A Multi-Perspective Approach for Web Service Composition

Maha Driss, Yassine Jamoussi, Jean-Marc Jézéquel, Henda Hajjami Ben Ghézala

A Multi-Perspective Approach for Web Service Composition

Maha Driss[1,2], Yassine Jamoussi[1], Jean-Marc Jézéquel[2], and Henda Hajjami Ben Ghézala[1]
[1]National School of Computer Sciences (ENSI)
RIADI-GDL Laboratory, University of Manouba, La Manouba, Tunisia
{yassine.jamoussi, henda.benghezala}@ensi.rnu.tn
[2]IRISA/INRIA, University of Rennes I, Rennes, France
{maha.driss, Jean-Marc.Jezequel}@irisa.fr

ABSTRACT
The new paradigm for distributed computing over the Internet is that of Web services (WSs). One of the key ideas of this new paradigm is the ability to create value-added Service-Based Applications (SBAs) by composing pre-existing services. Building SBAs necessitates the discovery and the selection of the most appropriate WSs that fit closely users' functional and non-functional requirements. Due to the large number of WSs that are advertised over public and private registries and the various functional and non-functional capabilities that are required by users, discovery and selection of WSs have become a real challenge nowadays. In this paper, we present a WS composition approach that is built upon both perspectives: intentional and operational. In the intentional perspective, we propose to model users' requirements for SBAs using the MAP formalism and specify the required WSs using an Intentional Service Model (ISM). In the operational perspective, we propose to discover the required WSs by querying the service search engine *Service-Finder* and select the most appropriate WSs by using many-valued concept lattices. To validate our approach, we use an analytical technique that is the monitoring to verify that the selected WSs assure the required users' non-functional capabilities.

Keywords
Service-Based Applications, Users' requirements, Service modeling, Service discovery, Service selection, Service Monitoring.

1. INTRODUCTION
In the last ten years, the Service Oriented Architecture (SOA) has emerged as a powerful solution to enable interoperability between distributed software applications known as Web Services (WSs). The key aspect of the SOA is the use of standardized protocols and languages that cover all aspects related to the definition of WSs, their advertisement, and their binding [1]. With this generic and flexible architecture, SOA naturally addresses the problem of applications' integration. Indeed, WSs may be easily composed together into a new application, regardless specific implementation platforms and programming technologies. This Service-Based Application (SBA) may be further published as a service creating a new collaboration network between different organizations [2].

iiWAS2011, 5-7 December, 2011, Ho Chi Minh City, Vietnam.

Building SBAs necessitates the discovery and the selection of the most appropriate WSs that fit closely users' functional and non-functional requirements. Functional requirements define functionalities provided by WSs and non-functional requirements define Quality of Service (QoS) properties such as availability and response time, etc [3]. The discovery of WSs is achieved by querying a WS search engine to browse WSs using functional and non-functional criteria. Among the set of WSs that are obtained by discovery, only the most appropriate WSs that best match users' functional and non-functional requirements are selected to be composed. Nowadays, discovery and selection of the most appropriate WSs reveal a number of challenges. The first challenge is the large number and diversity of available WSs. A second challenge is that these WSs are advertised over multiple public and private registries. Finally, the third challenge is that the variety of users' functional and non-functional requirements requires costly and time-consuming selection of the most appropriate WSs.

In this paper, we propose a new requirement engineering approach for WS composition that is built upon both perspectives: intentional and operational. This approach allows: (i) modeling SBAs in terms of functional and non-functional users' requirements, (ii) discovering the required WSs, (iii) selecting the most appropriate WSs, and (iv) monitoring the selected WSs to verify that these services assure the required users' non-functional capabilities.

This paper is organized as follows. Section 2 presents an overview of the proposed approach. Next, Section 3 focuses on the intentional perspective throughout the explanation of the requirement elicitation and specification step. Section 4 focuses on the operational perspective throughout the explanation of the WS discovery, selection, and monitoring steps. Then, Section 5 reviews the related work. Finally, Section 7 provides conclusions and outlines future works.

