SEMANTIC WEB SERVICES – DISCOVERY, SELECTION AND COMPOSITION TECHNIQUES

Sowmya Kamath S[1] and Ananthanarayana V.S.[2]

Department of Information Technology,
National Institute of Technology Karnataka, Surathkal, Mangalore INDIA
[1]sowmyakamath@ieee.org
[2]anvs@nitk.ac.in

ABSTRACT

Web services are already one of the most important resources on the Internet. As an integrated solution for realizing the vision of the Next Generation Web, semantic web services combine semantic web technology with web service technology, envisioning automated life cycle management of web services. This paper discusses the significance and importance of service discovery & selection to business logic, and the requisite current research in the various phases of the semantic web service lifecycle like discovery and selection. We also present several different composition strategies, based on current research, and provide an outlook towards critical future work.

KEYWORDS

Semantic Web Services, service discovery, service composition, SOA, Semantic Web

1. INTRODUCTION

The current trend in web evolution is the web as a provider of services. W3C defines a web service as a self described application that uses standard Internet technologies to interact with other web services. Current web service standards are primarily based on enabling the interoperability of heterogeneous software components over the web and rely on the W3C standard - Extensible Markup Language (XML). Service designers describe services using natural language, e.g. WSDL descriptions, which is often too imprecise and inadequate. Hence, essential service life cycle phases like discovery, execution, and compositions require manual effort and are at best semi-automated.

The Semantic Web [1] is a vision in which data on the World Wide Web is annotated with semantics that can be automatically processed by machines, so that software agents can understand and process information that need manual effort by humans at present. McIllraith et al [2] proposed extending this concept to the domain of Web Services. They proposed that by including semantics during service creation or in service descriptions can result in easier B2B integration and better automation in service discovery & composition. Various initiatives by both academia and industry are under way to standardize and realize these concepts.

Jan Zizka (Eds) : CCSIT, SIPP, AISC, PDCTA - 2013
pp. 151–158, 2013. © CS & IT-CSCP 2013 DOI : 10.5121/csit.2013.3616

Web service discovery is an important aspect of the web service framework but the current method for discovery is matching the service request parameters to the designer defined keywords in service descriptions. This has serious limitations due to natural language descriptions, which may or may not be available. In real UDDI registries, service publishers leaving the WSDL service description elements blank is a common occurrence. Since keyword matching based approach depends on service publishers to provide a well-written WSDL service description for each Web service they publish, this is also its most serious flaw. Semantically enhancing Web service descriptions with ontologies help overcome this drawback. Since ontologies and web services are developed independently the service request and advertisement can be annotated with multiple ontologies, thus facilitating better efficiency.

Implementing the business logic of certain client requests may involve the invocation of operations offered by other web services. A service implemented by combining the functionality provided by other web services is a composite service and the process of developing a composite web service is referred to as Service Composition. Protocols and languages used to address issues of service compositions must be supported by web services composition middleware. It is impossible for a single service to perform the complex tasks required by user requests; hence service composition is unavoidable and likely.

This paper provides an overview of the concepts and most recent technology developments in the area of semantic web services and is structured as follows: Section 2 introduces Semantic Web Services, and Section 3 focuses on the process of Service Discovery, Selection and Invocation. Section 4 presents most recent developments in Service Composition of available semantic web services based on business requirement. In section 5, we discuss Service Execution and integration issues. Section 6 highlights the inherent challenges in this area, followed by Conclusion and References.

2. SEMANTIC WEB SERVICES

The earlier web service technology stack provides only a syntax based operation thus limiting the actual Web service usage to a certain extent. The emerging concept of Semantic Web Services aims to provide a technology to allow for the inclusion of semantics as envisioned by the concept of the Semantic Web. Using the AI concept of ontologies as the underlying data model and other semantic description frameworks like rules and deductive reasoning, the process can be automated for service discovery, selection, composition, and execution [2].

Semantic Web Services (SWS), like traditional web services, are the server components of a client–server system that enable machine-to-machine interaction on the Web. They use markup, which makes business data machine-readable in a complete and refined way. Semantic Web services define a standard way for semantic data interchange, due to which data from different sources and services can be combined or reused.

