UDC 621.93

ADVANCED APPROACH TO WEB SERVICE DISCOVERY AND SELECTION

Dmytro S. Pukhkaiev, Oleksii Oleksenko, Tetiana M. Kot, Larysa S. Globa Alexander Schill
National Technical University of Ukraine "Kyiv Polytechnic Institute", Kyiv, Ukraine
Alexander Schill
Technical University of Dresden, Dresden, Germany

Web Service Composition (WSC) is a process that helps to save much programming and cost effort by reusing existing components – web services. This process consists of two major stages – Web Service Discovery and Selection (WSD, WSS). This paper presents an overview of current state-of-the-art WSD and WSS methods. It also provides an analysis and highlighting of major problems like lack of support of the syntactical description in fuzzy logic algorithms in WSD and complex approach shortage in WSS problem. Moreover WSC approach and SLA-Aware WSC System are presented.

Introduction

Currently, many companies offer their services on the Internet. This creates a demand for tools to perform WSC in particular WSD and WSS.

WSD may be defined as the process of finding a machine-processable specification of a web service that meets certain functional criteria. In this paper, we offer a survey of Semantic Web service discovery approaches and define the main problems that should be solved in existing approaches in order to be used in real world automatic WSC system.

WSS is the next step in performing WSC. Overall goal of WSC is to provide end-user with fully working application, composite web service, which satisfies his needs. Thus, an important aspect is to ensure that the composite web service does not violate any non-functional properties i.e. QoS parameters. Parameters such as response time, availability, robustness, reliability and many others form user's experience and feedback indicating WSC efficiency.

However, despite much research effort, state-of-the-art methods of Web service selection with QoS parameters taken into consideration cannot solve problem of WSS in complex, focusing only on narrow tasks. Such tasks as improving speed of composition [1] or focusing on user preferences[2] are important aspects, but solving one task and neglecting others is a problem which needs to be solved. In this paper an approach that overcomes above mentioned problems, as well as WSC System that performs QoS-aware web service selection are presented.

The remainder of this paper is structured as follows. Existing approaches for WS discovery and their main shortcomings are discussed in Section 2. SLA-aware approach for WS selection and WSC SLA-aware System are introduced in Section 3. Real-world scenario of composite web service development using WSC System is shown in Section 4. Conclusion and future work are specified in Section 5.

Discovery

A WS discovery stage can be basically defined as a match-making process. Match-making is the process of finding an appropriate service provider for a service requester through a middle agent. It consists of three main steps - advertising of web-service specification to middle agent (e.g. UDDI registry [3]); requesting a middle agent for a service provider with best matching service capabilities; matching against the stored WS specification and returning a resulting set of stored specifications.

Definition of Comparative Evaluation Criteria

Growing number of web services and ways to specify their functionality makes their discovery more and more difficult. A lot of algorithms and approaches were proposed to solve this discovery issue. In this chapter criteria, allowing evaluating and comparing them, are introduced.

Criteria were divided in three main groups - quantitative criteria, matching criteria and technology support criteria.

Quantitative criteria are:

1. Response Time. Defines, how long it takes to process a web service discovery query.

2. Performance. The WS discovery stage may be considered as a special Information Retrieval (IR) problem [4]. For IR systems evaluation, two following measures are used: Recall and Precision. Recall is a subset of the relevant documents that are retrieved. Precision is a fraction of retrieved by matchmaker results that are relevant. High performance indicates that discovery algorithm has both high Recall and Precision.

Matching criteria are:

3. Matching Elements. The parts of WS specification that are used in matchmaking process. Possible options are:

- IO: Inputs and Outputs.
- PE: Preconditions and Effects or Post-conditions.
- Non Functional Parameters.

4. Multi Stage Matching. Performing a discovery in several stages, sequentially or in parallel on different elements, followed by merging the results. This approach leads to more accurate results through increasing the matching complexity, which in turn increase the query response time. Thus, it is necessary to achieve a balance between accuracy and response time in such approaches. Some of them allow users to manage the trade-off between accuracy and response time [5].

Technology support criteria are:

5. Support for UDDI. Initially all discovery approaches used UDDI syntax for matchmaking. However, while data in UDDI registries are stored using XML, semantic discovery approach can support UDDI only by combining the ontology matchmaking and UDDI semantic matchmaking.

6. Support for Different Ontologies. Web services are autonomous, heterogeneous and developed independently, using different ontologies in requester and provider sides. Support for Different Ontologies indicates that ontology conversion can be performed and web services with different ontologies may be used [5].

