PREDICTING PERFORMANCE OF WEB SERVICES USING SMTQA

Ch Ram Mohan Reddy[1], D Evangelin Geetha[2], KG Srinivasa[2],
T V Suresh Kumar[2], K Rajani Kanth[2]

[1]B M S College of Engineering, Bangalore - 19, India
[2]M S Ramaiah Institute of Technology, Bangalore - 54, India
crams19@yahoo.com, degeetha@msrit.edu, kgsrinivas78@yahoo.com,
tvsureshkumar@msrit.edu, rajanikanth@msrit.edu

ABSTRACT

Web Service is an interface which implements business logic. Performance is an important quality aspect of Web services because of their distributed nature. Predicting the performance of web services during early stages of software development is significant. In this paper we model web service using Unified Modeling Language, Use Case Diagram, Sequence Diagram. We obtain the Performance metrics by simulating the web services model using a simulation tool Simulation of Multi-Tier Queuing Architecture. We have identified the bottle neck resources.

KEYWORDS

Software Performance Engineering, Web Services, Unified Modeling Language, Simulation

1. Introduction

Web Services plays a very important role in Internet Technology. A Web service [9] is defined as an interface which implements the business logic through a set of operations that are accessible through standard set of Internet protocols. The eXtensible Markup Language (XML) based protocols namely Simple Object Access Protocol (SOAP) [15], Web Service Description Language (WSDL) [1] and Universal Description, Discovery and Integration (UDDI) [17] are the three major building blocks of Web services. The conceptual Web services architecture is presented in figure 1. It is based upon the interactions between three roles: service provider, service registry and service requester [11].

Performance is a one of the major problem in Web Services. Web service frameworks do not include the functionality required for web service execution performance measurement from an organization perspective. Here it is essential to address the performance is full in the context of Web Services. It is also important to discuss performance prediction techniques which are useful in reducing the cost for development. We use Software Performance Engineering (SPE) for Web Service.

SPE is a methodology to predict the performance of software systems early (analysis phase) in the life cycle [20]. SPE continues through the detailed design, coding and testing stages to predict and

D.C. Wyld, et al. (Eds): CCSEA 2011, CS & IT 02, pp. 58–66, 2011.

DOI: 10.5121/csit.2011.1207

manage the performance of the evolving software and to monitor, report actual performance against specifications and predictions. SPE is important for software engineering and in particular for software quality. The Software Performance Engineering Process uses multiple performance assessment tools depending on the state of the software and the amount of performance data available.

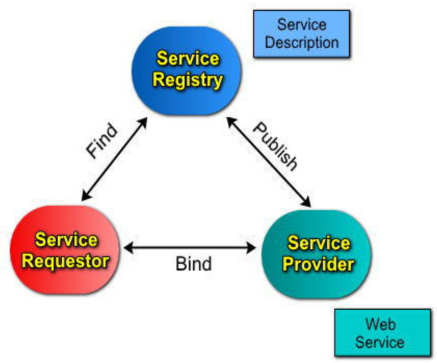

Figure 1. Web services architecture [22].

SMTQA (Simulation of Multi-tiered Queuing Applications) is a process-oriented simulation tool, developed for the performance evaluation of software that follows multi-tier architecture [8]. It provides full visualization of model structure, parameters and output reporting. It addresses the following issues in a distributed environment.

- Simulate the multi-tier architecture with open workload and multi-classes
- Consider the software resource requirements of Use case Performance Engineering (UPE) approach.
- Simulate the behavior of the servers with replicas
- Balance the workload among the replicas using dynamic load balancing algorithm
- Obtain the performance metrics such as server utilization, average response time, average waiting time, average service time, probability of idle server and probability of dropping of requests required for capacity planning
- Generate the graphs for analyzing the performance metrics.

