2007 IEEE/WIC/ACM International Conference on Web Intelligence

Web Service Discovery with additional Semantics and Clustering

Richi Nayak Bryan Lee
Faculty of Information Technology, Queensland University of Technology, Brisbane, Australia
r.nayak@qut.edu.au

Abstract

Due to the lack of semantic descriptions of the Web services, the search results returned by the service registries are effectively inadequate. This paper presents the Semantic Web services Clustering (SWSC) method that extends the semantic representation of services and groups the similar Web services in order to improve the service discovery. The empirical analysis shows the improvement in service discovery with the use of SWSC.

1. Introduction

The increasing usage of Web services on the Internet has led to much interest in the area of service discovery [1, 7, 10]. When a service provider creates a new service, it describes the service using service Description Language (WSDL). To make the service available to service consumers, the service provider registers the service in a Universal Description, Discovery and Integration (UDDI) registry by supplying the details of the service. When a service consumer wants to use a service, it queries the UDDI registry to find a service that matches its needs and obtains the access point of the service. Due to the lack of semantic descriptions of the Web services, the results returned by registries are effectively inadequate [10]. Many services are returned that are only able to partially complete the request while locating the specific service capabilities.

Researchers have worked in the direction of addressing the shortcoming of UDDI by finding relationships between search terms and service descriptions, and of WSDL to represent semantics while describing a service [1, 2, 3, 5, 7, 9, 10]. We present the Semantic Web Services Clustering (SWSC) method that extends the semantic representation of services for grouping the similar Web services in order to improve the service discovery. With the enhanced semantics and groupings as proposed in the SWSC method, searching within UDDI or Web service repositories becomes more intuitive. The method is three-fold. First, a combination of ontology languages is applied to the WSDL to add semantics to the structure. Second, a clustering algorithm groups the collections of heterogeneous services according to semantic similarity. Third, the user query terms are matched against the clusters to return the most suitable services. The most appropriate services according to the requirement are much easier to find with the related terms of the clusters.

2. SWSC: A method to Cluster Web Services Based On Semantics

2.1 The SWSC Stage 1: Pre-Processing

The SWSC method starts with analyzing the WSDL and checking its configuration and structure for further processing. Each Web service developer has a unique way of structuring the WSDL; hence it is critical to ensure that the input data is kept to the required standardization. The WSDL definition must exist for the specific service and made available publicly. The service should provide a number of operations (at least one or more) through one or more interfaces. The WSDL files should be tested to expose interfaces for services being available on the network. A fundamental characteristic "publish, locate and invoke capability" should also be tested and the SOAP message should also be monitored to check for ambiguity and correctness. The configuration analysis i.e. a high-level testing of Web services is conducted to provide the input and output dependencies between Web services, the invocation sequence and the functional descriptions. The structure and element analysis should be done to ensure that a WSDL document contains specific elements such as Types, Message, Operation, Binding, Port and Service to describe the service as collections of network endpoints or ports.

2.2 The SWSC Stage 2: Transformation

The WSDL information of Web services is transformed in a richer semantic representation language OWL-S (Ontology Web Language for Services) [5] using a series of methods. OWL-S provides a more abstract flow defined by its inputs, outputs, preconditions and effects. The transformed OWL-S information permits a comprehensive specification of various aspects of Web services that assists in efficient service discovery. OWL-S consists of three sub-ontologies known as the service profile, process flow and grounding. The service profile basically describes what the service does, its purposes and advertisement. It also constructs the service requests and possibly matchmaking. This contains most of the information relevant for service discovery. The process model basically discusses how the service works, to provide for service invocation and possibly composition. The grounding service is the key to building the entire process to the detailed specifications of message formats, protocols.

