A Framework for Semantic Web Services Discovery using Improved SAWSDL-MX

*Agushaka J. O and Junaidu S. B
Department of Mathematics, Ahmadu Bello University Zaria-Nigeria
*jagushaka@yahoo.com and sahalu@abu.edu.ng

Abstract **-** The increasing growth in the popularity of Web Services makes discovery of relevant Web Services a significant challenge. To this end, the central objective of this paper is to propose a framework that proffers solutions to some problems identified in the course of study of SAWSDL-MX, a hybrid matchmaker based on OWL-S MX, that uses both logic based reasoning and content based information retrieval techniques for services specified in SAWSDL description. As part of solution to the problems identified, non-functional specification like quality of service (QoS) was added to the discovery process and also, users are allowed to specify queries for services using terms in their own ontology. This is achieved by specifying a mapping from user ontology to domain ontology and also a QoS algorithm was developed and incorporated into the architecture of SAWSDL-MX to enable service discovery based on non-functional specification. An example is given in this paper to validate our framework. The implementation is an ongoing work.

Keywords: Semantic Web Services Discovery, SAWSDL-MX, OWL-S MX, Non-Functional Specification (QoS)

1. Introduction

Web applications are applications that are accessed via web browsers over a network such as the Internet or an intranet. Web has gone through many transformations starting from the first generation non-interactive content pages to the new generation Web Services. Web Services are the third generation web applications; they are modular, self-describing, self-contained applications that are accessible over the Internet Cubera et al (2001). In its simplest form, a Web Services is a class whose methods are callable remotely. Method calls to and responses from Web Services are transmitted via SOAP. Thus any client capable of generating and processing SOAP messages can use a Web Services regardless of the language in which the Web Services is written. Once a Web Service is deployed, other application (and other Web Services) can discover and invoke the deployed service. The major issue with the current Web Services is **Description.** Currently, WSDL contains information useful to humans not computer/applications i.e. the description lacks explicit semantic and this can lead to misinterpretation of meaning of description. As such, discovery which is a process of finding Web Services that matches request can't be automated. Currently, UDDI supports Web Services discovery based on keyword search and taxonomy based search which can be less effective than desired. The popularity of Web Services solution led to the proliferation of Web Services thereby making the keyword based matching of requests for services inadequate. Semantic Web technology gave hope to this problem however, the use of ontology to add meaning to Web Services description came with its own problem (ontology heterogeneity). Use of common ontology may solve this problem but individual users or communities of users are expected to query for services of interest to them using descriptions that are expressed using terms in their own ontologies. The need to specify a mechanism for mapping user ontology to the domain ontology used to describe services is essential. Also, the need to add non-functional specification to the discovery process cannot be overemphasized. Consider a scenario where a request for service satisfies all the users' requirements (functional) but the cost (non-functional) of invoking the service is overbearing. If non-functional specification is not included in the discovery process, this very service will be returned to the user who will unknowingly invoke and bear the consequences. These amongst others motivated this research. Semantic Web Services Discovery is based on matching semantically described goal descriptions (goal queries) with semantic annotations of Web Services (capability descriptions) (Walsh et al, 2002). Several capability-based semantic Web Services discovery solutions have been proposed in the literature (Patil et al, 2003; Duftler et al, 2001; Cardoso et al, 2002; DAML-S 0.7 Draft Release, 2002). A capability description annotates either the inputs or outputs of Web Services (Cardoso et al, 2002) or describes an abstract service capability (Paolucci et al, 2002; Duftler et al, 2001) and can be applied in the frame of both OWL-S (Patil et al, 2003) and WSMF/WSMO (Ankolenkar et al, 2002; Paolucci et al, 2002). The discovery process in the capability-based semantic discovery approaches can only happen on an ontological level, i.e., it can only rely on conceptual and reusable things. The greatest difficulty in a Web Services discovery mechanism is *heterogeneity* between services (Garofalakis et al, 2004). Heterogeneities include different operating platforms, different data formats, as well as heterogeneities of ontologies. Regarding ontology heterogeneities, semantic Web Services may use different ontologies or different ontologies description language such as OWL, DAML, RDF, and so forth to describe the services. There is also heterogeneity between semantic Web Services and non-semantic Web Services. Therefore, when developing a discovery system, these heterogeneities should