2. Related Work

Meeting Performance Requirements is a key factor towards acceptability of given software. The performance of the SOA is mainly studied as web service performance, since the web service is the one of the key enabling technologies of SOA. A prototype service oriented application has been implemented and the actual performance is measured in [18]. Kohl Hoff's work focuses on the performance evaluation and analysis of the SOAP protocol in web service [5]. Spitznagel and Garlan have used Queuing network for analyzing simple client server system in [3]. Garlan et al. illustrated how formal approaches to software architecture can lead to enhancements in software quality, including improved clarity of design, support for analysis, and assurance that implementations conform to their intended architecture in [6]. Gamble's work focuses on detecting architectural mismatches between web services by generating a minimal web service architectural style [4].

Due to the complexity of processors and micro architectures, simulations are used to predict their performance [10]. In the context of processor simulation two approaches exist, i.e. trace-driven simulation and execution-driven simulation. Trace-driven simulation uses captured or synthetically generated trace files as input and simulates their timing behavior on a modeled system. This approach is an old technique and widely used [10], [13]. Execution-driven simulation uses soft-ware programs as input and simulates their functional execution. Simple Scalar [21], [7] is an example for this approach. The execution-driven approach suffers from the drawback of a fix instruction set and the necessity to port operating systems and drivers to the simulation framework. In this paper, we have used model based simulation, since the prediction of performance is done during feasibility study of Software Development Life Cycle (SDLC).
Software performance engineering methods (SPE) [2] use annotated Unified Modelling Language (UML) diagrams to model the system and soft-ware under study [12], [14], [19]. Since UML does not allow for the modeling of nonfunctional aspects many authors apply the UML Profile for schedulability, Performance, and Time Specification (SPT) [16] to enhance the diagrams with the necessary semantics [12], [19]. The UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE) is the successor of the SPT profile, allows for a detailed modeling of performance aspects, and supports UML 2.

We have modelled the functionalities of web services in general using UML 2.0 use case diagram and sequence diagram.

3. Methodology

To assess the performance of web services, we have exploited the methodology given in [use case point]. The web services proposed steps involved in prediction are as follows.

1. Develop the use case model for the general web services.
2. Generate the sequence diagrams, for representing the flow of events in each use case.
3. Consider the execution environment of the software components.
4. Simulate the model using SMTQA and obtain performance metrics.

4. Application of Methodology

The Use Case model of Web Services and the corresponding sequence diagrams are given in figure 2 and figure 3 respectively.

Four steps involved in the process of engaging a Web service are:

a. The requester and provider must be known to each other to initiate communication between them.

   There are two cases.

   1 In a typical case, the requester who would initiate the process must be aware of service provider.

     There are two ways:

     - The requester may get provider's address directly from the provider
     - The requester may use a discovery service

   2 In other case, the provider may initiate communication between requester and provider. They would get to know each other wherein the provider agent somehow obtains address of the requester agent.

b. If the requester and provider agree on service description (a WSDL document), policy constraints would make successful communication between requester and provider. However, it does not assure that requester and provider communicate with each other. It simply suggests that both must have the same policies of the service description.

   There are different ways this could be achieved:

   - Through direct communication between requester and provider, to know the service description and polices.
   - Requester must accept the policies of service provider.
   - Both requester and provider must follow an industry defined standard.
   - Service description and polices defined and published by the requester and offered to provider.

c. The service description and polices are input to, or embodied in, both the requester and the provider as required.

d. SOAP messages can be communicated between requester agent and provider agent on behalf of their service provider and service requester.

