Dynamic Web Service Composition and Parameters Matchmaking

Maria Allauddin
College of Electrical and Mechanical Engineering
National University of Sciences and Technology
(NUST) Islamabad, Pakistan

Farooque Azam
College of Electrical and Mechanical Engineering
National University of Sciences and Technology
(NUST) Islamabad, Pakistan

ABSTRACT

Service Oriented applications are becoming very popular due to ease of Web services Usage. One use of Web Services in computer applications is its automated Composition. Excess amount of work has been done for automated web service composition but still there is a space to fill out for particular requirements. It's been said that full automation of services is not much beneficial. So there is a need of minor interaction of user. Our framework presents a Service Composition including user interaction. Communication between client application and sever is done using XML messages. Using Xml messaging reduces the effort to invoke complex services. User selects the services to be composed from the application. Services are analyzed and matchmaking is performed. In matchmaking we have performed a check to compare number of input output parameters. If the number is same, composition is performed without any interruption. If numbers of parameters do not match the user is prompted to select or enter the required parameters.

General Terms

Web Service Composition

Keywords

Dynamic Web Service Composition, Qos, Matchmaking, XML

1. INTRODUCTION

Web services are tiny components that serve as substitute of applications. This is a superb inclination in technology of time. Web services are present on internet and are small individual components which when work together reflect as an application.

As web services are made to be served on internet, it is the basic requirement that they should be platform independent. They are flexible enough to be integrated with other services as well as function in total impendence. Every individual operation of the service is self governing. Due to this services need not to follow any specific rules or logic of programming.

Let's take an example of online software that deals with tourism: a tourist portal or a travel guide. This tourist portal may need more than one service at a time. For example, ticket booking, area information, hotel reservation, and weather information.

Assume all the above examples are web services available. Let's suppose that hotel reservation depends on area information. So there is need to use more than one service sequentially. This is called web service composition.

A speculative definition can be as: "Service composition is incorporation of several existing web services in such a coherent order that fulfills user requirement".

So service composition has made considerable effect on large usage of existing services. High use of service composition reduces the hectic job of new development and creates new applications.

Web services can be described in two ways 1) functionality it provides 2) the way it performs its functionality. Similarly services can be composed in two ways. 1) Static Web Service Composition 2) Automated or Dynamic Web Service Composition. Static needs the services to be executed fully manually which is time hectic job. Where as in dynamic composition the services are invoked automatically one by one as is seen as a single service. In full automation there must not be any intervention of user during composition process. However this cannot be achieved in reality. Few problems faced are given below:-

• Dynamic composition needs very little user involvement which makes it difficult to find out an exact required service on huge repository of internet.

• Second not all the services on internet are public. So there is a need to select service of user interest. Not only to functionality but also its accessibility.

• Transactional support can be very small in fully automated composition as different service providers may have different conceptual models.

• Compositional correctness cannot be guaranteed as automation cannot verify middle stages of composition.

• Full automation is possible for specific infrastructures. If there is need of general application in which requirements change every time. A little user intervention is helpful.

So for a dynamic environment where user requirements for a service change frequently a semi-automated service composition is necessary.

In this paper, our basic intention is to provide an automated service composition with varying user requirements while consuming minimum time. Composition process is interface based i.e. a matchmaking if performed for number of input output parameters. User is prompted if the quantity of parameters is not same. The user has option to enter other parameters or select some from first output as required. We have used basic Xml messaging to service invocation that helped us

out to invoke and compose services without any problem for coping with different structures of web services. Our purpose is to enhance the composition process.

A. Contribution

We have proposed a framework to overcome Dynamic Web Service Composition problems for varying user requirements.

Using simple xml messaging we have provided a flexibility of communication for integration. Also it's helpful for complex type services where most frameworks fail to integrate services. There is no need to worry about data types and hence complex data types. It provides a reliable and trust-worthy composition. User can select service of his own interest and hence overcomes the problem of updates and availability. Compositional correctness is guaranteed by matchmaking and final input decision by user.

