A Method for Automated Web Service Selection

Hong Qing Yu and Stephan Reiff-Marganiec
Department of Computer Science
University of Leicester, UK
{hqy1,srm13}@mcs.le.ac.uk

Abstract

Automated Web service selection is a key challenge in Service Oriented Architecture (SOA) research. With SOA becoming more popular in industry, and more Web services becoming available in the e-business market, the main issue changes from service discovery to service selection and ranking. In this paper, we propose a novel nonfunctional property-based service selection method by modifying the Logic Scoring Preference (LSP) method with Ordered Weighted Averaging (OWA) Operators. Moreover, the dynamic mechanism for evaluating metadata based QoS criteria of each service is presented for this method.

1. Introduction

Web services are a technology designed to support interoperable machine to machine software interactions. The execution of services takes place on remote system(s) hosting the requested service(s). The typical architecture of Web services includes three roles, namely service requestor, service broker and service provider. Once the requestor sends a service request to the broker, a matched service provider should be sent back as a WSDL URL by the broker searched from a UDDI or published service metadata repository. Finally the requestor can invoke the provided service through the SOAP message protocol. One of the crucial issues is the automated service ranking for dynamic service selection environment [1]. When the broker finds a set of services which all satisfy the functional requirements of a service request, then the non-functional requirements can be used as service ranking criteria.

Consequently, the problem can be recast into an automated multicriteria decision problem for multiple entities. The solution to this problem requires a selection method such that each single criterion can be evaluated and multiple criteria can be properly aggregated. The final quantitative results should reflect the best suitable service based on the request. The main challenges discussing in this paper are:

(1) Automated QoS-based service ranking mechanism: most of the selection methods strongly rely on human intervention. For example, criteria used for selection are typical tightly bound to a priori defined evaluation mappings; changing preferences and criteria requests for manually retuning the evaluation mapping. Meanwhile, the changes of criteria and preferences are unavoidable in the dynamic environment. Predefining the service selection criteria and preferences can not reflect the evaluation of the system at run-time.

(2) Dynamic retuning of aggregation functions: most aggregation functions cannot show relations between individual criterions. Currently, the aggregation functions used for service ranking are based on the simple operations such as arithmetic metrics (essentially weighted sums) or geometric metric (essentially weighted products. However, these functions are not enough to represent logic relations such as simultaneity and replaceability.

To explain the concepts of simultaneity and replaceability, let us consider an example. A car buyer may say that they **simultaneously** need good performance and a good price with performance being more important (weight is 0.7) than price (0.3). If there are two cars evaluated as (0.9, 0.0) and (0.6, 0.5), then the first car (0.7 0.9 + 0.3 0.0 = 0.63) is better than second car (0.7 0.6 + 0.3 0.5 = 0.57) by using the traditional weighted sum aggregation function. However, this does not reflect the simultaneity relation of "performance and price **both** are important criteria". The correct aggregation function should penalize the first car choice. On the other hand, people can argue that the geometric function can do this. However, if the requirements change to **replaceability**, which means that the criteria (performance and price) can replace each other (e.g. a lower performance is acceptable if the price is good). As result, the correct aggregation function should only penalize the car if both of the evaluation scores are 0. In this situation the weighted product is not the correct aggregation function. In a more realistic scenario, we may have more complex relations on the range between simultaneity and replaceability.

In the literature, the Logic Scoring Preference (LSP) method is introduced to solve the second problem by adding a parameter to retune the aggregation function [2], which is defined by domain experts for each situation. However, we need to define a way to dynamically obtain the value of this parameter. The parameter is influenced by the combination of simultaneity (conjunction) and replaceability (disjunction) referred to as *degree* or *orness degree* in LSP. Meanwhile, the Ordered Weighted

Averaging (OWA) operators introduced in [3] can be proven to reason the *orness degree* on the fly in the fuzzy logic research area.

The main contributions of this paper are (1) a novel multicriteria-based automated Web service selection method and a ranking approach and (2) a type-based evaluation mapping system to address the issue of automated service evaluations. The method includes a multi-criteria aggregation function which combines LSP metrics with OWA Operators.