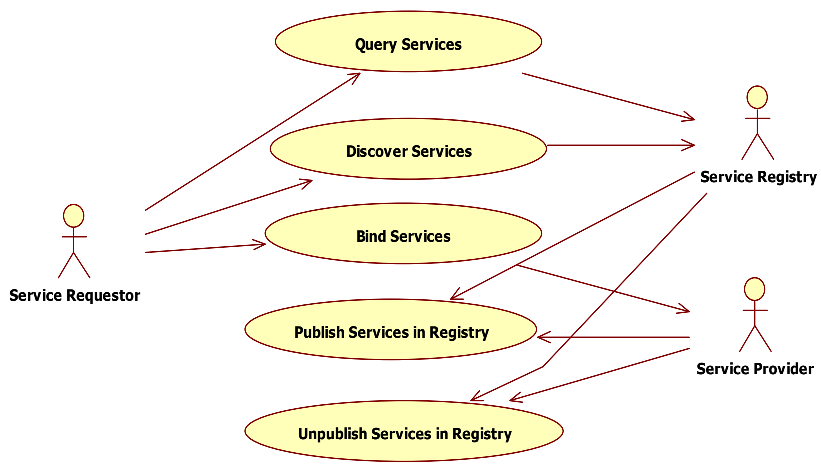

Figure 2. Use Case model for Web Services.

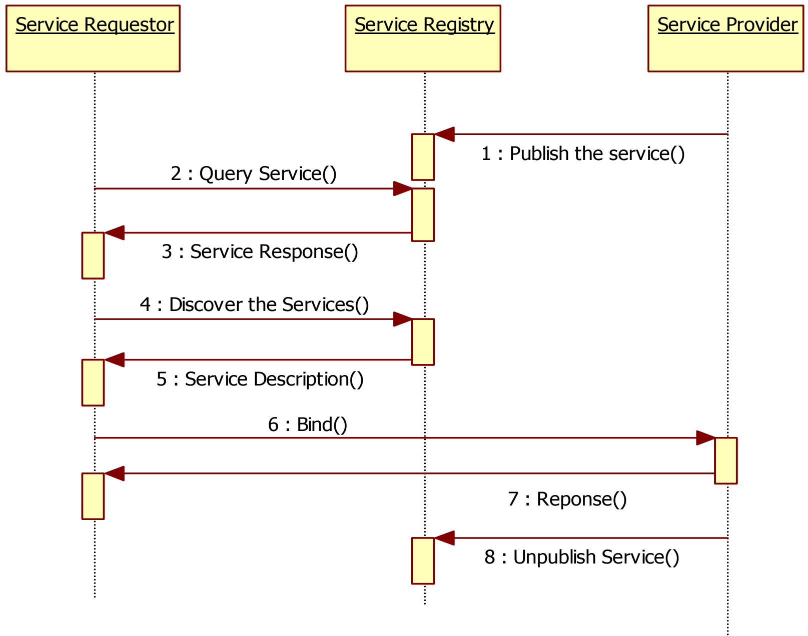

Figure 3. Sequence Diagram for Web Services.

5. Simulation Results

Since the Web Services Technology is layered in nature, the simulation model is designed as multi-tier architecture. The model is simulated using the tool SMTQA [8]. The model is simulated for 1000 requests. 3-tier architecture is considered. The layers considered are service requests, service registry and service provider. The performance metrics; resource utilization, average response time, average service time, average waiting time is obtained. The graphs are generated for the performance metrics against arrival rate. In figures, the graphs for average response time against arrival time obtained in resources of the tiers are presented. From these figures, we could observe that in Internet 1 and Internet 2, the response time is high compared to other resources. The obtained performance metrics for web services is presented in Table 1. From this table, we could observe that Internet 1, Internet 2, Service Registry Disk are the bottleneck resources, due to the performance metrics, the Average Waiting Time and Probability of dropping of sessions.