The WSDL2OWL-S tool [8] is used to convert the WSDL into OWL-S format, the service profile, process model and grounding. Additionally, we have used the details from the Web service advertisement sent by the service provider to enhance the OWL-S terms. The formal (W3C) OWL-S specification and properties [5] are followed to define the OWL-S with parameters derived from the WSDL definition and service advertisement details. One of the most significant portions of Semantic Web services is to take into consideration the specification of conditions and constraints, which consist of the preconditions and effects of a service. As mentioned in Stage 1, preconditions need to be satisfied before the consumer can be allowed the execution of the service. The post conditions, also known as effects, need to be realized upon successful execution of the service.

2.3 The SWSC Stage 3: Clustering

Each of the Web services listed with the UDDI registry, at this stage, will have an OWL-S file, WSDL and a description, which are the input dataset for grouping the similar Web services. The SWSC method gives different weight to each of these components - service description, OWL-S service profile, WSDL, OWL-S profile model and OWL-S grounding - in determining the similarity according to their significance in Web service discovery. A majority of times, the search terms are based upon the descriptions of the Web services. Accordingly, the SWSC method focuses more on the description of the Web service, its Service Profile and WSDL, rather than the Groundings and Process Model.

Terms are extracted from scanning the OWL-S, WSDL and the description of a Web service. The stop words including commonly occurring words such as a, the, is, was, there, etc and the OWL terms such as input, output, etc are removed from the list of terms. Each of the extracted terms is expanded using the WordNet ontology to enhance its semantics [6]. This increases the possibility of better searching and matching. Each of these components forms their own similarity matrix. The accumulative similarity coefficient between each pair of Web services is calculated based on the following formula where $Sim_{ws1\text{-}ws2}$, $SimDes_{ws1\text{-}ws2}$, $SimSerP_{ws1\text{-}ws2}$, $SimWSDL_{ws1\text{-}ws2}$, $SimPModel_{ws1\text{-}ws2}$, $SimGround_{ws1\text{-}ws2}$ are similarity coefficient measuring commonality based on similar terms between two services overall, service description, OWL-S service profile, WSDL, OWL-S profile model and OWL-S grounding respectively.

$$
\begin{aligned}
Sim_{ws1-ws2} &= w1 * SimDes_{ws1-ws2} + w2 * SimSerP_{ws1-ws2} \\
&+ w3 * SimWSDL_{ws1-ws2} + w4 * SimPModel_{ws1-ws2} \\
&+ w5 * SimGround_{ws1-ws2} -------(Eq1)
\end{aligned}
$$

w1, w2, w3, w4 and w4 are weight parameters assigning significance to each corresponding coefficient. These weight values are pre-set as w1 = 1.0, w2 = 0.3, w3 = 0.2, w4 = 0.1 and w5 = 0.1. This formula gives the terms in the description a much higher weight than others. The overall similarity coefficient is rounded off to 1.0 if the aggregated similarity value exceeds 1.0.

The values of the individual similarity coefficient are required to be calculated first. The Jaccard coefficient is used to calculate the similarity using the terms between two Web services [7]. This measure calculates the similarity based on the terms that are present in two Web services being compared. Terms that do not describe either of the Web services are regarded as unimportant. The Jaccard coefficient suits well in our situation as it is only practical to compare Web services based on their terms and not all the possible terms and service descriptions that exist in the entire search engine. Terms describing a service that are absent in the search sessions being compared are therefore insignificant. Let X and Y be the two different Web services. The similarity of these service descriptions is defined as:

$SimDes_{X\text{-}Y} = (T_{XY}) / (T_X + T_Y + T_{XY})$ ----*(Eq 2)*

where T_{XY} is the number of common terms used in describing X and Y, and T_X and T_Y are the number of terms used in X only and Y only respectively. Similarity coefficients for service profile, WSDL, process model and grounding between a pair of services are calculated and then combined as in equation 1. The whole process is repeated for each pair of services. Having obtained the similarity measure between the services, a *similarity* matrix is constructed for use as input for clustering. A similarity matrix is an *m* by *m* matrix containing all the pair wise similarities between the (*m* number of) Web services.

