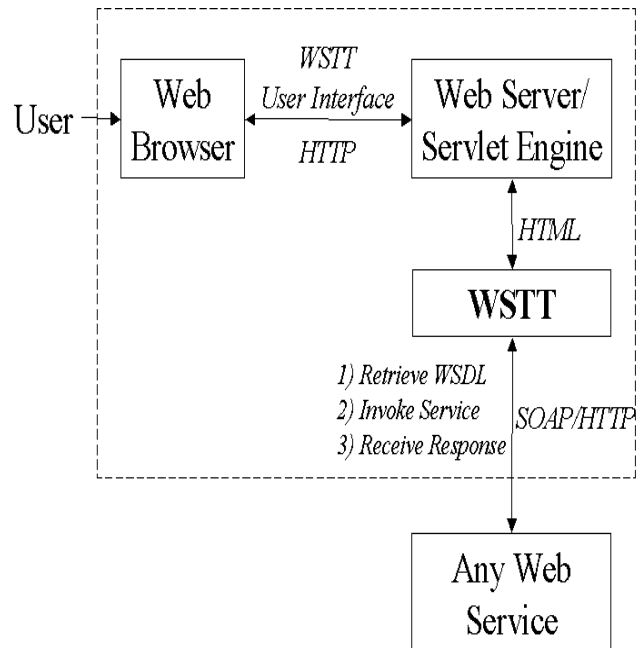


Figure 1. WSTT Framework

User follows these steps to use the tool:

1. *User enters the URL of a Web Service*: User specifies a URL in which a web service description document, or in other words the WSDL document, is located (top frame). Upon submission of this URL, WSTT:
 - a. retrieves the WSDL document via HTTP,
 - b. parses the WSDL document, and
 - c. displays available web service methods to the user in the bottom left web frame titled “Service Methods”.
2. *User views the available methods and selects one*: Upon selection of a method, WSTT parses the WSDL document and displays the input parameters of the selected method in the frame titled “Parameter Values”.

