ISSN (e): 2250 – 3005 || Vol, 05 || Issue,01 || January – 2015 ||
International Journal of Computational Engineering Research (IJCER)

Constraint Aware Dynamic Web Service Composition for A Specific Business Requirement

Raghu Rajalingam[1], Shanthi[2]
[1] PG Scholar, Dept of Computer Science and Engineering, Prist University, Tanjore, Tamilnadu, India
[2] Asst Professor, Dept of Computer Science and Engineering, Prist University, Tanjore, Tamilnadu, India

ABSTRACT:
In today's fast moving technology world everyone wants a better and quality of IT services they consume. Here the IT services are referred to web services. In recent time development IT solutions is reduced and most of the time spent on integrating the existing IT solutions. In this space web services has dominating the software industry. Every one focusing converting the existing solution to some form web services and offering the services to customer. The existing technology of web services are extended to completely customize the way services are composed as per the customer requirement so that it will add or provide value as well as the maximum benefit to the customer. Dynamically composing the web services based on the user constraint and providing services to customer always challenging. The proposed paper focus on to addressing how to compose web services based on the user constraint dynamically. The proposal is win-win for both service provider as well as the service consumer.

KEYWORDS: *Service oriented computing; composition of services; dynamic composition; UDDI registry; web services.*

I. INTRODUCTION

In today's fast moving technology world everyone wants to have better quality of service whatever they want to use with their constraints addressed. Here user constraints wary from user to user and it cannot be same for everyone. User wants to use a better reliable service or user wants to use less expensive services or with multiple constraints such as less expensive with more reliable service constraints. Currently lots of research is focused on in this area how to compose web services dynamically to address user constraints.

Composition of web services plays a critical role in case of B2B and integration of enterprise application. Here some time composition of web service may be a single service or it could be more than one dependent different web service to complete the user request. Most of the time service composition consumes lots of development time as well as to create new application. Now there is another parameter is added called user constraint and this will add more time to the existing problem.

The problem with current scenario is Service Consumer (SC) and Service Provider (SP) get in to legal agreement for consuming and providing service for a pre-determined duration such as 1 years or 2 year time frame. SC has to use the services offered by the SP with whom he has a legal agreement for that duration whether the services offered by him is good or bad. Imagine if services offered by the SP are really bad or not meeting the expectation of the SC, then he has to terminate the agreement to find better SP or give feed back to the SP that he is not happy with his service. Here SC's are forced to use the available services and SP dominate with their services without any improvement. SC doesn't have a much choice as there are few SP's and there is no much competition to improve the quality of the services.

How about provide a multiple option's to SC's so that the overall service delivery quality will improve. SC's will choose the service only meet their explicitly expressed constraints. Here SC's are not tied to any SP's and they can choose any service as per their requirement as well as their constraints. This will create a healthy competition among the SP's to provide better quality of service. Following are some of issues need to be taken care.

- Web services are dynamically created and updated so the decision should be taken at execution time and based on recent information.
- How to access easily the different web services provided by the different service provider with different conceptual data model without any technical difficulties.
- How to ensure the access of the web services allowed only for the authorized user. It should not be the case where everyone accesses all web services.

Web services can be categorizedin two ways on the basis of their functionality.

- Semanticannotation describes the functionality of web service and
- Functional annotation describes how it performs itsfunctionality.

WSDL (Web Services Description Language) isused for specification of messages that are used forcommunication between service providers and consumer.Web services can be composed in two ways.One is static web service composition and other is automated/dynamic web service composition. In static web service composition, webservices are manually composed, i.e. each web service is executed sequentially order one by one to fulfill the service consumer request.

Static web service composition is time consuming and tedious task. This requires lots of developer effort to ensure services are properly composed.In case of automated web service composition, a generic framework with intelligence are used compose web services which may internally have many web services to fulfill the service consumer request. From the service consumer point of view it is consider as single service.

Web serviced are composed either centralized data flow or decentralized data flow approach. In case of dynamic web services composition, both have advantages and some limitations. The limitation of centralized dataflow is thatall component services must pass through a composite service.

Preliminaries

Below section provide some insight about the web services composition, automated web services composition and actors involved in dynamic web services composition.

1. Web Services Composition

Web services are distributed applications. The main advantage of web services over other techniques is that web services can be dynamically discovered and invoked on demand, unlike other applications in which static binding is required before execution and discovery.

