

Synthetical Evaluation of Multiple Qualities for Service Selection

Xia Wang*, Tomas Vitvar*, Mick Kerrigan[†] and Ioan Toma[†]

*Digital Enterprise Research Institute (DERI), National University of Ireland, Galway, Ireland

[†]Digital Enterprise Research Institute (DERI), Leopold-Franzens Universität Innsbruck, Austria

Email: firstname.lastname@deri.org

Abstract—Automating Service Oriented Architectures by augmenting them with semantics will form the bases of the next generation of computing. Once a set of services that can fulfill the users functional requirements have been discovered it is important to be able to choose the service that best meets the users non-functional requirements. This task is becoming harder as more services are becoming available. The process of selecting the service, that the user will eventually invoke, from the set of discovered services can be performed by a combined evaluation of the quality of each service (QoS). In this paper we propose such a QoS-based approach. Initially we specify a QoS ontology and its vocabulary using the Web Services Modeling Ontology (WSMO) for annotating WSMO Web Service descriptions with QoS data. We continue by defining quality attributes and their respective measurements along with a QoS selection Model. Finally, a fair and dynamic selection mechanism using an optimum normalization algorithm is presented.

I. INTRODUCTION

Web services with well-defined semantics, named Semantic Web Services (SWS), provide interoperability between web services by describing their own capabilities in a computer-interpretable way [14], [15]. The greatest advantage of SWS is that they enable machines to automatically perform complex tasks by manipulating a series of heterogeneous web services based on semantics. Most aspects of SWS, such as automatic discovery, selection, composition, invocation, or monitoring of web services, are tightly related to the quality of these services. QoS as part of the service description is an especially important factor for service selection [8] and composition [25].

For discovery of services, the service requester provides some requirements on the capability of the requested service. There are also many service providers publishing their services by advertising their capabilities semantically. A service discovery engine can be used to match the requirements of the user against the advertised capabilities of the provider services. In the case that there are several similar services yielded by the discovery process, it is necessary to select between these different services based on non-functional properties. Generally the most important of these properties are those related to the quality of the provided service, therefore selection is a combined evaluation of these service qualities.

In the literature, this issue has not been thoroughly addressed, due to the complexity of QoS metrics. Sometimes, the quality of a service is dynamic or unpredictable. Moreover, current most work focuses on the definition of QoS ontology languages, vocabulary or measurements and to a lesser extent

on a uniform evaluation of qualities. In our former work, we defined a selection model for semantic services [16], in which the details of quality-based selection algorithm are specified. This paper will go on to discuss how to base the selection on a combined evaluation of the multiple and diverse qualities of services.

The Web Service Modeling Ontology (WSMO) [17] is a conceptual model for describing Web Services semantically, and defines the four main aspects of semantic web services, namely Ontologies, Web Services, Goals and Mediators. With respect to WSMO only a small amount of work has been done on selection of services has been performed, mainly in [6], which introduces a number of generic selection mechanisms to be used with WSMO. Through this paper we use the WSMO descriptions to describe the QoS model, specific quality metrics, value attributes, and their respective measurements. Furthermore, we propose an algorithm to normalize all the different quality attributes, and to provide a dynamic and fair evaluation of services. This is done by considering the users quality requirements together with the set of quality advertisements provided by the service provider and synthetically evaluate all metrics of closeness in quality attributes by normalization. A weight matrix is applied to obtain the final evaluation.

The paper is structured as follows, Section II provides an overview on the current related work. In Section III, a QoS ontology language designed for the needs of web services is defined in the context of WSMO, and a QoS model for service selection is presented. Our QoS-based service selection algorithm is evaluated Section IV. In Section V experimental results are presented to show the validity of the algorithm.

II. RELATED WORK

Most of the related work in using QoS for service selection focuses on the development of QoS ontology languages and QoS vocabularies, as well as on the identification of various QoS metrics and their measurement with respect to Semantic Web Services. These areas of work aim to foster the automation of web service discovery, selection, and composition. Here, we can distinguish between four directions of research:

- 1) [13] and [7] emphasize the definition of QoS aspects and metrics. For example in [13], all possible quality requirements are enumerated and organized into several categories, including runtime-related (*Scalability*,

Capability, Performance, Reliability, Availability, Robustness/Flexibility, Exception handling, or Accuracy), transaction support related (*Integrity* with respect to the ACID properties), configuration management and cost-related QoS (*Regulatory, Standards supported, Stability/Change cycle, Cost or Completeness*), and security-related QoS (*Authentication, Authorization, Confidentiality, Accountability, Traceability and Auditability, Data encryption, or Non-repudiation*). Also, they shortly present their definitions or possible determinants. Unfortunately, they failed to present quantifiable measurements.