The rest of the paper is organized as follows. Section 2 introduces the background information and techniques we use in this paper. Firstly, the inContext dynamic service selection platform is explained to show the service ranking requirements. Secondly, the Logic Scoring Preference method and Ordered Weighted Averaging operators are introduced. Then, the relation between LSP and OWA will be discussed. Section 3 illustrates a novel way of applying OWA concepts to deal with problems occurring when automating LSP. Section 4 discusses the technique for dynamically selecting evaluation methods. Based on these new techniques, Section 5 proposes the new LSP method for automated Web service selection. Section 6 illustrates the implementation. Section 7 will discuss some related work and Section 8 draws the conclusion and future work.

2. Background

2.1 "inContext" Research Project

The complexity of business processes and the dynamic nature of the co-operations make it difficult for the business modeler to select appropriate services, manage the compositions efficiently and understand requirements within a dynamic context correctly. The "inContext" project [4] platform provides dynamic context information which affects the Web service measurement at run-time; Figure 1 shows an overview of the architecture. The most interesting part of this platform for this paper is the user context based service selection process (the core logic of which is encapsulated in the Relevance Engine). On the one hand, the platform has the ability to supply the current context status of the end-users such as "location", "availability", "devices", "task" and "budget" (the details of users' context model can be found in [5]). Thus, the criteria for measuring Web service can be gained by reasoning on the context data. On the other hand, the platform also dynamically stores the QoS metadata of registered services and allows updating this at any time. We are involved in this project to design the QoS-aware Web service selection mechanism. Thus, this paper will only concentrate on discussing the service selection part of the whole framework. However, it is worthwhile noting here that the user context data is gathered during the execution of the system automatically by mining past activities and monitoring current events, hence (and crucially) not providing an extra burden on the user. The service meta data is provided by the service provider and is then augmented with monitored run-time data.

The platform makes use of weights in the ranking of criteria; these are encapsulated in a user profile which is seen as part of the context (it can be changed dynamically); however this is where the user needs to specify some data at least once to express their wishes.

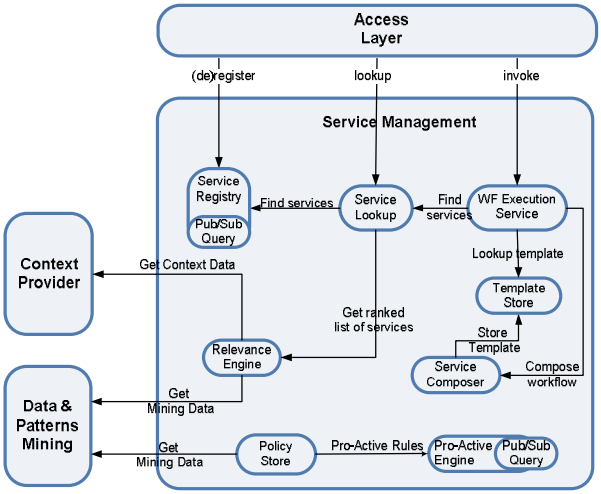

Fig. 1 The inContext platform

2.2 The modified LSP method

Logic Scoring Preference (LSP) introduced in [2, 6] is one professional evaluation method initially designed for solving hardware selection problems [7]. LSP extends the traditional scoring techniques [8] and at its core is an evaluation function (1). The traditional scoring techniques only consider the semantics of importance but ignore the relation between criteria such as replaceability, simultaneity and mandatory-ness, which normally are observable properties of human reasoning [9] – we gave an insight in the car selection example in the introduction. To capture these relations, Conjunction/Disjuction (GCD) operators were introduced in LSP to represent the relation of requirements.

$$L = (|\omega_1|E_1^r + |\omega_2|E_2^r + \cdots + |\omega_n|E_n^r)^{1/r} \qquad (1)$$

where $0 \le E_i \le 1, \sum_{i=1}^{n}|\omega_i| = 1$, and ω is the weight of each criterion. $\omega < 0$ means that a lower evaluation result is better (e.g. for cost), $\omega > 0$ means a higher result is preferable (e.g. for speed). E is the defined evaluation

function for providing the scores of the service for each criterion. The *r* presents the logic relation between different criteria. The complete table of GCD operators with accepted symbols, degrees and values for parameter *r* are given in Table 1 [7]. The definition of the GCD operators is the disjunction/conjunction degree *d* (also known as *orness degree*) of combining simultaneity and replaceability. The idea here is that at the ends of the spectrum we have conjunction (and) and disjunction (or) respectively. Values in the range are closer to or, and hence have a higher degree of 'orness' – that is they behave more like or (we could provide a similar definition for 'andness'). We will use these defined operators without adding more new operators. We will use the closest values of *d* when a calculated degree value result is not found in Table 1. Moreover, *r2* is the *r* value for a 2 criteria situation, *r3* for 3 criteria situation and so on. The *d* indicates the orness degree as explained above.

