C3Q: A Specification Model for Web Services Within Virtual Organizations

Mahdi Sargolzaei, Hamideh Afsarmanesh

C3Q: A Specification Model for Web Services within Virtual Organizations

Mahdi Sargolzaei and Hamideh Afsarmanesh

Springer-Verlag, Computer Science Editorial, Tiergartenstr. 17,
69121 Heidelberg, Germany
{M.sargolzaei1,H.afsarmanesh}@uva.nl

Abstract. Generic representation of web services is targeted, in order to generate machine readable specification of business processes that run at each partner organization within the virtual organization (VO). A holistic and formal service specification model is defined that can then be used unambiguously for discovery of business services, i.e. for finding a suitable service available at the VO to perform a given task. Especially, the proposed model, called C3Q, augments the description of service by its behavioral specification. A light extension of the WSDL documents is represented to specify all aspects of C3Q. Finally, a GUI is implemented to assist users with the behavioral description of the VO services.

Keywords: Service oriented architecture (SOA), Web services, Service specification.

1 Introduction

Fast pace in development of services prompts exploration of the role of Service Oriented Architecture (SOA) in assisting organizations to deal with service interoperability and flexibility demands. Using SOA and its available standards enable organizations to better connect their operations. In VOs, as a first step, the services should be specified concisely, such that they are recognizable, discoverable, comparable and integrable. Currently, web services are the most promising technology that implements the concept of SOA, and provides the basis for the development and execution of business processes that are distributed over the Internet. A web service is defined as a self-contained, modular, loosely coupled, reusable software application that can be described, published, discovered, and invoked over the World Wide Web [1]. In the last decades, the Web Service Definition Language (WSDL) has emerged as the most prominent standard for the specification of business services (BSs). This standard however does not provide the specification basis for a service client to get a full understanding of "What of the service does exactly" and "How the service performs". This lack of information about services results in the mismatch of provider's objective with consumer's demands, considering the functionality of the corresponding service. In spite of several proposed additional standards, a comprehensive view on which aspects of a service is required to be concisely specified is still missing [27]. In fact, in order to share

effectively the BSs and facilitate their reusability and integration in VOs, it is needed to define and register the BSs in a common VO directory. Despite the simple appearance of the above requirements, several complexities and challenges rise that need to be addressed, as mentioned below:

- No uniformity in service definitions, since VO partner organizations are and remain independent and autonomous.
- Lack of common ontology for defined services.
- No defined specification as needed for formal and concise representation of services.
- No address of service functionality, as required for developing and deploying software services.
- Lack of unambiguous machine interpretable (e.g. an XML-based standard) representation of services.

This research work aims to define a model to address and resolve these obstacles and challenges, and to provide a basis for discovery and composition of services in VOs discussed in [13], [1] and [23].

The paper is organized as follows. Section 2 briefly outlines some theoretical and technical aspects from the related works in order to serve as the base for our service specification. In Section 3, we sketch out our model of VO services, called C3Q. Moreover, a GUI is implemented to assist users with the behavioral description of the services, as demonstrated in this section. Section 4 represents how we extend the WSDL documents in order to describe all aspects of C3Q. Finally, we conclude the paper in the last section.

2 Related Work

Nowadays, web services have turned into a main area of research in the field of Service-Oriented Architecture [12], and has been widely accepted by service industries. One key point for the success of web services technology is the employment of XML-based standards, such as SOAP and WSDL for communicating and self-describing [8]. The Web Services Description Language (WSDL) as the most adopted standard for web Service description is limited to self-describing of the structure of the messages and operations, not the concept and the capability of the service. This limitation which is known as “lack of semantics” [9] in describing service capability, consequently required human intervention to interpret the semantics of the message content and the capability of the web service in order to ensure a valid and befitting use of the service. Apart from lack of semantics, another limitation of WSDL is that it does not address the configuration of stateful web services, so-called the behavior of the services. The behavior plays a vital role in service composition and improve service discovery as discussed in the [23]. Thus, lack of semantics and behavior is a major drawback of WSDL, and consequently become a barrier in achieving automatic or semi-automatic service discovery, composition and execution.

