ISSN 2047-3338

QoS Based Dynamic Service Composition

Dr. E. Kirubakaran[1], D. Ravindran[2], Dr. D. I. George Amalarathinam[3]
[1]AGM, Outsourcing Department, BHEL, Tiruchirappalli, India
[2]Associate Professor of Computer Science, St Joseph's College, Tiruchirappalli, India
[3]Director, MCA, Jamal Mohamed College, Tiruchirappalli, India
[1]e_kiru@yahoo.com, [2]dr_indran@yahoo.com, [3]di_george@jmc.edu

***Abstract*– Web services are application components maintained by different service providers, always accessible, implementation independent and can be used if a situation needs. In a loosely coupled distributed system, web services play an important role. Higher level service, composite web service, may be constructed, not from the scratch, but by using the methods available in the primitive web services. When such composite web services are formed, for each service, many candidate services would be available. Among these candidate services, one service has to be selected and executed to accomplish composition. For such selection, QoS attributes may be used because all those candidate services would be similar with respect to functional attributes. In this paper, service selection based on the QoS factors viz response time, availability and dependency, is proposed. Using these factors, priority will be assigned to candidate services and the one with top priority will be selected. After the service invocation, the factors will be updated paving the way for a dynamic ordering of the candidate services.**

***Index Terms*– SOA, Composite Web Service, Quality of Service and Orchestration**

I. INTRODUCTION

IN a distributed computing environment, the objective will be the design, development and implementation of distributed system by using various technologies. The design starts with object oriented design aspects and proceeds with component concepts and Service Oriented Architecture (SOA). Interaction among the server-side applications can be achieved with a help of different techniques like socket programming, Remote Method Invocation (RMI), Common Object Request Broker Architecture (CORBA), and data interchange with eXtensible Markup Language (XML). Among these techniques, Component based development gives the system, being designed, the total independence. Components created can work independently without worrying about the other components. It leads to a loosely coupled system. An application can access a component, running in a different machine and created with a different language, with the help of the interfaces provided by that component.

A web service is an application component that is always available. A web service has some business logic, hidden inside and interfaces with which it exposes its features. It is implementation independent. It can be used anywhere irrespective of the operating systems and programming languages. SOA comes into the picture if such web services are used in the development process. SOA is an architecture in which various services, maintained by different service providers, are linked together to work as a system. These services are available always and can be accessed anytime, anywhere depending on the need. A system/application designed with SOA, can be realized with the help of web services. Web services are computational entities that are autonomous and heterogeneous [1].

A web service can have many methods. Few methods from various web services may be invoked in a particular sequence to get a higher level service, called composite web service (CWS). When such composite web services are created, for a method, many candidate methods would be available. Selection of a method among the candidate methods is always a problem. Here in this paper, a technique that uses the Quality of Service (QoS) parameters, is proposed. A unique value is evaluated, by using these QoS parameters and the weight, and used to assign priority to the candidate methods. Selection of candidate methods will be automatically done with this value. The priority of these methods is not static but it changes based on the factors like access time, availability and reliability.

Rest of the paper is organized as follows: Section 2 will discuss the related work whereas the Section 3 discusses the modeling of the web based flight information system. Section 4 discusses the implementation of QoS based composition of web service with reference to the flight system, as an example. Section 5 concludes by summarizing the work and opens up the area for further study.

II. RELATED WORK

Composite web services are constructed by using the primitive web services. The methods are invoked in a sequence to get a higher level service. It is termed as service orchestration with respect to the composite service perspective. *Orchestration* describes how web services can interact with each other at the message level, including the business logic and execution order of the interactions. These interactions may span applications and/or organizations, and result in a long lived, transactional, multi-step process model [1], [2].

There is a possibility of many similar methods, called candidate methods, available for selection, when a service is being invoked as part of constructing the composite web service. Web services which fulfill the functional requirement and meet or exceed the non-functional requirement are the candidates, according to Zhu Xilu, Wang Bai and Wei Gengyu [3]. They propose a QoS based web service discovery technique by assigning weights to different QoS criteria. According to Zhengdong Zhu et al [4], for this selection, functional and non-functional attributes viz reputation, price, response time, success rate and reliability, are considered [3], [4]. They considered only response time as the running reliability and success rate is almost 100%. Among the various services used as part of composite service, some of them are sensitive than others and if they fail, user's satisfaction would remarkably decline, according to Wang Ya-sha et al [5]. They considered the response time of *time critical section* for the satisfaction.

Zhi-Zhong Liu et al [6] proposed a new optimization algorithm, which uses selection probability calculation formula of each candidate service and this gets updated, after each iteration. Aijun Jiang et al [7] proposed a QoS-based tool framework that an abstract composite web service is modeled, transformed to concrete service and tested. If not satisfactory, the developer rectifies the parameters and performs the procedure recursively till satisfied. Ming Qiao et al [8] proposed a broker based architecture that enables dynamic QoS monitoring and adaptation for composite web services. Here a negotiation is initiated between the service requestor and the broker and if accepted, the requestor binds to the service. The contract is stored in the database for the future.

