ISSN:0975-9646

Dynamic Referencing of Web Services via Service Discovery and Natural Language Processing

Kanmani Sivagar
Assistant Professor,
Department of Computer Science and Engineering
SRM University,Tamil Nadu, India- 603203

Krishna Chaitanya
Department of Computer Science and Engineering
SRM University
Tamil Nadu, India- 600029

Abstract This paper proposes adding reference of Web services to project by using Web service discovery framework for finding semantic web services by using natural language processing. This framework allows user to search through set of semantic Web services in order to find a best match given by user. Search is done by using keywords which make end user need not have knowledge about semantic languages, which makes easy for the user to search web service. Techniques like part-of-speech, lemmatization, and word sense disambiguation are used in order to make match between keywords and semantic Web services. After determining sense from relevant words which are gathered from Web services and user input query a matching process takes place. Matching algorithm is used in order to find best match between user input and keywords.

Keywords- Natural Language Processing, Semantic Web Service, Web Service Discovery, WSMO, WSMO-Lite.

1. INTRODUCTION

With emergence of Web services and Service oriented architecture business process components have become more. Complex task are being completed by using Web services in SOA. In general Web services are described by Web pages containing information about their operations in natural language. All these web pages contain information in plain text with no machine interpretable structure and therefore cannot be used by machines to automatically process information about a Web services.

Semantic Web service description use ontology to describe about the behavior of Web services. Semantic described in ontology enable system to interpret what a Web service is doing. However this ontology is created by human and contains natural language which allows humans to understand the concepts but a system can only understand ontology. Therefore in order to extend the relationships Natural language processing techniques can be used.

When using ontology machine can discover and compose Web services automatically based on semantic which are defined. NLP technique can to overcome the ambiguity between ontology and service descriptions. End users must be able to discover Web services based on keywords which are written in human language. A discovery mechanism should be developed in such a way which gaps the bridge between keywords written in natural language and Web service descriptions.

In this paper we propose semantic web service discovery framework which helps users to search using keywords from web service description and then adding web service reference to project dynamically. Dynamic adding of web service reference to project helps users and developers to add their web services with ease and can invoke methods which they need. This way helps developers to add more web services and can check for available methods and can use the methods.

2. RELATED WORKS

This section reviews the languages and tools which are required for Web service discovery. Initially a compression is made between different types of Web service description languages in section 2.1. Since in this paper we are covering discovery of semantic Web services, three main semantic Web service description languages are described and comparison is made between each other in section 2.2. At last several approaches for discovering Web services are discussed in section 2.3

2.1 Service Description Languages

Web service discovery mainly depends on how Web services are described an overview of different types of languages which are used for Web service discovery is given first. These languages differ in models and formalisms which are used for describing Web services. These languages can be simple piece of plain text describing the Web service to large and complex semantic description of the Web services behavior by mean of ontology.

To make machine use Web service automatically they are commonly defined in languages such as Web service description layer, Web application description layer, and hRESTs. These languages are used to describe the bindings, operations, input and output, and the endpoints of a Web service. These description makes humans and application to understand where, when, and what a Web service is expecting from user on syntax level. In order to bridge gap between syntactical languages and semantically enriched languages middle layer languages were defined. Most of the Web service description languages use only the description written in languages such as WSDL.

2.2 Semantic Languages for Web Service Discovery

Below given table list the core difference between three semantic Web service description languages OWL-S, WSMO and WSMO-Lite [11]. These languages mainly

differ in syntax they have. WSMO [4] framework is used for describing Web services and it contains four entities such as Ontologies, Web Services, Goals and Mediators. Ontologies provide terminology which can be used by other WSMO elements. In general Web service describes the capabilities, interface, and internal working of Web service. Goals describe user desires. Mediators provide bridge between different Ontologies. WSMO uses language called WSML which contain powerful logic formulae to describe the different WSMO elements. It even contains grounding feature which link concepts with WSDL [2] data types so that automatic invocation can be done.

	OWL-S	WSMO	WSMO-Lite
Syntax	OWL	WSML	RDL/XML
Parts	Service profile, Service model, Service grounding	Ontology, Web service, Goal, Mediator	Ontology, Web service
Reasoning	Logic language	Logic and rule language	Logic language

Table 1

The main difference between OWS-L, WSMO, and WSMO-Lite.

