Dynamic Composition of Web Services by Ontology Mediation Techniques

N.Sharmila Banu

***Abstract*—The Service Oriented Architecture (SOA) must be able to bind arbitrary web services at runtime. Service invocation is one by client components (stub) which was generated from service description at design time. Late binding needs an object representation of XML data types. The dynamic binding of services with previously unknown interfaces which exploits the potential of dynamic composition if supported in the invocation phase. In this paper, the approach is based on Ontology Mediation Techniques. It overcomes the interoperability between service interfaces. Ontology based models help to mediate and reconcile differences among service interfaces. It supports matching and transformation operations between XML elements and semantic concepts. Stubless invocation is achieved by exploiting without the need of fine-grained annotations of message and type definitions in service interface description.**

***Index Terms*— Web services, semantic web, service composition, service invocation, XML, service oriented architecture, ontology mediation, ontology mapping.**

I. INTRODUCTION

Web service is a technology, that allows applications to communicate with each other (i.e)., communication between client and server. Client makes request to the server and server responds as per the client request. A web services architecture contains three fundamental operations: They are Publish, Find, and Bind. Service providers *publish* services to a service registry. Service requestors *find* required services using a service registry and *bind* to them [4]. Web services are platform independent and language-dependent, it use XML languages. There are mainly 3 components in web services. They are, Simple Object Access Protocol (SOAP), Web Services Description Languages (WSDL) and Universal Description Discovery and Integration (UDDI)[4]. WSDL is an XML based language which explains, how to interface with XML based services. Through the WSDL, the client learns where the service can accessed and what operation it will perform. It contains six elements. They are, Port type, Port, Message, Types, Binding and Services.

A. *Service Oriented Architecture*

The Service Oriented Architecture is used to provide design and development of various organizations. It provides loose coupling between service providers and consumers. Service Broker is act as a third party component which results the loose coupling between providers and consumers. Request of service consumer, flows through the service broker and execute to address the request. Composite services can be specified in Business Process Model and Notation (BPMN) and business process execution language (WS-BPEL) and executed directly by the requester or a third party component [1].

B. *Stubless Invocation*

There are two types of stubbing. They are, explicit stubbing and implicit stubbing. Explicit stubbing are the stubs are resolved at compile time and implicit stubbing are the stubs are resolved at run-time. Implicit stubbing is also called as stub less invocation of web services. The invocation of services with a similar interface requires generating a new stub. For describing service interfaces, WSDL is a standard specification, based on XML [1]. If an application invokes a similar service from various providers, it must regenerate the stubs, because service from the different providers may not look same. It's difficult to implement a SOA based application using client stubs without falling back to generating and loading stubs at runtime.

Our message based invocation of services creates stubless and dynamic service clients that aren't strongly coupled to a specific service provider [4]. Concepts and relations are embedded in the exchanged messages. Ontologies provide the representation of concepts and relations. Ontology based model reconcile differences among service interfaces. WSIF supports a simple Java API for invoking web services. Other existing invocation frameworks are Apache WSIF, Apache Axis2, Codehaus XFire. The frameworks are highly dependent upon particular toolkit APIs [1].

C. *Composition of Web Services*

Composite services means combining multiple individual web services into larger components. No single web services to fulfil the user request. So, composite services which helps to satisfy the user requirement by providing various results. It simply states, set of atomic services together with data flow and control the services.

D. *Ontology Mediation*

Ontology mediation means reconciliation of different ontologies. Ontology can be in different languages, which need to convert them in a common format, so mapping can be

Manuscript received May, 2014.
***N.Sharmila Banu**, Department of Computer Science and Engineering, University College of Engineering (BIT Campus), Tiruchirappalli, India,*

specified. Ontology mapping is the process of identifying concept and attribute correspondences between ontologies through matching process. Many systems use the match operator to automatically find similarities between two ontologies. The match operator returns the similarity between ontologies. After similarities between ontologies have been found, the mapping between the ontologies needs to be specified [8].