Table 1 Performance Metrics obtained for Web Services using SMTQA

	Average Response Time	Average Service Time	Average Waiting Time	Probability of Idle Server	Probability of dropping of Sessions
SRS	0.00012	0.00012	0.000	0.013	0.000
Internet 1	0.207	0.057	0.150	0.003	0.761
SPCPU	0.00002	0.00002	0.000	0.044	0.000
SBDsk	0.016	0.016	0.000	0.040	0.000
Internet 2	0.291	0.077	0.214	0.000	0.723
SPCPU	0.86	0.86	0.000	0.035	0.000
SBDsk	2.381	1.500	0.881	0.011	0.789

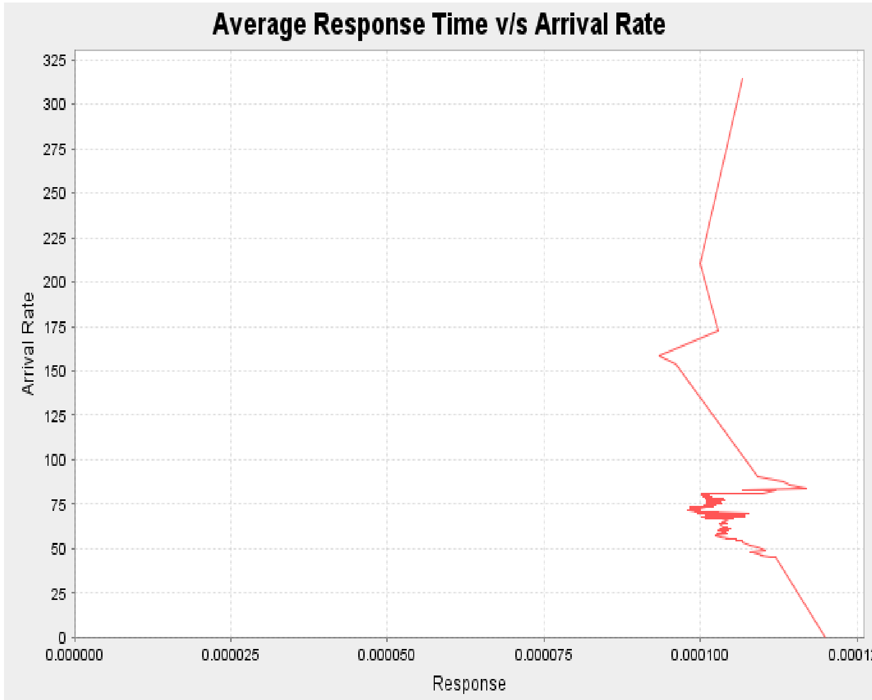

Figure 4. Service Request CPU.

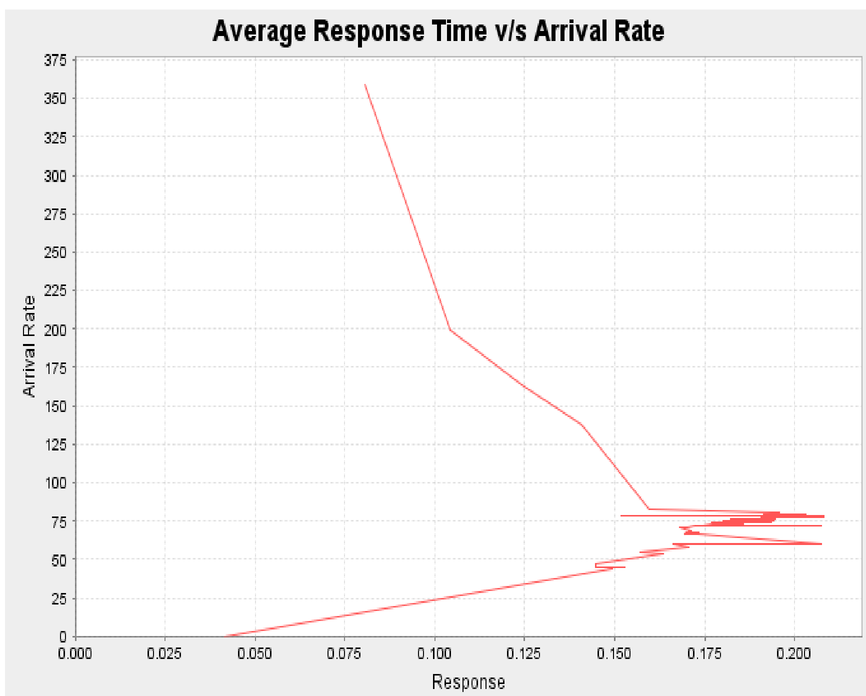

Figure 5. Internet 1.

