Comprehensive Survey on Semantic Web Service Discovery and Composition

K. Venkatachalam, N.K. Karthikeyan, S. Kannimuthu

Department of IT, Sri Krishna College of Engineering and Technology, Coimbatore, India.
Department of CSE, Karpagam College of Engineering, Coimbatore, India.

Received February 2016; Accepted 18 April 2016; Available 25 April 2016

ABSTRACT

Web Services have acknowledged much interest due to their ability to perform uniform and universal access via Web standards to software components residing on various platforms and written in different programming languages. Web service technologies are failed to make automatic discovery and composition; human involvement and effort is needed which is not suitable in the complex business environment. Semantic web services support the automatic discovery, composition, and execution of web services. This paper reviews the Semantic Web Service Discovery and Composition concepts, analyzes the Semantic Web Service Discovery and Composition from several perspectives and identifies opportunity for future research in this area.

KEYWORDS: OWL, Semantic web, Webservice, Webservice Discovery, Webservice Composition, WSDL

INTRODUCTION

Service Oriented Architecture (SOA) is the paradigm for building applications based on distributed computing methodology which runs over the Internet. Web services are well defined, reusable, software components that can be invoked over the Internet in a programmatic way using SOAP (Simple Object Access Protocol) messaging [42] and developed by SOA paradigm. The primary benefit of web service is interoperability. This is achieved through vendor, platform and language independent way.

Web services are power-driven by eXtensibile Markup Language (XML) and other major technologies: Web Service Description Language (WSDL), Simple Object Access Protocol (SOAP) and Universal Description, Discovery and Integration (UDDI). After developing a Web service, programmers has to create its definition in the form of a WSDL document that exposes the service’s location on the web and the functionality it provides. Then the information about the service is stored in a UDDI registry. UDDI registry allows the users to look for and locate the services they require. Web Service Discovery (WSD) is the process of finding the suitable web service for the given request from service descriptions. WSD can be broadly classified into Syntactic-based, Semantic-based and Hybrid approaches. Figure 1 shows the Taxonomy of Web Service discovery with some of the approaches.

Existing service discovery protocols like UDDI [18], ebXML[53] etc., rely on exact syntactic-based matching techniques. But, the exact matching of patterns or keywords may be too restrictive when applied to the dynamic and heterogeneous environments. To resolve the limitation of traditional service discovery models and improve the effectiveness of discovery, the novel discovery solution is adopted using semantic languages. But, semantic-based approaches do not exploit semantics explicitly. So, hybrid semantic web service discovery method is developed to improve the semantic web service discovery performance by making use of both logic based and IR based matching. To test whether matching engine performs flexible matches (i.e., service that
sufficiently similar), the degree of match is used in semantic web service discovery. Matching degree of every parameter can be either exact, plugin, subsume, or fail with a decreasing quality.

Web Service Composition (WSC) can be broadly classified into Syntactic-based, Semantic-based, Rule-based, Type-based and Contextual or pragmatic approaches [Figure-2]. Existing web service composition approaches pertaining to the syntactic-based uses WSDL descriptions. WSDL messages are simple syntactic descriptions without any semantics. This entails programmers to reach specific agreements on the interaction of web services and makes automatic web service discovery and composition difficult. This problem is resolved by using universal standards for the exchange of semantic data, which makes it easy for developers to combine data from different sources and services without losing meaning. Rule-based approaches are declarative approach where constraints on nodes include both the functional and non-functional specifications. Semantic-based WSC focuses on the behavior

![Fig. 1: Taxonomy of Web Service discovery](image)

<table>
<thead>
<tr>
<th>Degree of Match</th>
<th>Matching Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>If requested service is equivalent to advertised service or if requested service is a direct sub-class of advertised service.</td>
</tr>
<tr>
<td>Plugin</td>
<td>If advertised service is the subsumes requested service</td>
</tr>
<tr>
<td>Subsumes</td>
<td>If requested service is the subsumes advertised service</td>
</tr>
<tr>
<td>Fail</td>
<td>If there is no subsumption relationship between the requested service and advertised service</td>
</tr>
</tbody>
</table>

Table 1: Degree of Match in Semantic Web Service Discovery

![Fig. 2: Taxonomy of Web Service Composition](image)

modeling of services while ignores the syntax parameters. To resolve this problem, type-based approaches are developed. There are three different types of compositional knowledge required for possible service composition, whereas semantic knowledge is domain-specific specialist knowledge that manages the web service composition.