2. APPROACH OVERVIEW
The overall approach that we propose to compose WSs on the basis of users' requirements is built upon both perspectives: intentional and operational. The intentional perspective brings out users' functional and non-functional requirements. The operational perspective captures WSs that best match users' requirements elicited and specified in the intentional perspective. The proposed approach is a process that consists of four successive steps 1-4 as it is shown by Figure 1.
The first step is performed in the intentional perspective. The remaining three steps are performed in the operational perspective. The following items summarize the five steps.

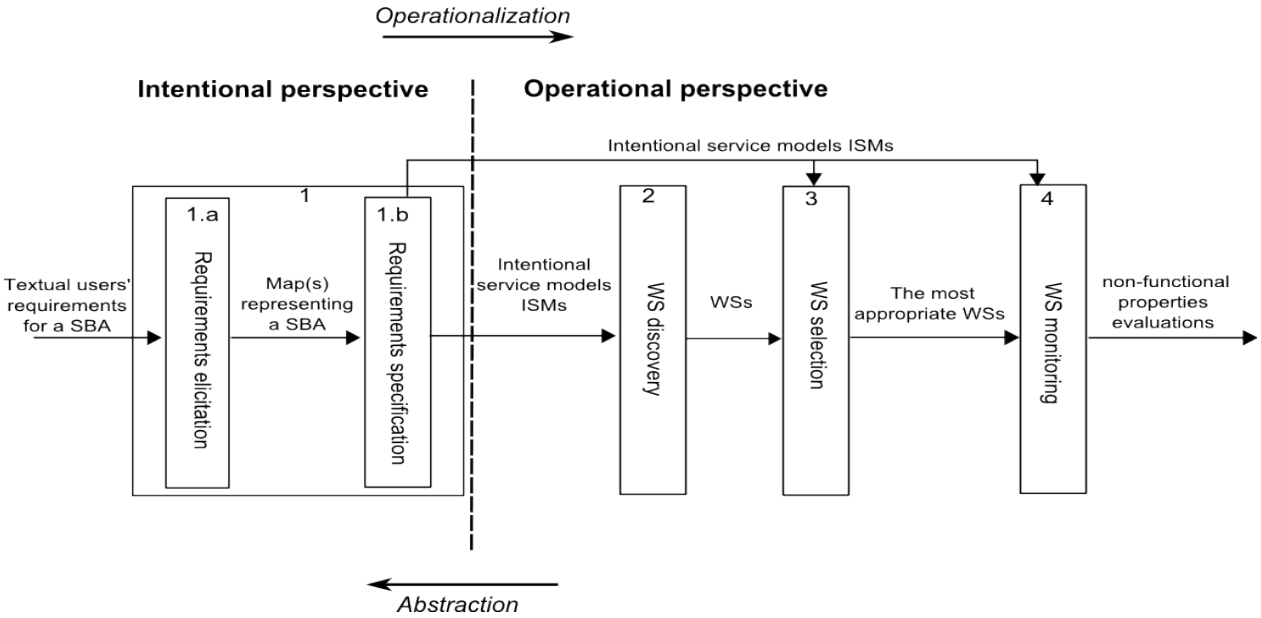

Figure 1. A multi-perspective approach for Web service composition

- ***Step1.Requirements elicitation and specification***: Users' requirements for SBAs are elicited in an intention-driven manner using the requirement engineering formalism MAP [4]. The MAP elicits and models users' requirements in a set of graphs called maps composed of intentions as nodes and strategies as edges. An intention presents a goal. A strategy presents an approach, a manner to achieve an intention. We refer to services presented by maps intentional services. These services focus on the intentions they allow to fulfill rather than on the functionalities they perform. In a previous work [5], an Intentional Service Model (ISM) was proposed to specify intentional services. This model omits the specification of QoS properties. In this work, we enhance the ISM to include QoS and we use this model to specify intentional services.

- ***Step2.WS discovery***: WSs that operationalize users' functional requirements are discovered by querying the WS search engine *Service-Finder* [6] by using a set of keywords. To efficiently discover WSs, we propose three-level filtration. In the first level, we omit redundant WSs. Services which pass the first level are filtered according to two QoS properties that are validity (i.e., we verify if the service endpoint is valid) and availability (i.e., we verify if the service is operational). In the third filter, we keep only services offering the required operation.