Ontology mediation is sub divided into three types; i) Ontology mapping ii) Ontology alignment and iii) Ontology merging. Ontology mapping means representation of correspondences between ontologies. Ontology alignment is the automatic discovery of correspondences between ontologies. Ontology merging is to creating a single new ontology based on number of source ontologies[19]. In this paper, ontology mapping is used for stubless invocation of web services and reconciliation of service interfaces in web services composition.

The remainder of this paper is organized as follows. Section II represents the works related to the concept, section III describe our approach for dynamic service composition and invocation and ontology mediation ,section IV represents the framework, section V describe the experiments and result and section VI concludes the concept.

II. RELATED WORKS

In this section, related work in the area of dynamic service, composition, invocation and ontology mediation are discussed.

Paganalli and Parlanti [1], construct planning graph to solve an input/output matching problem. The composition services as syntactic, semantic and structural matching problem. Their approach relies on service profile model. For generating final solutions, they use graph plan algorithm. Structural and semantic properties are encoded in the service descriptions. It provides light weight semantic and data centric approach for the run-time specification of composite services. Leitner, Rosenberg and Dustdar [4] proposing the Dynamic and Asynchronous Invocation of services framework (DAIOS), it is a message based service which allows clients to invoke remote service through a message based stubless interface. The client request is handled by Daios, which chooses to invoke the service interface whose input message has the lowest structural distance metric to the provided data.

As proposed by Zheng and yan [2], the composition services as a syntactic matching problem, where the output parameters of one web service can be used as the input parameters of another one web service. For generating final solutions, they use backward search algorithm and the planning model. For pruning redundant web services, they use classical approaches with four strategies. Hewett, Kijsanayothin and Nguyen [3] the composition services as a state space search model. Each transition from one state to another state represents an invokable service. For obtaining final solutions, they use depth first search technique from an initial state to goal state and for pruning unnecessary web services. Zheng and Yan [2] and Hewett, Kijsanayothin and Nguyen [3] explains that messages are defined as a sequence of simple type elements. Our work is similar to these two approaches. Planning based approaches is enhanced with semantic service specifications to enable semantic based matchmaking on service contracts. Rich semantic models such as, OWL-S, WSMO.

Buhler, Starr, Schroder and Vidal [5], proposed a composite pattern for web service invocation, the capability of a system to bind a service and invoke one of its offered operations at run time. They addressed disadvantages of existing solutions as; dynamic invocation is supported whose message structure does not contain complex data types. While handling the complex data types it requires a pre processing step to generate an internal representation of XML data and the client code is highly dependent upon specific toolkits. Demian and Ananthanarayan [9] proposed broker based architecture for effective and efficient web service discovery and composition. The semantics and flow semantics of web service enable the web service discovery and composition mechanism. De Bruijn [8] described the mapping process of different ontologies. They described a practical approach of representing mappings using graphical tool to edit ontology mappings, discovering mapping using an alignment process and mapping language.

Granitzer et al [6], explains more about ontology mediation which helps to mediate the different ontologies. The service engineers will not always refer the same ontology. So, heterogeneous ontologies should be mediated.

III. FRAMEWORK

The objective of this work is to reduce the annotations of message to enables the dynamic composition of services and the automatic run-time binding of services. This approach is based on two principles. In this paper, model the service as message processor; whereas its interface is modeled in terms of messages [1] and the second principle is rely on ontology properties encoded in service description for dynamic service discovery, composition and invocation.

The message model is based on three levels. They are conceptual, logical and physical level. At physical level, data arranged is properly formatted messages to be sent on the wire e.g. (XML, SOAP). Service invocation is obtained by sending the message endpoint over proper communication protocol. In logical level, the target message is embedded with syntactic and structural model. At conceptual level, concepts and relations are embedded in exchanged message. In a domain, ontologies are used to represent concepts and relations. Ontology based model is used to reconcile and mediate the differences among service interfaces. No one refer the same ontology, so semantic heterogeneities should be mediated [6][7].

