# Some Aspects of Reliability for Computational Implementation in Hierarchical Service Oriented Computing

**Abhijit Bora**[1] **and Tulshi Bezboruah**[2]

*[1,2]Department of Electronics & Communication Technology, Gauhati University, India*
*E-mail: [1]abhijit.bora0099@gmail.com, [2]zbt_gu@yahoo.co.in*

**Abstract—***We design, develop and implement a hierarchical SOAP based web service as Service Oriented Computing platform for clinical advice using Tomcat web server, MySQL database server with JAVA programming language. The reliability of the system is evaluated to predict the operability of the service against massive request in a day. We present here in detail the methodology, the experimental arrangement, the observed metrics overheads and the statistical analysis of the service's failure activities.*

## 1. INTRODUCTION

Implementation of Service Oriented Architecture (SOA) for Service Oriented Computing (SOC) is gaining popularity among organizations due to its flexibility and adaptability nature over network connectivity. Traditional technique of developing service based system needs high cost, long development time, product quality etc. Alternative techniques have been implemented for quick delivery of the service with extreme and agile methodology of programming. Among them, the Web Service (WS) is most popularly used which can enhance the distributed computing over internet. Among SOA and WS, the service can be atomic or composition of more than one service [1]. They can serve complex composition of service over internet. However, the reliability evaluation of such service computing is also gaining popularity among users, software practitioners and research communities. The WS reliability is defined as the probability that the WS can respond properly within some specific exposure period of access time. Different kinds of reliability models have been introduced for years [2]. The classification of the reliability model is made as per Software Development Life Cycle (SDLC). This paper emphasizes on an evaluation framework and reliability results that we have achieved in our experiment.

## 2. RELATED WORKS

In the year 2005, Nicanor et al. had illustrated a reliability analysis methodology for a web based application. They had implemented a statistical analysis technique to study the system reliability nature by comparing real and ideal experimental scenario as in [3].

In the year 2006, Juric et al. had presented a comparative analysis for WS and Remote Method Invocation (RMI) implementation in a server machine. The performance and functional complexities were discussed in this study as in [4].

In the year 2006, A. E. Saddik had implemented SOAP based WS as an experimental set up and monitored the scalability and performance of the system against different load level of agents as in [5].

In the year 2013, Mirandola et al. had worked on a methodology to evaluate the reliability of heterogeneous assembly of SOC based applications. Their experimental results pointed out the applicability of the method for accurately estimating the system reliability as in [6].

In the year 2014, Medhi et al. had presented the performance results of hierarchical SOAP based WS for operability of the system using. NET framework as in [7]. The study lacks of reliability aspects in WS.

In the year 2015, Bezboruah et al. had carried out a comparative performance study of hierarchical WS against massive consumers. They used cluster based and non cluster based load balancing tomcat web server for the computational system as in [8]. The study lacks of reliability analysis against high load of users.

In the year 2015, Singh et al. had studied web based instrumentation system for information retrieval from database and discussed some performance key parameters that influence the system as in [9]. The authors did not study any failure nature of the system, and hence lack of reliability evaluation.

## 3. OBJECTIVE AND METHODOLOGY OF RELIABILITY EVALUATION

Proposing a methodology to evaluate the reliability of hierarchical SOC is the main objective of this study. To establish the applicability of the method, we design, develop and implement a hierarchical SOC based system to retrieve the data from the database. We follow the methodology of the architecture as given in Fig. 1. The architecture solely maintains the role of parent, child and broker nature of WS. We develop a prototype database for clinical advice that contains 15000 record of disease medicine mapping. We use this database to provide searched information whenever there is a valid request made from end user through the architecture. Business logic (BL) method is developed for executing the necessary query instructions. We create a test script using Mercury Load Runner testing tool, to access the system, and that too can monitor the Hyper Text Transfer Protocol (HTTP) transaction status. The test script is executed against massive virtual user (VU). The VU access the system and follows the instructions as given in the test script. Each request and response creates a SOAP message which is forwarded back and forth in between WS along with the information. The SOAP request and response message structure that is generated while communicating WS is captured through Wireshark Network analyzer tool [10]. The message samples are given in Figures 2-3.The ramp up duration of 5 min with 10s think time is taken for monitoring the load metrics.

