IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 9, 2020, accepted March 7, 2020, date of publication March 23, 2020, date of current version April 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982592

Servicing Your Requirements: An FCA and RCA-Driven Approach for Semantic Web Services Composition

MAHA DRISS[1,2], AMANI ALJEHANI[2,3], WADII BOULILA[1,2], (Senior Member, IEEE), HAMZA GHANDORH[2], AND MOHAMMED AL-SAREM[2]

[1]RIADI Laboratory, National School of Computer Sciences, University of Manouba, Manouba 2010, Tunisia
[2]College of Computer Science and Engineering (CCSE), Taibah University, Medina 42353, Saudi Arabia
[3]College of Computer and Cyber Sciences, University of Prince Mugrin, Medina 42353, Saudi Arabia

Corresponding author: Maha Driss (maha.idriss@riadi.rnu.tn)

ABSTRACT The evolution of Service-Oriented Computing (SOC) provides more efficient software development methods for building and engineering new value-added service-based applications. SOC is a computing paradigm that relies on Web services as fundamental elements. Research and technical advancements in Web services composition have been considered as an effective opportunity to develop new service-based applications satisfying complex requirements rapidly and efficiently. In this paper, we present a novel approach enhancing the composition of semantic Web services. The novelty of our approach, as compared to others reported in the literature, rests on: i) mapping user's/organization's requirements with Business Process Modeling Notation (BPMN) and semantic descriptions using ontologies, ii) considering functional requirements and also different types of non-functional requirements, such as quality of service (QoS), quality of experience (QoE), and quality of business (QoBiz), iii) using Formal Concept Analysis (FCA) technique to select the optimal set of Web services, iv) considering composability levels between sequential Web services using Relational Concept Analysis (RCA) technique to decrease the required adaptation efforts, and finally, v) validating the obtained service-based applications by performing an analytical technique, which is the monitoring. The approach experimented on an extended version of the OWLS-TC dataset, which includes more than 10830 Web services descriptions from various domains. The obtained results demonstrate that our approach allows to successfully and effectively compose Web services satisfying different types of user's functional and non-functional requirements.

INDEX TERMS Web services composition, requirements, semantic Web services, QoS, QoE, QoBiz, FCA, RCA.

I. INTRODUCTION

The widespread use of the Web and the development of network technologies is the next step in the evolutionary implementation chain of distributed applications that led to the emergence of the Web service paradigm. Web services have emerged as a new technology that, through interoperability opportunities it offers, ranks now as a focal point of multiple technological actors from various fields such as e-commerce, e-learning, e-government, or other fields. The W3C defines Web services as software components with one or more transactions ranging from simple to complex. These components are published, discovered, and invoked across the Web through the use of the Internet as communication infrastructure and XML as data format [1].

The emergence of the Web service paradigm marked a significant evolution in the history of the Internet, which was intended to act as a vector for data exchange. With Web services, the Internet became a platform for self-describing, easily integrated, and loosely-coupled software components [2], [3]. Web services have come to alleviate the problems encountered by enterprises in terms of interoperability, and this by implementing Service Oriented Architecture (SOA) [4], which is based on a set of open standards. The standardization process involves three layers of the necessary infrastructure of SOA: the communication protocol, the description specification, and finally, the publication and location specification.

The associate editor coordinating the review of this manuscript and approving it for publication was Zhangbing Zhou.

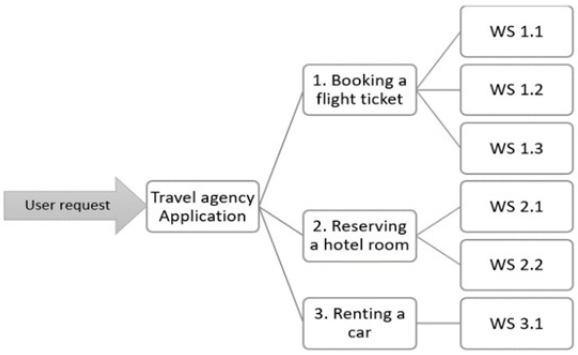


FIGURE 1. Travel agency service-based application.

One of the essential advantages of the Web service paradigm is reuse. Web services, as presented, are conceptually limited to relatively simple features that are modeled by a collection of operations. However, it is necessary to build new applications by composing services to meet more complex requirements [5], [6]. The ultimate trend of this new approach is to leverage Web service components for the applications' integration. The integration is what we call the composition or aggregation of Web services. When composing Web services composition, there are some challenging issues related to the discovery and selection of appropriate Web services that satisfy both functional and non-functional user's/organization's requirements.

Figure 1 depicts a scenario of a travel agency application that can be implemented by the Web services composition process.

A composition process is triggered by a user request that is made up of three operations: booking a flight ticket, making a hotel reservation, and renting a car. To implement this scenario, we will find many composition solutions as there are many Web services functionally performing each of the required operations. A random composition choice of the available Web services will probably return the following composition solution: C1={WS1.1, WS2.1, WS3.1}, where C represents a Web services composition, and WS represents a Web service. If we applied an appropriate selection method on the available Web services by considering not only functional requirements but also different types of non-functional parameters specified by the user, such as response time, cost, availability, service reputation, or other parameters, we would get more efficient and user-satisfying composition solution. The most appropriate composition solution would be C2={WS1.2, WS2.3, WS3.1}, in which we have low response time and costs and high service reputation comparatively to C1.

Another issue related to services composition is that the standardized languages used to describe Web services interface (i.e., WSDL [8] and OWL-S [7]) would not provide the requester with the quality information expected from the service. Indeed, its sole role is to define the syntactic (i.e., what does the service look like) and semantic information (i.e., what does it mean). However, quality of service (i.e., how well does it perform) of atomic and composite Web services still needs to be addressed.

Furthermore, composability levels between selected services is another crucial issue that should be taken into consideration, since composing two services having a low level of composability requires additional efforts for the adaptations. In the first composition C1, we might find that we need an additional adaptation that permits the output of one service to be linked as an input to another.

The Web services composition process is complicated and complex to implement; thus, we need an effective discovery and selection approach to enhance the composition of Web services by considering different types of non-functional requirements. Up to our knowledge, existing works related to this problem are mainly focusing on proposing approaches that meet only functional requirements and a single type of non-functional requirements that are related to the quality of service (QoS) properties [9].

This work seeks to address the following questions related to Web services composition issues:

1) What model would be used to identify and specify the user's/organization's requirements?
2) What technique would be used to enable the effective discovery and selection of Web services satisfying user's requirements?
3) How to consider composability levels between composed Web services to decrease the required adaptations?

To answer the previous research questions, we aim in this paper to propose a novel approach of Web services composition, which allows the discovery, selection, and composition of appropriate Web services according to the user's/organization's functional and non-functional requirements. The main contributions of the proposed approach are:

1) Modeling user's/organization's functional and non-functional requirements using an ontological description and BPMN [23];
2) Discovering appropriate atomic Web services that match user's/organization's functional requirements by applying a two-filters-based matching algorithm;
3) Considering different types of non-functional requirements related to QoS [9], QoE [10], and QoBiz [11] properties;
4) Selecting the optimal Web services and suggesting substitutes by applying Formal Concept Analysis (FCA) technique [12];
5) Focusing on composability levels between composed services to minimize as much as possible the required adaption efforts by performing Relational Concept Analysis (RCA) technique [13];
6) Monitoring the obtained service-based applications to validate the user's satisfaction.

This paper is organized as follows: Section 2 reviews the related works, Section 3 describes the proposed approach, Section 4 focuses on the experimentations, and Section 5 concludes and outlines future works.

II. RELEVANT WORKS RELATED TO REQUIREMENTS-DRIVEN WEB SERVICES COMPOSITION

Web services can be composed together to build an effective, rapid, and economic enterprise's applications that perform new complex functionalities satisfying the user's/organization's requirements more efficiently. Nowadays, different requirements-driven Web services composition approaches are proposed with diverse characteristics and orientations. In this section, we are going to review some of these approaches to understand their main capabilities and limitations.

In [14], a dynamic Web Services composition platform (StarWSCoP) is proposed to support Quality of Service (QoS) metrics. The authors suggest an extension of the WSDL to handle QoS, such as cost, time, and reliability. Also, an ontology-based semantic layer is added at the UDDI registry used for Web services discovery and selection with filtration based on the user's QoS requirements.