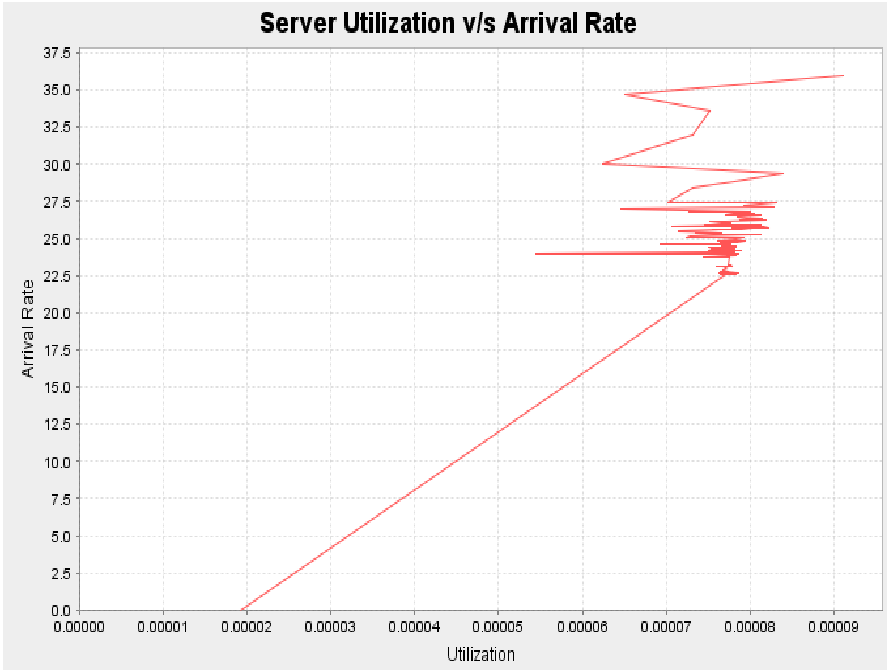

Figure 6. Service Registry CPU.

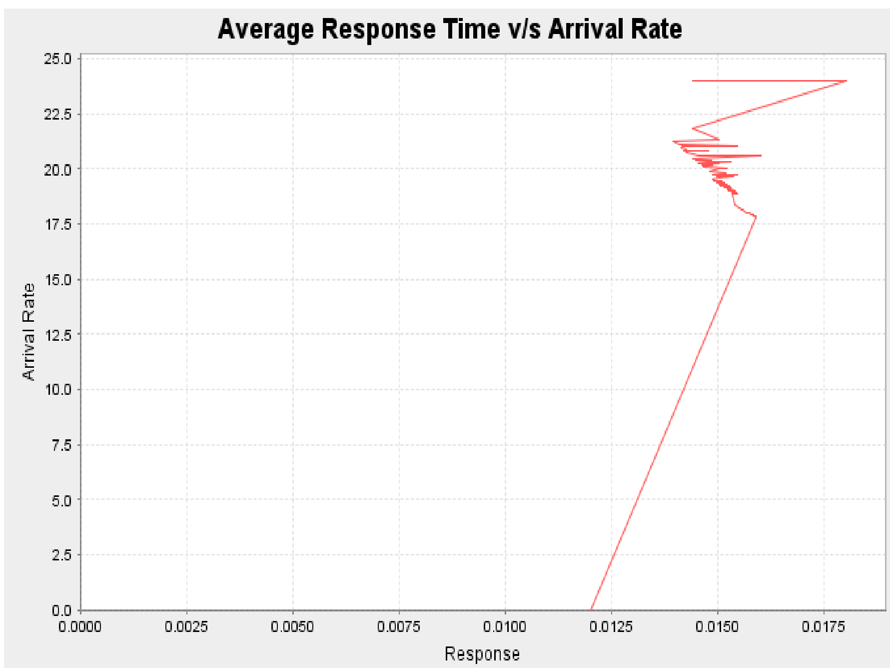

Figure 7. Service Registry Disk.

