ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 2, May 2014

International Conference On Advances in Computer & Communication Engineering (ACCE - 2014)
on 21st & 22nd April 2014, Organized by
Department of CSE & ISE, Vemana Institute of Technology, Bengaluru, India

Web Service Composition Selection Satisfying Multiple QoS

Shree Krupa K, ShwethaPriya A,Rachitha

Student of Vemana I.T, Deptarment of CSE, Bangalore,India

Student of Vemana I.T, Deptarment of CSE, Bangalore,India

Vemana I.T, Deptarment of CSE, Bangalore,India

ABSTRACT*:* Web service composition plays a vital role for selecting an effective web service. Satisfying Non-functional requirements, along with the functional requirements of the user is difficult to achieve in the exiting web service composition. To satisfy this complexity, this paper proposes Web service composition selection satisfying multiple QoS to satisfy multiple QoS parameters with primary goal being to satisfy the functional requirements of the user. According to the type of web service composition, the QoS of web service composition is computed and the best web service composition is selected which satisfies the functional requirements of the user. An application example is used to explain this approach effectively.

KEYWORDS: Functional requirements, Non-functional requirements, Web Service Discovery System, web service, multiple QoS, QoS Normalization.

I. INTRODUCTION

All the globalised web applications have enhanced the use of web services, where users can not only seek and use a flexible web application but also publish web services based on various technology criterions such as SOAP, UDDI and WSDL [1]. The necessity to achieve a predetermined goal that cannot be realized by a standalone service gave origin to Web service composition [2]. From a user's perspective, composite services, or more objectively, the set of services should seem like a single service. Internally in a composition, services can interact with each other to exchange parameters. So the necessity to compose the web services to provide much stronger function gave rise to composite web service. Web services composition is a special composite Web service which is logically composed by Web services. Web services invoke each other and generate composite service for offering a service function for users. The composite service performance can be evaluated with two aspects. One is functional evaluation, where the service providing functions should be completely matched with users' request. The other is non-functional evaluation, where the Quality of service is satisfied. Now, the research aiming at web service composition mainly considers how to satisfy the functional requirement of web service composition. Satisfyingthe non-functional requirement of web service composition has become an important problem in web service composition.

With the explosive growth of the number of services published over the Internet, it is difficult to select satisfactory web services among the candidate web services which provide similar functionalities. Quality of Service (QoS) is considered as the most important non-functional criterion for service selection. Our paper proposes service selection based on QoS requirements or how to increase the efficiency for service composition and selection with multiple QoS attributes. Quality of Service (QoS) is a combination of several qualities or properties of a service, such as: availability, security, response time and throughput. An application example is provided to illustrate publishing of the services, searching the services in data base by keyword and QOS and normalize the services based on the quality parameters.

II. RELATED WORK

Previous approaches aim at determining the optimal service composition using brute-force-like algorithms (i.e., the execution time and costs are exponential even if simplifications are used), new approaches are satisfied with finding a nearly-optimal solution. These approaches are using (meta-) heuristics. (Meta-) heuristics are general search methods that exclude a huge number of solutions, because they do not consider optimal solution in their population. Hence (meta-) heuristics are more efficient than exact algorithms. The main disadvantage however is that they do not find the optimal solution in most cases, because, the exclusion of solutions is based assumptions, thus there is no guarantee, that the excluded solutions do not contain the optimal one. All existing approaches can be divided into exact approaches finding the optimal service composition and (meta-) heuristics selecting a nearly service composition. Instead of exact algorithms, (meta-) heuristics seem to be the better choice for QoS-aware service selection and composition.

In recent years, some approaches use the power of Data mining algorithms in knowledge extraction and pattern discovery among the huge amounts of data in web service selection. Wu et al. present a Bayesian network based QoS assessment model for web services. That could predict the service capability in various combinations of users QoS requirements. This approach is used to evaluate the capability of each service, and the one with best capability is selected as the binding service. Though it uses Bayesian network classification algorithm for each provider/service to predict the level of QoS, it is computationally complex and is based on probabilities, moreover it just considers local constraints in web service selection and doesn't mention the global constraints. Ben Mabrouk et al. present a heuristic approach for service composition in dynamic environments. This solution uses the K-means algorithm to classify the web services to QoS levels, then it uses the result of this clustering in an utility function in order to rank web service candidates for each task as a local selection part, then it uses a search tree to select the best services to form the composition plans in an ordered way. The proposed solution is computationally expensive in both of the clustering algorithm and the structure of the search tree in a composition plans with the high number of activities, also it suffers from the deficiency of the clustering techniques, because clustering is an example of unsupervised learning. Furthermore it does not mention the semantic matching between output and input of the services.