be borne in mind. UDDI service registry does not allow, as it is, to store any semantic information related to service declarations, as such most semantic Web Service discovery engines allows for services to be registered in their service database and only services that are registered with the engine are considered during matching. However, in the work of Pierre (2007), a WSDL-S to UDDI mapping was defined and a service publication and querying API named LUCAS (layer for UDDI compatibility with annotated semantics) is placed around it. Jyotishman et al (2005) proposes a framework for ontology-based flexible discovery of Semantic Web Services. The proposed approach relies on user-supplied, context-specific mappings from a user ontology to relevant domain ontologies used to specify Web Services. A description of how user-specified preferences for Web Services in terms of non-functional requirements (e.g., QoS) can be incorporated into the Web Services discovery mechanism to generate a partially ordered list of services that meet user- specified functional requirements. OWL-S/UDDI (Srinivasan et al, 2004)) matchmaker combines UDDI's proliferation into the Web Services infrastructure and OWL-S's explicit semantic description of the Web Services. Glue (www. swa.cefriel.it/Glue) is a WSMO compliant discovery engine that aims at developing an efficient system for the management of semantically described Web Services and their discovery. OWL-S MX by Matthias et al (2008) is a hybrid OWL-S semantic service matchmaking algorithm. It uses both logical based and content based retrieval techniques for Web Services discovery. The hybrid semantic service matching uses six different filters to calculate the degree of semantic match between the request and advertisement. The first four filters are purely logic based and the next two are hybrid, which use the IR similarity metric values. SAWSDL –MX by Kapahnke et al (2008) is an approach that does hybrid semantic Web Services discovery using SAWSDL based on logic based matching as well as information retrieval based techniques. It is significantly based on but a further refinement of OWL-MX. Our work seeks to extend the work of Kapahnke et al. In the next section, we give some of the problems we identified in the cause of our study of SAWSDL-MX and then proffer solutions to them and that serves as our extension of Kapahnke's approach. We extended SAWSDL-MX to allow user to specifying queries using descriptions expressed using terms in their own ontology. This is to be achieved by specifying mapping between user ontology to service ontology. We also included non functional specification e.g. quality of service in the discovery process of SAWSDL-MX. It is important to note that the implementation of this work is ongoing and that this is a framework that is validated using the example here given. The rest of the paper is organized as follows: service matching using SAWSDL-MX (that we extended) is given in section 2, section 3gives our proposed framework. Architecture of the framework is in section 4, an example of service discovery using our framework is in section 5, future work in section 6, section 7 is related work and finally conclusion and acknowledgements is given.

2. Service Matching with SAWSDL-MX

SAWSDL-MX was developed based on some assumptions. Details of which can be found in Kapahnke et al (2008). SAWSDL-MX performs logic-based, syntactic (text similarity-based) and hybrid matching of service against request. The request for service is given as a standard SAWSDL document. This approach is particularly based on OWLS-MX (Matthias et al, 2008) and WSMO-MX (Kaufer et al, 2006) for OWL-S and WSML respectively.

2.1. Logic-Based Operation Matching

SAWSDL-MX applies four matching filters of increasing degree of relaxation: Exact, Plug-in, Subsumes and Subsumed-by, which are adopted from OWLS-MX but modified in terms of an additional bipartite concept matching to ensure an injective mapping between offer and request concepts. Details on the filters can be found in (Kapahnke et al,2008).

2.2. Syntactic Operation Matching

SAWSDL-MX implements the same similarity measures as that of OWLS-MX, which are the Loss-of-Information, the Extended Jaccard, the Cosine and the Jensen-Shannon similarity measures. Also the architecture of SAWSDL-MX allows the integration of other text similarity measures such as those provided by SimPack which is also used in the iMatcher matchmaker (Kiefer et al, 2008). The weighted keyword vectors of inputs and outputs for every operation are generated by first unfolding the referenced concepts in the ontologies. The resulting set of primitive concepts of all input concepts of a service operation is then processed to a weighted keyword vector based on TFIDF (Term Frequency and Inverse Document Frequency) weighting scheme, the same is done with its output concepts. The text similarity of a service offer operation and a request operation is the average of the similarity values of their input and output vectors according to the selected text similarity measure.

2.3 Hybrid Operation Matching

SAWSDL-MX combines logic-based and syntactic-based matching to perform hybrid semantic service matching. There are different options of combination:

- A compensative variant uses syntactic similarity measures in situation where the logic-based filters don't apply with respect to logic-based false negatives. It helps to improve the service ranking by re-considering them again in the light of their computed syntactic similarity.

- An integrative variant deals with problems concerning logic-based false positives by not taking the syntactic similarity of concepts into account only when a logical matching fails, but as a conjunctive constraint in each logical matching filter.

SAWSDL-MX inherited the compensative variant from OWLS-MX.

2.4. Limitations of SAWSDL-MX

We identified the following limitations amongst others:

- Users are not afforded flexibility to specify queries for services of interest to them using descriptions that are expressed using terms in their own ontologies i.e. ontology heterogeneity problem still persists.
- It lacks support for service selection based on non functional specification of offered or registered services

To address these limitations, we propose solutions that serve as extension and improvement on SAWSDL-MX. These extensions are:

1. We added a component to SAWSDL-MX that allows for specifying mappings from a user ontology to relevant domain ontologies used to specify Web Services. This is to take care of problem of ontology heterogeneity.
2. We also describe how user-specified preferences for Web Services in terms of non-functional requirements (e.g., QoS) can be incorporated into the Web Services discovery mechanism to generate a partially ordered list of services that meet user-specified functional requirements.

We give details of our proposal in the next section

3. Proposed Framework

This section describes our framework for ontology-based flexible discovery of Semantic Web Services which is an extension proposed for SAWSDL-MX.

3.1 Mapping User Ontology to Domain Ontology

Ontologies are the basis for shared conceptualization of a domain, and comprise of concepts with their relationships and properties (Gruber, 1993). In the work of Caragea et al (2004), a precise definition of ontology was given which we adopt in this work. They looked at ontology in terms of hierarchy, that if we define S to be a partially ordered set under ordering $\leq$. Then an ordering $\preccurlyeq$ defines a hierarchy for S if the following three conditions are satisfied:

(1) $x \preccurlyeq y \rightarrow x \leq y; \forall x, y \in S$. Then $(S, \preccurlyeq)$ is better than $(S, \leq)$,

(2) $(S, \leq)$is the reflexive, transitive closure of $(S, \preccurlyeq)$

(3) No other ordering exists that satisfies (1) and (2).

