Quality Standards Ontology for Web Services Discovery

F. Losavio[ab]

MoST, Centro ISYS, Escuela de Computación, Facultad de Ciencias
Universidad Central de Venezuela
Caracas, Venezuela

N. Levy[c]

CEDRIC,
CNAM
Paris, France

A. Ramdane-Cherif[d]

LISV,
Université de Versailles
St-Quentin en Yvelines, France

A. Matteo[e]

MoST, Centro ISYS, Escuela de Computación, Facultad de Ciencias
Universidad Central de Venezuela
Caracas, Venezuela

H. Hadj Salem[f]

LISV,
Université de Versailles
St-Quentin en Yvelines, France

[a] Thanks to the CDCH, ADIRE project No. PG-03-2008/2 , the UCV No 2009000624 project of Fonacit, Venezuela, and the O2M project, France.
[b] Email: francisca.losavio@ciens.ucv.ve
[c] Email: Nicole.Levy@cnam.fr
[d] Email: rca@lisv.uvsq.fr
[e] Email: alfredo.matteo@ciens.ucv.ve
[f] Email: hasna.hadjsalem@gmail.com

This paper is electronically published in Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

Abstract

Web Services (WS) technology is failing in providing service-discovering techniques considering non-functional properties. Semantic search is based on functionality, regardless of non-functional requirements. Quality properties of services are considered marginally; definitions and metrics are not provided, making difficult common understanding and retrieval of concepts. Using quality standards is made difficult due to conceptual inconsistencies among different standards. This work proposes an ontological WS discovery approach based on semantic matching process of functional and non-functional requirements. A WS is considered a software component providing functionality accomplishing specific quality goals. Our WS discovery process matches functional and non-functional requirements, ranks according to preferences, considering three related ontologies: 1. An ontology to integrate quality standards and retrieve properties and metrics, 2. An ontology to model relations between these standards and preferences to rank functionality and/or qualities, 3. An ontology relating quality models to WS functionality. A prototype tool supports this approach.

Keywords: Web Services discovery, ontology, Web Services quality, quality standards, ISO/IEC 9126-1, non-functional requirements, matching process

1. Introduction

Web services (WS) technology arises as a challenge to face the evolution of software systems. One of the problems is to discover efficiently and precisely the adequate WS with respect to the application requirements. Syntax-based WS discovery tools, like UDDI (Universal Description Discovery and Integration), lacks precise documentation and does not consider the variability of the terminology; semantic-based search using ontologies to capture WS concepts, like OWL-S [1], is now being adopted; however this technique is mostly based on functionality and not on non-functional requirements which can be an important issue to characterize a service by the quality it provides. The search for a WS based only on functionality can produce a huge amount of results. Therefore non-functional requirements must be taken into account to refine the search. On the other hand, a quality model is a useful framework to specify software quality requirements. ISO/IEC 9126-1 [18] defines a quality model framework as a set of abstract quality properties (characteristics), which are refined into lower level quality properties (sub-characteristics) and measurable quality properties (attributes). In this context, only software product quality will be considered; the quality model framework will be customized to describe the quality related to a WS, which is considered as a software component offering services, i.e. providing a kind of functionality with precise quality properties to satisfy.

The goal of this paper is to present an ontology-based approach to WS discovery considering a matching on both functional (FR) and non-functional requirements (NFR); this approach is focused on standards to offer a unified view of the terminology on software quality. WS are grouped in broad categories or types, representing the core functionality of a family of WS applications, for example transactional WS. NFR related to WS functionality are expressed as a set of quality properties; for example, security, reliability, and efficiency and are specified by a quality model. An ontology, inspired from a previous work of Losavio, Matteo and Levy [2], represents the quality view of a family of applications in a domain, considering architectural and functional quality, and it allows to model software quality using different standards. In order to facilitate common understanding in the use and retrieval of quality properties and their metrics, this ontology integrates three well known standards on software product quality: WSA (Web Services Architecture) of W3C (World Wide Web Consortium) [3] defining the SOA (Service Oriented Architecture) quality properties, ISO/IEC 13236 for QoS [18], and ISO/IEC 9126-1 [20] for the quality model specification. A unified terminology on software quality is missing and even more, a term may have different meaning in

different standards. An example of the variability and ambiguity of the terminology is the diversity of definitions of the concepts related with software evolution: ISO/IEC 9126-1 defines changeability (sub-characteristic of maintainability) as "the capacity of software to support changes"; flexibility is defined by the IEEE 1061 standard as "the ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed"; WSA [3] defines extensibility & scalability as a property for flexible applications, in the sense of "adaptation to changing volume of information or to the addition/deletion of new components", and they also define management & provisioning as a "maintenance facility, allowing the capability to be adapted to a changing environment". Our ontological approach will help to trace a term through the different standards defined in the ontology. Another practical use of this approach is to retrieve at different abstraction levels, the quality properties and the metrics characterizing a WS, allowing traceability among quality standards and facilitating the discovery process. A user for example, can speak in terms of having a "short response-time"; however the QoS can be expressed in terms of "latency". It will help to retrieve commonly accepted metrics, even if the quality property is defined in a standard that does not specify any metrics.

