25th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises

A Novel Approach For Semantic Web Service Discovery

Randa Hammami
ReDCAD Laboratory
Sfax, Tunisia
randa.hammami@ieee.org

Hatem Bellaaj
ReDCAD Laboratory
Sfax, Tunisia
bellaaj@yahoo.fr

Ahmed Hadj Kacem
ReDCAD Laboratory
Sfax, Tunisia
ahmed.hadjkacem@fsegs.rnu.tn

***Abstract*—As the number of web services is increasing, finding the best service according to the users' requirements becomes a challenging task. Traditional method of web services discovery is based on keyword match. Due to this, many web services which are most relevant to the user request are left undiscoverable. Some other emergent approaches for discovering web services are based on semantics which improves the quality of the discovered web services in terms of relevance and satisfaction of user's need. In this paper, we briefly discuss the challenges facing semantic web services discovery and present different approaches for this process. We present also our proposed algorithm that aims to help users in the process of selection of the most appropriate semantic web service.**

Keywords- semantic web service; discovery; OWL-S; ontology; matchmaking

I. Introduction and Motivations

Though the large amount of information available on the Web is one of its main positive aspects, it also has a negative side: the increasing availability of Web Services (WSs) makes it difficult for users to find the WS they are looking for in order to satisfy their requirements. We refer to this process as service discovery which is the "whole process of retrieving services that are able to fulfill a need of a client" [1]. Reference [2] affirms that discovery is the most important task in the web service model because WSs are useless if they cannot be discovered. A large number of web service discovery methods have been proposed. The Universal Description Discovery and Integration (UDDI) has achieved positive results in the Web market, but it still has some limitations. First, the WS discovery process is mainly based on keywords matching. Second, WSDL (Web Service Description Language) documents describe only the functional and syntactic aspects of a service (operations, messages, data types...). Third, a service requester needs to spend lot of time in finding the appropriate and relevant WSs based on his needs which is one of the most lonely and time-consuming tasks. Thus, the UDDI service discovery mechanism itself does not provide fast and accurate service discovery but produces results that are coarse in nature.

In order to address these problems, Semantic Web Services (SWSs) have been proposed to automate WS discovery, to ensure data exchange between various systems and to provide the interoperability and reuse of information resources and the integration of data from various sources [3].

Semantic web languages rely on ontologies - a formal specification of a shared conceptualization [4] - to explicitly specify the content of the tags being used. In this paper, we adopt OWL-S as one of the major semantic web services description languages. OWL-S makes use of three different ontologies, namely: Service Profile, Service Model and Service Grounding [5]. Since we are studying SWS discovery, efforts should be concentrated on the Service Profile as it states what the WS does and describes its features. For example, the user may want to look for a service that finds a physician within a specific hospital in a particular region. Usually, this task must be performed by a human who uses a search engine to find a service, read the web page and execute the service manually to determine if it satisfies the constraints. With OWL-S, the information necessary for automatic SWS discovery is enabled in *Service Profile* advertisements in terms of service properties and capabilities.

The remainder of this paper is organized as follows: section 2 covers a brief overview of some SWS discovery approaches. In section 3 we propose a new approach to enhance semantic web services discovery. Section 4 provides an experimental evaluation. Finally, the paper has been concluded with section 5.

II. Related Works

To process user's queries, [6] proposes an OWL-S matchmaker that ranks services according to the calculated matching level between the outputs and inputs of the request against the outputs and the inputs of all the advertisements. If we assume that OutR represents the concept of an output of a request and OutA represents the concept of an advertisement, the algorithm distinguishes four degree of matching:

- *Exact*: If OutR and OutA are same or if OutR is an immediate subclass of OutA.
- *Plug-in*: If OutA subsumes OutR, then OutA is assumed to encompass OutR.
- *Subsume*: If OutR subsumes OutA.
- *Fail*: if there is no subsumption relation between OutA and OutR.

Reference [7] proposes to categorize the registry in offline mode based on the functionalities of the web services by using a clustering algorithm. Then the matchmaking algorithm matches the terms from the user request with the ontologies attached with each clusters, and then the semantic ranking is performed, so that highly ranked web services will be returned to the user.

Similarly, [8] relies on the standard classification of industry sectors marked in SWS description files to put forward a service discovery algorithm based on services classification to register the same category web services together and let them interconnect with the specific domain ontology. Then it calculates the similarity between user's request and services' description of the selected category.

Other research works make use of different semantic similarity formulas such as Google distance [9], Levenstein distance [10] or other measures [11] to express the similarity between different concepts of SWSs. However, these approaches browse all the SWSs present in the registry to perform their calculations, classify services according to their degree of match and display them to the user, which is a time-consuming task. Thus, we are suggesting a new matching algorithm in the next section.