Then all ontology does is to associate orderings to their corresponding hierarchies. This means that given a set of concepts that define a domain, ontology associates ordering among the concepts i.e. they can be parsed as hierarchical tree with the nodes denoting the different concepts and the edges, relationship between these concepts. In order to make ontologies interoperable, so that the terms in different ontologies are brought into correspondence, we need to define mappings. These mappings are specified through

Interoperation Constraints. Which they define as:

Let $(H_1, \preccurlyeq_1)$ and $(H_2, \preccurlyeq_2)$, be any two hierarchies. Then the set of Interoperation Constraints (IC) the set of relationships that exist between elements from two different hierarchies. For two elements, $x \in H_1$ and $y \in H_2$, we can have one of the following Interoperation Constraints:- $x : H_1 = y : H_2$, $x : H_1 \neq y : H_2$, $x : H_1 \leq y : H_2$, *and*,

$x : H_1 \not\leq y : H_2$.

This means that given two ontologies defined as above, they interoperate if we can define a relation between the elements (which are actually concepts in the respective domains). Equality could mean they occur at the same level in the tree structure. Also, less than could mean a parent-child relation and so on.

The interoperation constraint gives a fine mapping from user ontology to the domain ontology used by the matchmaker.

3.2 Incorporating Non-Functional Requirements

Quality of Service can be defined as a set of non-functional attributes that may have significant effect on the service quality offered by a Web Services. Examples of these non-functional attributes include scalability, performance, availability etc. Different QoS attributes might be important in different applications and different classes of Web Services might use different sets of non-functional attributes to specify their QoS properties. For example, response time may be an important QoS criterion for a service which provides online voice conferencing, as opposed to, availability for a service which provides online reservation. As a result, we categorize them into:

Domain Independent Attributes: The domain-independent attributes represent those QoS characteristics which are not specific to any particular service (or a community of services). Examples include scalability, throughput etc.

Domain Dependent Attributes: domain-dependent attributes capture those QoS properties which are specific to a particular domain. For example in a gift packaging domain, gift decoration rating may be a QoS attribute.

This gives rise to a situation where a user might consider some non-functional attributes valuable for his/her purpose

(and hence, defined in the user ontology). These attributes are used to compose a quality vector comprising of their values for each **candidate service** (services that satisfy functional requirement of the user). These quality vectors are used to derive a quality matrix, $\mathbb{Q}$.

Doina et al, (2005) defined a quality matrix, $\mathbb{Q} = \{V(Q_{ij}); 1 \leq i \leq m; 1 \leq j \leq n\}$, which refers to a collection of quality of service attribute-values for a set of candidate services, such that, each row of the matrix corresponds to the value of a particular QoS attribute (in which the user is interested) and each column refers to a particular candidate service. In other words, $V(Q_{ij})$, represents the value of the i^{th} QoS attribute for the j^{th} candidate service. These values are obtained from the profile of the candidate service providers and mapped to a scale between 0 & 1 by applying standard mathematical maximization and minimization formulas based on whether the attribute is positive or negative. For example, the values for the attributes Latency and Fault Rate needs to be minimized, whereas Availability needs to be maximized. Also, to give relative importance to the various attributes, the users can specify a weight value for each attribute, which are used along with the QoS attribute values to give relative scores to each candidate service using an additive value function, f_{QoS}. Formally,

$$f_{QoS}(service_j) = \sum_{i=1}^{m} (V(Q_{ij}) \times weight_i)$$

$where, m$ is the number of QoS attributes in $\mathbb{Q}$ (Doina et al, 2005).

This incorporates non-functional requirements into the discovery process

4. Proposed Architecture

This section shows how the solutions identified with SAWSDL-MX1.0 are incorporated to have an improved and refined prototype. Basic aspects of SAWSDL-MX1.0 are maintained. The figure 1 gives the architecture.

The proposed new SAWSDL-MX consists of all the components of SAWSDL-MX1.0 as found in Kapahnke et al (2008). The new components include QoS matcher and Mapping User to Domain Ontology which in the next section. From the perspective of service providers, the new SAWSDL-MX allows the registration of SAWSDL Web Services offers along with their QoS ratings, at the service registry. For requesters, SAWSDL-MX provides an interface for submitting queries by means of a SAWSDL document specifying details about the desired service interface in the user's ontology. After which the user's ontology is mapped to the domain ontology of the matchmaker, as is in SAWSDL-MX, the domain ontology in our proposed framework is also OWL-based. After the service discovery process using the logic, syntactic or hybrid filters, QoS matcher of the proposed improved SAWSDL-MX takes as input those services returned by the filters and calculate their additive value function (f_{QoS}) scores and returns a ranked list of service offers with f_{QoS} scores greater than user specified threshold. This guarantees that the returned services match both functional and non-functional specification of the query