The organization of paper is such that Section 2 describes the previous research related to web service discovery. Section 3 presents the detailed overview of proposed framework including algorithm. Section 4 presents the implementation and key mechanism and Section 5 gives evaluation and results. Finally, conclusion and future work is given in section 6.

2. RELATED WORK

Schahram Dustdar* and Wolfgang Schreiner in [1] performed composition frameworks analysis, some modules and supporting features are given by them for composition design and development. They say that the current web service technology is quite limited as seen from results of different research papers. This is due to dependency of this technology on standards as SOAP, WSDL or UDDI. They compared different composition strategies by finding similar features. Finally they concluded that interactions among services with different specifications, their non-functional attributes, and internal changes invisible to the outside world have to be considered with more attention.

Freddy L´ecu´e1, Eduardo Silva2, and Lu´ıs Ferreira Pires[2] have given a framework for automated composition. They used SPICE ACE for automated composition. Functional and non functional properties are used to make a distinction between services and their requests. Composition factory part is used to figure out what new service is added and what are its non functional properties. Causal link matrix is used to identify weather a composition is valid or not. A request is neglected if non functional properties of composition are neglected. They aim to examine and combine a process aspects approach with their framework.

Faisal Mustafa, T. L. McCluskey [3] made a sketch of major challenges tackled by automated web services composition. The problems are associated with distributed, dynamic and unsure disposition of web service. Their model is semi-automatic but fixes few problems of fully automated service composition. They pointed out that internet has huge repository of services it is not possible to automatically analyze them. And hence integration is difficult. They said that second difficulty for automated service composition is that web services are updated frequently. So there is a need to have current information and decision of composition should be based on that. Their technique has few drawbacks. If server is not working input output problems occur. Also their repository does not contain updates information.

Pat. P. W. Chan and Michael R. Lyu [4] gave Dynamic Web Service Composition using Nversion programming. This method expands the reliability for planning among web services. If a server fails, there is another to provide the required service. Their composed service a free of deadlock is consumes less time for composition. Since the frame work is dynamic it uses updated versions without the need of rewriting the specifications. To verify the correctness of algorithm experimental evaluation and results are given at the end.

LIU AnFeng et al. [5] proposed a technique that supports web services interface. They used peer to peer ontology with overlay network to provide composition. The composition uses domain ontology and DHT Distributed Hash Table for composition and discovery. Their analysis shows that separation of details of interface from underlaying details makes it easy to understand. The composition is fast and fault tolerant and hence provides Qos Based execution.

Kazuto Nakamura, Mikio Aoyama [6] presents a structure for dynamic web service composition. They used value based composition and provided QoS. A meta model of Value is used together with the VSDL. Quality of web services is defined by this model. Providing values to the available services. To compose the services Value added service broker architecture is used and to define relationship among values the value-meta model is employed.

R. Jaya prakash, R. Vimal raja [7] evaluated different web service composition methods using Business Application. They considered basic perspectives like QoS, scalability, and
Correctness to analyze different methods. They performed the analysis by making an online book shop. The Web service composition approaches are compared with each other based on connectivity, Exception handling and Compensations. They indicated the results as Good, average and Low. They describe that different methods provide different automation level and we cannot conclude that higher automation is better due to high complexity of web service environment. So full automation cannot be provided. Though highly automated methods are appropriate for making implementation structures that are requirement specific it is almost impossible to implement in highly fluctuating environments.
Freddy L´ecu´e, Alain L´[8] outlined major challenges of Semantic Web Services. They used xCLM for automated service composition which provides formal model to face the challenges. They described that advantage of functional level composition is use of OWL-S. They analyzed different proposals and concluded that no formal model is helpful for automation of composition yet. Their framework is robust, secure and verifiable. Matchmaking of input output parameters is performed at functional level. Matchmaking functions summary is provided that gives details of how different comparisons are made. Overall the Service composition is viewed as fundamental link composition.