For a simple semantic Web service description language WSMO-Lite was created. It is a lightweight set of semantics service description which is written in RDFS that can be used for annotations of various WSDL elements using SAWSDL annotation mechanism. WSMO-Lite only uses Ontologies and Web service descriptions but it doesn't contain any grounding information which makes it dependent of SAWSDL.

2.3 Service discovery engines

Web service discovery can be distinguished between two types of approaches for Web service discovery i.e. discovery which is based on clustering operation parameters on the one hand and rich semantics on other hand.

2.3.1 Service discovery based on clustering

One of the approaches for Web service discovery is by searching for similarities among different WSDL service descriptions [2]. Woogle is a Web service search engine which employs clustering techniques for grouping operations parameters for a given query. Operation parameter clustering techniques are also employed in Seekda which is used for extraction of service semantic from a WSDL file which enable run time exchange of similar and composable Web services. Seekda is a service finder framework which comes under platform of service discovery where information about various services is gathered from sources like web pages and blogs. This information is automatically added to a semantic model by using automatic service annotation.

2.3.2 Service discovery via rich semantics

Predefined Ontologies can be used as another approach for semantic Web service discovery. By using semantics similarities between Ontologies, related semantic Web service can be discovered. For identification of these similarities Mediation space can be used this space mediate on data-level as well as semantic-level for discovery of related semantic Web service according to ontology.

3. PROPOSED WORK

Semantic Web service discovery framework is based on keyword-based discovery process for searching Web services that are described using a semantic language [1]. The search mechanism incorporates NLP technique in order to build a match between a user input query and a semantic Web service description. Logical based specifications defined in Web service description are not taken into consideration, but definitions of concepts in other imported Ontologies are exploited.

Section 3.1 covers about architecture of SWSD framework and description of main components. Section 3.2 explains how SWSD reads semantic Web service description and which elements from these are used for discovery. Section 3.3 explains algorithm that is being used.

3.1 Framework architecture

Input given to SWSD framework consists of set of Web services that are described in semantic language. The description are analyzed which result in extraction of words that could represent context of the Web services. Now extracted words are disambiguated and multiple senses are attached to same words. Next disambiguated words are matched with the disambiguated words from the search query resulting in list of top matched Web services. Finally links which are obtained can be used for adding reference to project dynamically. Below figure 1 describe the architecture of SWSD framework. The process is divided into four major task i.e. Service Reading, Word Sense Disambiguation, Match making, and dynamic adding of reference. Service Reading is used for parsing a semantic Web service description [1], extraction names and non-functional description of used components. The WSD task is to determine the sense of set of words. Match making step determines similarity between the different sets of senses which used for showing list of service. Last, dynamic adding of reference is used for adding reference of extracted Web service link to the project.

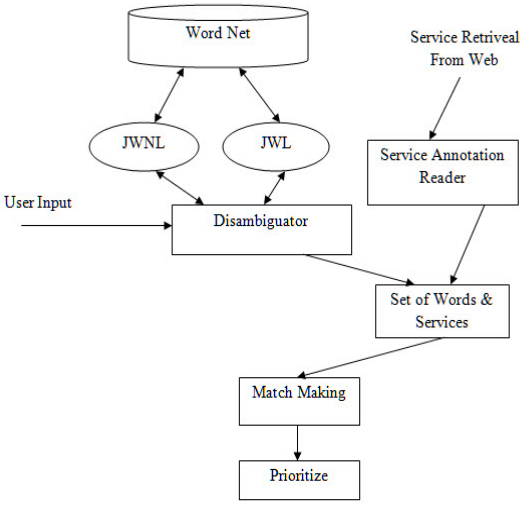

Figure 1. Proposed Architecture of SWSD

3.2 Semantic Web service reader

To enable search engine to look through Web service descriptions written in different languages it has to have different service description reader. These readers are different for each language. In semantically described Web service reader must be able to parser a description and extract concepts, attributes, and relations from WSMO-Lite etc. The first step in the process is searching semantically described Web service to implement readers for different languages and formats. Semantic Web service reader must be able to extract various elements out of a Web service. In WSMO Web service name and non-functional description of elements help in understanding the context of Web service. The non-functional description is written in natural language and contains a human description of the specified element. By extracting this description one can able to establish the context of the Web service operations. Before extracting words the description has to be parsed. Different languages need different parsers. For WSMO a WSML [3] parser like WSMO4J can be used.