To call a web service, message has been sent to the program which is normally encoded in XML contains the information. The program receives XML replies contains the returned values. Normally, to invoke a service, there is the need of

pre-processing step to compile goal service description into an internal representation of XML data types. The client must marshal input data into a request message, and then unmarshal the returned data from the response message.

The planning graph is works in a forward direction from the initial stage to goal stage and it expand itself by each level at a time until a solution is found. Graph plan algorithm checks whether the goal are present at the current level, the algorithm searches f or a valid plan. If the plan is valid, the goals are satisfied at the end of the plan otherwise the plan is expanded by adding another level. If a solution is found, the graph plan ends its search and returns the sequence of step to reach the goal.

In Fig 1, explains the function of ontology mapping and its workflow. Ontologies are written in different languages, which need to convert them into a common format so that mapping is required. Furthermore, the ontologies need to be imported in the tool, which is used to specify the mapping. Next, Match operator is used to automatic finding of similarities between ontologies. After similarities between ontologies have been found, the mapping between the ontologies needs to be specified. For service invocation, it depends on stub based client interface which limits the scope of dynamic service composition; because stub-based interfaces are rely on service interfaces and they are generated at run-time. Whereas, stubless invocation which helps to achieve runt-time binding of services whose interface was unknown at design time.

In this paper, service composition and invocation approach address the issue by removing the need for generating an internal representation of XML data types at design time. Ontology properties encoded in the service description are exploited that allows the run-time specification of composite service plans, and execution through the stubless invocation of constituent services. In proposed system, ontology mediation techniques is used for mapping and transformation operation between XML elements and semantic concepts, which is to reduce the need for fine-grained annotations of message and type definitions in service interface descriptions. Concepts and relations are embedded in the exchange messages.

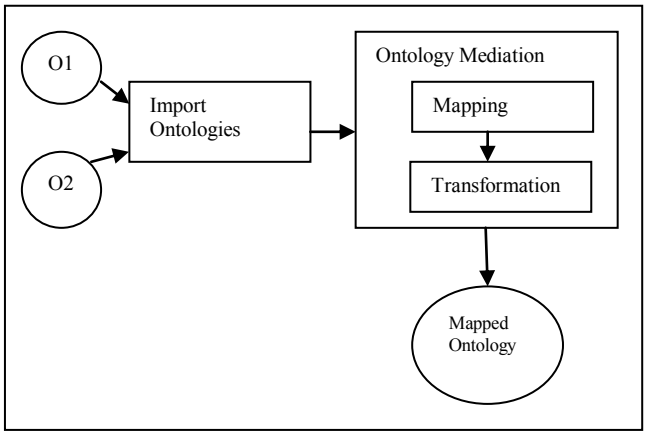

Fig. 1.*Ontology Mapping*

In existing system, the semantic and structural properties are encoded in service description for run –time service composition and execution of stubless invocation. Semantic annotations are used to describe the relationship between different service interfaces. In proposed, ontology mediation is used to reduce the semantic annotations for defining the relationship between various service interfaces. Semantic properties is exploited to connect various atoms belongs to same concepts. In Fig 2, given below which explains about proposed architecture and their functions.

A. Ontology Mediation

It means, sharing of data between heterogeneous knowledge bases and to allow applications to reuse data from different knowledge bases. Ontology based model helps to mediate an reconcile difference among service interfaces. It supports mapping and transformation between semantic concepts and XML elements for identifying similar interfaces. Ontology mapping is used to establish semantic relationships between two ontologies. MAFRA framework is used.

B. MAFRA Framework

MAFRA (Mapping Framework for distributed ontologies) supports dynamic ontology mapping process is to support instance transformation. There are two dimensions. Horizontal dimensions are, Lift and Normalization, similarity, semantic bridging, execution and post-processing. The vertical dimensions are, evolution, cooperative consensus building and GUI.