4.1 Mapping User to Domain Ontology:

This component takes the user ontology and maps it to the domain ontology of the matchmaker. Ontologies associate orderings to their corresponding hierarchies. For example, let $S = \{Food, ChineseFood, SeaFood\}$ (Figure 2). We can define the partial ordering $\leq$ on S according to an $is\text{-}a$ (or sub-class) relationship. For example, $SeaFood$ $is\text{-}a$ sub-class of $ChineseFood$, $ChineseFood$ $is\text{-}a$ sub-class of $Food$ and, also $SeaFood$ $is\text{-}a$ sub-class of $Food$. Besides, every class can be regarded as a sub-class of itself. Thus, $(S, \leq) =$ $\{(ChineseFood, ChineseFood), (SeaFood, SeaFood), (Food, Food)$ $(SeaFood, ChineseFood), (SeaFood, Food), (ChineseFood, Food)\}$, is the reflexive, transitive closure of the set: $(S, \prec) = \{(ChineseFood, Food),$ $(SeaFood, ChineseFood)\}$, which is the only hierarchy associated with $(S, \leq)$. We defined interoperability constraints to specify mapping from user to domain ontology. For example, let $O^{U}_{Chinese}$ be user ontology for specifying Chinese food different from $O_{ChineseFood}$ (domain ontology), assuming that the ontologies $O^{U}_{Chinese}$ and $O_{ChineseFood}$ associate $is\text{-}a$ orderings to their corresponding hierarchies, we can have the following interoperation constraints, among others- $Chicken : H^{U}_{Chinese} = Poultry : H_{ChineseFood}$,$Fish : H^{U}_{Chinese} = SeaFood : H_{ChineseFood}$,$Chicken : H^{U}_{Chinese} \neq Appetizer : H_{ChineseFood}$,and so on.

