WEB SERVICE COMPOSITION PROCESSES: A COMPARATIVE STUDY

Aram AlSedrani and Ameur Touir

Department of Computer Science, King Saud University, Riyadh, KSA

ABSTRACT

Service composition is the process of constructing new services by combining several existing ones. It considered as one of the complex challenges in distributed and dynamic environment. The composition process includes, in general, the searching for existing services in a specific domain, and selecting the appropriate service, then coordinating composition flow and invoking services. Over the past years, the problem of web service composition has been studied intensively by researchers. Therefore, a significant amount of solutions and new methods to tackle this problem are presented. In this paper, our objective is to investigate algorithms and methodologies to provide a classification of existing methods in each composition phase. Moreover, we aim at conducting a comparative study to discover the main features and limitation in each phase in order to assist future research in this area.

KEYWORDS

Service composition, Composition planning, Discovery method, Selection method, Execution method.

1. INTRODUCTION

Service-oriented architecture (SOA) is a structural model composed mainly of services. Its purpose is to develop distributed systems in heterogeneous environments with location transparency, availability, discovery, and reusability of its components [1]. The fast growth in the utilization of SOA led to the growth of the number of services published. Web services offer huge potentials with a large number of simple services provided by several service providers on separate servers[2]. However, today's world has become not only more complex but also more dynamic. The single service offers simple and primitive functionality that became inadequate to satisfy the needs of future requirement. Therefore combining multiple services to provide complex composite service is of a high demand nowadays.

The composition of services process consists of several phases as shown in figure 1. The first stage is the composition planning which aims at specifying service requested and decomposing it into an abstract set of tasks. The next phase, the service discovery, a search for services that match the functionality and non-functionality requirements for each task in the composition is performed. Next, from the multiple services discovered in the previous phase, service selection comes to select the most appropriate service for each task in the composition in order to satisfy user requirement. The last phase is the service execution where the individual task in the composition is invoked and executed to come up with the final service.

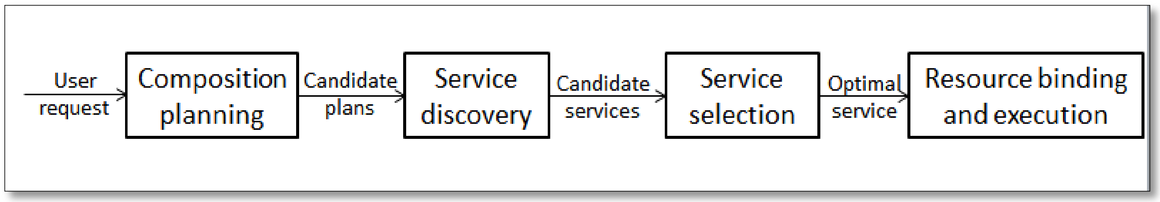

Figure 1: Service Composition Processes

In this paper, we will present a broad taxonomy of service composition methods and phases by survey and classify the existing solutions for each composition phase. This will allow us to identify the different approaches and discuss their adaptability and capability to solve such a problem. The rest of the paper is as follow: in section 2 we discuss the concept of composition in SOA, then in section 3, we overview the effort made in service composition with intelligent agent technology. Section 4 will elaborate the four phase's description as well as the comparative study of the classified methods in each phase.

2. SERVICE COMPOSITIONS IN SOA

SOA, as mentioned previously, promotes composition especially between web services including several phases such as planning, discovery, selection, and execution of services. Composing web services reinforces the reusability of service resources as well as opening new dimensions of developing web base complex applications [2].

Web service composition involves the integration of several existing web services to provide more complex and powerful service. The goal of service composition is to reuse existing web services and composing them into a complex process. However, the process of composition is considered as a high complex task due to many reasons. According to Rao [3], the complexity of service composition raises because of three main reasons. First, the huge increase in the number of web services available over the internet causes the increase of the service repositories available for searching. Moreover, web services continue changing and updating of that, require a dynamic composition at runtime, which causes intensive computations. Another reason when different providers develop web services in various models making the mapping between services in the compositions a difficult task.

Web service composition methodologies could be classified to syntactic based and semantic based composition. In syntactic composition only the syntax of web service description is considered in order to build the composition workflow and the dependency of services is manages through service inputs and outputs [4]. There are two main approaches to syntactic composition. First, the centralized approach, in this approach a mediator is responsible for combining services and mapping dependencies between services. Second, the decentralized approach, in this approach the composition is performed via the collaboration of services through peer to peer interaction.

The semantic-based composition considers the meanings and purpose of services. Machines can automatically select, compose and execute web services to achieve specified task according to the user constraints [5]. The semantic composition is achieved based on the semantic description of web services that support creating an ontology to define different aspects of web services relationships. Artificial intelligence planning algorithms are used to compose services automatically and dynamically. Bertoli in [6] proposes checking planner model, where the user set an objective used by AI planning algorithm to produce a composition plan in goal-driven architecture. The plans for this approach are constructed completely at the beginning phase, and then they are performed. However, the dynamic nature of SOA environment may cause a failure of composition created by this approach. Figure 2 depicts the different web service composition approaches.

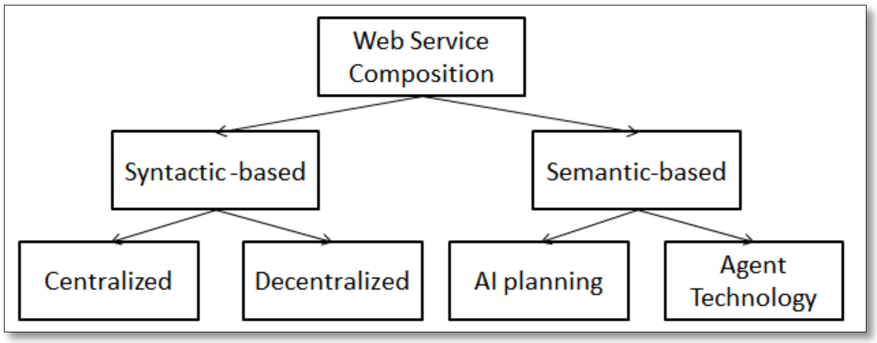

Figure 2: Web Service Composition Classification

Although service composition provides applications with complex services, this technique has some concerns such as the security and the quality of service. Indeed, by involving multiple services from interacting with multiple providers in the composition process, the security issue increases along with its management. Moreover, the performance and the quality of service remain a challenge in the automated composition.

3. Service Compositions in Agent Technology

In a Multi-Agent System (MAS), service composition is achieved by the coordination between multiple agents to discover, build, select, and execute composite service considering semantic of services parameters. A typical MAS for service composition consists of the following: (i) a planning agent that builds composition plan taking into account functional dependencies between individual service. (ii) a discovery agent that matches the tasks in the plan with existed services in a registry. (iii) A selective agent that is responsible for finding the best services to match with particular task based on some criteria. In some MAS, a coordinator agent is provided to organize the composition processes between the involved agents [7]. In other cases, the coordination between agents is carried out using peer to peer network [8].