San-Yih Hwang, Ee-Peng Lim, Chien-Hsiang Lee, and Cheng-Hung Chen [9] formulated the dynamic WS selection procedure in a dynamic environment that is failure prone. They proposed

FSM usage to invoke operations of service in an order. They define AR to find a probability that execution of services will terminate successfully. They used Eigen vector to show aggregated reliabilities. The approach can be used only in industrial applications and hence is environment specific.

Liping Liu , Anfeng Liu , Ya Gao [10] used Particle Swarm Optimization for Service Composition. PSO is meaningful for the composition of complex services spread on internet. If there is requirement of multiobjective composition only PSO can do so. A non inferior pareto solution is provided by PSO group search. The solution meets all the required constraints. They used general service overlay model. They say that full automation of service composition is complex rather unachievable task that's us why most of algorithms are semi automated. They said that their algorithm can be applied to specific compositions.

Zhang Hai-tao, Gu Qing-rui [11] presented a dynamic process of domain ontology-based method. Their method considered semantics of service for composition. They verified their work by experimental examples. Their module for service composition makes a portfolio as soon as a service request is received. When all services are determined for integration it starts calling webs services. An OWL-Agent is used to mark functions of the service by forming OWL-S documents that call the services. Shortcomings of this approach is it must have a fixed field of experts in the field service portfolio template firstly, and then selected the user needs to match the template in the service of specific Web services. The limitation of the method is the construction of the field of service composition templates requires human intervention, so degree of automation is reduced.

Yujie Yao, Haopeng Chen [12] solved the rulebased Web service composition problem. They gave a framework in which selection engine executes business rules. Pereto optimal solution is obtained by an algorithm called NSGA-II. This is a selection algorithm. They said that the work can be extended by a large scale implementation. A comparison with some other non linear techniques can be performed to find benefits. There is still a space to research about the communication between selection engine and composition engine.

Farhan Hassan Khan, M.Younus Javed, Saba Bashir [13] presented a framework for dyamic web service composition and execution. At first they discussed major problems of dynamic composition. Then proposed an algorithm for dynamic web service composition. They mentioned composition issues like reliability, availability, data distribution. They introduced the concept of multiple repositories for system reliability. Avalibility is also guaranteed by this concept. An aging factor is used to retrieve uptodate information. They claim that their system is reliable, fault tolerant and performs fats data retrival. They said that dure to limitations of UDDI there is a need to extend this framework to service crawling.

Kaouthar Boumhamdi, Zahi Jarir [14] research contains a contribution in dynamic composition of Web services. First they stated some approaches of Dynamic Web Service Composition which are of vital importance. They made a comparison of those approaches and on bases of final analyses they proposed a new architecture. They argue that the architecture is flexible and modular. They gave the idea of service adaption on the fly. The adaptation is according to user requirements and sometimes according to the availability of resources. In addition this to preserve a better QoS, dynamic reconfiguration is guided. This is done by integrating the more appropriate Web service composite. They used a BPEL selection manager and hence their approach can be applied to specific scenario.

While an excess work has been done for dynamic web service composition. There are still few requirements to be handled for betterment of composition process. Our aim in this paper is to resolve few composition issues for a generalized environment in which any service can be selected to be composed.

3. PROPOSED FRAMEWORK

The framework shown in Fig 1 is quite simple and understandable. It includes following steps.

1. User queries the system for required service.

2. Service from Web Service Database is selected.

3. Information of selected service is retrieved.

4. User enters required input parameters for first service.

5. User adds the service to composition module. And selects next service.

6. On basis of information of service matchmaking is performed.

7. If number of parameters match.

    i) Composition is performed.

    ii) Results are displayed.

8. If number of parameters do not match. User is prompted to take action.

9. Composition is performed after user's action.

10. Results are displayed.

All communication with web server is done by XML messaging.