- 2) [11] and [4] focus on the creation of QoS ontology models, which propose QoS ontology frameworks aiming to formally describe arbitrary QoS parameters. In their QoS ontology models, they define in detail many of the attributes of QoS parameters, such as nature (*static* or *dynamic*), aggregation (if a QoS parameter is composed of two or more defined quality parameters), relationship to other QoS parameters, and QoS impact (which represents the way QoS parameter values contribute to the service quality perceived by the user). From their ongoing work we know that they did not consider, yet, QoS-based service matching.
- 3) The work [8], [10], and [9] tried to study real evaluation and proposes QoS-based service selection, while the authors failed to present a fair and effective evaluation algorithm. In [10], computing QoS attributes of web services was discussed: four typical properties were taken as an example, and a travel service flow was analyzed to show the importance of QoS in service selection and composition.

In [9] it was proposed to extend a web service QoS model based on the consideration of a configurable, synthetic fuzzy evaluation system. Also, a QoS requirements' description model was defined in order to improve service selection among functionally equivalent services. Although their QoS model considered to unify the quantitative and qualitative qualities for combined evaluation, they remained abstract and did not present any evaluation mechanism.

The work presented in [8] is similar to ours. They considered the computation of QoS during dynamic selection of services, and tried to build an open, fair, dynamic QoS computation framework to evaluate the QoS of a vast number of web services. There are, however, some differences to our approach:

- The measurement of linguistic-based qualities was not considered.
- The algorithm uses average ranking, neglecting nuances in different quality properties.
- A possible maximum value is used to normalize the QoS matrix, although such kind of value is worth to be deliberated.
- Analyzing the experimental data, after normaliza-

tion the final result looks like

$$G' = \begin{pmatrix} 0.769 & 1.429 & 1.334 & 1.111 \\ 0.946 & 0.571 & 0.666 & 0.889 \end{pmatrix}.$$

For this way of normalisation, it is hard to make a fair evaluation of all qualities, because the metrics do not have the same range, one quality attribute even has a higher weight, while its real impact will be decreased by its smaller value.

Therefore, our approach is to normalize each quality metric into values between 0 and 1 by specifically defined measurements, which is fair to each quality metric. That means, we propose a different normalization algorithm.

- 4) [28] and [29] focused on augmenting QoS classes and properties to extend the DAML-S or OWL-S [9] language profiles. [28] defined a QoS ontology for DAML+OIL using description logic notions to express different QoS templates. [13] incorporated QoS into UDDI and SOAP messages [7] to improve the service discovery process. In this paper, we emphasize the extension of WSMO with a QoS ontology class.

Although QoS ontologies, QoS vocabulary, and their measurements have already been studied extensively, a fair and dynamic algorithm for evaluating QoS attributes in combination is still missing. Possible reasons for this are:

- 1) Multifaceted QoS comprises a set of numerous and complicated quality attributes. Moreover, each quality may have different quality types with wide ranging values. Some qualities give rise to linguistic-based metrics, others to numeric values.
- 2) Diversity: different qualities have different measurement rules. Sometimes the value tendency of a quality is that a smaller value is better, for example the price of a service, and sometimes it is the opposite.
- 3) Difficulty: it is hard to fairly and uniformly evaluate all qualities in combination, and make a final quantified analysis for the QoS of a web service considered.

III. QoS ONTOLOGY LANGUAGE AND VOCABULARY IN WSMO

The Web Service Modeling Ontology (WSMO) [17] is a conceptual model for describing various aspects related to Semantic Web Services. WSMO is made up of four top level elements, namely Ontologies, Web Services, Goals and Mediators. Briefly, *Ontologies* provide the terminology and formal semantics for the other elements in WSMO, *Web Services* define a semantic description of services including their functional and non-functional properties, *Goals* specify the requesters requirements for a Web Service and *Mediators* resolve the heterogeneity problem by implementing ooMediators (between ontologies), ggMediators (between goals), wgMediators (between web services and goals), and wwMediators (between services) [12].

TABLE I
QoS ONTOLOGY IN WSMO

<p>Class QoS sub-Class nonFunctionalProperties hasMetricName type string hasValueType type valueType hasMetricValue type value hasMeasurementUnit type Unit hasValueDefinition type logicalExpression multiplicity = single-valued isDynamic type boolean isOptional type boolean hasTendency type {small, large, given} isGroup type boolean hasWeight type string</p>
--

In WSMO, quality aspects are part of the non-functional information on a Web Service description and are simply defined as: *Accuracy, Availability, Financial, Network-related QoS, Performance, Reliability, Robustness, Scalability, Transactional, Trust*. Such kind of QoS definition is neither expressive nor flexible enough for QoS attributes. Therefore, for the purpose of selecting services, in this paper we introduce a new class, QoS concept classes, that refines the non-functional properties class in WSMO. Further we define a QoS model following the same syntax to extend WSMO. The defined QoS model may be referred to by the Web Service and Goal entities, and quality factors can adequately be considered during the process of service selection.

