Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2014, 2(1): 73-79

Review Article ISSN: 2394 - 658X

A Survey on Semantic Based Automatic Web Service Compositions

Mohan H G and Devaraj F V

Department of Computer Science Engineering, PES Institute of Technology and Management, Shivamogga, Karnataka, India
mohan.hg0408@gmail.com

ABSTRACT

Today Web is no more a collection of pages but a collection of services which can be collaborated over the Internet. Composition is a process of combining two or more Web services together in order to fulfill the request. Web Services are created and updated dynamically at run time hence it is not possible to derive service compositions manually. Semantics is used to automate and wire services. This paper presents the need and issues involved in semantic based Web service composition. It also provides an overview of contemporary research initiatives towards the same. The generic framework for semantic service composition has been provided in this paper and the languages which support Services compositions are compared with respect to several parameters.

Key words: Web Services, composition, semantics, ontology, domain knowledge

---

INTRODUCTION

A Web service is an emerging software application that can be identified by a URI, whose interface and bindings are capable of being identified, described and discovered by exchanging XML based messages and supports direct communication with other services via Internet-based protocols [1]. Web service is a part of Service Oriented Computing which enables improved coordination amongst multiple computing platforms, applications, and business partners. Web services are independently developed applications that are exposed as services and interconnected using Web network infrastructure with standards such as XML, UDDI, SOAP and WSDL. With these standards it promises the interoperability various applications running on different platforms. Fig. 1 shows the 'find, bind, and execute' paradigm of Web services [1]. The registry act as a repository to store details of all the existing services. With the changing user needs, a single service cannot fulfil the functionality requested by the user, in such a scenario several services are combined together. The Services are created and updated on demand, making it impractical to generate the service compositions manually. Automation of Web service composition is a mechanism of creating new Web services from available Web services. Semantics is an essential part of activities, which includes defining services, selection and composition [2].

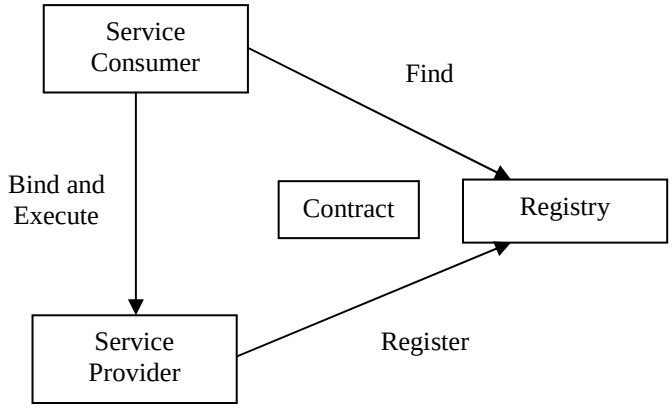

Fig.1 The Web Service Model

However the dynamic support for activities such as service discovery, selection and composition is still challenging. The main requirement of a composition scheme is the ability to describe service capabilities such as Inputs, Outputs, Pre-conditions and Effects (IOPEs). Thus, to efficiently select and integrate inter-organizational and heterogeneous services on the Web at runtime is an important step towards the development of the Web service applications. This unique requirement of automating service compositions has attracted number of research efforts.

Automation of Web service composition is a mechanism of creating new add-on Web services from available services automatically. This needs semantic support in Web service description and reasoning. The semantics is an essential part in the automation of service composition. Semantic Web services provide semantic descriptions of functionalities and processes in achieving automation of processes such as discovery, selection and composition. The semantic Web service is a Web service where internal and external descriptions are in a language which has well-defined semantics. It enables rich machine understandable descriptions of their capabilities in order to ease automation of activities such as service selection and composition. The Inputs and outputs describe members of concepts in the domain ontology [2]. The composition process is driven by the service request. The request is decomposed into several sub-requests until a level is reached in which atomic services can satisfy the sub-requests. This requires a language that supports Web Service composition and the mechanism to execute business processes. Some of the existing languages include WSFL of IBM, XLANG of Microsoft, BPEL and BPEL4WS [3]. All these languages have a limitation of supporting only static compositions and have no scope for semantic representation. The languages such as RDF, OWL and OWL-S are built on Semantic Web and these languages provide the facility of representing semantic information in the form of domain ontology [4].