The main goal of this work is to propose a matching process combining FR and NFR for WS discovery. Expert-defined preferences are used to rank quality properties, and user-defined preferences are used to rank selected services. Three related ontologies have been defined: an ontology to integrate the three quality standards (*Onto-Std-Qualities*), an ontology to relate the quality properties with user or expert-defined preferences to rank them (*Onto-Relation-ComPref*), and an ontology for the quality model and the concepts related to the offered services (*QStdOnt*). This ontology contains instances of the WS core functionality (FR), for example transactional WS, and the associated quality properties expressed by the corresponding quality model for example integrity, time behavior, and availability. A straightforward usage of our approach is to be easily applied to improve the FR matching process with NFR issues, in semantic-based matchmaker tools.

Besides the introduction and the conclusion, this paper is structured into four main sections: section 2 introduces the software quality standards used. Section 3 describes the proposed WS discovery process and the ontologies involved. Section 4 presents the prototype tool based on Alive Matchmaker; the discovery process is applied to a transactional WS, as a case study to illustrate our approach with several scenarios of usage; finally in section 5 related works are discussed.

2. Software quality standards and functional core of WS applications

In what follows, three standards on software product quality used in our approach are presented. These particular standards were selected as representative of widely known and accepted standards used in practice within the software community.

- The ISO/IEC 9126-1 standard [18] is a framework to express software product quality as a hierarchical tree-like structure composed by six high abstraction level quality characteristics that can be refined into sub-characteristics, until the quality attributes or measurable elements are attained (see Table 1). In this work, the ISO/IEC 9126-1 quality model has to be customized to the WS domain.

Recently, the new standard ISO/IEC 25010 [4] has been introduced to update ISO/IEC 9126-1 with eight characteristics, instead of six. According to this new

version, *compatibility* (co-existence, interoperability) and *security* (confidentiality, integrity, non-repudiation, accountability, authenticity), have been added as high level characteristics. Moreover, the functionality characteristic name has been changed to *functional suitability* (completeness, correctness, and appropriateness); the sub-characteristics correctness corresponds to accuracy in ISO/IEC 9126-1. The *maintainability* characteristic has also important changes with respect to the sub-characteristics: modularity and reusability have been added as new sub-characteristics; modifiability keeps the same name but integrates now two sub-characteristics of ISO/IEC 9126-1, stability and changeability; analyzability has not been changed. Compliance with other standards is no more considered part of the quality model. In this paper we will be using the still "official" ISO/IEC 9126-1 version of the quality model. However, the new ISO/IEC 25010 standard can be easily added to the Onto-Std-Qualities ontology, establishing the equivalence with ISO/IEC 9126-1. On organization using the old standard can then easily change for the new one using this ontology.

Table 1. The ISO/IEC 9126-1 quality characteristics and sub-characteristics.

Quality Characteristics	**Quality Sub-characteristics**
Functionality	Suitability, Accuracy, Interoperability, Security, Compliance
Reliability	Maturity, Fault tolerance, Recoverability, Compliance. Availability is a combination of the first three sub-characteristics.
Usability	Understandability, Learnability, Operability, Attractiveness, Compliance.
Efficiency	Time behavior, Resource utilization, Compliance.
Maintainability	Analyzability, Changeability, Stability, Testability, Compliance.
Portability	Adaptability, Installability, Co-existence, Replaceability, Compliance.

- The WSA of the W3C [3] specifies the "de facto" industrial standard for a reference architecture, shown in Table 2. The WSA requirements document of the W3C describes seven critical top-level quality goals that are the minimal set of requirements for a common architecture that a WS-based application should comply. The SOA architecture underlying WS applications follows a general client-server model with a message- passing style for components, and a peer-to-peer pattern for connectors is compliant with the WSA standard. The WSA requirements are also refined into sub-characteristics; for example security is refined into threat of accessibility attacks, authentication of the parties, authentication of authorship of data, authorization, confidentiality, data integrity, origin: enable non-repudiation of origin and receipt between transaction parties, general recommendations for security policy and security management. Metrics are not provided in this standard.

Table 2. The WSA requirements of W3C

WSA Requirements	**Description**
Interoperability	Must interoperate within different environments.
Reliability	Must hold availability and stability.
WWW Integration	Must be consistent with the WWW evolution.
Security	Must provide a reliable environment to perform its online processes.
Scalability and Extensibility	Must allow flexible applications in the sense of adaptation to changing volume of information or to the addition/remotion of new components.

Team Goals	Must meet the needs of the user community.
Management and Provisioning	Must provide maintenance facility, allowing the adaptability to a changing environment.

Table 3. Functional requirements for WS applications and quality model for each functionality.

