Semantic Web Service Composition and Discovery using OWL-S Descriptions

K.Vivekanandan¹, B.Muruganantham², D.Venkatesh³
¹Professor, CSE, Pondicherry Engineering College, Puducherry
²,³Computer Science and Engineering, SRM University Kattankulathur, Tamil Nadu

venki.08512@gmail.com
bmuruganantham@gmail.com

Abstract: Web Service Discovery can be described using OWL-S. It follows web service discovery model because it has conceptual and trivial semantic web service descriptions using service profile ontology of OWL-S. The Web Service Discovery model uses Web Service matchmaking algorithm which enlarges matching techniques used in Structural Case Based Reasoning, allowing the retrieval of Web services based on the formalional information of OWL ontologies. The abstract description for composition capabilities is done for returning Web service composition. Comparision of Semantic-Profile and Semantic-MX systems is done and their performances are also calculated.

Keywords – Web service discovery, conceptual descriptions, Semantic-Profile, Semantic-MX, formalional information

1. INTRODUCTION

WEB services have brought a communication revolution in heterogeneous domains where the efficient collaboration among different parties is important, such as in e-commerce and e-business. However, the increasing use of Web services has raised new challenges, such as the automated Web service discovery. Web is continuously enriched with Web services and it is transformed from a Web of documents into Web of documents and services. The problem that arises is how a human or an agent could be assisted during service selection. The XML representation of Web services (WSDL [1]) guarantees syntactic interoperability but it is unable to semantically describe services. Semantic Web services (SWSs [2]) aim at making Web services machine understandable and utilizing Semantic Web technologies for Web service annotation and processing. The idea is to provide ontology-based descriptions of Web services that could be processed by ontology reasoning tools. In that way, intelligent agents would be able to automatically understand what a Web service does and what it needs in order to perform a task.

2. BACKGROUND AND MOTIVATION

2.1 Semantic annotation of web services

Web service discovery can be defined as the problem of locating suitable Web services to fulfill a given objective. In the SWS paradigm, discovery is performed over semantic descriptions of Web services. WSMO-DF and OWL-S SP are two frameworks that regulate the way descriptions should be defined.

2.2 The WSMO Discovery Framework

WSMO-DF is based on the WSMO framework for Web service discovery. In WSMO-DF, a Web service is a computational entity, which is able, by invocation, to achieve a goal. A service, in contrast, is the actual value provided by this invocation [3]. Therefore, there are abstract Web service and concrete service descriptions. The former describes Web services in terms of their abstract functionality, whereas the latter contains a more detailed information about the service. For example, a hotel offers an abstract service for booking rooms, and requesters provide concrete descriptions of their requirements, for example, number of rooms, date, etc. In accordance to the distinction between Web service and service, WSMO-DF follows two levels of abstraction during Web service discovery, based on Light or Rich semantic descriptions. In the former, Web services are represented as Complex Concepts (CCs), mapping them in domain.
ontologies as a whole, and the matchmaking examines the subsumption relationships of CCs. The latter is the most fine-grained level, where Web services are modeled in terms of state transitions, obligations of requesters, etc.

2.3 The OWL-S Service Profile Ontology

OWL-S is an OWL ontology [10] that offers the conceptual model for semantically annotating Web services. The modeling is performed based on four upper ontologies, namely Service, SP, Service Process, and Service Grounding. In brief, the SP provides the information needed for an agent to discover a service (advertisement). An advertisement contains descriptive information, such as the service name, and information about the provider. It describes also the functional properties of the service, that is, inputs, outputs, preconditions, and effects, and nonfunctional properties, such as quality. The Service Process and Service Grounding provide information for an agent to make use of a service. Each advertisement can be either a direct instance of the OWL-S Profile concept or it can be defined based on a Profile subclass hierarchy. The Profile-based Web service discovery involves the procedure of matchmaking service requests and advertisements, both represented as Profile instances.