In this paper we present a survey of several Web Service composition approaches. A comparison of existing schemes is established based on certain requirements of Service composition. The needs, issues and challenges of deriving compositions have been discussed. The generic semantic based automatic composition architecture is provided for better understanding of composition process. The existing composition languages have been compared against each other on certain parameters which makes the composition scheme as effective as possible.

WEB SERVICE COMPOSITION

The number of Web services is growing exponentially in recent days. The Web services act as distributed device for computation. Furthermore, by composing existing Web services into single complex service, a new and effective solution can be derived. Composition is a result of inability of a single Web service to achieve the growing needs of user demands [4].

Consider the scenario of a travel agency activity. A consumer can book a ticket through a travel reservation system and can later have the liberty of cancelling the ticket. A travel agency Web service hence must provide three operations: first of which allows the consumer to send source and destination of the travel to the service, second operation demands the service to confirm the ticket and other operation provides the facility to cancel the confirmed ticket. All these operations of travelling agency are described in a WSDL.

The travelling agency is a business process of booking a ticket, which involves executing operations in sequence. The WSDL description does not contain the sequencing of operations i.e., order of invocation. The WSDL does not describe the correlations among the operations. The correlation information is needed before linking operations within a business. Sequencing and correlation of operations creates a fundamental aspect of Web service composition called Business process. Legal contracts and quality of service information must be added to the WSDL descriptions to automate the process of composition, it is known as collaboration description.

Web service composition is an emerging approach. The various issues that have greater impact on service composition are:

Coordination
The Web service communication involves simple interactions and operation invocations. The composition of Web services requires coordination in achieving correctness and consistency. The coordination makes mutual agreement for the response of distributed transactions.

Transaction
Atomic transactions are core component in business activities; a transaction protocol is added to the Web service framework. It is defined for centralized and peer-to-peer transactions.

Context
It is any information, used in the execution of Web service. The output is adjusted according to the personalized and customized behaviour of the client. The information such as a consumer name, address, location, device type, hardware and software are part of the context.

Conversation Modelling
The conversation in Web service environment includes service discovery and binding. It used for the coordination among parties and have middle-ware properties.

Execution Monitoring
The Web services execution is centralized or distributed. The Centralised execution is a client-server model, in which the server is the central scheduler and controls the execution of the components. While in distributed execution, the Web services exchange their execution context.

Dynamic
The services can add and removed at any juncture. Thus the composition method must accommodate all the existing services at run time.

Semantic Matching
The semantic links must be constructed between the output parameters of one service and input parameters of another service. The semantic links are given a weight based on the degree of matching.

Semantic Evaluation
The result of a structured composition must be evaluated based on the services selected in the composition process, providing a parameter to compare the composition results.

User Request
The composition system should consider the user request while selecting the services for composition. All possible compositions must be provided along with their semantic value.

Automation
The primary requirement is that the generation of the composition result must be automated. This minimises the user involvement and accelerates the process of producing a composite service that satisfies the user requirements.

Non-functional requirements are not directly concerned with the specific services delivered by the system. They are emergent system properties that can be used to evaluate the performance of a system. The non-functional requirements of the composition system are as follows:

Non-determinism
The number of composition schemes for a request is not known until run time. All possible compositions must be provided as output to the user.

Scalability
The number of available Web services increases as new services are added by the service providers. The composition approach must work as effectively with larger set of services.

Correctness
The Web services can be added and removed at run time thus, the result of composition process must be correct with respect to the user request and all available Web services at composition time must be considered.

Performance
The performance of the semantic based automatic composition system is measured by the time taken to find all available structured compositions for the given request.

Availability
The availability requirement specifies that the system must be available to the user, whenever the user needs the system. The user can send the request at any time.

Effectiveness

The effectiveness of the composition system is measured based on the false negative and false positive values of composition result. The lower values of false positive and false negativeness indicate higher effectiveness.

