User Requirement and Behavioral Aspects in Web Service Discovery

Wala Ben Messaoud[1], Khaled Ghedira[1] and Youssef Ben Halima[2]
[1]*Institute ENSI University of Manouba/SOIE, 41 Liberty Street Bouchoucha, Bardo, Tunisia*
[2]*ENSI University of Manouba/RIADI Labs, Manouba University Campus, Manouba, Tunisia*

Keywords: Behavioral Aspect, State Chart Scheduling, Web Service, Web Service Discovery, WordNet.

Abstract: In web service (WS) discovery, behavioral aspect has been defined as the sequence of WS operations. The motivation to introduce the behavioral aspect is to offer to the consumer the possibility to choose his WS according to his requirements. The aim is to include the execution manner of WS operations as a new criterion and to apply a selection method if more than one WS candidate is filtered. In this paper, we envision to implement WS discovery approach based on behavioral aspects to fulfill the selection of the precise execution order. This approach ensures an execution order of operations in accordance to consumer needs. The execution manner criterion is defended by state chart as a scheduling method and WordNnet as a lexical database. Moreover, semantic equivalences have to be considered in order to solve equivalence between many WS candidates which satisfy consumer needs.

1 INTRODUCTION

The literature on web service discovery is almost common in recognizing the existence of a major problem in the WS consumer's requirements. Current work on service discovery focuses on discovery types not on the analysis of consumer intervention.

WS discovery is the process of satisfaction of a user request according to his requirements. It refers to the process of finding WS that implements the technique of search desired, interviewing service books, to know what WS is available for binding. Our approach unlike the other discovery approaches, allows WS consumer to involve his exigency by entering some sentences as a WS query. The aim is to satisfy WS consumer by analyzing his inputs.

The WS consumer requires a new aspect allowing functional phase (organized operations, free operations) and non functional phase (cost, time, availability). As a solution, we define the behavioral aspect as the execution manner of WS operations. Indeed, a consumer who requires looking for a WS with a tidy list of exigencies may not be satisfied by ordinary WS discovery. Actually, the motivation to introduce the behavioral aspect is to offer to the consumer the possibility to choose his WS according to his needs. The aim is to include requirement criterion as a new aspect and to apply a selection method if more than one WS candidate are filtered. The behavior aspect should guarantee quality of service (cost, reliability, time ...).

Our approach consists on using Statechart as a scheduling method to highlight the execution order of WS operations and WordNet as a lexical database to prove the semantic part of consumer query. We used BPEL4WS (Business Process Execution Language for Web Services) to specify and execute business process for WS composition and orchestration. More precisely, to explain how WS operations are invoked and executed.

The remainder of the paper is organized as follows: Section 2 presents some concepts used in our approach. Our solution is reported in section 3. Section 4 explains more our approach by an example. In section 5, we feature the related work. Conclusions are presented in section 6.

2 CONCEPTS

Our approach brings answers to many posed issues. It is based on some concepts facilitating the implementation of the different phases of the approach like WordNet, intending to find synonyms to all consumer inputs.

In the goal to order these inputs, we use state charts where the WS operations are the transitions allowing to move from one state to another. Once we have prepared the state chart part, we launch the

semantic aspect of WS discovery, we extract the BPEL sequencing of each WS found and then we select the relevant one.

2.1 WordNet

The goal of WordNet was to develop a system that would be consistent with the knowledge acquired over the years about how human beings process language.

In (Miller, 1995), WordNet is defined as a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations.

WordNet is a large semantic network interlinking words and groups of words by means of lexical and conceptual relations represented by labeled arcs. WordNet's building blocks are synonym sets (synsets), unordered sets of cognitively synonymous words and phrases (Christiane, 2005).

The authors in (Morato, 2004), define WordNet as one of a series of manually compiled electronic dictionaries, is restricted to no specific domain and covers most English nouns, adjectives, verbs and adverbs. WordNet offers researchers a cost-free use and well-documented open code. It is an ideal tool for disambiguation of meaning, semantic tagging and information retrieval.

In our work, the consumer inputs is a set of keywords that define his requirements. If we don't specify clearly these inputs, we risk falling in linguistic polysemy case. So, we seek to find synonyms to all consumer inputs to properly filter the service concerned. That's why we have chosen to use WordNet as ontology.