- ***Step3.WSs selection***: The most appropriate WSs that fit closely users' functional and non-functional requirements are selected automatically by applying Formal Concept Analysis (FCA) [7]. FCA is a mathematical framework that permits the identification of groups of objects sharing common attributes and organizes them into conceptual hierarchies called concept lattices. By using FCA, we construct many valued-concept lattices for WSs according to their QoS properties values.

- ***Step4. WS Monitoring***: selected WSs are monitored to verify that they assure the required functional and non-functional requirements. Monitoring experiments are conducted using *WildCat* [8], a generic monitoring framework for developing and supervising software applications. Services are observed by Wildcat sensors which provide QoS evaluations at runtime.

At the end of the process, the user receives as output a SBA composed by the most appropriate WSs that provide the functionalities and the QoS required by users and some sets of equivalent WSs, one per each specified intentional service. In case the evaluation results obtained by monitoring are not those expected by the users, we propose to refine this composition process at the three steps: (i) Requirements elicitation and specification step: the map and the ISMs can be refined by modifying the intentions and the QoS constraints, (ii) WSs discovery step: others keywords can be used to annotate a specific intentional service, or (iii) WSs selection step: a different WS can be selected in the list of discovered WSs. This process is iterative and it can be executed as many times as required to refine the selection and to adapt services to users' functional and non-functional requirements.

3. INTENTIONAL PERSPECTIVE

In the intentional perspective, we adopt a requirement-driven representation that allows a high level modeling of SBAs. We use the requirement engineering formalism MAP [4] to represent users' requirements and the Intentional Service Model (ISM) to specify services presented by maps. There are two main reasons for using the MAP formalism: first, the MAP was already applied to service modeling domain, so we can use previous knowledge and experiences. Second, the Map makes clear distinction between intentions and strategies to achieve them. This provides an explicit representation of requirements' variability.

3.1 Requirements elicitation

A map is a labeled directed graph with intentions as nodes and strategies as edges between intentions. An intention is a goal that can

be achieved by the performance of the map. Each map has two distinct intentions Start and Stop to respectively begin and end the navigation in the map. A strategy is an approach, a manner to achieve an intention. A map is composed of one or more sections. A section is a triplet <*SourceIntention* Ii, *TargetIntention* Ij, *Strategy* Sij> that captures a specific manner to achieve the target intention Ij starting from the source intention Ii with the strategy Sij.

Figure 2 depicts an example of a map of a currency converter application. This application has three intentions that are: *Locate geographically the customer*, *Obtain the current conversion rate*, and *Calculate the new value for price*. To achieve the *Locate geographically the customer* intention, we propose the *By converting IP address to country name* strategy. *By entering countries or currencies names* is proposed to fulfill the *Obtain the current conversion rate* intention. *By multiplying old price by currency conversion rate* is the strategy to accomplish the *Calculate the new value for price* intention. The currency converter application terminates with *By payment* or *By cancellation* strategies.

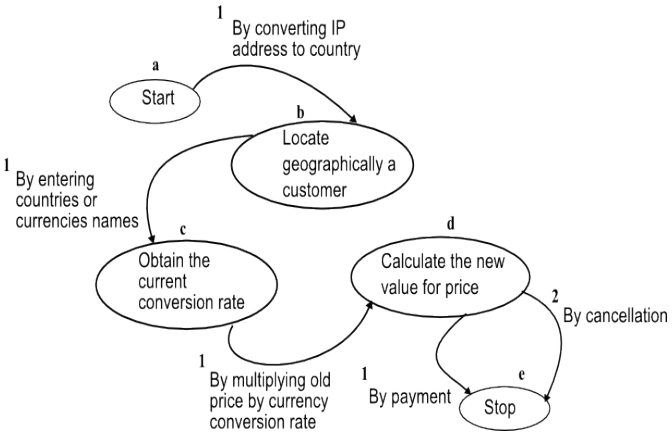

Figure 2. The map of the currency converter application

3.2 Requirements specification

To identify intentional services from maps, we propose to associate each section of the map to a service. The Intentional Service Model (ISM) is used to specify these services. ISM was proposed in a previous work [5], but this proposition omits the specification of QoS properties. In this work, we enhance the ISM to include QoS and we use this model to specify intentional services. The key idea of the ISM is that a service allows the achievement of an intention given an initial situation and terminating with a final situation. As shown by colors used in Figure 3, our ISM model describes intentional services through four main aspects: the service interface, the service behavior, the service composition and the service QoS.