SEMANTIC WEB SERVICES COMPOSITION FRAMEWORK

The architecture of the generic semantic composition system is presented in Fig. 2. The service providers add their service description (WSDL) file to the repository that can be accessed by all entities. The WSDL file is parsed to extract the details regarding a service which includes service location, quality, functional aspects etc. The semantic information can be represented in WSDL files by semantically annotating them. These descriptions of the services must be aligned with the domain ontology. Based on the service request, the services that provide the required functionalities are selected using domain ontological relationships. Once services are selected, all possible compositions are generated. Each composition is evaluated based on parameters such as quality of service, semantic matching, execution time etc. Finally the best suited composition is executed by the execution engine and result is provided to the service requestor.

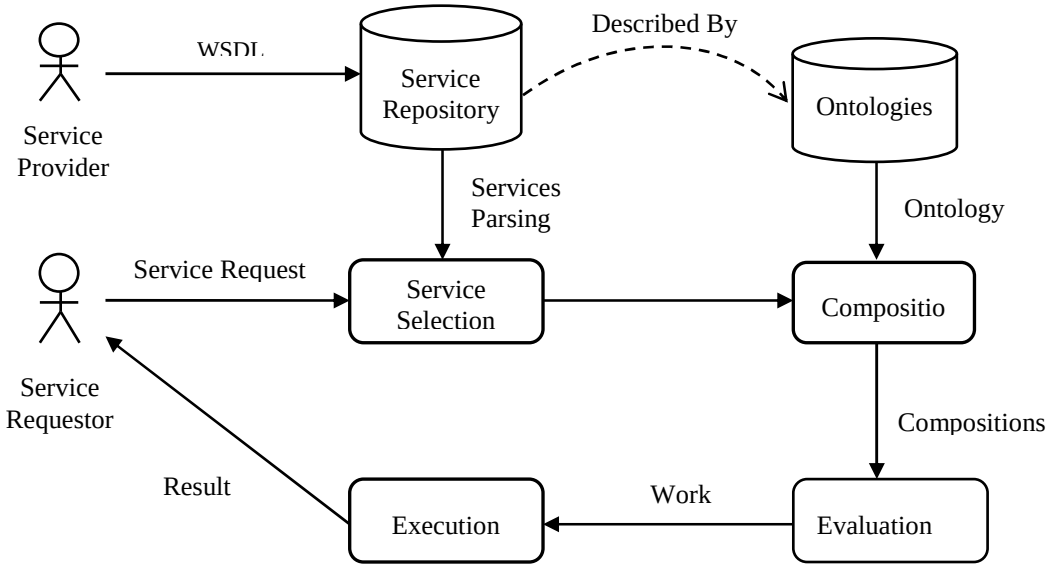

Fig.2 The Generic Semantic Web Services Composition Framework

OVERVIEW OF EXISTING WEB SERVICES COMPOSITION METHODS

A single Web service cannot satisfy the functionality requested by the consumer; hence there is a necessity of combining available services together. This need has attracted many research initiatives efforts on Web service composition. The overview of related contemporary research efforts in the field of automatically composing Web service are presented in this section.

Semantic Matching Based Composition

The semantic matching of the service descriptions are achieved in Massimo Paolucci et al [5]. The services are selected on the criteria of semantic matching between descriptions of the services being requested and the services which are advertised. The services are represented based on their functionality, specified as inputs, outputs, pre-conditions and effects. An input is the information required by the service to produce the output. The output is a confirmation that the service is executed successfully. The pre-conditions represent the conditions that need to be satisfied for the successful execution of the service. The execution of a service changes some conditions in the domain such conditions are described as effects.

The matching degree among the requested and advertised parameters is determined by their relationship in the domain ontology. There are four types of match exact, plug-in, subsume and fail. Match degree is exact, when the advertised and requested parameters are same. Plug-in degree indicates that the advertised parameter is subclass of the requested parameter. The subsume match indicates that the requested parameter is subclass of the advertised parameter. Fail match type indicates no relationship among the parameters.