2.2 Statecharts - Automaton

In literature, authors define statechart as visual formalism for description of complex systems behaviour. Digital controllers, which act as reactive systems, can be very conveniently modelled with statecharts and efficiently synthesized in modern programmable devices (Łabiak, 2010).

In formal grammar, an alphabet Σ is a finite and not empty set of symbols. Σ * is the closure of Σ. A language on an alphabet Σ is a subset of the set Σ *.

A transition diagram allows achieving an operational vision of the concept of language. It is a finite collection of states and transitions.

The statechart notation was developed by David Harel (Harel, 1987). Statechart diagrams are useful for modelling the lifetime of an object. They are used to describe the system behavior using a finite automaton.

The automaton is represented as a directed graph known as state graph which consists of a finite set of vertices known as nodes, together with a set of directed links between pairs of vertices called arcs. Vertices are represented by circles and arcs by arrows. We can also represent an automaton with a state-transition table (Bhattacharjee, 2014).

The transition table can be associated to the automaton which describes extensively the transition function δ. A column is a character of the alphabet. A line is a state of the automaton (the initial state is preceded by an arrow "→", the final state is preceded by a star "*").

The value δ (q, a) for $q \in Q$, $a \in \Sigma$ corresponds to the state at the intersection of the row q and the column a. Note that from this table it is easy to find all the statements and the alphabet and thus identify exactly the automaton.

An automaton reads a finite string of symbols a_1, a2, a_n, where $a_i \in \Sigma$, which is called an *input word*. The set of all words is denoted by Σ*. *Accepting word is the* word $w \in \Sigma^*$ which is accepted by the automaton if $q_n \in F$. An automaton can recognize a *formal language*. The language $L \subseteq \Sigma^*$ recognized by an automaton is the set of all the words that are accepted by the automaton. The Deterministic Finite Automata has a finite internal memory available. At each input letter the state of the internal memory is changed depending on the letter scanned.

The previous memory state and the input letter together determine what the next state of the memory is. The word is accepted if the internal memory is in an accepting state after scanning the entire word (Kari, 2013).

In our approach, we use automaton as a scheduling method of WS operations (where we order the WS operations in the aim to respect the non functional/functional properties). Each operation represents a transition from a state to another. The automation generates a language (ordering list of operations) that specify the words to accept.

2.3 WSDL

A WSDL document is, at its simplest, a collection of elements contained within a root definition element. These elements describe a service and how an endpoint implementing that service is accessed (ALBRESHNE, 2009)

Each WSDL includes two parts, the abstract and the concrete. Abstract part describes the messages

sent and received. The operation associates a message exchange pattern with one or more messages.

As for concrete part, it specifies transport and wire format details for one or more interfaces, a port (an endpoint) associates a network address with a binding and a service which groups together endpoints that implement a common interface.

An operation is similar to a function in a high level programming language. A message exchange is also referred to as an operation. Operations are the focal point of interacting with the service.
In our work, we use WSDL for each WS to extract the execution order of operations.

2.4 BPEL4WS

BPEL for WS is a programming language for the execution of business processes. It is based on WSFL (Web Services Flow Language) and XLANG (XML LANGuage), is derived from XML.

According to (Milanvoic, 2004) and (OASIS, 2007), BPEL composes web services to get a specific result. The composition result is a named process, partners are defined as services and activities are described by exchanged messages. In other words, a process contains a set of activities and it invokes external partner services using a WSDL interface.

BPEL is used to describe the execution of business process implicating WS. It consists on business BPEL allowing communication with WS, handling XML Data and managing exceptions.

BPEL provides several structure activities:
- <sequence>: define an ordered sequence of WS activities.
- <flow>: define parallel activities.
- <switch>: Case-switch construct for implementing branches.
- <while>: define loops.
- <pick>: select one of several alternative paths.

In WS discovery, there are three layers; operational, organizational and intentional. Intentional layer enables modeling purposes. It is a conceptualization of strategic needs of required modeling by an individual subject, group of individuals, a work unit or organization that involved in the system development process.

In some cases, the WS consumer cannot be a domain expert, he launch his query by entering a list of sentences. These sentences will be transformed to keywords by an algorithm to facilitate the satisfaction process. The consumer inputs can be a list of WS that need to be orchestrated as operations.

3 SOLUTION: BEHAVIORAL BASED APPROACH

The WS consumer seeks his WS with a list of well-defined requirements (functional and non-functional) but he is not satisfied by the ordinary types of discovery.

