**Semantic Web Ontology for Automated Web Services Discovery**

Jayashree, K., R. Chithambaramani, R. Babu

Assistant Professor C.S.E. Department, Rajalakshmi Engineering College, Anna University, India.
Assistant Professor I.T. Department, Rajalakshmi Engineering College, Anna University, India.

<table>
<thead>
<tr>
<th>ARTICLE INFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Article history:</td>
</tr>
<tr>
<td>Received 25 August 2014</td>
</tr>
<tr>
<td>Received in revised form 19 September 2014</td>
</tr>
<tr>
<td>Accepted 29 October 2014</td>
</tr>
<tr>
<td>Available online 16 November 2014</td>
</tr>
</tbody>
</table>

| Keywords: |
| Automatic discovery, Web Ontology Language (OWL), Web services ontology. |

**ABSTRACT**

Web services help the integration of diverse application over the Internet. The web services are published by the service providers in service registry having different categorization. Large numbers of categories are available in the service registry so it is difficult to discover the related services from categories. The semantic web ontology approach is considered to be efficient for selection of services from different categories. Multiple services are discovered and combined. Combining similar web services is a crucial approach for web service discovery. In this paper ontology is developed to automate discovery of web services. Discovery progresses through four phases such as exact match, similar match, constraint reasoning and best first match.

**INTRODUCTION**

A web service is a software interface that describes a collection of operations that can be accessed over the network through standardized Extensible Markup Language (XML) messaging. It is based on the XML language to describe an operation to execute or data to exchange with another web service. A web service is a standard way of integrating web-based applications using the XML, Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL), and Universal Description Discovery Integration (UDDI) open standards over an Internet protocol.

Semantic web increases the intelligence of the web, enabling richer discovery, data integration, navigation and automation of tasks. Semantic web ontology facilitates exact, efficient, and automatic retrieval of the required web services derived from the semantic matchmaking of web service capabilities (Pokraev et al, 2003). To promote the automation of service discovery, different semantic languages have been created that allow describing the functionality of services in a machine interpretable form using semantic web technologies. The problem is that users do not have intimate knowledge about semantic web service languages and related toolkits. Service providers publish their services along with their constraints to the UDDI registry. A Novel approach for web services discovery based on functional parameters has been proposed.

Services are searched in the UDDI registry by matching IOPE of user request with semantic web services description. Using the match making algorithm the services are matched. After filtering out all unmatched services, the URL of best service is discovered and it is invoked by the service client and finally binds to the service provider. In this paper, we present an automatic discovery framework for web services. The rest of the paper is organized as follows. Section 2 discusses the related work and Section 3 presents the proposed work. In Section 4, we discuss the implementation using a sample web service application. Section 5 presents the conclusion and future work.

**Related work:**

Service discovery is the process by which a client (service requestor), identifies candidate services to achieve the client's objectives, by interacting with peers or special purpose middle agents (matchmakers). From an architectural point of view, the discovery process involves three types of stakeholders: service providers have the goal of providing particular types of services and publish or advertise their services to enable potential clients to discover and utilize them. Service requestors have the goal of finding services that can accomplish some internal objective, and matchmakers are middle agents that can be used to find the service providers that...
match the stated requirements of a requestor. Matchmakers that both find and invoke services as proxies for the requestor are called brokers. For current generation web services, this is predominantly a manual process, in the sense that registries like UDDI are designed to be searched by developers of client systems, rather than middle agents or users of client systems during execution. In the web services architecture, web services correspond to service providers, services clients are the requestor, and UDDI registries are the middle agents.

In contrast, semantic matchmakers use semantic relations to find services described using Semantic Web languages (kunal et al. 2004). In general there are design tradeoffs to be made in selecting level of abstraction of the capability description used in a query to a matchmaker. Typically, a requestor has a specific goal, while service providers publish generalized descriptions of their capabilities intended to capture the range of specific activities they support, such as selling a general class of items. To be effective for matching against published service descriptions, capability descriptions should more abstract than specific goals, but may need to include information about how the goal should be achieved, and under some constraints.

Aabhas V. Paliwal et al. (2012) addresses issues related to automated service discovery through semantic-based categorization of web services and selection of services based on semantic service description rather than syntactic keyword matching. The semantic categorization of web services is achieved by ontology framework as offline in UDDI. Semantic enhancement of web services is achieved by parameter based service refinement and semantic similarity based matching. The matching of increased service request with retrieved service description is achieved by Latent Semantic Indexing (LSI). Ranking of semantic relationships, hyper clique pattern discovery, additionally used for the invention.