Semantics in a web service is necessary in order to handle the difficulties faced in seamless integration because of the diverse terminology used in different related services. Hence semantics plays an important role in the life cycle of a web service. Fig 1 depicts the role of semantics in the various stages of the web service lifecycle.

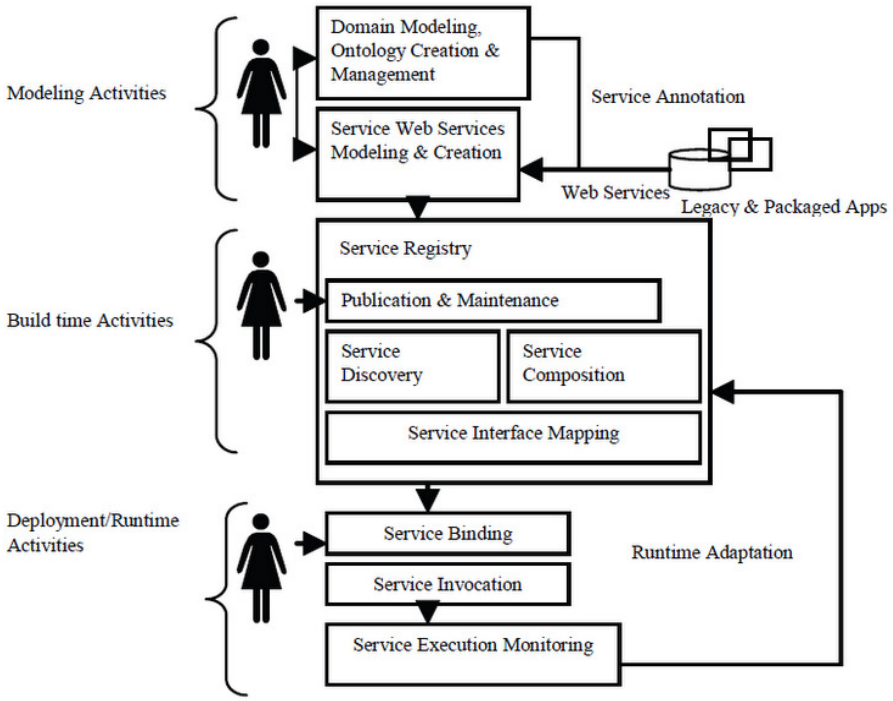

Figure 1. Semantics in the lifecycle of a Web service [3]

3. SWS DISCOVERY, SELECTION AND MATCHMAKING

The complexity of web services vary in function from simple applications such as weather reports, currency convertors, credit checking, credit card payment, etc to complex business applications like those of Online Book stores, insurance brokering system, online travel planners etc. Currently, three methods of web service discovery have been identified - availability of web services in centralized repositories such as UDDIs (Universal Description, Discovery and Integration), in specialized Web portals over the internet (e.g. Xmethods [4], webservicex [5], webservicelist [6] etc), and by customized searches using a search engine. Discovery of web services is an important issue from user's perspective, since it is the first step in service consumption.

Various standards such as WSDL, SOAP, and UDDI have been developed to support discovery of web services. UDDI is a centralized repository that constitutes metadata about web services and advertises requirements to service providers. Each business registered with UDDI categorises all of its web services according to a defined list of service types called taxonomies [7]. The UDDI search is based on metadata of services like service name, providers name and t-Model name. Hence UDDI bridges the gap between service providers and consumers, helps in discovery and invoking of services through a public or private dynamic brokerage system [8]. It also provides search facilities to users in order to invoke published services, based on keyword matching. Therefore, these standards are purely syntactic in nature, resulting in an inefficient search mechanism.

An alternative mechanism to discover web services is through web crawling. The discovery mechanism of web service access points is no longer available through public service registries (i.e. UDDI) [9] Currently, web services are more geared towards B2B integration and for communication between trusted partners, for which a private/shared registry is more suited than a

public registry. Also, problems like centralized node failures and bottleneck issues in registries faced while using a centralized repository like the UDDI make searching of web services through web crawling more attractive. Web search engines can be effectively and efficiently used as an alternative to web service registries as proposed by Al-Masri et al [10]. Other approaches are crawling of web services based on service descriptions and using ontologies in the discovery process [11].