Syntactic discovery aims to compare between the syntactic query based on keywords and syntactic descriptions of WS. WS consumer launch his query by entering a set of keywords, the result of the comparison between syntactic query based on keywords and syntactic descriptions of the services is a set of WS that don't satisfy the consumer exigencies (the WS name is exactly the keyword entered by the consumer but the content has not the same meaning sought).

The semantic discovery is mainly based on ontology, defined as structured set of terms and concepts representing the meaning of information field, developed to facilitate knowledge sharing and reuse. WS consumer launch his query by entering a set of keywords, the result is a set of WS that don't satisfy the consumer exigencies. For this reason, we intend to define a new WS discovery approach based on behavioral aspect according to our definition. We define the behavior as the execution manner of WS operations. WS discovery approach based on behavioral aspect ensures an execution order of operations in accordance to consumer needs.

The purpose of this paper is to design a discovery technique for choosing the WS with the most relevant behavior. So, the query language to develop should be based on system statechart. It is used to check the compatibility of behavior required by the customer and those of WS found.

The work requires its valuation by an implementation that acquires to client to get his WS with the execution order of the desired operations. This implementation creates a WS automaton according to the operations order. WS automaton accepts only languages that correspond to it. These languages will be generated from BPEL files (Business Process Execution Language) of WS operations searched in WS directories. Some languages will be selected if they are accepted by the WS automaton.

Figure 1 shows that the implementation is done in three phases.

3.1 Phase 1 - Transform Consumer Sentences to Keywords

If the WS consumer is not a domain expert, he may enter his requirements as sentences. An algorithm is defined to transform each sentence in a keyword used in Phase 2.

3.2 Phase 2 - Create WordNet File for Each Keyword - Create Automaton

WordNet is used to define synonyms file for each keyword entered by the consumer. The analysis of consumer inputs is done on a semantic level not on a syntactic level.

In the same time, these keywords allow to create the automaton corresponding to the behavior of searched WS (defining the standard language to accept).

The automaton is defined by an initial state, a list of transitions and a list of final states. A number of states are defined according to the consumer inputs. To switch from a state to another, we must pass through a transition. Consumer keywords define transitions.

3.3 Phase 3 - Extract Sequences from WSDL/BPEL Files of WS Searched - Select the Relevant WS

Consists on searching WS semantically (Semantic Aspect) basing on non functional properties (NFP). For each WS selected, we extract the list of its operations from WSDL (if it is a simple WS) or from BPEL file (if it invoke other WS). The extracted sequencing is transformed to word that will be accepted or rejected by the automaton already defined.

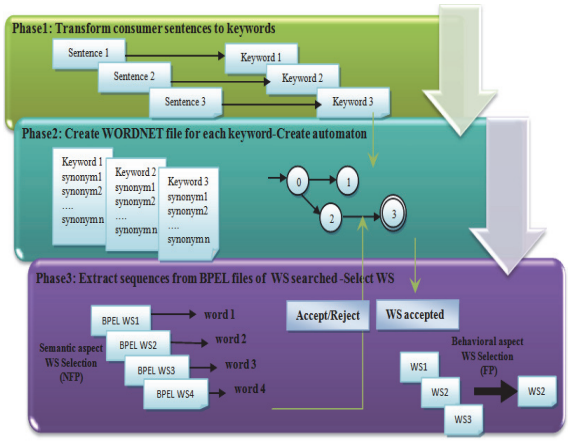

Figure 1: Implementation Steps of Behavioral approach.

Accepted words correspond to WS accepted. The last step in Phase 3 is to select the most relevant WS basing on functional properties (FP) that we called (Behavioral Aspect).

4 EXAMPLE: STAY RESERVATION

Let's consider the example named «Stay reservation» in Figure 2, where the consumer wishes to book a plane ticket, rent a car, buy a concert ticket and book a hotel stay (or rent a house) with a heated pool. All these operations should guarantee minimum transfer cost and reduced transfer time.

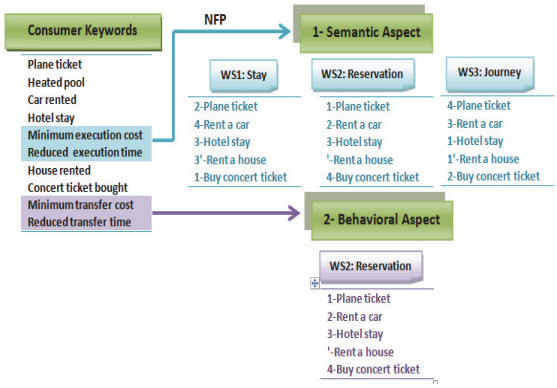

Figure 2: Behavioral WS discovery.

