Advanced Approach to Web Service Composition

Article *in* Advances in Intelligent Systems and Computing · March 2015

DOI: 10.1007/978-3-319-15147-2_29

CITATIONS
4

READS
94

5 authors, including:

Dmytro Pukhkaiev
Technische Universität Dresden
12 PUBLICATIONS **23** CITATIONS
SEE PROFILE

Oleksii Oleksenko
Technische Universität Dresden
17 PUBLICATIONS **157** CITATIONS
SEE PROFILE

Tetiana Mykolayivna Kot
National Technical University of Ukraine Kiev Polytechnic Institute
27 PUBLICATIONS **36** CITATIONS
SEE PROFILE

Larysa S Globa
National Technical University of Ukraine Kiev Polytechnic Institute
149 PUBLICATIONS **185** CITATIONS
SEE PROFILE

Advanced Approach to Web Service Composition

D. Pukhkaiev, O. Oleksenko, T. Kot, L. Globa and A. Schill

Abstract 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 the current state-of-the-art WSD and WSS methods. It also provides an analysis and highlights 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 Service-level agreement (SLA) aware WSC System are presented.

Keywords Web service composition · Web service discovery · Web service selection · QoS · SLA

D. Pukhkaiev · O. Oleksenko · T. Kot (✉) · L. Globa
Information Telecommunication Networks Department,
National Technical University of Ukraine, Kyiv Polytechnic Institute,
Peremogy Ave. 37, Kyiv 03056, Ukraine
e-mail: tkot@its.kpi.ua

O. Oleksenko
e-mail: ooleksenko@stud.its.kpi.ua

D. Pukhkaiev
e-mail: dpukhkaiev@stud.its.kpi.ua

L. Globa
e-mail: lgloba@its.kpi.ua

A. Schill
Faculty of Computer Science, Dresden University of Technology, Nöthnitzer Str. 46,
01187 Dresden, Germany
e-mail: alexander.schill@tu-dresden.de

A. Wiliński et al. (eds.), *Soft Computing in Computer and Information Science*,
Advances in Intelligent Systems and Computing 342,
DOI 10.1007/978-3-319-15147-2_29

1 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 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 nonfunctional properties, i.e., Quality of Service (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, the state-of-the-art methods of Web service selection with QoS parameters taken into consideration cannot solve the 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 the 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 Sect. 2. SLA-aware approach for WSS and WSC SLA-aware System are introduced in Sect. 3. Real-world scenario of composite Web service development using WSC System is shown in Sect. 4. Conclusion and future work are specified in Sect. 5.

2 Discovery

A WSD stage can be basically defined as a matchmaking process. Matchmaking is the process of finding an appropriate service provider for a service requester through a middle agent.

2.1 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 that allow us to evaluate and compare them are introduced.

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

Quantitative criteria are:

- Response time. To define how long it takes to process a WSD query.
- Performance. The WSD stage may be considered as a special Information Retrieval (IR) problem [3]. 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:

- 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.
  - Nonfunctional parameters.
- Multistage matching. To perform 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 [4].

Technology support criteria are:

- Support for UDDI. Initially, all discovery approaches used UDDI syntax for matchmaking. However, while data in UDDI registries are stored using Extensible Markup Language (XML), semantic discovery approach can support UDDI only by combining the ontology matchmaking and UDDI semantic matchmaking.
- 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 [4].
- Support of probabilistic languages. In real-world systems, the 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 [5]. 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.

2.2 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).

Syntactic methods search through the text description of a Web service, keywords and qualifiers. Nonsemantic Web services can be discovered using UDDI [6]. 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 do not require any other specifications except Web Services Description Language (WSDL).

The 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 Web service is a "web service which functionality is described by use of logic-based semantic annotation over a well-defined ontology" [5]. Due to the variety of semantic Web service description languages and means of service selection, different discovery approaches exist. The main approaches are:

- 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 [4].
- Nonlogic-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 nonlogic-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.
- Logic- and Nonlogic-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, nonlogic-based approaches using both implicit semantics of services and logic approaches [4] may be applied. The basic idea of the Logic- and nonlogic-based approaches is that nonlogic-based matching techniques may be applied in case of a logic-based matching failure.
- Logic- and Syntax-based Approaches. Approaches from this category use both Logic-based matching and Syntax-based discovery.

