ADAPTIVE WEB SERVICE SELECTION BASED ON DATA TYPE MATCHING FOR DYNAMIC WEB SERVICE COMPOSITION

R Kanesaraj Ramasamy[1], Fang-Fang Chua[2], Su-Cheng Haw[3] and Chin-Kuan Ho[4]

[1]*Multimedia University Cyberjaya, Malaysia, kanes87@gmail.com*
[2]*Multimedia University Cyberjaya, Malaysia, ffchua@mmu.edu.my*
[3]*Multimedia University Cyberjaya, Malaysia, sucheng@mmu.edu.my*
[4]*Multimedia University Cyberjaya, Malaysia, ckho@mmu.edu.my*

ABSTRACT. Although there are many web services provided for access in World Wide Web (WWW), some services are not available at all times. It is very important to ensure all services are available when a service composition takes place. A web service that meets the requirements of the workflow but does not match the data type will still cause a failure in composition. To address this concern, we propose an adaptive web service selection method which is able to replace a current web service which has been used for composition but fails during execution time. The proposed algorithm will select the most appropriate web service based on web service discovery engine recommendation and match the requirement based on WSDL description. Upon matching the requirements of the workflow, the selected web service will be matched according to the input and output data type. The goal of this paper is to ensure every web service that meets the requirements of the workflow does not get rejected when the data type does not fulfill the matching criteria.

Keywords: web service selection, data type matching, dynamic adapter

INTRODUCTION

Web Service has become popular because it can be reused and is able to reduce the effort of development for the developer. Recently, the use of web service became more significant when the implementation of web service composition was introduced. According to Lin et al. (2011) web service composition is the process of integrating existing web services and can be referred as a prospective method to build an application system. Web services are considered as self-contained, self-descriptive and modular applications that can be located, published, and appealed in the context of the web. In the present day, more companies and corporations carry out their business implementations through the WWW. Therefore, as mentioned by Lin et al. (2011) it is critical that the efficient and effective selection and integration of heterogeneous and inter-organizational services on the internet at runtime is possible, because it serves as a very crucial step towards the development of web service applications.

In conducting web service composition, there are two major options one can utilize, namely: the dynamic and self-adaptive methods. Alrifai et al. (2010) mention that dynamic or automated service composition requires handling of the discovery and services selection automatically. Self-adaptive methods use software that evaluates its own behavior and adjusts this

behavior when the evaluation indicates that what is intended to be accomplished is not being accomplished or when there is a possibility of better performance or functionality. Kang. et al. (2010) conclude that to solve the contradiction between self-adaptive efficiency and optimization effect existing in the execution of current self-adaptive web service composition, it is crucial to note that self-adaptability and autonomic behaviors are becoming more and more crucial in the complex distribution system's landscape.

This research paper is the continuation of previous work on Web Service Discovery Engine (WSDE). WSDE is developed to search web services from web service repositories and rank the searching result based on user defined QoS attributes. The whole research is divided into three different phases. First phase covers the WSDE, second phase is the selection of web service and finally the composition of web service and development of cloud-based mobile application. As for this paper, we have analyzed the existing selection algorithm and present the idea of conducting web service selection dynamically based on WSDL description matching and also data type matching. We have introduced a new web service selection adapter which solves two main issues: (1) Availability of web service during composition. (2) Reducing the rejection of web service when the input or output data type matching fails. This is a common problem as web services developers tend to develop their own data type for their web service. The rest of this paper is organized as follows: In related works, we present and evaluate existing work to address web service selection problem using various methods. In the design section, we present the overall architecture of our selection phase and finally draw a conclusion at the end.

RELATED WORK

Nowadays, web service selection is widely employed in many industries such as e-commerce, medical and bio-informatics. Many researchers intensively studied and produced ample of work on various techniques and applied them for web service selection issues in the past few years. The methods also have been improved to solve selection issues. However, the research work on the optimal web service selection problem with constraints remains unsolved (Tang. et al. 2010).Generally, addressing quality of service in web service selection is always a challenge faced by researchers.

