An Efficient Technique for Web Services Identification

Amjad Farooq[1] and Rabia Arshad[2]

[1,2]Computer Science and Engineering Department,
University of Engineering and Technology, Lahore-Pakistan

***Abstract*– The growth of applications over the network where user needs system to system interaction, created the need for a structure which provided interaction not only with the user but also help application to application interaction. This problem has a common name of Application Integration. It can be resolved using Service Oriented Architecture, Enterprise Applications and Web Services. Moreover, with the addition of the intelligence and autonomy of software agents, transactions may be equally automated for consumer-to-consumer, business-to-consumer, and business-to-business collaborations. But it is a clear observation that the world is a global village and the numbers of services on the web are increasing day by day. Because of this increasing number, a problem is raised to find an appropriate web service which satisfies user needs. The user wants an efficient way to find an appropriate web service which satisfied his needs in a short time. In this paper we describe an efficient technique to find web services by combining two strategies including 'input parameter search' and' synonym based search'. We believe that this technique would result in a set of best matching services that satisfy the users' needs. We also describe an algorithm of how service discovery can be done, using the aforementioned technique.**

***Keywords*–** Semantic Web Services Discovery, Service Oriented Architecture, Ontology and SWS Search using Parameter

I. INTRODUCTION

In today's world there are a huge number of web services and to find an efficient and correct service which satisfies the user needs is a difficult challenge. The main problem of lies within keyword based searching. These may be different words used in different terms inside different domains. At the result of this search we often receive a collection of irrelevant information. To avoid this irrelevant information, semantic web technology is introduced. [1] Semantic web services are used in applications that exist on Networks and quickly respond to the new business needs. For the service discovery in the proposed approach, we search the semantic web service description syntactically. Semantic description of web services plays a very important part in semantic web service discovery [4].

To participate in the work of semantic web discovery we proposed an approach for service discovery in semantic web services. Our approach considers both correctness and efficiency to search web services. This paper gives a short description of the work we have done. Section 2 describes literature survey of the work which already been done in the field of semantic web service discovery. Section 3 provides our proposed algorithm. Section 4 gives a description of the matching algorithm for filtering the web services and section 5 gives an idea of how this approach is implemented. Finally section 6 describes how to optimize our proposed algorithm and recommends some future work that can be done in this area.

II. LITERATURE SURVEY

Web services facilitate today's industry to provide interoperability between applications which are developed on various Frameworks. Web services are defined as "a software application identified by a URI, whose interfaces and bindings are capable of been defined, described, and discovered as XML artifacts. A Web Service supports direct interaction with other software agents using XML-based messages exchanged via internet-based protocols."[2] The web services interface is described in a machine process able format like (WSDL, UDDI etc.). Other systems interact with web services using SOAP messages and XML serialization. There are many frameworks used for the development of semantic web services. Service discovery is an important part of these frameworks. Comparisons of some are given in Table 1:

TABLE1: SERVICE DISCOVERY APPROACH IN DIFFERENT FRAMEWORKS.

Framework	Service Discovery	Limit
Service Discovery Framework	OWL-S matchmaker use different discovery techniques based on information retrieval, Artificial Intelligence techniques and syntactical and semantic descriptions and constraint matching.	Limitations on service applicability and service quality.
WSMT	Matching abstract goal description with semantic annotation of web service.	Other tools are not integrated in this toolkit.
IRS-II	Matching abstract goal description with semantic annotation of web service.	Does not perform the service discovery and selection automatically.
SADI		Does not propose new Technologies
UPML		Inter and intra relationship between different parts of architecture

In the past many techniques have been used to search the service in semantic web service discovery. Some of these are briefly described below:

A. Input/output Parameter Matching

A web service has two sets of parameters normally input parameter and output parameter. This parameter is used for SOAP request in web services. A web service search query Q is satisfied if (P is subset of web service input parameter)^ (web service output parameter subset of Q). Q has also two sets of parameter P= {input parameter} and R= {output parameter}.

