A Semantic Search Engine for Web Service Discovery by Mapping WSDL to Owl

M. Abdul Naseer¹ S, Mohamed Yasir Hussain² A. Mohamed Ileeyas³ R. Sahul Hameed⁴
¹³ P. G. Scholar ²Network Engineer ⁴Asst. Professor
¹³ ⁴Dept of Computer Science & Engineering, ³Mys Technologies

Abstract---Web services is a web based API which provides functionality over web to integrate within web based applications. In such systems, there is often a machine-readable description of the operations offered by the service written in XML based markup language called the Web Services Description Language. The WSDL contains description about how the service can be called, what parameters it expects, and what data structures it returns. This WSDL file will be accessed by the search engine for the process of service discovery. Web services annotation is redefined to enhance the services discovery mechanism by mapping it to semantic annotations (OWL-S). The ontology mapping engine search for the keyword relationship with the existing keywords kept in the local ontology base and creates an ontology keyword list. There will be situation in which the local ontology database may not have the set of keyword relations to form an ontology keyword list. For that propose a standardization engine to fetches ontology which are not present in the local ontology database. This process of mapping mainly relies on a local ontology repository which is standardized using the global ontology base and by applying Onto-Ranking. It helps to manage the resultant that we get from the discovery process in an unsorted manner.

Keywords: Semantic Web Service, ontology, mapping, WSDL, OWL-S, ontology-based standardization.

I. INTRODUCTION

Semantic Web Services (SWS) like conventional web services are the server end of a client-server system for machine-to-machine interaction via the World Wide Web. The main idea of the Semantic Web is to extend the current human-readable web by encoding some of the semantics of resources in a machine-process able form. Moving beyond syntax opens the door to more advanced applications and functionalities on the web. Computers will be able to search, process, integrate, and present the content of these resources in a meaningful manner. As a Semantic Web component, SWSs use markups that make the data machine-readable in a detailed and sophisticated way (as compared to human-readable HTML pages which are not usually easily understood by computer programs. here are five steps to deal with any web service.

1) The process of “Advertisement” in which the service provider aims to publish information about the benefits of the service and how to use it
2) The process of “Discovery” which can be defined as the process of finding the list of services that can possibly satisfy the user requirements;
3) The process of “Selection” that targets to select the most suitable WS. This process is usually based on application-dependent metrics (e.g., QoS).
4) The “Composition” process that integrates the selected WSs into a compound process.
5) The process of “Invocation” that invokes a single WS or compound process by providing it with all the necessary inputs for its execution.

In this project, focusing only on the web service discovery process. The discovery process is the process of finding a service that can possibly satisfy user requirements, choosing between several services, and composing services to form a single service. Here introduce the new discovery mechanism. This mechanism has the ability to provide optimal results for any service request and fulfills the above requirement. After the introduction part, this paper is organized in the following manner, first it contain the Discovery mechanism in Section II, about OWL-S Component in Section III. Section IV reviews some of the related works. Section V reviews the Ontology Search and Standardization Engine. Section VI views about Mapping Algorithm. Section VII presents the performance analysis of searching mechanism comparison with other mechanism. Section VIII concludes this paper.

II. DISCOVERY MECHANISM

WSDL represents a non-semantic definition language; so, this paper introduces algorithms used to redefine WSDL using one of the semantic definition languages. Then, an overview of the web ontology language for services (OWL-S) is presented. OWL-S is one of the most important and widely used semantic definition languages for web services. There is much background information which is cited within the different sections as the context needed.

---

Fig. 1: Organization of Discovery Process
WSDL is an XML format used to describe network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. WSDL is often used in combination with SOAP and an XML Schema to provide web services over the Internet. A client program connecting to a web service can read the WSDL to determine what operations are available on the server. Any special data types used are embedded in the WSDL file in the form of XML Schema. The client can then use SOAP to actually call one of the operations listed in the WSDL. In 2007, W3C introduces version 2.0 of WSDL. WSDL 2.0 has many new features such as interface inheritance, extensible message exchange patterns, and an abstract component model.