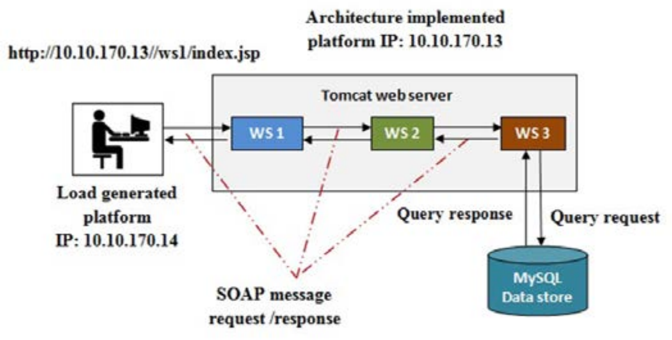
Architecture implemented platform IP: 10.10.170.13
http://10.10.170.13//ws1/index.jsp
Tomcat web server
WS 1
WS 2
WS 3
Load generated platform IP: 10.10.170.14
Query response
Query request
MySQL Data store
SOAP message request /response

**Fig. 1: Experimental arrangement**

```
<?xml version= '1.0' encoding = 'UTF-8' ?>
<S:Envelopexmlns:S=http://schemas.xmlsoap.org/
soap/envelop/ >
<S:Body>
<ns2:hello xmlns:ns2= "http://org/" >
<name>
Cold
</name>
</ns2:hello>
</S:Body>
</S:Envelope>
```

**Fig. 2: Sample of SOAP requested message structure**

## 4. SOFTWARE AND HARDWARE BENCHMARK FOR EXPERIMENTAL SETUP

The software and hardware specifications that are used for hosting the architecture and generating the load are given below.

**Architecture implementation platform:** Processor: Intel®Xeon®CPUE5620 @ 2.40 GHz processor speed; RAM: 8GB; Memory storage: 600GB; Web server: Apache Tomcat (version 7); Database software: MySQL (version 5.0); Operating system: Windows Server 2008 (64-bit) R2 Standard; Software support tool: NetBeans (version 7.0) Integrated Development Environment (IDE); Java Development Kit (JDK) (version 7.0) and Java Runtime Environment (JRE) (version 7.0).

**Load generation platform:** Processor: Intel® Pentium® Dual CPUE2200 @ 2.20 GHz processor speed; RAM: 1GB; Memory storage: 150GB; Operating system: Windows XP Professional Service Pack 2; Load testing tool: Mercury Load Runner version 8.1.

### Testing results

The different results of HTTP transaction status against VU are shown in Table 1. The first transaction failure is shown against 1000VU. Hence, we collected 30 days sample to study the failure distribution against 1000 VU.

```
<?xml version= '1.0' encoding = 'UTF-8' ?>
<S:Envelopexmlns:S=http://schemas.xmlsoap.org/so
ap/envelop/ >  <S:Body>
<ns2:helloresponse xmlns:ns2= "http://org/" >
<return>
Clinical instructions for Cold fetched from database
in tabular format
</return>
</ns2: helloresponse> </S:Body>
</S:Envelope>
```