A numerous standards and languages have been proposed to describe semantics of web Services, such as OWL-S [22], WSMF [10], WSMO [16]. Due to the lack of

semantics definition in WSDL documents, many research efforts are being put into the extension of WSDL by semantics annotations, such as WSDL-S [2] and SAWSDL [15]. WSDL-S can annotate the information provided in WSDL using different semantic languages, such as RDF and OWL. SAWSDL is also de- fined as an extension of WSDL to describe the semantics of its elements through providing the mechanisms to bind ontology concepts to semantic annotations of WSDL.

Although several researchers have tackled "the lack of semantic" problem and some tools (e.g., [21]) have achieved good results by specification of syntactic and semantic properties of web services, they hardly consider the behavioral signature of a service, which describes the sequence of operations, which the user is actually interested in. This is partly due to the unavoidable limitations of today's standard specifications, e.g. WSDL, which do not encompass such aspect. Despite this, the behavior of stateful services representation is a very important issue to be considered during discovery and composition of services, to provide users with an additional means to refine the search and automate the composition in such a diverse environment. A few formalisms have been proposed that are able to model the behavior of a service. For example, session types as a formalism for structuring interactions and reasoning over communicating processes, can be applied as a model to describe the behavior of services. Session types, which can be assigned to end-point processes, describe the user view of an interaction. In [7], the authors have specified component behavior as session types showing that session types can also describe the behavioral signature of services.

Besides the functional description of services, it is essential to capture non-functional properties of BSs or quality of services (QoS) in order to meet the performance requirements of clients (such as availability) and even providers' requests (e.g. cost). Several research works in this area are instead in favor of extending the WSDL capabilities rather than introducing additional languages on top of it. The works done in [20] and [6] are two instances that capture the QoS specifications with WSDL file. A lightweight WSDL extension called Q-WSDL (QoS-enabled WSDL) is introduced in [6], to specify the QoS characteristics of web services.

3 C3Q Model of VO Services

For the sake of developing an architecture to support service oriented VOs, we first define a holistic service model, on top of which this architecture can be founded. Here, a new meta-data based on this mode is introduced to formalize the description of business processes. This model aims at addressing all characteristics of BSs, through an unambiguous formal description. From the service analysis point of view, all BSs intended to be shared and reused within the VO need to be unambiguously specified according to C3Q specified by their service providers.

We propose a concise representation of BSs as web services, C3Q model, namely addressing the Capability, Costs, Conspicuity, and Quality criteria of services. As such, VO web services can be uniformly defined and published in order to support

their sharing and reusing. The Capability is the most important part of the C3Q, which represents the functional properties of the BSs including syntax, semantics, and behavior. The other elements of C3Q model, including cost, conspicuity and quality criteria of services, contain the other important required description of non-functional properties of BSs. These three are also defined in the next subsections. Since different BSs may offer similar functionality with distinct non-functional characteristics, it is necessary to consider both functional and non-functional properties as the BS competency, in order to fully satisfy the demands of a service client, especially during the service selection phase [18] and [17].

We can apply a number of different notational options for representing each C3Q's aspect. We have however adopted one specific notation and specification way for formalizing each of these aspects, as it is later addressed in this section.

3.1 Syntax

The description of the syntactic properties of a service are usually represented by XML-based standards and languages, such as Web Service Description Language (WSDL), Universal Description, Discovery, and Integration (UDDI) and Simple Object Access Protocol (SOAP) [5]. A WSDL description is an XML document that contains the following information about a specific web service:

– **What** the service does, which is described in terms of the service's operations, as well as the input and output parameters that define the operation's messages.
– **How** the service is acceded, that describes data structures, binding implementation and protocols needed for sending messages through the web to reach the service location.
– **Where** the service is located, i.e. the hosting address that executes the service implementation.