Jos de Bruijn et al. (2005) proposed a methodology for higher degree of automation in the location and use of web services that can be achieved by adding explicit semantics to web service descriptions. Such semantically enriched descriptions are usually referred to as semantic web services and they are expected to enable businesses to dynamically locate partners which provide particular services, and to facilitate semi-automated cooperation with them.

Buhwan Jeong et al. (2008) have given clear picture of automated service discovery based on capability profile for manufacturing services by extend the UDDI registry specification to include semantic descriptions about manufacturing services and to support reasoning about those descriptions for service discovery. Specifically, they have provided OWL-based definitions for manufacturing service capability profiles and a DL-based reasoning procedure for matching queries to service descriptions. OWL has been used to describe the manufacturing service capability profile. An extended UDDI registry is used to mediate contracts between designers and manufacturers, the OWL language to augment manufacturer's capability profiles, and DL-reasoning to discover such profiles. The extensions to the UDDI specification and implementation described help us to advertise and discover services in domains other than manufacturing.

Ian Horrocks et al (2005) proposes a methodology towards OWL rules that adds considerable expressive power to the Semantic Web since it has expressive limitations, particularly with respect to properties. SWRL, a Horn clause rules extension to OWL that overcomes many of these limitations. SWRL extends OWL in a syntactically and semantically coherent manner. The basic syntax for SWRL rules is an extension of the abstract syntax for OWL DL and OWL Lite. SWRL rules are written in a syntax similar to the OWL DL model-theoretic semantics. SWRL rules are given an XML syntax based on the OWL XML presentation syntax and a mapping from SWRL rules to RDF graphs is given based on the OWL RDF/XML exchange syntax.

Georgios Meditskos and Nick Bassiliades (2010) combined object-based structural matching techniques that are used in the domain of Structural Case-based Reasoning (SCBR), with Description Logic (DL) reasoning over Profile instances, enhancing the discovery with services that cannot be retrieved using only logic-based reasoning. They allowed the existence of Profile taxonomies, incorporating domain knowledge through Profile instance class membership relationships and also enhanced the discovery procedure by considering ontology roles, exploiting the excellent classification capabilities of DL reasoning. In this way, the strong points of the WSMO-DF and OWL-S SP modeling paradigms are combined.

Zhonghua Yang et al. (2005) provide a methodology to achieve automated integration based on semantic web services architecture. Service-oriented integration architecture, enhanced with semantics, which construct a service-oriented environment via virtualization by taking a WS-Resource Framework approach. All Web services are semantically enhanced using service upper ontology OWL-S and manufacturing domain ontologies which are written in standard Web Ontology Language (OWL). Within this semantic rich environment, integration is achieved by web services composition, and automation is accomplished via the combination of Semantic web and software agents. The key to the Semantic web services-based approach is the enterprise integration which enables the business process modeling to transform an integrated enterprise into a VE (virtualized enterprise). The service semantics enhancement uses the OWL and OWL-S, and automation is achieved by software agents who manipulate the semantics of web services in a service-oriented environment.

Jyotishman Pathak et al. (2005) describe a framework for ontology-based flexible discovery of semantic web services. This approach relies on user-supplied, context-specific mappings from user ontology to relevant
domain ontologies. To discover a service, user has to provide some functional requirements that meet certain selection criteria can be transformed into queries which are processed by a matchmaking engine. This framework also describe how user-specified preferences to discover web services in terms of non-functional requirements (e.g., QoS) can be incorporated into the web service discovery mechanism to generate a partially ordered list of services that meet user-specified functional requirements.

Jing Dong et al. (2009) have enforced OWL-S ontology framework extension for dynamic web service composition. Web service may be composed at the abstract service level rather than the concrete level by raising the amount of abstraction and therefore they need to project an abstract service layer. This abstract services are connected with an instant pool, contains all instances of the abstract service that facilitate fail-over and dynamic. The service instance pool permits filtering and plugging in candidate services at runtime. To automatically generate the method on the fly, they used a planner prototype which supported Java Theorem Prover (JTP). The OWL-S extension includes the data on the input/output, flow management, semantic property, and candidate instance pool of the abstract service within the ontology hierarchy.

