Trends of Web Service Composition

Khalid Mansour
Faculty of Information Technology
Zarqa University
Jordan
{kmansour@zu.edu.jo}

ABSTRACT: *Web service composition is about maximizing the benefits of different Web services by combining a certain number of Web services to deliver a fully end-to-end service. This paper presents a systematic review of the two major trends of Web service compositions: top-down composition and bottom-up composition. The top-down composition starts with a well defined goal and search criteria. On the other hand, the goal and search criteria in the bottom-up composition are not well defined. The bottom-up composition is often called Web service mining. Web service mining is a new trend that aims at discovering useful and interesting compositions of existing Web services.*

Methods: *Two main web service composition approaches are surveyed.*

Results: *Web service composition mechanisms are presented and discussed in each approach.*

Conclusions: *Different composition approaches may lead to different composite services in terms of functionality.*

Keywords: Web Service, Web Service Composition, Web Service Mining, Interesting Compositions

Received: 19 September 2019, Revised 17 January 2020, Accepted 5 February 2020

DOI: 10.6025/ijwa/2020/12/2/37-46

1. Introduction

Service-oriented computing (SOC) paradigm considers services as the main constituent elements that support low-cost and rapid development of interoperable distributed applications in heterogeneous environments [3] [31].

The technology of Web services allows enterprises to express their internal processes as services that can be accessed by the Internet. Some of the resources of large companies such as Google and Facebook are made available through Web services [29]. Web service technology started as an initiative to solve the problems of interoperability and integration amongst existing Web applications [27]. Many business processes that are implemented by Web services would reduce the cost of building new business applications since the existing Web services can be reused to build new applications.

Web services are distributed computing applications over the Internet that can be accessed via a set of homogeneous XML interfaces. The W3C defines a Web service as"a software system designed to support interoperable machine-to-machine interaction over a network". A Web service consists of a set of computational or physical activities with a number of resources to fulfill customers' needs [29] [39].Web services can be described, published, discovered and interacted through certain Internet protocols. A Web service needs to be described by a service provider and published to a service registry such as Web Services Description Language (WSDL). As an alternative method, a service provider can publish some documents forWeb service discovery such asWeb Services Inspection Language (WSIL) documents. Consequently, other applications can discover and invoke Web services.

Web service composition is a value-added procedure that aggregates different Web service functions and produces a new function(s) that cannot be provided by any atomic or other composite Web services. This paper considers two trends of Web service compositions: top-down and bottom-up composition approaches. The top-down composition trend starts with a well defined goal and search criteria and the goal identifies the functionality of the new composition. On the other hand, the goal and search criteria in the bottom-up composition trend are not well defined. The bottom-up composition is often called Web service mining [39][1]. The two composition trends differ in their composition requirements and mechanisms. In addition, the value of the resultant composition vary: the outcomes of the top-down composition is expected and planned for, while the outcomes of the bottom-up composition is less predictable and aims to find useful and interesting Web service compositions [39].

In order to reduce the size of search space in the bottom-up approach, a general goal can be provided, for example, a general goal for a person is to live a long healthy life, then theWeb service mining seeks to find useful and interestingWeb service composition that can fulfill the general goal. Such compositions can include descriptions for certain life styles, general health related recommendations or advising drinking certain herbal mixes etc.

To the best of our knowledge, this is the first research that reviews the two trends of Web services composition. In addition to briefly introducing the most recentWeb service composition technologies, this paper also aims to accentuate the differences in the mechanisms of both the top-down and bottom-up compositions as well as the expected outcomes.

The rest of this paper is organized as follows: Section 2 reviews the recent methods in the top-down service composition. Section 3 reviews the service mining methods. The last section concludes the paper.