We will specify a QoS upper ontology named WSMO-QoS. It is a complementary ontology that provides detailed quality aspects of services. Developers benefit from WSMO-QoS for QoS-based matchmaking and QoS measurement.

A. QoS Ontology and Vocabulary

Based on [4], [9] and [11], we define a new class *QoS* (Table I), which is a subclass of (nonFunctionalProperties) class already defined in WSMO. The new class *QoS* can be attached to a *webService* or *Goal* class. Please notice that the current WSMO conceptual model remains unchanged, we just refine the (nonFunctionalProperties) class.

Each QoS metric is generally described by *MetricName*, *ValueType*, *Value* (given or calculated at service run-time), *MeasurementUnits* (e.g. \$, millisecond), *ValueDefinition* (how to calculate the value of this metric), and *Dynamic/Static*. For purposes of QoS-based selection, there are also four features defined, namely, *isOptional*, *hasTendency*, *isGroup*, and *hasWeight*. The following is a simple interpretation for every property of Table I:

- Types of the parameter *valueType* may be *linguistic*, *numeric* (int, float, long), *boolean* (0/1, True/False) or other. Therefore, there will be different forms of preprocessing according to the different value types.
- The property *MetricValue* defines a metric's values, which are either real ones or a string such as '*calculate*'. If *MetricValue* = '*calculate*', then this attribute should

refer to its *valueDefinition* for dynamic value calculation.

- The property *MeasurementUnit* specifies the concrete unit of every quality metric, with possible types such as $Unit = \{\$, millisecond, percentage, kpbs, times, \dots\}$. Also class unit has a conversion function between different measurement units, e.g., to transform second to millisecond.
- Parameter *hasValueDefinition* is either a logical expression defined as in [17] or the string '*NULL*'. If *hasValueDefinition* = '*NULL*', then this value definition cannot explicitly extracted from the context of service description, but must dynamically be invoked from its service provider. In this case this quality attribute must be dynamic, that is *isDynamic* = *true*.
- Through property *isDynamic* the nature of a quality is defined as static or dynamic. For a static quality, its values are given by a priori, and can directly be used during the selection process. If *isDynamic* = *true*, this quality metric must dynamically be invoked and obtained from its service provider, and its values are calculated at runtime.
- If *isOption* = *false*, this attribute, assumed to be noted as q_k , is necessary, and $q_k \in q_R$, where q_R is the required quality set. This property is detailed in Subsection II.B.
- *hasTendency* is an object property representing the expected tendency of the value from the user's perspective. For example, the price of a service is expected to be as low as possible, so that its *hasTendency* = '*low/small*'. On the contrary, the security of a service should be as high as possible, i.e., *hasTendency* = '*high/large*'. When *hasTendency* = '*given*', the user expects the value of this quality to be as close the given value as possible. Also, in a quality inquiry, *hasTendency* = {*low/small, high/large, given*} denotes, respectively, that { $\geq, \leq, =$ } for its *MetricValue*.
- *isGroup* indicates if this quality attribute is defined by a group of other qualities or not. For example, *security* is composed of *nonRepudiator, DataEncryption, Authorisation, Authentication, Auditability, and Confidentiality* [4]. Hence, *isGroup* = *true* means that in the pre-processing stage the group value must be calculated first.
- Finally, *hasWeight* is a value denoting the weightiness of the property, especially when synthetically measuring several metrics. In this context we define the weight value either ranges in [0, 10] or '*NULL*', different end users have different weight values for their service requirements. Note, in this paper this property only is used by WSMO goal, which describes user's desire; In the description of web service, its value is '*NULL*'.

During the selection process when parsing a QoS profile, to obtain a metric's value for which *hasMetricValue* = '*calculate*' holds, its *hasValueDefinition* property needs to be checked to determine how to calculate it. If *hasValueDefinition* = '*NULL*' and *isDynamic* = 1,