Example. Table 1 depicts four Web services advertisements using OWL-S SP approaches. The a1 advertisement is classified in the Order class and requires a title and an account as inputs in order to return a book that can be sent to Greece. Similarly, the a2 advertisement is classified in the Order class and requires a title and an account in order to return a magazine that can be sent to Greece. The a3 advertisement is also classified in the Order class and requires a title and an account in order to return a magazine that can be sent to UK. Finally, the a4 advertisement is classified in the Search class and returns a book based on the title. The above characteristics are expressed in the CC model by defining appropriate complex classes that describe services as a whole, whereas in the OWL-S SP model, each advertisement is expressed as an instance of the appropriate Profile subclass.

<table>
<thead>
<tr>
<th>TABLE 1 WEB SERVICE DESCRIPTION EXAMPLES</th>
</tr>
</thead>
<tbody>
<tr>
<td>OWL-S SERVICE PROFILE INSTANCES</td>
</tr>
<tr>
<td>1) a1 : Order,(a1,Title) : hasInput,</td>
</tr>
<tr>
<td>(a1, User) : hasInput, (a1,Book) :</td>
</tr>
<tr>
<td>hasOutput, (a1, gr) : to</td>
</tr>
<tr>
<td>2) a2 : Order,(a2,Title) : hasInput,</td>
</tr>
<tr>
<td>(a2, User) : hasInput, (a2, Magazine) :</td>
</tr>
<tr>
<td>hasOutput, (a2, gr) : to</td>
</tr>
<tr>
<td>3) a3 : Order,(a3,Title) : hasInput,</td>
</tr>
<tr>
<td>(a3, User) : hasInput, (a3, Magazine) :</td>
</tr>
<tr>
<td>hasOutput, (a3, uk) : to</td>
</tr>
<tr>
<td>4) a4 : Search,(a4,Title) : hasInput,</td>
</tr>
<tr>
<td>(a4, Book) : hasOutput</td>
</tr>
</tbody>
</table>

2.4. SCBR Similarity Metrics

In SCBR, both cases and queries are represented as objects, enhancing the typical attribute-value representation of the traditional CBR with domain knowledge.

**Definition 1.** “An Object O is a triple (ID,C,P) where ID is the unique identifier of the object, C is the object class type, and P is a set that contains attribute-value pairs of the form (p,V), where p is an attribute and V a set of values.”

The domain model is represented as a class hierarchy and the objects are initialized with a single class type and property-value definitions. For example, let A <= B denote that class A is subclass of class B; p ∈ Att(A) denote that the attribute p is defined in class A , o.p denote the set of values of object o for property p, that is, the set V , and o -> A denote the class type of object o. Let three classes A,B, and D where A <= B and B <= D. If o -> A, then o is also an object of B and D, due to inheritance. Furthermore, let pA and pB be two attributes, where pA ∈ Att(A) and pB ∈ Att(B). If o -> A, then both expressions o.pA and o.pB are valid, since attributes are inherited to subclasses.
this way, every object encapsulates domain knowledge, regarding class relationships and property-value definitions, which is used for matching cases and queries through the interclass and intraclass similarity metrics.

2.4.1 Intraclass Similarity

The intraclass metric defines the similarity of two objects in terms of the values in their common attributes, based on two value matching functions: the Vs function for simple values, for example, integers, strings, etc., and the Vr function for relational values, that is objects.

Let two objects OA and OB and their common attribute p. The partial intraclass similarity Sp for the property p is defined as

\[
SP(OA,OB) = \begin{cases} 
Vs(ID_{OA}.p,ID_{OB}.p), & \text{if } p \text{ is simple} \\
Vr(ID_{OA}.p,ID_{OB}.p), & \text{if } p \text{ is relational}
\end{cases}
\]

The overall intraclass similarity of two objects OA and OB is defined by aggregating their partial similarities.