In [15], [16], a requirement-oriented approach was proposed starting with modeling user's requirements using a meta-process formalism (i.e., MAP) that is used to model the required services to be composed in a single graph and then specifying them using an Intentional Service Model (ISM). Next, to discover compatible services, a service engine, which is Service-Finder, was used to search for operational (i.e., functional) services. These services are filtered using FCA to select the best ones.

Reference [17] proposed an approach based on a classification technique based on RCA in order to consider various functional and non-functional requirements. The proposed approach allows users to specify their requirements by entering a set of keywords for discovering Web services and alternatives that match the user's request. Also, services were filtered and classified using multiple checkers, such as the compatibility checker, composability evaluator, and RCA classifier to generate lattices to be interpreted in order to select the optimal Web services for the required composition.

In [18], the authors define two new Web services representation algorithms in order to facilitate access to Web services at the discovery phase. The first one is called Rules-Based Text Tagging (RBTT), which is based on semantic tagging and is used to cleanse the WSDL files from unnecessary information to keep only the important ones. The second algorithm is called Symbolic Reputation (SR) and is used for Web services recommendations. SR returns information about service relationships with other services, which helps in finding alternative services. Moreover, the author suggests computing the QoS score along with reputation scores to return appropriate services to the requester. The service reputation score is feedback given by other users who have utilized the same requested service before.

Reference [19] presented a methodology called (π-SODM), which is a service composition model that supports both functional and non-functional requirements in the services-based software development process. The π-SODM is an extension of Service-Oriented Development Method (SOD-M), which relies on Model-Driven Development (MDD), which depends basically on models for structuring software applications at different points of view. The author defined new meta-models at the Platform Independent Model-Level (PIM). The proposed models aim to enhance the design and implementation processes of the services-based software by considering non-functional requirements.

In terms of Web services discovery and ranking techniques, [20] proposed a QoS-based method for discovering and ranking Web services by using Relevancy Function Value (RVF), allowing to compute a value relative to six QoS parameters that are response time, throughput time, availability, accessibility, interoperability, and cost. The main goal of this work was to build a Web services repository that could be searched by service requesters by specifying their required QoS parameters. The repository will also use RVF to re-rank available services.

Reference [21] focuses on the dynamic composition of Web services, taking into account users' requirements, especially non-functional requirements. In this work, a new QoS-based framework proposed allowing to elaborate Web services composition solutions using planning graph and Harmony Search (HS) algorithm. The proposed solutions provided functional composite services that satisfy QoS by maintaining the user's global QoS constraints. The search process for relevant Web services is done using a planning graph technique that returns all probable solutions using the HS algorithm and then ranking the results based on the user's requirements. A semantic matching score proposed and computed to handle the interoperability issues between Web services.

Table 1 presents a comparison between the works as mentioned above. The comparison is conducted upon five main criteria, which are: 1) the semantic modeling of requirements, 2) the consideration of non-functional requirements, 3) the identification of alternative Web services, 4) the consideration of Web services composability levels, and finally, 5) the consideration of user's satisfaction.

In light of the above analysis, in our work, we aim to satisfy all mentioned criteria by proposing a novel approach, which performs the development of new and value-added service-based applications based on user's/organization's functional and non-functional requirements. Functional requirements will be modeled semantically using an ontological description and graphical meta-model notation. Different types of non-functional requirements are specified in terms of QoS, QoE, and QoBiz properties in order to ensure user satisfaction. The selection of the most appropriate services and suggesting their potential substitutes (i.e., satisfying both functional and no-functional requirements and requiring less adaptation efforts) is achieved by applying formal and mathematical techniques. The validation of the user's satisfaction is ensured by monitoring the obtained service-based applications.

TABLE 1. Comparison between relevant related works.

Criteria	[21]	[20]	[19]	[18]	[17]	[15] [16]	[14]	Our Approach
Semantic modeling of requirements	√ Through using ontology	X	X	X	X	X	√ Using service discovery engine along with ontology-based semantic matching	Using an ontological description and BPMN
Consideration of non-functional requirements	Focusing only on QoS	Focusing only on QoS	Focusing only on QoS	Focusing only on QoS	Focusing only on QoS	Focusing only on QoS	Focusing only on QoS	QoS, QoE, and QoBiz
Identification of alternative Web services	√ Using planning graph technique	√ Using RVF	X	√ Calculating QoS score	√ Using service retriever and compatibility checker	√ Using FCA technique	X	Using FCA
Consideration of Web services composability levels	√ Using semantic similarity score	X	X	X	√ Using composability evaluator	X	√ Using a Wrapper	Using RCA
Consideration of user's satisfaction	X	X	X	X	X	√ Calculating precision and recall functions	X	Calculating precision and recall functions + Using monitoring technique

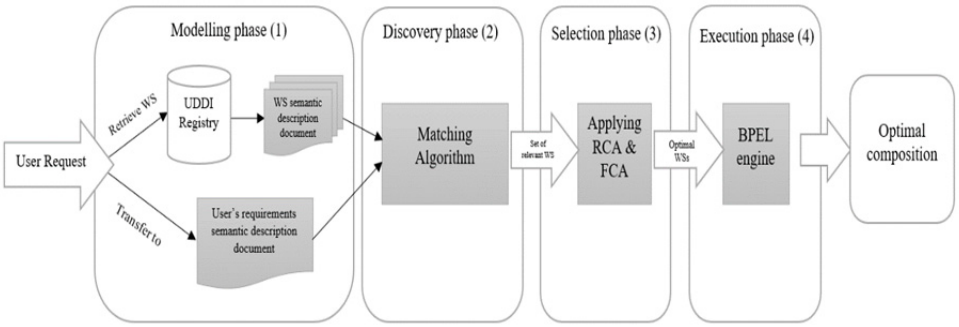


FIGURE 2. Overview of the proposed approach.

III. PROPOSED APPROACH

Our approach is a process, which is divided into four successive phases 1-4, as it is shown in Figure 2.

- Phase (1). Modeling of user's/organization's requirements: it is performed using a semantic ontological description and a graphical meta-model. Modeling user's requirements semantically permits us to address the functional and the non-functional requirements properly, and thus improve the composition process;

- Phase (2). Discovering appropriate Web services: it is ensured by applying a two-filters-based algorithm. This algorithm consists of computing the semantic similarity between the Web service semantic description files and the semantic description file of the user's/organization's requirements elaborated in phase 1;

- Phase (3). Selecting the optimal high-quality Web services: it is performed automatically by applying FCA and RCA;

- Phase (4). Executing the obtained composition: it is performed using an orchestration engine.

A. PHASE 1: MODELING USER's REQUIREMENTS

In this phase, a composition scenario is modeled using the Business Process Model and Notation (BPMN) [23]. BPMN is a well-recognized graphical meta-model used to represent a graph/map for the flow of business processes [24]. BPMN has several notational elements. An activity node represents a Web service. A `link` represents different possible flows and is chosen based on the result of the evaluation of a condition related to a specific activity. A `gateway` represents decision points that represent workflow conditions. A `sequenceflow` represents a link from a gateway node to an `activity` node. A `pool` represents the combination of a composition of `flowobject` (s), `gateway` (s), and `sequenceflow` (s). A `messageflow` describes the exchange of messages between pools. A `pool` may have an activity `flowobject` that can be represented by another `pool`. Each `pool` represents a workflow, and a business process is associated with a set of pools [35]. The BPMN specifications play a crucial role in offering a mapping that exists between the graphics of the notation as well as the fundamental constructs of execution languages. In our context, BPMN is considered an easy-to-comprehend abstract representation of the user's functional requirements. Besides, several environments (e.g., Eclipse IDE) adopt this notation and offer the possibility to convert it automatically into standardized languages (i.e., BPEL), allowing the coordination (i.e., orchestration and choreography) of composing services.