The consumer launches his query by entering a list of sentences such as «Plane ticket», «Heated pool», «Rent a car», «Hotel stay», «Minimum execution cost», «Reduced execution time», «Rent a house», «Buy concert ticket», «Minimum transfer cost» and «Reduced transfer time».

The first step is to specify NFP from FP. NFP are used in semantic aspect as selection properties. In this example, «Minimum execution cost» and «Reduced execution time» are NFP. As a result, we find WS1 named «Stay», WS2 named «Reservation» and WS3 named «Journey». All of this WS satisfy the consumer needs semantically.

For the behavioral aspect, «Minimum transfer cost» and «Reduced transfer time» are chosen as FP to select the most relevant WS.

To explain in more detail, we follow the steps mentioned previously:

- *Transform consumer sentences to keywords:*

Phase 1 presented in Figure 3 consists on defining an algorithm to transform consumer inputs to keywords

usable in phase 2.

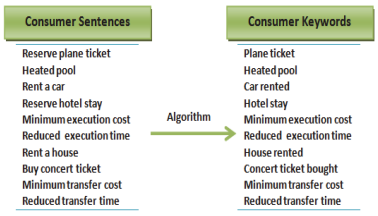

Figure 3: Stay Reservation: Phase 1.

➢ *Create WordNet file for each keyword:*

The system launches WordNet. As a result, we get for each keyword a file with a list of its synonyms. Each row of WordNet file is a keyword synonym. Figure 4 presents two examples respectively bonded to keywords «Plane ticket» and «Car rented».

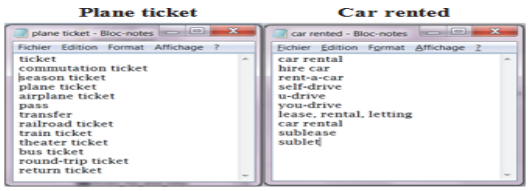

Figure 4: WordNet result.

➢ *Create automaton:*

At the same time, to respect consumer's requirements, we should consider it to create automaton.

The automaton in Figure 5 shows all states that the WS «Stay reservation» can be in during the course of its life. Furthermore, it shows the possible transitions between the states and the events that initiate these transitions.

We follow the automaton to check the compatibility of behavior required by the customer and those of WS found. The following abbreviations are used to develop the automaton.

Firstly, we define the states:

0: Stay not reserved
1: Plane Ticket reserved
2: Car rented
3: Hotel Stay reserved
4: House rented
5: Concert Ticket bought

Secondly, we define the transitions by the following abbreviations:

- Reserve_ticket : P
- Rent_ car : C
- Reserve_Hotel : S
- Rent_ house : H
- Buy_Concert_ticket : B

The WS automaton $(Q, \Sigma, \delta, q_0, F)$ is defined by:

$Q = \{0, 1, 2, 3, 4, 5\}$: the states number depends on operations number. (It is equals to the real operations number +1). The real operations number is calculated by neglecting free operations and for the parallel operations we just count one operation. In our case, we have in general five operations. If we neglect the free operation (Buy_Concert_ticket) and we count just one operation for the two parallel operations (Reserve_Hotel and Rent_House), we will have a number of four real operations. So, the states number equals to five (the real operations number=4 +1).

$\Sigma = \{P, C, S, H, B\}$

$\delta(0, P)=1$, $\delta(0, B)=5$, $\delta(1, C)=2$, $\delta(1, B)=5$, $\delta(2, S)=3$, $\delta(2, H)=4$, $\delta(2, B)=5$, $\delta(3, B)=5$, $\delta(4, B)=5$, $\delta(5, P)=1$, $\delta(5, C)=2$, $\delta(5, C)=3$, $\delta(5, H)=4$

The transitions table is represented by a matrix where the rows are the states and the columns are the operations.

$q_0 = 0$: the initial state is always the 0.

$F = \{3, 4, 5\}$: final state is defined if we look over all the transitions (for the parallel operations, we count just one). In this example, to achieve 3, we pass by PBCS. To achieve 5, we pass by PCSB. To achieve 4, we pass by PBCH.