Intelligent agents have the ability to interpret the semantic of web services and thus they are capable of providing a meaningful composite service. This process includes discovering similarity of service functionality; also, agents are capable of matching web service to the corresponding task through the understanding service semantics. Moreover, the reactivity nature of intelligent agents offers solutions in dynamic provision and composition of services. Wei [9] provides an architecture which is fully multi-agent based. His approach provides a dynamic composition and end-to-end quality assurance. Agents are divided into classes such as: (i) a User Agent to provide automatically decomposition and formalization of the tasks to generate a Task Graph, (ii) a Search Agents to find relevant services. (iii) a Registry Agent to help to find the optimal composition path. (iv)a Management Agent to implement an optimal service executable plan. (v) an Execution Agent to implement the composite service.

4. WEB SERVICE COMPOSITION PHASES

4.1. Composition Planning

The first phase of service composition is the planning which means decomposing the user request service into multiple tasks and assembling them in a plan to satisfy the user request. In traditional web service description, the planning process is conducted by matching the input of service S_{i+1} with the output of the service S_i to consider $S_i \rightarrow S_{i+1}$ part of the composition plan. However in real case scenarios, this process is not sufficient due to the differences in variable names between one provider and another. Therefore, the semantic of web services is an essential factor in determining what function the web service is performing regardless of the syntax values it represents. In semantic web service, the process of planning depends on the service profile. In particular, it is performed by connecting the precondition of S_{i+1} with the effect (postcondition) of S_i to consider $S_i \rightarrow S_{i+1}$ part of the composition plan [10].

According to Service composition literature, composition planning could be performed with different methods as illustrated in figure 3. Static planning is the approach where application developer is responsible manually of constructing the flow of service plan as well as specifying the goal, I/O, the control parameters, and the functional and the non-functional constraints. This method, in fact, provides the developer with a high degree of control over the complete process. However, it creates several challenges which it is time-consuming, hard to maintain and error-prone [11]. Therefore, the need for automatic service composition planning is desirable. In another hand, automatic planning overcomes these challenges. According to [4] an automatically generated plan should satisfy the following characteristics:

- Plans should support complex structure such as conditions, loops and non-deterministic.
- Plans allow the creation of new objects at runtime as needed to accomplish the composition.
- Utilize the non-functional service attributes to differentiate between candidate plans.
- Plans should be compatible with semantic description standards, and support semantic construction mechanisms such as hierarchies. Plans should support extended goals.

There are two automatic planning methods: workflow-based methods and AI-planning methods [12]. The workflow-based composition method is one of the initial methods addressing the problem of automatic composition. The idea emerged from the similarity of planning composition to the business process in workflow systems. The workflow process presented as an acyclic directed graph with data flow and controls description. The research in [13] proposed a solution that uses the composition as workflow approach. They defined a Transactional Workflow Ontology (TWFO) to describe the workflow. The service description is stored in OWL-S registry and is searched for each task in the composition workflow.

4.1.1. Workflow-based planning

Two main methods used in composition planning as workflow: Workflow net and Business process modeling notation (BPMN). The workflow net [14] is based on high-level Petri nets. The use of Petri net as modeling of workflow allows an overall visualization of business processes. Moreover, it provides formal analysis and verification approaches. Workflow nets modeled the transaction in states as tasks in composition. It uses the AND, XOR joint, sub-workflow tokens to build a workflow net. In another hand, BPMN [15] simplify communication between business processes. It models the BP in different levels of abstraction, and it is close to BPEL composition language. BPMN models composition plan as flowchart-like diagram consists of events, AND-gateway, OR-gateway, and activities.

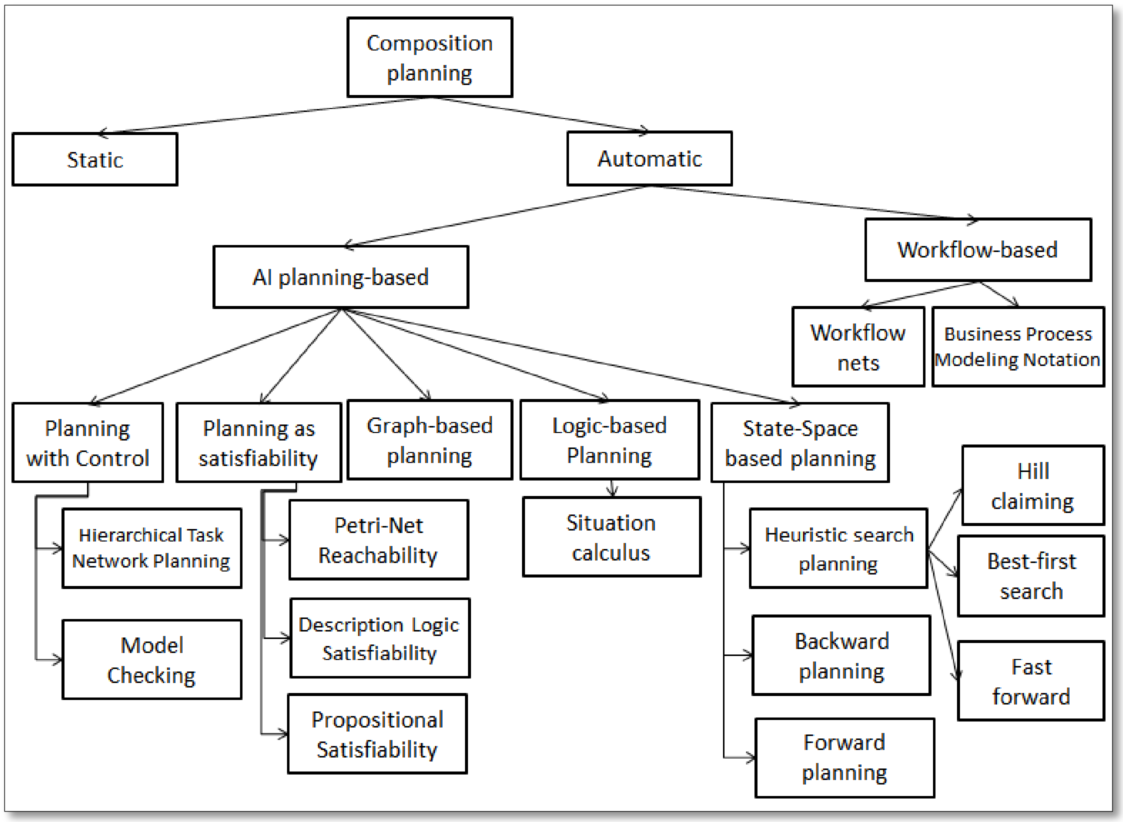

Figure 3: Planning Methods Classification

4.1.2. AI Planning-based

In the AI Planning-based method, the service composition is addressed as a planning problem. In general, an AI problem is described with (S, s0, G, A, T) [16]. Whereas S is the set of all possible states, The initial state is s0, G is the goal state, A is the set of possible actions, and T is the transaction relation from one state to the next if a specific action is done. In the context of service composition we can formulate the problem as G the composite service requested and A is the set of web services available, S, s0, T refers to state model of available services. The result of an AI

Planning composition is the construction of candidate plan that describes the arrangement of web services execution to deliver the final service.