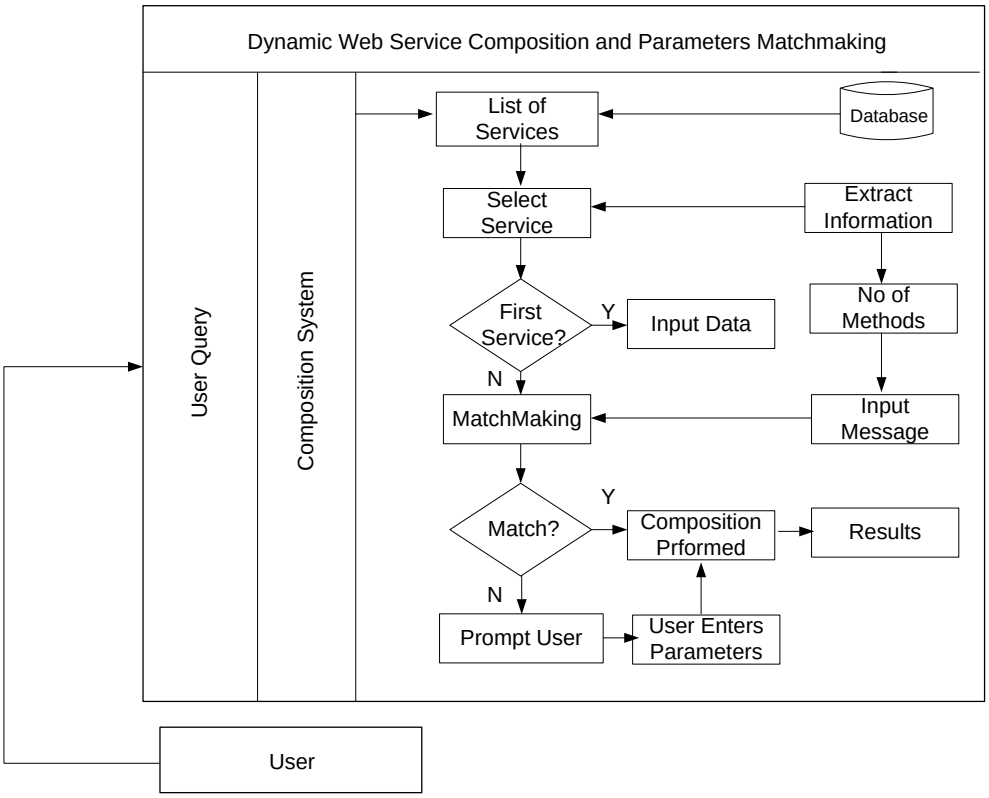

Fig 1. Framework for Dynamic Web Service Composition and Parameters Matchmaking

A. Pseudo Code:
The pseudo code of proposed technique is given as:

Algorithm: Web Service Composition
Input: Request for Web service
Output: Composed Service Results
User requests a desired service from Database;
User selects a service from list;
For each Selected Service
    Information about service is retrieved.
    The information includes number of methods and input message.
For First selected service user enters input data in XML message.
For remaining services selected Matchmaking is performed.
    If number of parameters match composition is performed and results are displayed.
    If number of output parameters are less than the number of required input parameters user enters the remaining parameters.
    If the number of output parameters are greater than the required input parameters. User selects some from previous result and/or enters other parameters. Composition is performed and results are returned to interface.

Fig 2. Algorithm for Dynamic Web Service Composition and Parameters Matchmaking

4. IMPLEMENTATION

The implementation has been done using Net beans 6.9. WSDL4J, SAAJ, Castor, JDOM, and XSLT. APIS are used for service call and invocation.

When user selects a service it is analyzed and available methods are displayed. Application displays the infrastructure of the subsequent input and output messages.

Our application creates a dummy Xml Message that is required to be sent for service invocation. When user enters input values they are merged with that dummy message.