Evolution: Evolving ontologies on the semantic web result in an update requirement of the corresponding semantic bridges.

Cooperative consensus building: It is responsible for establishing a consensus on semantic bridges between two communities participating in the mapping process.

Lift & Normalization: It lifts the content of the ontologies and normalizes their vocabulary and compute the similarity between ontologies which support mapping discovery.
Similarity: This module establish similarity between entities from source to target ontology.

Semantic Bridging: It is established between the similar entities between two ontologies.

Execution: This module transforms from the source ontology into target ontology by evaluating semantic bridge.

Post-processing: This module, takes the results of execution module to check and improve the quality of the transformation results.

Graphical User Interface: Mapping is a difficult process. GUI allows the users drive the mapping process, provide domain constraint and background knowledge, create semantic bridges, refine bridges according to the results of the execution module.

C. Composition Engine

STRIPS model is used to represent the registered services profile. There are 3 types of operators. They are, functional, structural and semantic operators. Functional properties represent service operation. By using invoke_service label, the operator type is identified. Structural operator represent the relationship between the message and atom data by using exact operator. Semantic operator represent the semantic properties that associated atoms with semantic concepts. It is a component that takes a stream of messages as input that specifies one or more scene graphs and produces one or more output. In composition engine, it matches the client input data and goal data which is executed in run-time.

First, the request message is analyzed and semantic annotations are embedded in the request message which is used to translate client request into shared ontologies. Second, it is translated into the MAFRA framework for mapping and transformation of service interfaces. Third, it was translated into STRIPS model for dynamic composition and then forwarded to the planner. Finally , if feasible solution is found, the planner returns the set of invocation otherwise; it terminates by concluding no solution exists.

D. Plan Interpreter

The Interpreter translates the plans which were produced by Composition Engine into executable actions. It contains 2 main features. The one is, Endpoint Selection implements a QoS optimization algorithm for selecting endpoints. The second one is, Plan Interpretation which translates the operators.

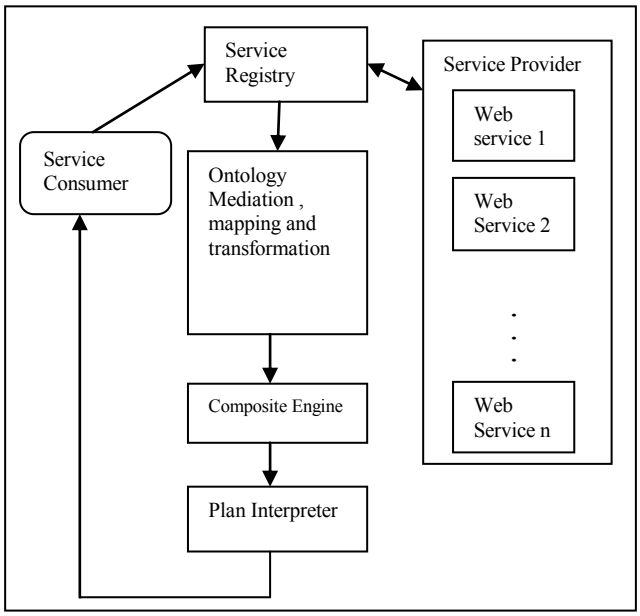

Fig. 2 *Proposed Architecture*

V. EXPERIMENTS AND RESULTS

Fig.3 represent the results of the experiments that carried out to evaluate the performance of composition mechanism. The test cases were defined by varying the number of registered services and the number of associated semantic concepts. Here, three types of request and its execution time are plotted. Request A, B and C are taken. the graph is plotted between number of registered services and service composition of 3 service request.