III. MATCHING ALGORITHM

Our matchmaker is based on semantic similarity measures. The algorithm defines a more flexible matching mechanism that is double folded. When a request is submitted, the algorithm finds appropriate services by matching the outputs, inputs and description of the requested service against those of the published advertisements. Then, if any advertisement is matched, it is added to the list of matching SWSs to be presented to the client.

A. Publishing Phase

Since publishing a semantic web service is not a time critical task, we attempt to exploit this time to pre-compute the degree of match between the advertisements for possible future requests (see Table I). To perform this pre-computation, the matchmaker maintains a registry with already published SWSs and a similarity matrix M such as $M(i,j) = D_{sim}(S_i, S_j)$ where Si and Sj are registered SWSs and D_{sim} refers to the similarity degree between them. Once there is a new SWS to add to this registry, the matchmaker performs similarity calculation between the published SWSs and the new SWS then inserts it into the matrix M. Consequently, each line (i) of M contains the similarity degrees of the corresponding service S_i and the rest of registered services sorted in descending order (i.e. from the most similar to S_i to the lowest one(s)). M is also updated when a SWS is deleted from the registry.

We notice here that the publishing time of a new SWS is equal to the sum of the similarity degrees calculation' time and the time spent to update the matrix M (1).

$$Time_{publish} = n * Time_{similarityDegreeCalculation} + Time_{MatrixUpdate} \quad (1)$$

B. Querying Phase

Since most of the matching information is pre-computed at the publishing phase, the matchmaker's query phase is reduced to some calculations and simple lookups in the similarity matrix (see Table II). The idea here is to save time when processing a query (Q) by looking for one SWS (A) that is similar to the requested service based on the similarity threshold (T) defined by the user. Then, we calculate the similarity degree between Q and SWSs which are similar to A (extracted from the similarity matrix M) as we presume that if Q is similar to A and B is similar to A then Q could be similar to B to a certain degree (we rely on the similarity properties detailed in [12], [13] and [14] to make this assumption). Thus, instead of browsing all the SWS in the registry one by one, the required time to process a user query is reduced to the time of similarity degree calculation between the requested service and some of the registered SWSs (2).

$$Time_{query} \leq (n * Time_{similarity\ degree\ calculation}) \quad (2)$$

TABLE I. ALGORITHM FOR PUBLISHING A NEW SWS

```
Inputs :
    • n = number of published SWSs
    • R = the registry of SWS
    • $M_{n,n}$ = similarity matrix
    • $S_{pub}$ = SWS to be published in the registry R

Outputs :
    • n = updated number of published SWSs
    • $M_{n,n}$ = updated similarity matrix M
    • $S_{pub}$ = published SWS

Algorithm:
Begin
    1. for each $S_i$ in R do
           $M(n+1, i) \leftarrow D_{sim}(S_{pub}, S_i)$
       end for
    2. Sort in decreasing order the line (n + 1) of the matrix M
       according to the similarity degrees.
    3. $n \leftarrow n+1$
    4. Publish $S_{pub}$

End
```

TABLE II. DISCOVERY ALGORITHM

```
Inputs :
    • n = number of published SWSs
    • R = the registry of SWS
    • $S_{req}$ = SWS representing the user's query
    • $M_{n,n}$ = Similarity matrix such as $M(i,j) = D_{sim}(S_i, S_j)$ where $S_i$
      and $S_j$ are SWSs
    • T = a similarity threshold chosen by the user such as: $T \in [0..1]$
Outputs :
    • {Ret} = set of selected SWSs that are similar to the request $S_{req}$
      of the user.
Algorithm:
Begin
    1. {Ret} $\leftarrow$ Ø ; {Pot} $\leftarrow$ Ø ; {AlreadyVisited} $\leftarrow$ Ø
    2. Repeat
           Calculate $D_{sim}(S_{req}, S_i)$ where $i \in 1..n$
       Until ($D_{sim}(S_{req}, S_i) >= T$) or (i>n)
    3. If (i>n) then
         i.  ask the user to reduce the similarity threshold T
         ii. go to 2
       else
           go to End
       end if
    4. {Ret} = {Ret} + $S_i$

    5. Repeat
         i.   $S_j \leftarrow$ the SWS that is the most similar to $S_i$ (extracted from
              the matrix M)
         ii.  Calculate $D_{sim}(S_{req}, S_j)$
         iii. if ($D_{sim}(S_{req}, S_j) >= T$) then
                  {Ret} $\leftarrow$ {Ret} + $S_j$
```

```
                    {Pot} ← {Pot} + $S_j$
                    j ← j+1
            end if
        until ($D_{sim}(S_{req}, S_j) < T$)
6.      { AlreadyVisited } ← { AlreadyVisited } + $S_i$
7.      while ({Pot} != Ø) do
    i.      For each $S_i$ in {Pot} do
                repeat
            o       $S_j$ ← the SWS that is the most similar to $S_i$ (extracted
                    from the matrix M)
            o       Calculate $D_{sim}(S_{req}, S_j)$
            o       if ($D_{sim}(S_{req}, S_j) >= T$) and ($S_j \notin$ {AlreadyVisited })
                    then
                    ▪       {Ret} ← {Ret} + Sj
                    ▪       {Pot} ← {Pot} + $S_j$
                    ▪       j ←j+1
             else
                    j ← j+1
                    end if
            until ($D_{sim}(S_{req}, S_j) < T$)
        end for
    i.      { AlreadyVisited } ← { AlreadyVisited } + $S_i$
    ii.     {Pot} ← {Pot} - $S_i$
    iii.    End while
End
```