In this literature study, researchers present an overview of semantic discovery of web services automatically during run time. Semantic owl, SWRL and match making algorithm helps to discover services automatically. Ontologies have been widely used in many areas such as knowledge management, content management, intelligent databases, electronic commerce and the semantic web (Fensel, 2001). Ontology plays a key role in the Semantic Web by providing machine readable vocabularies and the relationships among them (Martin, 2003). In this paper, we present a semantic discovery framework for automatic discovery of web services using ontology.

**Proposed work:**

Service Oriented Architecture (SOA) consists of three key components, namely, Service Consumer (SC), Service Registry (SR), and Service Provider (SP). These components interact with each other to publish, bind, find and use web services as given in Figure 1.

![Service Oriented Architecture](image)

Fig. 1: Service Oriented Architecture.

Service providers who offer web services for public use, register their services in service registries using the publish operation. These web services are expressed in WSDL format. UDDI provides a platform-independent standard framework for describing and publishing web services for discovery over the Internet. A service consumer would look up the registry to find an appropriate web services and requests the service from the service registry using find operation. The service registry returns a known provider (Service URL) for the requested service. The user service then binds itself to the service provider for service execution. The service input parameters are sent to the service provider who executes the service and returns the results to the consumer. These interactions happen through the use of SOAP messages. The research challenge is to discover
web services automatically when there are huge numbers of services stored in the service registry. A semantic discovery framework is shown in Figure 2.

It helps to discover a web service automatically, when the services are created with all the user requirements. Service providers publish the services along with their constraints to the UDDI registry. Services are then searched in the UDDI registry and it is processed through four phases such as exact match, similar match, constraint reasoning and best first match. Using the match making algorithm the services are matched. After filtering out all unmatched services, a best matched service is discovered. A URL is discovered from the UDDI registry and it is invoked by the service client and finally binds to the service provider.

Domain ontology models a specific domain, which represents part of the world. Particular meanings of terms applied to that domain are provided by domain ontology. For example the word card has many different meanings. Ontology about the domain of computer hardware would model the "punched card" and "video card" meanings. Using the details of WSDL to OWL convertor, domain ontology has been updated for the service and that is stored in ontology repository. The service client makes a service requests query which consists of domain name, service name, Input and Output parameters along with their types and constraints. The service requests go through four phases.

3.1. Exact Match:

The exact match is the first phase (Subsumption Reasoning) of the service matchmaking. Subsumption determines the service’s position in the ontology hierarchy. In this the domain name is first matched. For example in a bike selling service, when price is given as input the service should return a bike that can be bought at that price. In Figure 3, the inputs to the service are instances of concept price and output is the instance of concept bike. As a result, a list of ‘N’ services is discovered in this phase.

3.2. Similarity Match:

The similarity match is the second phase of the service matchmaking. Based on Input and Output parameters, the services are matched using semantic matchmaking algorithm. The algorithm takes OWL-S query as client’s input and iterates over every OWL-S advertisement in its repository in order to determine a match. For example, in the ontology given by Figure 3, when a request contains passion, a service with concept bike should be given more weightage than concept vehicle as passion is closer to car in the ontology. Similarly, a service with passion should be given an appropriate score (less than bike) when a request for bike is made by
taking into account the fact that passion could only partially satisfy the request.

![Diagram of vehicle ontology](image1)

**Fig. 3:** A Fragment of vehicle ontology.

### 3.3 Constraint Reasoning:

The constraint reasoning is the third phase of the service matchmaking. Constraint knowledge has been formalized using SWRL. Based on constraints like input type, field and value the services are matched. A semantic matchmaking algorithm should consider constraints while performing matches. Hence, an advertisement which satisfies the constraints should be given better ranking than an advertisement which does not satisfying it. A rule engine is used (eg. JESS) to perform constraint reasoning.

### 3.4. Best-First Reasoning:

The strategy of the best first match is implemented to select the best one among all candidate services that sufficiently match the required service based on FCFS principle. Figure 4 shows the service request process through four phases.