WSDL4J, Castor Object Model, and JDOM are used to analyze and compose a local model of the Web service. An instance of WSDL4J's is created. This instance is reader interface for WSDL definition loading. Collection of methods related to that WSDL is returned by the application. Input and output message structure is obtained by Message objects. Message interface returns all parts of the message to be sent.

Now request message can be made on basis of information retrieved. Application makes the original XML request message to be sent when user enters the required input. To handle complex message which is almost not feasible using other strategies XML message is helpful. The above schema is referred to build such message that will be sent as part of SOAP message. If the expected response is also complex, the type already defined in schema is referred again.

SAAJ API is used to invoke the service. It is flexible and straight forward. SAAJ needs following steps to invoke web service: - Create connection, the message, and then add message content as required, after that message is sent to its destination and the result is processed. XSLT enables us to further process this result to transform into XML.

All above is the process to invoke single web service. During composition user selects more service, only additional step is matchmaking is performed and output of first service is sent as input to other.

In matchmaking only number of input output parameters are matched. If these are same composition is performed without any interruption. If number of output parameters of first service is more than the number of required input parameters for second. User is prompted to enter the required parameters. While if number of parameters of first are more than the required input parameters for second user is again prompted to select some from the intermediate output and/or enter from text box.

At the end of all services invocation response final result is displayed to the user.

5. EVALUATION AND RESULTS

Once a service is selected the method it provides must be known. The evaluation time of web service was noted for various methods exposed by the web service. The web services were randomly evaluated and the timings were noted. Figure 3 shows evaluation time of different services having 1 to 6 number of methods.

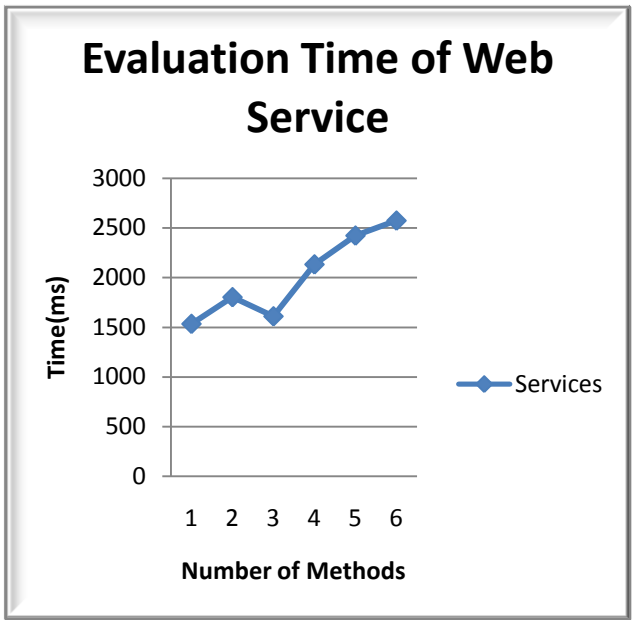

Fig 3. Graph for Evaluation of Web Services

After getting basic information different web services were invoked in isolation to find out how much time an Xml message and SAAJ takes to invoke an individual service. We found that service invocation time is less than the time utilized for basic information capturing. Also using SAAJ it takes less time as compared to other methods used for service invocation. Figure 4 shows a bar graph for invocation of single web service at a time using SAAJ API.