2.3 Comparison

Results of the algorithm comparison are shown in Table 1, 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 recall and

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	DO	PL
				IO	PE	Nonfunctional				
Logic-based	OWL-S IDE (Srinivasan+ 06) [11]	Average	Average	+	+	−	−	−	−	−
	(Somasundaram+ 06) [12]			+	−	+	−	+	−	−
Nonlogic-based	(Li+ 07) [13]	Low	Low	+	−	−	+	+	+	−
logic- and nonlogic-based	SAWSDL-MX2 (Klusch+ 09) [9]	High	High	+	−	−	+	−	−	−
	FuzMOD (Ngan+ 07) [10]			+	+	−	−	−	+	+
Logic- and syntax-based	FUSION (Kourtesis+ 08) [8]	High	Average	+	−	+	+	−	−	−

precision measures. Based on these results it may be defined that integration of Logic-based and Nonlogic-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 Nonlogic-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- and Nonlogic approach provides the highest response time. Better result has the Logic-based approach and the Nonlogic approach provides a significant improvement in speed. Combined Logic- and Syntax-based algorithms have the lowest response time, as it allows performing preliminary selection on syntactic description. However, existing algorithms that use such approaches have the following disadvantages:

- Inability to match the parameter PE, and, as a result, reduced precision, which reduces performance.
- Lack of support of different semantic description standards (see Sect. 2.1).

The most promising of the considered algorithms is FuzMOD [10], since it is 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.

One way for solving this issue is to use hybrid Logic- and Syntax-based algorithm. The main idea behind this algorithm is the separation of description processing—if semantic discovery mechanism fails, basic keyword search is being used, though results of such matching are considered less reliable. For semantic part of such algorithms, FuzMOD may be used. Keyword search may be based on Jaro–Winkler algorithm. It could be enhanced by synonym search using WordNet service. Development of such algorithm is a subject of future work.

3 Selection

3.1 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. The main goal of this stage is to select Web services with the best possible nonfunctional parameters, also called QoS parameters. Violation of these parameters such as performance, reliability, accessibility, availability, scalability, cost etc. can significantly affect the 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].

3.2 Comparison of the State-of-the-Art SLA-aware WSS Approaches

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

The 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 another requirements.

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.

The Breadth First Use algorithm [16] utilizes only response time and throughput. This implies the 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 2 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 unite the approach of WSS with WSD—such combination provides significant step comparing to state-of-the-art methods described above.

Table 2 SLA-aware WSC approaches comparison

	Preference-based	Heuristic	Genetic	Breadth first use
Full stack of QoS parameters	−	+	+	−
Subjective QoS	+	−	−	−
Monitoring	−	+	±	−

3.3 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 a broader concept than WSS. The 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 seven steps:

- Extracting of discovery parameters from the workflow design;
- Matchmaking with providers Web service specification;
- Generating list of matching Web services;
- Extracting input parameters—list of Web services which satisfy functional parameters from Web service discovery service;
- Utilization of integral indicator of Web service quality compliance in order to grade found Web services by nonfunctional parameters;
- Web service selection and composition itself;
- 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—Business Process Model and Notation (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 nonfunctional 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 fulfill subjective QoS parameters support constraint. Service combiner combines selected services into executive Business Process Execution Language (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 the 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:

$$\mathrm{Nf} = \mathrm{Operator}(\mathrm{R_i Q_o Sp_i}) \tag{1}$$

where $\mathrm{QoSp_i}$—one of the QoS parameters (e.g., performance, reliability, robusteness, accessibility etc.), R_i—ranking of corresponding QoS parameter. $\mathrm{QoSp_i}$ 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 the workflow. In particular, the operator depends on composition pattern of Web services (loop, sequence etc.) and QoS parameter itself [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].

4 WSC System

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

4.1 Architecture Overview

Figure 1 depicts the Business Concept Model (BCM) according to [18] approach. This model represents the very basic view on the WSC System. It does not contain any implementation specific information.
The Business Concept Model consists of following concepts.

- User. The client of the system. He is provides an abstract BPEL file to the system.
- Abstract BPEL. A file which is provided by the user of the system, it doesn't contain any specific Web services links, but only the information necessary for the Service Discovery stage.