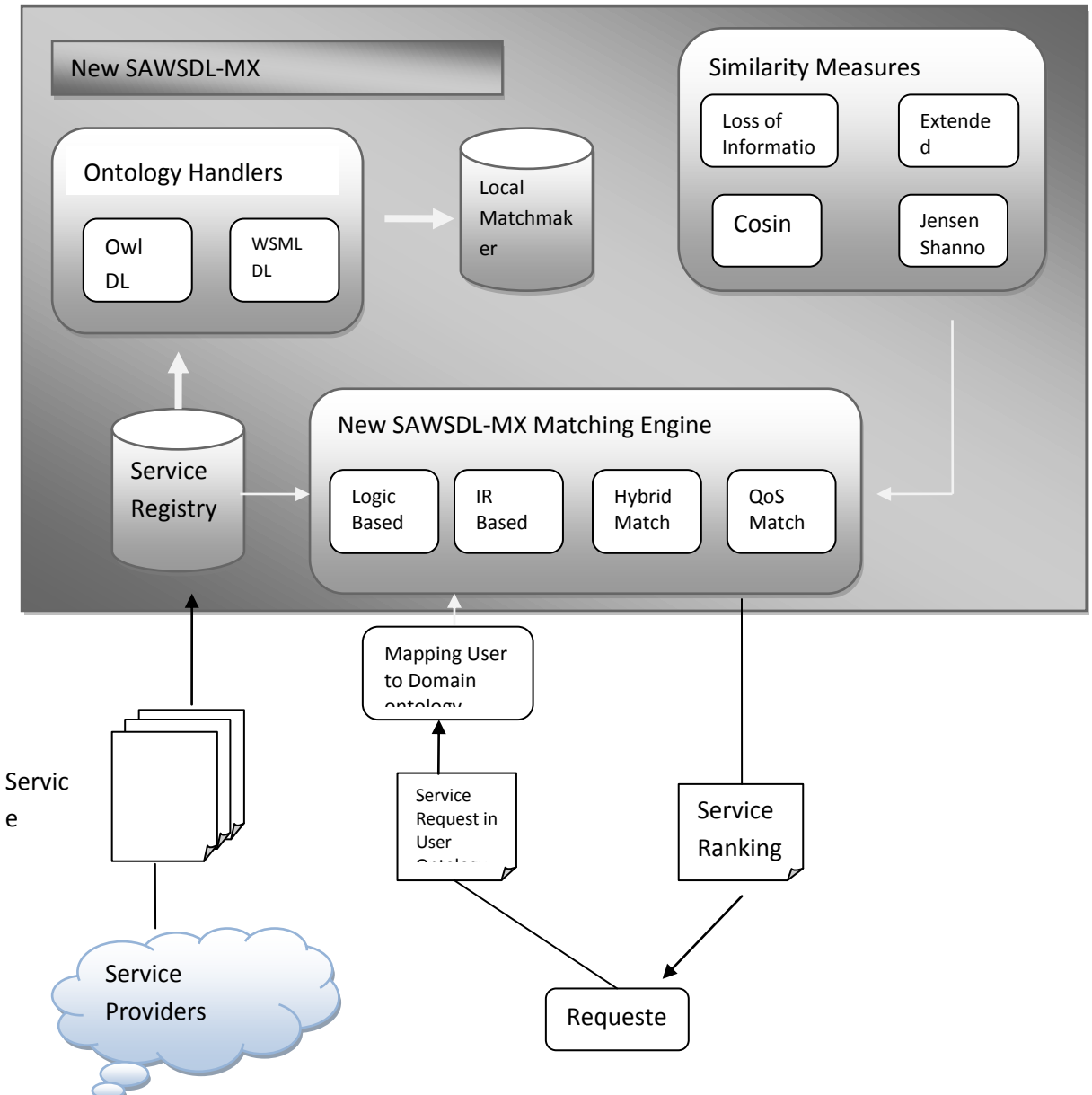
4.2 QoS Matcher

This component incorporates non-functional aspects into the discovery process. Definition (**Candidate Service Providers**): Let $\mathbb{S} = \{S_1, \cdot \cdot \cdot, S_n\}$ denote the set of services which are available (or registered with our system). We call, $\mathbb{S}' \subseteq \mathbb{S}$, the set of candidate providers, if they meet the requested functional properties of the user in terms of inputs, output, precondition and effects (IOPE's). This component takes as input, the candidate services returned by either logic-based or IR-based or hybrid matcher and return services that satisfy functional specification and have overall score (for the non-functional attributes) greater than some threshold value specified by the user. If several services satisfy these constraints, it is at the discretion of the

user. But, if no service exists, then an exception is raised and the user is notified appropriately.

The Service Requester specifies a request for service using the Service Requesting API. Such a request is described using OWL-DL. That allows to apply standard subsumption reasoning used for OWL-S MX. The requester also specifies the interoperation constraints (ICs) between the terms and concepts of its ontologies to the domain ontologies.

For our first prototype, the constraints are defined manually. With the help of these translations, the service requesting API transforms the requester's query, into a domain-specific query. The matchmaking engine then tries to find service advertisement(s) which match the user's request. This process is the same as SAWSDL-MX1.0 and it returns a set of candidate service providers which is the taken as input by the QoS Matcher. For a particular service request query, our system selects one or more services which satisfies user's constraints (in terms of functional requirement) and has an overall score (for the non-functional attributes) greater than some threshold value specified by the user. But, if no service exists, then an exception is raised and the user is notified appropriately. The QoS Algorithm is given in 4.3



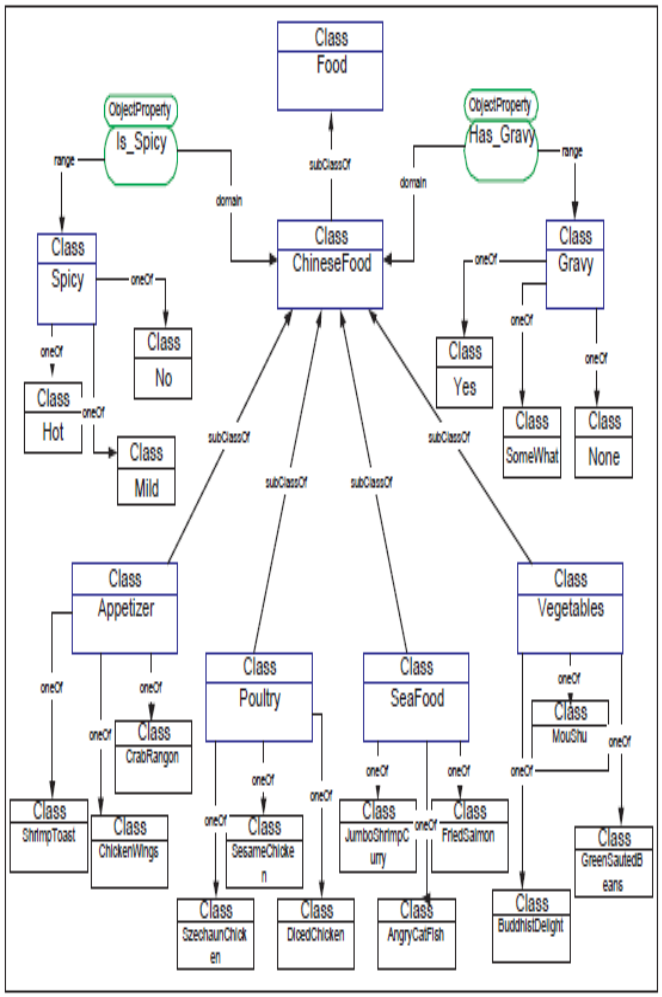


Figure 2: Domain Ontology for Chinese Food

4.3. QoS Algorithm

Finds services which belong to candidate services that best match a requesters non-functional requirements A_i; it returns set of services $(Service_j, f_{QoS})$ with additive value function

$f_{QoS} \geq threshold\ score\ vale\ (\varphi)$
$Function\ QoS\ (Attribute\ A_i, Weight_i, \varphi)$
$local\ \mathbb{Q}, result$
$\mathbb{Q} = \{V(Q_{ij}); 1 \leq i \leq m; 1 \leq j \leq n\}$
$for\ all\ Candidate_{Service}\ do$

$$f_{QoS}(Service_j) = \sum_{i=1}^{m} (V(Q_{ij}) \times Weight_i)$$

$if\ f_{QoS} \geq \varphi\ then$
$result := result \cup \{(Service_j, f_{QoS})\}$
$end\ if$
$end\ for$
$return\ result$
$end\ Function$

5. Service Discovery using Proposed Framework

We now give a detailed example to ascertain the workability of our framework.