Operation	Symbol	d	r2	r3	r4	r5
DISJUNCTION	D	1.0000	+infty	+infty	+infty	+infty
STRONG QD (+)	D++	0.9375	20.630	24.300	27.110	30.090
STRONG QD	D+	0.8750	9.521	11.095	12.270	13.235
STRONG QD (-)	D+-	0.8125	5.802	6.675	7.316	7.819
MEDIUM QD	DA	0.7500	3.929	4.450	4.825	5.111
WEAK QD (+)	D-+	0.6875	2.792	3.101	3.318	3.479
WEAK QD	D-	0.6250	2.018	2.187	2.302	2.384
SQUARE MEAN	SQU	0.6232	2.000			
WEAK QD (-)	D--	0.5625	1.449	1.519	1.565	1.596
ARITHMETIC MEAN	A	0.5000	1.000	1.000	1.000	1.000
WEAK QC (-)	C--	0.4375	0.619	0.573	0.546	0.526
WEAK QC	C-	0.3750	0.261	0.192	0.153	0.129
GEOMETRIC MEAN	GEO	0.3333	0.000			
WEAK QC (+)	C-+	0.3125	-0.148	-0.208	-0.235	-0.251
MEDIUM QC	CA	0.2500	-0.720	-0.732	-0.721	-0.707
HARMONIC MEAN	HAR	0.2274	-1.000			
STRONG QC (-)	C+-	0.1875	-1.655	-1.550	-1.455	-1.380
STRONG QC	C+	0.1250	-3.510	-3.114	-2.823	-2.606
STRONG QC (+)	C++	0.0625	-9.060	-7.639	-6.689	-6.013
CONJUNCTION	C	0.0000	-infty	-infty	-infty	-infty

Table 1: GCD operators, orness degree and parameter *r*

For this paper, we only use up to 5 criteria and hence the table is only shown to that depth. Meanwhile, Fodor and Marichal independently propose the LSP definition as formula to measure the orness of any mean operator *M* by studying Dujmovic's degree of conjunction and disjunction [10, 11]:

$$OrnessD\big(M(x)\big) = \frac{\int_0^n M(x)dx - \int_n^0 Min(x)dx}{\int_0^n Max(x)dx - \int_0^n Min(x)dx} \qquad (2)$$

where M(x) is a GCD operator, Max(x) is the pure conjunction (C) and Min(x) is the pure disjunction (D). However, this definition does not solve the problem of dynamically calculating the orness degree without knowing the selected GCD operator, it is more a replacement for the lookup table (Table 1).

LSP aims to evaluate quantitative features for the comparison of different entities. Recently, LSP has been used to deal with Hardware/Software systems, Data Management systems and web site evaluation and selection problems [7, 12, 13]. The four main steps of LSP evaluation method are: (1) specifying evaluation variables, (2) defining elementary criteria, (3) analyzing degree decision and (4) analyzing cost/preference. Note that these are steps involving much human decision making. The LSP method certainly has lots of power to evaluate the quantitative aspects of competitive Web services and support selection decision making. However, when consumers' preferences are changing because of their different context status, the LSP method does not support fully automatic service selection methodology. The first problem is that the orness degree changes when the preferences (weights and criteria) are changed in the unpredictable environment. Therefore, we cannot predefine the orness in the evaluation formula. The second major problem is identifying criteria evaluation methods dynamically. In the static environment, the evaluation methods can be predefined for the specified criteria and the mapping relations are also static for the finite criteria. However, if we have a large set of criteria, then the evaluation methods are very difficult to design and to be mapped. We start to discuss the solution of the first issue in section 3 and the second issue in section 4. Before these discussions, we will introduce an important definition used in this paper.