WS type (core functionality)	Main functional requirements	ISO/IEC 9126-1 Quality Model to express NFRs for each WS type				
		Functionality	*Reliability*	*Maintainability*	*Portability*	*Efficiency*
Information and collaborative environments	Data Base operations: query, access, modification, exchange	- accuracy	- availability	- changeability		
Transactional	e-commerce operations: data exchange, access control, encrypting	- security (integrity) - accuracy	- availability			- time behavior - resource utilization
Workflow	Process monitoring operations: Control planning	- suitability				
Web Portal	E-search and e-communication operations: consult, access		- availability		- adaptability: scalability	- time behavior - resource utilization
Security	Integrity operations: access control, encrypting	- security				

- The ISO/IEC 13236 for Quality of Service (QoS) [20] expresses general characteristics of communication in distributed and open environments; it is used here to specify metrics corresponding to quality attributes, which are measurable elements at the lowest level in the hierarchy of the ISO/IEC 9126-1 quality model. Main groups of characteristics are related with time, governance, capacity, integrity, safety, security, reliability and precedence. The standard claims that other characteristics may be added, with the evolution of information technology. On the other hand, the goal of ISO/IEC 13236 is to assist in the specification and design of technology-based software systems; it describes how to characterize, specify and manage requirements related with the quality of the service (QoS). It provides a common language to services, clients and providers.

According to ITU (International Telecommunication Union), standard X.902 (Information technology, Open distributed processing, Reference Model), a QoS is defined as a set of quality requirements present in the collective behavior of one or more object parameters. Mechanisms are part of the management functions and parameters are part of the context of a QoS. The range of attribute values (value of the QoS parameters) is established by metrics for WS quality requirements. A functional requirement originates from a client entity that uses a service, and it is translated to different QoS requirements, expressed as parameters. A mechanism is realized by the client entity to satisfy one or more QoS. A WS is considered in this

context, as a software component offering services, i.e. providing precise quality properties for a certain functionality. Different types of WS are grouped in Table 3 on the basis of the main functionality or service they provide [5]. They constitute the core functionality for families of WS-based applications.

3. Ontologies and WS discovery

The process of WS computing involves three basic steps: service discovery, service selection and service composition [6]. In this work we will be interested mostly in service discovery that relies greatly on the service description, involving functional issues such as name, type, operations, and I/O data formats. Service selection relies more on quality properties, and it is responsible of identifying which is the best WS with respect to the user needs. However, the final user has no skills to define technical parameters such as throughput and latency, and speaks more in terms of high-level properties such as performance or efficiency. The WS selection process may be accurate only if services are described very precisely for the discovery process. However, most discovering techniques based on SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) and UDDI with XML, only rely on a syntactical description of WS interfaces. Recently, semantics approaches based on ontologies have been developed. The Semantic Web is mostly concerned with the interoperability of static information available on internet and their descriptions, as a common language understandable by different machines and platforms. A semantic WS is a WS described by semantic annotations, such as the OWL-S ontology [1], which is a rich model for describing WS functionality, however, it offers few concepts to express their qualities properties or QoS, and does not provide equivalence relationships among terms expressed at different abstraction levels.

3.1 WS discovery Process

The main activities of the proposed WS discovery process, combining syntactic and semantic issues are:

First activity: matching of FR.
The WS query using FR is expressed as an OWLS file; the matching is performed syntactically using a Web server, containing the information on the WS functionality; the matching results are the URLs with the descriptions of the services represented also by OWLS files, responding to the functional requirements of the client.

Second activity: matching on NFR.
This search is also based on a matching to retrieve QoS metrics from the properties defined in the quality model for the specific WS functionality, that are specified in the QStdOnt ontology; an inference engine of Protégé is used, and the queries are expressed in SQWRL; for example, search for a Transactional WS that requires efficiency, with a certain throughput, measured by a resource-utilization metric.

Third activity: WS ranking according to expert-assigned common preferences.
The services, found after the matching of NFR, are ranked according to their

common preferences specified in Onto-Relation-ComPref Ontology. Generally, levels (high, medium, and low) are associated to each quality property in order to rank them. Domain experts assign these preferences to quality properties. For example, WS with high-level rate of Internet access are frequently demanded.

Fourth activity: WS ranking according to user-assigned functional preferences and/or priorities.

A particular user assigns his/her own functional preferences. For example, the service description is syntactically similar to the query specification; services are defined at a more abstract level (superclass) than those services with more specific outputs (subclass). If the user has expressed preferences for the FR and/or assigned priorities among the quality properties associated to the functionality, matching is used again to classify these services, according to their priority. For example, the quality property confidentiality is defined in (ISO/IEC 13236, 1993) as the protection against unauthorized viewing of data; it can have greater priority than freshness, also defined in (ISO/IEC 13236, 1993) as the time since the data was produced. The URL with the QoS is the result of this activity. The user can make exact or approximate searches (P>/</=x), where P is a metric corresponding to a quality property and x is the value (QoS) of the metrics. The prototype tool supporting this process will be described in section 5.

3.2 Ontologies

The approach proposing a standard quality model related to the WS functionality (see Table 3), to express in terms of quality properties the NFR associated to a WS application, can be found in [5]. In this work, three ontologies have been defined to support the WS discovery process: an ontology to integrate the three quality standards (Onto-Std-Qualities), an ontology to model relations between these standards and common preferences (Onto-Relation-ComPref) and finally an ontology, which is updated according to service requests (QStdOnt) (see Figure 1).