Zeng et al. (2004) proposes two service selection methods driven by QoS without considering the context. To address this issue, Gao, et al. (2011) present an approach to service selection by measuring the similarity between the web service publication and the service request. Their matchmaking approach is flexible and they present a method to compute the similarity score between two categorical values based on set theory and semantics. Since QoS changes dynamically with users' concern, atomic services of a composite web service should be replaced with better quality service. However, the performance of composite services will degrade if the replacement takes place frequently. Furthermore, so far there is no proper work on quantitative prediction of concern change (Li. et al. 2009). Thus, the best way to retain the performance of composite services is by predicting the change of QoS in the selection phase so that it can reduce the number of unrelated web services. A web service selection Goodness-Fit Selection (GFS) algorithm is proposed by using Structural Equation Modeling (SEM) to solve multivariable equation group of QoS attributes and user concerns on them, predict and finally verify their variables. This work can reduce the error in prediction and quantitatively predict the change of quality of service dynamically. This is helpful to users as they can select the optimal web service through the GFS algorithm based on the prediction result (Li, et al. 2009).

Denghui et al. (2014) presents an improved version of an ant colony (ACO) with Pareto for web service selection. They have proved that this algorithm has improved efficiency compared with the traditional ant colony algorithm. In this paper, the authors have used QoS attributes in their searching algorithm. They have reduced the searching area but did not perform a selection for composition as presented in our proposed work. Oussalah et al. (2013) introduces an adaptive service composition method using Interface Description information. Their work is only valid for a specific case and it will need to be prepared by the developer. Once a new set of cases is to be implemented then the developer will need to make changes again. Storing of operations is required before any compatibility process is done. This work has improved efficiency but it will fail once a new operation or data type is introduced by the web service provider. Similar to Li, et al. (2009), Bacciu (2010) also stresses users' concern on QoS in their paper. Thus, the author presents a method that allows for a fuzzy-valued description of QoS parameters. Due to multiple QoS preferences set by users such as response time and availability, they propose a matchmaking procedure based on fuzzy set theory (Bacciu, 2010). The comparison of analysis of algorithms is shown in Table 1.

Analysis of Web Service Selection Algorithms

Table 1. Analysis of Selection Algorithms

Related Works	Web Service Selection Algorithms			
	Data Type Matching	**Similarity Prediction**	**Ant Ontology Optimization**	**WSDL Interface Description-Based Matching**
Gao, H., et al, 2011	√	√		
Li, M., et all, 2009		√		
Denghui.W, et al, 2014			√	
Oussalah.Y, et al, 2013				√
Bacciu, D., 2010	√			
Proposed Method	√			√

Based on Table 1 and the previous discussion on selection methods, it is important to consider QoS as one of the factors for composition. In our proposed method, we have included QoS verification and ranking in the WSDE. For the Selection phase, we have proposed an approach using the WSDL structure to read the description of the web service to match the requirement from the workflow. Meanwhile data type matching is used to avoid rejecting web service which meets the client requirement. Generally, it is common for web service providers to create their own data type for their web service. So, we need to match the data type provided by the service provider with the data type used in our workflow. This will reduce rejection of high QoS valued web services.

PROPOSED SOLUTION

Dynamic Web Service Adapter Scenario

Web Service Composition consists of three main phases and in this paper we are focusing on the second phase, the selection phase. Figure 1 shows the Dynamic Service Adapter Scenario of our selection phase. Figure 1 depicts how our web service adapter takes part when a web service fails to operate or become unavailable in a web service composition workflow. Once the web service fails, the workflow will request for a new web service from the adapter based on the requirement.

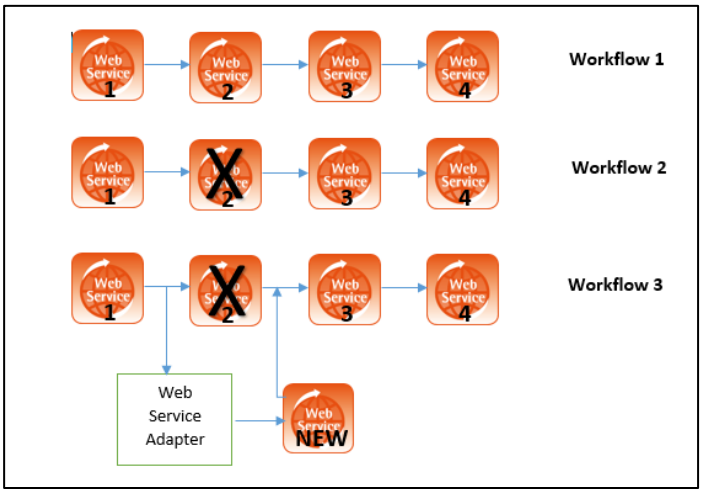

Figure 1. Dynamic Service Adapter Scenario

Figure 1, shows how the adapter works to find a new web service as a replacement. **Workflow 1** shows the composition of web services with no failure during execution. **Workflow 2** shows the same composition but with a failure of one of the web services in web service composition workflow. At this case, a new web service is required to be replaced to ensure the workflow does not fail. This was identified to be a solution for a system which is supposed to provide early warning notification to the end user whenever floods or landslides occur in Malaysia. This is a real-time system which will plot the warning sign on the map. By default it will be using the OpenStreetMap (OSM) service as stated in Haklay et al. (2008) which runs in our own server. As replacement, we have choices such as Google Map service and Bing Maps service. This replacement of service is required to be done on real-time if our server faces down time.