Example 1: Let $Z = \{S_1, S_2, S_3, S_4\}$ be the set of services returned that satisfy the functional requirement of the user based on logic, syntactic or hybrid filters. Now if the user is interested in scalability, availability, ratings, latency and throughput. Assuming based on the QoS attributes specified in the service description, the quality matrix is given below

$$\mathbb{Q} = \begin{pmatrix} Attributes/Service & S_1 & S_2 & S_3 & S_4 \\ scalability & 0.85 & 0.73 & 0.67 & 0.3 \\ Availability & 0.85 & 0.84 & 0.09 & 0.25 \\ Rating & 0.91 & 0.94 & 0.5 & 0.35 \\ Latency & 0.05 & 0.07 & 0.11 & 0.01 \\ Throughput & 0.83 & 0.96 & 0.55 & 0.23 \end{pmatrix}$$

The additive value function for each service is given below

$$\begin{pmatrix} Service & S_1 & S_2 & S_3 & S_4 \\ f_{QoS} & 2.02 & 1.96 & 1.0 & 0.67 \end{pmatrix}$$

Assuming the user specifies
$weight\ (scalability) = 0.8, weight\ (availability) = 0.9,$
$weight\ (rating) = 0.5, weight\ (latency) = 0.7,$
$weight\ (throughput) = 0.1\ and\ threshold\ score\ (\varphi) = 1$
Clearly, we see that only services $S_1, S_2\ and\ S_3$ will be returned after calculating their respective f_{QoS} scores. There will be implicit ranking of these returned services based on the one with the highest f_{QoS} score to the lowest.

6. Future

In this paper, we presented only the framework of our proposed matchmaker, the implementation is however, an ongoing work. Also in this initial framework, interoperability constraints are specified manually by the user, a possible future work is to incorporate (semi) automatic correspondences between user and domain ontology should be considered. Also, allowing the matchmaker to accommodate model reference to multiple ontologies description will further enhance its capabilities.

7. Related Work

Ontologies have been identified as the basis for semantic annotation that can be used for discovery. Ontologies are the basis for shared conceptualization of a domain (Gruber,

1993), and comprise of concepts with their relationships and properties. Use of ontologies to provide underpinning for information sharing and semantic interoperability has been long realized (Gruber et al, 1991; Kashyap et al, 1994; Wache et al, 2001). By mapping concepts in a Web resource (whether data or Web Services) to ontological concepts, users can explicitly define the semantics of that resource in that domain. The semantic relationships in ontologies are machine readable, and thus enable inferencing and asking queries about a subject domain. The W3C standard OWL (www.w3.org/TR/2004/REC-owl-guide-20040210) is based on RDF(S) and supports the modeling of knowledge/ontologies (Herrmann et al, 2006). OWL supports developing of ontologies in a powerful way, but lacks in describing the technical details of services. WSDL-S (adopted in industry as SAWSDL) extends WDSL in order to use semantic capabilities of OWL to provide semantically enriched meanings of WSDL services – WSDL-S connects WSDL and OWL in a practical way (Akkiraju et al, 2005). In the work of Pierre (2007), a WSDL-S to UDDI mapping was defined and a service publication and querying API named LUCAS (layer for UDDI compatibility with annotated semantics) is placed around it. Jyotishman et al (2005) proposes a framework for ontology-based flexible discovery of Semantic Web Services. The proposed approach relies on user-supplied, context-specific mappings from a user ontology to relevant domain ontologies used to specify Web Services. A description of how user-specified preferences for Web Services in terms of non-functional requirements (e.g., QoS) can be incorporated into the Web Services discovery mechanism to generate a partially ordered list of services that meet user- specified functional requirements. Our approach is similar to this in the sense that both provide mappings from user to domain ontology however, their approach is OWL based while ours is SAWSDL based. OWL-S/UDDI (Srinivasan et al, 2004)) matchmaker combines UDDI's proliferation into the Web Services infrastructure and OWL-S's explicit semantic description of the Web Services. Glue (www. swa.cefriel.it/Glue) is a WSMO compliant discovery engine that aims at developing an efficient system for the management of semantically described Web Services and their discovery. OWL-S MX by Matthias et al (2008) is a hybrid OWL-S semantic service matchmaking algorithm. It uses both logical based and content based retrieval techniques for Web Services discovery. The hybrid semantic service matching uses six different filters to calculate the degree of semantic match between the request and advertisement. The first four filters are purely logic based and the next two are hybrid, which use the IR similarity metric values. SAWSDL –MX by Kapahnke et al (2008) is an approach that does hybrid semantic Web Services discovery using SAWSDL based on logic based matching as well as information retrieval based techniques. It is significantly based on but a further refinement of OWL-MX. Our work extended the work of Kapahnke et al. as stated earlier.

