A Semantic End-to-End QoS Model for Dynamic Service Oriented Environments

Nebil Ben Mabrouk, Nikolaos Georgantas, Valérie Issarny

A Semantic End-to-End QoS Model for Dynamic Service Oriented Environments

Nebil Ben Mabrouk, Nikolaos Georgantas, Valérie Issarny
INRIA Paris-Rocquencourt, France

{nebil.benmabrouk,nikolaos.georgantas,valerie.issarny}@inria.fr

Abstract

In Service Oriented Computing (SOC), modeling the Quality of Service (QoS) is a cornerstone for providing services with quality guarantees. As the technological advances and wide adoption of handheld devices (e.g., PDA and smartphones) and wireless networks (e.g., UMTS, WiFi and Bluetooth) have made service environments more dynamic, QoS models must change accordingly.

In this paper, we present a QoS model that provides the approporiate ground for QoS engineering in SOC. Our model focuses on emerging QoS features related to the dynamics of service environments such as user mobility and context awareness of application services. It also considers QoS on an end-to-end basis by covering QoS features of all the resources and actors involved in service provisioning (e.g., network, device, service, end-user). Our model represents QoS with rich semantic information and makes use of the Web Service Quality Model (WSQM) proposed by OASIS.

1 Introduction

In Service Oriented Computing (SOC), providing services with quality guarantee capabilities requires having solutions that include Quality of Service (QoS) models as a key feature. Indeed, QoS models provide the approporiate ground for QoS provision in service oriented environments.

In such environments, QoS can be influenced by several factors including the hardware and network infrastructure (capabilities of computing devices, network connectivity), the quality level provided by application services and the end-user characteristics (mobility, generated traffic). This implies that, in order to obtain an accurate evaluation of QoS, none of these aspects should in principle be neglected at the QoS modeling stage. Thus, QoS should be considered on an end-to-end basis, meaning that QoS models should cover quality features of all the resources and actors involved in fulfilling a service. In practice, these include (i) the service environment and its underlying hardware and network infrastructure, (ii) application services and (iii) end-users.

Furthermore, QoS models should consider emerging features related to the dynamics of service environments. Indeed, the large success of mobile devices and wireless networks have made service environments more dynamic, thus QoS models must change accordingly. For instance, emerging features such as the adaptability and context awareness of application services should be considered as QoS properties affecting the quality level required by users. A third issue concerns the expressivity of QoS models. Related to this, a promising approach addressing QoS modeling relies on the Semantic Web technologies, notably ontologies. Ontologies allow for capturing and defining QoS knowledge with rich semantic information. The importance of such representation is two-fold: First, it is a machine-understandable specification for QoS that provides the appropriate ground for several service engineering capabilities such as logical reasoning on QoS and automation of QoS management in dynamic service environments. Second, as they represent a common and shared knowledge of QoS, ontologies allow for broadening QoS understanding across different specifications that address QoS for varied purposes, thus enabling to cope with the syntactic heterogeneity of QoS specifications. Ontologies are often computationally expensive, nevertheless they are increasingly used in the context of dynamic SOC [11].

Despite the considerable amount of research devoted to QoS ontologies, none of them considers jointly QoS on an end-to-end basis and emerging QoS features related to the dynamics of service environments. On one hand, we have ontologies tailored to a specific aspect of QoS centered on the service level [8, 6, 2]. These ontologies do not consider other important resources involved in services' provision such as the network and hardware infrastructure. On the other hand, we have 'general' QoS ontology languages [14, 3] that specify basic concepts for QoS and allow defining additional concepts. These languages are not representative QoS models since they do not give any tangible specification of QoS. To the best of our knowledge, there are no QoS ontologies supporting the dynamics of service environments.

This paper introduces a semantic QoS model that copes with the above issues, i.e., it considers QoS on end-to-end basis and puts a special emphasis on emerging QoS features related to the dynamic nature of service environments. Our model comprehends four ontologies that specify: (1) Basic concepts needed for QoS description, (2) QoS properties related to the service environment and its underlying hardware and network infrastructure, (3) QoS properties of application services and (4) QoS at the end-user level.

For ontologies (1) and (3) we recall QoS properties defined by the Web Service Quality Model (WSQM) [10] which is a prominent standardization effort proposed by the OASIS WSQM technical committee for the specification of Web Services' QoS. It defines a well-founded taxonomy of QoS and provides a wide range of QoS properties. WSQM employs a number of terms that have specific meanings to QoS. In this paper, we adopt this terminology and we list it below with explicit definitions. We use the term *quality factor* to refer to an attribute used to represent and assess QoS, the term *quality group* to refer to a group of quality factors concerning a common aspect of QoS, and the term *quality item* to refer to an atomic attribute within a quality factor (e.g., the response time quality item fall under the performance quality factor, the confidentiality quality item fall under the security quality factor).