3.2 Semantics

We refer to the conceptual description of web services as their semantics, which are typically defined with an ontology, i.e. an explicit specification of a conceptualization of knowledge related to services. The definition of service ontology in the VO context encompasses a group of vocabularies that specify semantic attributes of services (e.g. context) which together provide a meaningful concept about the service [3]. In fact, the semantic description of BSs would enrich the information about services to the level that cannot be specified by their mere syntactic description. Purpose-classification of the BS (e.g. goals and context) are good examples of the semantic aspects of the BS specification, aims to categorize services in order to improve service discovery and matchmaking.

The proposed service semantics within our C3Q-based service description consists of a set of semantic attributes. These semantic attributes provide a rich description of the Conceptual information needed for representing semantics related to services. For example, goal as a semantic attribute can describe the business logic of the service

(e.g. Monitoring). It is possible to also define a semantic attribute for the existing elements of the WSDL document, e.g. the operation's category.

In order to obtain semantic information about items used in the semantic attributes for the sake of semantic discovery, we must link to a particular reference domain ontology for that item. Such ontologies encompass a set of well-founded structured data that provide significant concepts with their semantic relation- ships, which can be used to improve the matchmaking of the services. In this research, we do not deal with the problems of ontology construction and matching. Rather, we assume the existence of a pre-defined domain-specific ontology or simply a taxonomy for a specific domain, which is related to a semantic at- tribute. Therefore, we capture the elements described below to specify a semantic attribute.

– **name**, which represents the title of attribute, e.g. goal or context.
– **taxonomyUri** that refers to the link of a related domain ontology or taxonomy for the attribute.
– **value** that represents the value of the attribute for this service.

An example of semantic specification in our model is represented in Figure 1.

```
<xwsdl:semantics>
    <attribute>
        <name> context </name>
        <taxonomyUri>
        http://example.org/onto\#context
        </taxonomyUri>
        <value> Purchase </value>
    </attribute>
    <attribute>
        <name> goal </name>
        <taxonomyUri>
        http://example.org/onto\#goal
        </taxonomyUri>
        <value> Monitoring </value>
    </attribute>
</xwsdl:semantics>
```

Fig. 1. Example of the WSDL extension by the semantic description.

3.3 Behavior

Beyond the semantic description of the operations that a service can provide, and the syntax of how they are to be invoked, a specification of the proper order in which those operations can be invoked is a prerequisite for correct implementation and use of a service. Behavioral specification of a service, refers to the specification of all admissible invocation orders of the operations of that service. The discovery of suitable services matching a query must consider the behavioral specification of candidate matches. What operations can be performed at a given point in time by a

client of a service may depend on the history of the previous operations that have already been performed (usually, by the same client) on that service. Therefore, specification of the behavior of a service is, in general, "stateful". However, these states are not always maintained within the service itself.

Consider a hotel booking example, as in Fig.2 illustrated, rooted in [14]. However, this service cannot be used properly unless, for instance, *getHotelDetails* operation is invoked only after a search operation. Any proper use of this service requires remembering whether or not a search has indeed been performed yet, and perhaps the results of such a search, etc. The REST architecture requires such information to be kept outside of the service implementation itself, on the client/user side (perhaps as cookies), and passed back and forth between the client and the service. From the perspective of a client, however, the stateless service in Fig. 2 cannot be used without considering the specification of its stateful behavior depicted in [23].

As a consequence, we extend the WSDL document to incorporate behavioral information of services. Our approach retains the original structure of the WSDL documents but enhances them by adding new tags. Figure 3 shows an example of the WSDL extension with the behavior description.