A broad variety of efficient approaches to perform web service discovery and composition have been developed during last year's. These approaches are more or less described in the corresponding literature. To overcome this situation we give a detailed survey on web service discovery and composition.

The major contribution of this paper is threefold:
1. Detailed survey on web service discovery and composition are presented.
2. Comprehensive analysis of web service discovery and composition is presented
3. Scope for future research directions on web service discovery and composition are given.

The rest of the paper is organized as follows. Section two presents a detailed review of literature Web service composition and Discovery. Section three discusses about the analysis of web service composition and discovery approaches. Section four presents the conclusion and possible trends for future research directions

Review Of Literature:
This section starts with various evolutions in distributed computing technologies contributed to the evolution of web
services and presents comprehensive review of web service discovery and composition.

A. Evolution of Web Services:
Providing Interoperability is the big challenges for any distributed computing methodologies. Figure 3 shows the evolution of web services. Distributed Computing Environment (DCE) is the earliest distributed computing methodology which makes use of Remote Procedure Call (RPC) for communication. The main drawbacks of DCE are: 1) It does not provide interoperability 2) it is not portable. Other technologies Java RMI and RMI-IIOP are restricted to java environment which does not apply to components written in other programming languages. CORBA solves the interoperability problem but the notion of IDL language used in CORBA looks like C++ and developers find difficulty in understanding language findings in order to use CORBA. COM / DCOM are best suitable for Microsoft Windows environment and it does not provide interoperability.

Semantic Web Services (SWS) addresses these problems which are built around universal standards for the interchange of semantic data, which makes it easy for programmers to combine data from different sources and services without losing meaning.

![Fig. 3: Evolution of Web Services](image)

B. Web Service Discovery:
A web service discovery process is carried out in three steps. First is to advertisement of web service by developers and provider advertise in public repositories by register the service using WSDL. Second, web service request given by the user.

<table>
<thead>
<tr>
<th>Table II: Techniques for discovering the services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sl.No</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>
7 **context-aware services**  
1) Users can access their services using portable devices according to their needs, position, and execution/environment conditions.  
2) The major requirement for these kinds of services is the dynamic retrieval and interaction with local resources.  

Semantic based services like ontology repositories and inference engines require a large amount of memory and computational resources.

8 **Adaptable Intelligent Discovery of Context Aware Services[6]**  
A middleware-level approach to support user-centric semantic service discovery.  
This approach make use of context-awareness based on user/device/service profile metadata to provide tailored views on services of interest, and supports intelligent matchmaking between requested and offered service capabilities using semantic-based approaches.

9 **semantic annotations for web services discovery[28]**  
The approach makes use of an interconnected network of semantic web services describing in OWL-S.  
This approach exploits advantages from a graph structure, chaining algorithm of expert system and Semantic annotations.

10 **PYRAMID-S[54]**  
1) Organizes the Web service registries based on domains.  
2) The feasibility is demonstrated through the implementation and experimental evaluation of a model.  
1) Discovery of valuable web services is tough and imperative task in several domains.  
2) Contemporary approaches for service discovery are based on matching of service retrieval.

11 **BPEL ranking based on graph matching**  
1) When there is no exact match found BPEL transform to behavior graph were nodes connectors are split and join.  
2) Use top down approach and apply transformation procedure.  
3) Ranked in decreasing order to calculate distance.  
Linking of nodes and finding distance between them is critical

12 **SPARQL (SPARQL Protocol and RDF Query Language)[39]**  
1) Used for the description and discovery of Semantic Web services  
2) Formal language to describe the preconditions and post conditions of services, as well as the goals of agents.  
The main advantage of this combined approach is to enhance the discovery with services that cannot be retrieved using only Logic-Based Reasoning (LBR).