![Diagram of service request process](image2)

**Fig. 4:** Four phase of discovery process.
4. Implementation:

We have simulated the semantic discovery framework using an airline reservation system which includes the following services:

- Booking
- Payment
- Ticket status
- Cancellation

Booking service is used to book the flight tickets. To book flight tickets, the user should search for the flights with the inputs. So a search service is made to know about the available flights. This service search for the given inputs and reply back if there are flights available or not. Payment Services is a banking gateway to book the flight tickets. Ticket status service is used to get the status of the ticket by giving ticket id as the input. With the help of this service, the user can view about the ticket details that have been booked. Cancellation service is created to cancel the ticket by giving ticket id as the user input. Using bottom up approach all the services are created using IBM RAD 7.0 IDE and deployed to web sphere application server 6.1 and the Service Provider publishes the service along with an WSDL file, Business and Service detail for respective service such as business name, business id, service name and discovery URL for the particular service are published to the UDDI registry by the service provider using the publishing API of the jUDDI Registry as been Figure 5.

Fig. 5: Publishing services.

Fig. 6: Domain Ontology.
WSDL to OWL converter tool take the inputs as WSDL URL and convert the WSDL file into three parts such as process.owl, grounding.owl and profile.owl are created and stored into ontology repository. Using details of this files domain ontology is created in NeOn OWL toolkit. The OWL Editor of NeOn Toolkit is a modeling tool for the creation and maintenance of semantic models written in the Web Ontology Language OWL. The domain ontology for airline service system is created and as shown in Figure 6.

Discovering a service:

For discovering a service, ontology is created for an airline domain called a domain ontology using NeOn tool. The NeOn tool is an extensible environment for creating and editing ontologies. It is very easy-to-use graphical user interface. Therefore the domain ontology is created with number of relationships and its properties.

A basic graph pattern (BGP) is a set of triple patterns written as a sequence of triple patterns. A group graph pattern is a set of graph patterns delimited with braces { }. Simple SPARQL Query for airline service is shown in Table 1:

Table 1: Sample SPARQL Query for airline service

<table>
<thead>
<tr>
<th>Data:</th>
<th><a href="http://cmdairways.com/airline/service">http://cmdairways.com/airline/service</a></th>
</tr>
</thead>
</table>

The Semantic Web Rule Language (SWRL) is a proposed language for the Semantic Web that can be used to express rules as well as logic, combining OWL DL or OWL Lite with a subset of the Rule Markup Language. A simple use of these rules would be to assert that the combination of the object properties implies the and can be another object properties property. Informally, this rule could be written as:

\[
\text{hasreservation}(x_1, x_2) \land \text{haspayment}(x_2, x_3) \Rightarrow \text{hasticketbooked}(x_1, x_3)
\]

The XML Concrete Syntax is a combination of the OWL XML Presentation Syntax with the RuleML XML syntax.

The service client request process follow through four using SPARQL & SWRL. After finding the exact matched services, the corresponding target namespace, operation name, port type, URL are fetched from UDDI registry which are already published and it is sent to the service client. Finally the execution of that service takes place.

Conclusion and future work:

The major challenges in the web service discovery are to give the proper service description in OWL UDDI registry and how to match customer service request with service description. Development of web service matching model proposed in this work uses semantic description web services based on ontology as the basis of service matching. The service matching method composed of four phases. In the first phase, the profile matcher, and in the second phase, the semantic matcher, matching of services is done based on semantic, and third and fourth involve constraint and best service match, based on the constraints given by the service provider services are filtered out and finally best service is chosen and given to service client. Using these descriptions and matching phases, the performance of web service matching can be improved effectively. In future, fault
tolerance can be implemented in the system and Quality of Service is considered for further improvement of system for providing best service based on the match making process.

REFERENCES


Buhwan Jeong, Jungho Jang, Boonserm Kulvatunyou, Jaegyong Chang and Hyunbo Cho, 2008. Discovering and integrating distributed manufacturing services with semantic manufacturing capability profiles, Department of Industrial and Management Engineering, Pohang University of Science and Technology (POSTECH) San 31, Hyoja, Pohang, pp: 790-784.


Georgios Meditskos and Nick Bassiliades, 2010. Structural and Role-Oriented Web Service Discovery with Taxonomies in OWL-S IEEE Transactions on Knowledge and Data Engineering.


Jing Dong, Yongtao Sun, Sheng Yang, 2009. OWL-S Ontology Framework Extension for Dynamic Web Service Composition IEEE Transactions on Knowledge and Data Engineering.


OWL Services (OWL-S) Ontology


Web Service Modeling Language (WSML) Working Group

Web Service Modeling Ontology (WSMO) Working Group