Figure 2 (a). System working as normal

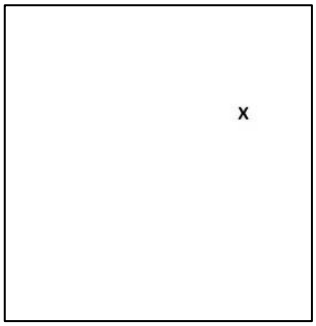

Figure 2 (b). Failure to load Map service

Figure 2(a) shows the normal process as stated in Workflow 1 where Dungun (a city in Malaysia) is marked as our flood monitoring station. Meanwhile if the map service fails, then the output will be as shown in Figure 2(b) whereby only the mark will be displayed. Thus, a real time replacement of service is very important to ensure the system is usable at all times. The input variable for OSM and Google Map is the same but the input data type for Google Map might be different. By using this adapter, it will match input variable to Google Map data type and the application will run without any failure. A replacement of a web service can be done manually/statically or dynamically. In **Workflow 3**, we have proposed a dynamic method that is able to solve web service failure. We are using an adapter which works independently whenever a web service fails in a workflow. This ensures reliability for the workflow as chances to fail will be least.

Proposed Web Service Selection Algorithm

The illustration of the proposed algorithm is shown in Figure 3. The WSDE phase has already sorted the web services according to the QoS values based on user preference. Web service will be provided to the selection unit as a list. Based on the list, the selection unit will filter it based on the requirement submitted to the adapter by the workflow. The filtration process is completed by using WSDL description given by the web service developer. We use Word.NET to find the synonym of the description and match it with the workflow requirement. The selection unit processes the data type matching of every service before sending it over to the composition unit. In this work, we solely focus on dynamic adaptation. We have proposed an algorithm for data type matching as shown in Algorithm 1. There will be two inputs required: (1) Data type accepted by the Web Service (2) Input data from the workflow. There will be two types of generated output namely: (1) Output to Web Service will be matched with Web Service data type or (2) Output to Workflow will be matched to workflow data type. Table 2 explains the data flow of web service selection phase.