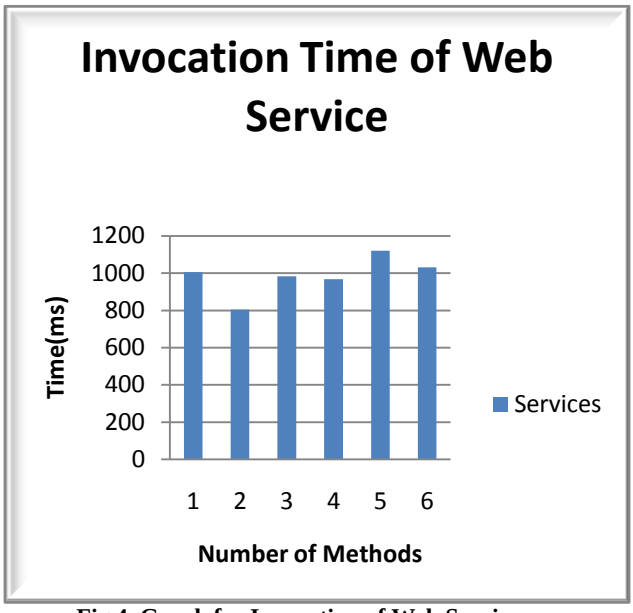

Fig 4. Graph for Invocation of Web Services

The web services were randomly composed to get fruitful output. The time for the execution of composite web service was logged. This was repeated for web services composed of 2, 3, 4 and 6 services. The graphical analysis of execution time of web service composition is shown in figure 5.

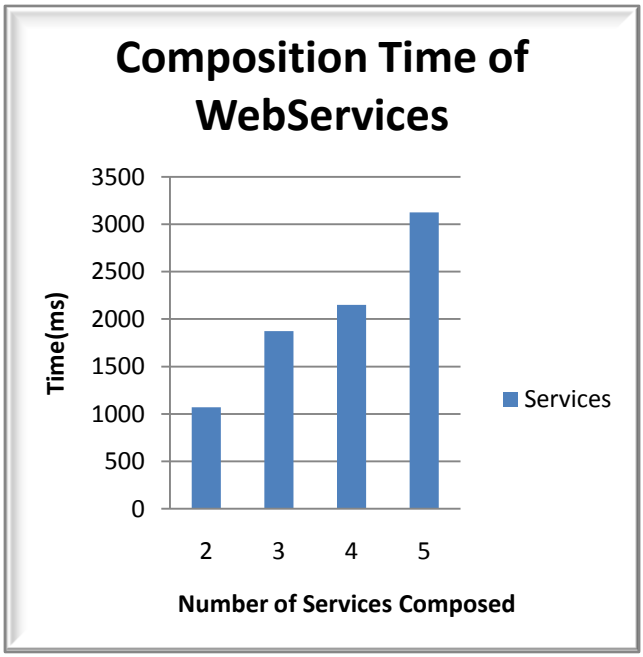

Fig 5. Graph for Composition of Different Services

A. Performance Assessment and Analysis

From graph of evaluation of services it can be noticed that the information retrieval is not dependent on number of methods exposed. The time shown includes the information retrieval of methods that service provides and creating a dummy message that includes input parameters.

Once the message is created and input is entered by the user. Less time is consumed for invocation of single service.

Composition time of services increase with increasing number of integrated services. Though the individual service takes less time we added all the times consumed to find out final time span. Time taken by user for matchmaking decision input is not included in the composition time graph.

When we compared our composition time with another frame work [13] we saw following results. Figure 6.

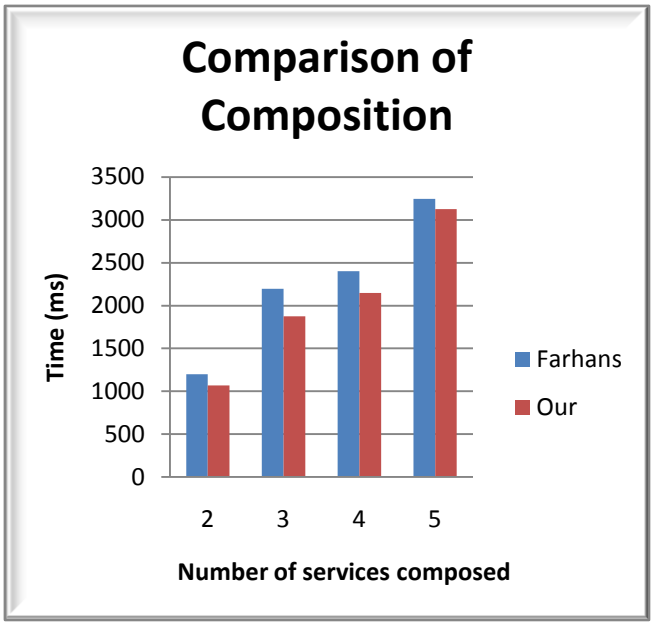

Fig 6. Comparison of Service Composition

[13] used WSIF for service invocation and composition and we have used SAAJ API.

