Web Service Response Time Monitoring: Architecture and Validation

Sara Abbaspour Asadollah and Thiam Kian Chiew

Department of Software Engineering, Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia
spour@siswa.um.edu.my, tkchiew@um.edu.my

Abstract. Web services are used in many Web applications in order to save time and cost during software development process. To peruse Web service response time, a suitable tool is needed to automate the measurement of the response time. However, not many suitable tools are available for automatic measurement of response time. This research is carried out in the context of quality of Web services in order to measure and visualize Web service response time. The method proposed in this research for accomplishing this goal is based on creating a proxy for connecting to the required Web service, and then calculating the Web services response time via the proxy. A software tool is designed based on the proposed method in order to guide the implementation that is still in progress. The tool can be validated through empirical validation using three test cases for three different Web service access situations.

Keywords: Web Service, Web method, performance, response time.

1 Introduction

Nowadays, the Internet is an important phenomenon in human life and the number of its users is growing fast. It supports human interactions and connections with information and data in textual and graphical styles. It provides many services for its users. Web services are one of the recent important innovations in software that bring many consequences in software design and implementation. Web services are used in many Web applications that provide services such as searching and buying goods with the best quality and price, booking and coordinating plane tickets or hotel rooms, reading online newspapers and books, transferring files, and many more. The services are provided through Web sites and Web services. However, since quality is becoming an important issue in software, the Web sites and Web services should be monitored in order to provide high-quality services.

Monitoring is defined as the process of data extraction during program execution [1]. There are a number of tools for monitoring Web sites and Web services; some of them monitor hardware, others monitor software, and there are some tools that monitor both hardware and software (hybrid monitors) [2]. Monitoring Web sites and Web services enables users and developers to identify their features in order to compare them and select the best ones to use. Moreover, after monitoring, users can also recognize the current and potential problems of Web sites and Web services.

Then, these users will be able to solve/avoid these problems in order to produce higher-performance applications and reduce weaknesses of their products.

Measuring response time of Web services facilitates finding and correcting the cardinal and fundamental problems that affect response time. The ability to rectify problems quickly and improve Web services response time encourages Web developer to use these Web services. Monitoring Web service response time helps users to select better Web services in times of their response time.

2 Background

The Internet is an important and huge source of information that affects human lives. Different types of information on the Internet such as text, graphics, images and multimedia increase the attention of different kinds of users in order to work with the Internet. Such information is usually stored on servers. Software developers develop Web applications and Web services for users to access this information.

Web services and Web applications have similarities and differences between each other. Web applications are designed for browsers (standalone applications) while Web services are designed to be (re)used applications. Since Web services are always used by other applications, they do not need to have user interfaces [3]-[4]. As a result, when a Web application is designed to use a Web service to fulfill some of its functionalities, the Web application response time will be dependent on the Web service response time.

Response time of a Web service is defined as the sum of transmission time and processing time. Processing time is measured as the time for processing a request. In the context of Simple Object Access Protocol (SOAP), processing time is the period from the point where a SOAP message arrives at the engine, until a corresponding SOAP response message is sent using a reply activity [5]. Meanwhile, Transmission time is the time from when client sends the request until server receives it, plus the time from when the server sends the response until client receives it. As a result, response time is the time needed to process a query, from the moment of sending a request until receiving the response [6].

The performance of a Web service is considered as an important issue for its users [7]. If a user feels that the performance of the service he/she uses is poor (like long response time), then he/she will try to find another service with better performance. Since Web service monitoring, calculates the response time and analyzes its results, then users will be able to make suitable decisions about choosing the best Web service that conforms to their requirements.

One way to prove the abilities of Web services is using the tools for checking their performance, especially response time. This research presents a method to measure Web service response time, and a designed tool based on the presented method. This tool is useful for Web service providers to prove the ability and the performance (short response time) of their services to their existing customers, and encourage them to use or continue using these Web services. The tool will also enable potential customers (usually Web application designers) to select and use suitable Web service. Another benefit of the tool is that it facilitates testing Web services after they are implemented. It is useful for Web service designers, developers, and testers to find out the response time of the Web service in order to identify critical performance points.

3 Related Work

Usually, Web users measure the performance of Web applications with respect to time (Response time). As mentioned above, one aspect affecting the response time of Web applications is the response time of the Web services used in these Web applications. Thus, the response time of Web services needs to be measured and monitored.

Response time for the Web service is "*the time needed, to process a query from sending the request until receiving the response*" [7]. Repp et al. divided the Web service response time into three parts, which are network transport time, task time, and stack time. By this definition, Repp et al. (2007) defined the Web service response time as follows:

$$T_{response}(WS) = T_{task}(WS) + T_{stack}(WS) + T_{transport}(WS) \quad (1)$$

Where:

- T_{task} is the time for processing the request (message) in both end-points and intermediate systems. It is the largest part of the response time.
- T_{stack} is "*the time from traversing the protocol stacks of source destination and intermediate system*".
- $T_{transport}$ is the network transport time.

