Dynamic Selection and Composition of Semantic Web Services Based on QoS

N. Thilagavathi¹, S. PuhaZHoli², A. Arutchelvi³ & A. Kokila⁴

¹Assistant Professor, Sri Manakula Vinayagar Engineering College, Puducherry,
², ³, ⁴ Student, Sri Manakula Vinayagar Engineering College, Puducherry,
(E-mail: ²puhazholi@gmail.com, ³maddy.chelvi@gmail.com, ⁴subha.koki@gmail.com)

Abstract: Web service selection and composition are gaining its momentum in the real world applications. It provides ways to interoperability between heterogeneous applications through different platforms. Many available services in service registry are similar in their functions with different QoS. In particular, the functionality of the existing services cannot be satisfied if the accurate services not available. The existing services are combined to fulfill the end user’s request. In this paper we propose a Dynamic composition of services framework which supports to describe the services semantically and selection and composition. Also we propose the selection of semantic services is made to best fit for composite schema. These semantic services are selected based on least execution time corresponding with IOPE. In experiment we study the performance of service selection with least execution time algorithm and feasibility for composite schema.

Keywords: Web services selection, discovery, composition, semantic web, annotations.

1. INTRODUCTION

Service oriented architecture [4] is used for linking resources on demand. In SOA, resources are made available to other participants in the network as independent services that are accessed in a standardized way. It is providing more flexible loose coupling of resources. It ensures the flexible and simplified delivery of services to clients. A web service [3] is a self-contained, modular unit of business or application logic which provides business functionality to other application via internet connection. Many web services architectures today are based on three components: the service requester, the service provider, and the service registry, thereby closely following a client/server model with an explicit name and directory service. Dynamic composition of Web Services is the key technology of SOA implementation, and Web Services selection is an important issue in dynamic Web Service Composition. Web Service is a standardized way of integrating web based application using a way to communicate (SOAP), a way to describe services (WSDL), and a name directory server (UDDI) via internet backbone. The SOAP is independent from any programming model and semantics of an implementation. WSDL provides XML format for describing web services. It describes only the syntactic interface of web services; hence the pure WSDL cannot be used for automatic web service composition.

The purpose of this work is to present, a model of semantic annotations for describing the web services and the request and in other hand the algorithm which discovers and compose the web services. As the client request is not satisfied by a single service but by composite web service. The result is the exact services from combination of several existing web services. The data given by the service requester is matched against the annotated concept of service described by the provider. Then the service discovery based on least execution time is selected for composite schema to provide best fit resulting service.

The functionalities of the web service are described as semantics according to specification of OWL-S. The publication of web services have been recorded in a registry in the form of a
semantic before the service is requested. The obtained service based on input, output, preconditions, and effects. The similarity occurred between the services in a graph structure of composition. At the end of selection, we find several composition plans for services that satisfying the request.

2. RELATED WORK

2.1. Definitions

• **Ontology**: The knowledge of what is to be oneself.
• **Matching**: The matching operation between two concepts with some similarity features.
• **Similarity measures [5]**: There are four levels of similarities:
  • **Exact match**: Two services are seen as equivalent and they represent the same concept.
  • **Subsume**: Output of one service is super concept of input of other service.
  • **PlugIn**: Output of one service is sub concept of input of other service.
  • **Disjoint**: The output and input are incompatible.

2.2. Various Descriptions Languages of Web Services

• **BPEL4WS**: The Business Process Execution Language for Web Services is a language with an XML based syntax, supporting the process oriented service composition. BPEL is currently standardized by OASIS [1].
• **OWL-S [2]**: Ontology Web Language is a service ontology that enables automatic service discovery, invocation, composition, interoperation, and execution monitoring. This has Service Profile, Service Model, and Service Grounding.
• Other semantic descriptions of Web services such as Web Service Modeling Language (WSML), Web Service Modeling Ontology (WSMO), Web Service Description Language Semantic (WSDL-S)[3], and Semantic Annotations for Web Services Description Language (SAWSDL).

3. DYNAMIC SERVICE SELECTION AND COMPOSITION FRAMEWORK

Dynamic Composition of Service [7] is a framework for the support of all the users. Figure 1 shows the framework architecture.

![Figure 1](image_url)

3.1. Service Creation

The service creation, publication, discovery and composition are performed using semantic service description. The services in this framework are semantically described in terms of inputs, outputs, preconditions, effects (IOPE) and goal. The ontology used in our framework is OWL.

3.2. Service Publication

The service publication supports different service description languages. We use jUDDI as service registry that offers API for publication and discovery of services. The directory of services is given in Table 1.

3.3. Service request

The service request consists of a set of semantic annotation. The request in the form of natural language which users without technical knowledge the possibility of requesting services in a simple way.

3.4. Service discovery

The service registry queries using semantic annotations through the jUDDI [6][8] API
Inquiry function for services with IOPE which are semantically related to the service request.

3.5. Service Selection Algorithm
The following algorithm takes the inputs and outputs of the request and seeks for similar services in registry with least execution time. Then the selected services are given for composition. The steps preformed by the algorithm in selection are as follows:

Input: URL
Output:

// Search in semantic network for all the services whose input and output are similar with goal.
1. Search the related services with least execution time by the formulae
   For i = 0 to n
   \[ \sum p_i \]
   // where p_i is the computational time of each service;
   // n is the number of related services executed.
2. \( S_a \leftarrow \) get related service that exactly matched based on input, output and least execution time.
3. If requested Service==Available Service then
   Return \( S_a \).
   Else
   // for each found services create a composition plan.
   For \( S=1 \) to \( n \)
   \( S_i \leftarrow \) get related services based on input and least execution time.
   \( S_o \leftarrow \) get related services based on output and least execution time.
   // If (all the inputs and outputs are treated) or (there are no more goals) then stop the process, and compose services \( S_i \) and \( S_o \).
   \[ \sum S \] // \( S \) is the composition plan
   End for.
   End If.
4. For each new goal, go to step (1).

3.3. Service Composition
The services which are selected are given to composition. The composition plans are given as follows in Figure 2, 3, and 4. The selected services in composite schema gives the best fit service as requested by service consumers. The composition plans of services are given below and algorithm in section 3.2.

Figure 2
Figure 3
Figure 4

4. CASE STUDY
For experimental purposes, we used different components in the framework in java using the
tools such as OWL-S for semantic web service description, Pellet, Apache Tomcat, jUDDI as registry. The machines used in server side (discovery, composition, delivery), client side (User Interface), dynamic registry (stores published services), are Intel Core 2 Duo, 2.4 GHz, 2GB.

5. CONCLUSION

In this paper, we have proposed a dynamic composition of service framework in which service is requested and delivered at runtime. This provides best accurate service to end user by composition using service selection algorithm with least execution time as its non functional attribute at selection phase.

The main drawback of our framework is that the composed service’s execution time cannot be found accurately. In future we consider the composed service time and on more non functional user preference.

References