Due to matchmaking step we are able to provide compositional correctness and transactional support also little user intervention of service selection guarantees the reliability of required services to be composed and the framework shows flexibility towards general varying requirements of service composition.

6. CONCLUSION

This paper presents a framework for service Dynamic service composition and parameters matchmaking. We discussed main problems faced by dynamic service composition. Among which are transactional support and compositional correctness. We made the system to be flexible in terms of automation hence we include user involvement at few steps for example selection of service and matchmaking decision. We have used SAAJ API and XML messaging that helps invoke complex services without hectic job of finding data types of any input output parameter. The values are passed as XML messages and hence consume less space for data type's examination and declaration. Matchmaking of input and output parameters guarantees compositional correctness and transactional support. Although at this stage we have only performed matchmaking of number of parameters latter we will try to find match of type of input and output parameters. More automation can be added by using some AI algorithm for service selection in future work.

7. REFERENCES

[1] Schahram Dustdar and Wolfgang Schreiner "A survey on web services composition", *Int. J. Web and Grid Services, Vol. 1, No. 1, 2005*

[2] Freddy L´ecu´e, Eduardo Silva, and Lu´ıs Ferreira Pires, "A Framework for Dynamic Web Services Composition".

[3] Faisal Mustafa, T. L. McCluskey "Dynamic Web Service Composition" 2009 International Conference on Computer Engineering and Technology

[4] Pat. P. W. Chan and Michael R. Lyu "Dynamic Web Service Composition: A -new Approach in Building Reliable WebService" 22nd International Conference on Advanced Information Networking and Applications

[5] LIU AnFeng, CHEN ZhiGang, HE Hui, GUI WeiHua "Treenet:A Web Services Composition Model Based on Spanning tree" IEEE 2007

[6] Kazuto Nakamura Mikio Aoyama "Value-Based Dynamic Composition of Web Services" XIII Asia Pacific Software Engineering Conference (Apsec'06)

[7] R. Jaya Prakash, R. Vimal Raja "Evaluating Web Service Composition Methods With The Help Of A Business Application" R. JayaPrakash et. al. / International Journal of Engineering Science and Technology Vol. 2(7), 2010, 2931-2935

[8] Freddy L´ecu´e, Alain L´eger "Semantic Web Service Composition through Matchmaking of Domain".

[9] San-Yih Hwang, Ee-Peng Lim, Chien-Hsiang Lee, and Cheng-Hung Chen ,"Dynamic Web Service Selection for Reliable Web Service Composition" Ieee Transactions On Services Computing, Vol. 1, No. 2, April-June 2008

[10] Liping Liu , Anfeng Liu , Ya Gao , "Improved Algorithm for Dynamic Web Services Composition", The 9th International Conference for Young Computer Scientists

[11] Zhang Hai-tao, Gu Qing-rui, "A Dynamic Web Services Composition and Realization on the Base of Semantic", 2010 IEEE

[12] Yujie Yao, Haopeng Chen, "A Rule-based Web Service Composition Approach",2010 Sixth International Conference on Autonomic and Autonomous Systems

[13] Farhan Hassan Khan, M.Younus Javed, Saba Bashir, "QoS Based Dynamic Web Services Composition & Execution", (IJCSIS) International Journal of Computer Science and Information Security,Vol. 7, No. 2, February 2010

[14] Kaouthar Boumhamdi, Zahi Jarir ," A Flexible Approach to Compose Web Services in Dynamic Environment", International Journal of Digital Society (IJDS), Volume 1, Issue 2, June 2011