The remainder of this paper is structured as follows. In Section 2, we give an overview of WSQM and we assess it with respect to our goals. In Section 3, we give the details of our model and its underlying ontologies, then we outline its usage for dynamic service computing. In Section 4, we evaluate related QoS models within different research communities. Finally in Section 5, we conclude with a summary of our contributions and the perspectives of this work.

2 WSQM Overview

WSQM (Web Service Quality Model)[10] is a conceptual model for Web Services quality that defines three basic concepts: *quality associates, quality activities* and *quality factors*. A quality associate refers to the person or the organization that is directly or indirectly managing QoS. The set of actions performed by quality associates throughout the whole services' lifecycle are called quality activities, whereas the quality factors are the attributes used to represent and assess QoS.

Our primary focus is on the quality factors which represent the core entities of QoS. Indeed, QoS models are usually centered on the definition and the classification of quality factors. WSQM divides quality factors into three quality groups with respect to system architecture, operation and business perspective:

- **System Information** is a set of static information on systemic functions that are defined and evaluated before using the service. It includes the *Security, Suitability for Standards, Manageability* and *Business Process* quality factors.
- **Service Measurement** refers to quality factors measured while a consumer is using the service. It is mainly about the *Performance* and *Stability* quality factors.
- **Business Value** assesses the service from a business point of view. It includes the *Service Cost, Service Suitability, Service Aftereffect* and *Service Brand Value* quality factors.

WSQM formally specifies the quality factors and their associated concepts using the Web Service Quality Description Language (WSQDL). WSQDL provides a detailed QoS specification, however it represents some shortcomings with respect to dynamic service oriented computing. In what follows, we give the details of WSQDL then we assess it regarding our goals.

2.1 WSQDL

WSQDL provides a XML-based description method for standardizing the expression of QoS exchanged between services and their consumers. It defines the constructs needed to specify the quality factors, their taxonomy, the way they are assessed and the relationships between them.

The main construct within WSQDL is *QualityFactor*. It is represented with a global structure that expands its inherent quality items into four levels reflecting the complexity of quality factors and allowing for the specification of fine-granularity details of QoS. These levels are: *Subfactor, Property, SubProperty* and *Function*. For instance, the Security quality factor is divided into three properties: confidentiality, integrity and non-repudiation. Confidentiality may include other sub-properties such as message level confidentiality and user confidentiality. By turns, the message level confidentiality can be defined using the XML-encryption function.

In addition, QualityFactor divides into four disjoint subclasses with respect to the way it is assessed: *MeasurementFactor, EvaluationFactor, BusinessValueFactor* and *BusinessProcessFactor*.

MeasurementFactor (e.g., Performance) defines quality factors that can be quantitatively measured using metrics. The metrics are defined with the *MetricType* construct which is given in terms of *MeasurementFunction, MeasureDirection, EnvironmentVariables* and *Metric*. The Metric construct is defined over the *MetricValue* and *Unit* constructs.

EvaluationFactor describes the type of appraisable quality factors determined by their conformity to an appraisal standard. The main construct defining this factor is *Conformity* which describes the degree of conformance between a quality factor and a standard specification. The Security, Manageability and Interoperability fall under this type of factors.

BusinessValueFactor addresses quality factors used to determine business aspects of QoS. In contrast to Evalua-

tionFactor, it has no standard evaluation and it may be assessed by users who estimate the business value of a quality level. Therefore, BusinessValueFactor is defined using the *UserAppraisal* construct.

Finally, BusinessProcessFactor defines the set of quality factors assessed by their ability to achieve business goals. These factors address features such as reliability of message transmission, transaction process and collaborability.

Complementary to the quality factors, which are centered on describing the individual aspects of QoS, the *Quality Chain* concept introduced by WSQDL gives a global view of QoS. It states that the quality factors are not totally independent and they may influence each other, hence impact the overall QoS. To this extent, Quality Chain defines a configuration of dependencies describing the correlation between quality factors. This can be used, for instance, to reflect the relationship between BusinessValue factors and other quality factors, which allows for specifying trade-offs that service providers can make in the quality levels they offer.