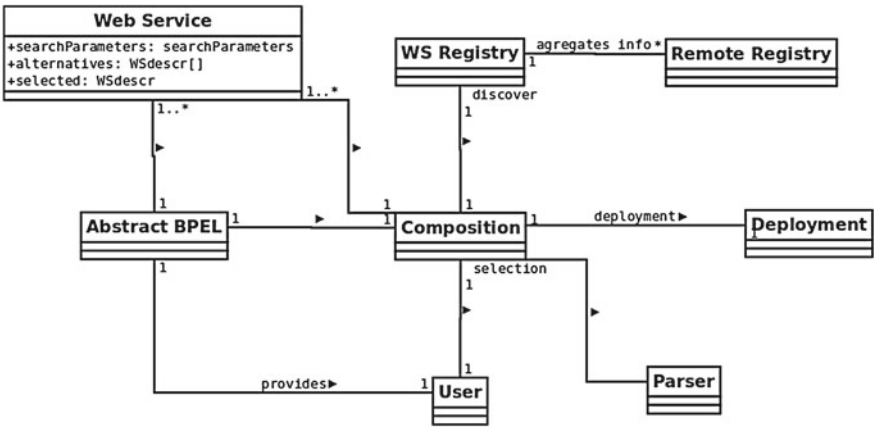

Fig. 1 Business Concept Model

- Composition. The core element of the BCM. It glues other concepts thus being able to provide Service Discovery, Selection and Deployment.
- Web Service. A component of the Composition which is represented by a composite Web service.
- WS Registry. Aggregates and provides a specific view on Remote Registries' Web Services. The core concept of the WSD stage.
- Remote Registry. UDDI or another registry which contains the list of Web services and descriptions.
- Parser. Based on the Service Selection stage parses and generates a concrete BPEL file out of the abstract. Modifies selected services WSDL files in order to use them in composition.
- Deployment. A concept which comprises functions necessary to deploy the composite Web service.

Figure 2 depicts the Business Interface Model (BIM) [18]. This model provides more detailed view on the WSC System. Here only components related to the actual WSC System remained. Core business types of the system are identified in the BIM. This means that these entities can exist independently of the other components' existence. Thus they can be interchanged with other implementations which are able to provide the same functionality. All core types have business interfaces in order not to expose internal structure therefore preserving encapsulation. Finally, the BIM provides key fields of each business type which are required to provide the declared functionality. Figure 3 depicts the Initial Component Specification Architecture (ICSA) [18]. It is defined based on the interfaces of the BIM, one component specification per interface. Since management interfaces were created to manage instances of core business types and their associated details, they are concerned with information that is managed independently. It leads to separate component specifications for Parser,

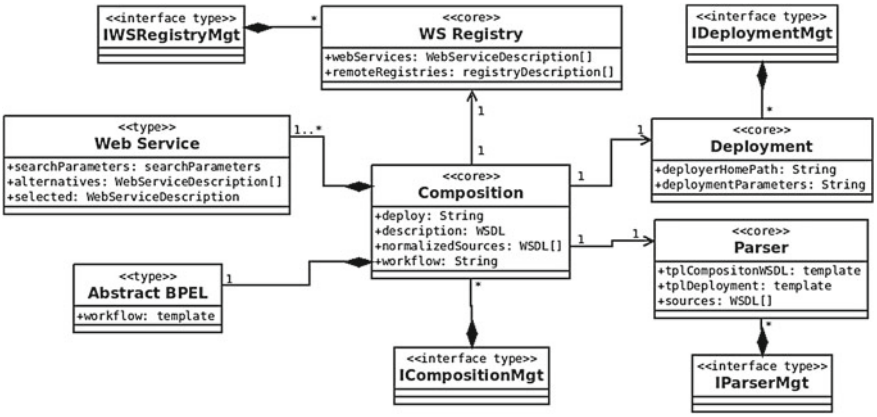

Fig. 2 Business Interface Model

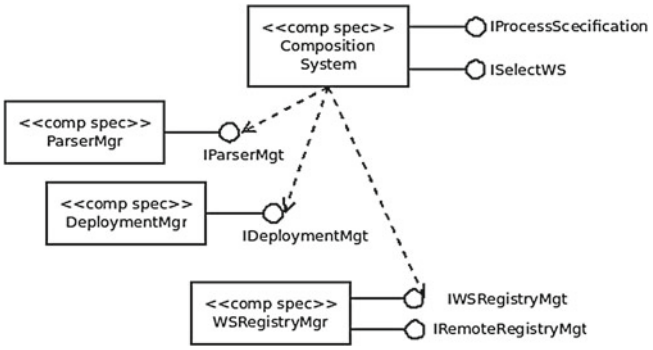

Fig. 3 Initial Component Specification Architecture

Deployment, Web Service Registry, and Composition types. After that these separate components can be bound together into the ICSA.