Support of probabilistic languages. In real world systems common issue is incomplete information about the web service functionality and user preferences for service discovery. A solution for this problem may be by using fuzzy, probability, and possibility theory [6]. Support for probabilistic extensions of semantic Web languages like pOWL, fuzzyOWL or pDatalog is needed to compare semantic service annotations under uncertainty and with preferences.

Web service Discovery Approaches

Discovery can be based both on the textual descriptions (Syntax-based discovery), and on the additional semantic descriptions (Semantic-based discovery).

Syntax-based. Syntactic methods search through the text description of a web service, keywords and qualifiers. Non-semantic Web services can be discovered using UDDI [3]. UDDI is an industry specification for describing, publishing, and finding Web services. Using UDDI developers can describe the functionality of their services and specify the technical details about the interaction with them. UDDI also defines a set of Application Programming Interfaces (APIs) that can be used for interaction with stored data.

The main advantage of syntax-based approaches is low response time due to simplicity of used algorithms (in comparison with semantic-based). They also don't require any other specifications except WSDL.

Main disadvantage of such approaches is a necessity of manual selection from search results, which is eliminating the usage of this approach in fully automatic web service composition systems.

Semantic-based. Semantic web service is a "web service which functionality is described by use of logic-based semantic annotation over a well-defined ontology" [6]. Due to the variety of semantic web service description languages and means of service selection, different discovery approaches exist. The main approaches are:

7. Logic-based Approaches. In this category of algorithms standard logic inferences are used. They determine the semantic relations between services on the basis of logical comparison of the service semantic descriptions. Strong mathematical basis makes logic-based approaches much more accurate than syntax-based approaches. Most of the semantic-based algorithms use this type of matching [5].

8. Non-Logic-based Approaches. Using formal logic leads to considerable increase in complexity of the system that makes usage of this approach time consuming with high computational complexity. The Non-Logic-based semantic web service discovery aims to overcome such disadvantages. This category does not make semantic descriptions comparison of services and, instead, rely on such techniques as graph matching, information retrieval and data mining.

9. Logic- & Non-Logic-based Approaches. Usage of exclusively explicit semantics for similarity evaluation in logical approaches makes them inadequate. In such case, some relative services can be dropped from the answer set. To improve it, Non-logic-based approaches using both implicit semantics of services and logic approaches [5] may be applied. The basic idea of the Logic-&Non-Logic-based approaches is that non-logic-based matching techniques may be applied in case of logic based matching failure.

Logic- & Syntax-based Approaches. Approaches from this category use both Logic-based matching and Syntax-based discovery.

Discussion

Results of the algorithm comparison are shown in Table I, from which matchmaking categories may be compared with respect to two criteria: Response Time and Performance.

For the comparison of algorithms by the Performance criterion experiments provided by [7] are used. The Performance has been evaluated based on the Re-

call and Precision measures. Based on these results it may be defined that integration of Logic-based and Non-logic-based methods leads to inclusion of the syntactically similar, but logically disjoint results to the result set. This leads to higher Performance of algorithms. Additionally, generally Logic-based matchmakers result in higher Recall and Precision compared to Non-Logic-based ones. Finally, in most cases, the Syntax-based matching only limits the searching domain [8] that makes no difference with the Logic-based matching in terms of the Performance.

For comparing the Response Time, the results of the evaluation experiments in [9] were mainly used. The Logic-& Non-Logic approach provides the highest Response Time. Better result has the Logic-based approach and the Non-Logic approach provides a significant improvement in speed. Combined Logic- & Syntax-based algorithms have the lowest Response Time, as it allows performing preliminary selection on syntactic description. However, existing algorithms that use such approach have the following disadvantages:

10. Inability to match the parameter PE (Preconditions and Effects), and, as a result, reduced Precision, which reduces Performance.

11. Lack of support of different semantic description standards (see Section II.A).

The most promising of the considered algorithms is FuzMOD [10], since it's the only algorithm that supports incomplete information about the web service functionality and user preferences for service discovery due to the usage of fuzzy logic in algorithm. However, it does not support the syntactical (textual) description discovery, which makes it impossible to work with one of the most common publishing web services technology - UDDI. Thus, the significant task is to develop Logic- & Syntax-based algorithm supporting fuzzy logic.

Selection

Web Service Selection Description

Web Service Selection is a second step in WSC. It starts when the list of web services with functional parameters is already created. Main goal of this stage is to select web services with the best possible non-functional parameters, also called QoS parameters. Violation of these parameters such as performance, reliability, accessibility, availability, scalability, cost etc. can significantly affect run of the application or even fail it entirely. Thus, it is very important to take into account QoS parameters and perform SLA-aware composition of web services [14].