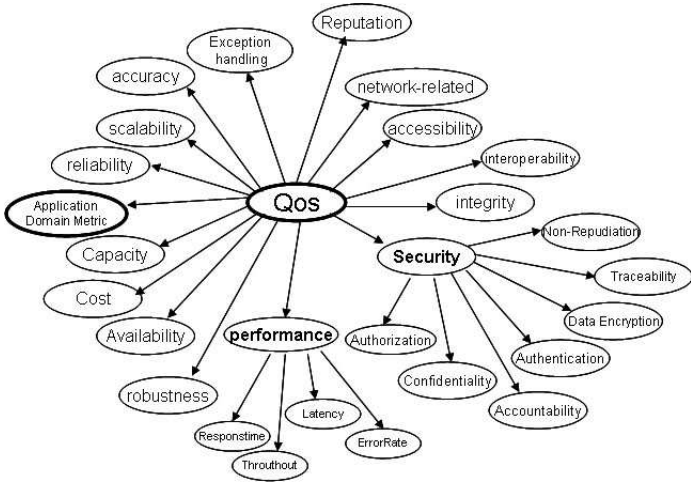


Fig. 1. QoS ontology and vocabulary

then the invocation function is to inquire the real-time value; otherwise, an error is encountered. If $isDynamic = 0$, its corresponding $hasMetricValue$ is an existing value, or there is an error, again.

In [1], [7], and [13] all the possible QoS requirements for web services were defined, mainly including performance, reliability, scalability, capacity, robustness, exception handling, accuracy, integrity, accessibility, availability, inter-operability, security ($isGroup$), and network-related QoS requirements. Fig. 1 gives a simple view on QoS vocabulary, which consists of many general QoS attributes and a scalable domain-specific QoS subset used, e.g., to define the hotel category for a hotel service. The definition and the discussion of concrete measurement of qualities is out of this paper's scope.

B. QoS Selection Model

The scenario of QoS-based service selection is described as follows. The user provides his requirements (including non-functional, functional, and quality properties) for the expected service, which are formed into a requirement profile, noted as $s_R = (NF_R, F_R, Q_R, C_R)$, where the denotations are the identifiers of Non-Functionality, Functionality, Quality and Cost (the details of such selection model is in [22]). On the other side, there are thousands of available services published in either a service repository or a kind of peer-to-peer service environment. The advertisement of a service s is denoted as $s_A = (NF_A, F_A, Q_A, C_A)$, similarly.

The first filter of service selection matches s_R with any available s_A on the basis of non-functional (NF , basically only the service name and service category) and functional (F , including inputs, outputs, preconditions and effects) features of services. We assume that m similar services are yielded, namely, $S = \{s_1, s_2, \dots, s_m\}$, $m \in \mathcal{N}$.

The second filter synthetically considers all quality features to select the service among S satisfying the user's requirements best. This matchmaking takes place between the pair of the QoS requirements Q_R and a quality profile Q_A of a candidates service $s_A \in S$, as illustrated in Fig. 2.

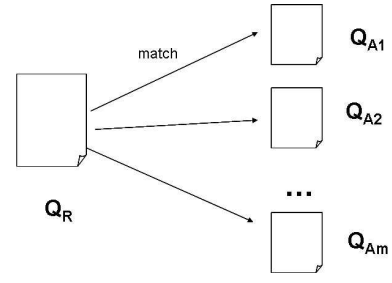


Fig. 2. QoS-based selection of services

For the purpose of matching, a QoS selection model is defined, in which metrics are defined both from the perspectives of users and providers of web services. We assume that $Q = \{q_1, q_2, \dots, q_i\}$, $i \in \mathcal{N}$, and Q_I denotes the quality set. Then

- Q_N is the necessary quality set for each service defaulted by machine, and $Q_N \subseteq Q_I$;
- Q_O is the optional quality set of the service defined as $Q_O = Q_I \setminus Q_N$; and
- Q_D is the default quality set of the service. When user does not explicitly give any quality requirements, i.e., when $Q_R = \emptyset$ and $Q_D \subseteq Q_I$, then Q_D will be taken as Q_R , i.e., $Q_R = Q_D$, where Q_R are the user's quality requirements. Generally, $Q_N \subseteq Q_D$.

There are two reasons for distinguishing between different QoS sets. One is to free the customer from multifarious definitions of his quality requirements, which sometimes need professional knowledge. For example, a customer cannot understand the meaning of *availability* of a service, but he apparently has a requirement for it. So, the customer may only provide qualities basing on his personal opinions, whereas the complementary part is left to be defined in the default quality set. The other reason is for the simple, high effective QoS-based approach for service selection.

Basing on the above analysis, there are three kinds selection modes with respect to Q_R :

- 1) Default mode. When $Q_R \neq \emptyset$, Q_R is redefined as union of the original user requirements and the default ones about service performance, as $Q_R := Q_R \cup Q_N$.
- 2) Totally based on the user's requirements, and $Q_R \neq \emptyset$.
- 3) Totally based on default definitions, if $Q_R = \emptyset$.