The service discovery uses the following two strategies;' Forward Service Discovery' and 'Backward Service Discovery'.[3]

These two strategies are given below:

Backward WS Discovery

1. For each resultant parameters *Ri* in *R*, find the web services set *WSSi* that includes *Ri* in *wsout*. *WSS1, WSS2, ...WSSm* are generated in this step, where $m = |R|$
2. Calculate *candidates n WSS* $= WSS \cap WSS \cap ... \cap WSS$ 1 2
3. For each service candidate *Ci* in *WSScandidates*, if $ws\ C\ P\ in\ (i) \subseteq$, add *Ci* into the discovery results set

Fig. 1: Backward WS Discovery

Forward WS Discovery

1. For each parameters *Pi* in *P*, find the web services set *WSSi* that takes *Pi* in *wsin*. *WSS1, WSS2, ...WSSn* are generated in this step, where $n = |P|$
2. Calculate *candidates n WSS* $= WSS \cup WSS \cup ... \cup WSS$ 1 2
3. For each service candidate *Ci* in *WSScandidates*, if $()\ ()\ out\ i\ in\ i\ ws\ C \supseteq R \wedge P \supseteq ws\ C$, add *Ci* into the discovery results set

Fig. 2: Forward Service Discovery

By overlapping of these two sets of forward and backward strategy, we are able to reduce the number of services that are evaluated for user searches.

B. Ontology Mapping of Input/output Parameter

Another approach which is used for semantic service discovery is ontology mapping of input and output in functional properties of semantic web services. The service provider and service requester use domain ontologies for building semantic web descriptions. If web service requester and web service provider use different ontologies, it needs an ontology mapping for resolving service discovery and interoperability. It provides an ontology mapping processor which takes two ontologies as an input; ontology of input/output parameter of service provider and ontology of input/output parameter of service requester. It then calculates the similarity of both ontologies. Ontology is complex information. The similarity calculation of ontologies is calculated on basis of structural, name and instance similarity.

III. PROPOSED APPROACH

Service discovery is an essential part of semantic web services Framework since the future web must allow making use of web services without continuous user intervention.

A Web Service contains two types of Parameters; Input Parameters and Output Parameters. The other information about web service what are the main tasks done by that web service. The functionality performed by web service is known as task performed by that service. We use input parameters, their synonyms, goal description and goal description synonym matches. The web services for given requirement and their input parameters are discovered using the following steps:

i. Match the web service using goal description given by user in deployed web services in IRS. The set of web services are called SW_g= GSW1…..GSWn.
ii. Match web services by every input Parameter SWSi….SWSn.
iii. Match another set of services that are discovered using synonym from dictionary or local database of goals and input parameters. This gives a set of services SWSs= $SWS_{goalsynonym}$+ $SWS_{input\ synonym}$.
iv. Calculate candidate services WSScan= SWg+SWSin+SWSs. Now this result set contains all possible services which satisfied users' requirement.

For each Candidate Service, if it satisfied (WS_{out}(subset of) output Parameter^input Parameter(sub set of) WSin ^WS_{goal}(match)user request goal) OR WSout(subset of) output $Parameter_{synoynm}$^ input $Paramete_{synoynm}$ (sub set of) WSin ^WS_{goal}(match)user request $goal_{synonym}$, add this service to resultant discovery set.

The flow chart for the above proposed algorithm is as follows:

IV. MATCHING ALGORITHM

Matching algorithm is an efficient, layered, simple and extensible algorithm. Different algorithms are applied for service discovery and selection. The proposed matching service algorithm is basically based on input parameters matching and goal description requested by user match with the goal description of the service. The composition of services is out of the scope of this paper.

The proposed algorithm mainly used semantic service description of web services. The service description provides input parameters and their types. Service descriptions play a very important role in discovery of services and composition of those services.

