International Journal of Database Theory and Application
Vol.7, No.5 (2014), pp.145-160
http://dx.doi.org/10.14257/ijdta.2014.7.5.11

Testing and Evaluation of a Hierarchical SOAP based Medical Web Service

Abhijit Bora and Tulshi Bezboruah

Department of Electronics and Communication Technology, Gauhati University, Guwahati-781014 Assam, India
Tel:+91-361-2671262(O); Fax: +91-361-2700311(O);
abhijit.bora0099@gmail.com / zbt_gu@yahoo.co.in

Abstract

Performance testing of hierarchical web service communications is essential from the perspective of users as well as developers, since it directly reflects the behavior of the service. As such we have developed a SOAP based research web service, suitable for online medical services to study the performance and to evaluate the technique used for developing the service. We call the service as MedWS (prototype research medical web service). Load and stress testing have been carried out on MedWS using Mercury Load Runner to study the performance, stability, scalability and efficiency of the service. The performance depends on metrics such as hits/sec, response time, throughput, errors/s and transaction summary. These metrics are tested with different stress levels. The statistical analysis on the recorded data has been carried out to study the stability and quality of the application. The present study reveals that the SOAP based web service which has been developed with Java programming language is stable, scalable and effective. We present here the architecture, testing procedure, results of performance testing as well as the results of statistical analysis on recorded data of MedWS.

Keywords: *Web Service, RDBMS, Java API, performance measurement*

1. Introduction

The Web Service (WS) is a modular and self contained software application that can be invoked over the World Wide Web (WWW) [1]. The WS can provide a service for specific business logic (BL). It is similar to the activity of remote procedure call using client server application. Professionally, a WS supports machine-to-machine interoperable interaction over an internet. It contains Web Service Description Language (WSDL) file as WS interface. Other systems interact with the WS using Simple Object Access Protocol (SOAP) [2]. The WS can be used alone for a particular operation or may be aggregated by other WS for the enhanced flexibility in operation. It has a potential to enhance Business to Business (B2B) collaboration by integrating compatible services to a service of higher dimension [3]. Every service may assume one or more roles such as: (i) being a service provider, (ii) a broker or (iii) a user [4]. The WS can be invoked by a client, such as, browser, mobile or other software based devices, typically which are the consumers of a WS in general. The main three platforms of WS are SOAP, WSDL and Universal Description, Discovery and Integration (UDDI).

ISSN: 2005-4270 IJDTA

1.1 Related Work

In the year 2005, N. Hashmi, D. Myung, M. Gaynor, S. Moulton [5] presented a WS based medical service for collecting sensor based network responses using Java and Microsoft .NET platform.

In the year 2005, M. B. Juric, I. Rozman, B. Brumen, M. Colnaric, M. Hericko [6] illustrated an analysis on performance and an in depth comparison of WS and remote method invocation using Windows and Linux platform.

In the year 2009, B. Abdelkader and B. Samia [7] introduced a hierarchical model to represent different scenario of WS composition with constraint management such as time, duration and time interval.

In the year 2010, V. Stoicu-Tivadar, L. Stoicu-Tivadar, D. Berian, V. Topac [8] presented a WS based system architecture called as TELEASIS, that insures support for medical and social tele-assistance for elderly people.

In the year 2012, K. Mohamed, D. Wijesekera [9] studied the comparative analysis for hosting RESTful WS versus SOAP-based WS.

In the year 2014, Lorenzo Bianchi, Federica Paganelli, Maria Chiara Pettenati, Stefano Turchi, Lucia Ciofi, Ernesto Iadanza, and Dino Giuli [10] proposed PHARMA, a WS based information system to support healthcare staff in cooperative execution of drug prescription, transcription and registration tasks and discussed the results of the usability evaluation carried out in a hospital.

2. The Objective and Methodology

The main objective of the proposed work is to design, develop, implement and test a proto type hierarchical SOAP based WS considering pharmacological data [11] to study various attributes like load, performance and scalability of the service.

The WS has been developed and implemented by using Java Application Programming Interfaces (APIs), apache tomcat web server and MySQL database server. It has three different services, namely: (a) the parent WS (b) the client WS and (c) the child WS. The data set for the service is 10000. The WS is hosted at apache tomcat web server. A virtual user (VU) script is generated using Mercury Load Runner to access the WS and to monitor the load and performance test. The data mapping in between diseases, medicines, components, manufacturer and clinical remarks is prepared. The flowchart and entity relationship (ER) diagrams are prepared to present the basic working principle of the WS. The architecture for the WS is developed. The testing is performed up to 1500 virtual users by deploying the WS on Mercury LoadRunner. The responses of the different testing are recorded. Statistical testing of the recorded responses has been performed to study different aspects of the WS. Figure 1 elaborate the basic methodology that we have used in developing the WS.