2.3 Ordered Weighted Averaging Operators

Ordered Weighted Averaging operators are introduced in the area of fuzzy logic [3], and have been shown to allow to determine the orness degree on the fly. The Ordered Weighted Averaging (OWA) operators are defined as follows:

Definition: *OWA operator*
Let W=(w_1, w_2, ..., w_n) with $\sum_{i=0}^{n} w_i = 1$.
Let A=(a_1, a_2, ...,a_n) and B=(b_1, b_2, ...,b_n) be bags where b_i is the i-th largest element of A.
An OWA operator of dimension n is a mapping

$$F: \mathbb{R}^n \to \mathbb{R}$$

such that

$$F(a_1, a_2, \ldots, a_n) = \sum_{i=1}^{n} w_i\, b_i$$

For example, W = (0.4, 0.3, 0.2, 0.1), then F (0.7, 1, 0.3, 0.6)=(0.4)(1)+(0.3)(0.7)+(0.2)(0.6)+(0.1)(0.3)=0.76.

A fundamental aspect of this operator is the re-ordering step. An aggregate a_i is not associated with a particular weight w_i but rather a weight is associated with a particular ordered position of the aggregate [14]. In [3],

the orness measure of any kind of OWA operator is defined as follows:

$$Orness(OWA) = \frac{1}{n-1}\sum_{j=1}^{n}(n-j)w_j \quad (3)$$

Also, a link between orness as defined for Fuzzy Logic (OWA) and orness as defined by formula (2) has been shown in [15]:

Theorem:
If the problem can be expressed as and OWA problem, then:
OrnessD(M(x)) = Orness(OWA)

This theorem gives a tool for linking the LSP method and OWA operators, by bridging the gap between the discrete orness definition of formula (2) and the contributions of formula (3) as we will discuss in the next section.

3. Applying OWA Orness Measure to LSP Method

On the one hand, we argued that the original LSP method does not support an automatic mechanism to select a suitable value for *r* for the aggregation function. On the other hand, [15] proved that if the ranking problem *can be* transformed into an OWA problem, then we can use the OWA orness definition to automatically calculate the LSP orness degree. Therefore, we need to show that the ranking problem can be transformed into an OWA problem, which we will do now.

In general an OWA problem is characterized by being expressible by two bags: one with weights and one with evaluation values. The latter comes in two forms: *A* and *B*, where A is ordered in the same order as the weights and B in order of the value of the elements.

Considering the services ranking problem we naturally have a bag of weights, but we have multiple bags of values as we have one set of criteria evaluation values. From this we can conclude that we can computer an orness degree per service, which is of course not what we want as we are looking for an overall aggregator which should differentiate the services.

The solution is to compute a bag of values that contains the average score for each criteria across all services evaluated.

Assume that we are considering m services and n evaluation criteria. Furthermore, $W = \{w_1, w_2, \dots, w_n\}$ is a bag of weights and $V_1 = \{v_{11}, v_{21}, \dots, v_{n1}\}$ … $V_n = \{v_{1m}, v_{2m}, \dots, v_{nm}\}$ are the evaluation results of each mapped criteria for the different services, with each vector presenting results for the *n* criteria of one service. Then:

$$V = \left\{\frac{\sum_{i=1}^{m} v_{1i}}{m}, \frac{\sum_{i=1}^{m} v_{2i}}{m}, \dots, \frac{\sum_{i=1}^{m} v_{ni}}{m}\right\} \quad (4)$$

is the set of average evaluation scores for our *n* criteria.

Using W and V as the respective sets, we have recast the service selection problem into an OWA problem, and hence can use Orness(OWA) to compute our orness degree, which in turn allows us to extract the value r to be used in the global aggregation function (1).

We will now show some very small examples that show the computed orness degree and provide insight about the operator expected for correct aggregation.