4.1.2.1. State-space based planning

In the state space based planning, a state space consists of states, actions, transaction function, and cost function. The aim of this planning is to search for the best solution to reach the goal state. Therefore, based on initial search point we can distinguish two type of planning: forward search starting with initial state reaching a goal state, and backward search where the opposite action is done. In both kinds, the solution is to find a set of actions that connect the initial state with the goal state [17]. However, searching for the best set of actions that lead to a solution increases the possible search space which forward and backward searches suffer from. In order to reduce the search space, heuristic functions are used to estimate the usefulness of search paths to choose from. Many heuristic functions have been proposed in this manner, the heuristic search planner HSP by [18] utilizes the additive heuristic function that performs a summation of elements costs to guide the hill-climbing search from initial state to the goal. Moreover, advanced work is done by the same HSP developer uses the best-first search instead of hill-climbing which results in faster plan search.

4.1.2.2. Logic-based planning

The logic-based planning is the process of defining a set of composition rules in classical logic to express a particular domain. The composition plan is constructed based on these rules and their constraints. The researchers in [19] enhance the composition rules by adding new constraints using logic-based service composition. They used the backward planning technique to improve the reasoning of the composition. Another method employed in AI- planning is the situation calculus where the problem domain and its changes are considered as a sequence of situations; each situation is created by an action performed on a state. These situations are modeled using classical first-order language. The model in [20] use the situation calculus theory in the first-order logic language within an intelligent infrastructure to enable deducing actions simply and resolutely. The main feature of this work is the ability to deal with user preferences changes during the composition process.

4.1.2.3. Graph-based Planning

Graph-based Planning has also been exploited to facilitate web service composition methods. A Graph-based Planning constructs a directed leveled graph. The graph is composed of two types of nodes, namely action nodes and proposition nodes. The two types of nodes are located in alternating levels consisting of proposition nodes followed by layers of action nodes [21]. Each action node in level (i) connects a precondition in level (i-1) with an effect of the level (i+1). The planning terminates when two alternate levels are identical. Many researchers adopt this method such as the graph-based planning used in the framework by [22] named Spice Ace. Spice Ace deals with a composition algorithm to construct the service composition graph by representing the candidate service compositions workflows that satisfy the user functional and non-functional request. Nodes represent services and arcs represent semantic connections.

4.1.2.4. Planning as satisfiability

In planning as satisfiability, the composition problem is expressed as reasoning problem that could be solved using problem-solving algorithms. The first method is planning as propositional satisfiability where planning is based on proofing that initial conditions with domain axioms and set of actions lead to goal situation [17]. Other method uses descriptive logic instead of propositional. In this case, web services are categorized by a common set of actions. The interaction protocols between these services are modeled as execution tree. The composition is solved by identifying an execution tree corresponding to a given desired tree. The last method creates a Petri-net based on single web services that represent all possible combination of actions and identify the goal as a state of Petri-net. The satisfiability checking technique of Petri-net is used to ensure that the goal state is reachable [23].

4.1.2.5. Planning with control knowledge

The last method used in composition planning is planning with control knowledge which is a domain-specific planning. The planning in this type is performed based on specific rules determined by the problem domain. The first planning with control method is the Hierarchical Task Network (HTN) [24]. This method provides a hierarchal abstraction by decomposing the services into tasks then continually decomposing into smaller subtasks until an initial task (initial state s_0 say) is reached. The second method is planning with model checking (PMC); it is based on verification techniques. Usually, the finite state model is used with PMC method. PBM constructs a composition plan by verifying whether the goal formula is true in a specific model. The main feature in PMC is that it supports planning with uncertainty to manage nondeterminism and partial observability situations. The study in [25] investigates the concept of applying PMC to provide an automated plan for semantic web agents. They introduced four plans based on uncertainty as follow: Strong plans to ensure reaching the goal; weak plans which have a certain possibility to reach the goal; Strong cyclic plans which ensure reaching the goal when eliminating loops; Conformant plans which are used when there is no observation at the run time. They also use the non-deterministic state transition for model Semantic Web domain.

In general, the two primary methods in composition planning have been extensively addressed by researchers and each method has its properties and limitations. According to [12] we can list the differences between the two main methods as follow:

- The workflow-based method is a semi-automated with the need for specific developer implementation at some points while AI Planning could be fully automated.
- The workflow-based method requires extensive domain knowledge.

The next table lists the main advantages and disadvantages of the different planning methods discussed earlier:

Table 1: Planning Methods Comparision

Planning method		**Advantage**	**Disadvantage**
Static		High degree of control.	• Can't re-plan when service failure. • Can't meet the non-functional requirement. • Time-consuming. • Error prone.
AI planning based	Planning with control	High degree of control.	Not reusable (Domain specific)
	Planning as satisfiability	Suitable for complex goals	Need extra reasoning analysis.
	Graph-based Planning	Graphs allow easy understanding of plans which speed up the development process.	High storage space.
	Logic-based planning	• Well representing of nondeterministic domains. • Improve control with constraints.	Can't plan complex goals.
	State-space based planning	Use general search algorithms.	Low-scalability (large state space)
Workflow planning		• Simplicity • Prevent deadlocks.	• Semi-automated • Require extensive domain knowledge.

4.2. Service Discovery

Service discovery is the process of searching for services where functional and non-functional requirement meets user's needs. Web services in SOA are based on standard protocols in describing and publishing services which facilitate the discovery process. Based on previous researches on service discovery we can classify discovery methods as shown in figure 4.

4.2.1. Syntactic-based method

In syntactic based discovery asymmetric matching between services attributes and user requirement is performed. [26]search the semantic web service description for inputs and outputs parameters syntactically. The authors use inputs, outputs and their synonyms as a matching technique.

4.2.2. Context-aware method

Discovering services in this method depend on the user's context [27]. Generally, the user context is analyzed to discover the most suitable services. Specifically, a context characterizes the situation of a user, a location or the interactions between humans, applications, and the environment. A context can be further modeled as a set of pairs including context types and context values in which the type describes a characteristic of the context and the specific value is associated with it. In [28] the authors proposes a context modeling method using ontologies to enhance the analysis of user context. The service discovery is achieved based on the relations among context values described in the ontology.