The hierarchical agglomerative clustering method, which is often used in information retrieval for grouping similar documents, is used [4]. This method uses a bottom-up strategy that starts by placing each Web service in its own cluster, and then successively merges clusters together until a stopping criterion is satisfied. This process results in arbitrarily shaped clusters as required in service discovery. The clusters (terms representing Web services) are stored in the UDDI registry database. The search function that the standard UDDI performs only returns the services that has the term matched, which is basic and insufficient. The SWSC method improves the search function by retrieving the best offers of services using the cluster matching. The SWSC method ranks the matched Web services and indicates the degree of relevance according to the term existence in clusters.

3. Empirical Evaluation

The effectiveness of the Semantic Web services Clustering (SWSC) method is shown by conducting three set of experimentations: (1) the basic keyword search based on using the WSDL information only; (2) the keyword search based on using the clusters that are derived from WSDL terms only; and (3) the keyword search using the the WSWC method. A total of 35 different Web services are used that are related to weather status (#7), computer shops (#7), car repairing (#5), food (#5), book stores (#4), location finder (#3), currency converter (#2) and computer applications (#2). Amongst these, 15 services are live services from the Internet and the remaining 20 are pseudo services for creating a variety of domains. These 20 services are created intentionally, that exhibit certain characteristics to test the rigidity of the methodology.

Experiment 1: Search Term Matching Using WSDL

This method of searching utilizes none of the semantics or ontologies or clusters. The term "cars" is searched and 1 match is found. However if the search term is changed to "automobile" or "vehicle", no match is found. This is because the search function only tries to match the term and not its semantics. The WSDL may not always contain terms that describe the nature of the Web service. If the user enters "automobile cars" then 1 match is found. The matched term is "cars" because it exists in the WSDL description. This basic experiment shows that the UDDI require flexibility to allow intuitive searching.

Experiment 2: Search Term matching with the clusters constructed based on WSDL only

Terms of the WSDL of each Web service are compared with other Web services to find the common groupings. The similarity matrix deriving the similarity between terms of WSDL of contained a low average of coefficients and most of them have come up as 0.0 whereby no similarity is detected. This shows that even with services of similar domains, WSDL is not sufficient enough to justify the similarity between them. This indicates that WSDL requires a form of ontology and semantics to be added onto its description to reveal the actual similarity coefficient.

Table 1 (columns under the experiment 2) shows the clustering results following this approach. The largest cluster group number 5 which has 9 services within has a lower similarity among its members. Cluster number 2 has a perfect 1.0 because all 6 of the services in this group are reporting weather status and use the same parameters in the WSDL. In fact, this group is developed by the same Australian service provider, hence the parameters and endpoints are similar. A comparison shows that the search term "books" returns no results using the WSDL keyword match only approach, whereas, the same search query of "books" used in this experiment, returns 6 matches of Web services (all relevant), including the related ones from clustering solution.

Experiment 3: Search Term matching with the SWSC method (Optimal Solution)

Much improvement can be seen in terms of the average internal similarity between objects within the cluster (4th and 5th columns in Table 1) as compared to results from the previous experiment (2nd and 3rd columns). However with only 6 clusters, it seems that the groups tend to squeeze into each other. For example, Web services related to the food domain correlate with the Web services on book sellers or finders because of the delivery commonality. In the 6 cluster solution results, they are all grouped into number 0. The results clearly show that there are insufficient clusters to fit the number of domains. The clustering solution with the desired number of 8 clusters shows the improvement (6th, 7th and 8th columns in Table 1). Indeed with 8 clusters now, the domains are segregated as required. Previously there were dissimilar objects sharing the same group. The segregation has relocated the objects into new clustering groups, which is where they belong. A comparison while using the search term "currency" shows that the keyword only (experiment 1) and WSDL only clustered solution (experiments 2) approaches return no results, whereas, the search using the optimal cluster solution produces 3 (relevant) matches of the currency conversion services.