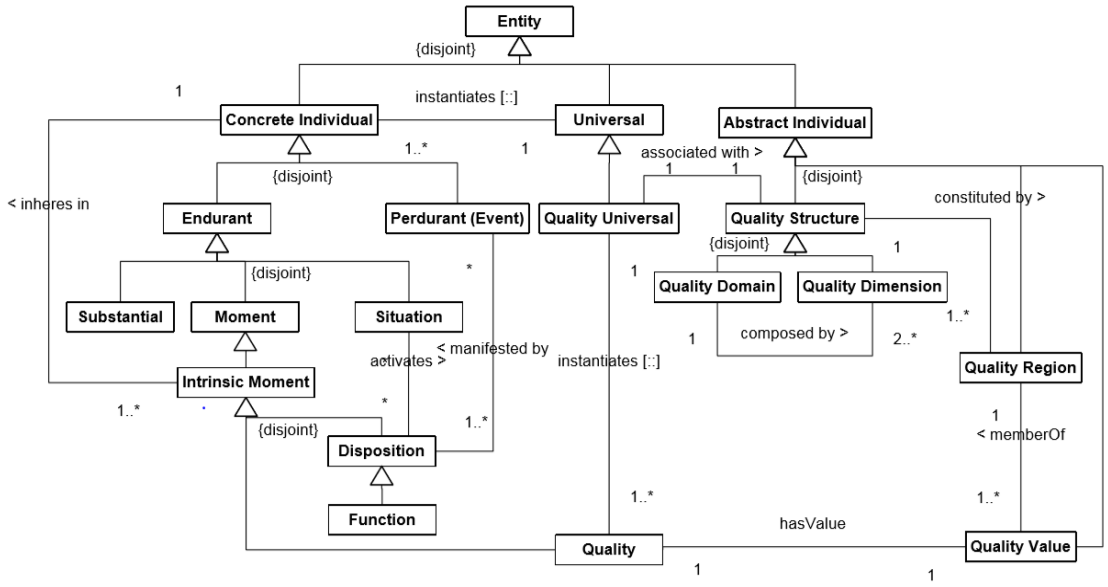


FIGURE 3. A fragment of UFO representing essential elements (including qualities and related notions).

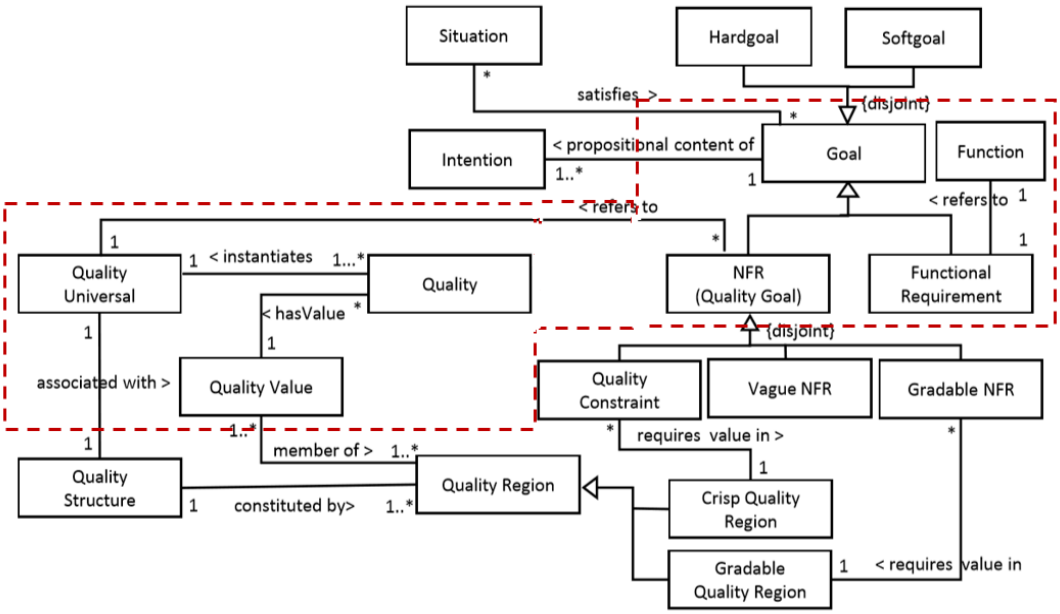


FIGURE 4. UFO fragment showing non-functional requirements and related concepts.

Nevertheless, the BPMN modeled scenario for Web services composition is restricted to specify only functional requirements. Thus, in order to provide more semantic information on both the functional and non-functional user's requirements, every requirement, which is specified in the composition scenario, is described semantically using an ontology. In [25], the authors use the Unified Foundational Ontology (UFO) in the analysis of the ontological requirements in requirements engineering. Reference [25] provides substantial information on the ontological interpretation of non-functional requirements using UFO, whereby it displays core elements such as the entity, the quality value, situation, concrete individual and abstract individual, or other elements.

The diagrammatic representation, shown in Figure 3, explains the interrelationship and effectiveness of the elements as mentioned earlier.

In [25], non-functional requirements are placed side by side with other related concepts to establish how suitable each element in developing ontological based non-functional requirements is, as exemplified in Figure 4.

What interests us, in our work, is the fragment contoured by the red dotted line. Since the UFO ontology presented in [25] has already been validated, we can take the benefits of using a fragment of this ontology in describing non-functional requirements semantically. However, instead of specifying only the ''Quality Value'' for each Quality, we modified it to include two data properties: ''Preference'' and ''Priority''. The former property expresses the required level related to a specific quality property (i.e., very high, high, medium, low, and very low), and the latter property defines the degree of importance accorded by the user to a specific required quality property (i.e., a value from 1 to 5).

Besides, concerning non-functional requirements, we propose to consider three types of properties: QoS, QoE, and QoBiz properties. QoS refers to the properties providing the service requester with quality information describing the Web service behavior such as availability, security, reliability, response time, robustness, scalability, or other behaviors [9]. In this work, we consider the following QoS properties, which are:

- The availability: it is the capability of a Web service to be executed and used [29].
- The response time: it is the amount of time taken to send a service request and receive its response [29].

QoE properties are related to the user's overall perception regarding specific Web service functionality or toward the Web service provider [10]. In this work, the reputation is considered as a QoE property. The reputation refers to the service trustworthiness, and it is measured from the ranking done by requesters. Finally, QoBiz properties concern financial aspects related to service utilization [11]. Cost is the price per transaction, which is considered as a QoBiz property in our work. It represents the amount of money needed to perform the required operation.

B. PHASE 2: DISCOVERY PHASE

In this phase, to discover Web services that match the user's functional requirements, which are semantically specified in the previous phase, we propose a two-filters-based algorithm. In the first filter, this algorithm computes the semantic similarities between keywords extracted from the requirements' descriptions and the available Web services' names. In the second filter, it computes the similarities between the same keywords and the semantic descriptions of Web services collected after passing the first filter. Web services' descriptions with highest scores of similarity (i.e., equal or greater than an empirical threshold) are retained. The proposed algorithm is based on WordNet [26]. WordNet is an accessible large lexical database for the English language, including three syntactical groups of words: one for nouns, one for verbs, and the last one is for adjectives and adverbs. WordNet consists in identifying a hierarchy of concepts, called synsets, which are sets of terms grouped in terms of different types of semantic relationships (i.e., synonymy, antonymy, hyponymy, meronymy, and holonymy) [27]. For synonymy relationship, several methods for calculating semantic similarity between words in WordNet exist. Among these methods, we cite the edge-based methods, which are widely used and which consists of measuring the path between words belonging to different nodes in the hierarchy. For these methods, the shorter the path is, the more similar the words are. WuPalmer algorithm [28] is among the most popular algorithms implementing an edge-based method for calculating semantic similarities between words. This algorithm is used in the two filters of our discovery algorithm, and it consists of defining the similarity of two concepts based on the common concepts. It calculates the path between these concepts using Equation 1:

$$sim\,(C1, C2) = \frac{2 * N3}{N1 + N2 + 2 * N3} \tag{1}$$

where *C3* is the least common super concept of *C1* and *C2*. *N1* is the number of nodes on the path from *C1* to *C3*. *N2* is the number of nodes on the path from *C2* to *C3*. *N3* is the number of nodes on the path from *C3* to root.

Our two-filters-based discovery is depicted in Algorithm 1.