**Definition 2.** Let two objects OA and OB and their set T of their common attributes. The intraclass similarity is defined, with respect to an aggregation function

\[
S_{\text{intra}}(O_A,O_B) = \Theta SP(O_A,O_B)
\]

2.4.2 Interclass Similarity

The interclass metric captures the hierarchical relationship of two object class types, based on a hierarchical matching function H that denotes the similarity of two objects in terms of their class types.

**Definition 3.** Let two objects OA and OB. Their interclass similarity is defined, with respect to a hierarchical matching function H, as

\[
S_{\text{intra}}(O_A,O_B) = H(C_A,C_B)
\]

The overall similarity of the two objects is defined by aggregating their intraclass and interclass similarities.

**Definition 4.** Let two objects OA and OB. Their similarity S is defined, with respect to an aggregation function as

\[
S(O_A,O_B) = \phi [S_{\text{intra}}(O_A,O_B), S_{\text{inter}}(O_A,O_B)]
\]

2.5 OWL-S Profile Metrics

In this section, we describe the DLH and DLR Profile aware similarity metrics, extending the intraclass and interclass SCBR metrics to an ontology environment, and enhancing them with DL reasoning. First, we introduce the notion of the object specification for representing advertisement and query instances in our framework.

**Definition 5.** An object specification is a quintuple \((ID,C,I,O,NF)\), where ID is the Profile instance identifier, C is the set of the most specific concepts to which ID belongs, I and O are the sets of I/O annotation concepts, respectively, and NF is the set of nonfunctional property-value pairs. We refer to an advertisement instance as an A specification and to a query instance as a Q specification. In this way, the Profile instances of Table 1 are represented as

\[
\begin{align*}
Q &= (a1, \{\text{Order}\}, \{\text{Title}, \text{User}\}, \{\text{Book}\}, \{(\text{to}, \text{gr})\}) \\
A_2 &= (a2, \{\text{Order}\}, \{\text{Title}, \text{User}\}, \{\text{Magazine}\}, \{(\text{to}, \text{gr})\}) \\
A_3 &= (a3, \{\text{Order}\}, \{\text{Title}, \text{User}\}, \{\text{Magazine}\}, \{(\text{to}, \text{uk})\}) \\
A_4 &= (a4, \{\text{Search}\}, \{\text{Title}\}, \{\text{Book}\}, \{\})
\end{align*}
\]

We approach the SWS discovery problem as the procedure of determining the similarity of an \(A(ID_a,C_a,I_a,O_a,NF_a)\) and \(Q(ID_q,C_q,I_q,O_q,NF_q)\) specification, based on three levels of similarity as follows:

1. **Taxonomical Similarity (TS).** It is computed over the \(C_a\) and \(C_q\) sets of an A and Q specification and denotes their similarity in terms of their
taxonomical categorization in a Profile subclass hierarchy.

2. Functional Similarity (FS). It is computed over the input (Ia and Iq) and output (Oa and Oq) sets of an A and Q specification (signature similarity).

3. Nonfunctional Similarity (NFS). It is computed over the values of the common nonfunctional properties of an A and Q specification.

2.5.1 The DLH Metric

The DLH metric represents the similarity of two ontology concepts in terms of their hierarchical position. It depends on a concept similarity function S and a set F of hierarchical filters. In the following, we assume that S(A,B) denotes the similarity of two concepts A and B, with respect to the function S, and that S(A,B)ε[0..1] with 1 denoting absolute match.

We generalize the A = B relation to a set of filters F and define that concept A matches concept B, with respect to a filter set F, if and only if there is at least one filter in F such that A = B, that is

\[ A \sim_{f} B \Leftrightarrow \exists f \in F: A \sim_{f} B \]

Definition 6. Let two concepts X and Y. Their DLH similarity is the normalized value to [0..1] defined, with respect to a concept similarity function S and a hierarchical filter set F, as

\[ DLH(X,Y,Z) = \begin{cases} S(X,Y) & \text{if } X \sim_{f} Y, \\ 0 & \text{otherwise.} \end{cases} \] (1)

We generalize (1) on two sets SA; SB of concepts as

\[ DLH_{set}(SA,SB,F) = \frac{\sum_{B \in SB} \max_{A \in SA} S(A,B)}{|SB|} \] (2)

Intuitively, for each concept B E(belong) SB, there should be at least one concept A E SA relevant to B, with respect to the filter set F. Otherwise, DLHset returns 0 (absolute mismatch). The overall DLHset similarity is computed as the mean value of the sum of the maximum DLHs for each concept B, since each B may have more than one relevant concepts in SA.