Model Driven Composition

The model driven composition approach provided by Bart Orriens et al [6] has two aspects, service composition and service composition management. In service composition, a business process is created by interacting with service developer. In service composition management, the interaction between application developer and composition system is established. It is responsible for execution and the management of compositions. Four stages are identified in composition, which includes definition, scheduling, construction and execution.

The abstract composite service requirement is defined in the beginning stage. The scheduling stage finds the sequence and timing of service execution. The unambiguous compositions of services selected from the list of matching services are derived in construction stage. In the last stage, the composition is presented to an executable Web service execution language.

Ontology Driven Composition

Ruoyan Zhang et al [7] describe three composition methods by matching interfaces, human involved composition and peer-to-peer composition. The composition based on matching interface is automatic, where the domain ontology is used to establish the sequences of operations. The process is started with user input parameters and services are chained until required output parameters are reached. The goal of composition process is to establish a shortest sequence of services. The weights of the edges represent semantic similarity value. Weights are assigned based on duration, computation cost, reliability and availability.

The composition approach with human assistance involves users in selecting the required Web services, and builds a composition. It considers all inputs by semantically matching them with all Web services. The set of matched Web services is ranked based on the semantic match value. The liberty of selecting the ranked Web services vests with the user. In peer-to-peer composition, each peer entity provides services, belonging to a specific domain. Every peer must belong to some community. A community is a cluster of peers which satisfy services for a particular domain. Each community has a master peer along with a backup peer. The master peer of a community has details of the entire master and backup peers of all communities, backup peers are mirrors of master peers. A peer entity receives the service request from a user, if peer is not the master of the community; it escalates the request to its master. Then, the master finds the communities for the request and relays the request to the master that community. The masters excavate which services in their community provide the needed result to the user.

The approach in [8] presents a formal model using Causal Link Matrix (CLM). The CLM is a model for operational level composition, where the services are chained on their semantic description. The semantic connections between the Web service parameters are significant in forming a new Web service. The CLM computes causal links and stores them in the form of a matrix. The aim of the process is to discover the best suited composition depending on the semantic links.

The Web service composition process involves three main issues. Firstly, Web service discovery aims at reaching the user goal. The Web service selection is also included in service discovery step. The second aspect is to find a work flow, which describes the interactions among the Web services. The last issue refers to interaction, conversation and choreography management of Web services. The three levels based architecture is provided to counter the Web service composition challenges in [9]. It has three modules which includes the Web services discovery, functional level and process level composition.

The Web services discovery step locates matching services for the specified request. This step is termed as service matchmaking with inputs and outputs of required Web service. The problem aims to discover the set of best match for advertisement services in the registry. The composition at functional level picks a set of services on combination, are able to satisfy the objective. This process is predominantly combined with service discovery step which are aimed to search suitable service. In process level, the composed services will be executed directly to obtain the goal. Process level composition establishes a pattern for implementation.

Context Based Composition

The approach provided in Sodki Chaari et al [10] is based on the context. Context is ‘any information that can be considered while segregating the state of an entity. An entity can be anything such as person, location or any object that is part of the interaction in Web service communication. Business application and service consumer are also entities’. A community is a container, which is a network of Web services that are from a specific domain. Communities are defined by the community providers. Communities are created on the criteria of functional ontology, which captures the operations, input and output parameters.

Goal template captures the capabilities of the requested service. It is defined as a pattern that identifies the required Web services, which are considered in the composition process. The flow is described with various synchronization activities such as and-join, or-join, and-fork, or-fork and loop activity. The discovery process selects the Web service community, functional constraints and matches the goal template with context parameters. The advantage of the approach is that control and data flow can be established manually, Web service discovery and selection processes can be automated using mechanisms that capitalizes on functional and contextual parameters.

Semantic Annotations Based Composition

Semantic descriptions provide more insights on the characteristics of the Web services, which allow services to be discovered automatically at run time. Semantic Web services can be automatically composed through the use of discovery mechanisms that can identify related services using Web Service Description Language (WSDL). Guliherme C. Hobold et al [11] presents an approach for automatic discovery and composition of semantic Web services. The approach creates a graph with services as nodes and semantic matching as links. Semantic matching uses annotated information using the Semantic Annotations for WSDL (SAWSDL).