Service discovery can help identify a suitable service at a semantic level. For example, it can help identify a room or flight availability service that satisfies most of the general parameters in the users' request. However, in order for the application on the requester's side to invoke the chosen service automatically, a mechanism that provides a detailed matching may be required to identify the actual interfaces of the required services [12]. Web service invocation involves creating these interface mappings from the request service to the chosen service. The vision of semantic web services for service invocation is that tools built using semantics can help reduce the burden of manually writing interface mappings as is the case with traditional web services.

4. SWS COMPOSITION

The basic web services infrastructure is adequate enough to implement simple interactions between a client and a web service. However, in a real world scenario, the implementation of a web service's business logic involves the invocation of other web services, thus requiring a combination of the functionalities of several web services. This is termed as a composite service [13]. The process of developing a composite service in turn is called service composition. A service composition has to define the order in which it invokes components. In most cases, the invocation order may be sequential, but can also include further processing based on some precondition as required by the business logic. In some cases, this order may also be random if specific ordering is irrelevant to implement the business logic in question.

When composing web services, the business logic of the client is implemented by several services. This is similar to workflow management, since the application logic there is also realized by composing autonomous legacy applications. This allows us to define increasingly complex applications as required by continuously evolving end-user requirements and business logic [14]. As different composed services work together, the output data of one may be the input for another. For example, retrieving the order details of a customer requires an order id. Once the order is retrieved, it has to be passed on to the service component that handles the billing and shipping.

Composition approaches can be categorized as manual, semi-automatic, and automatic. In manual composition, if some user task requires a composition of Web services that must interoperate, then the user must choose the relevant Web services, manually specify the composition, make certain that any software for interoperation is custom-created, and provide the input at required stages (for example, selecting a hotel from among several options). Manual Web service composition from scratch can be difficult and time consuming. Therefore the emphasis is on new functionalities that support dynamic and automated tasks such as discovery, selection and composition.

With semantic markup of Web services, the information essential for selecting, composing, and providing the services is provided at the sites where the service descriptions are made available. With automated composition, the end user or application developer specifies a goal (a business goal expressed in a description language or mathematical notation) and an "intelligent" composition engine selects adequate services and offers the composition transparently to the user

[14]. The main challenges are in identifying and selecting candidate services, composing them as required by the business logic, and analyzing the extent to which they match a request.

Composition strategies can be broadly classified into uninformed, heuristic, evolutionary and QoS based approaches. We present a few –

4.1 Uninformed Approach

Uninformed approach is the most general and straight forward approach to WS composition. Uninformed search algorithms do not make use of any information. There are many ways to develop algorithms based on such approach. One such composition algorithm based on iterative deepening depth-first search [15]. In this algorithm, for each service s in the set of all services S, a requirement set is defined and tested against to determine the suitability for selection for composition. This is continued till S becomes an empty set Ø, we have now found a valid composition and can return it, else the search terminates. As this is based on the exhaustive search strategy, this method is slow and memory consuming for bigger repositories since it does not utilize any additional information about the search space. It is well suited for smaller repositories when the composition problem requires an exhaustive search.

4.2 Heuristic approach

Use of additional information about the search space can increase the efficiency remarkably. In an informed search, a heuristic c helps to decide which nodes are expanded next. If the heuristic is good, such algorithms may dramatically outperform uninformed approach. A greedy algorithm can be defined that internally sorts the list of currently known candidate compositions in descending order according to a heuristic in form of a comparator function c. One such efficient function combines the size of the set unsatisfied parameters, the composition length and the number of satisfied parameters [16].

In order to select the best one if several choices are available, we compare the number of required parameters. If a composition has no unsatisfied concepts, it is a valid solution. If both candidates, S1 and S2 are valid, the solution involving fewer services wins. Only if both of them have the same number of satisfied parameters, we again compare the required concepts. If their numbers are also equal, preference is given to the one with the shorter composition. This approach is fast but not accurate. Superior performance could be measured by utilizing problem specific information encapsulated in a fine tuned heuristic function.