2. Top-down Web Services Composition

The top-down Web service composition is the main trend where most studies addressed this type of composition from different aspects, see [33] [22] [30] [37] [32] [19] [27] . Figure 1 shows a simple Web service architecture showing a Web service provider

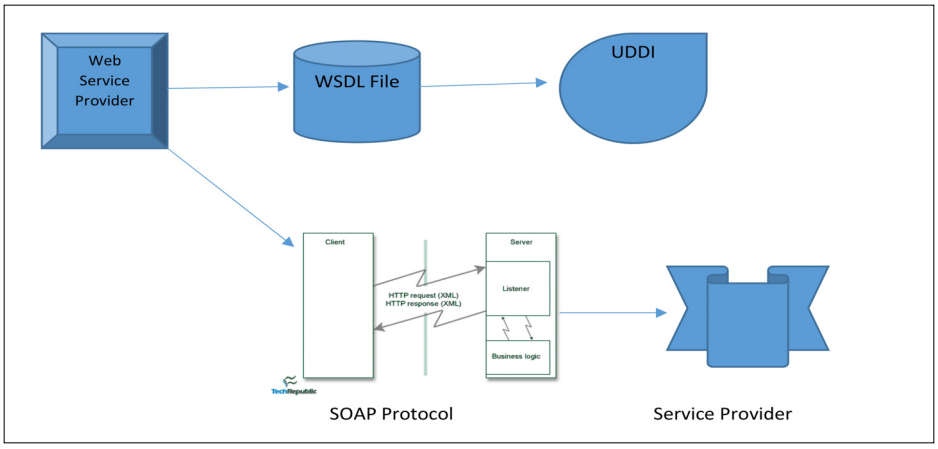

Figure 1. Web Service Architecture

[1]bottom-up composition and Web service mining will be used interchangeably in this paper

sending a UDDI registry information about the available Web services via a WSDL file, a Web service requester contacts the UDDI registry searching for a Web service of a certain functionality. UDDI provides information -if any- to the service requester. Finally, the service requester contacts the service provider via a Simple Object Access Protocol (SOAP) message then the service provider replies with a SOAP message as well, see the numbering sequence in Figure 1. The service provider, the service requester and the service registry are the main three roles involved in any Web service application.

Two main methods can be used to develop a Web service: the SOAP-based Web services (WS-* Web services) and Representational State Transfer (RESTful) Web services that utilizes the REST model [23].

The SOAP-based Web services depends on WSDL, SOAP and the UDDI registry. The WS Web services use SOAP calls to implement service registration, service discovery and invocation. The SOAP-based Web services are generally used for integrating complex applications and hence it consumes large computational power. On the other hand, the RESTful Web services are lightweight, stateless, identified by URIs and can be used for ad hoc integration over the Web. The mashup is a well known method that enables users to create situational applications based on existing application components [21].

RESTful Web services uses a fixed set of operations: GET,vPUT, DELETE and POST.

2.1. Web Service Composition Standards

Two different top-down complementary standards are used to standardizeWeb service composition, the first one targeted the old Web service paradigm and proposed a number of XML-based standards such as BPEL[8]. The second approach uses the concept of semantic Web services and developed standards such as OWL-S [15]. A brief introduction to a few standards of each of the above types is presented next:

• XML-based Standards

1. Web Services Business Process Execution Language (WS-BPEL). WS-BPEL or BPEL[2] for short is an XML-based language for Web service composition. It was appeared in 2004. Different types of primitives are introduced in BPEL: the primitives invoke, reply and receive are used for interactions amongst the Web services under consideration. Other primitives such as wait, assign, throw, exit and empty indicate wait for some time, copy data, error state, termination the current composition and doing nothing respectively. More complex activities can be formed by combining the mentioned primitive activities using structured activities (constructs) such as while, flow and sequence.

2. Business Process Modeling Language (BPML). BPML[3] is a language for business process modeling and it was appeared in 2002. The recent version of BPML includes several concepts of Web Service Choreography Interface (WSCI) that considers the choreography of Web services. The MPML modeling language is similar to BPEL in terms of the basic and structural activities. For example, the basic primitives action, assign, and call are used to invoke services, assign a new value to certain message and instantiate a process a process respectively. Structured activities are also provided such as choice, while and sequence.

3. Electronic Business Using XML (ebXML). ebXML[4] is an international initiative established to enable enterprises of any size to conduct business over the Internet. ebXML consists of four main components: messaging service, registry, trading partner information and Business Process Specification Schema (BPSS). The messaging service component enables exchanging business messages amongst organizations that is independent of any file transport mechanism (e.g., SMPT , FTP) or network type. The registry stores information about businesses. The trading partner information uses the protocol Collaboration Protocol Profile and Collaboration Protocol Agreement to provide an XML definition of a document that contains details of how an organization is able to conduct business electronically and specifying the details of how two organizations have agreed to conduct business electronically respectively. Finally, BPSS provides an XML modeling document that defines theWeb service composition members.