III. MULTIPLE QOS AND NORMALIZATION

A. Multiple QoS

QoS describes the quality information in some aspects of web service and is very important for combining web services and constructing the web application which is according with the requirement of users [3].We propose Web Service Discovery System to search the services, based on the operations available in the services; this will help to get the appropriate services in the registry and from the access point of the WSDL.Search by QOS in the Registry, which includes the performance parameters which characterize the interaction with the Web service and the following 5 features:
Response Time: time elapsed from the submission of a request to the time the response is received.
Accessibility: represents the degree that a Web service is able to serve a request.

Compliance: represents the extent to which a WSDL document follows WSDL Specification

Success ability: represents the number of request messages that have been responded.

Availability: represents the percentage of time that a service is operating.

B. QoS Normalization and Ranking

The purpose of the QOS Normalization is to select the best service(s) or their compositions to fulfill users' requirements. Different service providers may compete to offer similar services. The concept of Quality of Web Service (QoWS) is emerging as a key feature in distinguishing between competing services. QoWS reflects the runtime and business requirements of web services, such as response time, availability, reliability, cost, and reputation. QoWS can be calculated by the following score function:

$$F = \sum_{Q_i \in Neg} W_i \frac{Q_i^{max} - Q_i}{Q_i^{max} - Q_i^{min}} + \sum_{Q_i \in Pos} W_i \frac{Q_i - Q_i^{min}}{Q_i^{max} - Q_i^{min}} (1)$$