2.2 Assessing WSQM for dynamic service computing

Although it provides a well-founded specification of QoS at the service level, WSQM presents three main shortcomings regarding our goals: First, it neglects QoS associated to other resources and actors participating in service provisioning (e.g., network, devices and end-users). Second, it does not consider emerging QoS factors related to the dynamics of service environments (e.g., adaptability, context-awareness). Third, as already explained WSQM adopts an XML-based approach to specify QoS, which makes it subject to the syntactic QoS specification problem.

Regarding the above issues, our approach aims to define a semantic QoS model that addresses all elements of service environments and takes into account the dynamic nature of these elements. Seeing the importance of WSQM, our model makes use of this standard specification to define QoS at the service level.

The proposed model is defined over a set of ontologies underpinned by the Web Ontology Language (OWL), which is a W3C recommendation designed for publishing and sharing ontologies. The next section details the developped ontologies and outlines their usage in the context of SOC.

3 A Semantic End-to-End QoS Model for Dynamic Service Environments

We present a semantic QoS model that addresses QoS on an end-to-end basis by covering quality factors of the main elements acting in dynamic service environments, in particular it deals with: (i) network and hardware resources, (ii) application services and (iii) end-users. Our model puts a special emphasis on QoS features related to the dynamic nature of these resources such as end-user mobility and adaptability and context awareness of application services.

Our model is designed according to a layered approach integrating certain modularity and flexibility requisites, thus aiming to provide distinct and easily manageable ontologies. As depicted in Fig. 1, it comprehends four ontologies:

1. ***The QoS Core*** ontology incorporates general constructs needed for QoS description (e.g, quality group and quality factor). The conceptual elements of this ontology are inferred from WSQDL.
2. ***The Infrastructure QoS*** ontology specifies quality factors related to the environment and its underlying network and hardware infrastructure. It focuses on the capabilities of mobile devices, the connectivity of wireless networks and the characteristics of the environment where users and services behave.
3. ***The Service QoS*** ontology specifies quality factors of application services. It extends the factors already defined by WSQM with other factors (e.g., adaptation and context awareness) supporting the dynamics of the application services. The added factors are carefully selected by examining the dynamic nature of service environments. Additionally, this ontology is extensible in that new quality factors (e.g., domain-specific quality factors) can be easily added.
4. ***The User QoS*** ontology addresses user concerns about QoS. The role of this ontology is two-fold: First, it provides the constructs needed to specify user QoS requirements. Second, it specifies quality factors associated to the user such as user mobility.

Ontologies 2), 3) and 4) specialize the general concepts defined in the QoS Core ontology; they are layered (Fig. 1) from lower level (i.e., infrastructure) to higher level (i.e., end-user) aspects.

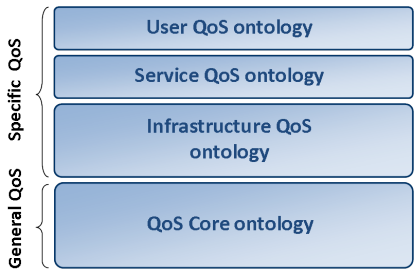

Figure 1. QoS model Overview

3.1 QoS Core ontology

In order to define base concepts required for QoS description, the QoS Core ontology makes use of the WSQDL standard schema, which, as discussed in Section 2.1, provides ample and detailed specification for quality factors, their taxonomy, the way they are assessed and the relationship between them.

The QoS Core ontology (Fig. 2) represents constructs from the schema with semantic information. The main constructs of this ontology are *QualityGroup, QualityFactor* and *QualityChain*. QualityGroup consists of one or more QualityFactor, which are represented from two key standpoints: their structure (i.e., *SubFactor, Property, SubProperty, Function*) and their type (i.e., *MeasurementFactor, EvaluationFactor, BusinessProcessFactor, BusinessValueFactor*). The QualityChain construct is defined over the *hasInfluenceOn* property indicating that two or more quality factors are in a quality chain relationship.

The QoS core ontology comprehends further constructs not represented in Fig. 2 such as *MetricType, Conformity* and their associated concepts. In this ontology, it is worth mentioning that the construct *Unit* within MetricType references other external ontologies to define its semantics. Indeed, the specification of measurement units is not addressed by WSQDL and it is not in the scope of the present work. Currently, Unit references the OWL time ontology [4].