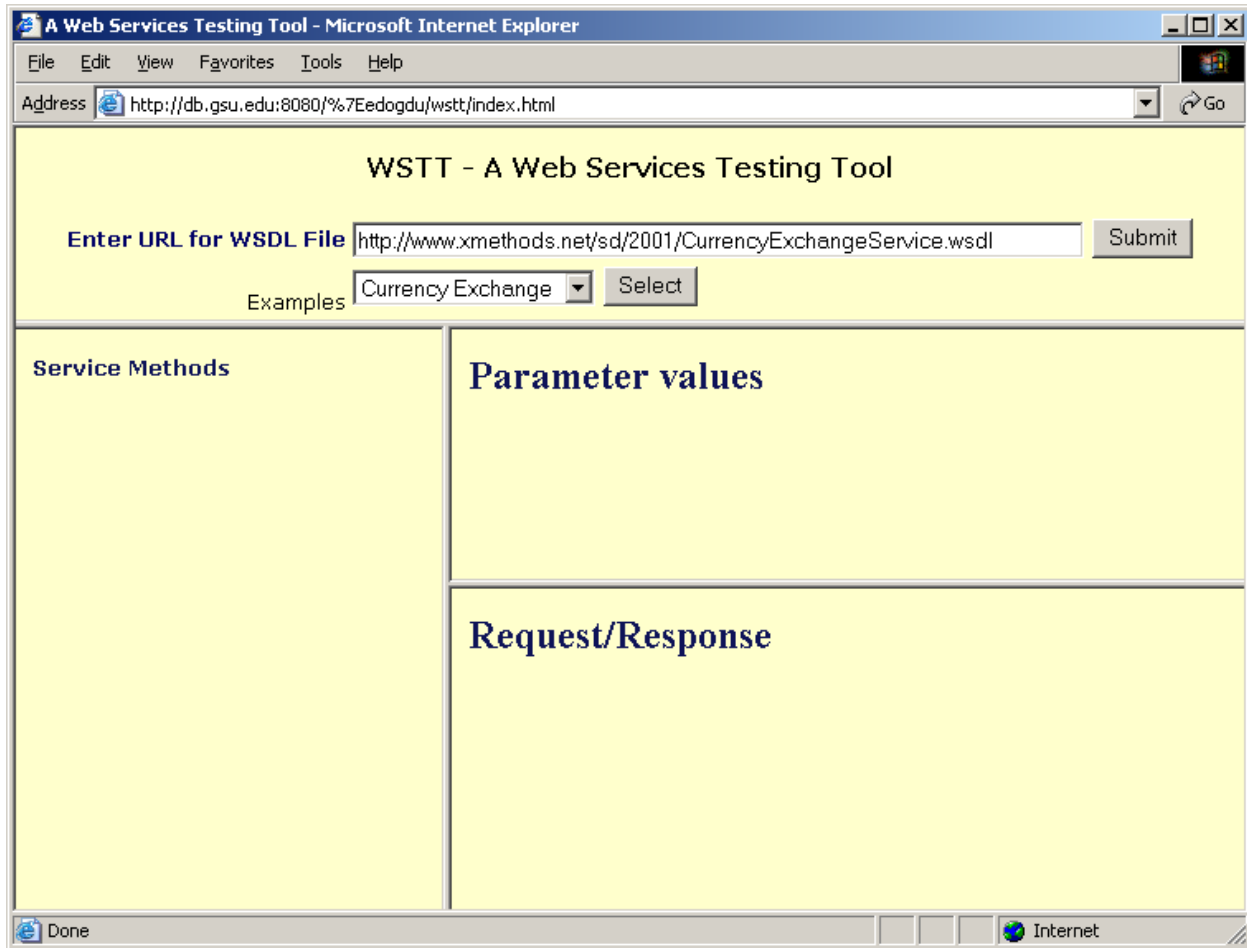


Figure 2. WSTT User Interface

3. *User enters appropriate parameter values for the method and invokes the method:* Upon submission of this request, WSTT:

- a. packages the web service request with the user provided parameter values using Apache SOAP and invokes the method via HTTP by sending the SOAP request message, and
- b. waits for the response from the web service. Upon receipt of the response from the web service, WSTT displays the results of the invoked method in the web frame titled “Request/Response”.

Figure 3 depicts an example use of the tool. In this example, a “currency exchange” web service is tested. WSDL of the service is available at:

<http://www.xmethods.net/2001/CurrencyExchangeService.wsdl>

Upon submitting the URL location of the WSDL document, WSTT retrieves and parses the document and displays the available methods. In this case there is only one method, namely “getRate”. After selecting the

method, getRate, WSTT parses the document and displays the input parameters, country1 and country2 of the method with appropriate text input HTML elements. User enters some values; in the sample screen “USA” and “Canada” are entered. The result of the method, “1.366” is retrieved from the service and displayed on the bottom-right frame.

4. Implementation of WSTT

Current version of WSTT is implemented using the following packages:

- Java 2 SDK, version 1.4, downloadable from: <http://java.sun.com/j2se>
- Tomcat (Servlet Engine), downloadable from: <http://jakarta.apache.org/tomcat/index.html>
- Apache SOAP, available at: <http://xml.apache.org/soap/index.html>
- Oracle XML Parser. The package is available at: http://otn.oracle.com/docs/tech/xml/xdk_java/doc_library/Production9i/index.html