4.2.3. Semantic-based method

The services in this method are described semantically by using ontologies. Ontologies provide an understanding of a specific domain and construct knowledge to specify the semantic and the constraints of domain terminologies [29]. Based on that, semantic techniques can discover the most relevant services. Moreover, by clarifying semantic similarities between ontology's concepts, related semantic Web services can be discovered. Many languages are supporting semantic service discovery and classified as follow:

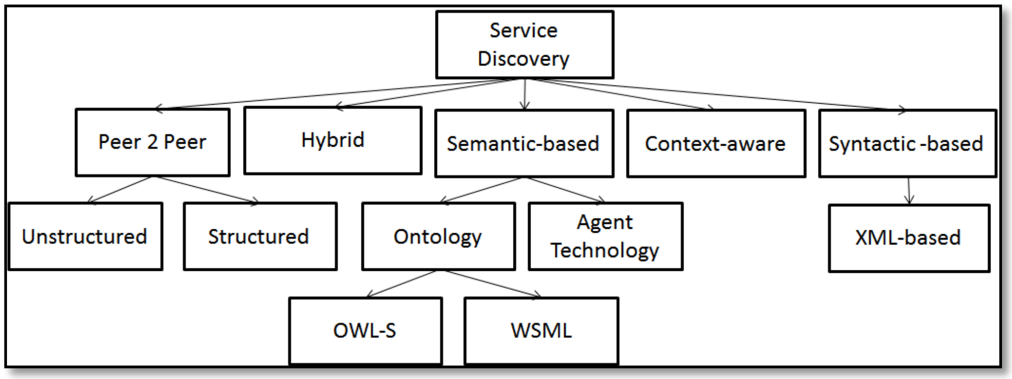

Figure 4: Discovery Methods Classification

4.2.3.1. Web Service Modeling Ontology (WSMO)

WWSMO is a model for semantic web service description [30] based on the conceptual design of Web Service Modeling Framework (WSMF). WSML define four major components to describe semantic web services: (i) Ontologies that provide the domain terminologies used by other elements. (ii) Goals that represent the user objectives and the problems that should be solved by Web services. (iii) Web services descriptions that define various attributes of a Web service. (iv) Mediators to facilitate the handling of interoperability problems between elements. The authors in [31] use WSMO as the basis for Web Service Modeling eXecution (WSMX) to provide semantic-based service discovery. WSMX is an execution environment for WSMO where discovery is performed by searching for the WSMO web service that is linked to specific goal through the mediator and return the corresponding web services.

4.2.3.2. OWL-S

OWL-S builds based on Web Ontology Language (OWL). The motivation for defining OWL-S is the use of ontologies to describe web service allowing services to be machine-interpretable. Thus, it enables the automation of Web service discovery, web service invocation, and web service composition [31].

4.2.4. Agent-based method

In the agent-based method, software agents are utilized for automated discovery of web services. The main characteristics of intelligent agents are reactive and autonomous that facilitates the discovery process. Since agents are aware of their environment, services that are added or removed from this environment will be recognized by agents and dynamically react to this changes. The use of software agents in service discovery has been the subject of several researches. The agent-based framework [32] provides an efficient mechanism to discover the required services from a collection of shared services. The framework is developed as a middleware interacting with both provider and user. The objective is to fetch service advertisements from the provider and hosted them in shared service space, and then matches the user service request with the services in the service shared space.

4.2.5. Hybrid method

The hybrid method incorporates both syntactic and semantic discovery of services. The aim of hybrid discovery is to minimize the computational expenses of reasoning in semantic method and to enhance the QoS in the syntactic method. The OWLS-MX matchmaker proposed by [33] is built for OWL-S services. It combines syntactic similarity and the degree of semantically matching. Moreover, it takes any OWL-S service as a query and returns an ordered set of relevant services based on text similarity threshold and degree of matching.

4.2.6. Peer-to-peer method

P2P method uses P2P paradigm to construct a self-organized architecture where all service providers participate in constructing P2P network. There are two ways of constructing such network based on the indexing mechanism which is Structured and unstructured P2P architecture. In the structured method, the peers in the network maintain the indexes of all resources of other peers. In unstructured P2P architecture, each peer maintains indexes of its resources. In [34] the researchers propose a routing algorithm based on flooding to passes the user request to all peers in the network and only the peer who has resources relevant to the request will respond to it.
Table 2 summarizes the advantages and disadvantages of each discovery method elaborated previously.

Table 2: Discovery methods comparison

Discovery method		Advantage	Disadvantage
Syntactic		• Simple • Uses standards (UDDI)	Can't discover similar semantic services.
Context-aware		• Improve automatic discovery with context analysis. • Adapt to the environment changes.	The analysis is complicated in the real world because the context is a general concept.
Semantic-based	Agent-based	• Facilitates dynamic rediscovering in the case of service fail. • Can deal with incomplete service description info	Need reasoning analysis.
	Ontology-based	• Facilitate automation of web service discovery. • Improve discover similar semantic services.	• Need intensive domain knowledge. • Different description languages lead to difficulty for users.
Peer-2-peer		Avoid central bottleneck.	• Routing in unstructured p2p using flooding causes network traffic.

4.3 Service Selection

The increase in the number of available web services on the internet led to the increase in the similarity of service functionality offered by different providers each with different QoS parameters. Therefore, the selection of the optimal atomic service to be combined with other services to perform complex composite service with the most satisfaction of QoS values is one of the significant requirements for service composition [35].

The input of service selection phase is the set of candidate services for various tasks involved in composition plan. A single candidate services set consists of services providing the same functionality offered by different providers through service QoS profile. Services in a set differ in non-functional properties such as QoS attribute values or user preferences. Moreover, a set may include services provided by the same provider who offers the exact service with different quality values to obtain satisfaction of most users.

Various solutions related to service selection were proposed. Figure 5 illustrates the categorization of these solutions related to optimization-based methods and decision making based methods. Decision-making methods are based on selecting services among multiple alternative services that best-fit decision maker goals and constraints. However, with a large number of candidate services, the optimization method is more efficient where it attempt to maximize or minimize one or multiple attributes taking into account user constraints.