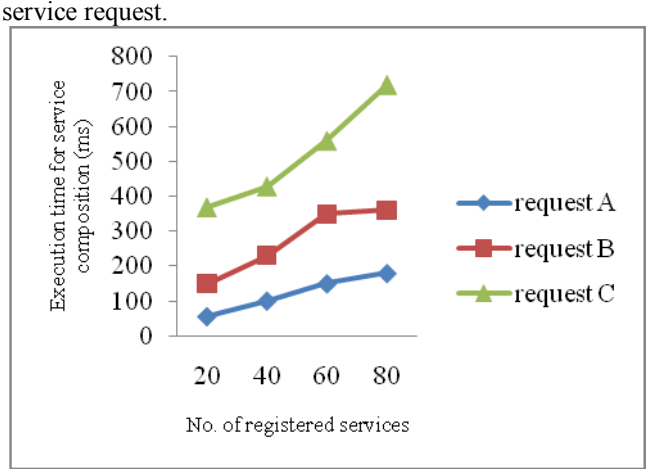

Fig. 3.*Execution time for service composition*

VI. CONCLUSION

In this work, proposed ontology mediation techniques are to reduce the detailed annotations of message and type definitions in service interface descriptions. It reduces the interoperability between the service interfaces and loose coupled interactions between the service providers and consumers through dynamic composition of services. The model of service invocation as message processor is to minimizing the requirements on service description complexity. The mapping and transformation operation between XML elements and semantic concepts are obtained by applying ontology mediation.

REFERENCES

[1] Federica Paganelli, Member, IEEE, and David Parlanti, "Dynamic Composition and Stubless Invocation Approach for Information –Providing Services," IEEE Transactions on network and service management, Vol. 10, No.2, pp. 218-230, June 2013.
[2] X. Zheng and Y. Yan, "An efficient syntactic web service composition algorithm based on the planning graph model," in Proc. 2008 IEEE Int. Conf. Web Services, pp. 691–699.
[3] R. Hewett, P. Kijsanayothin, and B. Nguyen, "Scalable optimized composition of web services with complexity analysis," in Proc. 2009 IEEE Int. Conf. Web Services, pp. 389–396.
[4] P. Leitner, F. Rosenberg, and S. Dustdar, "Daios: efficient dynamic web service invocation," IEEE Internet Comput., vol. 13, no. 3, pp. 72–80, May 2009.
[5] P. Buhler, C. Starr, W. H. Schroder, and J. M. Vidal, "Preparing for service-oriented computing: a composite design pattern for stubless web service invocation," in Proc. 2005 IASTED Conf. Software Engineering, pp. 276–281.
[6] M. Granitzer, V. Sabol, K. W. Onn, D. Lukose, and K. Tochtermann, "Ontology alignment—a survey with focus on visually supported semiautomatic techniques," Future Internet, vol. 2, no. 3, pp. 238–258, 2010.
[7] Lécué, S. Salibi, P. Bron, and A. Moreau, "Semantic and syntactic data flow in web service composition," pp. 211–218.
[8] J. de Bruijn et al,"Ontology mediation, merging and aligning". In J.Davies, R.Studer and P.Warren (editors), Semantic Web Technologies, pages 95-113. Wiley,2006.
[9] Demian Antony D'Mello and V.S. Ananthanarayana,"Dynamic web service composition based on operation flow semantics", International journal of computer applications, vol 1-no 26.
[10] Alexander, M., M. Boris, S. Nuno and V. Raphael , 2002. MAFRA A Mapping Framework for Distributed Ontologies, Proceedings of 13th European conference on knowledge Engineering and knowledge managementEKAW-2002,Madrid,Spain.

N.Sharmila Banu, she has completed her B. Tech in Dr. Navalar Neduchezhiyan College of Engineering and Technology affiliated to Anna University - Chennai, India and currently a final year M.E student at the Department of Computer Science and Engineering, University college of Engineering (BIT Campus), Tiruchirappalli. Her area of interest is Web Services.