IV. Implementation

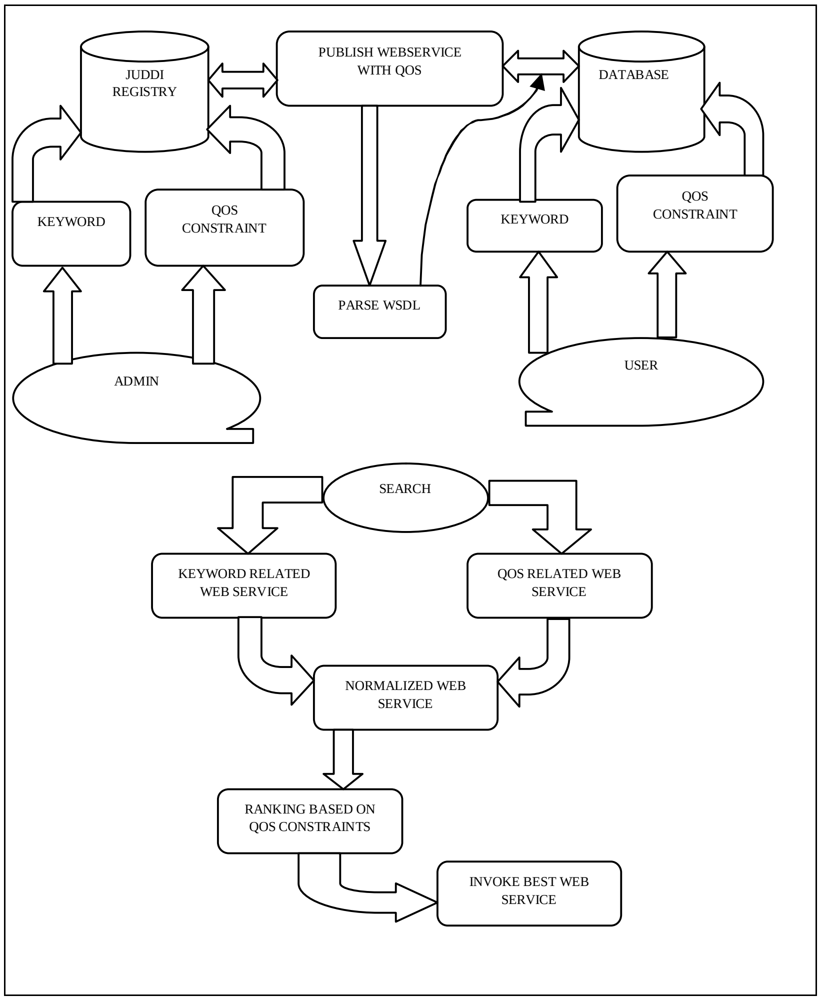

Figure 1: System Architecture

The implementation consists of developing client applications for an embed device, as well as server applications to centralize data and make it permanently accessible to the family caregivers and rescue team. The development is iterative

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 2, May 2014

International Conference On Advances in Computer & Communication Engineering (ACCE - 2014)
on 21st & 22nd April 2014, Organized by
Department of CSE & ISE, Vemana Institute of Technology, Bengaluru, India

and scenario-based and also describes architecture and details are given in the chapter III. In order to reduce the production costs and to facilitate technology transfer, we choose an open source approach. The source produced is under CECCIL-C license. This license similar to the Lesser-GPL allows the source as component in proprietary software. Then, the share of production cost of these components is possible between industrial companies.

The System Architecture is given in figure 1 where the functional and non-functional web services are registered. To publish the web service in JUDDI, we have to supply the service name, service description, access point WSDL, QOS Parameters (success ability, response time, throughput, reliability, availability, documentation etc). Authentication is required to publish the service to register the service with the help of security API in JUDDI. This will give the authentication token to register the service to publish the service and the published API in JUDDI used to generate the service key and business key.

The UDDI Registry implements the UDDI specification. UDDI is a Web-based distributed directory that enables businesses to list themselves on the Internet and discover each other. The UDDI registry is both a white pages businessdirectory and a technical specifications library. The Registry is designed to store information about Businesses and Services and it holds references to detailed documentation.

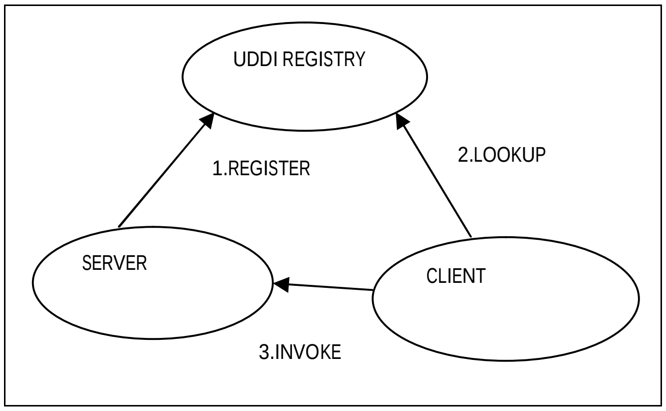

Figure 2: Invocation Pattern using the UDDI Registry

The invocation pattern using the UDDI Registry is shown in figure 2. A business publishes services to the UDDI registry. A client looks up the service in the registry and receives service binding information. The client then uses the binding information to invoke the service. The UDDI APIs are SOAP based for interoperability reasons. The UDDI v3 specification defines 9 APIs:

UDDI_Security_PortType defines the API to obtain a security token. With a valid security token a publisher can publish to the registry. A security token can be used for the entire session.

UDDI_Publication_PortType defines the API to publish business and service information to the UDDI registry.

UDDI_Inquiry_PortType defines the API to query the UDDI registry. Typically this API does not require a security token.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 2, May 2014

International Conference On Advances in Computer & Communication Engineering (ACCE - 2014)
on 21st & 22nd April 2014, Organized by
Department of CSE & ISE, Vemana Institute of Technology, Bengaluru, India

UDDI_CustodyTransfer_PortType, this API can be used to transfer the custody of a business from one UDDI node to another.

UDDI_Subscription_PortType defines the API to register for updates on a particular business of service.

UDDI_SubscriptionListener_PortType, defines the API a client must implement to receive subscription notifications from a UDDI node.

UDDI_Replication_PortType defines the API to replicate registry data between UDDI nodes.