Another method for measuring Web service response time is provided by Cardoso et al. (2004). He defined the task response time (T_{task}) as the time that an instance (object created at runtime) takes to be processed by a task, composed of two major components, which are delay time (DT) and process time (PT).

$$T_{task} = \text{Delay } T_{task} + \text{Processing } T_{task} \rightarrow T_{task} = DT_{task} + PT_{task} \quad (2)$$

Where:

- DT is the non-value added time taken for processing an instance by a task.
- PT is the time that it takes for processing a workflow instance in a task. In other words, it is the whole time that a task needs to process an instance. DT consists of queuing delay time (QDT), and task setup delay time (SDT).

$$DT_{task} = QDT_{task} + SDT_{task} \quad (3)$$

Where:

- QDT is the waiting time for an instance in queue until its processing starts. There are different methods to put a task in a queue on the server side, such as First In First Out (FIFO) and Server In Random Order (SIRO). The reason for engendering queue on the server is that a number of requests that arrive from the clients could be huge, and the server needs to process and respond to all of them. Therefore, the server places these requests into a queue and processes them accordingly.
- SDT is the waiting time for an instance until the task related to this instance setup on the server completely and instance processing starts.

As mentioned above, $T_{transport}$ (network time) is another factor that is considered in calculating the response time. Cahoon et al. (2000) presented the transport time as the amount of time that a message spends on the network. Thus, it can be calculated based on the message size and the network bandwidth by using the following formula:

$$T_{transport} = \text{Size} \times (8 / \text{Speed}) \tag{4}$$

Where:

- *Size* is the number of bytes in the message.
- *Speed* is the bandwidth of network in bits per second.

4 Monitoring Web Service Response Time

The proposed method for measuring and monitoring the response time for a Web service consists of two stages: running Web services, and monitoring the response time for each Web service.

The first stage of this method is accomplished through running Web methods of Web services. This task needs four inputs: Web service address, Web method name, time duration for running the Web method, and the number of calls for running the Web method of that Web service. After setting these inputs, a set of parameters is needed to call the Web method. Then, a factory creates suitable value for each parameter of the Web method to be passed for calling the Web method. Once these values are defined, a request is sent to the Web service in order to run its Web method. Before sending the request, the current date and time are saved, and after receiving the response from Web service, the date and time are saved again. The response time of the Web method is calculated by subtracting the *sending request time* from the *receiving response time*. Finally, the values such as *Web service name*, *Web method name*, *request date*, *request time*, *response date,* and *response time* are recorded. This process is repeated according to the running duration defined earlier in order to create more records to be used in the second stage of the method.

The second stage of this method aims to monitor the response time of the Web service. This is fulfilled by using the recorded data from the first stage. The input values for this part are Web Service name, Web method name, and monitoring time interval, i.e. the starting and ending date and time for monitoring. Then, all records belonging to that Web method are selected and filtered according to the inputs. These records represent the response times of the monitored Web method.

In order to apply this method, a software tool is proposed. This tool is designed based on the proposed method. Fig. 1 shows the architecture of the proposed tool.

In this architecture, the client and server are connected via network and they communicate with each other using Hyper Text Transfer Protocol (HTTP). HTTP transaction consists of two sections: a request from a client to a server, and a response from the server to the client [10].

In this architecture, UI calls a method of a Web service, and then the proxy serializes the parameters of the method to the .Net framework. After that, .Net framework sends them by a SOAP message to HTTP through the Internet. The Web service returns the result of calling the method. This response is serialized into a SOAP message and is passed through IP, TCP, HTTP, and .Net framework. Finally, the proxy diserializes the response and returns the value to the UI.

Using this architecture, two main functionalities can be achieved, (4.1) setting and running a Web service, and (4.2) monitoring and generating reports.

4.1 Setting and Running a Web Service

Calculating and recording the Web service response time is accomplished by setting, then running a Web service module. This module can be decomposed into three sub-modules. A sub-module for parsing the description document of a Web service (WSDL) is needed. It shall extract the useful information from the document in order to use this information for calling the desired Web method of any Web service. The second sub-module is needed to invoke a desired Web method of a Web service. Finally, another sub-module is needed for recording the obtained information to be analyzed during the monitoring process.

4.2 Monitoring and Generating Reports

This module shall be used for reporting the measured values of Web service response time. It is composed of two sub-modules, one for loading the saved data from the data storage, and another one for generating useful reports to monitor the desired Web service.

5 Method and Technique Validation

The validation of this research can be done via three test cases and two test Web services. After implementing a software tool based on the proposed method, the tool can be validated via these test cases and Web services by calculating the response time manually and automatically. One of these Web services will be used for finding the response time of its Web methods manually (manual testing), and the other one for automatic testing. The manual response time measurement will be done by the user while the automatic response time calculation will be accomplished via the proposed tool. Test cases were designed in order to test the three parts of response time (processing time, transporting time and queuing time).

The purpose of the first test case is measuring the processing time. For the first Web service (manual testing), there will be some code added to its Web methods for measuring the processing time of each Web methods in order to compare its results with the ones that will be obtained from the proposed tool during automatic testing. If these two sets of results are similar to each other, then it proves that the proposed tool for measuring the processing time works correctly.