- ***The intentional service interface***: there are three elements specifying intentional service interface: *Intention*, *Initial Situation*, and *Final Situation*. An intentional service allows the achievement of an *Intention*. *Initial Situation* and *Final Situation* are respectively input and output parameters of the intentional service.

- ***The intentional service behavior***: there are two elements specifying intentional service behavior: *Pre-condition* and *Post-condition* that are respectively the initial and the final states of the intention.

- ***The intentional service composition***: there are two kinds of services: *Atomic* and *Aggregate* services. An *Atomic* service has an operationalized intention that can be achieved directly by an operational service. An *Aggregate* service has a high level intention that should be decomposed till operational services are identified.

- ***The intentional service QoS***: there are two elements specifying intentional service QoS: *QoS Characteristic* and *QoS Constraint*. *QoS Characteristic* is the quality to be attained or preserved. *QoS Constraint* allows the expression of preferences over a *QoS Characteristic.*

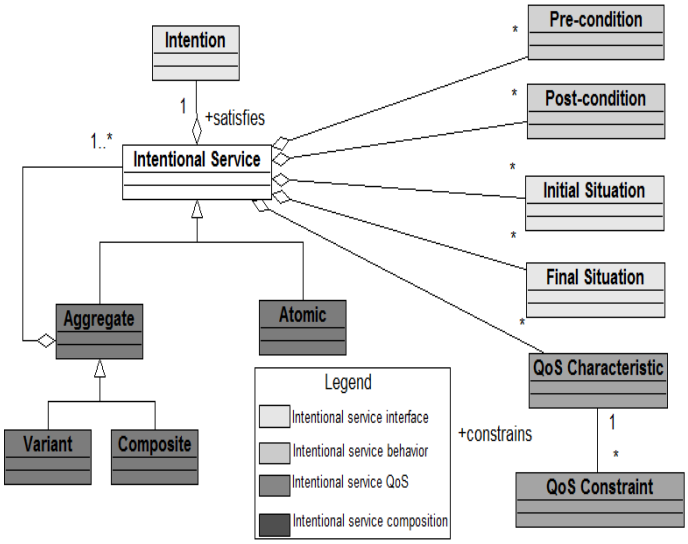

Figure 3. The Intentional Service Model (ISM)

4. OPERTIONAL PERSPECTIVE

To each intentional service, we propose to associate an operational WS that can perform the expected intention. In the operational perspective, we propose to discover, select, and monitor operational WSs to assure that they best fit users' functional and non-functional requirements.

4.1 WS discovery

To discover operational WSs, we make use of the WS search engine *Service-Finder* [6]. *Service-Finder* made available searching in more than 25.000 services with 200.000 related Web pages [6]. In order to understand the problem of WS discovery and selection, we propose to demonstrate the discovery and the selection of the intentional service $S_{LocateGeographicallyACustomer}$ taken from the currency converter application case study illustrated in Section 3. To query *Service-Finder*, we consider the following keywords {IP, Country} which describe the service $S_{LocateGeographicallyACustomer}$. For this query, Service-Finder returns a set of 93 WSs[1].

To reduce this set, we process a three-level filtration. In the first level, we omit redundant WSs (Filter1). Services which pass the first level are filtered according to two QoS properties (Filter2):

- Validity: we verify if the service endpoint is valid.
- Availability: we verify if the service is operational.

In the third filter, we keep only services offering the required operation (Filter3). In this example, we are searching an operation that converts ip address to country name. This operation takes ip as input and returns country name as output. In Table 1, we can see the resulting sets of filtered services.

[1] This result is obtained on August 5th, 2011.

Table 1. The number of filtered WSs for the intentional service $S_{LocateGeographicallyACustomer}$.

Intentional Service	*Service-Finder*	Filter 1	Filter 2	Filter 3
$S_{LocateGeographicallyACustomer}$	93	57	42	17

4.2 WS selection

Services returned by discovery step fit closely users' functional requirements. In the selection step, we propose to select the most appropriate services that best match users' non-functional requirements that are specified by the ISM models. For the $S_{LocateGeographicallyACustomer}$ service, we suppose that users require services having an availability that is greater than 98% and a response time that is less than 300ms. Real time monitoring values of services availability and response time are provided by *Service-Finder*.