4.3 Evolutionary Approach

This approach defines meta-heuristic optimization algorithms that use biology-inspired mechanisms like natural selection, mutation, survival of the fittest etc [17]. Once a proper genome that correctly represents the service sequences is defined, standard creation, mutation, and crossover operators can be applied for better performance. The two factors that need to be considered here are the composition size and the number of unsatisfied parameters. It has been found that the Evolutionary approach is slower but always finds the correct composition to all requests.

4.4 QoS Based Approach

Considering Quality of Service parameters in SWS composition is a non-functional aspect of a composition algorithm. The defined QoS parameter set represents a name/value pair of some required QoS characteristics such as performance, cost, or reliability. Use of QoS parameters in

composition enables QoS-aware SWS selection and composition, thus addressing the quality based user requirements. However, it requires dynamic composition as all parameters needs to be updated regularly, which is an additional overhead, but provides a better selection of services. Since a composed service uses other services to form itself, its quality also depends on the quality of the services it is composed from [18].

Several composition strategies use the QoS based approach. Approaches using Causal Link Matrices use functional as well as non-functional properties to find the best composition [19]. The CLMs contribute to the automated process of Web service composition by classifying Web services according to a formal link called "causal link", which is similar to a logical dependency among input and output parameters of different Web services.

5. SWS INVOCATION AND INTEGRATION

Service execution comprises of all the activities that need to be carried out at runtime to invoke one or several Web services in a coordinated manner. These activities include commencement, control and verifications of service invocations. Since each service affects the outcome simply by implementing its functionality, it is essential to the service user that the service provider warrants that certain properties for execution are guaranteed [20]. Two of the most important properties are guaranteed termination and dependability. This is imperative since sustaining a consistent state before and after execution even in the presence of failures is essential. Those aspects are very important since execution is in distributed environments where more than one software entity might be involved, and in the execution of composite services.

The distributed approach for service execution di□ers from the centralized approach, in the sense that at runtime execution is not limited to being handled by just one execution agent but might involve several distinct agents. This fundamental expansion results in several advantages but brings in new characteristics that need to be tackled in order to retain reliable execution. Semantic Web services offer the promise of automating the task of integration, which could potentially save development time and reduce implementation costs. However, these claims are yet to be verified in rigorous benchmarking exercises by applying the technology to real-world scenarios.

6. RESEARCH CHALLENGES

Semantic Web services pledge the automation of core Web service life cycle tasks. They are predicted to aide in seamless interoperation between systems, so that manual effort is reduced. Nevertheless, Semantic Web services have not yet been adopted by the industry. There may be several reasons for this. One of the major challenges for the Semantic Web in general and for Semantic Web Services in particular is the unavailability of semantically annotated content for use. Currently, there is little Semantic Web content available. The task of annotating all web data is currently quite daunting, due to which current efforts in moving towards semantically enriched applications has been limited to a few domains where research is more active. Existing web content like static HTML pages, XML documents, multimedia etc should be upgraded to semantic Web content [21]. Since current Web services are mostly WSDL specifications, SWS can be derived from them by having the ability to automatically convert WSDLs into their semantic counterparts like SAWSDL or OWL-S.

Also the current web service scenario is predominantly WSDL based, and public UDDIs to discover these published services are not available. The only way now is to crawl the Web for WSDL specifications and metadata about services and use these to our ends. This introduces other potentially huge challenges – reducing the result set sufficiently to identify the specific

domain, arriving at the ontology specific for that domain (which may or may not be available) and also finding the other constituents of the requirement in case composition is needed [22].

7. CONCLUSIONS

With the explosion in information sharing through the World Wide Web, the proposed Semantic Web concepts have captured the interest of many researchers. Web service composition is an important technology in domain of Web service which needs automation in service discovery and selection of a candidate set, composition of selected candidate services and execution of the composed service. To turn the Web into the Semantic Web will require a move beyond the data-centric approach of annotating information on Web pages to annotating exposed functionality in Semantic Web services technologies.

REFERENCES

[1] Tim Berners Lee, James Hendler and Ora Lassila, “The Semantic Web”, Scientific American Article May 17, 2001

[2] McIlraith, S.A.; Son, T.C.; Honglei Zeng; , "Semantic Web services," Intelligent Systems, IEEE , vol.16, no.2, pp. 46- 53, Mar –Apr 2001

[3] Cardoso J., “Semantic Web Services: theory, tools and applications., 2007, IGI Global

[4] XMethods, http://www.xmethods.net, May 2013.