OWL-S defines an upper ontology used to describe the properties and capabilities of web services in OWL. OWL-S is a W3C submission since 2004 and from that day on many researchers uses it to describe SWSS. The OWL-S authors target to enable automatic web services discovery, invocation, composition, and interoperation. Fig. 2 shows the basic components of the service class used to describe the web services using OWL-S.

III. OWL-S COMPONENT

It has to care about the process of discovery, so that it will present the OWL-S service profile component in more details because its information will be the use of our discovery mechanism. Service profile information can be categorized into three categories:

1) Information that can be readable by human beings, it includes the service name, a description that presents the function of the service and the contact information of the service provider.

2) Information that deals with the functionality of the service. This information includes full data about Inputs, Outputs, Preconditions and Effects (IOPE). The OWL-S Profile represents two aspects of the functionality of the service: the information transformation (represented by inputs and outputs) and the state change produced by the execution of the service (represented by preconditions and effects);

3) Information that includes the quality guarantees, called profile attributes, that are provided by the service, possible classification of the service, and additional parameters that the service may want to specify.

![Fig.2: Ontology Services](Image)

IV. RELATED WORKS

The Semantic Web Services Initiative Architecture (SWSA) committee has created a set of architectural and protocol abstractions that serve as a foundation for Semantic Web service technologies. The objective is to define an interoperability model that can underpin a variety of architectures without prescribing specific implementation decisions. The SWSA architectural framework attempts to address five classes of Semantic Web agent requirements: dynamic service discovery, service engagement, service process enactment and management, community support services, and discovery services. A discovery architecture that is designed to be distributed, scalable, and reliable and address heterogeneity problems stemming from resource limited devices such as PDA, mobile phone etc. The interfaces the functionalities of the proposed architecture are described. The use of shared space makes this architecture reliable, fault tolerant, scalable as each node in the Internet can be imagined hosting a shared space. It enables easy composition of Web service as the requests are handled at RDF Triple level. In the current implementation, shared space is centralized.

The presented approach to hybrid semantic Web service matching, called OWLS-MX, utilizes both logic based reasoning and non-logic based IR techniques for semantic Web services in OWL-S. Experimental evaluation results provide strong evidence in favor of the proposition that the performance of logic-based matchmaking can be considerably improved by incorporating non-logic based information retrieval techniques into the matching algorithms. It complements crisp logic-based semantic matching of OWL-S services with token-based syntactic similarity measurements in case the former fails. The results of the experimental evaluation of OWLS-MX provide strong evidence for the claim that logic-based semantic matching of OWL-S services can be significantly improved by incorporating non-logic-based information retrieval techniques.

The representation of real-world semantics in logics only is known to be inadequate due to its limited expressivity. In addition, automated reasoning on Web service semantics expressed in first order logics turned out not to be sufficiently scalable to the Web in practice the semantic service profile in OWLS specifies the semantics of the service signature that is the inputs required by the service and the outputs generated. Furthermore, since a service may require external conditions to be satisfied, and it has the effect of changing such conditions, the profile also describes the preconditions to be satisfied before, and the expected effects that result from the execution of the service.

Ontology itself has the concept which is the foundation of knowledge base; on the other hand. The object model is the center of object oriented software engineering. Because ontologies are closely related to modern object-oriented software design, it is natural to adapt existing object-oriented software development methodologies for the task of ontology development. Selected approaches originate from research in artificial intelligence; knowledge representation and object modeling are presented in this paper. Some issues mentioned in this paper are related with their connection; some are addressed directly into the similarities or differences point of view of both.

It also presents the available tools, methods, procedures which show the corporation with object modeling and ontologies. Ontology is different from object-
oriented modeling (represented in UML) in several ways. First, the most profound difference is that the ontology technology is theoretically found on logic. While ontology allows automated reasoning or inference, object-oriented modeling does not. Another difference is the treatment of properties; while the ontology technology treats properties as the first class citizen, the object-oriented modeling does not. That is, while the ontology technology allows inheritance of properties, the object-oriented modeling does not. While the ontology technology allows arbitrary user-defined relationships among classes (a type property), the object-oriented modeling limits the relationship types to the subclass-super class hierarchical relationship. While the ontology technology allows adding properties to relationships such as symmetry, transitivity, and inversion so that they are used in reasoning, the object-oriented modeling does not.