Further, for purposes of efficient and flexible service selection and from the user's perspective, in our model only several qualities are defined in the necessary set, viz., $Q_N = \{cost, responseTime, reliability, accuracy, security, reputation\}$, and similarly $Q_D = \{cost, responseTime, reliability, accuracy, security, reputation, executionTime, exceptionHandling\}$. While the definitions are extendable and changeable for specific application system.

There are many approaches to collect values of quality metrics:

- Directly from the service descriptions, e.g., sometimes the price of invoking a service is given a priori.

- Simple calculation of a quality value basing on the defining expression in the service description.
- Collection through active monitoring, e.g., execution duration defined in [8].
- Dynamical inquiry from the current server.
- Periodical update of quality values for statistical purposes in a log.
- Obtaining requester's feedback on quality characteristics, e.g., *Reputation* [8] of a service.

Not only the collection of quality requirements is dynamic, unpredictable, and even difficult during run-time. Also, the value characteristics of quality metric can be approximately concluded as:

- 1) Numerical metric, denoted by a number but with different value ranges.
- 2) Ordinal and linguistic-based metric, denoted by a term from an ordered finite collection of terms, e.g., the reputation of a service may be evaluated by $\{Low, Very\text{-}low, Medium, Very\text{-}high, High\}$.
- 3) Regional metric, denoted by a numerical region $[min, max]$.
- 4) Graded metric, e.g., rank of a hotel service in $\{1, 2, 3, 4, 5\}$.
- 5) Boolean value numeric or enumerative scales.

It is worth to be noticed that this QoS model is easy to be extended or customised. The user may customise his Q_N, Q_D, Q_I at will. The detailed definition of all quality attributes is out of this paper's scope. Instead, we mainly focus on the QoS foundation of the selection model, and the combined evaluation of the quality attributes.

IV. SELECTION ALGORITHM

QoS-based selection of services is very complex, not only due to the diversity of multifarious quality metrics with different value types, value range, and measurements, but also since an effective algorithm is missing, which evaluates all metrics in combination.

We assume that $Q_R = \{r_1, r_2, \dots, r_k\}$ expresses the profile of a user's quality requirements, which includes k quality metrics. Similarly, the quality profile of m candidate services in set S is denoted as $Q_S = \{Q_{A_1}, Q_{A_2}, \dots, Q_{A_m}\}$, where $Q_{A_i} = \{q_{i1}, q_{i2}, \dots, q_{ij}\}$, $i, j \in \mathcal{N}$. It defines that the advertisement of service S_i has j quality metrics provided.

It is well-known that there are two cases during the match-making,

- 1) $Q_R = \emptyset$, then $Q_R := Q_D$;
- 2) $Q_R \neq \emptyset$, then $Q_R := Q_R \cup Q_N$. The Q_R is then matched with each Q_{A_i} , $i \in \mathcal{N}$.

As it is obviously rather unlikely for any Q_R or Q_{A_i} to happen to have the same number of quality metrics. So, in the first preprocessing step, we take Q_R as benchmark for alignment with every Q_{A_i} . This process includes:

- 1) To re-arrange the metrics of Q_{A_i} in the same order.
- 2) If Q_{A_i} is lacking a quality, then add a metric and set its value to 0.

- 3) To tailor the qualities which are not listed in Q_R .

Therefore, the matrix of QoS for service matchmaking $M_Q = \{Q_R, Q_{A_1}, Q_{A_2}, \dots, Q_{A_m}\}$ looks like:

$$M_Q = \begin{pmatrix} r_1 & r_2 & r_3 & \dots & r_k \\ q_{11} & q_{12} & q_{13} & \dots & q_{1k} \\ q_{21} & q_{22} & q_{23} & \dots & q_{2k} \\ \dots & \dots & \dots & \dots & \dots \\ q_{m1} & q_{m2} & q_{m3} & \dots & q_{mk} \end{pmatrix}_{(m+1) \times k}$$

Here, M_Q is a $(m + 1) \times k$ matrix, with the quality requirements Q_R in the first row, and the quality information of candidates services in the other ones. Each column contains values of the same quality metric. For uniformity, matrix M_Q has to be normalized with the objective to map all real values to a relatively small range, i.e., the elements of the final matrix are real numbers in the closed interval $[0, 1]$. The main idea of the algorithm is to scale the value ranges with the maximum and minimum values of each quality metric for thousands of current candidate services. Accordingly, the maximum and minimum values are mapped to the uniform values 1 and 0, respectively, totally depended on their definition of *hasTendency*.

For instance, a user searches a flight constraining the ticket price to be below \$300, and three service providers asks for \$250, \$280, and \$260, respectively. In this case the minimum and maximum are \$250 and \$280. Then, the calculation of relative closeness for this quality metric reads as $(1 - \frac{250-250}{280-250}) = 1$, $(1 - \frac{280-250}{280-250}) = 0$, and $(1 - \frac{260-250}{280-250}) = 0.667$.