8. Conclusion

It is a fact that users will always want to request for services of interest to using terms defined in their own ontology. So, we included a mechanism that allows mapping from user ontology to the domain ontology used to describe the services. This mapping is specified in form of interoperation constraints between the user and domain ontologies. Since ontology associates ordering to corresponding hierarchy, if we can associate a concept from user ontology to another in the domain ontology, we can then form a relation between all the concepts in the respective domain i.e. user and domain ontology. Also, Web Services are distributed as well as autonomous by their very nature, and can be invoked dynamically by third parties over the Internet, their QoS can vary greatly. Thus, it is vital to have an infrastructure which takes into account the QoS provided by the service provider and the QoS desired by the service requester, and ultimately find the (best possible) match between the two during service discovery. Integrating QoS features in the profile of Web Services is to the advantage of both users and providers. QoS profiles of Web Services are crucial in determining which service best addresses the user desires and objectives. If the discovered Web Services are accompanied with descriptions of their non-functional properties, then the automated Web Service selection and composition that takes place, considers the user's QoS preferences in order to optimize the user's Web Service-experience regarding features such as performance, reliability, security, integrity, and cost. On the other hand, QoS can give Web Service providers a significant competitive advantage in the e-business domain, as QoS-aware services meet user needs better and thus attract more customers. As major contribution, the extension of SAWSDL-MX to allow for mapping of user to domain ontology, gives users the flexibility to make queries without having to use the domain ontology. Also the incorporation of non-functional specification in the discovery process of SAWSDL-MX makes it user centered in that it returns what the user wants bearing in mind tradeoffs that may exist.

8.1 Limitations of our Framework

Potential of the proposed Framework is illustrated in the examples in the earlier section. However, a more concrete feel can be obtained when the proposed Framework is implemented. Also, our proposed framework takes care of

only problem of ontology heterogeneity. Problem of ontology description heterogeneity still persists.

9. Acknowledgements

I will first of all like to register my profound gratitude to God Almighty for the opportunity to start and successfully complete this program. My sincere gratitude goes to my wonderful and accomplished supervisor Prof. S. B. Junaidu, for the guidance and free will to express myself in the course of this research. Sir, I am indeed grateful. Also, I will like to say a big thank you to my minor supervisor in the person of Dr. N. Choji. To my lecturers, I say a bucketful of thanks and pray God grants each of your heart desires. My colleagues in the department, what can I say but a gracious thanks for your support and encouragements. Ha, Mr. Kana, what can I have done without your encouragements and support? May God grant your heart desires according to his riches in glory. This acknowledgement can never be complete without saying big up to you my wonderful parents, brothers, sisters, nieces, nephews and all relatives. You all are dear to me as such I register my warmest appreciation and say "thank you all"

10. References

1. Akkiraju, R. J., Farell, J., Miller, A., Nagarajan, M., Sheth A. and Verma K. (2005). Web Services Semantics – WSDL-S [online] Available http://www.w3.org/2005/ 04/FSWS/Submissions/17/WSDL-S.htm.
2. Ankolenkar, A., Burstein, M., Hobbs, J.R., Lassila, O., Martin, D.L., McDermott, D., McIlraith, S.A., Narayanan, S., Paolucci, M., Payne T.R., and Sycara, K. (2002). The DAML Services Coalition, "DAML-S: Web Services Description for the Semantic Web", The First International Semantic Web Conference (ISWC), Sardinia (Italy).
3. Business Process Execution Language for Web Services Version 1.1. [online] available ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf
4. Caragea, D., Pathak, J. and Honavar V. (2004). Learning Classifiers from Semantically Heterogeneous Data Sources. In 3rd Intl. Conference on Ontologies, DataBases, and Applications of Semantics for Large Scale Information Systems.
5. Cardoso, J. and Sheth A. (2002). Semantic e-Workflow Composition. Journal of Intelligent Information Systems (JIIS), 21(3), 191-225. Kluwer Academic Publishers
6. Cardoso, J., Miller, J. and Emani, S. (2008). Web Services Discovery Using Annotated WSDL. *In*: Reasoning web Fourth International Summer School 2008 Springer.
7. Curbera, F., Nagy, W. and Weerawarana, S. (2001). Web Services: Why and How. *In* Workshop on Object-Oriented Web Services – OOPSLA 2001, Tampa, Florida, USA.
8. DAML-S 0.7 Draft Release, 2002.
9. Duftler, M.J., Mukhi, N.K., Slominski, A. and Weerawarana, S. (2001). Web Services Invocation Framework.
10. Duy Ngan L., Soong Goh, A. E.,Tru Hoang C. (2007). A Survey of Web Services Discovery Systems: International Journal of Information Technology and Web Engineering, Vol. 2, Issue 2.
11. Fensel, D., and Bussler, C. (2002). The Web Services Modeling Framework WSMF. *In*: Electronic Commerce Research and Applications, Vol. 1, Issue 2, Elsevier Science B.V.
12. Garofalakis, J., Panagis, Y., Sakkopoulos, E., and Tsakalidis, A. (2004). Web Services Discovery Mechanisms: Looking for a Needle in a Haystack? Paper presented at the International Workshop on Web Engineering
13. GLUE. Available from: http://swa.cefriel.it/Glue
14. Gruber, T.R. (1991). The role of common ontology in achieving sharable, reusable knowledge bases. *In* J. A. Allen, R. Fikes, and E. Sandewall, (Ed), San Mateo, CA,. Morgan Kaufman
15. Gruber, T.R. (1993). A Translation Approach to Portable Ontology Specifications." Knowledge Acquisition, 5(2), 199-220.
16. Herrmann, M., Golden, R. (2006). Why SOA lacks reuse. *In:* JBoss World 2006, http://jbossworld.com/jbwv_2006/soa_for_ the_-real_world/ HERRMANN_CO-Layer_final.pdf
17. Herrmann, M., Muhammad, A. A. and Dalferth, O. (2007). Applying Semantics (WSDL, WSDL-S, OWL) in Service Oriented Architectures (SOA): 10th Intl. Protégé Conference.
18. http:// twww.w3.org/TR/sawsdl/
19. Jyotishman P., Neeraj, K., Doina, C., Vasant, G. H. (2005). A Framework for Semantic Web Services Discovery. *In* ACM 7th Intl. workshop on Web Information.