III. ARCHITECTURAL MODELING

A composite web service invokes the methods in other web services in a particular sequence. Many methods may be available, as candidate methods, for each method invoked. These candidate methods are assigned with some QoS factors. Based on these QoS Factors' values and the weight assigned to each of these factors, the candidate services are ordered. The method that tops the list will be selected automatically as it has the priority over other candidate methods. The model considered for implementation is depicted in Fig. 1.

In the model proposed in Fig. 1, there are three airlines companies viz Abhi Airlines, Joy Airlines and Nimmy Airlines, offering the airlines information. These three airlines companies' services are accessed by three different search engines viz search engine 1, search engine 2 and search

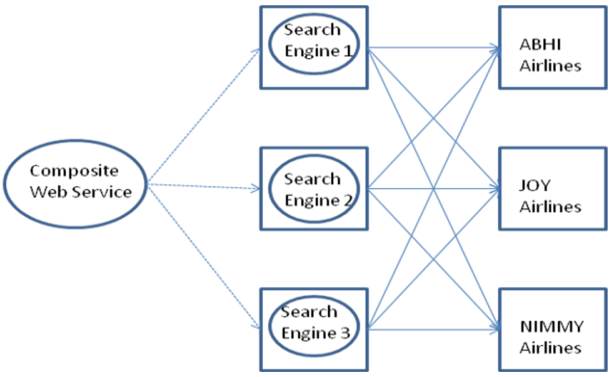

Fig. 1: Airlines information retrieval model

engine 3. These three search engines provide services for the clients, by accessing the information from three airlines companies. Three search engines compete with each other or they are the candidate services when a client wants to search for flight information. These three search engines will have QoS attributes viz the response time, availability and Dependency. Values are assigned to these attributes and these values are modified dynamically by a separate web service based on the situation at that moment. These attributes are also assigned with some weights. Using these values and the weights assigned, a unique value will be evaluated. This value is used to order the services, offered by the search engines. QoS parameter values are updated periodically, once each for the composite web service invocation.

IV. IMPLEMENTATION

The airlines systems have different services, which accesses the data from the database connected with them. These services are accessed by the search engines that provide multiple services to the clients. These services available with search engines are considered as candidate services. For the implementation purpose, each engine has a service that receives flight number and date and returns the seat availability information, for the reservation Composite Web Service (CWS).

These web services are implemented as a component with a help of Enterprise Java Bean (EJB) specifications. This is depicted in Fig. 2.

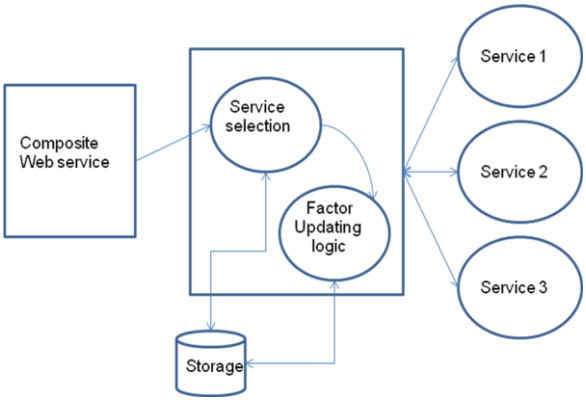

Fig. 2: CWS implementation

The CWS visualizes the methods with the components, as candidate methods and it selects any one method depending on the priority calculated with the help of QoS values attached with each factor. Response time (t_r), availability (a) and dependency (d) are the QoS factors considered in this implementation. Response time factor contains the average access time and whenever the service is invoked, this time (t_r) will be updated. Availability is the ratio of the successful access of the service to the total number of accesses made with the service. Dependency factor is the ratio of the number of access to this service to the total number of accesses made with any of these candidate services. These two values are always between zero and one. Both, availability and the dependency factors are positive metrics whereas the response time is a negative metric. When a service is not available for access, automatically the values '*a*' and '*d*' will go down. The weights attached with these factors, viz w_t, w_a, w_d may be assigned values within the logic that implements the selection of candidate services. Priority (P) for the candidate services is assigned by calculating a priority value as follows:

$$P = (\max(t_{r1}+t_{r2}+t_{r3}) - t_r)* w_t + a * w_a + d * w_d$$

where t_{r1}, t_{r2} and t_{r3} are the average access timings of the three services respectively.

The weights attached with these factors may be changed as and when the need arises, depending on the composite web service. Availability and Dependency factors are changed by suspending the services deliberately.

Fig. 3 shows the screen shot of the output given by a client program that implements the processing logic. This program invokes a candidate service with the higher priority and updates all the QoS parameters. If the higher priority service fails, next service in the order will be executed. Fig. 3 shows the change in priority after the 17th access of service3. Database contents used to implement the candidate service selection and the updation of QoS parameters, is shown in Fig. 4.