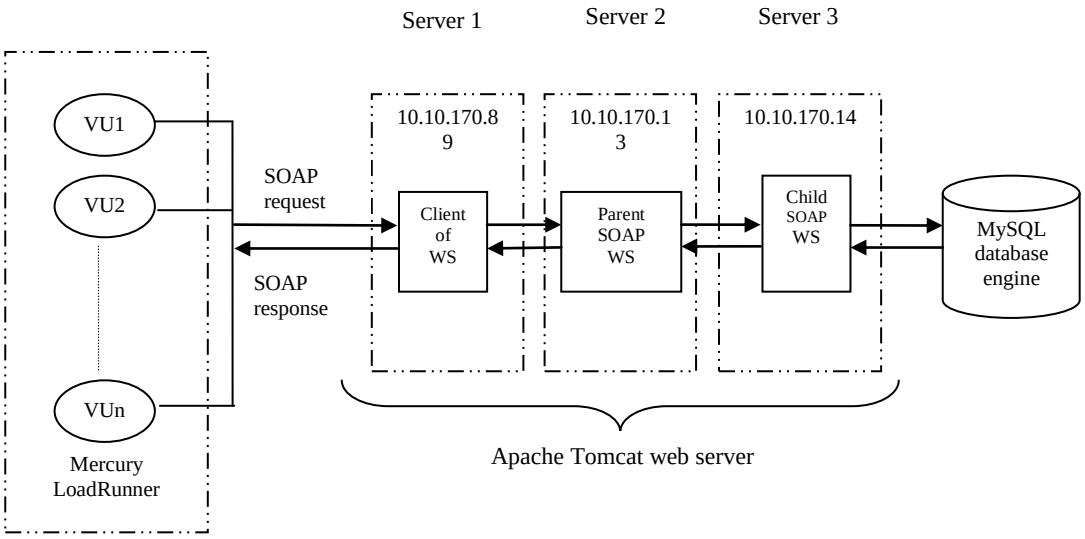

Figure 1: Methodology for Prototype Hierarchical SOAP based WS Communications

3. Software Aspects of MedWS

The Java programming language is an open source choice for developing web applications. It guarantees concurrency control and makes it relatively easy to create SOAP based WS. The most important features of APIs for XML are that they all support industry standards, ensuring interoperability [12]. The WS application can be developed and implemented using Java APIs with Spring framework which is Model-View-Controller Model 2 (MVC2) architecture [13] and tools provided by an integrated WS Stack called Metro. The Metro stack enables any one to create and deploy secure, reliable, transactional, interoperable WS and clients [14]. Different software specifications for the proposed work are: (i) web server: Apache Tomcat version 7, (ii) the database server: MySQL 5.0, (iii) platform: NetBeans 7.0 IDE and (iv) web browser: Mozilla Firefox. Along with these, Java Development Kit version 7 (JDK 7) and Java Runtime Environment version 7 (JRE 7) is also used. The WS and its client applications have been run on servers with hardware specifications: Intel® Xeon® CPU E5620; Processor speed: @2.40 GHz; RAM: 8 GB and Memory space: 600 GB. The operating system is 64-bit Windows Server 2008 R2 Standard.

3.1 The Architecture

The architecture of the propose WS is shown in Figure 2. This architecture represents the basic functionality of all the three WS.

3.1.1 The Client WS: The client WS contains the user interface and presentation code, such as, Java Server Page (JSP) pages, Hyper Text Markup Language (HTML) form controls, server side class files etc. The HTML form controls are provided in Client WS for capturing the end user data. The role of this client is to capture data and forward it to parent WS.

3.1.2 The parent WS: The parent WS is responsible for capturing and forwarding the request from the client WS to child WS. The parent WS is also responsible for informing the client WS about the response of child WS.

3.1.3 The Child WS: The child WS holds the required BL functions, validations and calculations related to the data. The parent WS acts as a mediator between the client WS and the child WS. The child WS manages the physical storage and retrieval of data from the database. It receives the data from the parent WS and sends it to the database and vice versa. The child WS holds the database queries for performing necessary operation.

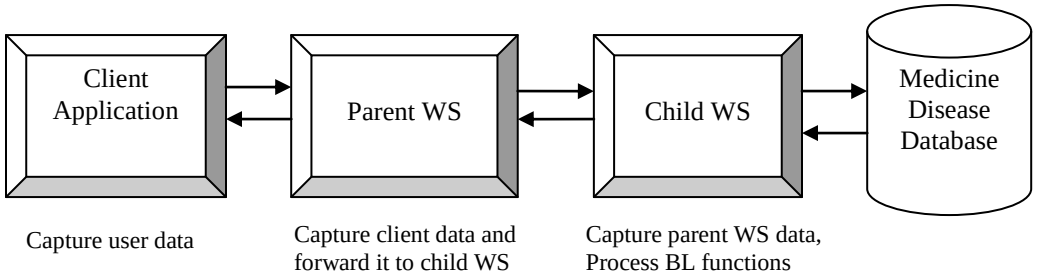

Figure 2. The Proposed Architecture for the MedWS

4. Design Aspects of MedWS

The SOAP based research WS using Java as programming language which we have developed is suitable for online health advices along with support for medicines that are available in the market. We have taken into account the disease related clinical details, namely: (a) the medicine name for a particular disease, (b) the manufacturing company name, (c) the medicine, and (d) the available medicine package in the form of tablet, syrup, injection, and lotion. We combine them together to produce an arrangement for implementing MedWS. The service is developed by taking into account of a pharmacological book published in India as sample data [11]. The Create, Read, Update, Delete (CRUD) operations are performed to generate the responses. When users open the client application, the end user interface will open. The interface contains one text box and one submit button. The users can enter a particular disease name in the text box and then can click "submit" button to submit query for required information. The clinical details against the entered data in the text box will appear on the browser. The response is in a tabular format with header columns labeled as disease name, component name, medicine name, company name and remarks. The response will be generated based on the records available in the dataset. If there is no matching found then a message as "No Existing Records Available" will appear on the browser.