20. Kapahnke, P. and Mathias, K. (2008). Semantic Web Services Selection with SAWSDL-MX.
21. Kashyap, V. and Sheth, A. (1994). Semantics-based Information Brokering. *In* Proceedings of the Third International Conference on Information and Knowledge Management (CIKM).
22. Kaufer, F.and Klusch, M. (2006). WSMO-MX: A Logic Programming Based Hybrid Service Matchmaker. Proceedings of the 4th IEEE European Conference on Web Services (ECOWS 2006), IEEE CS Press, Zurich, Switzerland.
23. Keller, U. (2004). WSMO Web Services Discovery.
24. Kiefer, C. and Bernstein, A. (2008). The Creation and Evaluation of iSPARQL Strategies for Matchmaking. Proceedings of the 5th European SemanticWeb Conference (ESWC), Lecture Notes in Computer Science, Vol. 5021, pages 463–477, Springer-Verlag Berlin Heidelberg.
25. Matthias, K., Benedikt, F. and Mahboob K. (2008). OWLS-MX: A hybrid SemanticWeb Services matchmaker for OWL-S services. Web Semantics: Science, Services and Agents on the World Wide Web.
26. Menascé, D. A. (2002). QoS Issues in Web Services. IEEE Internet Computing, vol. 6, no. 6, pp. 72-75.
27. Miller J., Verma K., Rajasekaran P., Sheth A., Aggrawal R., and Sivashanmugam K. (2004). WSDL-S: A Proposal to W3C WSDL 2.0 Committee, http://lsdis.cs.uga.edu/projects/WSDL-S/wsdl-s.pdf
28. OWL (Ontology Web Langugae); W3C Recommendation http://www.w3.org/TR/2004/REC-owl-guide-20040210/
29. Paolucci, M., Kawamura, T., Payne, T.R. and Sycara, K. (2002). Importing the Semantic Web in UDDI. Appeared In Web Services, E-Business and Semantic Web Workshop.
30. Paolucci, M., Kawamura, T., Payne, T.R. and Sycara, K. (2002). Semantic Matching of Web Services Capabilities. Proceedings of the 1st International Semantic Web Conference.
31. Patil, A., Oundhakar, S. and Sheth, A. (2003). Semantic Annotation of Web Services, Technical Report, LSDIS Lab, Department of Computer Science, University of Georgia.
32. Pierre, C. (2007). Toward a Semantic Web Services discovery and dynamic orchestration based on the formal specification of functional domain knowledge. Thales Land & Joint Systems, LIP6 Computer Science Laboratory in proceedings of ICSSEA.
33. Radatz , J. and Sloman, M. S. (1988). A Standard Dictionary for Computer Terminology: Project 610. IEEE Computer, 21(2).
34. Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/soap/.
35. SOAP Version 1.2 Part 1: Messaging Framework [online] Available http://www.w3.org/TR/soap12-part1/. Srinivasan, N., Paolucci, M. and Sycara, K. (2004). Adding OWL-S to UDDI, implementation and throughput. Paper presented at the First International Workshop on Semantic Web Services and Web Process Composition. Dan Diego, California, USA.
36. Sycara, K., Paolucci, M., Ankolekar, A. and Srinivasan, N. (2003). Automated discovery, interaction and composition of Semantic Web Services. Journal of Web Semantics, vol 1, Elsevier.
37. Tversky, A. (1977). Features of Similarity. Psychological Review. 84(4): p. 327-352.
38. UDDI Version 3.0 [online] http://uddi.org/pubs/uddi-v3.00-published-20020719.htm
39. Universal Description, Discovery and Integration (UDDI), http://www.uddi.org/
40. Verma, K. (2005). A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. Journal of Information Technology and Management.
41. Verma, K. (2007). Allowing the use of Multiple Ontologies for Discovery of Web Services in Federated Registry Environment: Athens. p. 1-27.
42. Wache, H., V ogele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., and Hubner, S. (2001). Ontology-based integration of information - a survey of existing approaches. *In*: Stuckenschmidt, H., (Ed.), IJCAI-01 Workshop: Ontologies and Information Sharing, 108-117.
43. Walsh, A. E. (2002). UDDI, SOAP, and WSDL: The Web Services Specification Reference Book (1st edition ed.) 0130857262: Pearson Education
44. Web Ontology Language for Web Services, http://www.daml.org/services/
45. Web Services Description Language (WSDL) [online] Available http://www.w3.org/TR/wsdl.

46. Web Services Modeling Ontology, http://www.wsmo.org/
47. WSDL: Web Services Description Language, http://www.w3.org/TR/wsdl.
48. WSML. Web Services Modeling Language (WSML). 2004 [cited 2004; Available from:http://www.wsmo.org/wsml/index.html.