The second preprocessing step is uniformity analysis. We distinguish different quality metrics with their value features. In our QoS model, we take the information of *hasTendency* of a quality metric $r_i, i \in k$:

- 1) if *hasTendency* = 'given', then we calculate the ratio by

$$q'_{ij} = \begin{cases} 1 - \frac{q_{max} - q_{ij}}{q_{max} - q_{min}} & \text{if } r_j \geq q_{max} \\ \frac{q_{ij} - q_{min}}{q_{max} - q_{min}} & \text{if } r_j \leq q_{min} \\ 1 - (|\frac{q_{ij} - r_j| - m}{n - m}|) & \text{if } r_j \in (q_{min}, q_{max}) \end{cases} \quad (1)$$

- 2) if *hasTendency* = 'small/low', then the ratio is calculated by

$$q'_{ij} = (1 - \frac{q_{ij} - q_{min}}{q_{max} - q_{min}}) \quad (2)$$

- 3) if *hasTendency* = 'large/high', then the ratio is calculated by

$$q'_{ij} = (1 - \frac{q_{max} - q_{ij}}{q_{max} - q_{min}}) \quad (3)$$

where $q_{max} = \max\{q_{ij}\}$, $q_{min} = \min\{q_{ij}\}$, and $n = \max\{|q_{ij} - r_{ij}|\}$, $m = \min\{|q_{ij} - r_{ij}|\}$, and $i \in k, j \in m$. In Fig. 3, where left to right on the scale line corresponds to

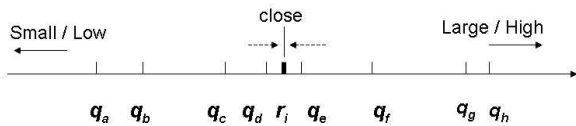


Fig. 3. Quality measurement

growing values, three cases of matchmaking are shown, whose tendency is small/low, close, and large/high, respectively. Also the value of r_j , $j \in k$ is scattered either among q_{ij} , $i \in m$ or right side or left side of the candidates values. Formula 1-3 present their algorithm.

Just take the Formula 1 as an example, it describes the case that user require the value of one quality as close to his given value as possible. We assume r_j with its value as u_j and the other quality $\{q_a, q_b, \dots, q_h\}$ with their value as $\{v_a, v_b, \dots, v_h\}$. There are also three cases in Formula 1. First, when $u_j \geq q_{max}$, just as the candidate set is $\{q_a, q_b, q_d, q_d\}$, then by Formula 1 we know q_d gets the best ratio as 1. Similar case happens when $u_j \leq q_{min}$. While when r_j scatters in $\{q_c, q_d, q_e, q_f\}$, the range of scale should be first defined by $(n - m)$, then ratios are calculated following the third case of Formula 1.

The weighted value for each quality metric is defined in the parameter of *hasWeight*. These are brought into the form of a diagonal matrix as $W = \{w_1, w_1, \dots, w_k\}$. Here, we assume that $\sum_{i=1}^n w_i = 10$ (which is not defined as 1, for the reason of magnifying the effect of experiments). Then, W is applied to matrix M_Q yielding

$$M_{Q'} = M_Q \times W = \sum_{i=1}^m (q'_{ij} \times w_i) \quad (4)$$

Finally, we can calculate the evaluation result for each quality metric by summing the values of each row. These abstract values are taken as relative evaluation of each service's QoS.

V. EXPERIMENTS

For reasons of comparison and simplification we borrowed test data from [8]. In their experiments, they implemented a hypothetical phone service (UPS) registry, which provides various phone services such as long distance, local, wireless, and broadband. They simulated 600 users to collect the experimental data. Especially, two phone services' test data are presented with seven quality criteria, including *Price*, *Transaction*, *Time Out*, *Compensation Rate*, *Penalty Rate*, *Execution Duration*, and *Reputation*. Their corresponding value types are \$, 0/1, microsecond, percent, percent, microsecond, and rank value in $[0, 5]$.

In order to be applied into our selection mode, we have to assume a requirement of a service customer and another two services for testing, then the M_Q is as Table.II.