4.1 The ER Diagram

The Flowchart of the program developed for the proposed work is presented elsewhere [15]. The ER diagram of the database is presented in Figure 3. The database is a MySQL solution which contains relevant information about the medicine, available components, company and clinical details. The database contains 5 tables. The data that the tables contain refer to:

Medicine: This table contains medicine names
Component: This table contains component name of the medicine
Disease: This table contains disease list
Company: This table contains company names of the medicine

Disease-medicine mapping: This table contains clinical details for particular disease

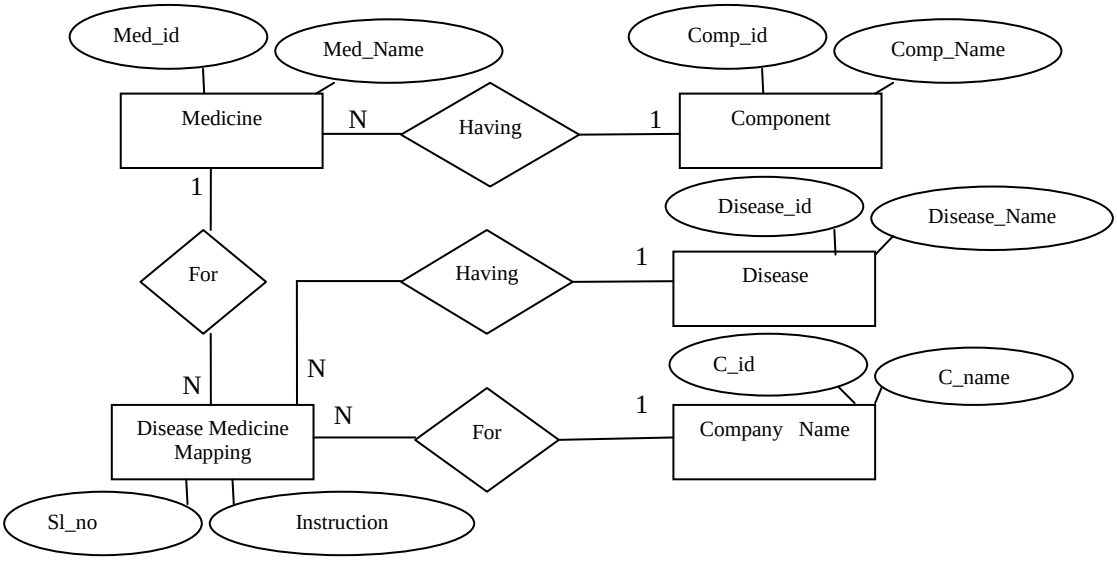

Figure 3. ER Diagram for Disease Medicine Dataset

5. Testing of MedWS

The MedWS is tested using an automated testing tool Mercury Load Runner version 8.1 to predict the systems' behavior and performance. It stresses the system, records the systems' performance metrics and analyzes it accordingly [16]. During the experiments, a user think time of approximately 30sec is incorporated in performing the transaction and an average steady-state period of 5min is set for all the experiments. The stress level is gradually varied, so that it can saturate the server. Each WS request causes an execution of SQL SELECT query to retrieve disease related clinical data from the database table. We follow the various steps for the test discussed and presented elsewhere [17, 18, 19]. The test case for select method invocation is given in Table 1 below.

Table 1. Test Case for Select Method Invocation

Step	Step description	Expected outcome
1	Open URL http://server1:8080/t22P_Client/index.jsp	Child WS index page will be displayed with the form elements: a) "Enter keyword" text field b) "Submit" button
2	Enter valid disease name and click "Submit" button a)Enter "Cold" in "Enter keyword" text field	Pass the parameter to child WS through parent WS for necessary SQL Select operation and wait for the response.

3	Child WS response is displayed	Response page http://server1:8080/ t22P_Client/result.jsp is displayed with a result set containing following data a) Disease name b) Component name c) Medicine name d) Company name e) Remarks/Instruction

5.1. Testing Parameters

The various parameters that we set during the testing procedure includes: (i) the stress level, which defines the number of virtual users accessing the WS, (ii) the think time, which defines the maximum time taken by the user in thinking before requesting a parameter, (iii) the network speed, which specifies the bandwidth (BW) that the virtual user will use in the network.

5.2. Test Responses

The metrics of the load and stress test that we have monitored include: (a) the response time in seconds, (b) the throughput in bytes/sec, (c) the hits/sec and (d) the number of successful virtual users allowed for transaction.

6. Experimental Results

The test is performed for 50, 100, 150, 200, 250, 350, 500, 600, 700, 800, 1000, 1200, 1500 virtual users. We measure all the performance parameter in maximum BW. The entire performance tests are conducted with ramp up schedule with 1 virtual user operating every 15sec. The steady-state measurement period is set at 5min duration. Then they are exiting the system simultaneously after the completion of the steady-state period.

Some of the sample responses of performance test are shown in Figure 4-6. Figure 4 shows the response for hits/sec against number of users for 700 virtual users. In this case, hits/sec increases with the increase in virtual users. It becomes maximum at 694 virtual users and then the parameter decrease gradually. The recorded average hits/sec for 700 virtual users is 3.71 with a maximum of 7.125.

Figure 5 shows the response for throughput against number of users for 700 virtual users. It is observed that throughput increases with the increase in virtual users. It becomes maximum at 492 virtual users and then the parameter decrease gradually. The recorded average throughput for 700 virtual users is 16627.857 with a maximum of 28358.25.

Figure 6 shows the response for response time against number of users for 700 virtual users. It is observed that response time increases with the increase in virtual users. It becomes maximum at 498 virtual users and then the parameter decrease gradually. The recorded average response time for 700 virtual users is 11.71 with a maximum of 12.885.