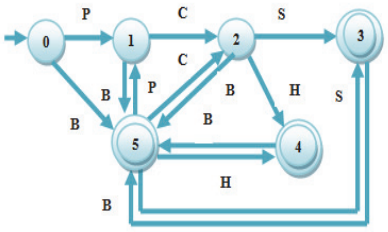

Figure 5: Automaton of WS «Stay reservation».

Then we deduce the transition table to facilitate the extraction of language to accept:

Table 1: Transition table.

	P	C	S	H	B
→0	{1}	Ø	Ø	Ø	{5}
1	Ø	{2}	Ø	Ø	{5}
2	Ø	Ø	{3}	{4}	{5}
*3	Ø	Ø	Ø	Ø	{5}
*4	Ø	Ø	Ø	Ø	{5}
*5	{1}	{2}	{3}	{4}	Ø

According to the transitions table, the accepted expressions are:

- {PBCS}: 1- Reserve_ticket 2- Buy_Concert_ticket 3- Rent_ car 4- Reserve_Hotel.
- {PBCH}: 1- Reserve_ticket 2- Buy_Concert_ticket 3- Rent_ car 4- Rent_ house.
- {PCBS}: 1- Reserve_ticket 2- Rent_ car 3- Buy_Concert_ticket 4- Reserve_Hotel.
- {PCBH}: 1- Reserve_ticket 2- Rent_ car 3- Buy_Concert_ticket 4-Rent_ house.
- {PCSB}: 1- Reserve_ticket 2- Rent_ car 3- Reserve_Hotel 4- Buy_Concert_ticket.
- {PCHB}: 1- Reserve_ticket 2- Rent_ car 3- Rent_ house 4- Buy_Concert_ticket.
- {BPCS}: 1- Buy_Concert_ticket 2- Reserve_ticket 3- Rent_ car 4- Reserve_Hotel.
- {BPCH}: 1- Buy_Concert_ticket 2- Reserve_ticket 3- Rent_ car 4- Rent_ house.

So, all other execution enchainment of WS operations will be rejected.

- *Extract sequences from WSDL or BPEL files of WS searched:*

The first step is to launch WS discovery semantically basing on NFP «Minimum execution cost» and «Reduced execution time». As a result, we find WS1« Stay», WS2«Reservation» and WS3«Journey». This step is an ordinary WS discovery based on semantic aspect that we can find in many works like (Kritikos, 2007).

The second step is to extract the sequencing of WS operations from the concerning file of each WS filtered. More precisely, to extract words that will be accepted or rejected by the automaton defined previously.

Every WSDL file contains the tag <operation> that defines the list of WS operations. We extract the execution order and transform it to words. Hence, all the WS words are ready to be filtered.

If we have composed WS, every BPEL file contains the tag <sequence> that defines an ordered sequence of WS activities. We extract this WS operations sequencing and transform it to word. Hence, all the WS words are ready to be filtered.

- *Select the relevant WS:*

Basing on FP «Minimum transfer cost» and «Reduced transfer time», we launch the selection phase of the most relevant WS.

The services candidate namely WS1«Stay», WS2 «Reservation» and WS3 «Journey» satisfy the consumer semantically but only WS2 is selected as the most relevant WS. Calculations are made in order to reduce transfer time and transfer cost.

5 RELATED WORK

The WS behaviour in literature is described by sequences of messages, data types, data constraints and properties that specify time limits within\where messages are exchanged (Elabd, 2011).

Previously, Maamar et al. propose an approach for modelling and specifying behaviours of WS in (Maamar, 2009). This approach sheds the light on two types of behaviours: control (that demonstrate the business logic that supports the functioning of a WS) and operational (that regulates the execution progress of this control behaviour by stating actions to carry out and the constraints to put on this progress). The idea is to coordinate both behaviours at run-time by developing conversational messages and transmit details between these two behaviours. Unlike (Maamar, 2009), our approach treats behaviour as a sequence of ordered operations and abstracts away from considering behaviours conversation.

Another alternative is to first propose a service system by describing the overall behaviour of each consumer, and then to instantiate such consumers retrieving services exposing a behavioural contract which is adequate to the matching given behaviour (Bravetti, 2009).

The work in (Sriharee, 2003) presents an approach based on ontology to improve descriptions of WS that are defined in WSDL with ontology-based behavioural information, the query for services are based on behavioural constraints and have a service ontology linked with each WS. It can benefit from inferring semantics of the service from the service ontology. This work neglects the consumer needs, it doesn't depend on the execution order of WS operations.