**Fig. 3: Sample of SOAP response message structure**

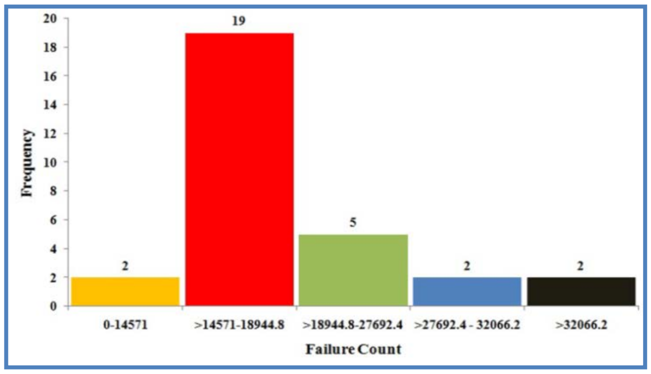

**Fig. 4: Histogram of failure count**

**Table 1: HTTP transaction status against VU**

| Test case | VU | Total transaction | Pass transaction | Fail transaction |
|---|---|---|---|---|
| SQL Select test case | 30 | 252 | 252 | 0 |
| | 100 | 1180 | 1180 | 0 |
| | 200 | 3360 | 3360 | 0 |
| | 400 | 10720 | 10720 | 0 |
| | 800 | 19587 | 19587 | 0 |
| | 1000 | 55158 | 37888 | 17270 |
| | 1100 | 66502 | 40043 | 26459 |

**Table 2: Frequency of failure**

| Failure ranges | Frequency |
|---|---|
| 0- 14571 | 2 |
| >14571 - 18944.8 | 19 |
| >18944.8 - 27692.4 | 5 |
| >27692.4 - 32066.2 | 2 |
| >32066.2 | 2 |

To better understand the sample, we draw the histogram of the collected transaction failure data. The histogram is shown in Fig. 4. The frequency table is given in Table 2. It is observed that the highest density of failure lies in the range of >14571 to 18944.8. The histogram is slightly right skewed. It is observed from histogram that the failure count (FC) distribution is weilbull distribution.

To estimate the shape (α) and scale parameter (β) of our weibull distributed data sample, we use EasyFit tool version 5.6 [11]. The α and β is observed to be 3.796 and 19901. The weibull cumulative distribution function (CDF) of each FC is calculated using (1) [12, 13].

$$\text{CDF}= 1- \exp \{(-\frac{FC}{\beta})^{\alpha} \} \quad (1)$$

## A. Goodness of Fit (GoF) evaluation using Kolmogornov Smirnov (KS) test

The KS test allows verifying whether the observed sample is from a specified continuous distribution. According to this test, the difference between the observed CDF and the expected CDF should be small. It defines two metrics K+ and K- as given below.

$$K+ = \sqrt{n} \max \left(\frac{j}{n}-x_j\right) \quad (2)$$

$$K- = \sqrt{n} \max \left(x_j - \frac{j-1}{n}\right) \quad (3)$$

“The K+ and K- describe the maximum deviation when the observed CDF is above and below the expected CDF. If the calculated K+ and K- values are smaller than KS (1±confidence level; n), then we can conclude that n observations are coming from a specified distribution at ± level of significance” [14]. The observed CDF and different GoF results for KS test is given in Table 3.

K+ and K- are calculated to be 1.31 and 1.42 respectively. The KS (0.99; n) value for n= 30 and at 99% confidence level is 1.4801. Since the computed statistics K+ and K- both less than the value from the KS table, the sequence of FC passes the KS test at this level of significance. That means we do not reject the hypothesis that the CDF of the FC follows the weibul distribution at (α=3.796, β=19901). Hence the population from where these data are obtained is distributed weibull.