Availability and response time values of the 17 services, obtained after the discovery step, are shown by Table 2.

Table 2. The Availability and response time values of the 17 WSs satisfying $S_{LOCATEGEOGRAPHICALLYACUSTOMER}$

WS	Name	Availability (%)	Response Time (ms)
S1	IP2Geo	100	198
S2	GeoIPService	85	395
S3	IP2CountryService	68	263
S4	ip2loc	98	22
S5	IPCountry_Service	99	390
S6	SIGeoLocation	99	224
S7	config	99	395
S8	GeoCoder	100	328
S9	IpAddressSearchWebService	96	691
S10	router	100	40
S11	CountryService	99	252
S12	UserInfo	99	603
S13	NetworkService	75	255
S14	network	100	220
S15	MoralPolitics	98	343
S16	Nets	91	699
S17	DirectTrackWebServices	100	263

To classify these services according to their QoS values, Formal Concept Analysis (FCA) is used. FCA [7] is a formal framework for clustering objects along the attributes they share. It describes clusters, called formal concepts, both extensionally and intentionally, i.e., as sets of objects and sets of shared attributes, and organizes them hierarchically, according to a binary incidence relation, into a complete lattice, called the concept lattice. FCA considers a dataset as being organized into a formal context, i.e., a triple (O,A,R), where O is a set of objects, A is a set of attributes, and R is the binary incidence relation between O and A, $R \subseteq O \times A$. The formal context can be easily represented by a cross table. Usually a data set would be expressed by each attribute with a value range of the attribute. In that case, many-valued context can be used, but formal context cannot express these kinds of data. A many-valued context (O,A,V,R) is composed of a set O of objects, a set A of attributes, a set V of attribute values and a ternary relation R between O, A, and V (e.g., $R \subseteq O \times A \times V$). For our problem, we define a many-valued context K where objects represent services and attributes represent normalized values of availability and response time (e.g., we transform availability values (response time values, respectively) provided by Table 2 into rage of [0,1] by dividing them by the maximum value of availability that is 100 (by the maximum value of response time that is 699, respectively)). Context K is illustrated by Figure 4.

ManyValuedServices*QoS | BinaryServices*QoS

A	B	C
ManyValuedServices*QoS	Availability	Response time
S1	[1]	[0.28]
S2	[0.85]	[0.56]
S3	[0.68]	[0.37]
S4	[0.98]	[0.03]
S5	[0.99]	[0.55]
S6	[0.99]	[0.32]
S7	[0.99]	[0.56]
S8	[1]	[0.46]
S9	[0.96]	[0.98]
S10	[1]	[0.05]
S11	[0.99]	[0.36]
S12	[0.99]	[0.86]
S13	[0.75]	[0.36]
S14	[1]	[0.31]
S15	[0.98]	[0.49]
S16	[0.91]	[1]
S17	[1]	[0.37]

Figure 4. The many-valued context K linking services to normalized values of availability and response time

In order to extract formal concepts from the many-valued context, each attribute of the many-valued context should be transformed into a derived formal context based on scale context. This procedure is called conceptual scaling. Figure 5 shows the derived binary formal context K' related to conceptual scales expressing users preferences: high availability $\geq$ 98% (which correspond to normal values that are ≥ 0.98) and low response time $\leq$ 300ms (which correspond to normal values that are ≤ 0.42).

ManyValuedServices*QoS | BinaryServices*QoS

A	B	C
BinaryServices*QoS	High availability	Low response time
S1	X	X
S2	0	0
S3	0	X
S4	X	X
S5	X	0
S6	X	X
S7	X	0
S8	0	0
S9	0	0
S10	X	X
S11	X	X
S12	X	0
S13	0	X
S14	X	X
S15	X	0
S16	0	0
S17	X	X

Figure 5. The derived binary formal context K' related to conceptual scalings: high availability $\geq$ 98% and low response time $\leq$ 300ms

FCA organizes formal concepts into complete lattices, called concept lattices. The lattice is composed of a set of formal concepts. A concept is defined as a pair (E,I) with $E \subseteq O$ is called extension and $I \subseteq A$ is called intension and intent(E)=I$\wedge$ extent(I)=E . The lattice structure allows easy navigation and search as well as optimal representation of information. Figure 6 depicts the concept lattice derived from our binary formal context K'. This lattice is built using the *Galicia* (Galois Lattice Interactive Constructor) tool [8]. *Galicia*

is a multi-tool open-source platform for creating, visualizing, and storing concept lattices.