13 **ADWebS[49]**  
Automatic discovery of semantic Web Services, which can be considered as an extension to one of the most common frameworks for semantic Web service, WSDL-S.  
1) Semantic annotations of Web service to provide the functional description of the services in WSDL’s < document > tag.  
2) Semantic relatedness between terms and pre-defined categories is calculated using Normalized Similarity Score (NSS).

14 **ontology bootstrapping process for web services[9]**  
1) This methodology is based on analyzing web services from multiple perspectives and integrates the results.  
2) Approach facilitates the automatic construction of an ontology that can help, classify, and retrieve relevant services, without the prior training required by previously developed methods.  
This leads to the reduction of effort in ontology construction and maintenance.

Service matcher is core part of web service discovery model, matches user request with available web service. Finally selection and invocation of the services. The different techniques to discover the services are tabulated in Table II.

C. **Web service Matchmaking:**  
Web service matchmaking is the process of finding the existing valuable services based on the functional and nonfunctional descriptions. Matchmaking process can be broadly classified as four types namely 1) Functional 2) Non-Functional 3) Hybrid and 4) Rank-Based, which is shown in Figure 4.

Though UDDI registry has many features, it incurs high precision and recall errors in search mechanism and it fails to provide semantic description of its content. So we rely on semantic based initiative which makes use of capability-based search will overcome the limitations of UDDI and would yield better search results.

D. **Web service Composition:**  
Web service composition uses collection of services which are all combined to attain a consumer’s request. That is the process of combining or bundling the services to meet the needs of a user. Automating this process is desirable to improve the speed and efficiency of the user responses. There are significant researches done on webservice composition.
Fig. 4: Classification of Matchmaking Techniques

Table III: Matchmaker’s techniques for service matching

<table>
<thead>
<tr>
<th>Sl.no</th>
<th>Name of the techniques</th>
<th>Features</th>
</tr>
</thead>
</table>
| 1     | OWL-S matchmakers      | 1) Logic based semantics matchmakers for OWL-S service are OWLSM and OWLs-UDDI [26].  
           2) PCEM converts OWL-S service to PDDL for matching |
| 2     | WSML Matchmakers       | This encompass with WSMO-MX(hybrid semantic matchmakers),GLUE(logic based)syntactic search engine in p2p networks. |
| 3     | WSDL-S/SA WSDL Matchers | 1) This kind of matchmakers uses METERO-S WSDL discovery infrastructure and UDDI based lumina3  
           2) It is semantic base web searching supports service description format [11]. |
| 4     | Monolithic DL based Matchers | Service matching is done within the logic theory performed by RACER that is implemented now in OWL-DL extended with non-functional. |
| 5     | Semantic Advanced Matchmaker | Support PE (Precondition Effect) and proposed an algorithm to provide ordered ranking based solution for unusual needs of PE (expectations of the customer) with well-defined semantics[10]. |
| 6     | SAWSDL-I Matcher       | 1) SAWSDL is the extension of WSDL, used to annotate the existing web services described in WSDL with semantics in a perceptive and low-cost way.  
           2) SAWSDL-iMatcher which uses iXQuery (an extension of Xquery) with various similarity joins for SAWSDL service discovery.  
           3) The SAWSDL-iMatcher supports four kinds of matching strategies such as Syntactic matching strategies, Description-text-based matching strategies, Semantic-annotation-based matching strategies and Statistical model based matching strategies [15]. |
| 7     | OWLS-TC in Matchmaker  | 1) This avoid the scalability problem when services increase.  
           2) Filters the service repositories by discarding service descriptions that do not refer to any functionality before the actual discovery takes place [32]. |