4.2 Case Study

Assume that the client of WSC System has a goal to develop service, providing vacation. Customization in this context means that the 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.

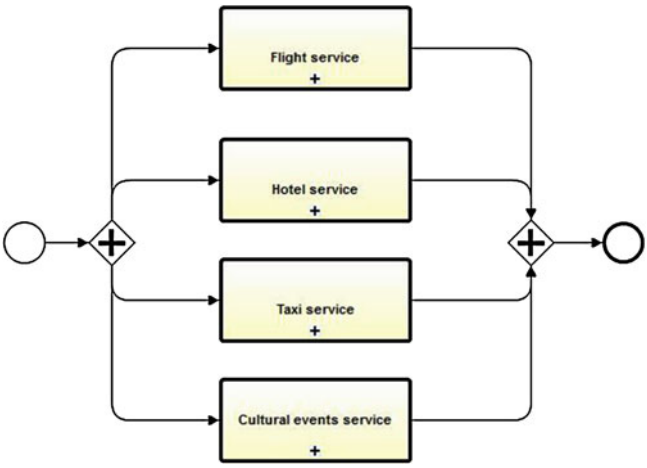

Fig. 4 Simplified BPMN diagram of provider's application

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 users' expectations. Thus, it is strongly recommended to provide application with nonfunctional parameters values. When QoS parameters are specified, WS-Agreement for the composite service is generated. In Fig. 4 BPMN diagram for Vacation Service is presented. The exact workflows are omitted for simplicity.
The 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, the client can choose whether composition will be done automatically or ask for human interaction. Eventually the client has to choose the composition that he prefers from the list and confirm the purchase of corresponding Web services. After this, the client has a functioning composite Web service.

In the 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.

5 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 a narrow task focusing and thus neglecting of other important aspects.

Presented SLA-aware WSC System is able to solve the problems of Web service Discovery as well as Web service Selection. The 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 the 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 the support of syntactical description into the overall system. After the integration, comprehensive system testing will be applied and quantitative results are provided. Also, the 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 algorithm. Appl. Soft Comput. **13**(7), 3409–3421 (2013)
2. Wei, Z., Junhao, W., Min, G., Junwei, L.: A QoS preference-based algorithm for service composition in service-oriented network. Optik—Int. J. Light Electron. Opt. **124**(20), 4439–4444 (2013)
3. Küster, U., Lausen, H., König-Ries, B.: Evaluation of semantic service discovery—a survey and directions for future research. In: Emerging Web Services Technology, vol. 2, pp. 41–58 (2008)
4. Keyvan, M., Suhaimi, I., Mojtaba, K., Kanmani, M., Sayed, G.H.T.: A comparative evaluation of semantic web service discovery approaches. In: Proceedings of the IIWAS2010, Paris, 08–10 November 2010
5. Klusch, M.: Semantic web service coordination. In: CASCOM: Intelligent Service Coordination in the Semantic Web, pp. 59–104 (2008)
6. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI. IEEE Internet Comput. **6**, 86–93 (2002)
7. Klusch, M., Fries, B., Sycara, K.: OWLS-MX: a hybrid semantic web service matchmaker for OWL-S services. Web Semant.: Sci. Serv. Agents World Wide Web **7**(2), 121–133 (2009)
8. Kourtesis, D., Paraskakis, I.: Combining SAWSDL, OWLDL and UDDI for semantically enhanced web service discovery. In: The Semantic Web: Research and Applications, pp. 614–628 (2008)
9. Klusch, M., Kapahnke, P., Zinnikus, I.: SAWSDL-MX2: a machine-learning approach for integrating semantic web service matchmaking variants. In: IEEE International Conference on Web Services (2009)

10. Le, D.N., Goh, A.E.S.: FuzMOD: a fuzzy multi-ontology web service discovery system. In: The 2nd IEEE APSCC, pp. 197–203 (2007)
11. Srinivasan, N., Paolucci, M., Sycara, K.: Semantic web service discovery in the OWL-S IDE. In: Proceedings of HICSS'06, System Sciences (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., Du, X., Tian, X.: A WSMO-based semantic web services discovery framework in heterogeneous ontologies IIWAS2010. In: Proceedings Web Services Environment. Lecture Notes in Computer Science, vol. 4798, p. 617 (2007)
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 (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. Springer, Berlin (2012)
18. Cheesman, J., Daniels, J.: UML Components: A Simple Process for Specifying Component-Based Software. Addison Wesley, Boston (2004)