V. ONTOLOGY SEARCH AND STANDARDIZATION ENGINE

The main challenge that faces the development of Semantic Web is the problem of standardization. Therefore, there are two points to concern about: 1) it does not make any sense to find the same definition for a certain concept repeated more than one time in the system. This does not include the case when there are different definitions for the same concept and each of them adds new information. 2) It is illogical to define a concept isolated from the already existing concepts’ definitions. In other words, it is better—from standardization point of view—to try to find a relation between the new concept definition and the already existing ones. The engine has three main stages: linguistic search, structural refining, and statistical refining. These three stages are described below:

1) Linguistic search. In this type of search, OSSE uses the text mining techniques to extract the most important keywords in the concept request. Then, it tries to find all the synonyms and related words to expand our list by the aid of WordNet, which includes over 90,000 word senses. The list of ontologies is arranged based on the keywords that they contain in terms of term frequency. For each ontology, the summation of term frequency values of each keyword that belongs to the concept request keywords list and belongs to the ontology at the same time is computed. This summation represents a measure of the degree of the ontology

\[
OLR = \sum_{i=0}^{NCF} \theta(i)T(F(k_i))
\]

2) Structural refining. In this stage, OSSE refines the list produced by the linguistic search. This refining is performed by searching in each ontology in the list to find any concept related to the required concept. If OSSE does not find any related concept in a particular ontology, this ontology is deleted from the possible ontologies list. “Data concerning the logical structure” which are collected using the inserting methodology, are considered to be the base of the structural refining. We present a simple example to clarify what “Related ontology” means; a simple example is designed to clear up that expression. We assume that the requested concept is called “CR” and it has three properties \{P1, P2, and P3\}. “CR” is compared against another concept called “C” which is part of four suggested concepts-trees that represent all the possible concepts structure in any ontology. Fig. 4 illustrates the four possible alternatives of Concept-to-concept Relationships:

1. Identical relation: “CR” and “C” have the same Properties;
2. super relation: “C” may be considered as a parent of “CR”;
3. Sub relation: “C” may be considered as a child of “CR”;
4. Neighbour relation: “C” and “CR” have some Common properties

Statistical refining. The choices of the service provider are stored in the “Concepts Mapping History” database. These data are used by OSSE to rerank the possible ontologies list. If there are two ontologies with the same rank in the previous step, OSSE uses this historical data to know the most preferred ontologies for the services providers.

Fig. 3: OSSE Implementation

Fig. 4: Concept-to concept Relationship

After these three steps, OSSE has an ordered possible ontologies list for the concept request. This list is presented to the service provider, who chooses the most suitable ontology from his point of view. His choice recorded in the “Concepts Mapping History” database and the selected ontology edited automatically. The process of ontology editing causes some changes in the database and the file system of the local ontologies repository. So, we design a simple C#.NET tool to modify the ontology OWL-S file to apply the new changes in the ontology logical structure. This tool is not a stand-alone tool but it is one of many sub tools that are used to implement our proposed mapping algorithm.
VI. MAPPING ALGORITHM

One of the most important steps in the process of mapping WSDL to OWL-S is the conversion from the WSDL types, typically XSD types, to OWL ontologies. There are two categories of XSD types: 1) Primitive (simple) XSD types, e.g., string, integer that need to be converted to OWL ontologies. They are defined directly as inputs or outputs of an atomic process in the service model file; 2) Complex XSD types which are translated into OWL ontology concepts.