**Table 3: KS GoF test for adequacy of distribution (x: Observed CDF; j: Rows; n: number of dataset i.e. 30 )**

| j | x | j/n | j-1/n | j/n-x | x-( j-1)/n |
|---|---|---|---|---|---|
| 1 | 0.264 | 0.03333 | 0 | -0.23 | 0.26 |
| 2 | 0.264 | 0.06667 | 0.03333 | -0.20 | 0.23 |
| 3 | 0.266 | 0.10000 | 0.06667 | -0.17 | 0.20 |
| 4 | 0.274 | 0.13333 | 0.10000 | -0.14 | 0.17 |
| 5 | 0.276 | 0.16667 | 0.13333 | -0.11 | 0.14 |
| 6 | 0.277 | 0.20000 | 0.16667 | -0.08 | 0.11 |
| 7 | 0.277 | 0.23333 | 0.20000 | -0.04 | 0.08 |
| 8 | 0.279 | 0.26667 | 0.23333 | -0.01 | 0.05 |
| 9 | 0.283 | 0.30000 | 0.26667 | 0.02 | 0.02 |
| 10 | 0.283 | 0.33333 | 0.30000 | 0.05 | -0.02 |
| 11 | 0.286 | 0.36667 | 0.33333 | 0.08 | -0.05 |
| 12 | 0.286 | 0.40000 | 0.36667 | 0.11 | -0.08 |
| 13 | 0.296 | 0.43333 | 0.40000 | 0.14 | -0.10 |
| 14 | 0.310 | 0.46667 | 0.43333 | 0.16 | -0.12 |
| 15 | 0.322 | 0.50000 | 0.46667 | 0.18 | -0.14 |
| 16 | 0.349 | 0.53333 | 0.50000 | 0.18 | -0.15 |
| 17 | 0.349 | 0.56667 | 0.53333 | 0.22 | -0.18 |
| 18 | 0.391 | 0.60000 | 0.56667 | 0.21 | -0.18 |
| 19 | 0.391 | 0.63333 | 0.60000 | 0.24 | -0.21 |
| 20 | 0.442 | 0.66667 | 0.63333 | 0.22 | -0.19 |
| 21 | 0.487 | 0.70000 | 0.66667 | 0.21 | -0.18 |
| 22 | 0.569 | 0.73333 | 0.70000 | 0.16 | -0.13 |
| 23 | 0.589 | 0.76667 | 0.73333 | 0.18 | -0.14 |
| 24 | 0.598 | 0.80000 | 0.76667 | 0.20 | -0.17 |
| 25 | 0.664 | 0.83333 | 0.80000 | 0.17 | -0.14 |
| 26 | 0.863 | 0.86667 | 0.83333 | 0.00 | 0.03 |
| 27 | 0.979 | 0.90000 | 0.86667 | -0.08 | 0.11 |
| 28 | 0.987 | 0.93333 | 0.90000 | -0.05 | 0.09 |
| 29 | 0.998 | 0.96667 | 0.93333 | -0.03 | 0.07 |
| 30 | 1.000 | 1.00000 | 0.96667 | 0.00 | 0.03 |
| | | | Maximum | 0.24 | 0.26 |

## B. Confidence interval of CDF for reliability evaluation

We estimate the mean value of CDF at 95% confidence interval for 1000 VU. The population means μ can be represented as [15, 16, 17].

$$\mu = \bar{x} \pm t_c SD/\sqrt{N} \quad (4)$$

In equation (4), we consider the mean CDF value as $\bar{x}$, the critical value from $t_{c(0.05,29)}$,the standard deviation as SD, the sample size as N and the margin of error as $t_c SD/\sqrt{N}$. We consider the different observed CDF obtained from Table 3 to evaluate the critical value, mean and margin of errors. The estimated values are given in Table 4. The population mean μ is calculated from equation (4). From Table 4, we can conclude the following with 95% confidence that mean CDF for a load of 1000 VU lies between 0.4633 ± 0.095 that is 0.559 and 0.368.

**Table 4: Estimated values for μ**

| N | $t_{c(0.05,29)}$ | Parameter | $\bar{x}$ | SD | $t_c SD/\sqrt{N}$ |
|---|---|---|---|---|---|
| 30 | 2.045 | CDF | 0.4633 | 0.255 | 0.095 |

### C. Reliability metrics

The reliability metrics (R) for weibull distribution is calculated using (5) [18].

$$R = 1 - CDF \tag{5}$$

Hence average reliability is calculated to be 1- $\bar{x}$ i.e. 0.537 and lies in between 0.441 and 0.632. That means the SOC will respond successfully for an average of 53.7% of execution phase. However, the overall reliability may be estimated in between 44.1% to 63.2% of execution phase against massive users of 1000 VU in one day.