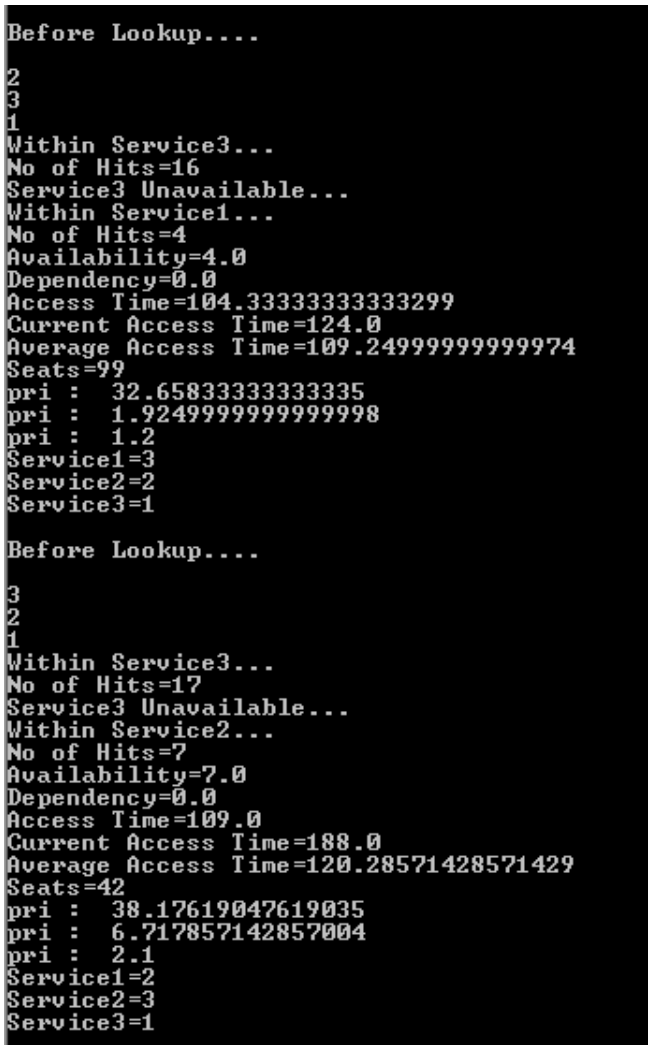

Fig. 3: Priority change

V. CONCLUSION AND FUTURE SCOPE

Among the candidate services, the best possible service has to be selected in the case of composite web services. The same can also be applied in the Universal Description, Discovery and Integration (UDDI) to select a suitable service for the client, based on the request criteria. Implementation of QoS factors that the best candidate service is selected. Candidate services are automatically reordered depending on the factors viz response time, availability and dependency. Along with these three factors, additional non-functional factors like throughput, security, cost, reliability etc may also be taken into consideration for better selection of services.

Depending on the environment, the services are assigned priority and ordered dynamically. Identification of best service will be done easily by taking the service that is in the first place in the priority order and the composite service methods execution flow can proceed without any hindrance. After the selection only, the dynamic updation of priority order is carried out and this will not have any impact on the service selection time.

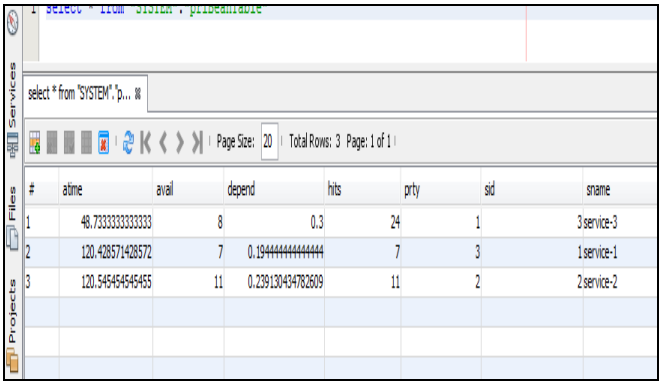

Fig. 4: Database contents

REFERENCES

[1] Nadia Busi, Roberto Gorrieri, Claudio Guidi, Roberto Lucchi and Gianluigi zavattaro, Towards a formal framework for choreography, In Proc. of International Workshop on Distributed and Mobile Collaboration (DMC 2005), IEEE Computer Society Press. WETICE 2005

[2] Chris Peltz, web services orchestration and choreography, Hewlett-Packard Company

[3] Zhu Xilu, Wang Bai, Wei Gengyu, QoS-aware web service discovery in P2P Network, IC-BNMT2009

[4] Zhengdong Zhu, Xuehan Dong, Yahong Hu, Zengzhi Li, Selection mechanism of Composite web services based on usability, International Conference on Computer Science and Software Engineering, 2008

[5] Wang Ya-Sha, Zhao Jun-feng, Li Ge, Using scenario oriented response-time management for composite web services, IEEE International Conference on web services, 2007

[6] Zhi-Zhong Liu, Zhi-Jian Wang, Xiao-Feng Zhou, Yuan-Sheng Lou, Ling Shang, A new algorithm for QoS-aware composite web services selection, 978-1-4244-5874-5/10/2010

[7] Aijun Jiang, Xiaoyong Mei, Shixian Li, Fudan Zheng, A QoS-based tool framework for developing composite web services, International Symposium on Information Science and Engineering, 2008

[8] Ming Qiao, Ferhat Khendek, Adel Serhani, Rachida Dssouli, Roch Glitho, Automatic QoS Adaptation for composite web services, 978-1-4244-3397-1/08