Table IV: Observed Techniques for composition of the services

<table>
<thead>
<tr>
<th>Sl.No</th>
<th>Observed techniques</th>
<th>Features</th>
<th>Merits/Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ontology-based framework for the automatic composition</td>
<td>A model which provides a set of compos ability rules that compare syntactic and semantic features of Web services [13].</td>
<td>Experimental results that proposed approach scales well and requires less time.</td>
</tr>
<tr>
<td>2</td>
<td>Active SWS (ASWS)</td>
<td>Used to interact with ordinary human users and other services and able to reasoning about their actions at runtime so as to compose autonomously and automatically.</td>
<td>It overcomes the problem existing of SWS composition (i.e.,this address the automated integration and composition of services functionalities and no need to perform the manual steps for composition of services[57].</td>
</tr>
</tbody>
</table>
| 3     | Dynamic based composition | 1) Planners allow to execution of information gathering with no state altering services.  
           2) The system finds similarities between the inputs and outputs of different web services with the help of trace data available from execution of a set of web services [58]. | It infers compos ability and substitutability by observing actual executions of the web service. |
| 4     | IDE for end-to-end composition of Web services | 1) Differentiated the web service types and instances using the tools E-flow, Self-Serv, METEOR-S, SHOP2 and SWORD.  
           2) Includes Web service requirement, service discovery, service selection, service discovery, | Proposed IDE supports composition of web services using both semantic web service (like OWL, OWL-s) and distributed programming techniques (WSDL, BPEL)[25]. |
service aggregation, deployment and composition.

The one best matched Web service for a node instantiation when compared to the customary approaches.

Easy to use interface for web services and the complexity of web service workflows [51].

The main drawbacks of web service composition are the lack of standards, open and lightweight test environment which makes comparison and evaluation composition methods impossible [59].

Experimental evaluation shows that when number of services increases, composition framework will dramatically decrease its performance [50].

Experimental evaluation is done by comparing with other type-based methods such as Matching-based approach (SM), the cosine-similarity-based approach (CS). The result shows that Graph-based Approach (GA) outperforms well [35].

Experimental results show that the proposed framework allows composing web services quickly with all of the characteristics [47].

Incorporation of semantics can facilitate automation of both service discovery and composition by eliminating syntactic barriers. The primary advantage of this approach is that composition process as well as discovery of atomic services take part in the composition, are considerably made easy by incorporation of semantic information.

Analysis:
This section presents analysis pertaining to the Web service Matchmaking / Discovery and Web service composition

A. Web service Matchmaking / Discovery:
Evaluation of matchmaking / discovery approaches should clearly point out the factors like runtime performance, scalability, reusability, complexity of descriptions required and the level of guarantees provided by a match. Based on the review of variety of research works, the performance of the Matchmaking / Discovery is evaluated by using runtime, Framework tool support and usability, Expressivity of formalism and matchmaking and Supported level of decoupling. Quality of Matchmaking / Discovery process is investigated by using precision, recall, F-Measure, Fallout, average precision and average recall. Table III and IV illustrates complete analysis of some of the significant approaches related to the web service composition and discovery

B. Web service Composition:
There are many research activities done on web service composition which will leads to the development of the Web Service Composition standards, frameworks and platforms. Recent researches have witnessed
numerous web service composition standards. After investigating these standards, it is concluded that semantic-based composition standards support automated discovery, composition and execution of web services in an effective way. When taking web service composition platforms, generally they are classified into commercial and open source platforms. Each group has its own advantages. Commercial platforms offer a set of tools to implement needs of optimization, security, process adaptation and simulation. Open source platforms are more flexible and free, which paves the way to add new features. Quality of web service composition method is best evaluated by Composition execution time, composition completeness (CC) ratio, Scheduling time and Scalability.

**Conclusion And Future Work:**

Web Services are reusable software components that perform specific, encapsulated tasks via standardized web oriented mechanism. This paper has reviewed and analyzed the current research literature on the various methods for web service composition and discovery. The characteristics of several web service composition and discovery methods are analyzed.

The following are some possible future research directions in web service composition and discovery.

1. Quality of Composition (QoC) can be evaluated by developing the QoC prediction model. This model will be used to optimize the web service composition.
2. Web service composition model can be constructed by considering the additional semantic features such as temporal and spatial availability of Web services.
3. Web service discovery can be done by using NLP. The advantage of using natural language queries is that it can lead to wide applications like supporting multiple spoken languages, supporting voice oriented querying, etc. User is also freed from the burden of learning query languages.

**REFERENCES**


37. Kun Yue, Mingliang Yue, WeiYi Liu & Xiong Li, 2010. 'A Graph-Based Approach for Type Matching in Web Service Composition', Journal of Computational Information Systems, 6(7): 2141-2149.