## 5. OVERALL RELIABILITY AND DISCUSSION

The overall assessment of reliability against 1000 VU is shown in Table 5. The SOC will respond successfully for a load of VU less than 1000 in one day. Since there is no failure activity generated below 1000 VU, strong reliability is estimated for the system.However for a load of 1000VU or more than that, some failure of HTTP transactions is observed. From Table 1, it is observed that, with increase in VU, the HTTP transaction increases. As with increase in VU, number of incoming request increases for which server side transaction increases. It is also seen that for VU stress of 1000, 1100 the failure transaction is generated. This may be due to collision of request processing in server side. With the increase in HTTP transactions, server side resource utilization also increases for which garbage collected heap error arises. With the occurrences of heap error in server side, some of the transaction is not allowed to complete its operational circle, for which it throws exceptions. It is observed that, for a stress of 1000 VU, out of 55158 HTTP transactions, 37888 had passed and 17270 had failed in computational processing. For a load of 1100 VU, the passed and failed transaction is 40043 and 26459 respectively out of 66502 HTTP transactions. These metrics reveals that after some stable VU stress level, the FC increases gradually with increase in consumers. From the statistical analysis, the collected FC data sample against 1000 VU is observed to be weibull distribution.

From histogram it is observed that, the highest FC lies in between >14571 to 18944.8 in one day. KS GoF test reveals the adequacy of the nature of the distribution. At 99% confidence level, it can be concluded that the FC follows weibull distribution.

**Table 5: Overall reliability assessment**

| Experimental observations | Results |
|---|---|
| HTTP transaction failure against 30, 100, 200, 400,800 VU | NIL |
| HTTP transaction failure against 1000,1100VU | Arises and increases gradually |
| Histogram of FC | Right Skewed with highest failure density in between >14571 to 18944.8 |
| Distribution nature | Weibull |
| KS test at 99% confidence level | K+ and K- value < KS $_{(0.01, 30)}$ value. Weibull distribution fits the failure data |
| CDF $\bar{x}$ | 0.4633 |
| CDF μ | 0.4633 ± 0.095 i.e. lies between 0.559 and 0.368 |
| Estimated overall reliability of SOC | Lies in between 44.1% to 63.2% |
| Reliability upto 800 VU per day | R=1; Strong reliability; User will get searchable information without failure |
| Reliability against 1000VU per day | R=0.537; moderate reliability with a probability of dissatisfactory information |
| Reliability against >1000VU per day | Degrades gradually; Dissatisfactory information might arise frequently |

The reliability is calculated to be 53.7% for the SOC. It is assumed that the system will respond properly for 53.7% of its execution circle with the probability that the consumer may not get the searched information from server side. The reliability degrades beyond the stress level of 1000 VU. The degradation of reliability may be due to parsing error of SOAP message, database engine error, memory management error and application server resource management error occurrence in server side, which increases proportionally to stress level.

According to our model only 53.7% of the sessions can be completed successfully against 1000 VU. In all other cases, the user observes request failure and dissatisfactory response generated without providing the searched information.

## 6. CONCLUSION

The reliability for computational implementation in hierarchical SOC is estimated. It is observed that the model shows strong reliability up to 800 VU stress level, and then it degrades gradually. Weibull distribution is accepted as it adequately describes the failure data of hierarchical SOC against massive user at 0.01 significance level. As such the proposed methodology can be accepted as suitable for assessing some aspects of reliability for implementation of computation in hierarchical SOC. It is necessary to evaluate the experimental system in different configuration of hardware and software specification and with different complex test cases, SOAP message structures to study the factors hampering the SOC based system along with application server and database engine to satisfy more HTTP request.