• Semantic-based Standards

Semantics improve software discovery and reuse. In addition, it facilitates composition of Web services and enables the integra-

[2]docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[3]xml.coverpages.org/bpml.html

[4]xml.coverpages.org/ebXML.html

tion of legacy applications. With semantic information annotated the Web services, the interface and function of a Web service is described with more specifications than the ones used by standard Web service technologies, e.g., WSDL. For example, if a service declares that it takes a string as input, it still does not provide enough information. For example, that string can be a DNA sequence, an output report from another program etc. The semantic information would provide such extra information using ontologies [4].The following standards are semantic based standards.

1. Web Ontology Language (OWL-S). OWL-S[5] previously called DAML-S. OWL-S uses semantics to describe and reason services. The OWL-S uses ontologies in describing services. The service profile ontology is used to describe services that facilitates service discover latter. A description of functional and non-functional properties are used in servicedescription and queries. The process model ontology describes both the composition and execution ofWeb services. The grounding ontology describes the accessing a service details.

2. Web Service Modeling Framework (WSMF). WSMF was proposed in 2002 to provide a suitable conceptual model fordeveloping and describing both atomic Web services and composite Web services. Its philosophy is based on maximal decoupling complemented by a scalable mediation service [6]. Its goal is to enable e-Commerce by applying semantic Web technology to Web services. WSMF consists of ontologies, capabilities repositories, Web services descriptions, and mediators. The capabilities repositories define the problems that need to be solved by Web services. The Web services descriptions define various aspects of a Web service. Finally, the mediators bypass interoperability problems. WSMF contains two major projects: the semantic Web enabled Web Services and the Web service modeling ontology.

3. Web Service Semantics (WSDL-S). The current WSDL standard lacks semantic expressivity required to represent Web services since the WSDL works only at the syntactic level. The semantic information specified in the WSDL-S document contains definitions of the input, output, precondition and effects of Web service operations. The WSDL-S is preferred over (OWL-S), for more information see the link below[6].

Using certain criteria presented in [29], Table 1 compares between the six Web service standards presented in this section. The first three standards lacks the semantic support while semantic-based standards lack many of the criteria used by the table.

A. XML Standards
A.1 XML 1.0
A.2 XML Namespaces
A.3 Infoset
A.4 XSD
B. SOAP Standards
B.1 SOAP (1.1 and 1.2)
B.2 MTOM
B.3 WS-Addressing

Table 1. Web Service Standards

2.2. Web Services Composition Methods

Web services composition is a complex task due to many factors: the number of Web services increases rapidly which makes finding new Web services more difficult. In addition, Web services can be created and updated on the fly which requires the composition system checks for new updates at every runtime [27]. Web service composition can be divided into three major categories [34] [35]: Explorative composition, semi-fixed composition and fixed composition. In the explorative composition, once a request from a client is specified, the service composition is created on the fly. In the semi-fixed composition, the service

[5]www.w3.org/Submission/OWL-S/

[6]www.w3.org/Submission/WSDL-S/

composition is specified statically while the real bindings are decided during runtime. The fixed composite category requires that composition structure is pre-specified and the component services are statically bound.

In the following paragraphs we discuss possible services composition methods: Firstly, the static and dynamic compositions are discussed. Secondly, manual, semi-automated, and automated composition types are explained. Finally, the orchestration and choreography types of the sequence of activities that make up a business process are presented.

In static composition which comes under the fixed composite category, the services composition is performed at the design time. The designer selects the services needed for the composition, then bound it together and deploy it, then it can be executed. The disadvantage of this type of composition is that the steps of the static composition is needed to be repeated again in case the functionality(s) of a service(s) in the composition is/are changed or the composition requirement is changed. Since changes in the business environments are expected to occur frequently, using the static composition approach faces difficulties to apply in reality. On the other hand, dynamic Web services composition, which comes under the explorative composition category is a more automated approach where services are determined and replaced at run time. Since the business environment is dynamic, the dynamic composition is more suitable than static composition. The drawback of this approach is that composing service during run time faces some difficulties related to time limits, measuring the correctness of compositions and others, see [11][10].

