A Method to evaluate QoS of Web Services required by a workflow

Narayan Debnath
Department of Computer Science
Winona State University, MN 55987 – USA
ndebnath@winona.edu

Paola Martellotto, Marcela Daniele
Departamento de Computación
Universidad Nacional de Río Cuarto
Río Cuarto – Argentina
{paola|marcela}@dc.exa.unrc.edu.ar

Daniel Riesco,Germán Montejano
Departamento de Informática
Universidad Nacional de San Luis
San Luis – Argentina
{driesco|gmonte}@unsl.edu.ar

***Abstract*. The Web services technology for invoking external applications from a workflow engine provides significant benefits to the WFMS. With the increasing number of web services that provide similar functionality, it is important to find the best web service that meets the user's needs, including both their functional and non functional requirements. The non-functional description of the service requires specifying at run time, the quality attributes that can influence the choice of a web service offered by a supplier. In this sense, it is essential to use metrics to evaluate the quality characteristics of Web services (QoS) in order to filter out Web services discovered, and obtain the most suitable one. The dynamic behavior of Web services regarding the development of new services and constantly changing existing ones, requires a continuous evaluation process, leading to capture web service information with respect to their quality and performance evaluation as requested by the workflow. This paper proposes a quantitative measurement method that allows evaluating QoS of web services, and provides a value that allows the comparison and selection that best meets the required characteristics.**

Keywords: ***Web Services, Workflow, quantitative method, non functional requirement.***

I. INTRODUCTION

One of the concepts of the last decade that has become important in the world of application development is the web services (WS). The WS essentially provides a standard means of communication between different software applications, giving the ability to access different services in a unified and interoperable way over the Internet [8].

This standard means of communication between processes and/or components is designed to be of multiplatform and multilingual type, i.e., no matter what language a WS is programmed in, or what platform it is running on, it will be accessible and usable for other applications developed in other platforms or programming languages.

In previous work [5] and [6], the authors of this paper have focused on the use of WS in the context of Workflow Management Systems (WFMS), and the need to invoke external applications in certain situations. The module that implements the invocation of applications from WFMS represents the existing software on which the workflow can be used for certain activities, taking into account that, in principle, this software can be found on any platform or location of the web. Today, this communication with external applications is static, which leads to various problems: any event occurring in the external applications will be transferred to the workflow at the time of its invocation, such as the unavailability of the application either temporary or not, or an update of it that results in a change of location, and even a change in response format which was originally understood by the Workflow.

On the other hand, the increasing number of WS available within an organization and on the web, raises a problem of searching them, and highlights the important need to find the best WS that meets certain requirements of the requester. For this reason, the requirements of a WS should not focus only on its functional properties, but also on describing the environment in which it develops, i.e., describing non-functional characteristics (QoS).

Each service can offer various options for quality characteristics based on technical requirements such as availability, performance and scalability, security and privacy policies, etc., all of which must be described and analyzed. At this point, it is vitally important to have a procedure to assess the quality of the WS. This procedure will provide a tool to estimate the quality of the WS in a specific way.

In consequence of the above so far, this paper proposes a procedure for measuring the quality of the WS available on the web, to simplify the invocation of external applications to a WfMS. The proposal is evaluated through a case study whose results were favorable in terms of their applicability.

This paper is organized as follows: Section 2 presents the state of the art about evaluating and measuring the quality of services. Section 3 includes a description of the desirable quality features in WS. Section 4 presents a general framework in the selection of services from a WfMS and the QoS specification of WS. Section 5 proposes a method to evaluate the QoS of WS. Section 6 describes the case study. Finally, we present the conclusions and references.

II. State of the Art

There are various studies related to the quality of WS and ways of measuring it.