## 7. ACKNOWLEDGEMENTS

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

## REFERENCES

[1] Gordon, A. D. and Pucella, R., "Validating a Web Service Security Abstraction by Typing", *Microsoft MSR-TR-2002-108*, December 2002.

[2] Ramamoorthy, C. V. and Bastani, F. B., "Software Reliability — Status and Perspectives", *IEEE, Trans. Soft. Eng.*, SE-8, No. 4, July 1982, 354 - 371.

[3] Nicanor, L. D. and Mejia-Alvarez, P., "Reliability evaluation of Web-based software applications,"*Proc. Of Sixth Mexican International Conference on Computer Science (ENC 2005)*, 26-30 Sept. 2005, pp.106-112, DOI: 10.1109/ENC.2005.36

[4] Juric, M. B., Rozman, I., Brumen, B., Colnaric, M. and Hericko, M., "Comparison of performance of Web services, WS-Security, RMI, and RMI–SSL", *The Journal of Systems and Software*, vol. 79, 2006, pp. 689-700.

[5] Saddik, A.El, "Performance measurement of Web Service based application", *IEEE Transactions on Instrumentation and Measurement*, vol. 55, issue. 5, 2006, pp. 1599 – 1605, DOI:10.1109/TIM.2006.880288

[6] Mirandola, R., Potena, P., Riccobene, E. and Scandurra, P., "A reliability model for Service Component Architectures", *TheJournal of Systems and Software, Elsevier* vol. 89, 2014, pp. 109-127.

[7] Medhi, S. and Bezboruah, T., "Investigations on implementation of e-ATM Web Services based on. NET technique", *International Journal of Information Retrieval Research*, vol. 4, issue. 2, April-June 2014, pp. 42-51.

[8] Bezboruah, T. and Bora, A., "Performance Evaluation of Hierarchical SOAP Based Web Service in Load Balancing Cluster-Based and Non-Cluster-Based Web Server", *International Journal of Information Retrieval Research*, vol. 5, issue. 4, 2015, pp. 20-31, DOI: 10.4018/IJIRR.2015100102

[9] Singh, H. K. and Bezboruah, T.,"Performance Metrics of a Customized Web Application Developed for Monitoring Sensor Data", *In Proc. Of theIEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS)*, July 2015, pp. 157-162.

[10] Wireshark, available at http://www.wireshark.org

[11] Easy Fit Distribution, available at http://www.mathwave.com/easyfit-distribution-fitting.html

[12] Chandra, M., Singpurwalla, N.D. and Stephen, M.A., "Kolmogorov Statistics for Tests of Fit for the Extreme Value and Weibull Distribution", *Journal of the American Statistical Association, Taylor & Francis*, vol. 76, no. 375, Sep. 1981, pp. 729-731

[13] Coles , S. G., "On Goodness-of-Fit Tests for the Two-Parameter Weibull Distribution Derived from the Stabilized Probability Plot", *Biometrika*, vol. 76, no. 3, Sep. 1989, pp. 593-598

[14] Jain, R, "The art of computer systems performance analysis: Technique for experimental Design, Measurement, Simulation and Modeling", *Wiley Professional Computing*, 2012, pp. 462-465

[15] Spiegel, M. R., "Theory and problems of probability and statistics, Schaum's Outline Series", *McGraw- Hill Book Company, SI edn*, 2000

[16] Pal, S. K.,"Statistics for Geo scientists, techniques and applications", *Concept Publishing Company*, 1998

[17] Kalita, M. and Bezboruah, T., "Investigation on performance testing and evaluation of PReWebD: a. NET technique for implementing web application",*IET Softw.*, vol. 5, issue. 4, 2011, pp. 357-365, DOI: 10.1049/iet-sen.2010.0139

[18] NIST/SEMATECH e-Handbook of Statistical Methods, http://www.itl.nist.gov/div898/handbook/, 29.8.2015. (Accessed at http://www.itl.nist.gov/div898/handbook/apr/section1/apr162.htm)