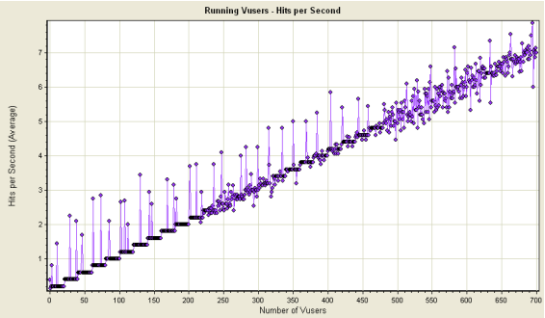

Figure 4. Hits/sec Against Number of Users for 700 Virtual Users

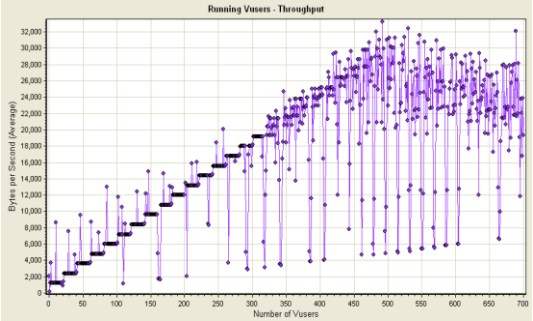

Figure 5. Throughput Against Number of Users for 700 Virtual Users

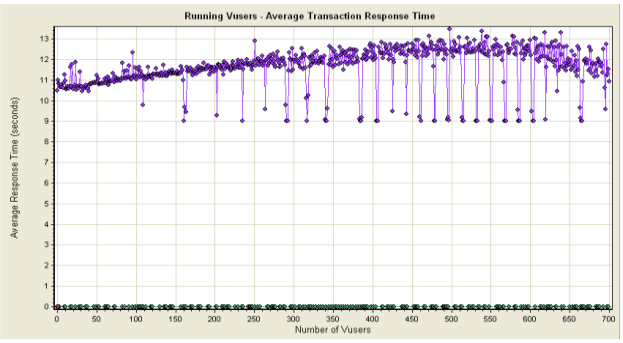

Figure 6. The Response Time Against Number of Users for 700 Virtual Users

We observed various metrics provided by the LoadRunner. The virtual user levels up to 1500 are tested to force the client application in invoking the MedWS to work beyond its capacity. The results are given in Table 2.

Table 2. Results for Select Operation on MedWS

Scenario	No. of users	Recorded parameters	Average	Connection refusal in %
Select operation	50	Response time in sec Throughput in bytes/sec Hits/sec	10.520 2328.670 0.396	0
	100	Response time in sec Throughput in bytes/sec Hits/sec	10.020 2489.270 0.681	0
	150	Response time in sec Throughput in bytes/sec Hits/sec	10.412 4726.324 0.953	0
	200	Response time in sec Throughput in bytes/sec Hits/sec	10.624 6934.176 1.181	0
	250	Response time in sec	11.564	0

		Throughput in bytes/sec Hits/sec	8334.323 1.455	
	350	Response time in sec Throughput in bytes/sec Hits/sec	11.709 11015.608 1.966	0
	500	Response time in sec Throughput in bytes/sec Hits/sec	11.991 14135.344 2.638	0
	600	Response time in sec Throughput in bytes/sec Hits/sec	10.358 9425.305 3.146	0
	700	Response time in sec Throughput in bytes/sec Hits/sec	11.816 16043.586 3.649	0
	800	Response time in sec Throughput in bytes/sec Hits/sec	11.776 16834.457 11.776	0
	1000	Response time in sec Throughput in bytes/sec Hits/sec	9.935 9853.486 4.990	1
	1200	Response time in sec Throughput in bytes/sec Hits/sec	10.734 11885.978 5.229	14
	1500	Response time in sec Throughput in bytes/sec Hits/sec	11.019 9049.034 2.969	61

7. Statistical Analysis of the MedWS

The statistical analysis for 50 users run for 5min in steady state is presented here. A sample of 30 repeated tests is taken for analysis. The recorded 30 samples are divided into six classes depending on their range. The class width and range for response time, hits/sec and throughput are given in Table 3-5, respectively.

Table 3. Class Width and Frequency for Response Time

Response time (s)	Observed frequency
10.424	1
10.9128	26
11.4016	0
11.8904	0
12.3792	0
>12.3792	3

Table 4. Class Width and Frequency for its/sec

Hits/sec	Observed frequency
0.392	1
0.648	26
0.904	0
1.16	0
1.416	0
>1.416	3

Table 5. Class Width and Frequency for Throughput

Throughput (bytes/s)	Observed frequency
2114.236	1
3606.015	26
5097.795	0
6589.574	0
8081.354	0
>8081.354	3

7.1. Distribution for Response Time, hits/sec and Throughput

Our objective for statistical analysis is to determine the distribution of response time, hits/sec and throughput. One of the ways of determination is to plot the histogram of the observed parameters as shown in Figure 7-9 respectively. According to histograms, the applied distribution is normal but slightly right skewed for response time, hits/sec and throughput.