Manual composition (belongs to the fixed composition category) involves a human designer who needs to create an abstract composite process using certain standard language (e.g., BPEL) then the designer binds the Web services to the abstract process manually. This composition method can be time consuming and error-prone procedure. In addition, as the case with the static composition, the composition needs to be repeated again in case of any change in requirements and/or functions provided by any Web service included in the composition.

On the contrary, the automated services composition (belongs to the explorative composition category) approach isthe complete opposite of the manual composition. It is expected that every step in the automated composition approach be automated. Automated composition approaches are based on the artificial intelligence (AI) planning techniques and the semantic Web. The input this type of composition are a set of Web services and a specified requirements and the output is the composite service that fulfill the composition goal. A fully automated services composition is a challenging task since the selection process can be affected due to the fact that Web services do not share a full understanding of their semantics [9][5].

The semi-automated composition improves the efficiency of the manual composition and at the same time reduce the complexity associated with the fully automated composition. With the semi-automated composition, the user is assisted at each step of the service composition process to inferring the entirety of the desired workflow [4]. The work in [20] presents a framework for semiautomated Web service composition in Semantic Web. The proposed framework allows for providing many composite services using one integrated service while maintaining a merged ontologies repository for the composite services.

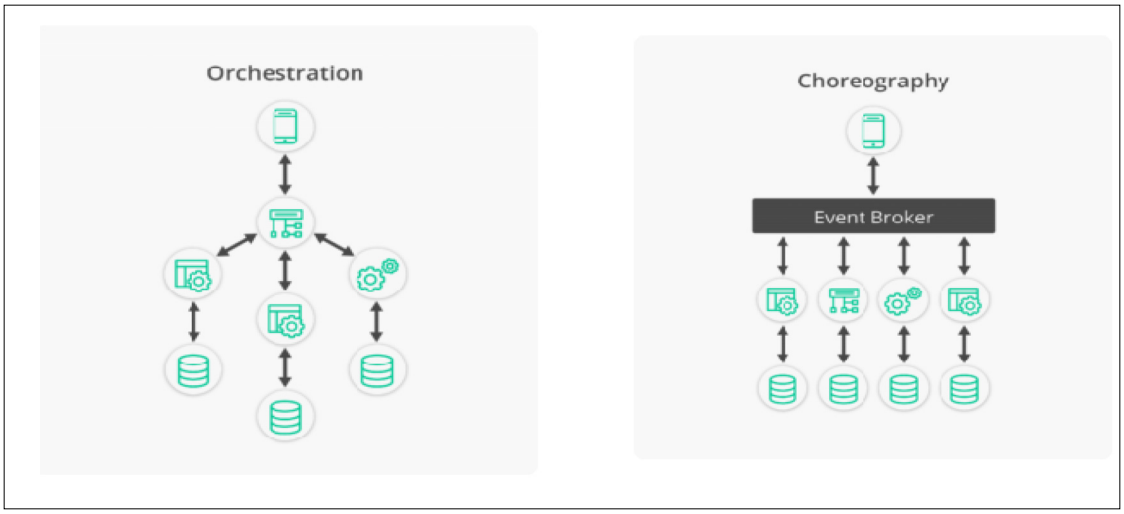

Figure 2. Orchestration and Choreography

Creating business processes from Web services can follow either the orchestration aspect or the choreography aspect [25]. According to [25] "Orchestration refers to an executable business process that can interact with both internal and external Web services" and "Orchestration always represents control from one party's perspective. In other words, service orchestration represents a centralized executable business process that coordinates the interaction among different services. The business executable process is responsible for invoking and combining the services. On the other hand, service choreography is a decentralized approach which provides description of the participating services by defining the exchange of messages, rules of interaction and agreements between two or more services. Figure 2 illustrates the concepts of Orchestration and Choreography.

Choreography involves collaboration between different services since these services comes from different providers.