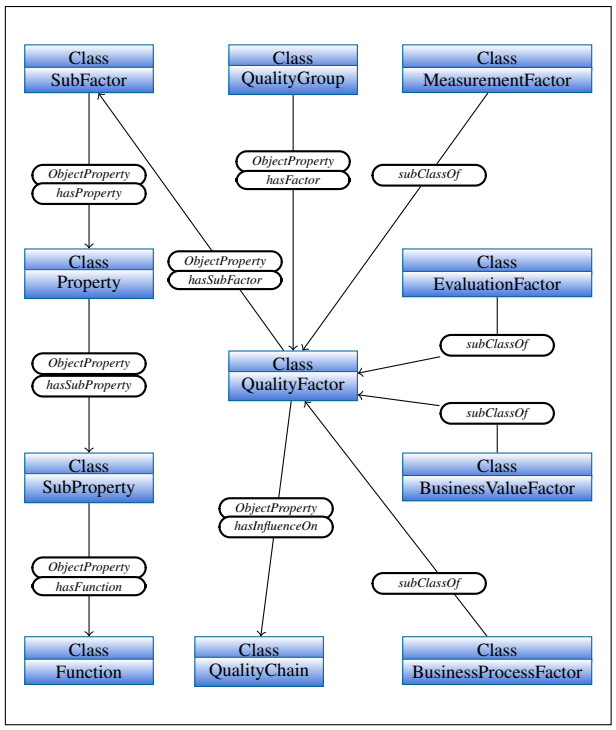

Figure 2. Overview the QoS Core ontology

3.2 Infrastructure QoS ontology

As already explained, QoS is difficult to define and to evaluate in an accurate manner without considering the infrastructure supporting the provision of services, namely the network and the devices enabling services and users to interact, as well as to the characteristics of the environment where sevices and users act. For this matter, we developped the Infrustructure QoS ontology (Fig. 3) which consists of three quality groups: *Network, Device* and *Environment*.

The Network quality group addresses the communication infrastructure in dynamic service environments. It consists of two quality factors: *Type* and *Performance*. The first factor indicates the network topology, which can be either *Wired* or *Wireless*. The second factor comprehends quality items specifying the performance of a network. Common performance metrics include *Bandwidth, Latency, Loss* and *Jitter* [7]. Bandwidth refers to the rate of data transfer; Latency refers to the total time taken to deliver a message; Loss represents the rate of message units lost during the delivery of a message. Finally, Jitter expresses the variation in Latency.

The Device quality group addresses hardware devices hosting application services (e.g., Server) or supporting end-users (e.g., PDA, SmartPhone, PC). It defines three quality factors outlining the capacity of devices: *Category*, *Mobility* and *Performance*. The first factor refers to the role of devices in service provision, which divides into two types: *Client* and *Server* [14]. The second factor describes the mobility of devices which can be either *Stationary* or *Mobile*. The third factor comprehends quality items specifying common device capabilities, i.e., *CPU, Memory, Storage Capacity* and *Power Consumption*.

The Environment quality group specifies quality factors intrinsically related to the environment where users and services exist. Such environments are populated with networked services supporting a wide variety of user applications. The quality of these environments can be assessed by evaluating their degree of support to user applications [5]. This can be specified using three quality factors: Service Density, Sustainability and Scalability [5]. Service Density refers to the number of services available in service environments. It indicates in part the ability of an environment to satisfy user requests, i.e., the higher the service density is, the higher is the probability to fulfill the users' tasks. Sustainability measures the environment's ability to sustain employed services if they fail. When a service part of the environment fails due to issues such as power limitations or mobility, the environment may set off some fault tolerance mechanism like identifying an alternative service which matches the original functionality. Scalability refers to the ability of the environment to support a large number of active users and to handle their requests in a satisfying manner.

3.3 Service QoS ontology

The Service QoS ontology is of fundamental importance in our model, since, in practice, the main QoS features are determined by application services. The key idea underlying this ontology is: (i) to recall WSQM standard quality factors for defining common QoS features of ap-

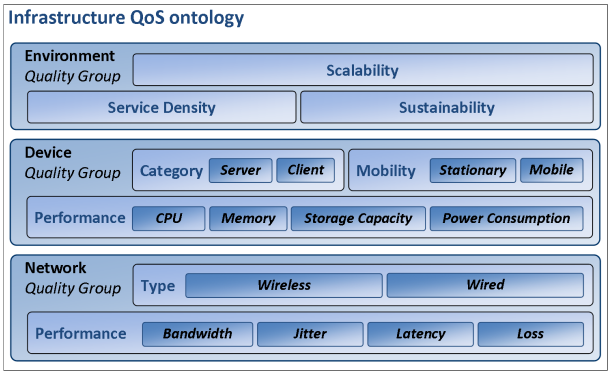

Figure 3. Overview the Infrastructure QoS ontology

plication services and (ii) to specify other quality factors to address the dynamic nature of services and the domain-specific QoS.