The first row is the supposed Q_R , the next two rows are taken from [8], and the last two ones are also hypothetical candidates services. From the definitions of each quality criterion of that example we know that *Price* and *Execution Duration*

TABLE II
EXPERIMENT DATA

Data	Pri	Trans	TimeOut	ComRat	PenRat	Execu	Repu
R	30	1	80	0.4	0.8	120	4.0
ABC	25	1	60	0.5	0.5	100	2.0
BTT	40	1	200	0.8	0.1	40	2.5
A_1	28	1	140	0.2	0.8	200	3.0
A_2	55	1	180	0.6	0.4	170	4.0

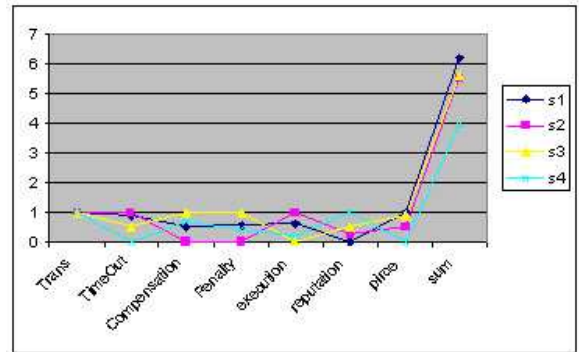


Fig. 4. Combined evaluation of qualities

are expected to be smaller, *Compensation Rate*, *Penalty Rate*, and *Reputation* are to be bigger, and *Time Out* is required to be as close as possible. The result of normalization carried out by our algorithm for the four candidate services referring to Q_R is:

$$Q' = \begin{pmatrix} 1 & 1 & 0.870 & 0.500 & 0.571 & 0.625 & 0 \\ 0.500 & 1 & 1 & 1 & 0 & 1 & 0.250 \\ 0.900 & 1 & 0.522 & 0 & 1 & 0 & 0.500 \\ 0 & 1 & 0 & 0.667 & 0.429 & 0.188 & 1 \end{pmatrix}$$

Assuming $W = \{4, 0, 0, 2, 1, 1, 2\}$, we apply Formula 4 to obtain a quality evaluation set, named $Q'' = \{6.196, 5.500, 5.600, 3.951\}$. That is, in case of putting a high weight on price, service s_1 is the best choice, the order of the results is in line with human intuition, see Fig. 4, and the result is consistent with [8], too.

Here a short discussion is presented. Our Qos-based model is dynamic and real-time, which is fully adapted to the current fantasticality network environment, and which keeps a well relativity and up-to-date, it is also fair on this point. Because it is always basing on the current available services to compare their current integrative capability. If services are added or deleted, the evaluation should be updated.

Also in a certain relatively stable service environment, a service provider may consider to change one of its property, it is easy to forecast its constraint for value. For instance, we take the service s_1 (the service ABC in Table. II) as an example to analysis the effect of the price on its QoS. From Fig. 5, we knew that if its price is go beyond the current maximum, it will lost its competition on price and keep invariable QoS value.

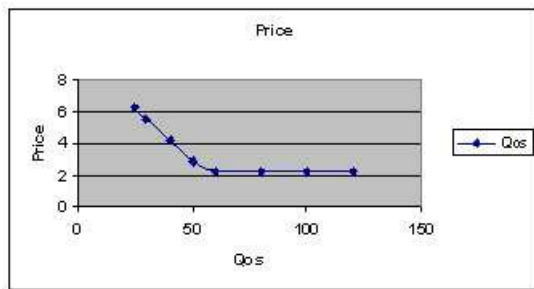


Fig. 5. Combined evaluation of qualities

VI. CONCLUSION

This paper proposed a QoS-based approach for web service selection, by presenting a fair and simple algorithm evaluating multiple quality metrics in combination. First, we specified a QoS ontology and its vocabulary in order to augment the QoS information in WSMO. Then, various quality attributes, their respective measurements, and a QoS selection model were defined in detail. Finally, a fair and dynamic selection mechanism was presented, which uses a normalisation algorithm oriented at value range optima. The approach was validated by a case study for a kind of phone service.

ACKNOWLEDGMENT

The authors would like to thank ...