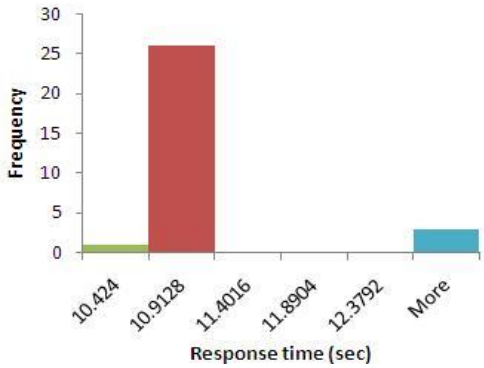

Figure 7. Histogram of Response Time

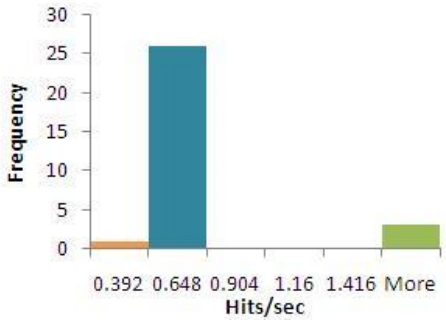

Figure 8. Histogram of hits/sec

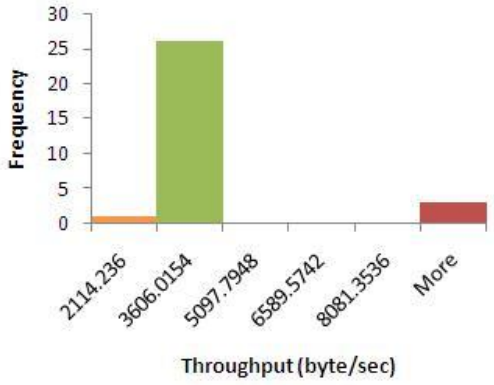

Figure 9. Histogram of Throughput

However, there is a major drawback with histograms, that is, depending on the used bin sizes; it is possible to draw very different conclusions. A better technique is to plot the observed quantiles against the recorded data in a quantile plot [19, 20]. If the distribution of observed data is normal, the plot is close to be linear. The resultant plots are shown in Figure 10-12. Based on the observed data, the response time, throughput and hits/sec do appear to be normally distributed.

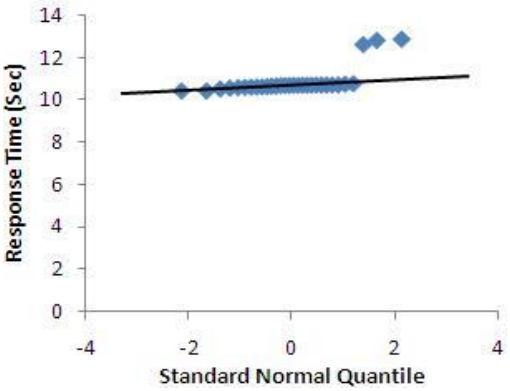

Figure 10. Quantile Plot of Response Time

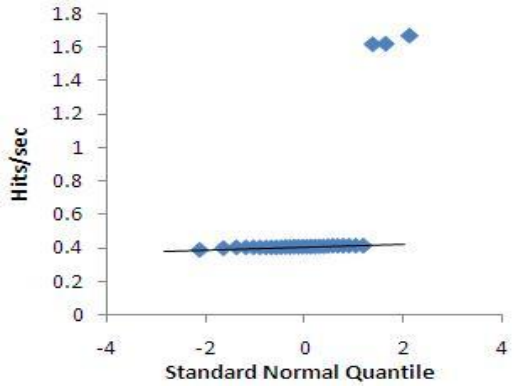

Figure 11. Quantile Plot of hits/sec

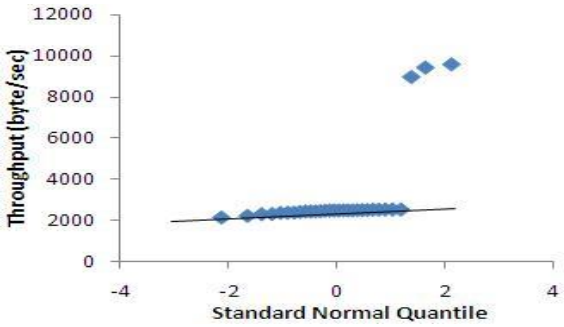

Figure 12. Quantile Plot of Throughput

The normal probability plot can be used to perform the test for normality. If the data samples are taken from a normal distribution, the plot will be appearing to be linear [17, 18, 19]. The normal probability plot for the response time, hits/sec and throughput are shown in Figure 13-15. The data follows a straight line, which predicts that the distribution is normal one.