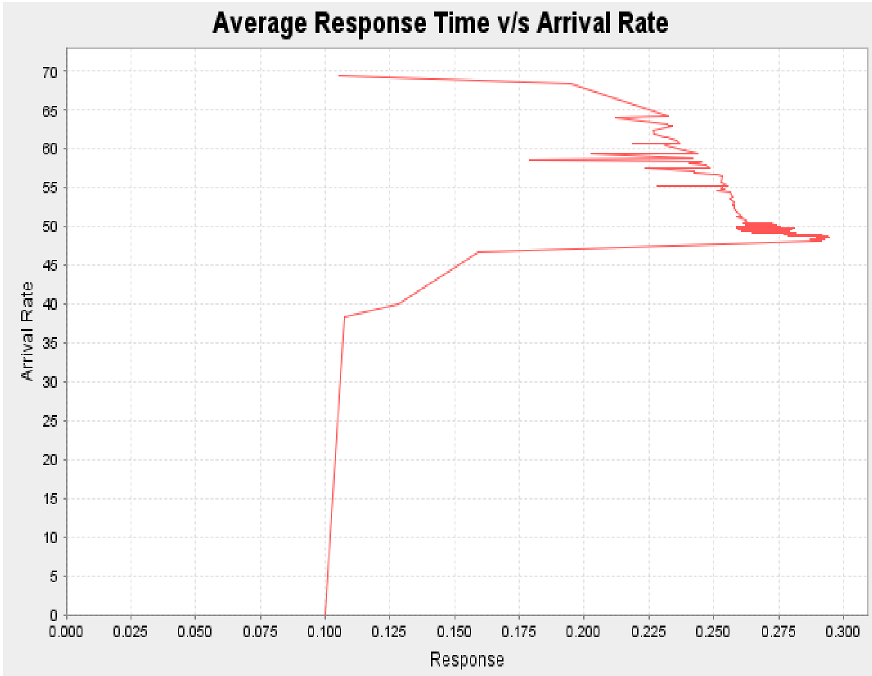

Figure 8. Internet 2.

From the graphs given in figure 4 to figure 8, we have observed the following:

In service request CPU, the response time is ranging between 0.00009 and 0.0002. The response time in internet is varying rigorously between 0.05 and 0.2. In service registry CPU, the response time is initially 0.00002 and gradually increasing up to 0.00008 and from that point onwards, it is varying between 0.00006 and 0.0009.

The response time in Internet 1 and Internet 2 is varying rigorously. In Internet 1, it is between 0.05 and 0.2 while in Internet 2, it is between 0.1 and 0.3. In service registry disk, it is varying between 0.0125 and 0.175.

6. Conclusion and future work

Due to the dynamic behavior of Web Services, predicting response time during early phases of SDLC becomes complex. Hence, in this paper, we have modeled Web Services using UML models, Use Case diagram and Sequence diagram. The model is simulated using the tool SMTQA and the performance metrics are obtained. The response time obtained for the hardware resources are analysed, bottleneck resources are identified. In future, we plan to develop methodologies for software performance prediction for Web Services.

References

[1] Booth D and Liu C.K, (2007) “Web Services Description Language (WSDL)”, Version 2.0 Part 0: Primer. W3C Recommendation, http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626.

[2] C.U.Smith and L.G. Williams, (2001) Performance Solutions, A practical Guide to Creating Responsive, Scalable Software, Addison-Wesley, Pearson Education.