2.3. Web Service Composition Life Cycle

Utilizing a composite service involves several steps called the life cycle of composite service. Four main are required for utilizing a composite service: the definition phase, the service selection phase, the deployment phase and execution phase [29] [34]. However, the life-cycle presented in [34] have the planning phase as the first step and does include the service selection phase. The following is a summary for each phase:

1. Definition phase. User requirements and preferences for the composite service are specified. The requirement is used to create an abstract process model, i.e., the abstract composite service. The abstract composite service specifies the control and data flow amongst the services, a set of activities, the quality of service requirements (QoS), etc. This phase should meet the expressibility and correctness requirements. The expressibility property indicates that the process modeling language should be capable of modeling complex structures such as sequence and iteration, representing data and specify the data flow amongst activities, supporting exception handling, etc. The correctness property is met if it is possible to ensure that the composite service acts according to its functional and non-functional requirements requirements.

2. Selection phase. This phase aims to find a good candidate service(s) for performing certain function(s). Since there are large number of available Web services, automation the selection phase expedites the process of selecting good candidate services. Service discovery can be based on syntactic matching or semantic matching. To increase the level of automation in the service selection phase, semantic matching is used since it provides more information than just the names and identifiers that are provided in the syntactic matching. Besides automation the service selective process, service selectability property is also important since the results of searching for a Web service can results in multiple services that have similar characteristics. After selection the best Web services, they are bound to their corresponding activities and the composite service is produced.

3. Deployment phase. After constructing the composition in the previous phase, the composed service is deployed to be invoked by users. This phase results in an executable composite service.

4. Execution phase. The execution engine instantiates and execute the composite service. During this phase, certain properties are required, for example the execution of the composite service should be adaptable since certain components of the composite service can change or disappear. One possible solution is to automate the replacement of Web services with others at runtime. Scalability is another important property; when the size of the composite services becomes large, it can affect the execution of the composite. Scalability can be evaluated during the execution of a composite service. The last two important properties are reliability and monitoring. The reliability indicates how robust the composition mechanism against the exceptional behaviors during the execution of the composite service is. On the other hand, monitoring the composite service during runtime is important in verifying the effeteness of the composite mechanism. For example, data related to QoS can be collected during service composition runtime.

2.4. Automation of Web Service Composition

Manual and static composition methods cannot cope with the increased complexity of theWeb services composition process; the number ofWeb services increases rapidly, the existingWeb services can become unavailable at any time, the inputs requirements of Web services may change as well as their outputs, etc. Automation of Web services composition becomes necessary in such dynamic environment. The automation of a process indicates that the process model can be generated automatically or the correct services can be located if an abstract process model is given [27].

Several prototypes are presented in the literature to either semi-automate or automated the composition process. For example, eFlow, Self-Serv and WISE prototypes/ frameworks support the semi-automated service composition. On the other hand,

FUSION, SWORD and ASTRO prototypes support automated service composition. Several platforms for service composition are also available: IBM Business Process Manager, Oracle BPEL Process Manager, Apache ODE and more others. For more details about the available service composition prototypes and platforms, see [12] [29]. The composition strategies used in service composition prototypes are based on workflow composition or AI planning [9] [36] [27].

If the process model is defined then the workflow composition can be used. On the other hand, the AI planning methods are used in case the set of preferences and constraint are available and at the same time, the process model does not exist. Consequently, depending on the AI planning methods, the process model can be generated automatically [27].

According to [16], when the service has an interface containing action definitions that is the representation of how web services actually behave, then interacting with a Web service is considered a planning problem. The AI service composition can be divided into five categories [27]: Situation calculus, Planning Domain Definition Language (PDDL), rule-based planning and theorem proving. The following is a brief summary for each category:

• **Situation calculus.** A logical language for representing changes where situations are the first-order objects which can be quantified over [13]. The work presented in [17] adapted and extended the Golog language to automate the construction of Web services where Golog is a logic programming language built on top of the situation calculus. The requirements and constraints provided by users are presented by the first-order language of the situation language. Each Web service is considered as a PrimitiveAction or a ComplexAction. A Primitive Action is an action that changes the state of the world while a Complex Action is a collection of Primitive Actions.

• **PDDL.** PDDL is a standardized input for the stateof-the-art planners. The language that can be used as a transfer format is supported by a wide range of planning engines [24]. When planning for a service composition, DAML-S descriptions -which is similar to PDDL representation- could be translated to PDDL format. Then different planners can be used for further service composition. A tool that transforms a Web services composition problem into an AI planning problem is presented in [24]. The AI planning problem is then delegated to a suitable planner. The proposed tool uses the PDDL language is used as a transfer format.

