International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 01 | Jan-2016 www.irjet.net p-ISSN: 2395-0072

QoS Aware Redundant Free Web Services Composition

S.Manimala[1], S.Selvapriya[2]

[1] *Assistant Professor, Information Technology, Jerusalem College of Engineering, TamilNadu, India*
[2] *Assistant Professor, Information Technology, Jerusalem College of Engineering, TamilNadu, India*

Abstract - ***Recently, there has been growing interest in developing web services composition search systems, but with the drawback of including redundant web services in the results. Hence, to remove the redundancy, Non Redundant web services Composition was developed in web services composition. The candidate compositions are found and decomposed into several non-redundant web services compositions by using the concept of tokens. In our proposed system, the NRC(Non Redundant Composition) system is improved with OWL-S for representing ontology. Matchmaking accuracy will be improved by using OWL-S. And non-redundant web services composition will be combined with the capability of QoS parameters (execution time, availability).***

Key Words: *Web services composition, QoS aware web service, Ontology*

1. INTRODUCTION

In the early stage of the Internet, most web technologies were designed to allow human-to-machine interoperability. With the rapid development of network technology and the growth of the Internet, web technologies are evolving to support machine-to-machine interoperability. A web service is defined by the W3C as a software system designed to support interoperable machine-to-machine interaction over a network and is described as a WSDL (Web Service Description Language) that includes the location of the web service and interface information such as input/output parameters.

Web services architecture has three components -: (a) service providers for publishing web services, (b) service requesters for querying and binding web services and (c) UDDI (Universal Description, Discovery, and Integration) [3] business registries for storing web services information.

The emergence of technology and standards for web services has triggered interest in techniques for searching web services. These interests can be divided into two categories: discovery and composition. Several approaches in web services discovery have been proposed for storing and searching the target web services efficiently. The focus of these approaches is on finding a single web service. Web services composition involves the combination of a number of web services to produce a more complex and useful service.

It can be useful when we are looking for a web service with specific inputs and outputs and there is no single web service satisfying the request.

Consequently, research on the finding web services composition efficiently has been conducted. Web services composition can be classified into four main categories: (1) composition schemes that consider only input/output parameters of web services, (2) web services composition schemes that consider input/output parameters, pre-conditions, and effects of web services, (3) semantic composition schemes that consider input/output parameters, pre-conditions, and effects of web services, and a domain ontology and (4) QoS-aware composition schemes.

A Non-redundant web services composition with ontological support is proposed in this paper. We have proposed the non-redundant web services composition along with user's QoS parameters in this paper. We also provide an ontological support to the web service objects.

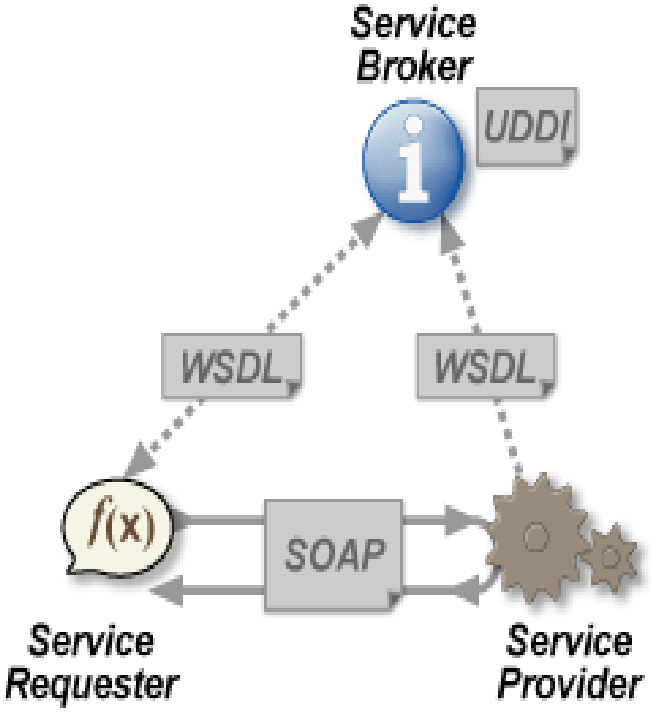

Fig -1: Web services architecture

2. PREVIOUS RELATED WORK

Previous research work on web services composition focused on non redundant composition. Two phase algorithm are by preceding approaches to identify the redundant web services and remove the redundancy with the help of a token manager [1]. These studies showed the usage of web services composition and non redundant service as well. . A Limitation of web services composition is the lack of external semantics. Few studies depicts that, presently, the service discovery is the process of discovering one or more documents that describe in a particular service. Service discovery mechanisms typically perform syntactic matching and it support only key word based search. Only the poor service discovery results produced by Service discovery mechanism. Another drawback of the existing service discovery mechanism is that the query service matching score is calculated and taking an account only the keywords from the user query's and the terms is the service descriptions