Using SAWSDL one can describe the semantics of elements defined in a WSDL. The annotations are included for the WSDL elements; each element is associated with the properties and concepts of the ontology. SAWSDL annotations can be applied to interfaces, operations, input and output parameters. The discovery process begins when the Web service of the composer is invoked. The request carries parameters such as list of available inputs, list of desired outputs, list of desired operations, maximum depth of compositions, timeout period and permission to rebuild compositions. A composition is characterised by paths in the graph that starts with the services selected. Each node is represented by a Web service and the edges are the semantic links between them.

Declarative Knowledge Based Composition

The composition method provided by Rik Eshuis et al [12], relies on declarative knowledge regarding the semantics of each service components. It constructs a service orchestration process that supports sequence, choice and parallelism. The approach has two steps. First, semantic links specifying data dependencies of the services are derived and organized in a network. Second, on a user request an executable composition is constructed from the network which satisfies the dependencies. The network can be used for different compositions.

The approach produces complex compositions from semantic links between the services. It facilitates reusing knowledge about semantic dependencies in the network to generate new compositions through new requests and modification of services at run time. The user request is decomposed to get sub-requests until the services matching the functionality are determined. The Conflict-driven Semantic Link Network (CSLN) is automatically generated for the services selected. The semantic links and causal laws defined in [13] are used to generate the CSLN. The selected clusters of services are composed by deriving an intermediate structure known as SDG. The SDG consists of process operators ‘AND’ and ‘XOR’. The ‘AND’ operator requires all incoming edges to be active before activating any outgoing edge. It activates all the outgoing edges at the same time which indicates the parallel execution of its parameters. The ‘XOR’ operator requires any one incoming edge to be active before activating the outgoing edge. It activates any one of the outgoing edges, which indicates the optional execution of its parameters. The structured composition is synthesized for a SDG by using the immediate dominator, immediate post-dominator and flow sets. The approach terminates in the linear time since SDG is acyclic and each recursive call is made for a successor of the node being processed.

A Brief Summary of various Web service composition approaches have been provided in the Table 1. The composition approaches have been compared with respect to parameters such as support for contextual parameters, resource monitoring during execution, semantic based composition using domain ontology, transaction support for business activities, QoS parameters that can be used to judge performance and effectiveness of the composition scheme and coordination among participating business processes.

Table -1 Brief Summary of Web Service Composition Approaches

Approach	Approach	Context Support	Execution Monitoring	Semantic Support	Transaction Support	QoS Monitoring	Coordination
Massimo Paolucci et al. [5]	Semantic based	no	no	yes	no	no	no
Bart Orriens et al. [6]	Model driven	no	no	no	yes	yes	yes
Ruoyan Zhang et al. [7]	Ontology driven	no	no	yes	no	yes	yes
Freddy Lecue et al. [8]	Semantic based	no	no	yes	no	no	yes
Alain Leger et al. [9]	Semantic based	no	no	yes	no	no	yes
Sodki Chaari et al. [10]	Context based	yes	yes	no	no	yes	no
Guliherme C. Hobold et al. [11]	Semantic based	no	no	yes	no	no	yes
Rik Eshuis et al. [12]	Semantic based	no	no	yes	yes	yes	yes

The language support is critical in Web service composition. A comparison of languages such as BPEL, BPEL4WS, DAML, DAML+OIL [14], WSCI, WSFL and WS-CDL are summarised in Table 2. BPEL and BPEL4WS provide transaction support for business collaboration but do not support semantic representations. They are supported by larger number of vendors. Almost all languages support minimum collaboration of services. All these languages support exceptions handling and fault correction and possess the ability to compose Web services. Only DAML and DAML+OIL support semantics representation and description of services. BPEL and BPEL4WS enjoys wide spread support from bunnies community, where are DAML and OIL are yet to be supported by the vendors.