• **Rule-based planning.** What matters here is the composability rules that consider the syntactic and semantic properties of Web services. The work presented in [18] uses the composability rules by comparing the syntactic and semantic features of Web services to judge whether two services are composable. Applying the proposed rules results in reaching a meaningful composition. The SWORD tool mention previously is an example for building composite Web services using rule-based planning.

• **Theorem proving.** This approach is based on automated deduction. At the start, the user requirements and the available services are described in a firstorder language, then constructive proofs are generated with SNARK theorem prover. The last step is to extract the descriptions of service composition from certain proofs.

Structural Synthesis of Program (SSP) for automated service composition is used in [14]. SSP is a deductive approach that uses specifications for synthesis. The service composition depends on the proofs-as-programs property of intuitionist logic. Moreover, [26] proposes a method for automatic composition of semanticWeb services using Linear Logic theorem proving. Finally, the work presented in [2] shows that the Linear Logic theorem prover can deal with both the service specification and the semantic Web information.

Other AI service composition methods that do not belong to any of the above categories exist, for example, [7] Case-based reasoning is used in dynamic Web services composition.

At the end of this section, we present two open research issues in service composition technology [12]: Social/ crowd computing support and engineering composite services. The existence of social networks and crowdsouring enable access to large number of individuals. As we know, Humans can perform some computational tasks better than machines such as ranking a number photos or providing an opinion on a given topic. Currently,Web services technology considering machine computations and does account for the specific needs that emerge when humans are involved in applications. New mechanisms are needed to bring together human and machine computations.

The engineering composite services challenge results from using several semantically unrelated notations for engineering

composite services. Further research is needed in the areas of: Unified methods, models and tools.

2.5. QoS Evaluation Criteria

Evaluating a composite service is important since a successful composite needs to fulfill certain functional and non-functional requirements. In addition, the evaluation criteria can be used to compare between different composites. The notion of quality of service (QoS) is usually used to evaluate the non-functional attributes [28]. Several QoS attributes are used to evaluate the non-functional requirements such as availability, response time, security, traceability etc. The research presented in [28] listed 19 QoS evaluation criteria. Certain aggregate functions are used to evaluate a composite service [1]. For example, the summation function can be used with QoS attributes like response time, price and reputation, multiplication function can be used with the availability and reliability attributes and the minimum function can be used with the throughput criterion.

3. Bottom-up Web Service Composition

Bottom-upWeb services composition or Web service mining is a new discipline research area that aims at finding useful and interestingWeb services compositions [38]. Web service mining is defined as "a search process aiming at the discovery of Web services" [39]. Unlike the top-down services composition approach, theWeb service mining composition aims at finding unexpected and interesting compositions starting without a specific goal or search criteria. Web service mining is inspired from the formation of natural and biological molecules where a certain number of atoms under certain conditions recognize each other and forms a molecule.

In case of the top-down composition approach, the more specific the goal and search criteria are, the search space becomes smaller and the relevant the formed compositions are more relevant. On the other, with help of a service mining tool, theWeb service mining task is to discover any interesting and useful service compositions in the available search space. For performance reasons, usually the Web service mining start with vague goal(s) and search criteria [38]. For example, a genome data can be submitted to the service mining tool, the results can be the expected disease(s) that are encoded in the genome and certain recommendations or treatments for the disease(s).

4. Web Service Mining Framework

A framework for miningWeb services is presented in [39]. This framework can be considered as the life cycle ofWeb service mining which has several differences with the life cycle of the top-down composition presented in Section 2.3. The proposed framework uses the sow $\rightarrow$ weed $\rightarrow$ harvest analogy. The services mining fragmroewwo!rk consists of the following phases:

1. Scope Specification. This is the sow phase that involves defining the context of mining by a domain expert. The seeds here refers to area of interest which are the Web services functional areas such as cell enzyme and drug functions.

2. Search Space Determination. To avoid the problem combinatorial explosion, this phase defines a focused library of existing Web services as the initial pool for further mining.

3. Screening the growing phase. This phase filters the Web services in the focused library. In addition, the potentially interesting composition leads are identified.

4. Verification or the weeding phase. The composition leads from the previous phase are verified using the invocation plans and other characteristics such as run time conditions.