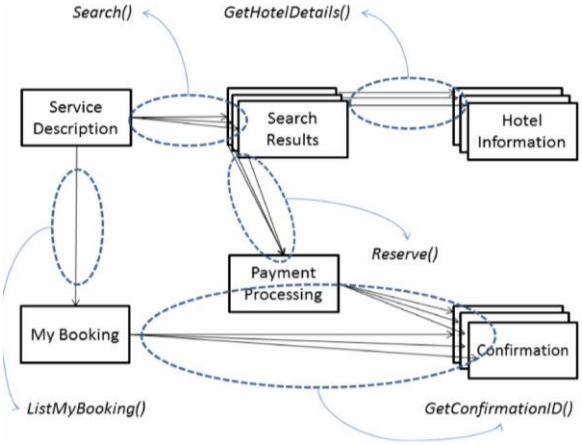

Fig. 2. Operations of the example: the Hotel booking service rooted in [14]

Since modelling the behavior of services in term of constraint automata (CA) seems rather difficult for the users, a GUI is developed to ease the behavioral specification of services and to allow its visualization. For this implementation, we have extended an open source java library, so-called Fizzim, which enables graphical modeling and designing of finite state machine (FSM). The GUI accepts a WSDL document as its input, then provides its behavioral description as a stateless web service. In fact, the preliminary behavioral-specification of the web service as a stateless web service consists of a single state automaton for each operation of the web service. For the stateful services, the states might be connected to each other to indicate the desired sequence of the operations' invocations during the service execution. Thus, a service integrator as a user of the GUI, should be able to add or remove transitions between the states of constraint automata. Fizzim supports our required extensions. It also exports the revised CA in the form of WSDL document, which is extended by the

needed tags to model the behavioral specification of the service, according to the XWSDL.

3.4 Quality Criteria of Services

Quality criteria of service consists of a numbers of properties, in which every property has its own effect on overall quality of service (QoS). A wealth of research work has been done to support QoS for BSs. Although so many QoS solutions are proposed, service developers and clients still are not able to handle QoS-related concerns easily. The reason lies in the fact that a universal QoS specification standard is still absent.

We may consider that a QoS specification should contain the elements de- scribed below, in order to assure an expressive formal description of quality of services.

```
<xwsdl:behaviour>
    <transition>
        <name> Search </name>
        <currentState> q0 </currentState>
        <nextState>  q1 </nextState>
    </transition>
    <transition>
        <name> getHotelDetails </name>
        <currentState> q1 </currentState>
        <nextState>  q2 </nextState>
    </transition>
    <transition>
        <name> reserve </name>
        <currentState> q1 </currentState>
        <nextState>  q3 </nextState>
    </transition>
    <transition>
        <name> getConfirmationId </name>
        <currentState> q0 </currentState>
        <nextState>  q5 </nextState>
    </transition>
    <transition>
        <name> getConfirmationId </name>
        <currentState> q3 </currentState>
        <nextState>  q4 </nextState>
    </transition>
</xwsdl:behaviour>
```

Fig. 3. Example of the WSDL extension by the behavior description.

- **Criterion** which represents a quantifiable aspect of a service like availability. Unit that is used as a standard for counting or measuring the corresponding quality criterion, e.g. hours per day, hours per week, decimal, etc. for the avail- ability.

- **Range (Min and Max)** which depicts the highest and lowest possible values of the quality criteria. The range is needed when we want to compare the same quality criteria with different units.
- **Value** that represents the amount of the corresponding quality criterion.

An example of QoS specification in XWSDL is represented in Figure 3. The <QualityCriteria> tag is a container tag, which contains at least one <Criterion> tag. The <Criterion> tag is also a container tag indicating the required attributes of the corresponding criterion including "Name" "Unit" and "Value". The <Value> tag also provides the minimum and maximum values of the criterion, i.e. the "min" and "max" attributes.

3.5 Cost

Cost is a key economic attribute that affects the selection and usage of BSs. Thus, we introduce cost as an additional QoS parameter for specification of BSs. The cost specification consists of three parts:

- **Initial price** which represent the value of the cost, e.g. 5 or 10.
- **Unit** that defines the unit of the cost, e.g. Dollar or Euro.
- **Price plan** which is used to model the method of cost estimating, e.g. per invocation, per transmitted byte charges, etc.

In Figure 4, an example of cost description in our proposed specification, i.e. XWSDL is represented.

```
<xwsdl:Cost>
    <InitialPrice> 10   </InitialPrice>
    <Unit> Dollar </Unit>
    <PricePlan> Per invocation  </PricePlan>
</xwsdl:Cost>
```

Fig. 4. Example of the WSDL extension by the cost.