The Taxonomical Similarity denotes the similarity of two specifications in terms of their concept membership sets Ci, and therefore, it is equal to their DLHset similarity.

Definition 7. The taxonomical similarity between A and Q specifications is defined as the DLHset similarity of their Ca and Cq sets, that is,

\[ TS(A,Q,F_T) = DLH_{set}(Cq,Ca,F_T) \] (3)
where $F_T$ is the set of the hierarchical relationships that we allow to exist among the concepts of the Ca and Cq sets.

2.5.2 The DLR Metric

The DLR metric denotes the similarity between A and Q specifications in terms of the values in their common properties. It is defined in terms of the Functional (FS) and Nonfunctional (NFS) similarities and a Web service filter $W_f$ that we analyze in the following.

**Definition 8.** Let two specifications A and Q. Their DLR similarity is the pair $(FS, NFS)$ of their Functional and Nonfunctional similarities, with respect to the Web service filter $W_f$, that is,

$$DLR(A, Q, W_f) = (FS(A, Q, W_f), NFS(A, Q)).$$

2.6 Functional Similarity

The FS is based on the DLHset similarity of the I/O sets of two specifications, so as to ensure that 1) all the advertisement inputs are satisfied by the query inputs and 2) all the query outputs are satisfied by the advertisement outputs (signature matching).

**Definition 9.** The Functional Similarity between A and Q specifications is the normalized value to $[0..1]$ that is defined with respect to the Web service filter $W_f$, as

$$FS(A, Q, W_f) = \sqrt{DLHset(Iq, Ia, F_I).DLHset(Oa, Oq, F_O)}$$

(5)

We use the geometric mean, instead of the arithmetic mean, because a Web service should be excluded if either of its input or output similarity is zero. In order to control the different degrees of relaxation during I/O matching, the FS makes use of a Web service filter $W_f$ that defines the values of the hierarchical filter sets $F_I$ and $F_O$ in (5). More specifically, we define the Exact ($W_e$), Plugin ($W_p$), Subsume ($W_{su}$), and Sibling ($W_{si}$) Web service filters with the following relationships to the $F_I$ and $F_O$ filter sets:

1. $W_e \rightarrow F_I = F_O = \{e\}$. This is the strictest filter that allows two specifications to match only if they refer to the same or to equivalent concepts in their I/Os

2. $W_p \rightarrow F_I = \{e, p\} \cup F_O = \{e, su\}$. This is a more relaxed filter and intuitively denotes an A- specification that could be used instead of a Q specification. The rationale is that all the inputs of the advertisement should be equivalent or subclasses of the query inputs, and all the outputs of the query should be equivalent or superclasses of the advertisement outputs.

3. $W_{su} \rightarrow F_I = F_O = \{e, p, su\}$. This filter relaxes even more the matching criterion and the Advertisement is allowed to have 1) more general inputs than the query, 2) more general outputs than the query.

4. $W_{si} \rightarrow F_I = F_O = \{e, p, su, si\}$. This is the most relaxed filter, allowing also the existence of sibling relationships among I/O concepts.

2.7 Non-Functional Similarity

The NFS is defined in terms of two functions: the $dt$ function for computing the similarity of two data type values and the $ob$ function for computing the similarity of two object values.

$$NFS(A, Q) = \sum_{dt(Td, Td') \in Td matches} [dt(Ida.d, Idq.d) + ob(Ida.o, Idq.o)]$$

(6)

2.8 Overall Specification Similarity:

The overall similarity $sim$ of A and Q specifications is defined in terms of their TS, FS, and NFS similarities.

$$Sim(A, Q, F_T, W_f) =$$
The aggregation of the triple similarity into a single value is computed as the weighted mean sim of the three similarities according to user requirements, that is,

$$Sim = \frac{aTS+bFS+cNFS}{a+b+c}$$

Where a,b,c are normalized weights.