[5] Remote Methods: Home of Web Services. http://www.remotemethods.com, May 2013

[6] Web Service List, http://www.webservicelist.com, May 2013.

[7] Vitezslav Nezval and Francois Bartolo, “A Model for Easy Public Searching of Web Services”, IceND 2011, pp. 209-222, Springer-Verlag Berlin Heidelberg 2011.

[8] Celik D, Elgi A, “A Semantic Search Agent Approach: finding appropriate semantic web services based on user requested term(s)”, ITI 3rd International Conference on Information and Communications Technology, 2005.

[9] Chen Wu and Elizabeth Chang, “Searching services “on the Web”: A public Web services discovery approach”, Third International IEEE. Conference on Signal-Image Technologies and Internet-Based System, 2008.

[10] Eyhab Al-Masri and Qusay H.Mahmoud, “Investigating Web Services on the World Wide Web”, WWW 2008, April 21–25, 2008, Beijing, China. ACM

[11] E. Al-Masri, and Q. H. Mahmoud, “WSCE: A crawler engine for large-scale discovery of web services,” ICWS pp. 1104-1111, 2007.

[12] Akkiraju, R., Goodwin, R., Doshi, P. and Roeder, S.: A method for semantically enhancing the service discovery capabilities of UDDI. Proceedings of the IJCAI-03 Workshop on Information Integration on the Web, pages 87-92, 2003.

[13] Alonso, G. and Casati, F. “Web services and service-oriented architectures", Proceedings of the 21st International conference on Data Engineering, 2005. ICDE 2005.

[14] Rosenberg, F., Celikovic, P., Michlmayr, A., Leitner, P., Dustdar, S., “An End-to-End Approach for QoS-Aware Service Composition.”, Proceedings of the 13th IEEE International Enterprise Distributed Object Computing Conference (EDOC'09), Auckland.

[15] Weise, T. et.al., Third International Conference on Internet and Web Applications and Services, 2008. ICIW '08.

[16] Bleul, Weise and Geihs., "The Web Service Challenge - A review on Semantic Web Service Composition", EASST,2009, Volume 17, ISSN 1863-2122

[17] Fischer et.al., "An Evolutionary Algorithm for Automatic Composition of Information-gathering Web Services in Mashups", 7th IEEE European Conference on Web Services, ECWS 2009

[18] Hai Yan et.al., "A novel semantic Web service composition algorithm based on QoS ontology"., International Conference On Computer and Communication Technologies in Agriculture Engineering (CCTAE), 2010

[19] Freddy Lécué and Alain Léger., "A formal model for Web service composition"., Proceedings of the 2006 conference on Leading the Web in Concurrent Engineering: Next Generation Concurrent Engineering.

[20] António Luís Lopes, Luís Miguel Botelho "Executing Semantic Web Services with a Context-aware Service Execution Agent", LNCS 4504: Service-oriented Computing: Agents, Semantics and Engineering, , 2007, Springer Berlin / Heidelberg
[21] J de Bruijn, D. Fensel, M. Kerrigan, U. Keller, H. Lausen, and J. Scicluna., "Modeling Semantic Web Services", Springer, 2011.
[22] Daniel Bachlechner and Kerstin Fink, "Semantic Web Service Research: Current Challenges and Proximate Achievements" IJCSA, Vol. 5, Issue no. 3, pp. 117-140

Authors

Sowmya Kamath S is currently pursuing her doctoral degree under the guidance of Prof. Ananathanarayana V.S in the area of Semantic Web Services at the Dept of Information Technology, National Institute of Technology Karnataka, Surathkal. Her research interests include the Semantic Web, Distributed Computing and Web Technologies.

Prof. Ananathanarayana V.S. obtained his doctorate degree from the Indian Institute of Sciences (IISc), Bangalore, India. He is also a Doctoral Fellow of the University of Newfoundland, Canada in the area of Web Services. He is currently Head of the Department of Dept of Information Technology, National Institute of Technology Karnataka, Surathkal, Mangalore, India. His areas of interests include Databases; Multidatabase based mining and Distributed Computing. He has authored/co-authored more than 200 scholarly articles in international conferences and journals of repute.