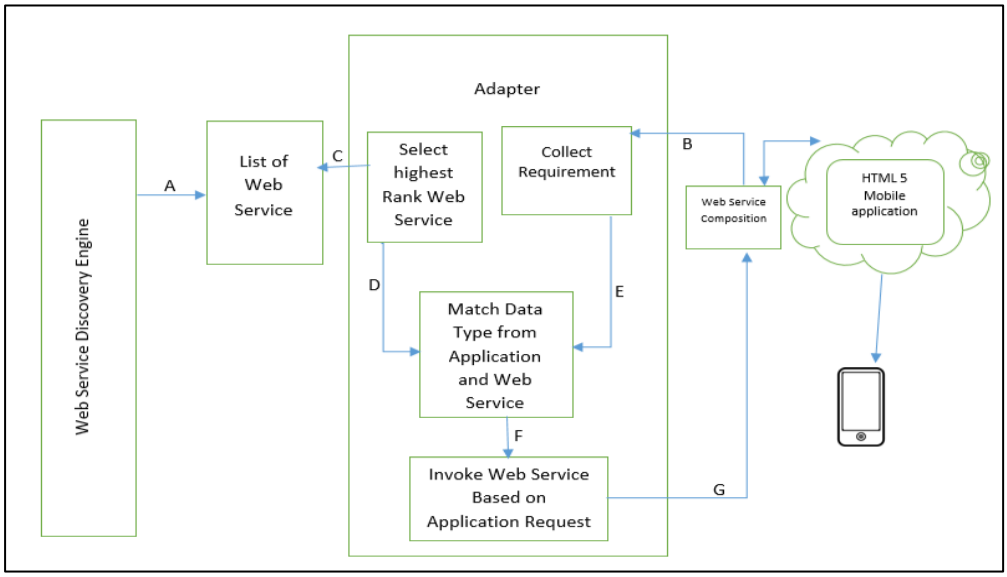

Figure 3. Overall Architecture of Web Service Selection

Table 2. Dataflow of Web Service Selection Phase

Activity Flow	Description
A	Web Service Discovery Engine will generate list of web services based on User's defined QoS Attributes and requirement provided by the workflow to adapter
B	The HTML 5 mobile application which is placed in the cloud environment will request for the data and also include the data type accepted in the application to the adapter.
C	The adapter will select the top ranked service as recommended by the Service Discovery Engine.
D	The selected web service will be analyzed on the data type accepted and output data type. If the output is not as expected by the application, the next web service will be requested.
E	The requested data type and sent data with its data type is sent to a matching engine.
F	The web service will be invoked based on the output of the data type matching.
G	The output returned by the web service is returned to the mobile application after converting the output based on the required data type.

Algorithm 1: Match Data Type from Application and Web Service

```
Input: Data type from web service selected and Application
Output: Converted output based on Application request
While Loop
    Receive Input Data type from Application
    Check for Input Data Type of Web Service
    Create a variable based on the web service input data type
    Assign the input value from Application to the variable created.
    Variable sent to web service for processing
End Loop
Receive Output from Web Service
Check Data Type matching
If matched = True
    Return to Output to Application
Else
    Convert the output to Application requested Data Type
    Return the converted output.
End IF
```

CONCLUSION AND FUTURE WORK

In this paper, we explain how the web service selection adapter works to find a new web service as a replacement. This adapter works independently to recover a failed workflow by dynamically replacing a new web service based on the workflow requirements. The adapter uses WSDL description to match user requirement and data type matching to match the data type between web service and workflow. To guarantee reliability during real time execution, we propose a web service selection based on the recommended web service by our WSDE. The WSDE covers the QoS based ranking to ensure quality of services is not compromised. In future, this research work will be expanded to Cloud-Based mobile application powered by

the web service composition. We aim to create web service powered mobile application to reduce coding effort and increase reusability.

ACKNOWLEDGMENTS

This research is supported and funded by Japan International Cooperation Agency (JICA) and Ministry of Education (MOE), Malaysia.

REFERENCES

Alrifai, M., Skoutas, D., & Risse, T. (2010, April). Selecting skyline services for QoS-based web service composition. In *Proceedings of the 19th international conference on World Wide Web*, 11-20. ACM.

Gao, H., Yan, J., & Mu, Y. (2011, July). Web service selection based on similarity evaluation. In *IEEE International Conference on Services Computing (SCC)*, 322-329. IEEE.

Haklay, M., & Weber, P. (2008). Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4), 12-18.

Kang, G., Liu, J., Tang, M., & Xu, Y. (2012, May). An effective dynamic web service selection strategy with global optimal QoS based on particle swarm optimization algorithm. In *Parallel and Distributed*.

Zeng, L., Benatallah, B., Ngu, A., Dumas, M., Kalagnanam, J., & Chang, H. (2004). Qos-aware middleware for web services composition. *IEEE Transactions on Software Engineering*, 30(5), 311–327.

Li, M., Huai, J., & Guo, H. (2009, September). An Adaptive Web Services Selection Method Based on the QoS Prediction Mechanism. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*, 1, 395-402. IET.

OpenStreetMap. [ONLINE] Available at: https://www.openstreetmap.org/. [Last Accessed 15 April 2015].

Oussalah, Y., & Zeghib, N. (2013). Adaptive Web Service Composition Based on Interface Processing. *IEEE 26th International Symposium Workshops & Ph.D Forum (IPDPSW)*, 2280-2285. IEEE.

Tang, M., & Ai, L. (2010, July). A hybrid genetic algorithm for the optimal constrained web service selection problem in web service composition. In *IEEE Congress on Evolutionary Computation (CEC)*, 1-8. IEEE.

Wang, D., Huang, H., & Xie, C. (2014). A Novel Adaptive Web Service Selection Algorithm Based on Ant Colony Optimization for Dynamic Web Service Composition. In *Algorithms and Architectures for Parallel Processing*, 391-399. Springer International Publishing

Wenmin, L., Wanchun, D., Xiangfeng, L., & Chen, J. (2011, July). A history record-based service optimization method for QoS-aware service composition. In *IEEE International Conference on Web Services (ICWS)*, 666-673. IEEE.