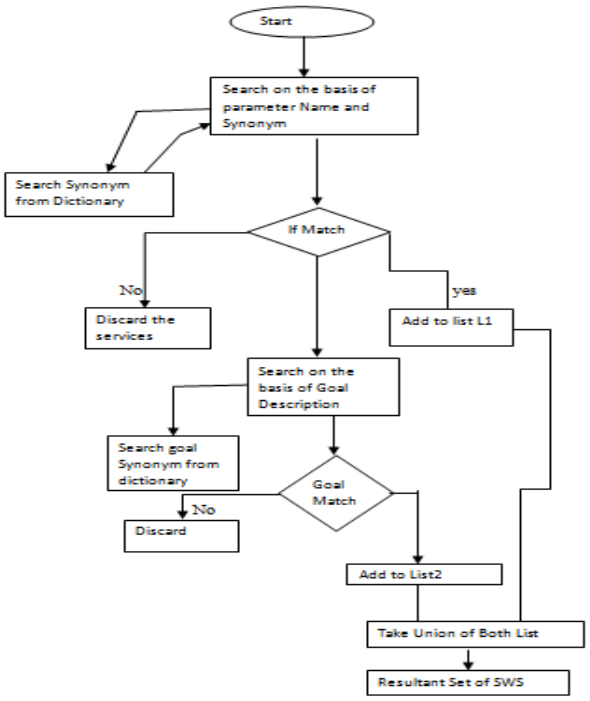

Fig. 3: Proposed Algorithm Flow Chart

A. Service Domain Matching

Web services high definition ontology defines the domain of services and the category in which the service lies. Mostly services are exposed for different categories. One service does not fulfill the requirement of two categories e.g. one service cannot be used for travel and ecommerce at the same time. In this step service categories will be filtered according to user requested service categories. This filtering reduces the number of services which are evaluated [8].

B. Input Parameter and Synonym Matching

The concept of matching the input parameters is to match the name of input parameter and synonyms search from within a dictionary. Matching of input parameter is based on different relationships. There are mainly four matching criteria on which input parameters are matched.

Matching level signifies the level of matching of client's Request and advertised service description. There are four types of level of match defined. Some of the detail of match algorithm is as follows.

- *Exact:*

The input parameter provided by requestor is same as service parameters.

- *Plug-In:*

The web service provided by provider is more general concept than required by user [7].

- *Sub-sumption:*

The provided output by web service is subclass of the search input.

- *Fail:*

The web service which does not meet criteria with the input given by requester.

C. Matching Goal Description

Each web service has its description i.e. what it has done and for what this web service is designed. When a client gives an input parameter to search the service and a goal description for which the user wants to search the service. This description matches with the goal description of all advertised services and adds services to the candidate service list. In the second step, a synonym of goal description is searched from our knowledge base and it matches the service on the base of this description and adds these discovered services into candidate service list.

D. Make Final List

We take the list of candidate services and perform an intersection of both. This gives us matching services from both lists. Now different ranking criteria are applied on the result set and the best service is selected according to the requirement.

V. IMPLEMENTATION

The pseudo code for matching input parameters and their synonym are given as follows:

```
Public MatchResult inputMatch (Vector searchInputList, Vector
advertisedInputList ) {
MatchResult matchResult = new MatchResult();
Ð f(advertisedInputList ==null) { // if no input service requires then exact match
matchResult .type= EXACT;
}
for (int i=0; i< advertisedInputList.count;i++ { // for each input of adv. service
Input advInput = (Input) advertisedInputList[i];

if(searchInputList != null ) { // for each searched input try to match
for (int j=0; j< searchedInputList.count);j++ {
Input srcInput = (Input) searchedInputList[i];
type = findMatchType (srcInput, advInput ); // find match type
```

Fig. 4: Pseudo Code for Algorithm

In the second step we take a list of services by matching goal descriptions given by user and goal description synonyms.

```
publict candidatelist findMatchgoaldescr(inputgoaldesc,advinputgoall
{
  for(int i=0;i<advinputgoallist;i++)
{
   if(inputgoaldesc==advinputgoal[i].description)
      Addserviceinto candidateservicelist;
else
        discardservice;

}
}
```

Fig. 5: Goal Description Matching Algorithm

And in last step we take intersection of these two lists, the pseudo code for taking intersection is given as follows:

```
Public candidateServiceList
TakeIntersection(candidateserviceParameterList,candidateservicegoalli
{
   For (int i=0; i< candidateserviceParameterList.coun;i++)
    {
          For (int j=0; j< candidateservicegoalList.coun;j++)
        {
            If (candidateservicelist[i]=== candidateservicegoalList[j])
            {
                Add into list; //Add into candidateservicelist
            }
        }
   }
}//Function end
```

Fig. 6: Final Result Set Algorithm

This gives the final list of services for ranking the best service which meets with user needs. This algorithm reduces the number of services for final evaluation.