According to the converter of complex XSD types have two alternative designs. The first one is to generate OWL-S specifications that make use of XSD types and make no use of OWL ontologies. The second alternative is to generate concepts that could be totally unrelated to ontologies available in the Semantic Web and therefore they are definitely useless from the automatic reasoning point of view. From our point of view, there is a third alternative that is to search into the Local Ontology Repository using the OSSE component to find a ranked list of the most related ontologies that already exist. Then, ask the service provider to choose the most suitable ontology concept. If the service provider does not accept any of the already existing ontologies, the system uses the second alternative.

Fig. 5 presents the steps of the conversion process. The XSD types are extracted one by one from the WSDL file. The first question is “Is this primitive or complex type?” If it is primitive the conversion process does not work and keeps it as it is. On the other hand, if it is a complex type the converter starts a process to find the most suitable OWL ontology. The process starts by creating a temporary ontology where each XSD complex type can be described by an ontology has one concept and many properties. This concept represents the input of the OSSE component. The output of OSSE is a ranked related ontologies list. This list is offered to the service provider to choose the most suitable ontology. In some rare cases, OSSE fails to find any related ontologies. In this case, the system inserts the temporary ontology to the local ontologies repository using the insertion methodology. After standardization we get the unsorted search results. It needs us to propose a new SWSs classification algorithm that depends on a new concepts ranking algorithm. This ranking algorithm has three components (self, parent, and children) and it uses the structural information available through the proposed repository. Without trying to benefit from the already available web services, any semantic discovery mechanism will face many problems to be widely used. So, the main task for this method is to help the service providers to restructure their WSs in a semantic manner.

VII. PERFORMANCE EVALUATION

Validating the proposed algorithm is done by using 1,000 of WSDL files which are included in OWLS-TC v3.0 and mapped using the proposed algorithm. The automatic phase of the mapping process is done within 1 hour. This time can be neglected when compared with the time consumed in the case of manual mapping. Fig. 15 shows a comparison between the WSDL2OWL-S tool and the proposed mapping algorithm. The figure presents the relation between the number of concepts generated by the system and the number of registered services. Here can obviously note that the number of used concepts will be increased when the number of services increases in both cases. But, in the case of WSDL2OWL-S tool the number of concepts increase with very high rate when compared to the case of our proposed algorithm. For example, when the number of registered services becomes 1,000, the average number of concepts per service is 2.66 in case of WSDL2OWL-S and 0.4 in case of the proposed algorithm. So, the proposed algorithm is more scalable than WSDL2OWL-S.
the experiments and the evolution concentrates on the classification process which is out of our current scope.

The Yahoo! Mail web service API8 is a full-featured interface to Yahoo! Mail. With it, you can build applications that display message summary information, parse message contents, manage folders, and even compose and send messages. The WSDL description (ymws.wsdl) of the Yahoo! Mail web service contains 21 operations to perform the tasks described before, 121 types, and 42 input/output messages (there is no use of fault messages). As stated before, the mapping process consists of two phases the automated phase and the manual phase. In the case of mapping the Yahoo! Mail web service, the automated phase generates the corresponding OWL-S file and a list of the most related ontologies corresponded to each service data type. This phase takes less than 1 minute. In addition, the manual phase that is performed by the service provider to choose the most appropriate ontology for each data type, takes around 1 hour. In the absence of such a mapping algorithm, the translation of a complex WSDL document (such as the specification of the Yahoo! Mail web service) takes about a week of man time. Therefore, there is a significant saving in the time spent

VIII. CONCLUSION

In this paper, the problem of enabling web services discovery get solved. There propose a mapping algorithm that helps to facilitate the integration of the current conventional web services into the new environment of the Semantic Web. This has been achieved by extracting information from WSDL files and using it to create a new semantic description files using OWL-S. The proposed mapping algorithm represents a starting point to transform our proposed intelligent discovery mechanism into an applicable one. Without trying to benefit from the already available web services, any semantic discovery mechanism will face many problems to be widely used. So, the main task for this algorithm is to help the service providers to redescribe their WSs in a semantic manner. Experimental results show that the proposed algorithm has the potential to significantly reduce the time and efforts of the mapping process. Moreover, they show that the algorithm consideration of the standardization problem promises that it will have a positive impact on the discovery process as a whole.

REFERENCES