State of the art SLA-aware WSS comparison

Various researches have been conducted to investigate subject of SLA-aware or QoS-aware WSS. These approaches have different goals and view WSS from different perspectives.

Preference-based approach [2] calculates composite service's QoS taking into account price, response time, reliability and reputation. Moreover, it uses coefficients based on user preferences to prioritize or other requirements.

TABLE 1. Comparison of discovery approaches

Approach	**Algorithm**	**Criteria**								
		Quantitative criteria		***Matching criteria***				***Technology support criteria***		
		Response Time	***Performance***	***Matching Elements***			***Multi Stage Matching***	***UDDI***	***Different Ontologies***	***Probabilistic languages***
				IO	***PE***	***Non Functional***				
Logic-based	OWL-S IDE (Srinivasan+ 06) [11]	Average	Average	+	+	-	-	-	-	-
	(Somasundaram+ 06) [12]			+	-	+	-	+	-	-
Non-logic-based	(Li+ 07) [13]	Low	Low	+	-	-	+	+	+	-
Logic- and Non-logic-based	SAWSDL-MX2 (Klusch+ 09) [9]	High	High	+	-	-	+	-	-	-
	FuzMOD (Ngan+ 07) [10]			+	+	-	-	-	+	+
Logic- and Syntax-based	FUSION (Kourtesis+ 08) [8]	High	Average	+	-	+	+	-	-	-

Heuristic approach [15] divides QoS parameters into three groups: additive parameters, multiplicative parameters and attributes aggregated by Min-operator. Also this approach provides SLA monitoring and reconfiguration.

Genetic algorithm [1] uses decomposition of global QoS constraints of composite web service into local ones for every web service. Then, it uses linear search to choose the best simple web service. Two groups of QoS parameters are used: positive (availability and throughput) which are maximized and negative (price and response time) which are minimized. Good performance during runtime is the main focus of this approach. Possibility of monitoring SLA is stated, but no mechanisms are presented.

Breadth First Use algorithm [16] utilizes only response time and throughput. This implies into low quality of the composition. Moreover, monitoring phase is not introduced.

Analysis of these approaches shows that only heuristic and genetic approaches cover sufficient number of QoS parameters. However, they do not support subjective QoS parameters which are necessary to compose optimal composition from user's perspective. Monitoring phase support is also a bottleneck while only heuristic approach is able to perform it.

This comparison has shown that the most reliable is heuristic approach. However, it lacks flexibility, especially in areas of new user-defined QoS parameters, objective QoS parameters support and user preferences. Table II summarizes the results of comparison.

Thus, development of SLA-aware WSS approach which is able to stand up to all the requirements provided in this section is an important task.

Another important issue is to integrate the approaches of WSS and WSD – such combination provides significant step comparing to state of the art methods described above.

Table 2. SLA-aware WSC approaches cosmparison

	Preference-based	**Heuristic**	**Genetic**	**Breadth First Use**
Full stack of QoS parameters	-	+	+	-
Subjective QoS	+	-	-	-
Monitoring	-	+	+/-	-

SLA-aware WSS Method and WSC System

In this subsection general description of SLA-aware WSS method and corresponding software implementation is provided. WSC is broader concept than WSS. SLA-aware WSC System should be able to perform both tasks of WSD and WSS. More detailed description is given in [14], although the main focus is on WSS.

Basic approach consists of 7 steps:

1) extracting of discovery parameters from the workflow design;

2) matchmaking with providers web service specification;

3) generating list of matching web services;

4) extracting input parameters - list of web services which satisfy functional parameters from web service discovery service;

5) utilization of integral indicator of web service quality compliance in order to grade found web services by non-functional parameters;

6) web service selection and composition itself;

7) runtime monitoring and reconfiguration.

Suggested WSC System which comprises WSD and WSS consists of five major blocks: Service Locator, SLA Extractor, Decision Maker, Service Combiner and Service Monitor.

Service Locator block is intended to find web services satisfying functional parameters provided by workflow design stage (BPMN file). This corresponds to the discovery stage of WSC. Found web services are organized into a list, sorted by the integral indicator of web service quality compliance for each activity.

SLA Extractor block extracts QoS information from WS-Agreements and provides Decision Maker module with non-functional parameters values. Decision Maker calculates rankings due to ontology rules considering user preferences provided by the client. These preferences have higher priority than ontology rules. Thus, QoS parameters of composite web service fulfil Subjective QoS parameters support constraint. Service Combiner combines selected services into executive BPEL file.