3. WEB SERVICE COMPOSITION

Dynamic composition of web services is a promising approach and at the same time a challenging research area for the dissemination of service-oriented applications. It is widely recognised that service semantics is a key element for the dynamic composition of Web services, since it allows the unambiguous descriptions of a service's capabilities and parameters. This paper introduces a framework for performing dynamic service composition by exploiting the semantic matchmaking between service parameters (i.e., outputs and inputs) to enable their interconnection and interaction. The basic assumption of the framework is that matchmaking enables finding semantic compatibilities among independently defined service descriptions. We also developed a common position algorithm that follows a semantic graph-based approach, in which a graph represents service compositions and the nodes of this graph represent semantic connections between services. Moreover, functional and non-functional properties of services are considered, to enable the computation of relevant and most suitable service compositions for some service request.

4. EXPERIMENTAL RESULTS

We tested OWLS-SLR and compared it to the OWLS-MXmatchmaker using the OWLS-TC with 1,007 OWL-S advertisements and 29 queries. We have chosen OWLS-MX since it is a well-known matchmaker, having been extensively tested on the OWLS-TC collection. Furthermore, it uses lightweight OWL-S SP descriptions like OWLS-SLR, it is able to incorporate the structural information of the domain ontologies through concept unfolding (nearest neighbors—NN). We used the M4 configuration of OWLS-MX as the best.

4.1 Loading and Query Response Time

The loading time involves the time needed to parse and process the advertisements and queries, whereas the query time involves the time needed to apply the matchmaking algorithm. OWLS-SLR depicts a considerably better loading and query response performance compared to OWLS-M4. OWLS-SLR loaded the data set in about 30 seconds, whereas OWLS-M4 needed more than 30 minutes. In OWLS-SLR, the UC-related distance is computed faster than the EC, since the latter requires the traverse of all the paths between the concepts. However, both configurations perform faster than OWLS-M4 that depicts a constant query response performance.

4.2 Precision and Recall

We used the relevance sets of the collection in order to perform precision and recall tests. Due to the fact that the collection defines only direct Profile instances, we created also a taxonomy-based collection in order to test the performance taking into account the TS. Note that OWLS-MX can handle only direct Profile instances. We have omitted the subsumed-by filter of OWLS-MX for presentation purpose. OWLS-M4 has, in general, better precision than OWLS-SLR in the collection without a Profile taxonomy, showing that the filter definitions that it follows, which are based on, fit better to the specific collection. However, by
performing domain-oriented discovery, OWLS-SLR outperforms OWLS-M4, justifying the advantage of a domain-oriented approach to SWS discovery. In most cases, we are interested in the first k results of a query.

5. CONCLUSION

In this paper, the OWL-S SP-aware framework for SWS discovery, using abstract and lightweight Web service descriptions. The main intention is to define a framework that can be used as a prephase in more fine-grained approaches that incorporate rich Web service descriptions, such as reconditions, effects, or state transitions. In this way, the complex and sophisticated algorithms would be applied on a smaller set of candidate Web service descriptions than the complete initial set. In an effort to enhance the instance-oriented SP discovery paradigm with the domain modeling capabilities found on CC approaches, we allow the existence of Profile taxonomies and Moreover, we defined a matchmaking algorithm that exploits the structural knowledge of ontologies, for example, sibling concept relationships, by considering advertisements and requests as objects and implementing concept (dis-) similarity measures. We presented also a comparison of OWLS-SLR to OWLSMX. The experiments have shown a considerably better performance on loading and querying of OWLS-SLR than that of OWLS-MX.

REFERENCES