3.6 Conspicuity

Conspicuity for a BS is the quality or state of being well-known and marked by a noticeable violation of good taste from the service client prospective. It represents means for identifying the validity of information related to a service, as claimed by its provider. A web service's conspicuity can either be measured through studying the behavior of the corresponding service provider or by capturing the past service consumers' feedbacks.

We have used the VO Supervisory Assessment Tool (VOSAT) [25] to assess the conspicuity. The approach adopted for VOSAT borrows ideas from [26], that monitors the behavior of VO members for identifying their level of trustworthiness. In this approach, all agreements in Operational Level Agreement (OLA) and Service Level Agreement (SLA) are considered as promises among the involved partners in the VO. The trustworthiness of each VO partner is reflected in this framework, as calculated by the VO Supervision Tool during the VO operation phase, related to the

claims made by each partner, as explained in [26] and [19]. In [24], different introduced states for promises include: conditional, unconditional, kept, not kept, withdrawn, released, and invalidated, which address different stages within the entire life-cycle of every promise. The life-cycle of every promise is then formalized and monitored, and the trust level of VO partners are assessed through a set of pre-defined causal- relationships among different promise states and the trustworthiness of VO members. Therefore, at any point in time, the trust level of the VO member would be reflected on its claims about different characteristics of its provided services, as well as on its feedback about others' services. The trust level of each partner is calculated in reference to its own performance in the VO by VOSAT during the VO operation phase. This information, i.e. the trust level is used as the conspicuity specification in the C3Q service competency model defined in this research. An example of conspicuity specification in the proposed XWSDL is shown below. Please note that the value of trust level would be between 0 and 1. An example of conspicuity description in our proposed specification is represented in Figure 5.

```
<xwsdl:Conspicuity>
    <Trustlevel> 0.6   </Trustlevel>
</xwsdl:Conspicuity>
```

Fig. 5. Example of the WSDL extension by the conspicuity.

The technical details of measuring the organizations' trustworthiness is reported in [25].

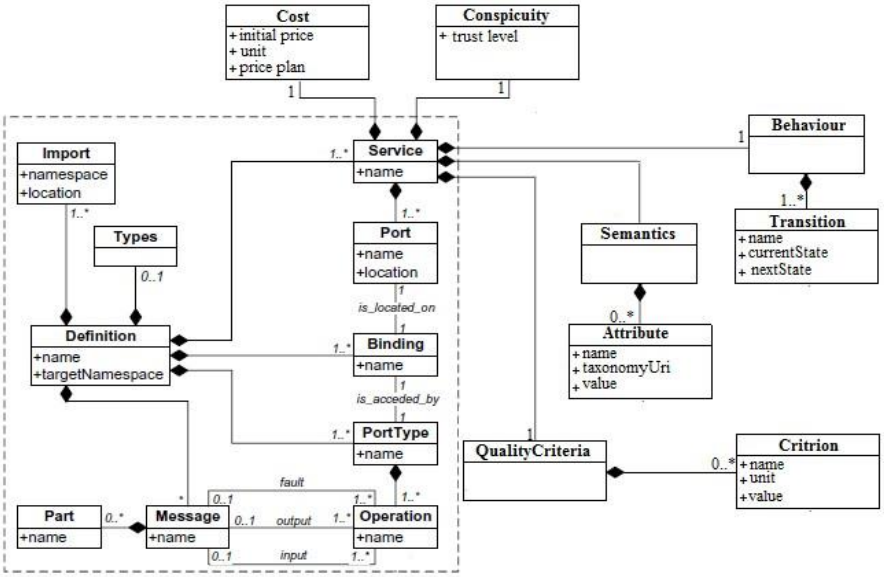

Fig. 6. The XSD tags of the schema

4 XWSDL

We extend WSDL description of web services in order to support the C3Q. The XWSDL extension follows the rules for extending WSDL [4] to guarantee that any service consumer unaware of the extensions can still parse, validate and use the extended version of WSDL files, i.e., XWSDL documents. A new namespace "XWSDL" should be used to identify the tags part of the extension.