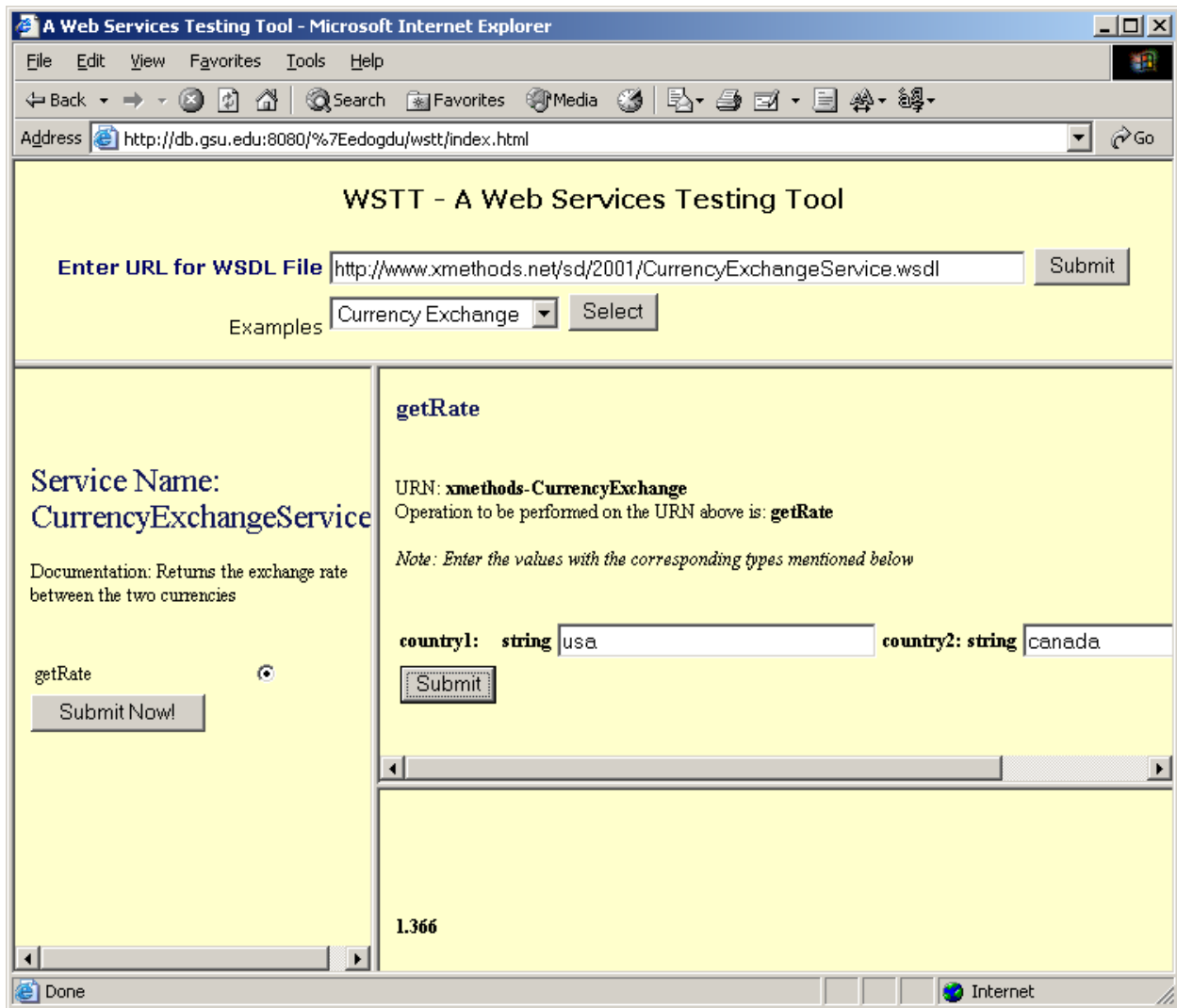


Figure 3. Example use of WSTT

Java SDK is the standard Java development kit. Apache Tomcat is the servlet engine that hosts the WSTT servlet application. Apache SOAP provides the web services interaction and invocation of remote web services methods, etc. Oracle XML Parser is used to parse the WSDL documents to find the methods and their parameters. Document Object Model (DOM) API of the parser is used to search over a WSDL document.

5. Features of WSTT

Although very simple, WSTT is one of the first web services testing tools freely available on the web. It is a user-friendly tool with a simple implementation. It automatically tests any web service with a WSDL document. It is remotely accessible from anywhere on the Internet via a browser, therefore does not require any software installation. WSDL processing using Oracle XML Parser is fast, application does not present any delays.

Currently, WSTT only invokes web service methods that do not require parameters of “complex” types but only simple types like integer, string, etc. This is being addressed in the new version of WSTT. Other features being added are: UDDI browsing and automatic client stub generation in several languages.

6. Related Work

Although Web Services technology is a relatively new one, there are already many tools and packages available that can be used in deploying and consuming web services. Most of those packages do not provide a testing tool, or they require some sort of programming to utilize web services, such as writing a simple client program or generating some client stub to access the web services, etc. WSTT, on the other hand, provides a user-friendly stand-alone interface without any requirement to write programs.

As we started working on this project, some tools are appearing in the marketplace. Two of them that are most related are:

- XMLSpy: This is one of the leading tools for developing XML applications. It has a SOAP debugger that lets the user to test web services via SOAP invocations. This is a sophisticated tool, yet it requires installation and is not a free product [6].
- WSDL Explorer: A web application that displays WSDL files, generates views of operations, allows invocation of operations, and allows viewing of sample message flow. It enables users to compare and contrast Web services without going through the time and trouble of importing them into a heavy development tool. WSDL Explorer provides the ability to browse WSDL files, and it offers immediate access to web service operations [7]. This is only tool that resembles our work. At the time we tested the tool it was giving errors.

7. Conclusions and Future Work

We have implemented a unique web application that can be used in testing any web service. Being a web application, WSTT does not require software installation and freely available and can be accessed from anywhere on the web using a browser.

Next version of WSTT is being enhanced to handle any parameter type including complex types. Additionally, WSTT will add options to generate client stubs for web services on the fly, such as Java clients. These stubs can be used to implement application integration features in existing applications.

Acknowledgements

We would like to thank Lakshmi Yadlapati for the implementation of WSTT as part of MS thesis work. Without her work, this project could not have been accomplished.

References

- [1] Erdogan Dogdu, An Extended Web Services Framework, *Proc. of the IASTED Int. Conf. on Communications, Internet and Information Technology*, Sep 18-20, 2002, St. Thomas, Virgin Islands, USA.
- [2] World Wide Web Consortium, Web Services Activity, <http://www.w3.org/2002/ws/>
- [4] Java Servlets, <http://java.sun.com/products/servlet/>.
- [3] Scott Seely. SOAP: Cross Platform Web Service Development Using XML. Prentice Hall, 2002.
- [5] Apache Tomcat, <http://jakarta.apache.org/tomcat>.
- [6] XMLSpy, <http://www.xmlspy.com>
- [7] WSDL Explorer, <http://www.alphaworks.ibm.com/tech/wsdlexplorer>.