5. Evaluation or the harvest phase. The interestingness of initial invocation plans are evaluated. In addition, modifications to the plans can be proposed. The modified plans is verified again.

5. Conclusion

This paper reviews the relevant work in the area of Web service compositions. Two trends of web services compositions are reviewed. The first trend is the top-down composition approach. The top-down composition starts with a defined goal and search criteria. The resulted composition can be evaluated by quality of service attributes such as response time, price, security etc. The

second trend is the Web service mining. The Web service mining approach does not require a specific goal or search criteria. The objective of this type of composition is to find all unexpected and interesting compositions in certain domain. TheWeb services mining is a relatively new research area. More research work is needed to improve the usefulness and interestingness of Web services composites.

References

[1] Alrifai, M., Skoutas, D., Risse, T. (2010). Selecting Skyline Services for QoS-based Web Service Composition. *In*: Proceedings of the 19th International Conference on World Wide Web, WWW '10, p. 11– 20, New York, NY, USA, 2010. ACM.

[2] Bellin, G., Scott, P. J. (1994). On the pi-Calculus and Linear Logic. *Theoretical Computer Science*, 135 (1) 11–65, 1994.

[3] Bugliesi, M., Marin, A., Rossi, S. (2014). Model checking adaptive service compositions. Science of Computer Programming, 94. 289–306, 2014.

[4] Di Bernardo, M., Pottinger, R., Wilkinson. M. (2008). Semi-automatic web service composition for the life sciences using the Bio Moby semantic web framework. *Journal of Biomedical Informatics*, 41(5) 837–847, 2008.

[5] Digiampietri, L. A., Pérez-alcázar, J. J., Medeiros. C. B. (2007). AI Planning in Web Services Composition: a review of current approaches and a new solution. SBS, p. 983–992.

[6] Fensel, D., Bussler, C., Ding, Y., Omelayenko, B. (2002). The Web Service Modeling Framework WSMF. *Electronic Commerce Research and Applications*, 1 (2) 113–137.

[7] Fouad, H., Baghdad, A. (2012). Dynamic web service composition: use of case based reasoning and AI planning. *In:* Proceedings of the fourth international conference on web and information technologies (ICWIT), p. 22–29, 2012.

[8] Fu, X., Bultan, T., Su, J. (2004). Analysis of interacting bpel web services. *In:* Proceedings of the 13th International Conference on World Wide Web, WWW '04, p. 621–630, New York, NY, USA, 2004. ACM.

[9] Hatzi, O., Vrakas, D., Bassiliades, N., Anagnostopoulos, D., Vlahavas, I. (2013). The PORSCE II framework: using AI planning for automated Semantic Web service composition. The Knowledge Engineering Review, 28:137–156, 2013.

[10] Khadka, R.. Sapkota, B. (2010). An Evaluation of Dynamic Web Service Composition Approaches. *In:* M. van Sinderen and B. Sapkota, editors, 4th International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing - ACT4SOC 2010, p 67–79, Portugal, 2010. SciTePress.

[11] Kuzu, M., Cicekli, N. K. (2012). Dynamic planning approach to automated web service composition. Applied Intelligence, 36 (1) 1–28, 2012.

[12] Lemos, A. L., Daniel, F., Benatallah. B. (2015). Web Service Composition: A Survey of Techniques and Tools. *ACM Comput. Surv.,* 48 (3) 33:1–33-41, Dec. 2015.

[13] Lin, F. (2008). Situation Calculus. *In:*Handbook of Knowledge Representation, p. 649–669.

[14] Lämmermann, S. (2002). Runtime Service Composition via Logic-Based Program Synthesis . PhD thesis, Podunk IN, 2002.

[15] Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuinness, D., Parsia, B., Payne, T., Sabou, M., Solanki, M., Srinivasan, N., Sycara, K. (2005). Bringing semantics to web services: The owl-s approach. *In:* J. Cardoso and A. Sheth, editors, Semantic Web Services and Web Process Composition, p. 26–42, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[16] McDermott, D. (2002). Estimated-Regression Planning for Interactions with Web Services. *In:* Proceedings of the 6th International Conference on AI Planning and Scheduling, p. 43–54, 2002.

[17] Mcilraith. S. (2002). Adapting Golog for composition of semantic web Services. p.482–493.