Service Monitor identifies changes of QoS parameters and reconfigures composite web service in case of their violations.

Integral indicator of web service quality compliance is the key parameter in composite web service evaluation. It shows the ranking of a web service for possible composition options. Thus, WSC System can choose the best composite web service regarding QoS parameters. Comparing to the QoS of a single web service (which is a part of a composite web service) ranking of a composite one is not a trivial task. QoS parameters for a composite service depend on the initial workflow. Calculations of QoS parameters for a composite service are presented in [14].

Applying of user-defined rankings changes the priority of QoS parameters for composition. Thus, client receives a service which satisfies his needs. If client decides not to specify any priority, default values of rankings will be applied.

Integral indicator of web service quality compliance can be presented as:

$$Nf = \text{Operator}(R_i, QoSp_i) \quad (1)$$

where $QoSp_i$ - one of the QoS parameters (e.g.performance, reliability, robusteness, accessibility etc.), R_i - ranking of corresponding QoS parameter. $QoSp_{i\,i}$ has a value from 0 to 1 proportionally to the actual value of the parameter. Operator in context of formula (1) can be overridden by the sum, multiplication, max or power operator, depending on composition pattern of web services (loop, sequence etc.) and QoS parameter [14].

Several workflow and WSC models which provide realization of proposed approach have been developed. Workflow model on design stage is presented in [17]. Workflow model on enactment stage and WSC model are given in [14].

Real-world scenario of web oriented application development using WSC System

This section provides presentation of WSC System and WSC approach based on possible real-world scenario.

Assume that client of WSC System has a goal to develop service, providing customized vacation. Customization in this context means that end user would be able to book a hotel and flight, taxi and tickets to some entertainment events using just one service. Lack of funds, programming skills or time implies into using third party services.

Client has various requirements to his service, e.g. response time, cost etc. After authentication in WSC System, client can start developing his service. BPMN or BPEL file has to be uploaded in order to provide system with workflow information. Next step is to specify QoS requirements. If none were provided, system will eventually find the best possible solution. However, even the most reliable one may not satisfy the expectations of users. Thus, it's strongly recommended to provide application with non-functional parameters values. When QoS parameters are specified, WS-Agreement for the composite service is generated. On Fig. 1 BPMN diagram for Vacation Service is presented. Exact workflows are omitted for simplicity.

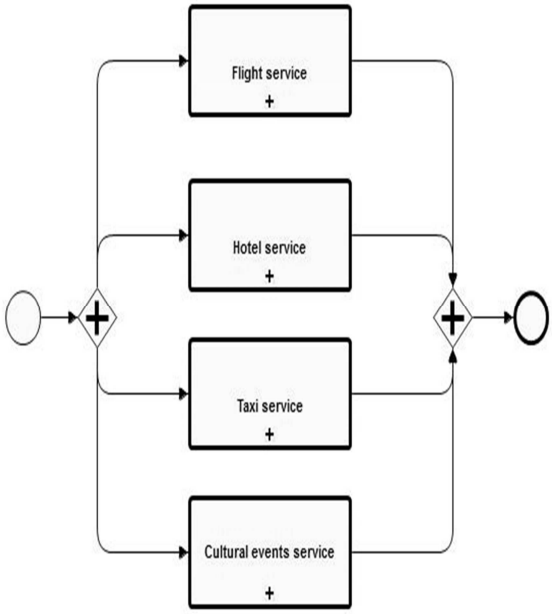

Fig. 1 Simplified BPMN diagram of provider's application

Service locator extracts the information about functional parameters from BPEL-file which was either uploaded or generated from BPMN. It also searches the appropriate services in UDDI or service brokers (considering functional parameters).

Client receives list of composite web services (combination of simple web services) satisfying functional parameters sorted by the integral indicator of web service quality compliance.

In system settings client can choose whether composition will be done automatically or ask for human interaction. Eventually client has to choose the composition that he prefers from the list and confirm purchase of corresponding web services. After this client has a functioning composite web service.

During runtime Service Monitor identifies changes of QoS parameters and reconfigures service in the way similar to initial WSC described above or asks for human interaction.

In case of violating functional parameters, composite web service is recomposed from scratch. Such state of the service cannot be allowed, because service does not provide declared functions. Eventually, the end user works with web interface where all single services are combined transparently.

Conclusion

Web Service Composition consists of two major blocks: Web Service Discovery, finding web services satisfying functional parameters and Web Service Selection, choosing the best possible combination of web services regarding functional parameters.