Criterion	C1	C2	C3
Weight	0.7	0.2	0.1
Service1	0.3	0.2	0.1
Service2	0.1	0.7	0.9

Example 1

Recall that when calculating overall scores, we do not want the evaluation results of less import criteria to outweigh the important ones; we referred to this as simultaneity earlier on. In Example 1 we can see that both services have low evaluation values for the most important criterion C1 – which means that to ensure simultaneity we require a conjunction LSP operator. By using formula (4) and (3), we calculate the orness degree as 0.2 which maps to the strong quasi conjunction operator.

	w1=0.7	w2=0.2	W3=0.1
Service1	0.9	0.2	0.1
Service2	0.7	0.7	0.9

Example 2

Example 2 shows a typical case of replacebility, where a good match on an important criterion is making a service preferable. In the example, we can see that both services have higher evaluation results for the most import criterion. Recall that for replacebility we would expect a disjunction operator. Again, by applying our formulae we can compute the orness degree (this time as 0.75) and we find that this maps to medium quasi disjunction.

4. Type-based Evaluation Mapping Methods

Most of the traditional criteria selection methods strongly rely on human intervention. For example, criteria used for selection are typical tightly bound to a priori defined evaluation mapping; changing criteria require manually retuning the evaluation mapping. Meanwhile, changes of criteria and preferences are unavoidable in the dynamic environment encountered in web service selection. Thus, predefining the service selection criteria and preferences cannot reflect the evaluation requirements of the system at run-time. We propose a type-based evaluation mapping method which is related to the types of the criteria, rather

than being dependent on the criteria themselves. We define three types of criteria and each type has a related evaluation method. The three types are "Numerical type", "Boolean type" and "Set overlap type". Further types could of course be defined if required, but we have not encountered a need for them so far.

The **Numerical type** is used for criteria which take numerical input to the evaluation method such as cost, time and measurement values. The mapped evaluation method is given by (5), where w is the weight of the criterion. When the criterion is of numerical type, the weight can be in the range [-1…0]. In this case, it means that a smaller numerical value is desired (as e.g. for price properties).

v_{max} is the maximum value of all competitive services for the criterion, v is the value for the service under evaluation. v_{min} is the minimum value of all competitive services.

$$E = \begin{cases} 1 - \left(\frac{v_{max} - v}{v_{max} - v_{min}}\right) & \text{iff } w \geq 0 \\ \left(\frac{v_{max} - v}{v_{max} - v_{min}}\right) & \text{otherwise} \end{cases} \quad (5)$$

The **Boolean type** is used for criteria which have a value that is evaluated to 1 or 0. The method is:

$$E = \begin{cases} 1 & \text{if the criteria is met} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

The **Set overlap type** is to define criteria which are measured by the size of the evaluation objects' satisfaction subset:

$$E = (e_1 + e_2 + \cdots + e_n)/n \quad (7)$$

with e_i being a score for each element of the set.

Now, whenever the criteria are changing or values are being updating at runtime, the system automatically applies the correct evaluation functions based on the detected criteria's type.

5. The Ranking Approach

We will now present the modified LSP method used for quantitative aspects of automatic Web service ranking. This method includes 5 steps developed from [16, 17].

I. Service users specify the weights to a set of desired criteria.
II. System obtains the user's context and sets the evaluation values to the criteria.
III. System obtains the metadata of each competitive service related to the desired criteria.
IV. Individual criteria for all services are evaluated to values in the [0, 1] interval by applying the type-based measurement formulae (5), (6) and (7).
V. The criteria are re-ordered according to their combined weights. The value for parameter r is automatically gained by calculating the orness using formulae (3) and (4).
VI. The overall score for each service is calculated using formula (1) and all services are ranked by their aggregated evaluation scores.

Note that step I is manual: the user's preferences must be captured. Typically users would do this once as part of their user profile which is then using data (in step II) adapted based on the context. Further, the weighted desired criterion set is a subset of criteria which are defined inside a particular service category. The criteria can be divided into two groups: hard criteria (the weight of the criterion is 1) and soft criteria (the weight of the criterion is in the range]-1, 1[ ).

Also note that in step III metadata not only describes the service's functional properties such as input, output, precondition and effect (IOPE), but also indicates non-functional properties, such as cost, speed, security and so on. The Semantic Web service concept is designed to organize the service metadata. The Web Ontology Language (OWL) [18] is one published ontology standard for constructing Semantic Web services. Moreover, the tools for retrieving information from the OWL standard are developed based on SPARQL techniques [19]. These tools can be used for obtaining and reasoning about the specified metadata. The metadata is inserted by service providers in their service description, and is then augmented with runtime information.