Method	correctness	errors
Experiment 1	21%	5%
Experiment 2	43%	25%
Experiment 3	78%	15%

Table 2. Precision of the results

Over 50 types of search terms are used to test the accuracy of the proposed method. Table 2 shows the overall result listing the accuracy (percentage no of times the actual services were found) and error rate (percentage no of times the method produced the incorrect result). This shows that the SWSC method significantly improves the number of times the relevant services are found in comparison to approaches of key-words only or clusters without any semantic information. However the error rate is also increased for the SWSC method in comparison to key-word only approach due to the wider coverage of former one. Overall, the SWS method (experiment 3) outperforms the other two approaches.

4. Conclusions

The future of Web services greatly depends on their ability to automatically identify the Web resources and execute them for achieving the intended goals of user. The plain WSDL does not provide sufficient details to add semantics to its structure, as well as, no formal model has been proposed to assist the formal search using the UDDI registry, at the best stage of our knowledge. In this paper, the method "Semantic Web Services Clustering (SWSC)" has been presented, which acts as a UDDI layer between UDDI registry and the public internet. The primary motivations of this research are driven by the ongoing development of semantics and ontologies with their vast contribution to the Web services research. SWSC takes advantage of the OWL-S ontology and WordNet lexicon to enhance the description with semantics. Furthermore, SWSC utilises the clustering algorithm to group Web services into domains for enhancing searching. The similarity functions not only consider the matching of text but the context of the terms The context captures the semantic connections as similarity links between Web services.. Experiments have shown that SWSC improves the retrieval of the best offers of services to be returned with regular search terms.

5. References

1. Benatallah, B., Hacid, M., Leger, Alain., Rey, C.,& Toumani, F. (2005) On Automating Web Service Discovery. VLDB Journal. 14 (1):84-96,
2. Colgrave, J., Akkiraju, R., & Goodwin, R. (2004). *External Matching in UDDI* . ICWS'04
3. Dong, X., Madhavan, J., & Halevy, A. (2004). Mining structures for semantics *SIGKDD Explore., 6*(2), 53-60.
4. Karypis, G., Han, E. H., & Kumar, V. (1999). Chameleon: A hierarchical clustering algorithm using dynamic modeling. *Computer, 32*(8), 68-75.
5. Martine et al. OWL-S: Semantic Markup for Web Services, W3C Member Submission
6. Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography, 3*(4), 235-244.
7. Nayak, R. (2007) "Facilitating and Improving the Use of Web Services with Data Mining", Chapter 12 in "*Research and Trends in Data Mining Technologies and Applications"*, Ed: D. Taniar, pp: 309 -327.
8. Paolucci, M., Srinivasan, N., Sycara, K., and Nishimura, T. WSDL2OQL-S. http://www.daml.ri.cmu.edu/wsdl2owls/
9. Ulrich, K., Birgitta, K.-R., Mirco, S., & Michael, K. (2007). *DIANE: an integrated approach to automated service discovery, matchmaking.* WWW'07.
10. Wang, H., Huang, J. Z., Qu, Y., & Xie, J. (2004). Web services: Problems and Future Directions. *Journal of Web Semantics, 1*, 309-320.

	Experiment 2		**Experiment 3 (a)**		**Experiment 3 (b)**		
Cluster	**# Objects**	**Similarity**	**# Objects**	**Similarity**	**Cluster**	**# Objects**	**Similarity**
0	5	+0.558	9	+0.499	**0**	4	+0.872
1	6	+0.659	11	+0.657	**1**	11	+0.657
2	6	+1.0	6	+1.0	**2**	6	+1.0
3	6	+0.589	3	+0.977	**3**	3	+0.977
4	3	+0.690	2	+0.779	**4**	2	+0.779
5	9	+0.104	4	+0.539	**5**	5	+0.955
					6	2	+0.936
					7	2	+0.879

Table 1: Clustering Results (discussed in Section 3)