- *Onto-Std-Qualities*: describes the quality standards ISO-IEC 9126-1, ISO-IEC 13236, and WSA (see section 2).
- *Onto-Relation-ComPref*: represents the static part of the knowledge: WS core functionality, related with the three standards. For example, Transactional WS and sub-classes WS-ISO-IEC13236, WS-ISO-IEC9126-1 and WSA. This ontology imports Onto-Std-Qualities. These sub-classes are the instances of the WS using some of the mentioned standards. For those Web services using more than a single quality standard to define its qualities, the concept OthersWS has been introduced. This ontology contains some SQWRL rules to express relationships between quality standards. Preferences levels assigned by domain experts are also memorized to display the discovered services in the preferred order. Onto-Relation-ComPref ontology imports Onto-Std-Qualities ontology.
- *QStdOnt*: represents the dynamic part of the knowledge. Web Services and their QoS may evolve since new services are published, others are abandoned, and

their QoS can be defined and/or modified. This ontology imports Onto-Relation-ComPref ontology.

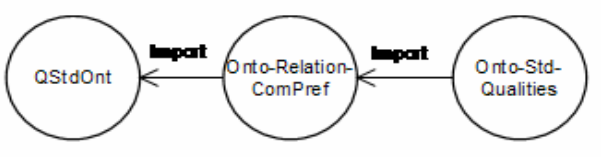

Figure 1. The three ontologies on quality standards involved in the semantic-based WS discovery process

Partial views of ISO/IEC 9126-1 and WSA standards memorized in the Onto-Relation-ComPref ontology are shown in Figures 2 and 3. Note that "Level" in these figures refers to the preference level assigned to the quality property.