UDDI_ValueSetValidation_PortType, by nodes to allow external providers of value set validation Web services to assess whether keyedReferences or keyedReferenceGroups are valid.

UDDI_ValueSetCaching_PortType, UDDI nodes may perform validation of publisher references themselves using the cached values obtained from such a Web service.

The service is searched with the help of anyone QOS parameters. It will give the set of services which are mapped with QOS parameters in the repository. Invoking the web service is dynamic, based on the result when searching the service. Invoking the best service based on the user input of the service. Dynamic Invocation Interface will help to invoke the web service dynamically.

Since a composition plan typically consists of multiple operations, we first present a set of aggregation functions to compute the QoWS for a composition plan. They combine the QoWS parameters from multiple service operations. The system presents two approaches for finding the best plan: exhaustive search and greedy search. The exhaustive search enumerates the entire space of composition plans. The greedy search achieves the polynomial complexity by using a divide-and-conquer strategy. It generates an optimal sub plan from each service through local search.

$$\begin{aligned}
\text{Latency (plan)} &= \textstyle\sum_{i=1}^{n} \log\left(lat(op_i)\right)\\
\text{Reliability (plan)} &= \textstyle\prod_{i=1}^{n} log\left(rel(op_i)\right)\\
\text{Availability (plan)} &= \textstyle\sum_{i=1}^{n} \log\left(avail(op_i)\right)
\end{aligned}$$

In Web Service Selection, Score function (Equation 1: Score Function) follows the following steps to find QoWS,

1. Normalization

2. Weighting

3. Sum

The above steps can be depicted using the following illustration. The matrix 1 of the illustration contains the QoS attributes. This QoS attributes will be normalized using the score function. The matrix 2 of the illustration is called as normalized matrix. In step 3 normalized matrixes will be assigned with weight based on the importance of the attributes.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization) Vol.2, Special Issue 2, May 2014

International Conference On Advances in Computer & Communication Engineering (ACCE - 2014)
on 21st & 22nd April 2014, Organized by
Department of CSE & ISE, Vemana Institute of Technology, Bengaluru, India

V. EXPERIMENT RESULTS

In our experiment we focus on the execution time of our method. This metric measures the response time of our algorithm with respect to the size of the problem, in terms of the number of activities and the number of services per activity. In these experiments, we measure the execution time of the ranking phase.

For the quantity allocation of the service qualitative parameters we use the QWS real dataset. This dataset includes measurements of 9 QoS attributes for 2500 real web services. The dataset was measured using commercial benchmark tools for web services, which were located using public sources on the Web, including UDDI registries, search engines and service portals. We use these metrics (i.e., response time, throughput, availability, validation accuracy, reliability) as a sample input data for our algorithm.

To accomplish the classification, we used the CBA 1 tool for implementing the classification algorithm. This tool has a graphical user interface and is designed particularly for implementing the CBA algorithm. Hence for this part we do not measure the response time.

For classifying the candidate services we have defined 3 different quality levels. These levels are specified to the distance of the QoS attributes to the user demands. After the classification carried out, the classes coefficient were defined as follow: 1for services in the first class, %N for services in the second class and # N for services in the third class.

The execution time of ranking phase for the calculated near-optimal composite service is shown in figure 3. These measurements are obtained by fixing the number of QoS constraints to 4 and varying the number of activities and the number of service candidates per activity between 10 and 50. The obtained measurements show that the execution time of our algorithm increases along with the number of activities and the number of services per activity, which is an expected result.