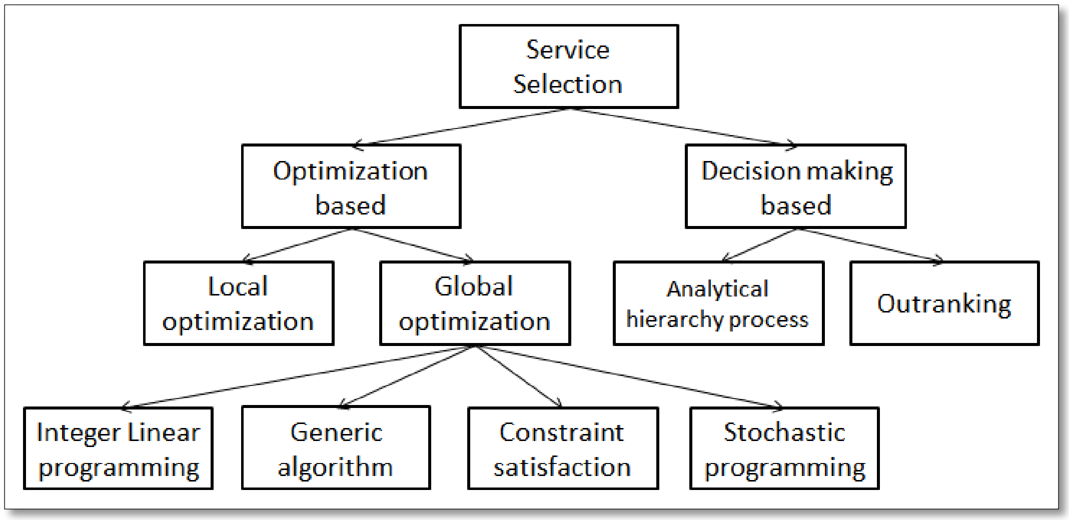

Figure 5: Selection Method Classification

4.3.1 Optimization-based method

The optimization method is performed based on two types: (i) local optimization where the best service is selected for the individual task involved in composition plan, and (ii) global selection of services to guarantee overall composition plan as a single unit.

4.3.1.1. Local optimization method

The service selection with local optimization chooses the best service for the individual task without considering the other tasks in composition or QoS requirement for the overall composed service. In this method, the candidate services for each task are ranked according to specific QoS attributes using utility theory. The utility theory mapped each quality attribute to values in utility functions. Thus, the service with higher utility function outcomes will be selected.

4.3.1.2. Global optimization methods

Although local optimization selects the optimal service for individual task, it may not lead to optimal global quality of the composed service. Moreover, users usually set constraints on the requested composed service, and it is not possible to apply these constraints to local optimization. Therefore, a global optimization that takes into account the quality of the whole composite service is more efficient selection method. Like a local method, global optimization selects services in order to maximize user satisfaction by using QoS attributes utility functions with consideration of user constraints. The service selection problem in global optimization can be modeled as Integer Linear programming, genetic algorithm, constraint satisfaction, or stochastic programming.

4.3.1.2.1. Integer Linear Programming (ILP)

In this method, the service selection problem is modeled as linear programming problem as in[36] where the input of the problem are variables assigned to each candidate web service, a linear objective function and a linear set of constraints. The ILP method aims at maximizing or minimizing the value of the objective function by changing the values of variables in the constraints boundaries.

4.3.1.2.2. Genetic Algorithms method

Another way of solving service selection as an optimization problem is the use of genetic algorithms. Genetic Algorithms (GAs) are heuristic search method that iteratively finds near-optimal solutions in large search spaces. Solving service selection in such algorithms as proposed in [37] starts with modeling the chromosome which is a single candidate solution from a set of solutions called population. Then the state of chromosomes will be evaluated based on the specific fitness function. According to that, highly evaluated chromosomes will be regenerated to produce better solutions. The reproduction process will be terminated based on some conditions such as a number of iteration. A typical GA however, cannot handle constraints directly. One of the used techniques in service selection problem is the penalty-based methods [38] where penalty function reduces the fitness of a solution based on the number of constraints violation which affects the evaluation results of such solution.

4.3.1.2.3. Constraint satisfaction method

Constraint satisfaction is another optimization method where a problem is modeled as a set of variables with their relations as constraints. Each variable has a domain consisting of all possible values that the variable can take and a set of constraints, assigning a value to a variable. One of the solutions of such a problem [35] is to assign to each variable value from its domain that satisfies all constraints. The work in [39] propose a composition process modeled as a Constraint Optimization Problem (COP). The authors argue that the proposed process be flexible enough by applying soft constraints instead of hard constraints. This means that the algorithm searches the best solution within a range of constraints instead of searching an exact value. Moreover, this process does not find the optimal solution for service selection problem.

4.3.1.2.4. Stochastic programming method

This method deals with problems under uncertainty. In particular, it uses random variables to model the uncertain parameters with known probability distributions. In [40] the authors use what it is called the Average-Value-at-Risk AVaR measures to quantify the uncertainty of time and cost quality attributes. In the optimization process, they minimize the AVaR of random variables for time and cost and then create the objective functions associated with those parameters.

4.3.2. Decision-based methods

Beside optimization methods, decision-based methods are used in service selection problems. Decision making can be defined as the operation of choosing between several entities based on input values and decision maker goals. In service selection context, decision making is the process of choosing among several candidate services that best-fit user goals and constraints. When the selection is based on one constraint, then the decision process is about choosing services that best fit this constraint. However, when there are many constraints, the process first prioritizes these constraints according to user preference, then ranks the services using some decision-making algorithms and chooses the high ranked service. In general, decision-making methods are efficient in solving problems where multiple constraints and a small number of alternatives exist. There are two decision-based methods used in solving service selection problem: analytical hierarchy process (AHP) and outranking algorithm.

4.3.2.1. Analytical Hierarchy Process (AHP)

AHP is one of the most used methods in Multi-Attribute Utility Theory (MAUT). The idea of MAUT is to maximize a specific utility function that is calculated based on the priority of the attributes of candidate services. As a consequence AHP decomposes a complicated problem into subproblems, which are organized into a hierarchical structure based on the relationships between those subproblems [41]. AHP in service composition context aims at pair-wise comparing attributes between candidate services to identify their weights in the utility function, then scoring candidate services based on their relevant utility function results that evaluate the quality attributes for each service. The outcome of AHP process is selecting the service with high ranked values. Figure 6 shows an example of how service selection processes in AHP.

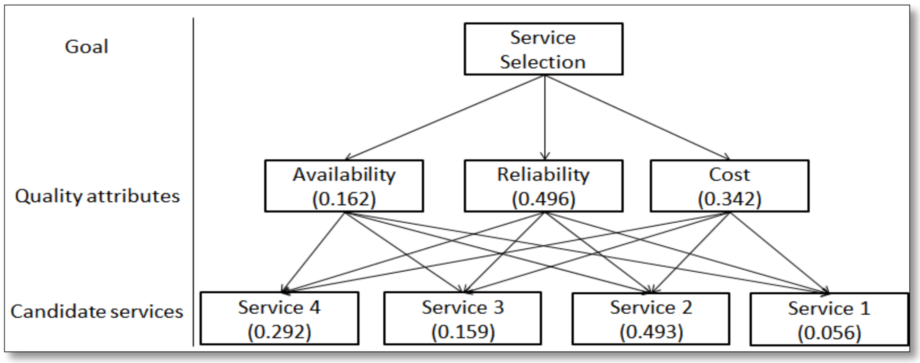

Figure 6: Analytical Hierarchy Process Example

4.3.2.2. Outranking method

The other method in decision making is outranking where all candidate services are compared and ranked according to comparison results. Therefore, a service is selected in this method if it outranks the other services in the majority of attributes and performs well in the rest. The most well-known outranking method is the ELECTRA [42] where quality attributes are modeled by using binary outranking relations whose meaning is at least as good as" instead of numerical modeling in AHP.