Table -2 Comparisons of Web Service Languages

	BPML	BPEL4WS	DAML	OIL	WS-CDL	WSFL	WSCI
Semantic Support	No	No	High	High	No	No	No
Transaction support	Moderate	Moderate	Moderate	Moderate	Moderate	High	High
Exception handling	High	High	High	High	Moderate	High	Moderate
Collaboration support	Moderate	High	High	High	Low	Moderate	Moderate
Business collaboration	No	Moderate	No	No	No	Moderate	No
Software vendor support	High	High	Low	Low	Moderate	Moderate	Moderate
Work flow control	High	High	High	High	Low	Moderate	Low
Role modelling	Low	Low	No	No	High	High	High

CONCLUSION

Several Web service composition methods have been proposed to derive reusable service composition. The WSDL can be annotated to represent semantic details, QoS parameters which are highly useful in selection of services against the user request. In this paper some Web service composition schemes have been discussed and compared against the requirements of effective service composition. The process of composition needs all services description to be aligned to the single domain ontology; it must be aligned so that the compositions can be derived across the ontology definitions. The composition output can be encoded in any process execution languages to execute the real World services. The language support is still at the early stage. OWL-S provides a means the description of web services that can be represented programmatically. The languages must support the semantics of Web services at their interfaces and behaviour. The problem of the composition of services is due to lack of support from industrial vendors for the composition languages.

REFERENCES

[1] James McGovern, Sameer Tyagi, Michael Stevens, and Sunil Matthew, *Java Web Services Architecture*, Morgan Kaufmann, **2003**.
[2] Grigoris Antoniou and Frank van Harmelen, *A Semantic Web Primer*, The MIT Press, **2008**.
[3] P Traverso and M Pistore, Automated Composition of Semantic Web Services into Executable Processes, *Third International Semantic Web Conference*, Hiroshima, Japan, **2004**, pp. 380.
[4] K Sivashanmugam, John A Miller, A P Sheth, and K Verma, Framework for Semantic Web Process Composition. *International Journal of Electronic Commerce*, 9, **2005**, pp. 71.
[5] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia Sycara, Semantic Matching of Web Services Capabilities, *First International Semantic Web Conference*, Sardinia, Italy, **2002**, pp. 333.
[6] Bart Orriens, Jian Yang and Mike P Papazoglou, Model Driven Service Composition, *ICSOC Springer-Verlag*, **2003**, pp. 75.
[7] I Budak Arpinar, Ruoyan Zhang, Boanerges Aleman-meza and Angela Maduko, Ontology-Driven Web Services Composition Platform, *Information Systems and e-Business Management*, vol. 3, **2004**, pp. 6.
[8] Freddy Lecue and Alain Leger, Semantic Web Service Composition Based on a Closed World Assumption, *IEEE 4th European Conference on Web Services*, **2006**, pp. 233.
[9] Alain Leger and Freddy Lecue, Semantic Web Service Composition through a Matchmaking of Domain, *IEEE 4th European Conference on Web Services*, **2006**, pp. 171.
[10] Sodki Chaari, Khouloud Boukadi, Chokri Ben Amar, Fredrrique Biennier, and Joe Favrel, Developing Service Oriented Enterprise by Composing Web Services Based on Context, *International Journal of Computer Science and Network Security*, vol. 8, **2008**, pp. 79.
[11] Guliherme C. Hobold and Frank Siqueira, Discovery of Semantic Web Services Compositions Based on SAWSDL Annotations, *IEEE 19th International Conference on Web Services*, **2012**, pp. 280.
[12] R Eshuis, Freddy Lecue and Nikolay Mehandjiev, Flexible Construction of Complex Service Compositions from Reusable Semantic Knowledge, *IEEE 19th International Conference on Web Services*, **2012**, pp. 631.
[13] Jinghai Rao and Xiaomeng Su, A Survey of Automated Web Service Composition Methods, *First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC*, **2004**, pp. 43.
[14] Ian Horrocks, DAML+OIL: a Description Logic for the Semantic Web, *IEEE Computer Society Technical Committee on Data Engineering*, **2001**, pp. 1.