As already mentioned, WSQM defines three quality groups (i.e., System Information, Service Measurement and Business Value). In addition to these groups, we define a quality group called *Dynamic Capabilities* which defines quality factors supporting the dynamics of service environments. The responsibility of application services with respect to the dynamics of service environments includes supporting adaptation, context awareness and facilitating the automation of user request processing.

- **Adaptation**: Services operating in highly dynamic environments need to continuously adapt themselves in order to react to changing conditions such as environmental conditions and user requirements [12]. To this regard, we define the *Adaptation* quality factor which addresses the ability of a service to adapt itself to changing conditions and to reconfigure itself accordingly.
- **Context-awareness**: Dynamic service computing focuses on fulfilling user tasks in a transparent way that places few demands on user attention. Invisibility of services can be accomplished in part by reducing input from users and replacing it with knowledge of context. Context awareness is further required to enable adaptation to changing conditions by exploiting context information, such as location and proximity to other devices and services. To address context awareness of application services, we define the *Context* quality factor, which describes the ability of a service to gather, manage, use and disseminate context information.
- **Automation Support**: Dynamic service environments focuses on fulfilling user tasks on-the-fly (i.e., at runtime) by dynamically locating and integrating available services. Hence, services' discovery, selection and composition in such environments need to be performed automatically, i.e., with little or no human intervention. Such assumption requires in part the description of services' semantics using a machine-understandable specification. To this regard, we define a quality factor named *Automation Support*, which describes the ability of a service to support automated management. It consists of a quality item named *Semantic Information Offerability* denoting the ability of a service to provide semantic description of its functional and non-functional features.

Besides the Dynamic Capabilities quality group, we define the *Domain-Specific* quality group to address quality factors associated to services' domains. Indeed, service oriented computing deals with open environments offering access to a wide variety of services from different domains. Further, users often need to address QoS factors related to particular domains. For instance, if we consider a multimedia application which diffuses entertainment videos, it will be pretty useful to users to know the "*encoding quality*"of the given videos in order to choose the appropriate media player or to select the right screen resolution. The Domain-specific quality group addresses such issues by allowing the definition of any domain-specific quality factor as long as the latter references concepts within appropriate domain ontologies to specify its semantics.

The main conceptual elements of the Service QoS ontology are depicted in Fig. 4. In this figure, the layering goes from lower level aspects (i.e., system) to higher level aspects (i.e., domain-specific). The white boxes within the figure represent the quality factors already defined by WSQM, whereas the colored boxes represent the new factors defined for dynamic and domain-specific application services.

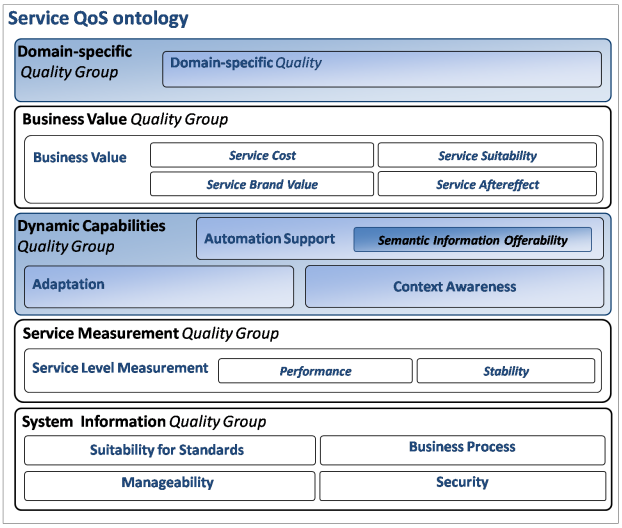

Figure 4. Overview the Service QoS ontology

3.4 User QoS ontology

This ontology addresses users' concerns about QoS. It is composed of two main components (Fig. 5): (i) user requirements modeling and (ii) user profile modeling.

Requirements modeling allows users to express their QoS requirements according to their own quality experi-

ence [16]. These requirements are formulated as a group of constraints defining the QoS level required by users. At the heart of requirements modeling, we distinguish the construct *Requirement*, which corresponds to the description of a constraint made by the user on the offered QoS. The constraint targets a *QualityFactor* from the QoS Core ontology, and it is given in terms of *Operator*, *Value* and *Unit*. The latter constructs change according to the type of quality factors. Requirements dealing with *EvaluationFactor, BusinessValueFactor* and *BusinessProcessFactor* (see Section 2.1) are expressed using boolean operators (i.e., *is-a, is-not-a*) and string values, whereas requirements addressing *MeasurementFactor* are given in terms of comparison operators (i.e., *equal, not-equal, more-than, less-than, max-value-of, min-value-of*), numerical values and measurement units. Users can further define composite requirements using *and* and *or* operators. Moreover, they are allowed to express their relative preferences about QoS requirements by assigning a certain *Weight* to every requirement.