Table 3 lists the main advantages and disadvantages of the several service selection methods.

Table 3: Selection Method Comparison

Selection method			**Advantage**	**Disadvantage**
Optimization based	Local optimization		Low computational cost	• Can't guarantee the overall composed service quality. • Can't consider global user constraints.
Optimization based	Global optimization	Integer Linear Programming	Easy to model multiple QoS criteria constraints.	• Insufficient for complex QoS attribute because it enforces linear constraints and objective function. • Low-scalability, increase in search space by assigning variables to every candidate service.
Optimization based	Global optimization	genetic algorithm	Suitable for large-scale optimization	• Provide near-optimal solution. • Unconstrained method, need an additional technique to incorporate user constraints.
Optimization based	Global optimization	constraint satisfaction	• Clear description of constraints. • Can handle complex constraints. • The generic search algorithm can be used.	Low-scalability with an increase of candidate services in the domain, search space, will increase.
Optimization based	Global optimization	stochastic programming	Suitable for solving the problem with uncertainty.	Based on random values that can't be accurate.
Decision making based			Easy to model multiple QoS criteria constraints.	Applicable to a small number of candidate services.

4.4. Composition Execution

The last phase of service composition is to execute and monitor the recently created composite service. The process involves binding with participant services providers, invoking services, passing data between services and verifying the existence and quality of services of the composition plan [11]. By reviewing the composition literature, we can classify execution of composition as static and dynamic (figure 7).

4.4.1. Static execution

The static execution of a composition depends on a predefined plan that specifies exactly the flow of services and the providers of those services. The problem of static service composition is handled by two different techniques [43]. The first is the Orchestration where the web services are invoked and executed from a predefined plan by a centralized controller. The second technique is the Choreography where the service is composed in a peer-to-peer manner, each Web service participating in a choreography knows exactly when to execute its functions and with what service to interact [44]. In general, the static execution has a great degree of control by assigning services in composition and specifying particular providers.

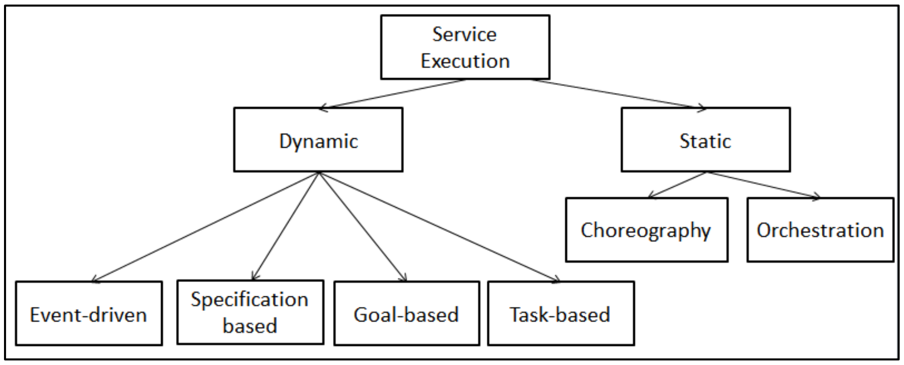

Figure 7: Execution Methods Classification

4.4.2. Dynamic execution

Web services operate autonomously within a dynamic environment. Therefore, several contingencies that affect the composition execution may occur. Web service could fail or be unavailable during execution time if it could not bind with the provider; in this case, a failure recovery strategy is responsible for substituting the failed service. However, the new service should have similar functionality with respect to the original one; when replaced in the composition, it should not change the composition context. When substitute service could not be found, re-planning of the entire composition must be performed.

Moreover, Web service quality values are frequently updated, due to service changes or changes in their environment such as network traffic. In particular, during the execution of a composite service, some participating services may update their quality attributes on-the-fly, others may become unavailable, and new services may emerge. Therefore, static methods of service composition are inefficient. Instead, dynamic composition methods are required; that takes into account the changes either in the services or the environment. In general, some of the changes in the web services that could affect the composition are as follow:

- One or more of the participant services fail to accomplish its task, or they become unavailable.
- A participant service could not provide its expected QoS or updated their values.
- New candidate services advertise for better QoS than participant service.

Dynamic service composition approaches can be classified as Task-based composition, Goal-based composition; Specification based composition and Event- driven composition.

4.4.2.1. Task-based composition

In task-based composition, the execution of services is based on abstract tasks. Users in this method must specify their tasks and constraints, and then the execution engine automatically map them into the services available [45]. One of the first attempts to use this method is the EFlow [46] which is a system for specifying and monitoring composite services. A composite service in the system is constructed as a graph to specify the order of execution of the nodes. The graph consists of three types of nodes: service node to represent the abstract task needed, decision node to specify the controlling rules through execution flow and event nodes for representing event type to ensure dynamic execution. The services for the substituting abstract task in the plan are discovered and selected semantically among service repository.

4.4.2.2. Goal based Composition

The execution in this method is based on a plan generated by an AI-planning where a user defines the desired service in term of goal and initial states [3]. Unlike the task based composition, in goal based, the plan is generated by reasoning to deliver service dynamically from the initial requirement to the desired goal. In this situation, the generated middle services between initial and goal in the plan needs reasoning about the execution order and ensuring that two consecutive services are compatible meaning that the execution engine can map the dependencies between the inputs of one service with the output of the other. An example of this method is the work by [47] which is a goal based composition model for business processes that aims at fulfilling the required BS functional and nonfunctional goals.

4.4.2.3. Specification-based composition

Specification-based composition inherits the concept of automatic generation of software programs. In this method, the composed service description is modeled as logical axioms, and the composite service specification is constructed from these axioms as a sequence needed to be proved. If the composition engine successes to find a prove, then the composite service description will be generated from that prove. Some of the research efforts within this method could be seen in [48]. The authors utilize the linear logic theorem to compose automatically semantic services. The web services in this work are presented by extra logical axioms and proofs in linear logic. The main issue in specification based method is the need for verification of the generated specification and consistency checking to ensure the validity of composition [43].