Despite much research effort still many problems exist. WSD problem is lack of the syntactical description support in fuzzy logic algorithms. WSS problem is narrow task focusing and thus neglecting of other important aspects.

Presented SLA-aware WSC System is able to solve problems of web service Discovery as well as web service Selection. SLA-aware WSC System covers such aspects as full stack of QoS parameters support, subjective QoS i.e. user preferences, and monitoring stage support.

Another important issue is synchronous utilization of WSD and WSS. It means that presented approaches are fully compatible. Thus, SLA-aware WSC System is able to perform full WSC.

Future work is aimed on integration of WSD fuzzy logic approach with support of syntactical description into the overall system. After the integration, comprehensive system testing will be applied and quantitative results provided. Also tutorial for WSC System is to be written.

References

1. Mardukhi, F., NematBakhsh, N., Zamanifar, K., Barati A.: *QoS decomposition for service composition using genetic algori*thm. In: Applied Soft Computing, Volume 13, Issue 7, pp. 3409--3421.
2. Wei Z., Junhao W., Min G., Junwei L.: A QoS preference-based algorithm for service composition in service-oriented network. In: Optik - International Journal for Light and Electron Optics, Vol. 124, Issue 20, pp. 4439--4444.
3. Curbera F., Duftler M., Khalaf R., Nagy W., Mukhi N., Weerawarana S.. "Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI," Internet Computing, IEEE, vol. 6, pp. 86-93, 2002
4. Küster, U., H. Lausen, and B. König-Ries, Evaluation of Semantic Service Discovery—A Survey and Directions for Future Research. Emerging Web Services Technology, 2008. 2: p. 41-58.
5. Keyvan M., Suhaimi I., Mojtaba K., Kanmani M., Sayed G. H. T.i, A comparative evaluation of semantic web service discovery approaches, Proceedings of the iiWAS2010, November 08-10, 2010, Paris, France.
6. Klusch, M., Semantic Web Service Coordination, in CASCOM: Intelligent Service Coordination in the Semantic Web. 2008. p. 59-104.
7. Klusch, M., B. Fries, and K. Sycara, OWLS-MX: A hybrid Semantic Web service matchmaker for OWL-S services. Web Semantics: Science, Services and Agents on the World Wide Web, 2009. 7(2): p. 121-133
8. Kourtesis, D. and I. Paraskakis, Combining SAWSDL, OWLDL and UDDI for Semantically Enhanced Web Service Discovery, in The Semantic Web: Research and Applications. 2008. p. 614-628
9. Klusch, M., P. Kapahnke, and I. Zinnikus, SAWSDL-MX2: A Machine-learning Approach for Integrating Semantic Web Service Matchmaking Variants, in IEEE International Conference on Web Services. 2009.
10. Duy Ngan Le ; Goh, A.E.S. FuzMOD: A Fuzzy Multi-ontology Web Service Discovery System. APSCC, The 2nd IEEE. 2007, pp. 197-203
11. Srinivasan, N., M. Paolucci, and K. Sycara. Semantic Web Service Discovery in the OWL-S IDE. in System Sciences. Proceedings of HICSS '06, 2006
12. Somasundaram, T.S., et al. Semantic Description and Discovery of Grid Services using WSDL-S and QoS based Matchmaking Algorithm. in ADCOM 2006
13. Li, H., X. Du, and X. Tian, A WSMO-Based Semantic Web Services Discovery Framework in Heterogeneous Ontologies iiWAS2010 Proceedings Web Services Environment. Lecture Notes In Computer Science, 2007. 4798: p. 617.
14. Pukhkaiev, D., Kot, T., Globa, L., Schill, A.: A novel SLA-aware approach for web service composition. In: IEEE EUROCON, pp. 327--334 (2013)
15. Berbner, R., Spahn, M., Repp, N., Heckmann, O., Steinmetz, R.: Heuristics for QoS-aware Web Service Composition. In: Proceedings of the IEEE International Conference on Web Services 2006, pp. 72--82. IEEE Computer Society Washington, DC (2006)
16. Aiello, M., el Khoury, E., Lazovik, A., Ratelband, P.: Optimal QoS-Aware Web Service Composition. In: IEEE Conference on Commerce and Enterprise Computing, pp. 491—494 (2009)
17. Kot T., Reverchuk A., Globa L., Schill A.: A novel approach to increase efficiency of OSS/BSS workflow planning and design. In: Proceedings of BIS'12, vol. 117, pp. 142-152

Received in final form April 12, 2014