Let us now consider the user profile modeling which aims at defining user-related quality factors (e.g., user mobility, user generated traffic). User *Profile* comprehends two quality factors: *Mobility* and *Traffic*. These factors have been defined based on the model proposed by Resta et al. [15]. User Mobility is divided into three patterns: Stationary users, QoS-driven users and Mobile users. The first pattern concerns really stationary users or users with constrained movements that does not affect service provisioning. QoS-driven users are mostly stationary but they move when their perceived QoS level drops below an acceptable threshold. Finally, the mobile users are characterized by continuously moving positions.

Concerning User Traffic, users are divided into three classes of load according to their generated traffic: low, medium, and high load. The lowest class of traffic accounts for users who are using the network for e-mailing and light Web browsing. The medium class of traffic accounts for users who are using the network for intensive Web browsing, file downloading, audio streaming, and so on. Finally, the highest class of traffic accounts for users who make an intensive use of the network, such as video streaming. User Mobility and Traffic are extensible in that specific patterns of mobility and traffic can be easily defined.

3.5 Usage of our model

Our model concentrates on QoS knowledge representation rather than a language to specify QoS. To this extent, our approach has been to decouple QoS knowledge from QoS specification by providing separate and reusable ontologies. Thus any appropriate QoS specification language can be used on top of our QoS model. More specifically, to employ our model for QoS description, the introduced ontologies have to be referenced by syntactic QoS

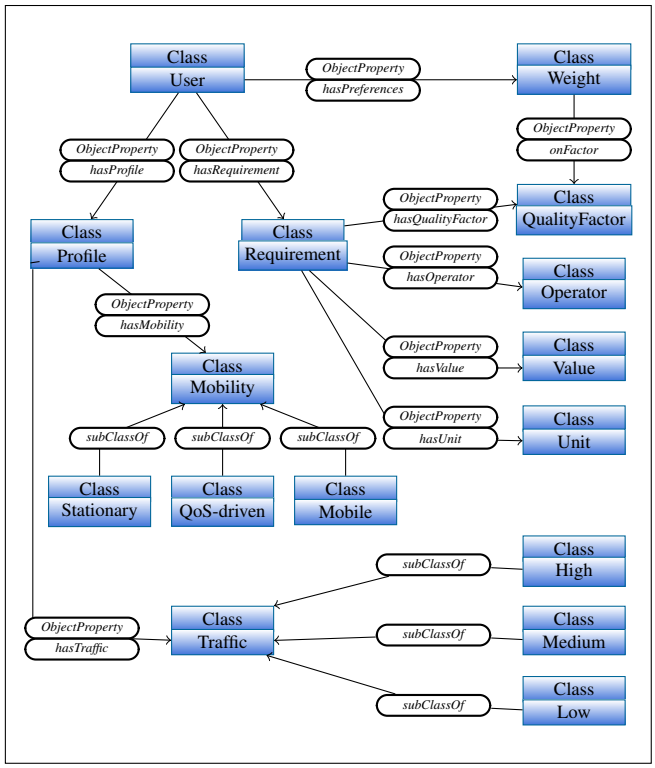

Figure 5. Overview the User QoS ontology

languages that additionally support semantic annotations, notably XML-based languages enriched with OWL semantic annotations [13].

Referencing QoS semantic concepts allows for coping with the syntactic heterogeneity problem and yields semantically enriched QoS descriptions that combine the accuracy of syntactic languages with the rich semantics of QoS ontologies.

This approach is important in broadening QoS understanding between heterogeneous specifications addressing QoS for different purposes. Thanks to our model, these specifications can reference common ontological concept and share the same vision of QoS. As depicted in Fig. 6, our model can be used to express user QoS requirements and to specify the QoS level provided by services.

The latter point leads to another, increasingly important, aspect, that is enabling service providers and users to agree on a contract concerning the quality of the offered service (i.e., Service Level Agreements SLA). Specifically, our model is useful for both the contract specification and contract enactment. In the first case, our ontologies can be used to define QoS terms of an SLA, or other terms that define aspects such as user device capabilities. Once the contract is defined, our ontologies can be also used to monitor SLAs and to adapt them to changing conditions in a dynamic way. The need for dynamic SLA specification, negociation and monitoring is getting increasingly important in SOC [16].