4.4.2.4. Event-driven composition

In event-driven composition, the execution of composed service is controlled by Event-Condition-Action (ECA) rules to ensure dynamic behavior. The event part of the rule will be triggered during execution when specific changes happen. The condition part will check if this change affects the composition. According to that, the action part will perform some activity to respond to this change.

Table 4: Execution Methods Comparison

Execution method		**Advantage**	**Disadvantage**
Static		• Fast execution (no demand for pre-processing) • High degree of control.	Can't response to the dynamic environment.
Dynamic methods	Task-based	Can control flow of the composition.	Require a good understanding of pre-defined tasks to sit user-request.
	Goal-based	Does not need to define a task workflow.	Computationally expensive.
	Specification-based	Enable capturing the concurrent features of Web services.	Validation overhead.
	Event-driven	Consider dynamic changes during run-time	Based on exclusive rules that may not consider all situations. (closed-word)

Many solutions are proposed to handle the dynamic environment through event driven composition. The solution proposed by [49] is a rule-based model using ECA rules to control the execution of composition workflow. [50] Utilize the ECA rules in multimedia conference systems to manage web service composition in the case of updating user requirement. The proper event will be triggered when the business process request is changed and will allow service rescheduling. However, the proposed composition is semi-automated and focuses on the changes made by the user for updating requirement. Moreover, the work does not consider the changes in QoS of participant services. Table 4, we summarize some of the main advantages and disadvantages of the various methods of web service composition execution.

5. CONCLUSIONS

Service composition offers a mechanism to expand the capability of single primitive services and exploit the services reuse. In this survey, we discussed the concept of service composition in SOA and agent technology; moreover, we examined the four phases of web service composition. For each phase, classification of the various methods used has been analyzed. Also, a comparative study of these methods is conducted by discussing the main features and limitations. The paper covered wide panoply of the taxonomy of service composition in order to serve best the user according to the request he asked for.

ACKNOWLEDGMENTS

This research has been supported by the Computer Science Dept. of the College of Computer and Information Sciences at King Saud University.

REFERENCES

[1] M. H. Valipour, B. Amirzafari, K. N. Maleki, and N. Daneshpour, "A Brief Survey of Software Architecture Concepts and Service Oriented Architecture," in IEEE International Conference on Computer Science and Information Technology, 2009, pp. 34–38.

[2] R. Yin, "Study of Composing Web Service Based on SOA," in Proceedings of the 2nd International Conference on Green Communications and Networks, 2013, vol. 2, no. Gcn 2012, pp. 209–214.

[3] J. Rao and X. Su, "A Survey of Automated Web Service Composition Methods," in Proceedings of the First international conference on Semantic Web Services and Web Process Composition, 2004, vol. 3387, pp. 43–54.

[4] V. Portchelvi, V. P. Venkatesan, and G. Shanmugasundaram, "Achieving Web Services Composition – a Survey," Softw. Eng., vol. 2, no. 5, pp. 195–202, 2012.

[5] F. Zeshan and R. Mohamad, "Semantic Web Service Composition Approaches : Overview and Limitations," Int. J. New Comput. Archit. Their Appl., vol. 1, no. 3, pp. 640–651, 2011.

[6] P. Bertoli, A. Cimatti, M. Pistore, and P. Traverso, "A Framework for Planning with Extended Goals under Partial Observability," in ICAPS-03 Proceedings, 2003, pp. 215–224.

[7] S. Kumar, "Agent-Based Semantic Web Service Composition," New York, NY: Springer, 2012, pp. 15–25.

[8] P. Kungas and M. Matskin, "Semantic Web Service Composition Through a P2P-Based Multi-agent Environment," in Proceedings of the Fourth International Workshop on Agents and Peer-to- Peer Computing, 2005, pp. 106–119.

[9] L. Wei, L. Junzhou, L. Bo, Z. Xiao, and C. Jiuxin, "Multi-agent based QoS-aware Service Composition," in IEEE International Conference on Systems, Man and Cybernetics, 2010, pp. 3125–3132.

[10] M. Klusch, "Semantic Web Service Description," in Intelligent Service Coordination in the Semantic Web, Birkhauser Basel publishing, 2008, pp. 41–68.

[11] A. KIM, M. KANG, C. MEADOWS, E. IOUP, and J. SAMPLE, "A Framework for Automatic Web Service Composition," Washington, USA, 2009.

[12] G. Baryannis, O. Danylevych, D. Karastoyanova, K. Kritikos, P. Leitner, F. Rosenberg, and B. Wetzstein, "Service Composition," in The S-Cube Book, Springer, 2010, pp. 57–87.

[13] J. Korhonen, L. Pajunen, and J. Puustjärvi, "Automatic Composition of Web Service Workflows Using a Semantic Agent," in Proceedings of the IEEE/WIC International Conference on Web Intelligence, 2003, pp. 566–569.

[14] W. Van Der-Aalst, K. Van Hee, and G. Houben, "Modelling and Analysing Workflow using a Petri-net Based Approach," in Proceedings of the second Workshop on Computer-Supported Cooperative Work, Petri nets and related for- malisms, 1994, pp. 31–50.

[15] BPMN, "Business Process Modeling Notation," Object Management Group, 2011. [Online]. Available: http://www.bpmn.org/.

[16] M. Carman, L. Serafini, and P. Traverso, "Web Service Composition as Planning," in Workshop on Planning for Web Services in ICAPS'03, 2003.

[17] J. Peer, "Web Service Composition as AI Planning – a Survey," Switzerland, 2005.

[18] B. Bonet and H. Geffner, "Planning as Heuristic Search: New Results," ECP, pp. 360–372, 1999.

[19] V. levizou and D. Plexousaki, "Enhanced Specifications for Web Service Composition," in proceedings of the European Conference on Web Services, 2006, pp. 223–232.

[20] K. Nariai, I. Paik, and M. Shinozawa, "Planning and Composition of Web services with Dynamic Constraints Using Situation Calculus," in Proceedings of The Fifth International Conference on Computer and Information Technology (CIT'05), 2005, pp. 1009–1013.

[21] S. J. Samuel, R. G. Road, and T. Nadu, "an Approach for Graph Based Planning and Quality Driven Composition of Web Services," Indian J. Comput. Sci. Eng., vol. 2, no. 5, pp. 672–679, 2011.

[22] F. Lecue, E. Silva, and L. F. Pires, "A Framework for Dynamic Web Services Composition," in Proceedings of the 2nd ECOWS Workshop on Emerging Web Services Technology, 2007, pp. 59–75.

[23] S. Narayanan and S. A. McIlraith, “Simulation, Verifcation and Automated Composition of Web Services,” in WWW ’02: Proceedings of the eleventh international conference on World Wide Web, 2002, pp. 77–88.

[24] M. Ghallab, D. Nau, and P. Traverso, “Hierarchical Task Network Planning,” in Automated Planning Theory and Practice, First., San Francisco: Elsevier Inc., 2004, pp. 229–262.