6. Prototype Implementation

We developed a prototype to demonstrate the proposed nonfunctional property-based service selection approach [20]. Service providers can register their services with different operations through an online interface (depicted in Figure 2). This essentially allows the registration of a service in a repository and does not diverge much from current practice with web services: it requires a WSDL file and a service description.

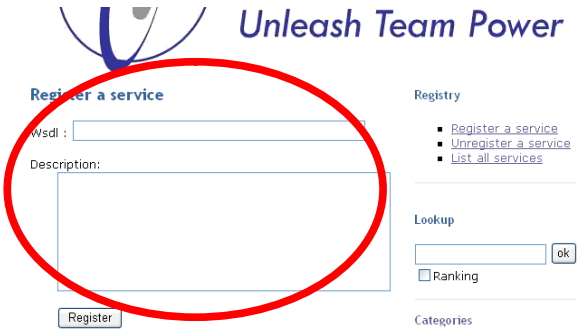

Fig. 2 Register and service search page

Once a service operation has been registered, the metadata can be edited or new data can be entered (see Figure 3).

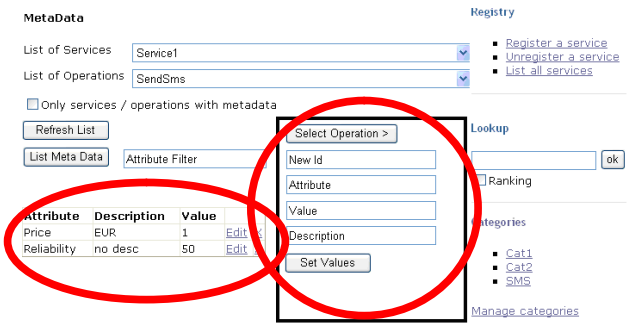

Fig. 3 Metadata management page

Service operations are assigned categories, which have an influence on which metadata attributes are expected: e.g. a printing service might have metadata to express print queue length, while a communications service might have metadata regarding online status of a user. All services might have metadata such as price. Service providers can create new categories (Figure 4). The evaluation mechanism uses the service categories when deciding which criteria to consider in the ranking. When searching, users (or workflow engines, etc) provide key words to describe the functionality that they are after.

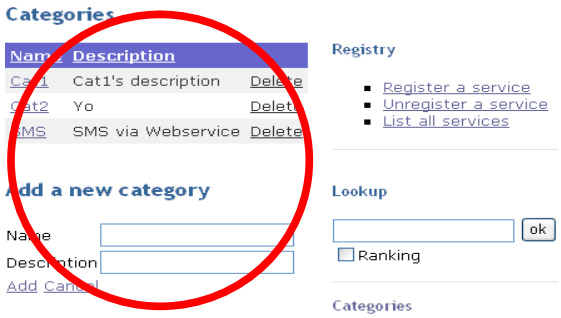

Fig. 4 Category management page

The lookup in conjunction with the relevance engine will search the best suitable service based on user's context (we will not illustrate how users register their context information to the system). If this is done through the interface shown in Figure 5, users would tick the "ranking option" box. For instance, the user may search for "SMS" services, then a ranked or unranked (depending on tick box) list of service will be displayed and the most suitable service is at the top of the list.

Fig. 5 Service Selection result

The search with ranking is of course somewhat slower when compared to an unranked search. We have not evaluated the details as in the current setting with only small numbers of services of a specific category available the difference is nearly not noticeable. More general, it is worth pointing out a few facts that allow for a somewhat slower mechanism to be satisfactorily in this context:

- Services often tend to be long-running units of work, possibly involving human interaction themselves. So when considering execution time of business services we are thinking in timeframes of hours and days – hence whether suggestions for services take seconds or a few minutes to appear is generally not an issue.
- Compared to manually pouring over service descriptions and then making a decision as to which service to use, we feel it is easy to argue that any automated method is of benefit.
- Finally, previous work in social science and telecommunications service selection has shown that users are generally happy to wait a little for a service to be selected if it guarantees better satisfaction.

Saying this, it is of course important that the mechanism runs reasonably fast, which we believe to be the case as much of the evaluation is calculation of relatively simple formulae and so far we have not seen much delay in the distributed setting in which we use the method.