4. Tool and case study

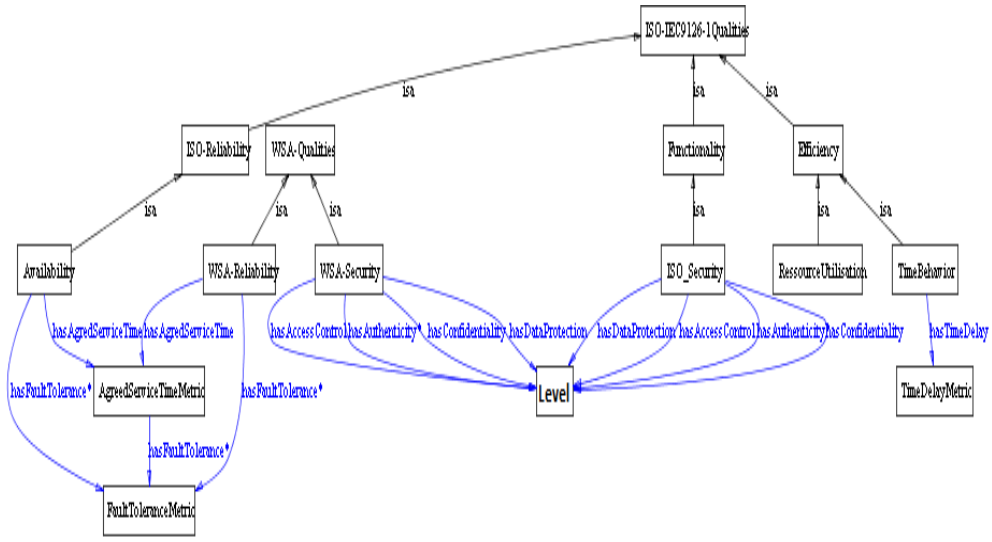

Figure 2. Partial view of the Onto-Relation-ComPref

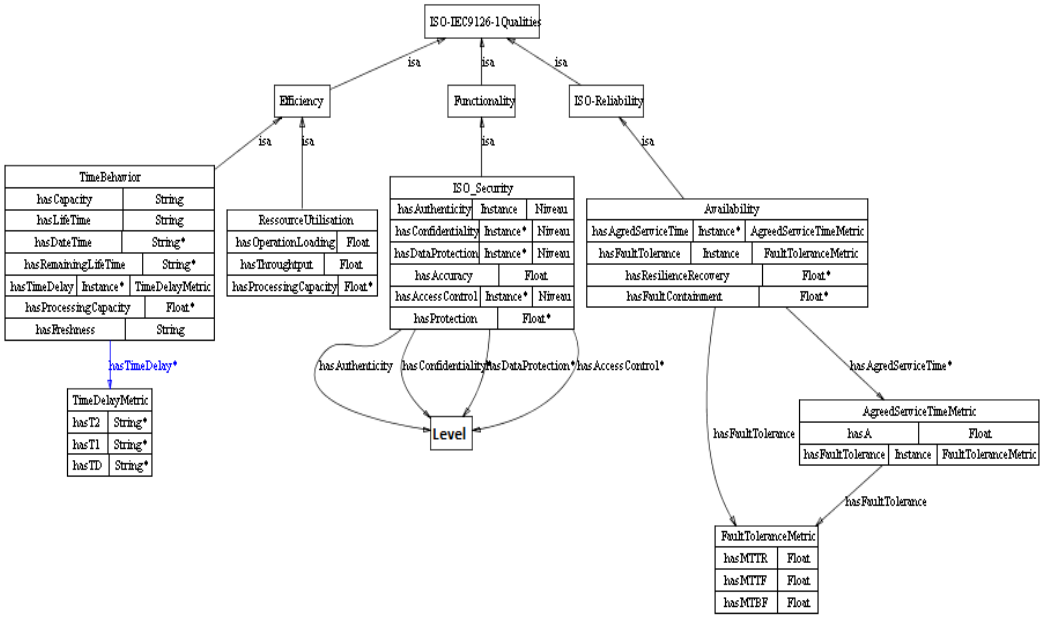

Figure 3. Partial view of the Onto-Relation-ComPref ontology: attributes, sub-attributes and metrics of ISO/IEC13236 related to the ISO/IEC 9126-1 quality characteristics

This section shows first the architecture of the tool supporting the WS discovery, according to the approach described in section 3. Then, a scenario is presented as case study.

As it has been pointed out, our process is based on semantic matching of both FR and NFR, considering also preferences. Several tools for WS discovery, generally called matchmakers, use specific languages to describe WS semantics, such as OWL-S (Martin & al., 2004) [1], WSMLg, SAWSDLh. In order to select a semantic matchmaker using FR and NFR, we analyzed the complete matchmakers classification of (Klusch, 2009) [7]. Our selection was focused on those tools based on the widely used OWL-S language for WS description; Alive Matchmaker was the only one claiming to consider FR and NFR. After testing it, we found that actually, only FR were handled, being just an hybrid matchmaker on services input/output; however, it allows to classify FR based on user's preferences, which is an interesting issue. This matchmaker has been adapted to our WS discovery process, to include NFR. The other matchmakers studied, ROWLS-Matchmakeri, IO:GSD-MM , FC-Matchj, and OWLS-iMatcher2k only allowed functionality-based discovery (Klusch, 2009)[7]. The lack of a unique and commonly accepted ontology on WS quality may be one of the difficulties to develop a tool for WS discovery based explicitly on NFR. The unified ontology for quality standards proposed in this work is a step towards this aim.

4.1 The tool

[g] www.w3.org/Submission/WSML/
[h] www.w3.org/TR/2007/REC-sawsdl-20070828/
[i] www.springerlink.com/content/d851706700h70714/fulltext.pdf
[j] lcl.uniroma1.it/dspace/bitstream/123456789/313/1/File_539.pdf
[k] www.ifi.uzh.ch/ddis/research/semweb/imatcher/

The tool has four main components, beside the ontology QStdOnt: a User Interface (UI), a Matchmaker (Match), an Inference Engine and a Registry Web Server (see figure 4).

1. Matching of FR

The components involved in this activity are:

- The User Interface (**UI**): the request is formulated, using an OWLS file, where the client specifies his FR.
- The **Registry**: it contains a set of WS. In this work, an Apache-Xampp Web Server has been used to publish WS, since UDDI only allows the publication of the syntactical description of the services (WSDL).
- The Matchmaker (**Match**), to search published services corresponding to the query based on FR. In the Alive Matchmaker used, the type of matching based on services input/output has to be specified: exact matching, inclusive matching (subclass, immediate subclass) or partial matching (super class, immediate super class).

The FR based matching result is a list of URLs pointing to service descriptions (OWLS files) responding to client's FR request.

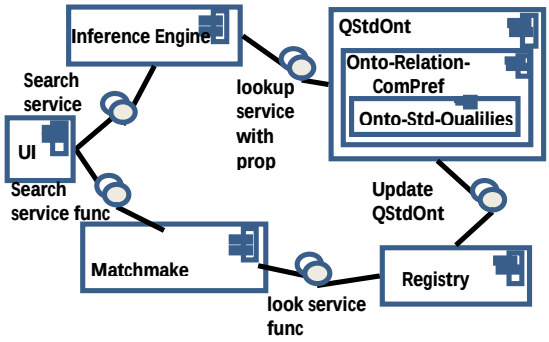

Figure 4. Architecture supporting the WS discovery process

2. Matching of NFR and 3) ranking according to common preferences

The NFR search uses the Inference Engine component, the Pellet inference engine (Protégé 3.4.4), executing the client's query in SQWRL on the QStdOnt ontology. The result is also a list of URLs with the QoS values corresponding to the request, ordered by common preferences.

Notice that by an equivalence relation, an abstract quality property of a given standard can correspond to another quality of another standard. For example reliability, defined by ISO/IEC 9126-1, can correspond to availability defined by WSA, and be measured by the AgreedServiceTime attribute, defined by ISO/IEC 13236 as the proportion of agreed service time that a connection is available.

Alive Matchmaker (Andreou, 2009) [8] allows functional preferences to be expressed and prioritized as shown in what follows:

Example 1:

- PS: Services having a textual description nearest to the query textual description are preferred
- PR: Services with greater reliability are preferred.
- PI: services with the most precise input as possible are preferred.