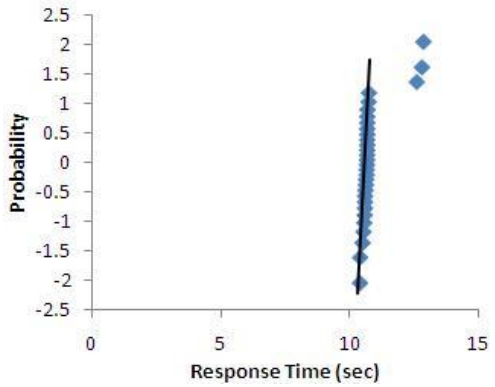

Figure 13. Normal Probability Plot for Response Time

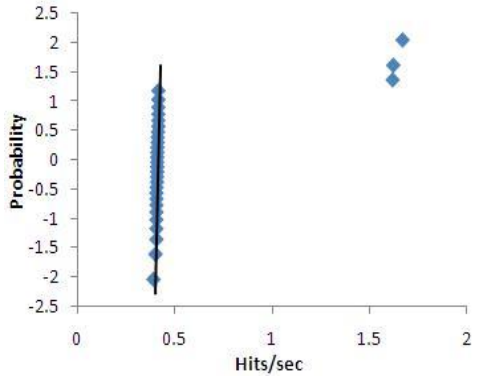

Figure 14. Normal Probability Plot for hits/sec

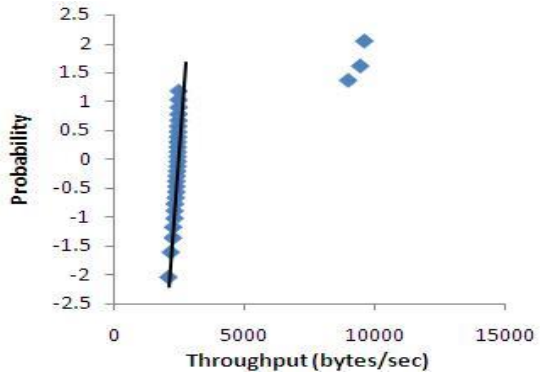

Figure 15. Normal Probability Plot for Throughput

7.2. Chi Square Test for Goodness of fit: Testing for Appropriateness of Distribution

Chi square test is performed to verify how good the frequency distribution fits to its expected distribution [21]. It enables us to test whether there is a significant difference between an observed distribution and a theoretical distribution.

To determine the goodness of fit between expected and observed data, the chi square equation can be expressed as [22]:

$$\chi^2 = \sum (f_o - f_e)^2/f_e \quad (1)$$

Where χ^2 is calculated chi square value, f_o is frequency of observation and f_e is expected frequency. The bigger the value of χ^2 the greater will be the difference between the observed and expected values. We compare the χ^2 value with a theoretical chi-square value obtained from the table presented elsewhere [23]. If the calculated χ^2 value is greater than the theoretical value then the fitness of good may be rejected.

It is observed from Table 3 above that the sample contains 3% of response values that are ≤10.424, 87% in the range of >10.424 to ≤ 10.9128, 0% in >10.9128 to ≤ 11.4016, 0% in >11.4016 to ≤ 11.8904, 0% in >11.8904 to ≤ 12.3792 and 10% are >12.3792.

From Table 4, it is observed that the sample contains 3% of hits/sec that are ≤ 0.392, 87% in the range of >0.392 to ≤ 0.648, 0% in >0.648 to ≤ 0.904, 0% in >0.904 to ≤ 1.16, 0% in >1.16 to ≤ 1.416 and 10% is >1.416.

From Table 5, it is observed that the sample contains 3% of throughput that are≤ 2114.236, 87% in the range of >2114.236 to≤ 3606.015, 0% in >3606.015 to ≤ 5097.795, 0% in >5097.795 to ≤ 6589.574, 0% in >6589.574 to ≤ 8081.354 and 10% are >8081.354.

We assume a null hypothesis H_o: the observed distribution fits the expected distribution. The alternate hypothesis H_A: the observed distribution does not fit the expected distribution.

Table 6. Chi Square Test for Response Values

Response time	Observed (f_o)	Expected (f_e)	$f_o - f_e$	$(f_o - f_e)^2$	$(f_o - f_e)^2/f_e$
≤10.424	1	3% of 30 = 0.9	0.1	0.01	0.011
>10.424 - ≤ 10.9128	26	87% of 30 = 26.1	0.1	0.01	0.00038
>10.9128 - ≤ 11.4016	0	0% of 30= 0	0	0	0
>11.4016 - ≤ 11.8904	0	0% of 30= 0	0	0	0
>11.8904 - ≤ 12.3792	0	0% of 30= 0	0	0	0
>12.3792	3	10% of 30= 3	0	0	0
				The calculated chi square	0.011

The degree of freedom (DF) is obtained by subtracting 1 from the number of levels (k) of the categorical variable: DF = k - 1. In our case it is 6-1=5. We test at the 0.05 confidence level. It is observed that the critical chi square value is 11.0705 for degree of freedom 5 and probability is 0.05.

Since the calculated χ^2 [Table 6] value is less than critical chi square value i.e. 0.011<11.0705, we accept the null hypothesis that is the data fits the expected distribution. Figure 16 shows the acceptance region of the null hypothesis.

Similarly, the chi square value for hits/sec and throughput is calculated to be 0.011 which is less than critical chi square value 11.0705 at degree of freedom 5 and probability 0.05. Hence we also accept the null hypothesis for hits/sec and throughput.

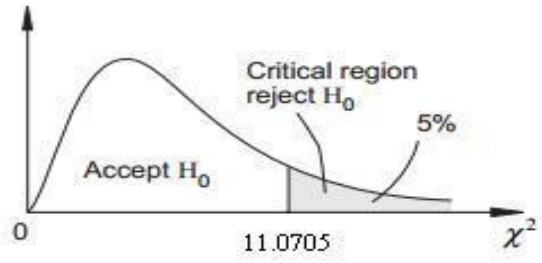

Figure 16. Goodness of Fit Test at 0.05 Level of Significance, Showing Acceptance Region

7.3. Confidence Interval of Response Time, hits/sec and Throughput

The 95% confidence interval for the mean values of response time, hits/sec and throughput are calculated using Microsoft Excel. We evaluate the number of sample size N, mean value $\bar{x}$ and confidence level. The evaluated values are given in Table 7 that are made based on different values for parameter obtained during load testing.