Semantic and ontological concepts have been introduced to dynamically compose web services in which clients invoke web services by dynamically composing it without any prior knowledge of web services. Semantic and ontological techniques are used to dynamically discover andcompose at the same time (run time).

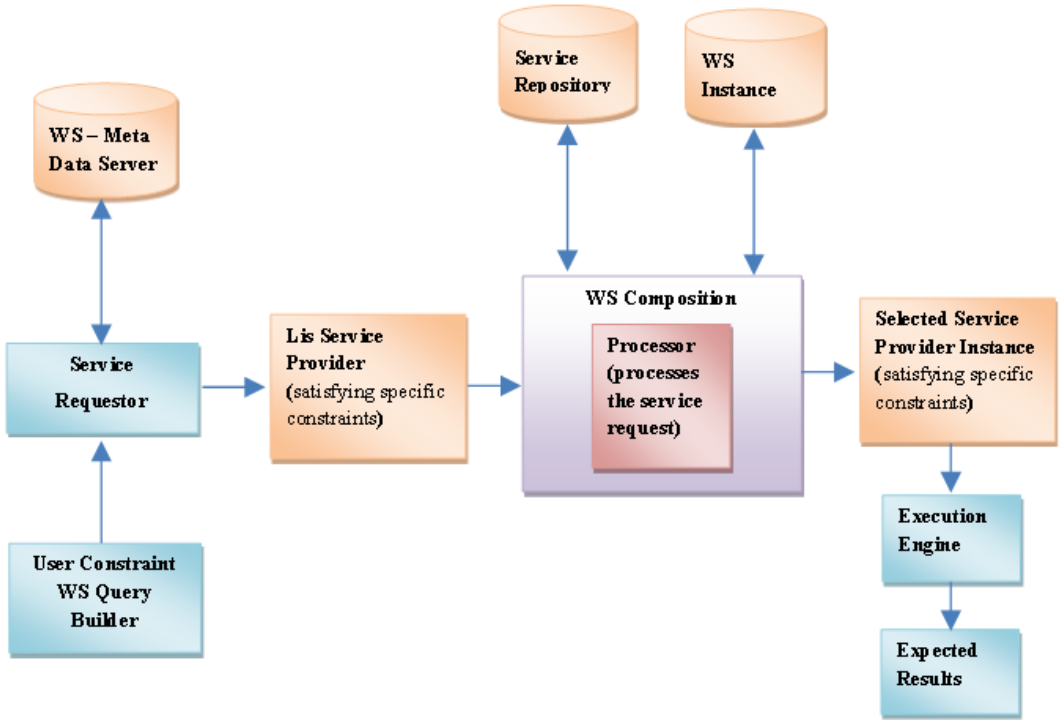
2. Automated Web services composition

The automated web service composition methods generate the request/response automatically. Most of these methods are based on AI planning. First request goes to Translator which performs translation from external form to a form used by system, and then the services are selected from repositories that meet user criteria.

Now the Process Generator composes these services. If there is more than one composite service that meetsuser criteria then Evaluator evaluates them and returns the best selected service to Execution Engine. The results are returned to clients (Requester). There should be well defined methods and interfaces through which clients interact with the system and get the response.

Proposed Solution

Here we are proposing a framework to address composing the web service dynamically based on the user context as well as with user constraints.



1. **User Constraint Query Builder** – User constraint query builder will construct the user query based on the user specific constraint for searching the web services from WS Meta Data server.
2. **Service Requester** – Constructs the user request object and submits the request to WS Meta data server.
3. **WS Meta data server** – All the Meta data related information is indexed for the faster retrieval of the search data.
4. **WS – Composition** – Constructs the WS – Client dynamically based on the user chosen web service.
5. **WS- Instance cache** – Is an in memory db to persist the instance of the frequently accessed WS instance.
6. **Service Repository** – Master DB of all the WS registered by the service provider.
7. **Execution engine** – A common framework to execute the WS calls.
8. **Response** – A common response object for the WS execution request.