3.4 Similarity matching Algorithm

In similarity matching user input sentence and semantic description is broken into letter and each individual letter is checked with each other and exact match words are given out. Based on the exact words maximum matched sentence are given out. For calculation of similarity only perfect matching items are used. For considering partial matches, similarity matcher makes use of a similarity-based approach for matching different sets or non-disambiguated words expressing close relatedness with values approaching 1 and non-relatedness with value 0.

$$\text{senseSim}(SS_u SS_w) = \sum_{S_u \varepsilon SS_u} \frac{\text{senseScore}(SS_u, SS_w)}{|SS_u| + |SS_u|} + \sum_{S_w \varepsilon SS_u} \frac{\text{senseScore}(SS_w, SS_u)}{|SS_u| + |SS_u|}$$

4. SEMANTIC WEB SERVICE DISCOVERY ENGINE

This section describes SWSD engine which is an implementation of SWSD approach and allows users to search for Web services on an existing repository by defining set of keyword. The steps required for the implementation of SWSD are closely related to the steps of SWSD framework. Section 4.1 introduces the SWSD engine as an implementation of SWSD framework. Section 4.2 describes how SWSD engine reads WSMO-Lite description. Section 4.3 explains how matching algorithm are used for computing.

4.1 SWSD engine

SWSD engine can only apply search on Web services which are annotated by using the WSMO framework as this framework is powerful and widely contain information about Web services. To read WSMO files, WSMO4J is used. WSMO4J is an API and a reference implementation for building semantic Web services and Semantic business process applications based on WSMO. By using this parser WSMO files can be parsed For WSD WordNet API is being used through two WordNet API's in order to find the sense between words [7] [8] [10]. For part-of-speech tagging the Stanford parser is used. The overall impletion is done is .net by converting java based files to dll files by using IKVM [5].

Fig 2 shows the user interface of SWSD engine User can input and then click on search button then after a search I made between user input and service description after obtaining service link by clicking on the link reference can be added to project dynamically.

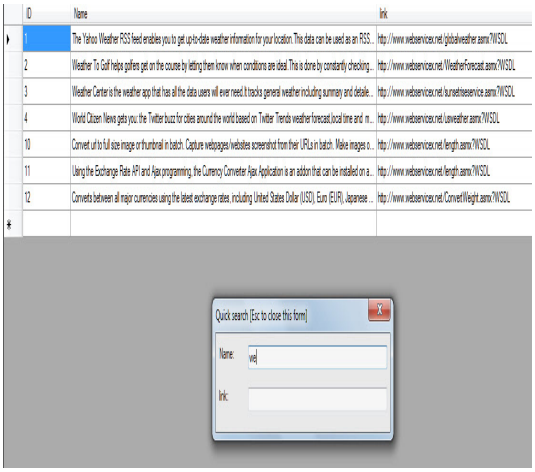

Figure 2. Result example.

4.2 Semantic Web service reader

For reading WSMO files WSMO4J API [6] is used which is used to extract different type of information from WSMO Web service and WSMO Ontologies. Fig 3 shows as example for WSMO Web service describing Google Search Web service. After reading WSMO Web service file found concepts are used to search in Ontologies to find description. Fig 5 shows an example WSMO ontology containing some concepts used in service description.

The names and non-functional description which are returned from readers will go through a NLP step. Nouns and verbs will be extracted from the non-functional descriptions using Stanford parser.

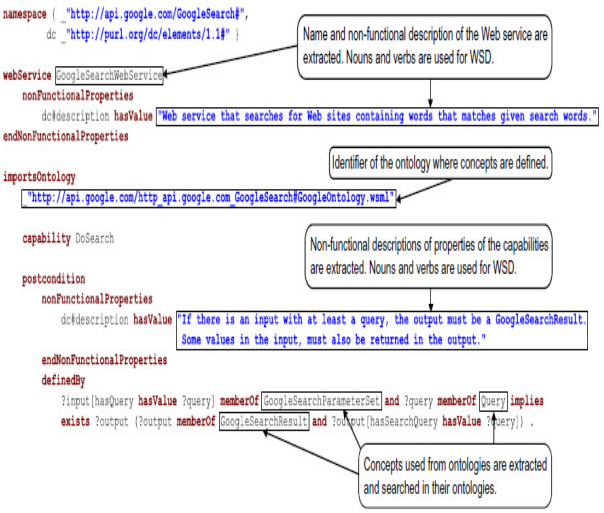

Figure 3. A WSMO Web service information extraction example.