Algorithm 1 Two-Filters-Based Semantic Discovery

Input: *u_keywords*: user's keywords, *serv_file_names*: list of services' file names, *serv_file:* list of services' files, *th*: threshold

Output: *Filter2List*: list of functional matched services

```
1.  Filter1List, Filter2List: list of services' file names
2.  Array1: array of double
3.  sim, avg_sim: double
/* First filter process */
4.  for all f_name in serv_file_names
5.      for all u_keys in u_keywords
6.          sim=WuPalmer (f_name, u_keys)
7.          if sim > th then
8.              Filter1List.add(f_name)
9.          end if
10.     end for
11. end for
/* The second filter process */
12.  for all f_name in Filter1List
13      cpt_names =readFile(f_name)
14.     for all u_keys in u_keywords
15.         for all cpt in cpt_names
16.             sim=WuPalmer (u_keys, cpt)
17.             Array1.add(sim)
18.         end for
19.     end for
20.     avg_sim=average(Array1)
21.     if avg_sim > th then
22.         Filter2List.add(f_name)
23.     end if
24.  end for
```

C. PHASE 3: SELECTION PHASE

The set of Web services, which are obtained from the discovery phase, are only satisfying user's functional requirements. Therefore, in this phase, we are looking for the optimal set of Web services that satisfy both functional and non-functional requirements. To automate the selection of optimal Web services, we propose to use the FCA and its extension, RCA.

FCA is a data examination method that aids in managing various categories of data using a set of data exploration and knowledge attainment procedures [12]. FCA is based on the Galois lattice theory. This theory allows to group individuals who share common properties. The groups of individuals and properties that are mutually corresponding are called formal concepts. These concepts are organized in a hierarchy, called concept lattice, according to a partial ordering, which is based on an inclusion relationship between groups of individuals and properties. A formal concept consists of two parts: the extension that contains individuals and the intension that contains the properties shared by the individuals. In FCA, data are represented in a Boolean array with two dimensions, called formal context, which defines an incidence relationship between the finite sets of individuals and properties. A formal context is formally described as a triplet $K = (O, A, I)$ where:

- O is the set of individuals;
- A is the set of properties;
- I is a binary incidence relation between elements of O and elements of A.$I \subseteq_O \times A$ indicates for each individual the properties associated with it.

A context can be represented graphically by an array. The size of this array is $|O| \times |A|$. Concepts of a context are naturally ordered by a relation of super/sub-concepts. The set of all formal concepts extracted from the context K, ordered by the relation of super-concept (or sub-concept), is denoted by $L = \langle CK, \leq K \rangle$ and is called a concept lattice.

For our service selection problem, we define a context K where individuals are the relevant services obtained after the discovery phase, and properties are QoS, QoE, and QoBiz values. The incidence relation is binary: a service is characterized by the value of QoS/QoE/QoBiz. By applying FCA on K, we want to identify relevant services that offer the best compromise of QoS, QoE, and QoBiz properties.

We choose to use FCA because it allows us to select and compose Web services dynamically. Indeed, in response to different QoS, QoE, and QoBiz properties, FCA does not provide a single service with optimal quality values, yet a collection of services will be provided. These services share common properties of QoS, QoE, and QoBiz, and thus are alternative candidates that can be interchanged dynamically to overcome problems of breakdowns or unavailability encountered during the service composition.

A service composition may be seen as a sequence of two or more atomic Web services. So, the execution of each of these services depends on the execution of the service that precedes it. To offer a more efficient approach and this by decreasing the required adaptation efforts needed for the composition, in this work, we will consider two levels of services' composability as follow:

- Fully-Composable Web services (FCWS): this level is necessary when the type and number of outputs of the source service are identical to the type and number of inputs of the target service in the composition. In this case, the two services are fully composable, and they do not need any adaptations to fulfill the required composition;
- Adaptable-Composable Web services (ACWS): this level is necessary when outputs of the source service are different from the inputs of the destination service, either in their types or number. In this case, the two services are called adaptable-composable services, and they need adaptations to fulfill the required composition.

To classify discovered Web services according to their composability levels, we propose to use RCA. RCA is an extension of the FCA, enabling the extraction of formal concepts from sets of individuals described by properties and relations between individuals. Contrary to the FCA, RCA operates on a set of contexts. This set of contexts is called a family of relational contexts (RCF). Contexts which compose an RCF are categorized into two types:

- Formal contexts: same as the contexts of the FCA, which link individuals to properties.
- Relational contexts: contexts that connect individuals of a formal context to individuals of another formal context.

The concepts, which are formed, are called relational concepts because they contain intensions referring to other concepts. Formally, an RCF is a pair (K, R) where:

- K is a set of formal contexts $Ki = (Oi, Ai, Ii)$;
- R is a set of binary relations $rk \subseteq Oi \times Oj$, where Oi and Oj are sets of individuals of formal contexts Ki and Kj that are called respectively domain and co-domain of rk.

RCF contexts are treated by classical derivation algorithms of the conceptual structures of FCA. Thus, from an RCF, we obtain a set of lattices, one per each context, called Relational Lattice Family (RLF).

The selection of composite services is carried by considering relational families per pair of services that run sequentially. Each family contains two formal contexts: services$\times$QoS/QoE/QoBiz and many relational contexts: services$\times$services, one for each level of the services' composability. The obtained RLF allows identifying concepts containing composite services that require minimum adaptations and offer maximum QoS, QoE, and QoBiz.

D. PHASE 4: EXECUTION PHASE

In the last phase, the selected services forming the optimal service-based application are coordinated and executed using an execution engine that can host and run composite services. The standardized language for Web services coordination and

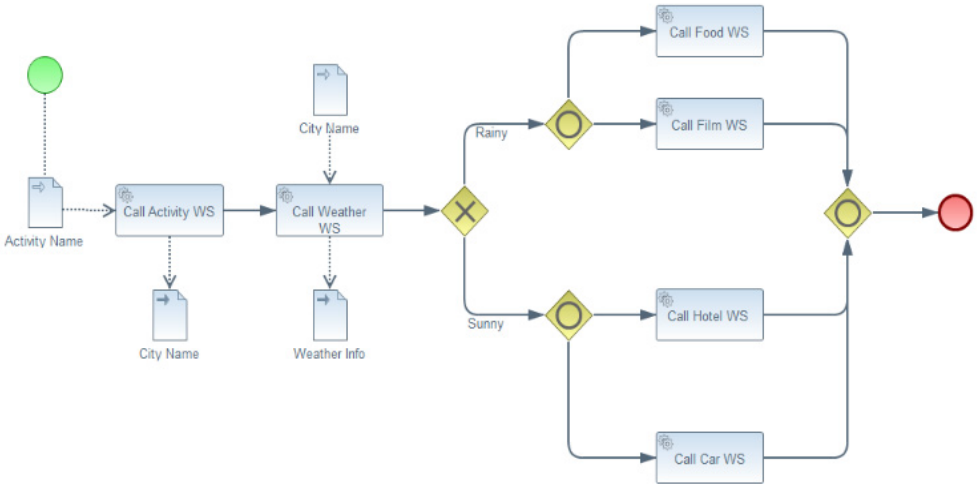


FIGURE 5. BPMN map of the proposed service-based application.

execution is the Web Services Business Process Execution Language (WS-BPEL) [22].

IV. EXPERIMENTATIONS

We used the OWLS-TC[1] dataset, which consists of more than 10830 semantic Web services described in OWL-S specification, to conduct our experimentations. Web services in this dataset are described using the two versions of OWL-S: version 1.0 and version 1.1. We chose services described using the most recent OWL-S version, which is 1.1, to perform our experimentations. The OWLS files in this dataset are also enhanced by 48 ontology domains described by OWL files. These ontologies are classified into several classes: communication, economy, travel, food, education, geography, medical, weapon, and simulation.

To take into consideration services' non-functional properties, we enriched OWL-S files with seven parameters: availability, response time, throughput, friendliness, success rate, reputation, and cost. As suggested in [30], cost values were between \$0 and \$30, response time values were between 0ms and 300ms, and the other parameter values were between 70% and 100%. Besides, we manually applied a pre-treatment on file names of this Web service by separating the attached words with an underscore.