REFERENCES

- [1] Anbazhagan, M. and Nagarajan, A., Understanding quality of service for Web services. IBM Developerworks website, <http://www-106.ibm.com/developerworks/library/ws-quality.html>, 2002.
- [2] DAML-S Coalition, DAML-S: Web Service Description for the Semantic Web. In Proc. International Semantic Web Conference (ISWC 02), 2002.
- [3] V. Deora et al, Incorporating QoS specifications in service discovery. In Proceedings of WISE Workshops, Lecture Notes of Springer Verlag, 2004.
- [4] Tsesmetzis, D.T., Roussaki, I.G., Papaioannou, I.V. and Anagnostou, M.E., QoS awareness support in Web-Service semantics, AICT-ICIW06, 128–128.
- [5] U. Keller, R. Lara, A. Pollers, I. Toma, M. Kifer and D. Fensel, D5.1v0.1 WSMO Web Service Discovery, Technical report, DERI Innsbruck, 2005.
- [6] M. Kerrigan, Web Service Selection Mechanisms in the Web Service Execution Environment (WSMX). In Proceedings of the 21st Annual ACM Symposium on Applied Computing (SAC), Apr 2006, Dijon, France.
- [7] K.Lee, J. Jeon, W. Lee, S. Jeong and S. Park, QoS for Web Services: Requirements and Possible Approaches, W3C Working Group Note 25, 2003.
- [8] Liu, Y., A.H.H. Ngu and Zeng, L., QoS Computation and Policing in Dynamic Web Service Selection. Proc. 13th Intl. Conf. World Wide Web, 2004.
- [9] Mou, Y., Cao, J., Zhang, S.S., Zhang, J.H., Interactive Web Service Choice-Making Based on Extended QoS Model, CIT 2005: 1130–1134.
- [10] Menasce, D.A., QoS Issues in Web Services. IEEE Internet Computing, 6(6), 2002.
- [11] Papaioannou, I.V., Tsesmetzis, D.T., Roussaki, I.G., Miltiades, E.A., QoS Ontology Language for Web-Services. AINA2006.
- [12] A. Polleres, R. Lara (eds.): A Conceptual Comparison between WSMO and OWL-S, WSMO Working Group working draft, 2005, <http://www.wsmo.org/2004/d4/d4.1/v0.1/>.
- [13] Ran, S.P., A Model for Web Services Discovery with QoS. SIGecom Exchange, 4(1):1–10, 2003.
- [14] McIlraith, S., Son, T.C. and Zeng, H. Semantic Web Services, IEEE Intelligent Systems, Special Issue on the Semantic Web, 16(2):46–53, 2001.
- [15] McIlraith, S. and Martin, D. Bringing Semantics to Web Services, IEEE Intelligent Systems, 18(1):90–93, 2003.
- [16] Wang, X., Zhao, Y., Kraemer, B.K. and Wolfgan, H., Representation and Discovery of Intelligent E-Services. In *E-Service Intelligence – Methodologies, Technologies and Applications (chapter 10)*, Lu, J., Ruan, D., and Zhang, G. (Eds.) 2006.
- [17] Roman, D., Lausen, H., and Keller, U., D2v1.1. Web Service Modeling Ontology (WSMO), WSMO Final Draft 10 February 2005.
- [18] Herrera-Viedma, E. and Peis, E., Evaluating the informative quality of documents in SGML format from judgements by means of fuzzy linguistic techniques based on computing with words, Information Processing & Management 39(2):233–249, 2003.
- [19] Herrera-Viedma, E., Peis, E., Olvera, M.D., Montero, Y.H. and Herrera, J.C., Evaluating the informative quality of Web sites by Fuzzy Computing with Words, In: E. Menasalvas et al. (eds.), Proc. Atlantic Web Intelligence Conference 2003, LNAI 2663, (Springer-Verlag, Berlin Heidelberg, 2003).
- [20] Herrera-Viedma, E., Pasi, G., Lopez-Herrera, A.G., Porcel, C., Evaluating the Information Quality of Web Sites: A Methodology Based on Fuzzy Computing with Words, Journal of American Society for Information Science and Technology, in press 2006.
- [21] Weikum, G., Towards guaranteed quality and dependability of information service, In 8th GI Fachtagung: Datenbanksysteme in Büro, Technik und Wissenschaft, 1999.
- [22] Wang, X., Zhao, Y. and Wolfgan, H., Selection Model of Semantic Web Service, 7th International FLINS Conference on Applied Artificial Intelligence, 2006.
- [23] Mich, L., Franch, M. and Gaio, L., Evaluating and designing Web site quality, IEEE Multimedia January-March 2003, 34–43.
- [24] Olsina, L. and Rossi, G., Measuring Web application quality with WebQEM, IEEE Multimedia October-December 2002, 20–29.
- [25] Zeng, L.Z., Benatallah, B., Anne H. H. Ngu, Dumas, M., Kalaganam, J. and Chang, H., QoS-Aware Middleware for Web Services Composition, IEEE Trans. Software Eng. 30(5):311–327, 2004.
- [26] Sumra, R., and Arulazi, D., Quality of Service for Web Services-Demystification, Limitations, and Best Practices, 2003.
- [27] Mani, A. and Nagarajan, A., Understanding Quality of Service for Web Services, IBM Developerworks, 2002.
- [28] Zhou, C., Chin, L.T. and Lee, B.S., DAML-QoS Ontology for Web Services. In International Conference on Web Services (ICWS 2004), pp.472–479, 2004.
- [29] Zhou, C., Chia, L.T., and Lee, B.S., Semantics in Service Discovery and QoS Measurement, IT Professional, 7(2):29–34, 2005.