7. Related Work

Currently Web service measurements consider using QoS as non-functional properties. They focus on defining QoS ontology languages and vocabularies, and identifying various QoS metrics and their measurements with respect to semantic services. Some of the work only concentrates on defining ontologies. For example, [21] and [22] propose QoS ontology frameworks aimed at formally describing QoS attributes. To our understanding, these works have defined neither the matched metrics for different non-functional properties nor schemes to

quantify the attributes. Meanwhile [23] enumerates a large number of non-functional properties and organizes them into several categories, such as run-time related, transaction support related, configuration management, cost-related QoS, and security-related QoS. However, the work assumes that all measured values are available somewhere. Thus, it fails to illustrate the quantifiable measurements.

There is other related work in [24], [25], and [26] attempting to conduct a detailed evaluation and proposing QoS-based service selection. However, they do not explicit how to define the desired QoS properties and where they come from. In terms of automated Web service selection, all the current work has two major disadvantages.

(1) They do not define the proper quantitative aggregation metrics for obtaining the final evaluation scores. The final score usually calculated by using only the average of all individual evaluation values of the criteria or conducted by weighted arithmetic mean method.

(2) Most of them are human oriented processes. They do not provide an automatic mechanism to facilitate the dynamic metric invocation and aggregation.

Our work addresses both of these shortcomings by presenting an automated process where the human intervention involved is restricted to providing the categorization of services and their metadata as well as obtaining user preferences. The automated method uses a run-time evaluation framework to score services taking into account delicate relations between weights and values.

8. Conclusion and Future Work

This paper proposed a novel method for automatically measuring and selecting Web services in a dynamic environment. The theory has been implemented as part of the "inContext" project platform. The method is implemented by combining the LSP metrics and OWA operators for calculating aggregate scores for each service and capturing criteria based scores based on a type-based evaluation mapping system. For the former, we apply OWA operators to automatically measure the value of the orness degree required by LSP. For the latter, we define three types of evaluation functions which are selected dynamically based on the type of the criteria to be evaluated. Comparing to other Web service selection methods, our method (1) can capture the high level satisfaction semantics in terms of fuzzy relations between different criteria (e.g. simultaneity and replaceability) and (2) requires minimal human intervention. Once service metadata is available and the selection criteria are defined (these are based on the categories of the operations, so the overhead is minimal), the ranking scores for individual service are computed completely automatically.

We have evaluated a first version prototype within the inContext project. One limitation of this method is that it only ranks services in single service selection scenario (local service ranking). In ongoing research, we are extending the method be used in service composition workflows (global ranking) where a service should be selected within the context of other preceeding and following services.

A further avenue of future work is to explore mechanisms to capture user requirements in more user friendly and flexible ways that would supercede the current user profiles.

9. Acknowledgement

This work is partially supported by the EU inContext (Interaction and Context Based Technologies for Collaborative Teams) project: IST-2006-034718. The authors would like to thank the project partners for stimulating discussion. Particular thanks are due to Marcel Tilly and Sebastien Peray at the European Microsoft Innovation Centre for their collaboration on the implementation and Dr. Emilio Tuosto at the University of Leicester for his comments.

10. References

[1] W3C Working Group, "Web Services Architecture", http://www.w3.org/.

[2] Dujmovic, J.J. Mixed Averaging by Levels (MAL) –A System and Computer Evaluation Method. In Proceedings of the Informatica Conference, Bled, Yugoslavia, 1973.

[3] Yager, R.R. On Ordered Weighted Averaging Aggregation Operators in Multi-criteria Decision Making, IEEE Transactions on Systems, Man and Cybernetics, vol. 18, pp. 183-190, 1988.

[4] inContext project: Unleash Team Power, http://www.in-context.eu, 2007.

[5] inContext project WP2 Deliverable 2.1: Analysis and specification and context modeling techniques, http://www.in-context.eu/page.asp?PageRef=10. 2007.

[6] Dujmovic, J.J. Extended Continuous Logic and the Theory of Complex Criteria. In Journal of the University of Belgrade, EE Dept., Series Mathematics and Physics, No. 537, pp. 97-216, 1975.