[18] Medjahed, B., Bouguettaya, A., Elmagarmid, A. K. (2003). Composing Web Services on the Semantic Web. *The VLDB Journal*, 12 (4) 333–351, November.

[19] Mokhtar, S., Fournier, D. (2006). Context-aware service composition in pervasive computing environments. *In:* N. Guelfi and A. Savidis, editors, LNCS 3943, volume 3943, p. 129–144. Springer-Verlag, Berlin Heidelberg, 2006.

[20] Mukhopadhyay, D., Chougule, A. (2013). A Framework for Semi-automatedWeb Service Composition in Semantic Web. *In:* Cloud Ubiquitous Computing Emerging Technologies (CUBE), 2013 International Conference on, p. 161–166, Nov 2013.

[21] Ngu, A. H. H., Carlson, M. P., Sheng, Q. Z., Paik. H.-y. (2010). Semantic-based mashup of composite applications. *IEEE Trans. Serv. Comput.,* 3 (1) 2–15, January 2010.

[22] Parejo, J. A., Segura, S., Fernandez, P., Ruiz-Corts, A. (2014). QoS-aware web services composition using GRASP with Path Relinking. *Expert Systems with Applications*, 41. 4211–4223.

[23] Pautasso, C., Zimmermann, O., Leymann, F. (2004). Restful web services vs. "big"' web services: making the right architectural decision. *In:* WWW, 2008.

[24] Peer, J. (2004). A PDDL Based Tool for Automatic Web Service Composition, pages 149–163. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[25] Peltz, C. (2003). Web Services Orchestration and Choreography. *Computer*, 36 (10) 46–52, 2003.

[26] Rao, J., Küngas, (2004). Logic-based web services composition: From service description to process model. *In:* In Intl. Conference on Web Services (ICWS), p. 446–453. IEEE, 2004.

[27] Rao., Su, X. (2004). A survey of automated web service composition methods. *In:* Proceedings of First International Workshop on Semantic Web Services and Web Process Composition, p.43–54, 2004.

[28] Salem, C., Serge, H., Lynda, M., Vincent, M., Samir, Y. (2011). Multicriteria Evaluation Based Conceptual Framework for Composite Web Service Selection. *Evaluation and Decision Models*: Real Case Studies.

[29] Sheng, Q. Z., Qiao, X., Vasilakos, A. V., Szabo, C., Bourne, S., Xu, X. (2014). *Web services composition: A decade's overview. Information Sciences*, 280:218–238, 2014.

[30] Viroli, M. (2013). On competitive self-composition in pervasive services. Science of Computer Programming, 78:556–568, 2013.

[31] Wei., Blake, M. B. (2010). Service-Oriented Computing and Cloud Computing: Challenges and Opportunities. Internet Computing, IEEE, 14:72–75, 2010.

[32] Wu, Z., Deng, S., Li, Y., Wu. J. (2009). Computing compatibility in dynamic service composition. Knowledge and Information Systems, 19:107–129, 2009.

[33] Xi, N., Sun, C. J. Ma., Y. Shen. Secure Service Composition with Information Flow Control in Service Clouds. Future Gener. Comput. Syst., 49:142– 148, 2015.

[34] Yang, J., Papazoglou, M. (2004). Service components for managing the life-cycle of service compositions. Information Systems, 29(2):97–125, 2004.

[35] Yang, J., Papazoglou, M. P., Orriens, B., van Heuvel, W. J. (2003). A rule based approach to the service composition life-cycle. In Web Information Systems Engineering, 2003. WISE 2003. *In*: Proceedings of the Fourth International Conference on, pages 195–298, Dec 2003.

[36] Yu, T., Zhang, Y., Lin, K.-J. (2007). Efficient Algorithms for Web Services Selection with End-to-end QoS Constraints. ACM Trans. Web, 1(1), May.

[37] Zhao, X., Shen, L., Peng, X., Zhao, W. (2009). Toward SLA-constrained service composition : An approach based on a fuzzy linguistic preference model and an. Information Sciences, 2014.

[38] G. Zheng. Web Service Mining. PhD thesis, Virginia Polytechnic Institute.

[39] Zheng, G., Bouguettaya, A. (2010). Web Service Mining Framework, pages 31–75. Springer US, Boston, MA, 2010.