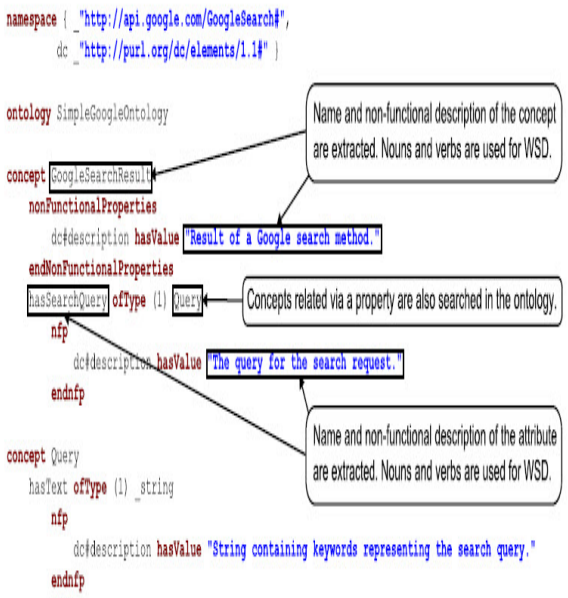

Figure 4. A WSMO ontology information extraction example

4.3 Impression matching

The word sets resulting from disambiguation process must be matched to get final similarity result. Information in a Web service description has different levels of importance in the matching process. All set of words belonging to a Web service will come out after disambiguation phase. Set of words coming from user input must be matched with several sets of words coming from Web service.

SUMMARY

In order to test performance of matching algorithm different types of input have been given and best results were obtained with top matches with similarity matching algorithm. Similarity matching algorithm has been best method for discovery similar Web service.

CONCLUSION

SWSD is used for discovering keywords BY SEARCHING Web services. Natural language processing technique and WordNet based similarity measures are used for matching keywords. By using an approach of similarity functions one for lexical and one for semantic similarities can be used for discovery of semantic Web services. Search for Web services is based on keyword. Matching is computed based on the similarity between the words in the user query and a Web service description. As a future work Web service search can be done by using voice commands. Where user will be given input query as a voice where it is converted to input string and NLP is done over the input and results will be obtained and will be back by system as voice.

REFERENCES

[1] Bener, A. B., Ozadali, V., & Ilhan, E. S. (2009). Semantic match maker with prediction and effect matching using SWRL. Expert Systems with Applications, 36(5), 9371–9377.

[2] Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). Web services description language (WSDL) 1.1 – W3C Note 15 March 2001.

[3] de Bruijn, J. (2008). D16 The WSML specification – WSML working draft 2008-08-08.

[4] de Bruijn, J., Bussler, C., Domingue, J., Fensel, D., Hepp, M., Keller, U., Kifer, M., Konig- Ries, B., Kopecky, J., Lara, R., Lausen, H., Oren, E., Polleres, A., Roman, D., Scicluna, J., & Stollberg, M. (2005). Web service modeling ontology (WSMO) – W3C member submission 3 June 2005.

[5] DERI Galway, 2008. Web service execution environment.

[6] EU IST, FIT-IT. (2008). WSMO4J API.

[7] Finlayson, M. (2012). JWI: The MIT Java Wordnet interface

[8] Greenwood, M. (2007). JWordNetSim.

[9] Navigli, R., & Velardi, P. (2005). Structural semantic interconnections: A knowledge based approach to word sense disambiguation. IEEE transactions on pattern analysis and machine intelligence (Vol. 27, pp. 1075–1086). IEEE Computer Society.

[10] The University of Sheffield. (2009). Pure java WordNet similarity library.

[11] Vitvar, T., Kopecky, J., & Fensel, D. (2007). WSMO-Lite: Lightweight semantic description for services on the web. In Fifth IEEE european conference on Web services (ECOWS 2007) (pp. 77–86). IEEE Computer Society.

[12] Walenz, B., & Didion, J. (2011). JWNL: Java WordNet library

[13] Jordy Sangers, Flavius Frasincar, Frederik Hogenboom, Vadim Chepegin. (2013) Semantic Web service discovery using natural language processing technique.