[7] Dujmovic, J.J. A Method for Evaluation and Selection of Complex Hardware and Software Systems. In Proceedings of 22nd International Conference for the

Resource Management and Performance Evaluation of Enterprise Computer Systems. New Jersey, 1996.

[8] Miller III, J.R. Professional Decision-Making, Praeger Publishers, 1970.

[9] Dujmoviæ, J.J. and Larsen, H. Properties and Modeling of Partial Conjunction/Disjunction. In B. De Baets et al (eds): Proceedings of Eurofuse Workshop on Data and Knowledge Engineering, published as Current Issues in Data and Knowledge Engineering. pp. 215-224, Exit Publishers, Warsaw (2004).

[10] Fodor, J.C. and Roubens, M. Modelling and Multicriteria Decision Support. Kluwer, Dordrecht, 1994.

[11] Marichal, J.L. Aggregation operations for multicriteria decision aid. PhD. Thesis, Institute of Mathematics, University of Liege, Belgium, 1998.

[12] Su, S.Y.W., Dujmovic, J.J., Batory, D.S., Navathe, S.B and Elnicki, R. A Cost-Benefit Decision Model: Analysis, Comparison, and Selection of Data Management Systems. ACM Transactions on Database Systems, 12(3), pp. 472-520, 1987.

[13] Olsina, L. A. and Rossi, G. Measuring Web application quality with WebQEM, IEEE Multimedia, 9(4), pp. 20-29, 2002.

[14] Carlsson, C. and Fuller, R. OWA operators for decision Support. In Proceedings of EUFIT'97 Conference, vol. II, pp. 1539-1544, 1997.

[15] Fenandez Salido, J.M. and Murakami, S. Extending Yager's orness concept for the OWA aggregators to other mean Operators. Fuzzy Sets and Systems. Elsevier B.V. vol. 139, pp. 515-542, 2003.

[16] Yu, H.Q. and Molina, H. A Modified LSP method for services evaluation and selection. In S. Gorton, M. Solanki and S. Reiff-Marganiec (eds): Proceedings of the 2nd European Young Researchers Workshop on Service Oriented Computing, pp. 87-93, 2007.

[17] Reiff-Marganiec, S., Yu, H.Q. and Tilly, M. Service Selection based on Non-Functional Properties, In Proceedings of Non Functional Properties and Service Level Agreements in Service Oriented Computing Workshop NFPSLA-SOC'07. 2007.

[18] W3C Working Group, "OWL Web Ontology Language Overview", W3C Recommendation, http://www.w3.org/TR/owl-features/.

[19] W3C RDF Working Group, "SPARQL syntax and specification", http://www.w3.org/TR/rdf-sparql-query/, 2006.

[20] inContext Project, *Service Lookup Engine,* http://213.239.218.49/RegistryManager, 2008.

[21] Papaioannou, I.V., Tsesmetzis, D.T., Roussaki, I.G. and Miltiades, E.A., A QoS Ontology Language for Web-Services. In 20th International Conference on Advanced Information Networking and Applications - vol 1 (AINA'06), pp. 101-106. 2006.

[22] Tsesmetzis, D.T., Roussaki, I.G., Papaioannou, I.V., and Anagnostou, M.E. QoS awareness support in Web-Service semantics. In Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06), pp. 128-128. 2006.

[23] Ran, S.P., A Model for Web Services Discovery with QoS. ACM SIGecom Exchanges, 4(1), pp.1-10, 2003.

[24] Liu, Y., Ngu, A.H.H. and Zeng, L., QoS Computation and Policing in Dynamic Web Service Selection. In Proceeding 13th International Conference, World Wide Web, pp. 66-73, 2004.

[25] Mou, Y., Cao, J., Zhang, S.S. and Zhang, J.H., "Interactive Web Service Choice-Making Based on Extended QoS Model", CIT (2005), pp.1130-1134.

[26] Menasce, D.A., QoS Issues in Web Services, IEEE Internet Computing, 6(6), pp.72-75, 2002.

[27] Canfora, G., PentaRaffaele, M.D., Esposito, R. and Villani, M.L, An approach for QoS-aware service composition based on genetic algorithms. In Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, pp. 1069-1075, 2005.

[28] Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J. and Chang, H., QoS-aware middleware for web services composition. IEEE Transactions on Software Engineering, 30(5), 2005.