Table 7. Mean, Standard Deviation and Margin of Error

N	Parameters	$\bar{x}$	Standard deviation	Confidence level
30	Response time, s Hits/sec Throughput, bytes/s	10.8583 0.5333 3101.341	0.652273 0.374979 2110.863	0.233409 0.134182 755.3486

From Table 7, we can conclude that with 95% confidence level the mean of response time lies between 10.8583 ± 0.233409 that is 10.62489 and 11.09171, the mean of hits/sec lies between 0.399118 and 0.667482 and mean of throughput lies between 2345.993 and 3856.69.

7.4. Regression Analysis

To examine whether the relationship between response time, hits/sec and throughput is linear is to plot the relationship. The response time is assumed as response variable. Hits/sec and throughput are assumed to be explanatory variable. The scatter plots of response time against hit/sec and throughput are given in Figure 17 and Figure18. The two scatter plots are having straight lines which reveals linear relationship. Greater the value of hits/sec, greater will be the response time. Similarly, the greater the value of throughput more will be the response time. As the graphs present straight line, we can conclude that the relationship is linear. The combined effect of throughput and hits/sec on response time is examined by multiple linear regression tests. The regression test is carried out at 95% confidence level. We assumed the null hypothesis (H_0): response time does not depend on hits/sec and throughput. The alternate hypothesis (H_1): response time is dependent on hits/sec and throughput.

The regression analysis is carried out in Microsoft Excel. The analysis of variance shows F ratio to be 5388.135 which is greater than the critical value. The critical value from the F table at significance 0.05 is ($F_{2, 27}$) 3.35, where 2 and 27 are regression and residual,

respectively. So, we can conclude F ratio to be significant at 0.05. This provides evidence of existence of linear relationship between response time, hits/sec and throughput. As such, the null hypothesis may be rejected. It implies that the equation has 95% chance of being true. The analysis also suggests that our model accounts for 99.73% variance on response time. Thus we can conclude that the hits/sec and throughput have influence on response time.

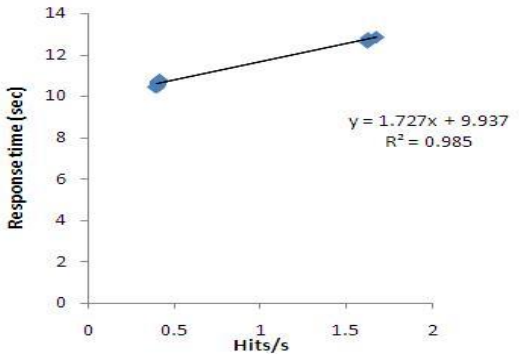

Figure 17. Response Time Against hits/sec

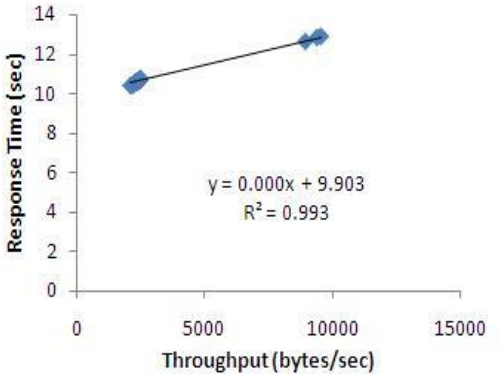

Figure 18. Response Time Against Throughput

8. Results and Discussion

The objective of our present investigation is to monitor the overall performance of hierarchical MedWS based on JAVA technique using tomcat web server and to predict the influence of the hits/sec and throughput on response time. The experimental results of the WS predict that up to 800 virtual users the service shows an ideal response without any refusal in connectivity with an average response time of 11.776sec. As we increase the number of virtual users above 800 the connection refusal is observed. For 1000 virtual users, the average response time is 9.935 sec with a connection refusal of 1%. Similarly for 1200 virtual users, the average response time is 10.734 sec and 14% connection refusal is observed. The highest connection refusal is observed at 1500 virtual users with 61% refusal with an average response time of 11.019 sec.

The increase in connection refusal at higher number of virtual users may be due to garbage collected heap, because of improper release of memory in time. Which may in turn, can also cause the decrease of server response. The sudden rise and fall of response time, throughput and hits/sec in different virtual users may be for database engine or due to not releasing or lately releasing server resources including memory for the consecutive request.

The histograms are slightly right skewed. However, the quantile plot and normal probability plots of the MedWS for response time, hits/sec and throughput, show linearity and normality which provides enough evidence for the scalability and acceptability of the MedWS communication with large number of virtual users. The normal probability plots are of straight line, which predict normality.

The chi square tests of goodness of fit using response time, throughput and hits/sec resembles that the observed data meets the expected data and hence fits the expected distribution.

From the statistical analysis it is observed that both hits/sec and throughput have individual as well as combined influence to response time. Individually, hits/sec and throughput influence approximately 98.5% and 99.3% to response time and together the effect is around 99.73%.

9. Conclusions

From our overall investigations it can be concluded that the connection refusal for the service increases with the increase of number of virtual user. The tomcat web server along with the hierarchical communications in between the WS seems to be stable up to 800 virtual users, which is acceptable in case of scalability and stability. The system gives low performance for 1500 virtual users. The collision in between request increases with the increase in stress level. With the increase in virtual users the response time increases. The throughput and hits/sec also increases gradually. The statistical analysis shows that the data distributions are normal and the observed parameters are similar to the expected parameters. The multiple regression analysis reveal that hits/sec and throughput have 99.73% combined influence on response time. It is necessary to analyze the system thoroughly with different possible test case environment and hardware resources, so that we can have an in-depth idea of the factors hampering the WS in handling more users' request.

From the above analysis we can conclude that the MedWS is stable, scalable and cost-effective. As such it may be deployed for rural health services for online prescription to inhabitants of such areas which may have very good societal impact.