Our approach retains the original structure of the WSDL documents and enhances it by new tags to the XML-based files. In order to extend WSDL, we need to define a schema of the elements of XWSDL as its name space. We used XML-Liquid 2.0 to design the schema and then translate it to an XML documents, which consist of XSD tags in order to define the elements of the schema. This XML document is used later as the namespace for defining XWSDL documents.

The Object Management Group (OMG) has proposed "Model Driven Architecture" to support existing and future OMG standards and object models, so they would become assets instead of expenses during the transforming technologies. Model Driven Architecture focuses on providing meta-model, which is simply a model of a modeling language. These kind of meta-models are defined by use of the Meta Object Facility (MOF1), which is the OMG's standard to specify meta-models aimed at describing other model. Nowadays, employing MDA in web service standards has received significant attention to assist the automated generation and extension of web service models [11] and [6]. Therefore, we apply MDA to our definition of XWSDL, in order to appropriately enrich web service descriptions based on the C3Q.

Representing an XML-based language in terms of a meta-model allows to enhance its comprehensibility and facilitate its extension [6]. Figure 6 introduces the XWSDL meta-model, as an extension of WSDL meta- model, from which the XWSDL XML Schema is derived. The basic WSDL meta-model is represented in the portion of Figure 6 bounded by a dashed line shape. Note that some classes related to specific documenting and extensibility features of XML are removed from this basic WSDL meta-model in favor of brevity and readability. The other classes and associations outside the dashed line shape in Figure 6 indicate our extension of the WSDL meta-model to include the description7 of C3Q for a web service. In other word, the whole classes and association in represented Figure 6, i.e. both inside and outside the dashed line shape, forms the XWSDL meta- model. Obviously, multiplicities 0..1 or 0..* indicate optional associations, while associations with multiplicities 1 or 1..* reveal needed associations. This means, for example, that the transition is obligated for the Behavior class, while the Attribute is optional for the semantics (see Figure 6). Note that the introduced meta-model of XWSDL can assist service providers in order to transform their WSDL documents to the proposed XWSDL descriptions.

5 Conclusion

In this paper, we presented an extension and improvement to the current web service description approaches and standards, in order to support more efficient service discovery and composition in VOs. First, we depict a data model namely C3Q to represent the various information needed for description of the BSs as web services. The C3Q is considered as the service's competency model within the VOs. Then, we introduce a light extension of WSDL that we have called XWSDL, to specify web services according to the C3Q model. XWSDL is the first model that provides a comprehensive description of capabilities over web services and highlights the important role of service behavior in the realization of the semi-automated service oriented computing. Since XWSDL is a lightweight extension of standard WSDL, the existing WSDL documents can easily be enriched without altering their original content. The meta-model of XWSDL is also presented here to assist transforming WSDL documents to the proposed XWSDL descriptions. In XWSDL, the power and flexibility of the C3Q model has been combined with the simplicity and convenience of standard WSDL, thus reaching the right balance between flexibility and expressivity for VO services.