The proposed model is therefore a general framework (Fig. 6) that provides the appropriate ground for several engineering capabilities including QoS requirements engineering and QoS-based service engineering (e.g., service discovery, SLA management, monitoring).

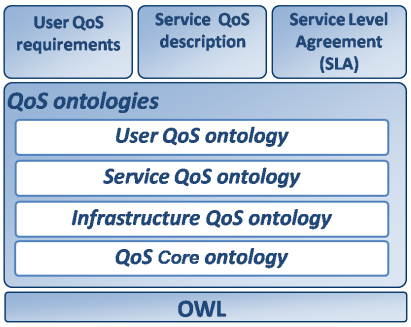

Figure 6. QoS specification framework

4 Related Work

QoS modeling and related issues have been the topic of wide research crossing distinct communities. For the purposes of this work, it is worth mentioning the service oriented computing community (e.g., [8, 6, 2, 14, 3]), the end-to-end QoS community (e.g., [7]), as well as the pervasive computing and mobile computing communities (e.g., [9, 15]). Even if different in purpose, these efforts intended at least to identify relevant quality factors affecting QoS and being associated to a specific resource (e.g., an application service, a network, a device).

For instance, focusing on the service oriented computing community, [8, 2] identify quality factors deemed useful to characterize services' QoS. [6] proposes a methodology enabling to express user QoS requirements and to link them to quality factors using the notion of tracability. More recently, some authors [14, 3] have derived semantic languages of QoS ontologies that represent generic models for QoS. These models identify a set of general and abstract concepts needed to define QoS factors and the way they are assessed. Nevertheless, several QoS factors (e.g, performance) are difficult to evaluate in an accurate manner without considering the networks or the devices enabling services and users to interact.

For this matter, other models considering QoS in an end-to-end manner have been proposed throughout the literature. Marchetti et al. [7] present a quality model for capturing and reasoning about quality aspects of multichannel services (a channel being the abstraction of a device and a network). This model enables a clear separation of quality aspects of services, networks, devices and users. Further, it embeds rules enabling the evaluation of end-to-end quality. Nevertheless, this model does not consider the dynamic nature of these elements (i.e., networks, devices, services, users) such as user mobility, thus, it cannot be used to give accurate description of QoS in dynamic environments.

To cope with such issue, another related active area of research focuses on QoS modeling while taking into account the dynamic nature of resources. Capra et al. [1] define a context-aware QoS model using semantic representation. The model considers three types of resources: *services, sensors* and *components*. Components can be either local (e.g., battery, memory, CPU) or remote (addressable software resources). Based on this work, McNamara et al.[9] addressed QoS issues related to the mobility of such resources. In this work, the authors combine mobility patterns with QoS information in order to give an accurate evaluation of QoS. This work presents two main drawbacks: first it considers specific patterns of user mobility (i.e., "seasonal" mobility, in the sense that it is related to the daily routines of users). Second it does not take into account QoS at the end-user level (i.e., user QoS requirements).

To this regard, a Wireless QoS-aware Mobility (WiQoSM) model is proposed in [15]. The authors introduce a model which considers QoS, user mobility, user behavior and wireless technology. This model defines generic mobility patterns and takes QoS at the end-user level into account. The main drawback of WiQoSM is that it totally separates QoS from the aforementioned aspects as it consider them in four disjoint models: 1) a QoS model, 2) a user mobility model, 3) a user traffic model and 4) a wireless technology model. This structure enables WiQoSM to express the above factors, but it does not allow to consider them jointly, which represents a major handicap for having a comprehensive QoS description.

Our model considers collectively the above aspects. Using the quality chain concept, we can describe relationships between QoS factors associated to networks, devices, application services and end-users, thus enabling a comprehensive and insightful description of QoS. Table 1 compares relevant QoS models (among the discussed ones) based on the criteria fulfilled by our model.

5 Conclusion

Semantic QoS modeling appears as a promising paradigm for QoS engineering in service oriented computing. In this work, we present a semantic QoS model that addresses QoS of main elements acting in service environments (i.e., networks, devices, application services and end-users) and takes into account the dynamic nature of these elements, thus enabling an insightful description of QoS. Additionaly, our model is extensible in that domain-specific QoS factors can be easily added. Furthermore, the proposed model focuses on representing QoS knowledge with rich semantic information rather than specifying a language for QoS. Coupled to XML-based QoS specifications, the developed ontologies can formulate a robust QoS description