In [11], a unified ontology was presented for QoS of WS, which applies to the three main scenarios: selection of WS's based on QoS monitoring (verification compliance with the offered QoS) and control (adapting the service in case of non-compliance of the offered QoS) based on QoS. To describe the WS, they proposed expanding the QoSModel class, according to OWL-S approach to describe services, incorporating three elements: ServiceProfiles, ServiceModel and ServiceGrounding. The proposal shows a formalization of the generic model using ontologies that model non-trivial relations between quality factors. In [12], an estimation model was proposed to assess the quality of a WS, using the standard evaluation criteria (ISO / IEC 9126), applied to the specific case of WS. The research proposes an adaptation of the Systemic Quality Model (MOSCA), so that it is applicable to new technology that runs business: the WS. The main contribution of this model is that of an already tested one (MOSCA), which in turn is based on quality standards and proposed metrics, and then to incorporate the specific aspects of the quality of WS and QoS. The article [9] proposes a framework for the automation of the Web service contract specification and establishment. An extensible model for defining both domain-dependent and domain-independent WS QoS dimensions and a method for the automation of the contract establishment phase are proposed. They describe a matchmaking algorithm for the ranking of functionally equivalent services, which orders services on the basis of their ability to fulfill the service requester requirements, while maintaining the price below a specified budget. Other interesting proposals were analyzed based on the LSP method [13]. This method provides a means for the development of complex functions using continuous preference logic. The same applies to complex software systems and computer and communications equipment, although not specifically to WS. The functions can be efficiently used for the evaluation, comparison and selection of such systems and hardware. Proposals based on LSP [14] [16] [17] [18] presented extensions of this method. They want to adapt LSP to the evaluation and dynamic selection of WS. Also, in [15], the authors propose adding a new layer to InContext platform (Interaction and Context Based Technologies for Collaborative Teams), to select appropriate services, manage the compositions efficiently and understand requirements within a dynamic context correctly.

III. Quality Characteristics Of Web Services

The description of a WS has two main components: their functional and nonfunctional features [7]. The functional description is a syntactic description that focuses on its functional aspects, detailing the operational characteristics that define the general behavior of a WS.

An important requirement for SOA-based applications is to operate reliably and deliver consistent service to a variety of levels. Therefore, the requirements of a WS should not focus only on its functional properties, but also on describing the environment that hosts it, i.e., describing non-functional capabilities or QoS of the service. Each service can offer various options for non-functional features based on the technical requirements resulting from the demand for availability, performance and scalability, security and privacy policies, etc., which must all be described.

The non-functional description defines the QoS of the service, and force the requester to specify, at run time, the quality attributes that can influence the choice of a web service offered by a supplier.

TABLE I. QoS of Web Services

QoS	Definition	
Performance	Speed to complete a service requirement is measured in terms of throughput, response time, latency, execution time and transaction time.	
	Throughput	Number of web service requests served within a period of time
	Response time	Expected time between the moment a request is sent and a response is received
	Latency	Time between the request for a service to start up and the requirement to be addressed.
	Execution Time	time a web service is required to process a sequence of activities
	Transaction Time	The time that elapses while the web service is completing a transaction
Availability	Probability that the web service is available	
Reliability	Ability of a service to operate correctly and provide the same quality despite the flaws in the system or network	
Robustness	Degree to which a service reacts properly to abnormal situations, such as invalid or incomplete entries	

IV. General framework on selection of WS from a WfMS

The description and automation of business processes are performed through the Workflow Management Systems (WfMS). The Workflow Reference Model, developed by the Workflow Management Coalition (WfMC) [1] defines a generic framework for building WfMS, allowing interoperability between them and other applications involved. This model defines one interface for the invocation of external applications [2]. To achieve the goal of optimizing workflow communication with external applications, in [5] and [6] the specification of this interface with WS is proposed.

To select the most suitable application among several semantically correct that behave the same but have different syntactic description and QoS, it is proposed to optimize the selection incorporating the QoS. The proposed scheme for WS invocation presented initially in [6] shows the WS offered by providers through UDDI [4], WS required in the invocation of external applications, corresponding to the workflow engine and

WS selection will eventually be invoked. The workflow engine considers two important elements to make the contrast process among the services offered and the required application:

- History of Previous Runs: the workflow engine performs the invocation of WS, monitors their quality attributes and records in the history of previous executions.
- Quality Characteristics of the WfMS engine: the workflow engine has a record of minimum quality characteristics required to invoke an application.

In the realization of contrast process at the QoS level, the workflow engine considers:

- If the WS has been previously invoked, the workflow engine searches the history of previous executions and checks according to information stored therein.
- If the WS has not been invoked before, it analyzes the characteristics during the verification process. It applies metrics for evaluating quality attributes.