From this lattice, we conclude that the most appropriate WSs are services that are presented by the concept which is in the bottom of our lattice (concept 0). The extension of this concept includes 7 services S1, S10, S11, S14, S17, S4, and S6. These services fulfill the *LocateGeographicallyACustomer* intention and have a high availability (≥ 98%) and a low response time (≤ 300ms).

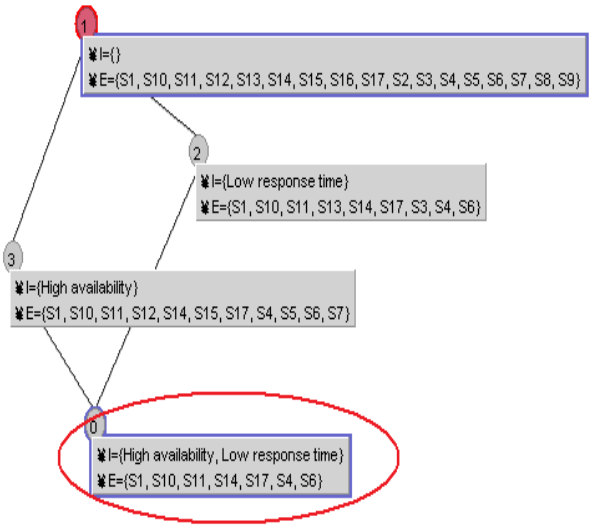

Figure 6. The lattice of the formal context K'

4.3 WS monitoring

To validate our approach, we use monitoring to verify that the selected WSs assure the required users' non-functional capabilities. Monitoring experiments are conducted using *WildCAT* [9], a generic framework for developing and supervising software applications. *WildCAT* allows monitoring software applications by easily organizing and accessing sensors through a hierarchical organization. *WildCAT* defines monitoring models as oriented tree structures. A monitoring model is composed of resources as nodes and attributes as leaves. Resources allow structuring the monitoring model. Attributes represent *WildCAT* sensors. The monitoring model of the currency converter application is illustrated by Figure 7.

The root resource is the currency converter orchestration. Nodes are represented by WSs of the currency converter application. Each WS has two sub-resources: availability and response time. Finally, each sub-resource has some attributes describing its current status.
For the *Availability* resource, the *Uptime* and *Downtime* attributes hold respectively the service uptime and downtime. For the Response time resource the *Treq* and *Tresp* attributes hold respectively the time when the first byte of the request of the WS is sent and the time when the last byte of the response of the WS is received.

To measure WS availability and response time, we use the following metrics:

- Availability(%)=[Uptime/(Downtime+Uptime)]*100
- Response time(ms)=Tresp-Treq

To evaluate the availability and the response time values of each WS of the currency converter application, we have performed a set of monitoring experiments and we have considered the average values. Monitoring results related to the 7 WSs corresponding to $S_{LocateGeographicallyACustomer}$ are provided by Figure 8. Monitoring results show that only S1, S10, and S4 have an availability that is greater than 98% and a response time that is less than 300ms as required by users.