Issue	Criteria	Maximilien[8]	MOQ[6]	QoSOnt[2]	Papaioannou[14]	Dobson[3]	Marchetti[7]	Resta[15]
1. Semantic modeling	Semantic description of QoS	•	•	•	•			
2. Support for devices	Support for device mobility						•	•
	Support for device capabilities						•	
3. Support for networks	Support for network connectivity							•
	Support for network performance							
4. Support for environments	Support for environment characteristics							
5. Support for application services	Support for adaptation							
	Support for context awareness						•	
	Support for common QoS factors	•	•	•	•	•	•	•
6. Support for users	Support for user requirements		•					•
	Support for user mobility and generated traffic							•

Table 1. Relevant QoS models compared according to the discussed issues

framework that combines the rich semantics of QoS ontologies with the accuracy of QoS specification languages.

The presented model makes part of our ongoing work addressing QoS-aware middleware for dynamic service environments. Key features of this middleware address semantic QoS-aware service discovery, composition and monitoring, which will be based on our comprehensive QoS model.

Acknowledgement

This research is partially supported by the SemEUsE project[1] funded by the french National Research Agency (ANR).

References

[1] L. Capra, S. Zachariadis, and C. Mascolo. Q-CAD: QoS and Context Aware Discovery Framework for Mobile Systems. In *ICPS*, page 453456, 2005.

[2] G. Dobson, R. Lock, and I. Sommerville. QoSOnt: a QoS Ontology for Service-Centric Systems. In *EUROMICRO '05: Proceedings of the 31st EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 80–87, Washington, DC, USA, 2005. IEEE Computer Society.

[3] G. Dobson and A. Sanchez-Macian. Towards Unified QoS/SLA Ontologies. In *SCW '06: Proceedings of the IEEE Services Computing Workshops*, pages 169–174, Washington, DC, USA, 2006. IEEE Computer Society.

[4] J. R. Hobbs and F. Pan. Time Ontology in OWL, 2006. http://www.w3.org/TR/owl-time/.

[5] S. Kalasapur, M. Kumar, and B. Shirazi. Evaluating service oriented architectures (soa) in pervasive computing. In *PERCOM '06: Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications*, pages 276–285, Washington, DC, USA, 2006. IEEE Computer Society.

[6] H. M. Kim, A. Sengupta, and J. Evermann. MOQ: Web Services Ontologies for QOS and General Quality Evaluations. In *European Conference on Information Systems (ECIS 2005)*, 2005.

[7] C. Marchetti, B. Pernici, and P. Plebani. A quality model for multichannel adaptive information. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 48–54, New York, NY, USA, 2004. ACM.

[8] E. M. Maximilien and M. P. Singh. A Framework and Ontology for Dynamic Web Services Selection. *IEEE Internet Computing*, 8(5):84–93, 2004.

[9] L. Mcnamara, C. Mascolo, and L. Capra. Trust and Mobility Aware Service Provision for Pervasive Computing. In *First International Workshop on Requirements and Solutions for Pervasive Software Infrastructures*, 2006.

[10] D. Min, E. Kim, Y. Lee, G. Kang, and J. B. Clark. Web Services Quality Model, 2007. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsqm.

[11] S. B. Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers. EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support. *J. Syst. Softw.*, 81(5):785–808, 2008.

[12] E. D. Nitto, C. Ghezzi, A. Metzger, M. Papazoglou, and K. Pohl. A Journey to Highly Dynamic, Self-Adaptive Service-Based Applications. *Automated Software Engg.*, 15(3-4):313–341, 2008.

[13] N. Oldham, K. Verma, A. Sheth, and F. Hakimpour. Semantic WS-Agreement Partner Selection. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 697–706, New York, NY, USA, 2006. ACM.

[14] I. V. Papaioannou, D. T. Tsesmetzis, I. G. Roussaki, and M. E. Anagnostou. A QoS Ontology Language for Web-Services. In *AINA '06: Proceedings of the 20th International Conference on Advanced Information Networking and Applications - Volume 1 (AINA'06)*, pages 101–106, Washington, DC, USA, 2006. IEEE Computer Society.

[15] G. Resta and P. Santi. WiQoSM: An Integrated QoS-Aware Mobility and User Behavior Model for Wireless Data Networks. *IEEE Transactions on Mobile Computing*, 7(2):187–198, 2008.

[16] K. Tserpes, D. Kyriazis, A. Menychtas, T. A. Varvarigou, F. Silvestri, and D. Laforenza. An Open Architecture for QoS Information in Business Grids. In T. Priol and M. Vanneschi, editors, *CoreGRID*, pages 37–49. Springer, 2007.

[1] SemEUsE project: http://www.semeuse.org