When the contrast process is performed at the QoS level, it is a filter of the preliminary list of WS candidates and gets the WS that best meets the requirements of the requester.

In the proposed scheme, the main component is the WfMS engine which acts as an agent for service discovery, based on functional requirements and QoSs specified by the user and the minimum requirements set by the engine itself. Thus, the engine provides a ranking of WS available and chooses the most appropriate one. It is also responsible for collecting and processing information from the WS offered by suppliers, and keeping it updated as the calls are made.

To invoke a WS, the WfMS engine specifies the functional and nonfunctional WS requester in a WSDL document. Fig. 1 shows the non-functional specifications of QoS of the WS to invoke external applications within the WfMS. QoS is defined as a new element in the types section of WSDL document that specifies the required WS (more details in [6]).

The tType data type represents an enumerated data type specifying the possible measurement units of QoS (numeric, milliseconds, percentage, etc.). The QoS priority element represents the user-defined priority for the QoS considered.

```
<xs:element name="QoSReq" type="xs: tQoSReq"/>
 <xs:complexType  name="tQoSReq">
    <xs:sequence>
       <xs:element name="QoSname" type="xs: string"/>
       <xs:element name="QoSunitType" type="xs: tType"/>1
       <xs:element name="QoSminValue" type="xs: float"/>
       <xs:element name="QoSmaxValue" type="xs: float"/>
       <xs:element name="QoSpriority" type="xs: integer"/>
    </xs:sequence>
 </xs:complexType>
```

Figure 1. Non-functional specification of web service.

V. METHOD FOR SELECTING A WEB SERVICE CONSIDERING ITS QOS

We proposed a measurement method for obtaining a quantitative value of each service and for allowing the comparison between them, making it easy for the workflow engine to dynamically invoke external applications required for the return of an expected result.

First, we define a measurement unit which depends on the characteristic being evaluated. Here we considered:

- Time Unit: assigned a value in milliseconds, seconds, minutes or hours, depending on the characteristics being evaluated.
- Percentage: assigned a numeric value between 0 and 100, which is taken from a representative sample, and which depends on the property being assessed.

Table 2 shows the measurement unit defined for each of the quality characteristics considered in this work. It also defines an identifier for each QoS and works with that identifier in the definition of the function that calculates the score achieved for each service, defined below.

TABLE II. MEASUREMENT UNITS FOR THE QOS OF THE WEB SERVICES

QoS	Measurement Unit	Identifier/Value
Performance	Time Unit	P
Response Time	Time Unit	RT
Latency	Time Unit	L
Execution Time	Time Unit	ET
Transaction Time	Time Unit	TT
Availability	Percentage	A
Reliability	Percentage	R
Robustness	Percentage	Rs

In addition, it assigns a weight to each QoS according to its measurement unit which is derived from a scale defined in Tables 3 and 4.

A. Defining the weight assignment considering the Measurement Unit

To assign a weight to measurement units of Time and Percentage, a scale is defined with the following possible values [0, 0.25, 0.5, 0.75, 1], and each measurement unit is assigned a weight according to that scale. As was indicated in Table 2, the QoS Performance (P), Response Time (RT), Latency (L) Execution Time (ET) and Transaction Time (TT) are evaluated considering the measurement unit of time (Time Unit). Table 3 shows the values to be assigned to each of these characteristics, considering that the time units can be microseconds, milliseconds, seconds, etc. Then, depending on the quality characteristic involved and the service that is to be evaluated, it will be necessary to define the desirable maximum and minimum time, and intermediate times for each QoS. This definition allows changing these settings as needed.

TABLE III. ASSIGNED VALUE TO QUALITY CHARACTERISTICS EVALUATED WITH TIME UNITS

Measurement Unit Time	Value by scale

Tmax: maximum time for the evaluated QoS	0
Tmaxm: defined as Tmax > Tmaxm > Tmed	0.25
Tmed: medium time defined by Tmed = (Tmin + Tmax) / 2	0.5
Tminm: defined as Tmed > Tminm > Tmin	0.75
Tmin: minimum time for the evaluated QoS	1

The Availability (A), the Reliability (R) and the Robustness (Rs) characteristics are measured in percentage and its value is obtained from Table 4.