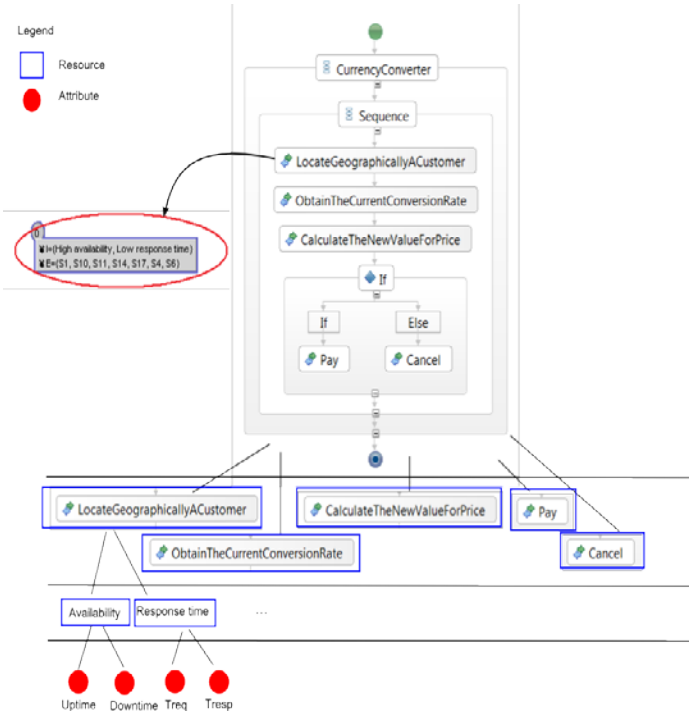

Figure 7. The *WildCAT* monitoring model of the currency converter application

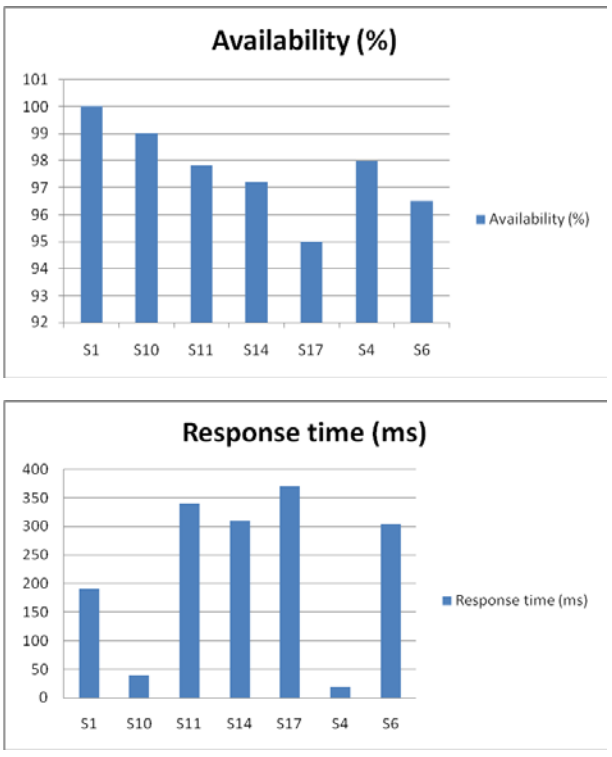

Figure 8. The availability and response time monitoring results

5. RELATED WORK

Many works in the literature like [10], [11], [12], [13], [14], [15], and [16] have addressed the classification of WSs using concept lattices.

Peng et al. [10] present an approach to classify and select services. They build lattices upon contexts where objects are WSs and attributes represent the operations of these services. The approach allows similar services clustering by applying similarity search techniques that compare operation descriptions and input/output messages data type.

Aversano et al. [11] propose an approach, based on FCA, to understand relationships between services, as well as between operations of a complex service, by analyzing service interfaces and documentation. Concept lattices are built upon a context obtained from keywords extracted from service descriptions or operation parameters. The approach allows an analyst to cluster similar services, highlights hierarchical relationships, commonalities, and differences between services.

Azmeh et al. [12] present a similar approach to classify and select services using the FCA. They propose *WSPAB* tool that permits the discovery, the automatic classification, and the selection of WSs. Classification is accomplished by defining a binary relation between services and operation signatures. In [13], Azmeh et al. uses FCA to classify WSs by keywords extracted from their WSDL description files. The obtained lattices can be used to identify relevant services and their substitutes.

Fenza and Senatore [14] describe a system for supporting the user in the discovery of semantic WSs that best fit personal requirements and preferences. Through a concept-based navigation mechanism, the user discovers conceptual terminology associated to the WSs and uses it to generate an appropriate service request which syntactical matches the names of input/output specifications. The approach exploits the fuzzy FCA for modeling concepts and relative relationships elicited from WSs. After the request formulation and submission, the system returns the list of semantic WSs that match the user query.

Contrarily to our approach, these works [10], [11], [12], [13], [14] do not deal with QoS properties to classify WSs.