A number of studies have focused on QoS-aware composition approaches. These studies adopted QoS-based models or mechanisms for selecting the best web services composition available. Few studies proposed an optimizing service selection method for composite services; the method was based on a generic QoS model and involved the use of established linear programming techniques. In an extension of this work, they presented a QoS-aware middleware platform, called AgFlow, supporting quality-driven web services composition. The AgFlow system consists of a service quality model and two alternative service selection approaches. Some studies proposed an approach to trigger and perform composite service re planning during execution because the actual QoS values may deviate from the estimation.

From the above studies, the need for semantic web services is proposed and is used to provide a semantic matching and Qos.

3. SYSTEM ARCHITECTURE

The system framework consists of three major phases

1. Finding web services composition 2. Decomposition of web services 3. Domain Ontology construction.

The input of the system is the user's query. The web services are created and deployed in the registry which runs within the application server. The registry returns the web services that are needed to satisfy the user's request. The web services that take the input parameter same as that of the user's query are taken and hence we have found a composition. Then the web services with the input same as that of previous web services composition are taken as next composition. This step is repeated till we get the web service composition output similar to that of the user query output. The composition is then decomposed with the help of token manager to remove redundancy. The task of the token manager is to create, assign and reassign tokens. The token manager maintains a list of uniquely covered parameters.

For explicit semantics the objects of web services should be mapped to the ontology class. By having a domain ontology that defines ontology classes and keywords we can compose two web services having different parameters.

4. METHODOLOY

The necessary web services are created and deployed in the registry. The user query is passed to the registry and appropriate web services are returned. The concept of ontology is applied to the web services to improve the matchmaking accuracy. The description of the identified modules is given below.

4.1 Web Services and Deployment

The necessary web services are created by specifying the name of the web service, the input parameters, output parameters and deployed in the application server's registry.

4.2 Finding Web Services Composition

The web services composition is found by taking the web services whose input is same as that of the user input. Hence we have found a web service composition. The next step is to find the web services whose input is similar to that of the previous composition's output. By repeating the above steps we can find the web services composition. Below is the algorithm:

Algorithm:

Input: Set of web services and user query input
Output: Web service compositions found
Begin

1. For each web service in the set
2. Find the web services that take input same as that of user's input /* Found a composition*/
3. End For
4. Do
5. Find those web service that take the input similar to that of previous composition's output
6. Until web service output is similar to query output

End

4.3 Web Services Composition

In this module the web services are decomposed with the help of token manager, which has the details of uniquely covered parameter of each web service. The task of the

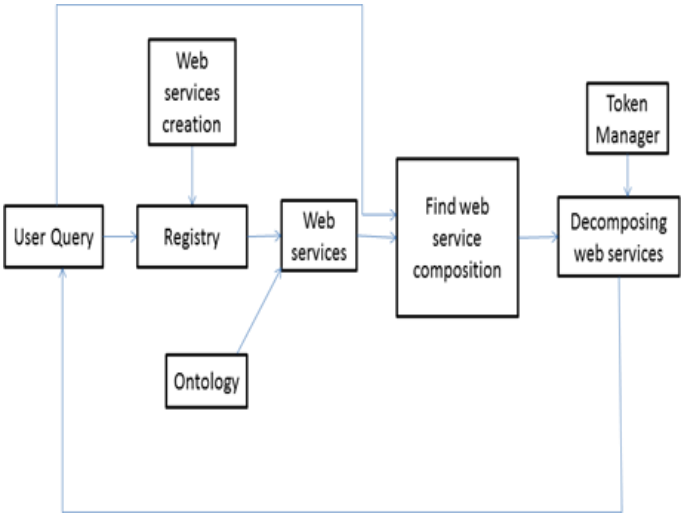

Fig -2: Web services architecture

token manager is to create a token, assign, unassigned and remove a token. Below is the algorithm for token manager to assign a token.

Algorithm:

Input: Composed web services
Output: Non redundant web services composition.
Begin

1. For each web service parameter requesting token from token manager
2. Token manager checks the parameter
3. If token is null
4. Assign a token
5. Else
6. Check the web service for number of parameters and decide which web service is redundant and remove it
7. End if
8. End For

End

4.4 Ontology Construction

The ontology classes are mapped to the web service objects. By having a domain ontology that defines ontology classes and keywords we can compose two web services having different parameters.