[3] C.Kohlhoff and R.Steele, (2003) “Evaluating SOAP for High Performance Business Applications: Real-Time Trading Systems,” Proceedings of www2003, Budapest, Hungary.

[4] C. Gamble, (2007) “A Minimal Web Service Architectural Style,” Computing Science, Technical Report Series, No. CS-TR-1015.

[5] D. Garlan and B. Spitznagel, (1997) “Architecture-Based performance Analysis,” Proceedings of International Conference on Software Engineering and Knowledge Engineering, Sept. 1997.

[6] D. Garlan an d B. Schmerl, (2006) “Architecture driven Modeling and Analysis,” 11th Australian Workshop on Safety Related Programmable Systems.

[7] D. C. Burger and T. M. Austin, (1997), "The simple scalar tool set, version T solterootedpeedups can 2.0," Tech. Rep. CS-TR-1997-1342..

[8] D.E Geetha, T.V. Suresh Kumar, P. Mayank, K.Rajanikanth, (2010) “A Tool for Simulating Multi-tier Queuing Applications”, Technical Report, Department of MCA, MSRIT, TRMCA 04.

[9] Gustavo A, Casati F, Kuno H and Machiraju V, (2009) “Web Services: Concepts, Architecture and Applications”, Springer Verlag.

[10] J. J. Yi, L. Eeckhout, D. J. Lilja, B. Calder, L. K. John, and J. E. Smith, (2006) "The future of simulation: A field of dreams," Computer IEEE,, vol. 39, no. 11,pp. 22-29.

[11] Kreger, (2001) H. `Web Services Conceptual Architecture (WSCA 1.0), IBM, www.ibm.com/software/solutions/webservices/pdf/WSCA.pdf.

[12] L. Pustina, S. Schwarzer, M. Gerharz, P. Martini, and V. Deichmann, (2007) "Performance evaluation of a dvb-h enabled mobile device system model," in WOSP '07: Proceedings of the 6th International Workshop on Software and Performance. New York, NY, USA: ACM, pp. 164-171.

[13] L. Eeckhout, K. de Bosschere, and H. Neefs, (2000) "Performance analysis through synthetic trace generation," in ISPASS'00: Proceedings of the 2000 IEEE International Symposium on Performance Analysis of Systems and Software. Washington, DC, USA: IEEE Computer Society, pp. 1-6.

[14] M. Woodside, D. C. Petriu, D. B. Petriu, H. Shen, T. Israr, and J. Merseguer, (2005) "Performance by unified model analysis (puma)," in WOSP '05: Proceedings of the 5th international workshop on Software andper-formance. New York, NY, USA: ACM Press, pp. 1-12.

[15] Mitra N and Lafon Y, (2007) “SOAP Version 1.2 Part 0: Primer (Second Edition).”, W3C Recommendaion, http://www.w3.org/TR/2007/RECsoap12-part0-20070427/.

[16] OMG, (2003) “UML profile for schedulability, performance and time specification: Version l.0”, Object Management Group.

[17] Riegen C.V,(2002), ÜDDI Version 2.03 Data Structure Reference, OASIS Open 2002-2003, http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.htm.

[18] S. Punitha and Chitra Babu (2008) “Performance Prediction Model for Service Oriented Applications”, 10th IEEE International Conference on High Performance Computing and Communications, HPCC 2008, 25-27 Sept. 2008, Dalian, China.

[19] S. Balsamo and M. Marzallo, (2005), "Performance evaluation of uml system Architectures with multi call queueing network models” in WOSP, pp.37-42.

[20] Smith C.U., (1990) Performance Engineering of Software Systems, Reading, M.A, Addison-Wesley.

[21] T. Austin, E. Larson, and D. Ernst, (2002) “Simple scalar: An infrastructure for computer system modeling," Computer, vol. 35, no. 2, pp. 59-67.

[22] Web URL - http://www.w3.org/TR/2002/WD-ws-arch-20021114/