[25] B. B. Anderson, J. V. Hansen, and P. B. Lowry, “Creating automated Plans for Semantic Web Applications Through Planning as Model Checking,” Expert Syst. Appl., vol. 36, no. 7, pp. 10595–10603, 2009.

[26] A. Farooq and R. Arshad, “An Efficient Technique for Web Services Identification,” Int. J. Multidiscip. Sci. Eng., vol. 2, no. 1, pp. 26–30, 2011.

[27] S. Pakari, E. Kheirkhah, and M. Jalali, “Web Service Discovery Methods And Techniques : A Review.,” vol. 4, no. 1, pp. 1–14, 2014.

[28] H. Xiao, Y. Zou, J. Ng, and L. Nigul, “An Approach for Context-Aware Service Discovery and Recommendation,” in IEEE International Conference on Web Services, 2010, pp. 163–170.

[29] N. H. Rostami, E. Kheirkhah, and M. Jalali, “Web Services Composition Methods and Techniques: A Review,” Int. J. Comput. Sci. Eng. Inf. Technol., vol. 3, no. 6, pp. 15–29, 2013.

[30] D. Fensel, F. M. Facca, E. Simperl, and I. Toma, “Web Service Modeling Ontology,” in Semantic Web Services, Springer Berlin Heidelberg, 2011, pp. 107–129.

[31] J. Sangers, F. Frasincar, F. Hogenboom, and V. Chepegin, “Semantic Web Service Discovery Using Natural Language Processing Techniques,” Expert Syst. Appl., vol. 31, pp. 1–27, 2013.

[32] T. Amudha, N. Saritha, and P. Usha, “A Multi-agent Based Framework for Dynamic Service Discovery and Access Using Matchmaking Approach,” in International Conference on Intelligent Agent & Multi-Agent Systems, 2009, pp. 1–4.

[33] M. Klusch, B. Fries, and K. Sycara, “OWLS-MX : A Hybrid Semantic Web Service Matchmaker for OWL-S Services,” Web Semant. Sci. Serv. Agents World Wide Web, vol. 7, no. 2, pp. 121–133, 2009.

[34] J. Zhou, N. A. Abdullah, and Z. Shi, “A Hybrid P2P Approach to Service Discovery in the Cloud,” I.J. Inf. Technol. Comput. Sci., vol. 1, pp. 1–9, 2011.

[35] M. Moghaddam and J. G. Davis, “Service Selection in Web Service Composition: A Comparative Review of Existing Approaches,” in Web Services Foundations, First., A. Bouguettaya, Q. Z. Sheng, and F. Daniel, Eds. New York: Springer, 2014, pp. 321–346.

[36] L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, “QoS-Aware Middleware for Web Services Composition,” IEEE Trans. Softw. Eng., vol. 30, no. 5, pp. 311–327, May 2004.

[37] L. Ai, “QoS-Aware Web Service Composition using Genetic Algorithms,” Queensland University of Technology, 2011.

[38] A. Hilton and T. Culver, “Constraint Handling for Genetic Algorithms in Optimal Remediation Design,” J. Water Resour. Plan. Manag., vol. 126, no. 3, pp. 128–137, 2000.

[39] F. Rosenberg, P. Celikovic, A. Michlmayr, P. Leitner, and S. Dustdar, “An End-to-End Approach for QoS-Aware Service Composition,” in IEEE International Enterprise Distributed Object Computing Conference, 2009, pp. 151–160.

[40] F. Rosenberg, P. Celikovic, A. Michlmayr, P. Leitner, and S. Dustdar, “A Stochastic Programming Approach for QoS-Aware Service Composition,” in Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID), 2008, pp. 226–233.

[41] X. Xie and K. Chen, “An AHP-Based Evaluation Model for Service Composition,” in Computational Science and Its Applications - ICCSA, 2006, pp. 756–766.

[42] J. Figueira, V. Mousseau, and B. Roy, “ELECTRE Methods,” in Multiple Criteria Decision Analysis, Springer, 2005, pp. 133–162.

[43] S. Dasgupta, “A Semantic Framework for Event-Driven Service Composition" A Dissertation in Computer Science and Telecommunications and Computer Networking Presented to the Faculty of the University of Missouri Kansas City in partial fulfillment of the requirements for t,” University of Missouri Kansas City, 2011.

[44] A. Karande, M. Karande, and B B Meshram, “Choreography and Orchestration using Business Process Execution Language for SOA with Web Services,” Int. J. Comput. Sci. Issues IJCSI´11, vol. 8, no. 2, pp. 224–232, 2011.

[45] J. P. Sousa, V. Poladian, D. Garlan, and B. Schmerl, “Task-based Adaptation for Ubiquitous Computing,” IEEE Trans. Syst. Man. Cybern., vol. 36, no. 3, pp. 328–340, 2006.

[46] F. Casati, S. Ilnicki, L. Jin, V. Krishnamoorthy, and M.-C. Shan, “Adaptive and Dynamic Service Composition in eFlow,” in proceeding of 12th International Conference Advanced Information Systems Engineering (CAiSE), 2000, pp. 13–31.

[47] K. Kritikos, S. Kubicki, and E. Dubois, “Goal-based Business Service Composition,” Serv. Oriented Comput. Appl., vol. 7, no. 4, pp. 231–257, 2013.

[48] J. Rao, P. Küngas, and M. Matskin, “Composition of Semantic Web Services Using Linear Logic Theorem Proving,” Inf. Syst., vol. 31, no. 4–5, pp. 340–360, 2006.

[49] L. Chen, M. Li, and J. Cao, “A Rule-Based Workflow Approach for Service Composition,” Parallel Distrib. Process. Appl., vol. 3758, no. 03, pp. 1036–1046, 2005.

[50] Z. Ying, C. Junliang, C. Bo, and Z. Yang, “Using ECA Rules to Manage Web Service Composition for Multimedia Conference System,” in 2nd IEEE International Conference on Broadband Network & Multimedia Technology, 2009, pp. 545–549.

AUTHORS

Prof. Ameur Touir, teaching in King Saud University in Riyadh KSU, he got his B.Sc. in computer science from Saint Etienne University in 1985. And his M.Sc. degree from Pierre & Marie Curie (Paris VI) University in Artificial Intelligence 1988. Ph.D. in Computer Science 1993 from Ecole Nationale Superieure des Telecommunications de Paris. Prof. Touir research interests is in Databases, spatial access methods, spatial query processing, spatio-temporal, object-oriented, design pattern.

Aram AlSedrani is a PHD student in King Sau d University KSU in Riyadh, KSA. She got her B.Sc. degree from KSU in computer science in 1998, and got her M.Sc. degree from KSU in 2009. Aram research interest in software engineering, service oriented computing.