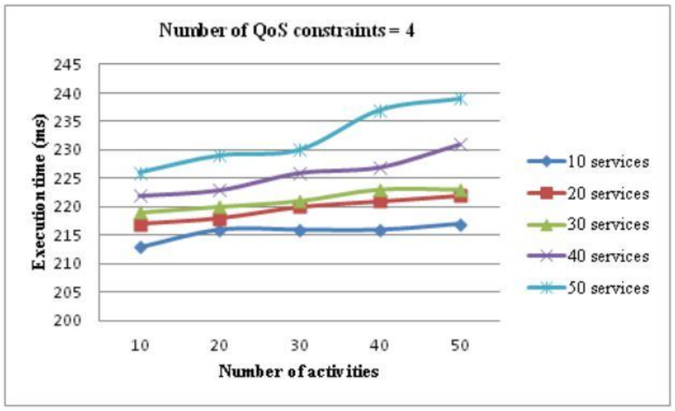

Figure 3: Execution time of the ranking phase for near-optimal composite service

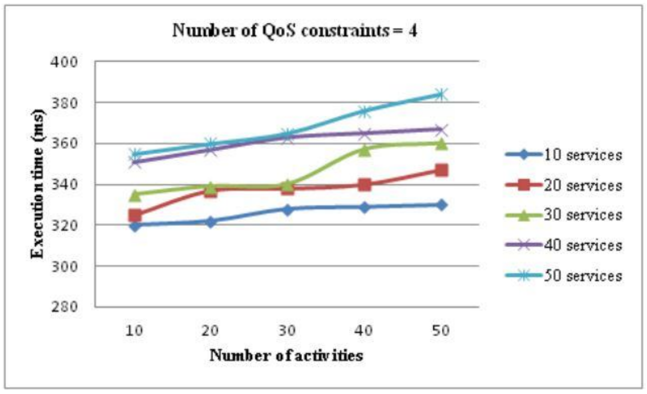

Figure 4: Execution time of the ranking phase for first alternative composite service

Also figure 4 demonstrates the execution time of ranking phase for the first alternative composite service. The increment of the execution time compared with the figure 3 is due to the increment of the number of tasks and candidate services so the spending time for calculating semantic similarity of output-input concepts increases subsequently. DL reasoning is the most time consuming process in large-scale problem of quality-driven semantic web service composition (i.e., number of tasks and candidate services greater than 100 and 350).This is caused by the large number of potential semantic links between tasks. As the figure 3 and 4 show, our method has a good execution time compared with the evolutionary algorithms which the later algorithms (e.g., genetic algorithms, PSO) take a long time to execute composite service (e.g., for a number of activities more than 25, it takes 2847 and 2236 ms, respectively for the PSO and GA algorithms). Moreover unlike our proposed method which aims to produce the composite services in an ordered way, the evolutionary algorithms produce the composite services in an unordered way and there is no evidence that the first composite service which produced by these algorithms be the optimal one, in other words the next produced composite services may be better than the first one.

VI.CONCLUSION

We address the web service selection problem by defining constraints within a quality model to balance QoS metric with functional quality. The functional quality evaluates the quality of semantic links between the semantic description of output-input parameters of web services, while QoS focuses on the non-functional criterion to retrieve the satisfactory services regard to the users' requirements and preferences. Our proposed approach has four advantages: First, it uses the classification data mining algorithm to specify the most eligible services respect to the user demand. Applying data mining algorithms to this field brings new ideas. Second, by producing alternative composite services satisfying QoS constraints, our algorithm could cope with changing conditions in dynamic service environments. Third it shows a satisfying capability in terms of execution time, which it is an important point in dynamic service environments and finally with applying the semantic similarity between services by semantic links it increases the accuracy of selection .Our proposed method makes part of our ongoing research by using strong data mining algorithm in order to decrease the execution time and improve the optimality of our heuristic algorithm, and besides the local optimization, by combining the global constraint of the user to the local constraints, considering the QoS constraints through the whole composition to ensure meeting global QoS constraints.

REFERENCES

[1] LI Yan, ZHOU Ming-Hui, LI Rui-Chao, CAO Dong-Gang, and MEI Hong, Service Selection Approach Considering the Trustworthiness of QoS Data, Journal of Software, vol.19,no.10, pp.2620-2627, October 2008.

[2] WANG Shang-Guang. SUN Qi-Bo and YANG Fang-Chun, Web Service Dynamic Selection by the Decomposition of Global QoS Constraints, Journal of Software, vol.22, no.7,pp.1426-1439, July 2011

[3] XIAO Fang-Xiong, HUANG Zhi-Qiu, CAO Zi-Ning, TU Li-Zhong, and ZHU Yi, Unified Formal Modeling and Analyzingboth Functionality and QoS of Web Services Composition, Journal of Software, vol.22, no.11, pp.2698-2715, 2011.