Proposed Technique

A. Methodology

The methodology of proposed model is given as:

1. Web services Meta data are indexed for the faster retrieval during the search operation.
2. The web services are registered in registries.
2. Service requester send request for service.
3. Translator converts the query into a form used by internal system.
4. The request arrives at composition module. Matching Engine checks for the requested service from WSDBs. If it finds the desired interface base service composition then it sends results to Evaluator.
5. Evaluator evaluates these selected web services in two steps. In first step it evaluates the web services on the basis of interface based search, whereas in second step it performs the evaluation on basis of functionality based rule. After evaluation it sends selected services to composer. The purpose of composer is to compose these component web services. Multiple WSDBs are introduced, so that if one goes down then we can make use of other databases. A timestamp (aging) is maintained with each URI in WSDB. If the request arrives before the expiration of that time then it looks for the service in WSDB.
6. If Matching Engine does not find requested service composition from web services database then it start searching from web.
7. The web services are searched from multiple registries and results are returned to Evaluator. Matching Engine alsosaves their references in WSDB with aging factor. The purpose of aging is that it maintains the updated information about web services as the contents are refreshed each time when aging time expires.
8. Evaluator evaluates these web services based on interface based and functionality based rules.
9. Composer composes these evaluated resultant services and sends result to Execution Engine. Execution Engine executes these web services and through translator results are sent back to requester.

Pseudo Code
Algorithm: Dynamic Web services composition based on user constraints.
Input: Request for web service
Output: Composed service

Web services with their constraints data are ingested in WS-Meta data server;

User enters request for web service along with their constraints;

WS-Query builder translates the web service request;

Web Service request will be submitted to the WS-Meta data server;

WS-Meta data serversearch services for the submitted user query;
WS-Meta data serverreturns the matching services as per the user ws query;

User selects and submits web service matching his requirement to WS-Composition Engine.
WS-Composition search for the submitted web service from WS-Instance cache;
If selected ws instance is available from WS-Instance cache then
Return instance to Execution engine.
If selected ws instance is not available from WS-Instance cache then
    Create a new instance and add to the WS-Instance cache.
Return instance to Execution engine.
Execution engine will execute the selected web service and return the result;

Here we have query builder using which user can specify user constraints is. Based on the query a search request will be submitted to WS-Meta data server. WS-Meta data server will return the search result based on the user constraint. If any record match with the user query result will be returned to the user. User with the returned list of web service matching his constraint can choose one and complete his operation. When user chooses a specific service matches the constraint then a search is done in WS-Instance cache is there already an instance is available. If no instance is available then a query will be sent to service repository to ensure that service still registered and available. Sometime services are registered for a short time and later removed from the service register. To avoid such problem it is always good to make check before composing or executing the web service. Core component of this framework is WS-Meta data server. The Meta data server contains only the Meta data of the web services along with use full information such as pricing, availability, reliability, user rating, is partner etc. As these information may not available in service registry. Service registry may contain service name, service provider and name space etc. We are creating an index server which very fast in retrieving the required data based on the user query and it contains the Meta data of the services. The web services information along with the constraints are ingested in the Meta data and frequently kept updated as a when data changes. Sometime the data contained in Meta data server will obsolete as this is not the master database for the web services. This is a secondary server for fast searching data. Service registry will be the master copy of web services.WS-Instance cache is another important component as it contains the frequently used web services to speed up the web service composition and execution. There will be few services which will be used frequently by the user. We cache such service instances in WS-Instance cache so that in future if the same web service requested then it will retrieve from the cache instead of creating new one. If referred web service instance is already available in WS-Instance cache then retrieve the instance and execute the web service call. If it is not available then create the instance and add to the WS-Instance cache. This approach used to increase the speed of web service execution.

II. CONCLUSION

With dynamic composition of the web services based on the user constraints always ensures better quality of the services rendered to the service consumer. By this approach it will create a healthy eco system among the service provider to provide better quality service.

REFERENCES

[1]. A Constraint-Based Approach to Horizontal WebService Composition,Ahlem Ben Hassine, Shigeo Matsubara and Toru Ishida,3 Cruz et al. (Eds.): ISWC 2006, LNCS 4273, pp. 130–143, 2006,Springer-Verlag Berlin Heidelberg 2006.
[2]. An approach to dynamic web service composition, Dmytro Pukhkaiev, Tetiana Kot, Larysa Globa, UDC 621.93, National Technical University of Ukraine “Kyiv Polytechnic Institute”, Kyiv, Ukraine.