A. EXPERIMENTAL SCENARIO

In order to understand the working mechanism of the proposed approach, experimentations were conducted on the entertainment-planning scenario illustrated in Figure 5, which depicts a BPMN map of a service-based application modeled in terms of user's functional requirements. In this scenario, first of all, the user searches for a city where he can practice a specific entertainment activity, say golf. Then, this user consults the weather information of the chosen city. If the weather is sunny during the selected dates, he/she can choose to book a hotel room or rent a car to go to the chosen city. Otherwise, if the weather will be rainy, the user cancels his trip and stays at home. In this case, he/she will choose between two other different activities; either watch a movie or order delivery from a restaurant. Every single task in this map represents the user's functional requirements that should be fulfilled by specific Web service. The BPMN map was modeled using the Eclipse BPMN2 Modeler.[2]

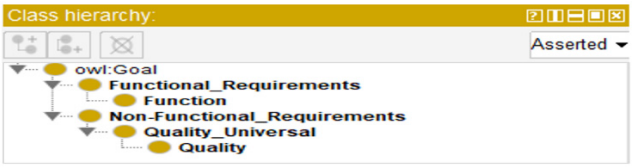


FIGURE 6. Requirements' ontology: main concepts.

The proposed requirement ontological description developed using protégé (version 5.5), as it is shown in Figure 6. A Web service is modeled as a goal to fulfill. This goal is divided into a set of functional requirements plus a set of non-functional requirements. Functional requirements are expressed in terms of required functions described by keywords used to specify activities in the BPMN map. For non-functional requirements, QoS, QoE, and QoBiz properties are specified in terms of user's preferences and priorities, as it is illustrated in Figure 7.

For the entertainment-planning scenario, Figure 8 presents the semantic description of the ''Weather_WS'' goal. This goal has as function ''Get_Weather_Forecast''. This goal has four ''quality'' instances, which are: availability, response time, cost, and reputation. For the ''Weather_WS'' goal, the user needs very high availability and reputation with a priority that is equal to '2' and '1', respectively. Besides, he/she needs services with very low response time and low

[1] http://projects.semwebcentral.org/projects/owls-tc/

[2] https://www.eclipse.org/bpmn2-modeler/

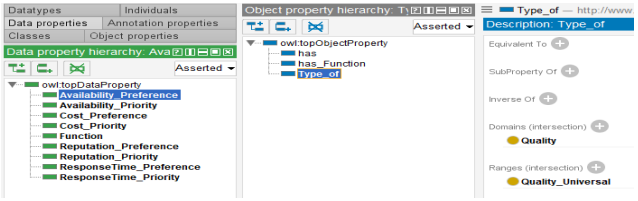


FIGURE 7. Data property and object property for the quality class.

```
<!-- http://www.semanticweb.org/justamani/ontologies/2019/2/WS#ActivityWS -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#ActivityWS">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Goal"/>
    <has rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Availability"/>
    <has rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Cost"/>
    <has rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Reputation"/>
    <has rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#ResponseTime"/>
    <has_Function rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Get_City_Provide_Given_Activity"/>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Availability -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Availability">
    <rdf:type rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Quality"/>
    <Type_of rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#QoS"/>
    <Availability_Preference rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Very High</Availability_Preference>
    <Availability_Priority rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">2</Availability_Priority>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/justamani/ontologies/2019/2/WS#ResponseTime -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#ResponseTime">
    <rdf:type rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Quality"/>
    <Type_of rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#QoS"/>
    <ResponseTime_Preference rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Very Low</ResponseTime_Preference>
    <ResponseTime_Priority rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">1</ResponseTime_Priority>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Cost -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Cost">
    <rdf:type rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Quality"/>
    <Type_of rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#QoBiz"/>
    <Cost_Preference rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Very Low</Cost_Preference>
    <Cost_Priority rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">1</Cost_Priority>
</owl:NamedIndividual>

<!-- http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Reputation -->

<owl:NamedIndividual rdf:about="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Reputation">
    <rdf:type rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#Quality"/>
    <Type_of rdf:resource="http://www.semanticweb.org/justamani/ontologies/2019/2/WS#QoE"/>
    <Reputation_Preference rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Very High</Reputation_Preference>
    <ResponseTime_Priority rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">1</ResponseTime_Priority>
</owl:NamedIndividual>
```

FIGURE 8. Semantic description of the "Activity_WS" goal.

cost. He assigned a priority that is equal to '1' for both response time and cost qualities. All the remaining goals are described in the same way.

B. EXPERIMENTAL RESULTS

To discover Web services satisfying the specified goals in the entertainment-planning scenario, the two-filters-based algorithm is performed on the OWLS-TC dataset. For the empirical similarity threshold, we considered 0.8 as in [33]. The first level filter returns the following results summarized in Table 2.

The second filter retains only the sets of services detailed in Table 3.

To validate the obtained results from the discovery phase, we manually check each returned service if it best satisfies the user's functional requirements. For this, we transpose our work in the field of information retrieval, and we use the precision and recall measures [32]. Precision assesses

TABLE 2. Results obtained after performing the 1st filter of the discovery algorithm by considering the entertainment-planning scenario.

Goal	Keywords	#matching Web services obtained after performing filter 1
Activity_WS	Activity, City	11
Weather_WS	Weather, Forecast	39
Car_WS	Car, Rent	36
Hotel_WS	Hotel, booking	26
Film_WS	Film	11
Food_WS	Food, Delivery	41

the number of true and relevant services identified among the returned set of services, recall assesses the number of returned services among the existing relevant services, though. These measures were calculated according to Equation 2 and Equation 3, respectively:

$$\text{Precision} = \frac{|\{\text{True relevant services}\} \cap \{\text{Returned services}\}|}{|\{\text{Returned services}\}|} \tag{2}$$

$$\text{Recall} = \frac{|\{\text{True relevant services}\} \cap \{\text{Returned services}\}|}{|\{\text{True relevant services}\}|} \tag{3}$$

Table 3 also provides validation of these results in terms of precision, recall, and response time of the two-filters-based algorithm. As it is shown in Table 3, the proposed two-filters-based algorithm delivers excellent results in terms of precision (97.44%), recall (98.52%), and response time (3870ms). For the same scenario, our approach in [16] provides 91.22% of precision and 87.42% of recall. The resulted sets from the discovery phase are considered as the most relevant Web services satisfying the user's functional requirements. These sets are used to perform the selection phase in order to retain only Web services that satisfy also specified QoS, QoE, and QoBiz preferences and priorities. Web services' QoS, QoE, and QoBiz (i.e., availability, response time, reputation, and cost) values are normalized. The values fall within a range that is [0,1] to execute the selection phase by min-max normalization method specified by Equation 4:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{4}$$

where x' is the new normalized value, min(x) is the minimum value in the list of the considered quality values, and max(x) is the maximum.

Normalizing quality values will allow categorizing them into seven qualitative ordinal values and this by applying the seven-point Likert technique [31]. These ordinal scales are: "Very Low", "Low", "Somewhat Low", "Medium", "Somewhat High", "High", and "Very High". The categorization of quality values is detailed in Table 4.

For our service selection problem, we define a context $\boldsymbol{K}$ for each goal, where individuals are the relevant services obtained after the discovery phase, and properties are QoS,

TABLE 3. Results obtained after performing the 2nd filter of the discovery algorithm by considering the entertainment-planning scenario.

Goal	Keywords	#matching Web services retained after performing filter 2	Precision (%)	Recall (%)	Response Time (ms)
Activity_WS	Activity, City	5	100	100	2545
Weather_WS	Weather, Forecast	3	100	100	2097
Car_WS	Car, Rent	16	93.75	100	6733
Hotel_WS	Hotel, booking	11	100	91.16	4989
Film_WS	Film	3	100	100	1988
Food_WS	Food, Delivery	11	90.9	100	4866
	Average		**97.44**	**98.52**	**3870**

TABLE 4. Likert ordinal scales and corresponding value ranges.

Ordinal scale	Very Low (VL)	Low (L)	SomeWhat Low (SWL)	Medium (M)	SomeWhat High (SWH)	High (H)	Very High (VH)
Values range	0-0.14	0.15-0.29	0.30-0.44	0.45-0.59	0.60-0.74	0.75-0.89	0.90-1

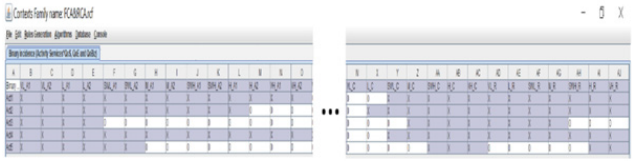


FIGURE 9. Binary context K linking activity services to QoS, QoE, and QoBiz qualitative ordinal values developed by galicia.