References

1. Afsarmanesh, H., Sargolzaei, M., Shadi, M.: Semi-automated software service integration in virtual organisations. Enterprise Information Systems 9(5-6), 528–555 (2015)
2. Akkiraju, R., Farrell, J., Miller, J.A., Nagarajan, M., Sheth, A.P., Verma, K.: Web service semantics- wsdl-s (2005)
3. Camarinha-Matos, L.M., Afsarmanesh, H., Oliveira, A.I., Ferrada, F.: Collaborative business services provision. In: Proceedings of *ICEIS'13 – 15th International Conference on Enterprise Information Systems,* Angers, France, 4-7 Jul 2013, vol. 2, pp 382-392 (2013)
4. Chinnici, R., Moreau, J.J., Ryman, A., Weerawarana, S.: Web services description language (wsdl) version 2.0 part 1: Core language. W3C recommendation 26, 19 (2007)
5. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Un- raveling the web services web: an introduction to soap, wsdl, and uddi. IEEE Internet computing 6(2), 86–93 (2002)
6. D'Ambrogio, A.: A model-driven wsdl extension for describing the qos of web services. In: Web Services, 2006. ICWS'06. International Conference on. pp. 789–796. IEEE (2006)
7. Dezani-Ciancaglini, M., Padovani, L., Pantovic, J.: Session type isomorphisms. In: PLACES. pp. 61–71 (2014)
8. Dhara, K.M., Dharmala, M., Sharma, C.K.: A survey paper on service oriented architecture approach and modern web services (2015)
9. Du, X.: Semantic service description framework for efficient service discovery and composition. Ph.D. thesis, Durham University (2009)
10. Fensel, D., Bussler, C.: The web service modeling framework wsmf. Electronic Commerce Research and Applications 1(2), 113–137 (2002)
11. Frankel, D., Parodi, J.: Using model-driven architecture to develop web services. IONA Technologies white paper (2002)

12. Huhns, M., Singh, M.: Service-oriented computing: Key concepts and principles. IEEE Internet Computing 9(1), 75–81 (2005)
13. Jongmans, S.S.T., Santini, F., Sargolzaei, M., Arbab, F., Afsarmanesh, H.: Orchestrating web services using reo: from circuits and behaviors to automatically generated code. Service Oriented Computing and Applications 8(4), 277–297 (2014)
14. Kopecky, J., Gomadam, K., Vitvar, T.: hrests: An html microformat for describing restful web services. In: Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT'08. IEEE/WIC/ACM International Conference on. vol. 1, pp. 619–625. IEEE (2008)
15. Kopecky`, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for wsdl and xml schema. IEEE Internet Computing 11(6) (2007)
16. Lara, R., Roman, D., Polleres, A., Fensel, D.: A conceptual comparison of wsmo and owl-s. In: Web services, pp. 254–269. Springer (2004)
17. Ludwig, H.: Web services qos: external slas and internal policies or: how do we deliver what we promise? In: Web Information Systems Engineering Workshops, 2003. Proceedings. Fourth International Conference on. pp. 115–120. IEEE (2003)
18. Menasce, D.: QoS issues in web services. Internet Computing, IEEE 6(6), 72–75 (2002)
19. Msanjila, S., Afsarmanesh, H.: Trust analysis and assessment in virtual organization breeding environments. International Journal of Production Research 46(5), 1253–1295 (2008)
20. Pei, S., Chen, D.: Research on dynamic web services composition framework based on quality of service. Information Technology Journal 10(8), 1645–1649 (2011)
21. Plebani, P., Pernici, B.: Urbe: Web service retrieval based on similarity evaluation. IEEE Trans. on Knowledge and Data Eng. 21(11), 1629–1642 (2009), http://dx.doi.org/10.1109/TKDE.2009.35
22. Rohallah, B., Ramdane, M., Zaidi, S.: Agents and owl-s based semantic web service discovery with user preference support. arXiv preprint arXiv:1306.1478 (2013)
23. Sargolzaei, M., Santini, F., Arbab, F., Afsarmanesh, H.: A tool for behaviour-based discovery of approximately matching web services. In: International Conference on Software Engineering and Formal Methods. pp. 152–166. Springer (2013)
24. Shadi, M., Afsarmanesh, H.: Behavior modeling in virtual organizations. In: Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on. pp. 50–55. IEEE (2013)
25. Shadi, M., Afsarmanesh, H.: Behavioral norms in virtual organizations. In: Working Conference on Virtual Enterprises. pp. 48–59. Springer (2014)
26. Shadi, M., Afsarmanesh, H., Dastani, M.: Agent behavior monitoring in virtual organizations. In: Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on. pp. 9–14. IEEE (2013)
27. Terlouw, L.I., Albani, A.: An enterprise ontology-based approach to service specification. IEEE Transactions on Services Computing 6(1), 89–101 (2013)