[3]. G. Alonso, F. Casati, H. Kunoand V. Machiraju. Web Services. Concepts, Architectures and Applications. - Springer, 2004. - 354 p.
[4]. M. P. Papazoglou, P. Traverso, S. Dustdar, F. Leymann: "Service-Oriented Computing: State of the Art and Research Challenges"; IEEE Computer, 40 (November 2007 (2007), 11; P. 38 - 45.
[5]. M.Papazoglou and D. Georgeakopoulos. "Service-OrientedComputing," Communications of the ACM, vol. 46, 2003. - P. 25-28.
[6]. W.M.P. van der Aalst, Benatallah B., Casati F. , Curbera F. , and Verbeek H.M.W.. Business Process Management: Where Business Processes and Web Services Meet. // Data and Knowledge Engineering. -2007. - 61(1). – P. 1-5.
[7]. Y. Jadeja, K. Modi, and A. Goswami. Context Based DynamicWeb Services Composition Approaches: a Comparative Study //International Journal of Information and Education Technology, vol. 2, Apr. 2012. - P.164-166.
[8]. S. Dustdar, and W. Schreiner, "A survey on web services com-position," in Int. J. Web and Grid Services, vol. 1, No. 1, 2005. - P.1–30.
[9]. S.Bansal, A. Bansal, and M.B. Blake, "Trust based Dynamic Web Service Composition using Social Network Analysis," IEEE International Workshop on Business Applications for Social Network Analysis (BASNA 2010), August 2010. - P. 1-8.
[10]. C. Molina-Jimenez, J. Pruyne, and A. van Moorsel. The Role of Agreements in ITManagement Software. Architecting Depend-able Systems III, LNCS 3549. Springer Verlag, Volume3549, 2005. – P. 36-58.
[11]. OMG Business Process Model and Notation (BPMN) [Online]. - 2011. - Available: http://www.omg.org/spec/BPMN/2.0/PDF
[12]. N. Russell, W.M.P. Van der Aalst, A.H.M. ter Hofstede, P. Wohed "On the suitability of UML 2.0 activity diagrams for business process modeling," Proceedings of the 3rd Asia-Pacific conference on Conceptual modeling APCCM '06, vol. 53, 2006. - P. 95-104.
[13]. Oberle, D., Bhatti, N., Brockmans, S., Niemann, M., Janiesch, C., "Countering Service Information Challenges in the Internet of Services," Journal of Business & Information System Engineering, vol.1, 2009. - P. 370-390.
[14]. H. Foster, S. Uchitel, J. Magee, J. Kramer, M. Hu "Using a Rigorous approach for engineering Web service compositions: a case study," in Proceedings of the 2005 IEEE International Conference on Services Computing (SCC '05), 2005. - P.217-224.
[15]. OASIS (2007) Web Services Business Process ExecutionLanguage (WSBPEL) [Online]. Available: http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf.
[16]. E.M. Maximilien and M.P. Singh., "A framework and ontology for dynamic web services selection," IEEE Internet Computing, vol. 8, Sep./Oct. 2004. - P. 84-93.
[17]. M. B. Blake, and D. J. Cummings, "Workflow Composition of Service-Level Agreements," in Proceedings of IEEE International Conference on Services Computing (SCC), July 2007,. - P.138-145.
[18]. A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. (2007) Web Services Agreement Specification (WS-Agreement) Online]. Available: http://www.ogf.org/documents/GFD.107.pdf.
[19]. A graph-based approach to Web services compositionHashemian, S.V.; Mavaddat, F. Applications and the Internet, 2005. Proceedings. The 2005 Symposium on DOI: 10.1109/SAINT.2005.4 Publication Year: 2005, Page(s): 183- 189.
[20]. Van der Aalst, W.: Don't Go with the Flow: Web Services Composition Standards Exposed.IEEE Intelligent Systems (Jan/Feb 2003).
[21]. Tony Andrews and Francisco Curbera and Hitesh Dholakia and Yaron Goland and JohannesKlein and Frank Leymann and Kevin Liu and Dieter Roller and Doug Smith and SatishThatte and Ivana Trickovic and Sanjiva Weerawarana: Business Process Execution Languagefor Web Services (2003)

Author Profile

Raghu Rajalingam is a R & D specialist with deep understanding and hands on experience in the area of Java enterprise architecture, Service oriented Architecture and Cloud computing.

Shanthi is an academician with interest in the area of data mining and service oriented architecture. She has good teaching and research experience.