VI. IMPROVING PERFORMANCE

This algorithm improves the performance of service discovery. All matching services are filtered from matching of input parameter, parameter synonym, goal description matching and goal synonym matching. In the last step filter the services for ranking by taking intersection of two sets of services. The intersection step reduces the number of web services for checking which meets with user needs. Ranking services after discovery is a time consuming process if lists of the filtered service are lengthy. This algorithm reduces the number of services for evaluating and time is also saved.

VII. CONCLUSIONS

In this work we have proposed a new approach for the semantic web discovery for achieving better performance. It introduces several features which were missing in current works. It also gives a user friendly semantic search of parameter names using different search methods and searches on the basis of goal description given by user.

One of the main problems in creating service description would require service provider to develop ontologies for different service type and annotation on non-functional parameter of the service. The problem in the effort required composing of Ontologies and omitting disambiguates in domain concepts. Once a comprehensive collection of domain ontologies were available, creation of service description and deployment was very easy. But the collection of Ontologies is a difficult task. But the creation of ontologies was not the main task of this thesis. This thesis concentrated on the service discovery process. Service discovery plays an important role in semantic web services development Framework.

In the web services that are highly data intensive a major role is performed by Ontologies. But manually Ontology creation is very difficult. Much research is performed by people on service discovery. The most common way for accessing web services by creating their directories. The service is described in a standard way and the available information is encoded so that its information can be used in a standard way.

After combining service parameter and on the basis of goal description search we got better results. We outlined an approach of web service discovery with matching input parameter and parameters synonyms in service description and goal description given by user search on ontological level. We used the generic goal concepts given by user. The main focus in this thesis was to improve service discovery mechanism in semantic web services development Frameworks.

VII. FUTURE WORKS

At its core, semantic web consists of different collaborative working groups and sets of enabling technologies. Now we discuss future work and recommendation for this thesis. First of all a stable framework is needed for the development of semantic web Applications. This tool would enable the combination of different types of applications. Because a web service calls from many different platforms, it should be communicated with every user. If user adds any information about their requirements, this tool should consume this additional information in a proper way and store it. This information can be utilized to improve the service parameters and ontology classes defined for the service. Different approaches and service discovery methods are integrated to get better discovery result set [4]. And make the service discovery approaches more transparent what is approach scope is accommodate dynamic descriptions or use only static descriptions? Because, dynamic information utilized to improve service discovery mechanism [5].

REFERENCES

[1] Michal Zaremba,"A Semantically Enabled Service oriented Architecture",in Digital Enterprise Research Institute (DERI),University of Innsbruck, Austria. Available http://www.heppnetz.de/files/wimbi2006.pdf

[2] Mark Endrel, et al. Patterns: "Service Oriented Architecture and Web Services" in IBM Red Books, IBM(2004), Available http://www.redbooks.ibm.com/abstracts/sg246303.html

[3] Yue Zhang,"Strategies for Efficient Syntactical and Semantic Web Services Discovery and Composition", in *Dept of EECS, University of California, Irvine, CA 926972625,Available*http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1640327

[4] Jussi Koskinen, Software Maintenance Costs. Information Technology Research Institute, ELTIS- Project University of Jyväskylä (2003)

[5] Uwe Keller, Ruben Lara,Holger Lausen,Axel Polleres,Livia Predoiu Ioan Toma*a* "Semantic Web Service Discovery" in

WSMX Working Draft { October 3, (2005),Available http://www.wsmo.org/TR/d10/v0.2/d10.pdf

[6] Thomas Gschwind, "Evaluation of Semantic Service Discovery", (A_Survey_and_Directions_for_Future_Research.pdf) in Emerging Web Services Technology, Volume II, October (2008).

[7] Pensri Pukkasenung, "An Efficient Semantic Web Service Discovery Using Hybrid Matching"Available http://pioneer.chula.ac.th/~sperapho/pub/web3.pdf

[8] Joanna Irena Ziembicki , "Distributed Search in Semantic Web Service Discovery" in Waterloo, Ontario, Canada, (2006), Available http://etd.uwaterloo.ca/etd/jiziembi200