QoE, QoBiz qualitative ordinal values. ***K*** of the ''Activity WS'' goal of our entertainment-planning scenario is illustrated in Table 5. As specified in the semantic description of the ''Activity WS'' goal, the user assigns two as a priority for the availability of services that should fulfill this goal (see Figure 2). So, we define for each availability ordinal value two sub-values in order to express the specified priority. Besides, for the positive quality properties (i.e., availability and reputation), any object that has a relationship with one of their ordinal values will have a relationship with all ordinal values that come before. For instance, if a (SWL) availability/reputation was checked, then all the previous ordinal values (i.e., (L) and (VL)) will also be checked. However, for the negative quality properties (i.e., response time and cost), any object that has a relationship with one of their ordinal values will have a relationship with all ordinal values that come after. For instance, if a (SWL) response time/cost was checked, then all the following ordinal values (i.e., (M), (SWH), (H), and (VH)) will also be checked.

The previous context ***K*** was built using *Galicia* tool (Galois Lattice Interactive Constructor).[3] Galicia is one of the open platforms developed for lattice manipulation that allows creating, visualizing, and storing concept lattices. Figure 9 shows our context ***K*** developed by Galicia.

Figure 10 illustrates the reduced labeling of the concept lattice derived from the previous context ***K***. This lattice allows us to find optimal services for the ''Activity_WS'' goal. From this lattice, we conclude that the optimal Web service is ''Act4'' (i.e., the extent of concept/node 6). This service

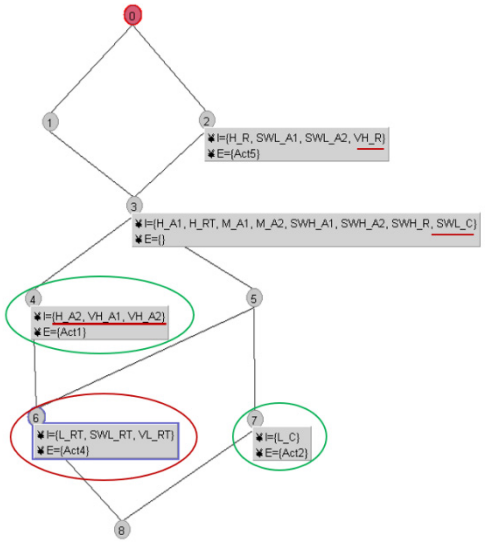


FIGURE 10. The lattice for the activity binary context.

completely satisfies the user's non-functional requirements with L_RT, H_A, SWL_C, and VH_R.

Back to our entertainment-planning scenario illustrated in Figure 5, to treat the composability problem related to sequential goals, namely ''Activity_Ws'' and ''Weather_WS'', we apply RCA. The considered relational family context is named ***RLF*** and contains two formal contexts: ActivityServices×QoS/QoE/QoBiz and WeatherServices×QoS/QoE/QoBiz, and two relational contexts: ActivityServices×WeatherServices, one for each composability level (i.e., FC and AC). Figure11 presents the two relational-contexts for both cases: FC and AC Web services.

[3]http://www.iro.umontreal.ca/~galicia/

TABLE 5. The binary context *k* linking activity services to QoS, QoE and QoBiz qualitative ordinal values.

Binary incidence (Activity Services*QoS, QoE and QoBiz)		VL_A1 (Very Low Availability 1)	VL_A2 (Very Low Availability 2)	L_A1 (Low Availability 1)	L_A2 (Low Availability 2)	SWL_A1 (Somewhat Low)	SWL_A2 (Somewhat Low)	M_A1 (Medium Availability 1)	M_A2 (Medium Availability 2)	SWH_A1 (Somewhat High)	SWH_A2 (Somewhat High)	H_A1 (High Availability 1)	H_A2 (High Availability 2)	VH_A1 (Very High Availability 1)	VH_A2 (Very High Availability 2)	VL_RT (Very Low Response)	L_RT (Low Response Time)	SWL_RT (Somewhat Low)	M_RT (Medium Response Time)	SWH_RT (Somewhat High)	H_RT (High Response Time)	VH_RT (Very High Response)	VL_C (Very Low Cost)	L_C (Low Cost)	SWL_C (Somewhat Low Cost)	M_C (Medium Cost)	SWH_C (Somewhat High Cost)	H_C (High Cost)	VH_C (Very High Cost)	VL_R (Very Low Reputation)	L_R (Low Reputation)	SWL_R (Somewhat Low)	M_R (Medium Reputation)	SWH_R (Somewhat High	H_R (High Reputation)	VH_R (Very High Reputation)
Act1	activity_beach	x	x	x	x	x	x	x	x	x	x	x	x	x	x						x	x			x	x	x	x	x	x	x	x	x	x	x	x	x
Act2	activity_farmland	x	x	x	x	x	x	x	x	x	x	x							x	x	x	x		x	x	x	x	x	x	x	x	x	x	x	x	x	x
Act3	activity_destination	x	x	x	x																	x				x	x	x	x	x	x	x	x				
Act4	activity_city	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x
Act5	activity_national_park	x	x	x	x	x	x															x					x	x	x							x	x

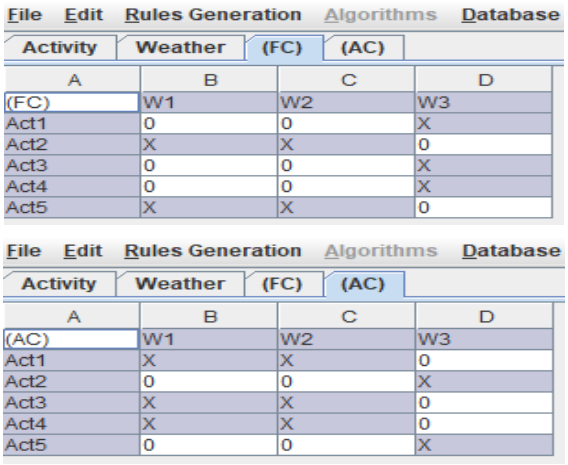


FIGURE 11. Fully-composable and adaptable-composable relational contexts.

The generated lattices, illustrated in Figure 12, denote that Act4 is the optimal Web service that satisfies ''Activity_WS'' goal. This service has an FC level with services in the extent of concept c4. This concept contains the optimal Web service satisfying the ''Weather_WS'' goal, which is W3.

Finally, for the remaining goals related to the entertainment-planning scenario, optimal Web services are selected by applying only FCA, since these services are executed in parallel; thus, no composability problems need to be handled. Table 6 summarizes the set of optimal Web services obtained after the selection phase for the entertainment-planning scenario.

This set will be used to execute the composition process implementing the entertainment-planning scenario automatically. BPEL orchestration code of this scenario is generated automatically by the BPMN2BPEL[4] plugin embedded in Eclipse IDE.

[4]https://sourceforge.net/p/bpmn2bpel/wiki/Home/

TABLE 6. Final selection results for the entertainment-planning scenario.

Goal	Selected Services
Activity_WS	Act4
Weather_WS	W3
Car_WS	C5, C6, C8, and C15
Hotel_WS	H9
Film_WS	Fi3
Food_WS	Fo2 and Fo6

To validate the proposed approach and to help the user to choose the optimal service in case the selection phase returns more than one service (e.g., returned services for ''Car_WS'' and ''Food_WS'' goals), a monitoring technique is used. Monitoring experiments are conducted using WildCAT [34], which is a generic framework allowing the development and supervision of context-aware software applications. WildCAT defines monitoring models as oriented tree structures. A monitoring model is composed of resources as nodes and attributes as leaves. Resources allow the structuring of the monitoring model. Attributes represent WildCAT sensors.

In this work, we perform monitoring experiments to measure services' availability and response time, and this by using the following metrics specified by Equation 5 and Equation 6, respectively:

$$Availability\ (\%) = \left[\frac{Uptime}{Downtime + Uptime}\right] * 100 \quad (5)$$

$$Response\ time\ (ms) = Tresp - Treq \quad (6)$$

Uptime and *Downtime* represent the service uptime and downtime, respectively. *Treq* and *Tresp* represent the time. The former represents the time when the first byte of the user's request is sent, and the latter represents the time when the last byte of the response of the service is received, respectively.