In order to express priorities on these criteria, a relation between them must be defined.

Example 2:

- PI is as important as PR,
- PS is less important that the other preferences.

→ (pI ~ pR) > pS

In order to rank services found according to required functional preferences and/or priorities, a Java program using the Alive Matchmaker library has been developed.

4.2 Case study

The case study presented aims at finding WS from an existing book directory of services. We are looking for WS that have a book input and price output. Once several candidates are founded, the idea is to order them considering NFR and user's preferences. Three different queries are presented.

1) Matching of FR

WS are searched from OWLS-TC[I] version 3.0. We search services that have a book input and price output. Figures 5 and 6 show the respective profiles of a published service and the query.

```
− <profile:Profile rdf:ID="BOOK_PRICE_PROFILE">
    <service:isPresentedBy rdf:resource="#BOOK_PRICE_SERVICE"/>
    <profile:serviceName xml:lang="en"> BookPriceService </profile:serviceName>
    <profile:textDescription xml:lang="en"> return price of a book </profile:textDescription>
    <profile:hasInput rdf:resource="#_BOOK"/>
    <profile:hasOutput rdf:resource="#_PRICE"/>
    <profile:has_process rdf:resource="BOOK_PRICE_PROCESS"/>
  </profile:Profile>
```

Figure 5. BookPrice.owls service profile

[I] http://projects.semwebcentral.org/projects/owls-tc/: A directory of services, ontologies and queries provided for developers to evaluate the performance of their matchmakers.

```
− <profile:Profile rdf:ID="BOOK_PRICE_PROFILE">
    <service:isPresentedBy rdf:resource="#BOOK_PRICE_SERVICE"/>
    <profile:serviceName xml:lang="en"> BookPrice </profile:serviceName>
    <profile:textDescription xml:lang="en"> Uses the ISBN to return price of a book </profile:textDescription>
    <profile:hasInput rdf:resource="#_BOOK"/>
    <profile:hasOutput rdf:resource="#_PRICE"/>
    <profile:has_process rdf:resource="BOOK_PRICE_PROCESS"/>
  </profile:Profile>
```

Figure 6. Query profile

As a result of this query, three services have been retrieved:

- `http://127.0.0.1/services/1.1/book_price_service.owls#BOOK_PRICE_SERVICE`
- `http://127.0.0.1/services/1.1/book_pricereviewbook_service.owls#BOOK_PRICEREVIEWBOOK_SERVICE`
- `http://127.0.0.1/services/SWSQ/BookPrice.owls#BOOK_PRICE_SERVICE`

2) Matching of NFR and **3**) ranking according to common preferences

Now, the problem is to order these retrieved services considering NFR and user's preferences. Three queries using the quality ontologies are presented in what follows.

Query A: simple query, exact search, negative preference L

Example: services with loading time less or equal to 1.5 (ranked according to common preferences for the « OperationLoading » in this case). Request and response are respectively shown in Figure 7 and Table 4.

```
TransactionalWebService(?x) ∧ hasQualities(?x, ?y) ∧
p1:hasOperationLoading(?y, ?z) ∧ swrlb:lessThanOrEqual(?z, 1.5) ∧
hasPreferenceValue(OperationLoading, ?a) ∧ swrlb:multiply(?b, ?z, ?a) →
sqwrl:select(?x, ?z, ?b) ∧ sqwrl:orderByDescending(?b)
```

Figure 7. SQWRL Query - Example A

Table 4. Result – Example A

Service's name	OperationLoading
Book_PriceReviewBook_Service	0.300000001192092896
ook_Price_Service	0.699999988079071

Query B: simple query, approximate search, positive preference, relation among quality standards

Example: services where security is defined (Figure 8 shows the query). If the search does not take into account relations between quality standards, only one WS is retrieved (Table 5). When taking them into account, three WS are retrieved (Table 6).

```
TransactionalWebService(?x) ∧ hasQualities(?x, ?y) ∧ p1:WSA-Security(?y) ∧
tbox:isOWLProperty(?z) ∧ abox:hasValue(?y, ?z, ?a) →
sqwrl:select(?x, ?z, ?a)
```

Figure 8. Query, example B

Table 5. Result of the query, without considering relationships among quality standard

Name	Property	Value
Book_Price Service	P1:hasProtection	0.85000002384185

Table 6. Result of the query with relations among the quality standards

Name	Property	Value
Book_Price_Service	p1:hasProtection	0.8500000238418579
Book_Price_Service	p1:hasProtection	0.8500000238418579
BookPrice	p1:hasProtection	0.800000011920929

Query C: complex query, simple search

Example: services having the two quality properties (attributes) Throughput and OperationLoading defined. « Throughput » has positive preference, « OperationLoading » has a negative preference and « Throughput » has greater priority than « OperationLoading ». Figures 9 and Table 7 illustrate this scenario. Three services are retrieved with the values of the two NFR attributes:

- http://127.0.0.1/services/1.1/book_price_service.owls#BOOK_PRICE_SERVICE Throughput= 100.0 and Operation loading= 0.7
- http://127.0.0.1/services/1.1/book_pricereviewbook_service.owls#BOOK_PRICEREVIEWBOOK_SERVICE Throughput= 80.0 and Operation loading= 0.3
- http://127.0.0.1/services/SWSQ/BookPrice.owls#BOOK_PRICE_SERVICE, Throughput= 80.0 and Operation loading= 1.9

```
// We prefer high values of Throughput
Preference<Match> PR = Preference.
<Double>natural().reverse().onResultOf(ThroughputExtractor);
// We prefer low values of Operation //Loading
Preference<Match> PI = Preference.
<Double>natural().onResultOf(OperationLoadingExtractor);;
//ThroughPut has greater priority than //OperationLoading
Preference<Match> pref = PR.preferredTo(PI);
```

Figure 9. Properties with preferences and priorities

Table 7. Services having the throughput and operation loading attributes

Service	Throughput	OperationLoading
Book_PriceReviewBook_Service.owls	80	0.3
BookPrice.owls	80	1.9
Book_Price_Service.owls	100	0.7

5. Related Works

Many works have been published on WS discovery. We focus here on those involving software quality and ontological approaches. Tran, Tsuji, and Masuda

(Tran, Tsuji, &Masuda in [9] have proposed WS-QoSOnto ontology on WS quality. They use a quality model defined in [10] that uses the QoS standard of W3C for WS and the specification defined in [11]. This ontology describes: QoS attributes, such as tendency (preferences), mandatory (satisfaction level demanded by the quality property of the QoS, as required, optional), weight (priority); QoS metrics and measurement; relations between quality properties, such as independent with, inverse of, opposite of, same as, aggregate of, influence of. Another topic found in the literature is the adaptation of OWL-S through its "profile" class with quality issues [12]. However, this work and also Bleul and Weise in [13] focus more on the problem of terms having the same metric defined by different units, rather than on terms with the same name and different semantics. An extension of OWL-S by Jean, Losavio, Levy, and Matteo [14] based on the OSQS ontology defined in [2] which inspired this work, is proposed to describe the quality of a WS as a function of one or more standards on software quality, to offer a unified standard terminology on software product quality, as well as a quality view of the domain. It considers also the extension of SPARQL, to include NFR in the expression of the query. A service can be discovered even if it is described by a standard different from the one used in the query, exploiting the equivalence relation between standards. NFR expressed as quality requirements and user preferences are considered to rank the services obtained, responding to WS functionalities. However, to exploit this ontology, an ontology-based database (OntoDB) must be imported and the OntoQL language, associated to OntoDB must be extended. With respect to the publication of services, Ran in [15] proposes to extend the current UDDI to publish QoS as well as their syntactical descriptions. A new data structure is proposed: QualityInformation is a QoS taxonomy model (tModel) to be defined in an UDDI extension. These taxonomies describe provided services. Moreover, besides the provider, client and registry, a new entity is proposed, called Web Service QoS Certifier to evaluate and certify the QoS of a WS, to guarantee the published QoS. A similar approach of Rajendran, Balasubramanie, & Resmi Cherian [16] to the UDDI extension, proposes a QoS broker, an agent-based architecture for dynamic Web service selection, which facilitates NFR specification by clients along with FR. An efficient mechanism for finding the most suitable Web service according to client's requirements has been defined. The broker is also a Web service, to enhance architecture interoperability. The QoS information is represented in the UDDI registry by a tModel, which allows specification, standardization and reuse of QoS related concepts. However, none of these approaches use ontologies, but a kind of intermediary component to handle quality description using a tModel. Ouhab and Malki (Ouhab & Malki in [17] propose a solution to exploit a quality ontology defined in (Maximilien, 2004). This proposition and ours are similar: the use of a matchmaker for FR (OWLS-MX vs Alive-matchmaker), a Web server for service publication (Apache Tomcat vs Apache Xampp), an inference motor for the NFR matching and ranking of WS ontology (Kaon2 vs Pellet), preferences are expressed in quality ontologies and the two propositions are developed in Java. However, both solutions have different criteria. We have developed an ontology to describe a quality model, constructed by combining different software quality standards (WSA, ISO/IEC 9126-1, ISO/IEC 13236) with relationships between them. The ontology in (Ouhab & Malki, 2009) [17] is taken from Maximilien [10], and it is based on a QoS specification defined in [11]. We are using instead, the ISO/IEC 13236 specification for QoS that is an accepted international standard for communication in open and distributed systems environments. We can conclude about the WS discovery process, that it should consider non-functional properties, related to the

functionality of the WS, and that the matchmaking process should include semantic-based search using ontologies, to deal with the variability of the terminology on software quality, and in particular on QoS, to improve the discovery results. Preferences on the WS functionality and/or quality should also be part of the process, since they refine the search.