In [15], Chollet et al. propose an approach based on FCA to organize the services registry at runtime and to allow the best service selection among heterogeneous and secured services. The services registry is viewed as a formal context where the services are the objects and the services types, functional, and non-functional characteristics (security characteristics) are the attributes.

In [16], Driss et al. propose a requirement-centric approach to WS modeling, discovery, and selection. Driss et al. consider formal contexts with services as objects and QoS characteristics (availability and response time) as attributes. The obtained lattices are used to check out relevant (that best fit functional requirements) and high QoS WSs.

All the works detailed above [10], [11], [12], [13], [14], [15], and [16] define binary contexts and not many-valued contexts. The advantage of many-valued contexts is that they allow the generation of many-valued lattices which facilitates the identification of the relevant concepts (i.e., that match users QoS preferences) by formulating queries defined in terms of conceptual scales. In addition, all these works do not provide any analytical technique (e.g., testing, simulation, monitoring, etc.) to validate the WS discovery and selection.

6. CONCLUSION

In this paper, we presented a new multi-perspective approach for Web service composition. This approach is built upon two perspectives: the intentional perspective and the operational perspective. The intentional perspective provides a requirement-centric view of the service-based applications. Whereas, the operational perspective provides a function-centric view. Main contributions of this approach are: (i) the coupling between the two perspectives to deal with users' functional and non-functional requirements, (ii) the use of many-valued concept lattices to select the most appropriate WSs according to a set of non-functional requirements, and (iii) the QoS assurance ensured by the monitoring technique.
Possible extension of our work is to define and apply retroactive adaptation strategies in case the evaluation results obtained by monitoring are not those expected by users.

7. REFERENCES

[1] Huhns, M.N. and Singh, M. P. 2005. Service-oriented computing: Key concepts and principles. *IEEE Internet Comp*. 9, 1 (2005), 75–81.

[2] Rao, J. and Su, X. 2005. A Survey of Automated Web Service Composition Methods. *LNCS, Springer*.

[3] Menascé, D. A. 2002. QoS issues in web services. *IEEE Internet Comp*. 6, 6 (2002), 72-75.

[4] Rolland, C. and Prakash, N. 2000. Bridging the gap between organizational needs and ERP functionality. *Requirements Engineering*. 5, 3 (2000), 180-193.

[5] Rolland, C., Kaabi, R. S. and Kraeim, N. 2007. On ISOA: intentional services oriented architecture. In *Proceedings of CAISE'07*.

[6] http ://www.service-finder.eu/

[7] Ganter, B. and Wille, R. 1999. Formal Concept Analysis: Mathematical Foundations. *Springer-Verlag New York, Inc., Heidelberg*.

[8] http://galicia.sourceforge.net/

[9] David, P. and Ledoux, T. 2005. WildCAT: a generic framework for contextaware applications. In *Proceedings of MPAC'05*.

[10] Peng, D., Huang, S., Wang, X. and Zhou, A. 2005. Management and retrieval of web services based on formal concept analysis. In *Proceedings of CIT'05*.

[11] Aversano, L., Bruno, M., Canfora, G., Di Penta, M. and Distante, D. 2006. Using concept lattices to support service selection. *Int. J. of Web Serv. Res.* 3, 4 (2006), 32–51.

[12] Azmeh, Z., Huchard, M., Tibermacine, C., Urtado, C. and Vauttier, S. 2008. Wspab: A tool for automatic classification and selection of web services using formal concept analysis. In *Proceedings of ECOWS'08*.

[13] Azmeh, Z., Huchard, M., Tibermacine, C., Urtado, C. and Vauttier, S. 2010. Using concept lattices to support web service compositions with backup services. In *Proceedings of ICIW'10*.

[14] Fenza, G. and Senatore, S. 2010. Friendly web services selection exploiting fuzzy formal concept analysis. *Soft Comput.* 14. (2010), 811-819.

[15] Chollet, S., Lestideau, V., Lalanda, P., Colomb, P., and Moreno, D. 2010. Heterogeneous service selection based on formal concept analysis. In *Proceedings of* NCSE'10.

[16] Driss, M., Moha, N., Jamoussi, Y., Jézéquel, J.-M. and Hajjami Ben Ghézala, H. 2010. A requirement-centric approach to web service modeling, discovery, and selection. In *Proceedings of* ICSOC'10.