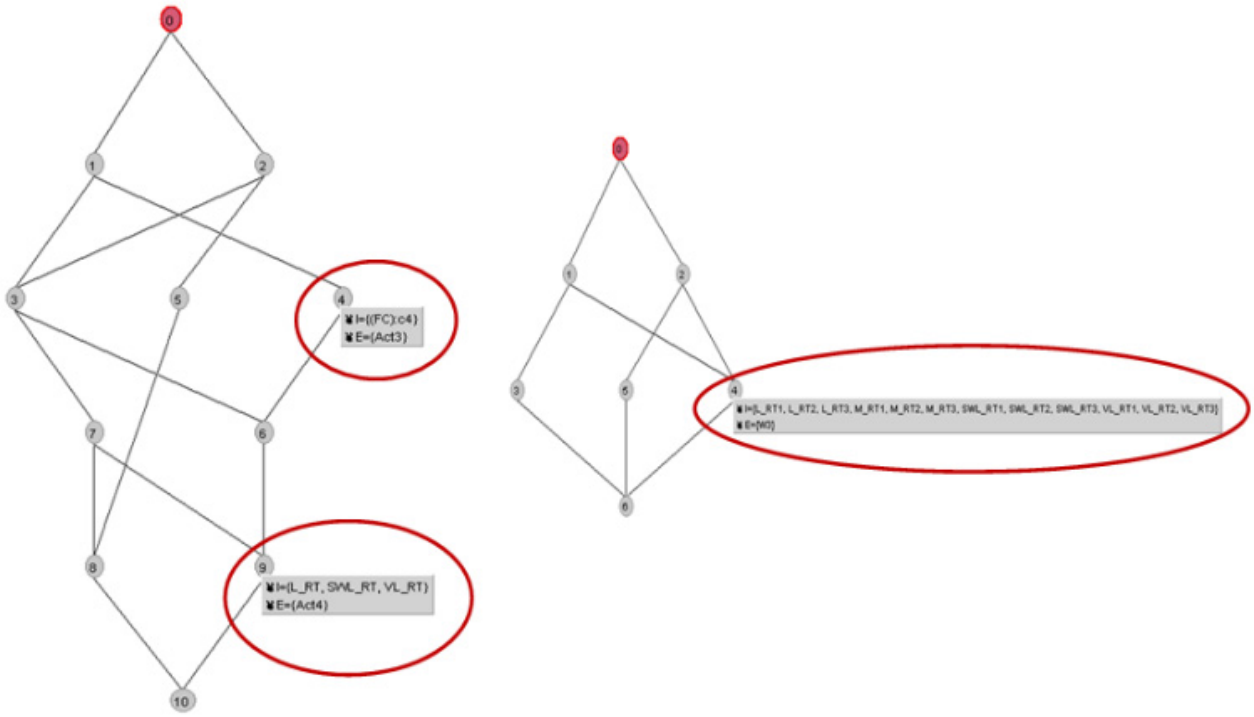


FIGURE 12. Generated lattices from RLF illustrating optimal and Fully-composable Web services for "Activity_WS" and "Weather_WS" goals.

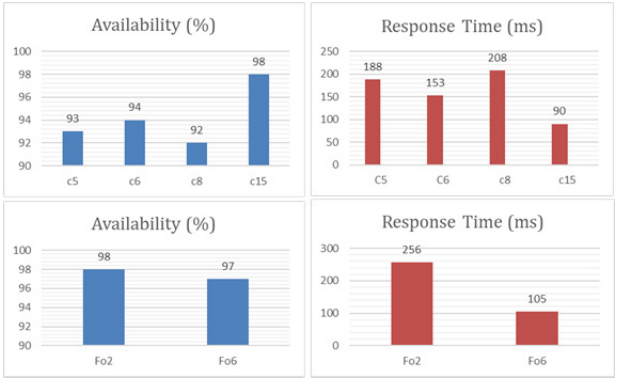


FIGURE 13. Monitoring results for services satisfying "Car_WS" and "Food_WS" goals.

To evaluate the availability and the response time of the returned services for the "Car_WS" and "Food_WS" goals, a set of experiments are conducted, and the average values are considered. For these two goals, the user gives the priority for services having the lowest response time. Figure 13 illustrates the monitoring results.

As it is shown in Figure 13, C15 is the optimal service fulfilling the "Car_WS" goal, whereas Fo6 is the optimal one for the "Food_WS" goal since it has the lowest response time value.

V. CONCLUSION

In this paper, we presented a novel requirement-driven approach that ensures the discovery, the selection, and the execution of optimal semantic Web services satisfying user's functional and non-functional requirements specified in terms of QoS, QoE, and QoBiz properties. This approach is built upon four phases: i) modeling user's/organization's functional and non-functional requirements using an ontological description and BPMN notation, ii) discovering appropriate Web services that match user's/organization's functional requirements by applying a two-filters-based algorithm, iii) selecting the optimal Web services fulfilling both specified functional and non-functional requirements and requiring minimum efforts of adaptations by applying FCA and RCA techniques, and finally, iv) executing the selected services, which form the optimal composition solution. This proposed approach experimented using an extended version of the OWLS-TC dataset, which includes more than 10830 semantic Web services descriptions. The experimental results demonstrate that our approach allows extracting the optimal composition satisfying the user's/organization's requirements with high accuracy (the average recall is 98.52%) and efficiency (the average precision is 97.44%). As future works, we plan to perform more experiments with large and complex datasets of microservices collected from different clouds. Moreover, we aim to enhance Web services compositions with automated fault diagnosis features for robust operation, and this feature could be accomplished by performing model-based and codebook techniques.

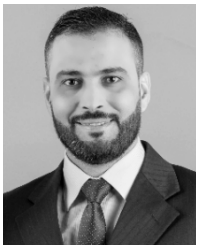
REFERENCES

[1] H. Haas and A. Brown. (2004). *W3C, Web Services Glossary*. [Online]. Available: http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/