Acknowledgements

The authors are thankful to the All India Council of Technical Education (AICTE), Govt. of India for financial support towards the work (F.No. 8023/BOR/RID/RPS (NER)-84/2010-2011 31st March 2011).

References

[1] Siddavatam, I., Gadge, J., "Comprehensive test mechanism to detect attack on Web Services", IEEE International Conference On Networks (ICON), **(2008)**; India.
[2] http://www.w3.org/TR/ws-arch/
[3] D. Chenthati, H. Mohanty, A. Damodaram, "RDBMS for Service Repository and Matchmaking", ISMS, 2nd International Conference., **(2011)**, pp. 300-305
[4] P.P.W. Chan, M.R. Lyu, "Dynamic Web Service Composition: A New Approach in Building Reliable Web Service", AINA, 22nd International Conference, **(2008)** , pp. 20-25.
[5] Nada Hashmi, Dan Myung, Mark Gaynor, Steve Moulton, "A Sensor-based, Web Service-enabled, Emergency Medical Response System", Proceedings of the Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services, **(2005)**, pp. 25-29.
[6] Matjaz B. Juric, Ivan Rozman, Bostjan Brumen, Matjaz Colnaric, Marjan Hericko, "Comparison of performance of Web services, WS-Security, RMI, and RMI–SSL", The Journal of Systems and Software , Elsevier, vol.79, **(2006)**, pp. 689–700.
[7] Belkhir Abdelkader, Bouyakoub Samia, "A Hierarchical Model for Web Services Composition", International Journal of Web Services Practices, vol. 4, no. 1, **(2009)**, pp. 44-50.
[8] V. Stoicu-Tivadar, L. Stoicu-Tivadar, D. Berian, V. Topac, "Web Service-based solution for an intelligent telecare system", Intelligent Engineering Systems (INES), 14th International Conference on IEEE Conference Publications, **(2010)** , pp. 313 – 316.
[9] K. E. Mohamed and D. Wijesekera, "Performance Analysis of Web Services on Mobile Devices", The 9th International Conference on Mobile Web Information Systems, Procedia Computer Science vol. 10 , **(2012),** pp. 744 – 751.
[10] Lorenzo Bianchi, Federica Paganelli, Maria Chiara Pettenati, Stefano Turchi, Lucia Ciofi, Ernesto Iadanza, and Dino Giuli, "Design of a RESTful Web Information System for Drug Prescription and Administration", IEEE journal of biomedical and health informatics, vol. 18, no. 3, **(2014)**, pp. 885-895.
[11] Drug Index, Passi Publications, India, January-March, **(2012)**.
[12] A.S. Christensen, A. Moller, M.I. Schwartzbach, "Extending Java for high level web service construction", ACM Trans. Program. Lang. Syst., vol. 25, no. 6, **(2003)**, pp. 814-875.
[13] http://www.javasamples.com/showtutorial.php?tutorialid=350
[14] http://www.oracle.com/technetwork/java/index-jsp-137004.html

[15] A. Bora, M.K. Bhuyan, T. Bezboruah, "Investigations on Hierarchical Web Service based on Java Technique", Proc. World Congress on Engineering (WCE), London, U.K., vol.2, **(2013)** , pp. 891-896.
[16] Application-testing tool: Mercury LoadRunner 8.0, Available at: http://pcquest.ciol.com/content/software/2004/104093002.asp
[17] M. Kalita, T. Bezboruah, "Investigation on performance testing and evaluation of PReWebD: a .NET technique for implementing web application", IET Softw., vol. 5, no. 4, **(2011)**, pp.357-365.
[18] M. Kalita, S. Khanikar, T. Bezboruah, "Investigation on performance testing and evaluation of PReWebN: a JAVA technique for implementing web application", IET Softw., vol. 5, no. 5, **(2011)**, pp. 434-444.
[19] M. Kalita and T. Bezboruah, "Investigation on implementation of web applications with different techniques", IET Softw., vol. 6, no. 6, **(2012)**, pp. 474-478.
[20] A. Bogardi Meszoly, Z. Szitas, T. Levendovszky and H. Charaf, "Investigating factors influencing the response time in ASP.NET web applications", LNCS, **(2005)**, pp. 223-233.
[21] http://fsweb.bainbridge.edu/dbyrd/statistics/goodnessfit.htm
[22] I. Levin, S. Richard, D. Rubin, Statistics for management, Pearson education, Inc., South Asia, **(2009)**.
[23] http://www.statisticslectures.com/tables/chisquaretable/

Authors

Abhijit Bora, Research Scholar, Department of Electronics and Communication Technology, Gauhati University, India received Master of Computer Applications (MCA) degree from Jorhat Engineering College (Under Dibrugarh University), India in 2008. His research interests include web service, web security and software engineering.

Tulshi Bezboruah (M'12) received the B.Sc. degree in physics with electronics from the University of Dibrugarh, Dibrugarh, India, in 1990, and the M.Sc. and Ph.D. degrees in electronics and radio physics from the University of Gauhati, Guwahati, India, in 1993 and 1999, respectively. In 2000, he joined in the Department of Electronics Science, Gauhati University, as a Lecturer. He is currently the Professor & Head, Department of Electronics and Communication Technology, Gauhati University. His current research interests include instrumentation and control, distributed computing, and computer networks. Prof. Bezboruah is a member of the IEEE Geoscience and Remote Sensing Society as well as an Associate Member of the International Center for Theoretical Physics, Trieste, Italy.