The aim of WS discovery based on behavioural aspect prove that WS consumer is the most important part in discovery process, whose role is to specify the WS description as well as its behaviour.

The authors in (Ramollari, 2008) propose an approach where the service provider enhances the WSDL document by means of a formal model of the WS behaviour, expressed in the stream X-machine formalism (SXM). This model is used by the service broker during publication and by the service consumer during discovery.

All these works treat behavioural aspects as conversational messages exchanged between WS and abandon the internal structure of WS execution. That is why the approach that we propose in this paper, involves the importance of sequencing of WS operations to guarantee the quality of service.

6 CONCLUSIONS

We hope you find the information in this template useful in the preparation of your submission.

Service consumers have a choice between different WS candidates that provide similar functions. Accordingly, comparing services requires more sophisticated patterns of discovery.

In this paper, we have proposed a WS discovery approach based on behavioral aspects. We sought to satisfy consumer needs by introducing new criteria based on user requirements. This approach is based on execution manner of WS operations. To fulfill the aim of our approach, we arranged consumer requirements in semantic and behavioral aspect. Hence, we have organized our work in three phases; phase1 consists to transform consumer sentences to keywords if the consumer is a non expert domain. Phase2 is a semantic WS discovery basing on WordNet as lexical database and then create the automaton to put in order WS operations and to extract the language accepted. Phase3 aims to extract sequences from WSDL/BPEL files of WS searched to select the relevant WS.

In future work, we plan implementing our approach by developing a query language that uses the underlying automaton to the required WS. A case study will be set up to explain the objectives of the approach. Also experimentation should be made to highlight the advantage of using the approach. A work is underway to fulfill this objective.

As for Selection phase, we can add other criteria to select the most appropriate service if many services satisfy the desired behavior.

REFERENCES

G. A. Miller, 1995. *WordNet*: A Lexical Database for English, *Communication of the ACM, Vol. 38, No. 11.*

F. Christiane, 2005. *WordNet and wordnets,* In: Brown, Keith et al. (eds.), Encyclopedia of Language and Linguistics, Second Edition. Oxford: Elsevier, pages 665-670.

J. Morato et al., 2004. WordNet Applications. *GWC 2004 Global Wordnet Conference (poster session). http://www.fi.muni.cz/gwc2004/. Brno.*

G. Łabiak and G. Borowik, 2010. Statechart-based Controllers Synthesis in FPGA Structures with Embedded Array Blocks. *International Journal of Electronics and Telecommunications. Volume 56, Issue 1, Pages 13–24, ISSN (Print) 0867-6747, DOI: 10.2478/v10177-010-0002-7.*

D. Harel, 1987. STATECHARTS: a visual formalism for complex systems*, Science of Computer Programming 8, 1987, pages 231-274.*

A. Bhattacharjee, B. S. Purkayastha, 2014. A Novel Approach to Construct Deterministic Finite State Automata. *International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 3 Issue 1, Page No. 3700-3703.*

J. Kari, 2013. *Automata and formal languages.* Fall semester 2013 University of Turku.

A. ALBRESHNE et al., 2009. Web Services Orchestration and Composition Case Study of Web services Composition, *WORKING PAPER.*

N. Milanvoic, and M. Miroslaw, 2004. Current Solutions for Web Service composition. *s.l.: IEEE Computer Society.*

OASIS, 2007. Web Services Business Process Execution Language (WSBPEL).

K. Kritikos and D. Plexousakis, 2007. OWL-Q for Semantic QoS-based Web Service Description and Discovery, *Proceedings of the 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference,.BEXCO, Bussan KOREA. pages 123-137.*

Elabd, 2011. Conformité de services Web par rapport à des spécifications de haut niveau.

Z. Maamar et al., 2009. A New Approach to Model Web Services Behaviors based on Synchronization. *International Conference on Advanced Information Networking and Applications Workshops.*

M. Bravetti, 2009. Foundational Aspects of Service Discovery based on Behavioral Contracts*, 3rd International Workshop on Verification and Evaluation of Computer and Communication Systems.*

N. Sriharee and T. Senivongse, 2003. Discovering Web Services Using Behavioral Constraints and Ontology. *Distributed Applications and Interoperable Systems - DAIS, pages. 248-259.*

E. Ramollari et al., 2008. Towards Reliable Web Service Discovery through Behavioral Verification and Validation. *Proceedings of the 3rd European Young Researchers Workshop on Service.*