[2] M. N. Huhns and M. P. Singh, "Service-oriented computing: Key concepts and principles," *IEEE Internet Comput.*, vol. 9, no. 1, pp. 75–81, May 2005.
[3] F. Leymann, "Web services: Distributed applications without limits-an outline," in *Proc. Database Syst. Bus., Technol., Web (BTW*, in Lecture Notes in Computer Science. Heidelberg, Germany: Springer, 2003.
[4] M. Papazoglou, *Web Services: Principles and Technology*, 2nd ed. London, U.K.: Pearson, 2012.
[5] S. Dustdar and W. Schreiner "A survey on Web services composition," *Int. J. Web Grid Services*, vol. 1, no. 1, pp. 1–30, 2005.
[6] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne, and X. Xu, "Web services composition: A decade's overview," *Inf. Sci.*, vol. 280, pp. 218–238, 2014.
[7] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara. (2004). *OWL-S: Semantic Markup for Web Services*. [Online]. Available: https://www.w3.org/Submission/OWL-S/
[8] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana. (2007). *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language*. [Online]. Available: https://www.w3.org/TR/wsdl20/
[9] M. Papazoglou, K. Pohl, M. Parkin, and A. Metzger, Eds., *Service Research Challenges and Solutions for the Future Internet: S-Cube-Towards Engineering, Managing and Adapting Service-Based Systems*, vol. 6500. Berlin, Germany: Springer, 2010.
[10] B. Upadhyaya, Y. Zou, I. Keivanloo, and J. Ng, "Quality of experience: User's perception about Web services," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 410–421, May 2015.
[11] A. Van Moorsel, "Metrics for the Internet age: Quality of experience and quality of business," in *Proc. 5th Int. Workshop Performability Modeling Comput. Commun. Syst.*, Sep. 2001, vol. 34, no. 13, pp. 26–31.
[12] B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundations*. New York, NY, USA: Springer-Verlag, 1999.
[13] M. Huchard, A. R. Hacene, C. Roume, and P. Valtchev, "Relational concept discovery in structured datasets," *Ann. Math. Artif. Intell.*, vol. 49, nos. 1–4, pp. 39–76, 2007.
[14] H. Sun, X. Wang, B. Zhou, and P. Zou, "Research and implementation of dynamic Web services composition," in *Advanced Parallel Processing Technologies*, X. Zhou, M. Xu, S. Jähnichen, and J. Cao, Eds. Berlin, Germany: Springer, 2003, pp. 457–466.
[15] M. Driss, N. Moha, Y. Jamoussi, J. M. Jézéquel, and H. H. B. Ghézala, "A requirement-centric approach to Web service modeling, discovery, and selection," in *Proc. Int. Conf. Service-Oriented Comput.* Berlin, Germany: Springer, Dec. 2010, pp. 258–272.
[16] M. Driss, Y. Jamoussi, J.-M. Jézéquel, and H. H. B. Ghézala, "A multi-perspective approach for Web service composition," in *Proc. 13th Int. Conf. Inf. Integr. Web-Based Appl. Services (iiWAS)*, 2011, pp. 106–111.
[17] Z. Azmeh, M. Driss, F. Hamoui, M. Huchard, N. Moha, and C. Tibermacine, "Selection of composable Web services driven by user requirements," in *Proc. IEEE Int. Conf. Web Services*, Jul. 2011, pp. 395–402.
[18] M. Aznag, M. Quafafou, N. Durand, and Z. Jarir, "Web services discovery and recommendation based on information extraction and symbolic reputation," *Int. J. Web Service Comput.*, vol. 4, no. 1, pp. 1–18, Mar. 2013.
[19] V. De Castro, M. A. Musicante, U. S. Da Costa, P. A. de Souza Neto, and G. Vargas-Solar, "Supporting non-functional requirements in services software development process: An MDD approach," in *Proc. Int. Conf. Current Trends Theory Pract. Inform.* Cham, Switzerland: Springer, Jan. 2014, pp. 199–210.
[20] M. Suchithra and M. Ramakrishnan, "Efficient discovery and ranking of Web services using non-functional QoS requirements for smart grid applications," *Procedia Technol.*, vol. 21, pp. 82–87, 2015.
[21] A. Bekkouche, S. M. Benslimane, M. Huchard, C. Tibermacine, F. Hadjila, and M. Merzoug, "QoS-aware optimal and automated semantic Web service composition with user's constraints," *Service Oriented Comput. Appl.*, vol. 11, no. 2, pp. 183–201, Jun. 2017.
[22] A. Alves, A. Arkin, S. Askary, C. Barreto, B. Bloch, F. Curbera, M. Ford, Y. Goland, A. Guízar, N. Kartha, C. K. Liu, R. Khalaf, D. König, M. Marin, V. Mehta, S. Thatte, D. van der Rijn, P. Yendluri, A. Yiu. (2007). *OASIS Web Services Business Process Execution Language (WSBPEL) TC*. [Online]. Available: https://www.oasis-open.org/committees/wsbpel/
[23] BPMN. (2011). *Business Process Model and Notation v2.0, OMG*. [Online]. Available: https://www.omg.org/spec/BPMN/
[24] M. Chinosi and A. Trombetta, "BPMN: An introduction to the standard," *Comput. Standards Interfaces*, vol. 34, no. 1, pp. 124–134, Jan. 2012.
[25] R. S. S. Guizzardi, F. L. Li, A. Borgida, G. Guizzardi, J. Horkoff, and J. Mylopoulos, "An ontological interpretation of non-functional requirements," in *Proc. FOIS*, vol. 14, Jan. 2014, pp. 344–357.
[26] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.
[27] C. Leacock, G. A. Miller, and M. Chodorow, "Using corpus statistics and WordNet relations for sense identification," *Comput. Linguistics*, vol. 24, no. 1, pp. 147–165, 1998.
[28] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proc. 32nd Annu. Meeting Assoc. Comput. Linguistics*, 1994, pp. 133–138.
[29] D. A. D'Mello and V. S. Ananthanarayana, "Semantic Web service selection based on service provider's business offerings," *Int. J. Simul., Syst., Sci. Technol.*, vol. 10, no. 2, pp. 25–37, 2009.
[30] Q. Yu and A. Bouguettaya, *Foundations for Efficient Web Service Selection*. New York, NY, USA: Springer, 2009.
[31] A. Joshi, S. Kale, S. Chandel, and D. Pal, "Likert scale: Explored and explained," *Brit. J. Appl. Sci. Technol.*, vol. 7, no. 4, pp. 396–403, 2015.
[32] W. B. Frakes, *Information Retrieval: Data Structures & Algorithms*. New Delhi, India: Pearson Education, 1992.
[33] M. Driss, Y. Jamoussi, N. Moria, J.-M. Jézéquel, and H. H. Ben Ghézala, "Une approche centrée exigences pour la composition de services Web," *Ingénierie Syst. Inf.*, vol. 16, no. 2, pp. 97–125, Apr. 2011.
[34] P. C. David and T. Ledoux, "WildCAT: A generic framework for context-aware applications," in *Proc. 3rd Int. Workshop Middleware Pervasive Ad-Hoc Comput.*, Nov. 2005, pp. 1–7.
[35] H. Ghandorh and H. Lutfiyya, "Automated mapping of business process execution language to diagnostics models," in *Proc. 5th Int. Conf. Cloud Comput. Services Sci. (CLOSER)*, vol. 1, 2015, pp. 251–259.

MAHA DRISS received the Engineering degree (Hons.) in computer science, the M.Sc. degree from the National School of Computer Science (ENSI), University of Manouba, Tunisia, in 2006 and 2007, respectively, and the Ph.D. degree conjointly from the University of Manouba, Tunisia, and University of Rennes 1, France, in 2011. From 2012 to 2015, she was an Assistant Professor in computer science with the National Higher Engineering School of Tunis, University of Tunis, Tunisia. She is currently an Assistant Professor of computer science with the IS Department, College of Computer Science and Engineering, Taibah University, Saudi Arabia. She is also a Senior Researcher with the RIADI Laboratory, University of Manouba. Her primary research interests include software engineering, Web service computing, distributed systems, the IoT, and artificial intelligence.

AMANI ALJEHANI received the M.Sc. degree (Hons.) in information systems from the College of Computer Science and Engineering (CCSE), Taibah University, Saudi Arabia, in 2019. She is currently a Lecturer with the University of Prince Mugrin, Saudi Arabia. Her primary research interests include Web service computing and data science.



WADII BOULILA (Senior Member, IEEE) received the Eng. degree (Hons.) in computer science from the Aviation School of Borj El Amri, in 2005, the M.Sc. degree from the National School of Computer Science (ENSI), University of Manouba, Tunisia, in 2007, and the Ph.D. degree conjointly from ENSI and Telecom-Bretagne, University of Rennes 1, France, in 2012. He is currently an Assistant Professor of computer science with the IS Department, College of Computer Science and Engineering, Taibah University, Medina, Saudi Arabia. He is also a Senior Researcher with the RIADI Laboratory, University of Manouba, and an Associate Researcher with the ITI department, University of Rennes 1. His primary research interests include big data analytics, deep learning, data mining, artificial intelligence, cybersecurity, uncertainty modeling, and remote sensing images. He has served as the Chair, a Reviewer, and a TPC Member for many leading international conferences and journals.

HAMZA GHANDORH received the Ph.D. degree from the Faculty of Engineering, University of Western Ontario, London, ON, Canada, in 2017, under the supervision of R. Eagleson. Since 2018, he has been an Assistant Professor with the Faculty of Computer Science and Engineering, Taibah University, Madina, Saudi Arabia, doing his research as a Principal Investigator with the Mixed Reality Laboratory, Taibah University. His research interests include software engineering and surgical simulation, human–computer interface design, integration of immersive experiences, and artificial intelligence.

MOHAMMED AL-SAREM received the M.S. degree in information technology from the Faculty of Informatics and Computer Engineering, Volgograd State Technical University, Volgograd, Russia, and the Ph.D. degree from the Faculty of Informatics, University of Hassan II Casablanca-Mohammedia, Mohammedia, Morocco, in 2007 and 2014, respectively. He is currently an Assistant Professor with the IS Department, Taibah University, Al Madinah Al Munawarah, Saudi Arabia. He has published several articles and participated in managing several international conferences. His current research interests include group decision making, multicriteria decision-making, data mining, E-learning, natural language processing, and social analysis.