5. EXPERIMENTAL RESULTS

The web services are created and deployed in the application server. As soon the user enters the input information the necessary web services are called. And at the background the web services compositions are found and decomposed to remove redundancy in the composition and also to choose the best composition based on user QoS parameters. The web service objects are mapped to domain ontology classes.

This ontology based web services composition helps to improve the matchmaking accuracy and also remove redundant web services based on the domain ontology.

Best Composition

WebService Name	Execution Time
ws8	178
ws9	180
ws11	82
ws13	769

Fig -3: Best Composition

Before Composition

WebService Name	Input Parameters	Output Parameters
ws1	AUTHOR, BOOKNAME	ISDN, COST
ws9	CCN, PRICE	ACCHOLDER
ws9	CCN, PRICE	ACCHOLDER
ws3	SSN	USERID
ws4	SSN	USERID
ws5	AUTHORISED,ISDN,USERID	ORDERID
ws6	AUTHORISED,ISDN,USERID,COST	ORDERID
ws7	COST	ORDERID

Fig -4: Before Composition

The ontology based non redundant web service composition is shown in figure 3.

6. GRAPHICAL ANALYSIS

The Execution time and Availability for the web services are identified and graphs are plotted.

Chart -1: Graph for Execution time

The chart1 shows the execution time of the web services against the order placement. From the graph it is clear that the domain ontology to the web services has reduced the execution time when compared to a non redundant composition system.

7. CONCLUSIONS

In this Paper, we presented an ontology based web services composition along with some the QoS parameters availability and execution time. The web services are created and deployed into the registry. The web services composition is found and decomposed into several sets to remove redundant service. The web service objects are mapped to the domain ontoloy classes. Hence, this system improves the matchmaking accuracy and also reduces the execution time of the system. From chart1 it is clear that the Improved NRC (Non Redundant Composition) system hihly reduces the execution time when compared to the NRC system.

REFERENCES

[1] Joonho Kwon, Daewook Lee , *Non-redundant web services composition based on a two phase algorithm*, Data & Knowledge Engineering Vol:71 (2012) pp:69–91

[2] M. Champion, C. Ferris, E. Newcomer, D. Orchard, *WebServicesArchitecture*,http://www.w3.org/TR/2002/WD-ws-arch-20021114/ Nov. 2002.

[3] L. Clement, A. Hately, C. von Riegen, T. Rogers, UDDI Version 3.0.2, http://uddi.org/pubs/uddi_v3.htm Oct. 2004. W. Treese, *Ten years on internet time* netWorker, vol. 10, no. 3, pp.15 – 17, 2006

[4] Rajesh Karunamurthy, Ferhat Khendek, Roch H. Glitho, *A novel architecture for Web service composition*, Journal of Network and Computer Applications Vol:35 (2012) pp:787–802

[5] Anjan Bandyopadhyay, Tapas Si, Partha Protim Mondal, Saurav Mallik, *Proposed Conceptual Model for Semantically Enabled Web Services Based On QoS* , Procedia Technology Vol:4 (2012) pp:579 – 583

[6] Daewook Lee, Joonho Kwon, Sangjun Lee, Seog Park, Bonghee Hong *Scalable and efficient web services composition based on a relational database* , The Journal of Systems and Software Vol:84 (2012) pp:2139-2155

[7] R. Berbner, M. Spahn, N. Repp, O. Heckmann, R. Steinmetz, *Heuristics for Qos-aware web service composition*, in: Proceedings of the 4th International Conference on Web Services, Chicago, IL, USA, 2006, pp. 72–82.

[8] F. Lécué, N. Mehandjiev, *Towards scalability of quality driven semantic web service composition*, in: Proceedings of the 7th International Conference on WebServices, Los Angeles, CA, USA, 2009, pp. 469–476.

[9] W. Jiang, C. Zhang, Z. Huang, M. Chen, S. Hu, Z. Liu, Qsynth: *A tool for Qos-aware automatic service composition*, in: Proceedings of the 8th International Conference on Web Services, Miami, Florida, USA, 2010, pp. 42–49.

[10] Y.-J. Lee, C.-S. Kim, *A learning ontology method for restful semantic web services*, in: Proceedings of the 9th International Conference on Web Services, Washington DC, USA, 2011.

[11] S. Kona, A. Bansal, G. Gupta, *Automatic composition of semantic web services*, in: Proceedings of the 5th International Conference on Web Services, Salt Lake City, Utah, USA, 2007, pp. 150–158.

[12] J. Ko, C. Kim, I. Kwon, *Quality-of-service oriented web service composition algorithm and planning architecture*, Journal of Systems and Software 81 (11)(2008) 2079–2090