6. Summary

Consumers increasingly use services in their everyday life. The problem of selecting the most valuable service, according to their FR and NFR is not yet completely solved. In this paper, we combine our works on ontology and quality model, to enhance service-selecting process based on quality requirements. For this purpose, we propose a new WS discovery process focused on semantic matching of FR and NFR. The FR-based search uses the Alive Matchmaker; the NFR-based search considers a matching with the QoS (metrics), retrieved from quality model properties for WS specified in QStdOnt ontology. The matching uses an inference engine and queries are expressed in SQWRL; this search is enriched by the use of preferences on the quality properties, in the onto-Relation-ComPref ontology. In order to validate our discovery process, we have developed a prototype. We have presented some queries showing how our approach improves consumer satisfaction. The result is an ordered list of URLs with services descriptions, responding to the client's functional requirements, satisfying the required quality and user's preferences. However, the prototype tool has to be completed by a better integration of the three main components of the architecture. It has also to be pointed out that the units of measures for the metrics have not been included in our ontology. Different units of measures have to be described, as well as the conversions between them; some existing ontologies include counting rules, metrics, and values, like for example QoSmMO (Tondello & Siqueira, 2008) [19] and could be integrated to our tool.

7. Bibliographical references

[1] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirin, N. Srinivasan, and K. Sycara, 2004. OWL-S: Semantic Markup for Web Services, W3C Member Submission,Nov.

[2] F. Losavio, A. Matteo, and N. Levy, 2009. Web Services Domain Knowledge with an Ontology on Software Quality Standards, in Proceedings of the Third International Conferences on Internet Technologies and Applications (ITA 2009), pp. 74–85.

[3] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard,. 2003. Web Services Architecture. W3C Consortium, Feb.

[4] ISO/IEC 25010, 2010. Software Engineering. Software Product Quality Requirements and Evaluation (SQuaRE). Quality Model. FCD Ballot, International Organisation for Standardisation/International Electrotechnical Commission.

[5] F. Losavio, A. Matteo, R. Rahamut, 2008. Web Services Domain Analysis based on Quality Standards, ECSA, pp. 345-358.

[6] B. Jeong, H. Cho, C. Lee, 2009. On the functional quality of service (FQoS) to discover and compose interoperable web services. Expert Systems with Applications 36 (2009), 5411-5418

[7] M. Klusch, A. Leger, D. Martin , M. Paolucci, A. Bernstein, U. Küster, 2009. Performance Evaluation of Semantic Service Matchmakers, 3rd International Semantic Service Selection Contest - October.

[8] D. Andreou, Matchmaker User Guide, 2010. http://ict-alive.svn.sourceforge.net/viewvc/ict-alive/ServiceLevel/alive- matchmaking/trunk/tutorial.html?view=log, IEEE Std 1061, 1992. Standard for a Software Quality Metrics Methodology, IEEE.

[9] S. Ran, A Model for Web Services Discovery with QoS, 2004. SIGEcom Exchanges, vol. 4, no. 1, pp. 1–10.

[10] E.M. Maximilien, M.P. Singh, A framework and ontology for dynamic Web services selection, 2004. Journal of IEEE Internet Computing, IEEE Educational, Activities Department vol. 8 (5) pp. 84–93.
[11] B. Sabata et al. 1997. Taxonomy for QoS Specifications, Workshop on Object-Oriented Real-Time Dependable Systems (WORDS '97), IEEE CS Press.
[12] G. Dobson, R. Lock, I. Sommerville, 2009 Quality of Service Requirements Specification Using an Ontology, SOCCER Workshop, Requirements Engineering, 2005
[13] S. Bleul and T. Weise, 2005. An Ontology for Quality-Aware Service Discovery. QASD 2005
[14] S. Jean, F. Losavio, N. Levy, A. Matteo, 2010. An Extension of OWL-S with Quality Standards, 4th Int. Conference on Research Challenges in Information Science (RCIS 2010), IEEE, pp. 483-494, Nice, May.
[15] V.X. Tran, H. Tsuji, R. Masuda, 2009. A new QoS ontology and its QoS-based ranking algorithm for Web services, Simulation Modeling Practice and Theory Volume 17, Issue 8, pp 1378-1398, September. Dependable Service-Orientated Computing Systems.
[16] T. Rajendran, P. Balasubramanie, Resmi Cherian, 2010. An Efficient WS-QoS Broker Based Architecture for Web Services Selection, International Journal of Computer Applications, vol. 1, no. 9, pp. 75-80.
[17] A.Ouhab, M.Malki, 2009. Découverte personnalisée de Services Web Sémantiques, Proceedings of the 2nd Conférence Internationale sur l'informatique et sesApplications (CHA'09), Vol-547, May 2009
[18] ISO/IEC9126-1. Quality characteristics and guidelines for their use (1, 2). Technical report, International Organisation for Standardisation / International Electrotechnical Commission, 2001.
[19] G. F. Tondello and F. Siqueira, 2008. The QoS-MO ontology for semantic QoS modeling, proceedings of the 2008 ACM Symposium on Applied Computing (SAC 2008), New York, USA, pp. 2336-2340.
[20] ISO/IEC 13236. Quality of Service: Framework. Version 1. (1999). International Organisation for Standardisation / International Electrotechnical Commission.