The second test case will measure the processing time and transporting time together by uploading the test Web services on another system that is accessible from the user system. Then, the processing time and transporting time will be measured for the test Web services both manually and automatically. The result of manual testing and automatic testing will be compared together to check the accuracy of the proposed tool.

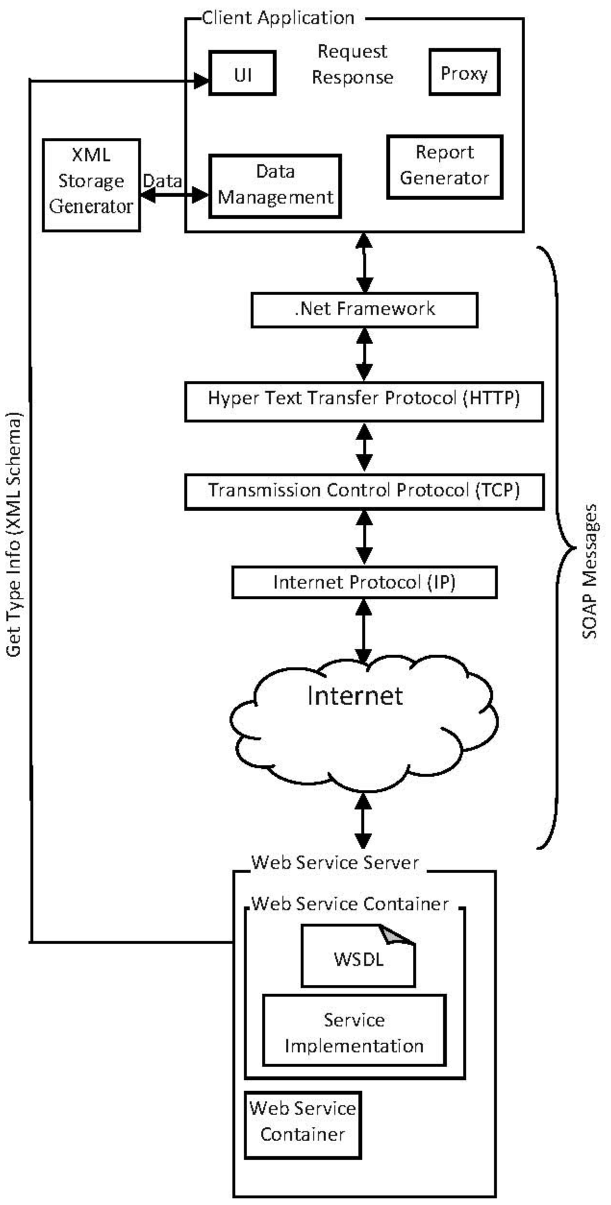

Fig. 1. Architecture of the designed tool for monitoring Web service response time

Finally, for the last test case, two users will call a Web method of the Web service at the same time. This technique will cause queuing time for the Web services. After running each test case, the obtained results of the manual testing and automatic testing will be compared with each other to validate the proposed tool.

6 Conclusion and Work in Progress

Web services are required by other applications in order to fulfill some of the functionalities of the Web applications. Measuring and monitoring the response time of Web services are important since users look for Web services with high quality performance, especially the Web services with short response time. One way to prove the abilities of Web services is using the tools for checking their performance, particularly response time. This paper presented a method and proposed an architecture for a tool to measure and monitor the response time of Web services. This tool is useful for Web service designers, developers, and testers to find out the response time of the Web service in order to identify critical performance points. Moreover, it enables the Web service providers to prove the ability and the performance (short response time) of their services to their existing customers, and encourage them to continue using these Web services. This tool will also enable potential customers (usually Web application designers) to select and use suitable Web services. The next step of this research is to implement and test the designed tool.

References

1. Chiew, T.K.: Web page performance analysis, PhD thesis, University of Glasgow (2009)
2. Haban, D., Wybranietz, D.: A Hybrid Monitor for Behavior and Performance Analysis of Distributed Systems (PDF). IEEE Transactions on Software Engineering 16, 197–211 (1990)
3. Killea, P.: Web Performance Tuning. O'Reilly, Sebastopol (1998)
4. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web services: concepts architectures and applications. Springer, Heidelberg (2004)
5. Pautasso, C.: Emerging Web Services Technology: Wewst 2007, Halle (Saale), Germany. Birkhauser, Basel (2008); Selected Revised Papers
6. Pautasso, C., Bussler, C.: Emerging Web Services Technology. Springer-Verlag New York Inc., Secaucus (2007)
7. Repp, N., Berbner, R., Heckmann, O., Steinmetz, R.: A cross-layer approach to performance monitoring of web services. In: Proceedings of the Workshop on Emerging Web Services Technology (2007)
8. Cardoso, J., Sheth, A., Miller, J., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. Web Semantics: Science, Services and Agents on the World Wide Web 1, 281–308 (2004)
9. Cahoon, B., McKinley, K.S., Lu, Z.: Evaluating the performance of distributed architectures for information retrieval using a variety of workloads. ACM Transactions on Information Systems (TOIS) 18, 1–43 (2000)
10. Grove, R.F.: Web-Based Application Development. Jones & Bartlett Pub., USA (2009)