IV. EXPERIMENTAL EVALUATION

We implemented a prototype of both of the publication and discovery algorithms using JAVA programming language and the WordNet Similarity for Java (WS4J) API to calculate semantic similarity. We selected 70 semantic Web services dealing with the medical care from the OWLS-TC v4 dataset to test and evaluate our prototype. Since time is an important indicator to show scalability properties of our proposed methods, we focused on it for these experiments.

Although publishing a SWS is not a time-critical task, Fig.1 shows that time spent to publish a SWS stays reasonable when advertisements' number increases.

Fig. 2 depicts the scalability of the matchmaking algorithm in terms of query response time (it includes query pre-processing time). It scales well even when we reach an important services number.

V. CONCLUSION AND FUTURE WORK

Along with the fast emergence of large sets of services, one of the most difficult problems arises: how to discover WSs by users who are generally not technically experienced? In this paper, we have proposed a SWS discovery algorithm since SWSs promise great potential gains. We also tested our prototype in terms of execution time: it shows a promising improvement of the performance of SWS matchmaking. Thus, we'll conduct further experiments and hopefully, we'll prove that the proposed matchmaker can find the set of requested services in a short time and with a good precision and recall.

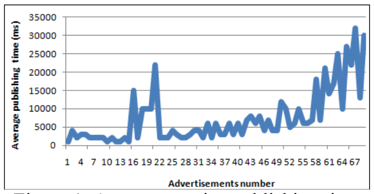

Figure 1. Average service publishing time

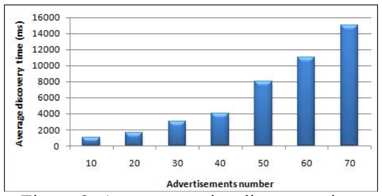

Figure 2. Average service discovery time

REFERENCES

[1] Küster, U., Lausen, H., & König-Ries, B. (2008). Evaluation of semantic service discovery—a survey and directions for future research. In Emerging Web Services Technology, Volume II (pp. 41-58). Birkhäuser Basel.

[2] Ngan, L. D., Kirchberg, M., & Kanagasabai, R. (2010, July). Review of semantic web service discovery methods. In Services (SERVICES-1), 2010 6th World Congress on (pp. 176-177). IEEE.

[3] Kogalovsky, M. R. (2013). Metadata in computer systems. Programming and Computer Software, 39(4), 182-193.

[4] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge acquisition, 5(2), 199-220.

[5] OWL-S: Semantic Markup for web Services http://www.w3.org/Submission/OWL-S/ (last visited in: 26/02/2016)

[6] Srinivasan, N., Paolucci, M., & Sycara, K. (2005). An efficient algorithm for OWL-S based semantic search in UDDI. In Semantic web Services and web Process Composition (pp. 96-110). Springer Berlin Heidelberg.

[7] Kumar, S. N., Pabitha, P., & Ahamed, A. K. (2013, November). web Service Discovery Based on Semantic Description. In Cloud & Ubiquitous Computing & Emerging Technologies (CUBE), 2013 International Conference on (pp. 199-203). IEEE.

[8] Lu, G., Wang, T., Zhang, G., & Li, S. (2012, June). Semantic web services discovery based on domain ontology. In World Automation Congress (WAC), 2012 (pp. 1-4). IEEE.

[9] Raj, J. R., & Sasipraba, T. (2012). web service discovery based on computation of semantic similarity distance and Qos normalization. Indian journal of computer science and engineering, 3(2), 235-239.

[10] Chen, L., & Xu, L. (2010, June). A framework for web service discovery based on ontology similarity. In Service Oriented System Engineering (SOSE), 2010 Fifth IEEE International Symposium on (pp. 197-201). IEEE.

[11] Budanitsky, A., & Hirst, G. (2001, June). Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In Workshop on WordNet and Other Lexical Resources (Vol. 2, pp. 2-2).

[12] Belanche, L. A. (2012). Understanding (dis) similarity measures. arXiv preprint arXiv:1212.2791.

[13] Tran, D. Q., & Nguyen, M. H. (2012). A mathematical model for semantic similarity measures. South-East Asian Journal of Sciences, 1(1), 32-45.

[14] Hau, J., Lee, W., & Darlington, J. (2005, May). A semantic similarity measure for semantic web services. In Web Service Semantics Workshop at WWW (pp. 10-14).