TABLE IV. ASSIGNED VALUE TO QUALITY CHARACTERISTICS EVALUATED IN PERCENTAGE

D= Measurement Percent	Value by scale
90%	0
between 90.01% and 95%	0.25
between 95.01% and 97%	0.5
between 97.01% and 99%	0.75
between 99.01% and 100%	1

For example, in the case of Availability quality characteristic, a sample consisting of 20 executions of the web service (EWS) is considered, 16 of which result successful. It determines the percentage of service availability (PA), considering the number of times that the web service is available (NTA). An algorithm is defined to calculate the value of NTA, whereas for each execution it takes into account that if the service is available then the value of NTA is assigned in value one. The availability percentage PA is calculated as follows PA = (NTA / EWS) * 100. Finally, we define a priority of preference for each quality characteristic. Table 5 shows the priority values that can be defined by the requester for service.

TABLE V. VALUE ASSIGNED TO THE QOS ACCORDING TO DEFINED PRIORITY

Priority	Value
Not evaluate	0
Low	1
Medium	2
High	3

The function with priority (Fp) assigned to each characteristic is defined by (1).

$$Fp = R*Rp + TR*TRp + L*Lp + TE*TEp + TT*TTp + D*Dp + F*Fpr + Rz*Rzp \qquad (1)$$

It is easy to visualize that if the variable has a high weight but the same was stated by the requester of service as of low priority, then the impact of the value of that variable in the final value will be less than or similar to another one that has a high priority but has not achieved a high weight. Similarly, you may want to evaluate only some of the QoS provided in this method, in which case it assigns zero (not evaluated) in the priority of these features, therefore it shall not affect the total score for each service. It may also happen that the service requester is only interested in the weight obtained for each QoS, simply set all the variables with the same priority.

VI. DESCRIPTION OF CASE STUDY

As a case study for this work, we chose to implement the search and selection of applications such as Weather Service, as part of activities related to the cultivation of peanuts, which in many cases the weather must be checked to determine feasibility of the task. Briefly, the main activities during the peanut planting are:

1. Preparation of the land for planting
   a. using a heavy tool called paratill
   b. done in August, with less rain than other weather conditions
   c. working 80 hectares per day
2. Weed spraying
   a. done with a tool called fallow
   b. can not be done if there is wind or rain
3. Planting
   a. starts in October
   b. The soil temperature should be between 18 and 20 °C
   c. The soil should be moist but there should not be mud
   d. Can not be done if it is raining, if the temperature is below 20 °C, or if it is dry
4. Spraying with pre-emergence
5. Spraying insecticide if there are any pests (spider mites)
6. Spraying every 24 days with fungicides
7. Peanuts start to ripen
   a. Begins in March / April
   b. there must not be rain or mud
8. Peanut harvest after 4 days of torn there must not be rain or mud.

The workflow that implements the process of planting peanuts needs to invoke WS to check the weather. The workflow engine performs the search of available WS on the Internet. The results obtained showed more than 70 WS, and about 25% are discussed in this paper.

While there are numerous WS distributed on the Internet, the most various kinds are grouped by different attributes such as the supplier or the location of the application, it gets to pick the Weather Service by the simplicity in the requirements specification and clarity to visualize and analyze the results. The workflow engine specifies these requirements in the WSDL document that defines the WS request and starts the search. As final result, the motor yields the set of WS available on the web.

In this paper we chose to implement as a case study the selection of the best WS in terms of response time and availability. To use a WS of Weather Service, the user must provide the same requirements, both functional and QoS, which will be recorded in the WSDL document. Figures 2 and 3 show the QoS specification of response time and availability. The quality analysis results are detailed in Tables 6 and 7, respectively. Table 6 shows the WS found to perform consulting services available on the Weather Service Website. The shaded rows correspond to the WS that meets the minimum requirements requested by the user of the workflow. Excluded from analysis, WS unshaded rows whose QoS does not meet these minimum requirements, or could not be calculated due to throwing an error when invoked. RT column shows the response time of the WS. WEIGHT column shows the weight given to each RT, according to Table 2. Finally, the PRIORITY column shows the priority assigned by the user to the QoS Response Time, as specified in Fig. 2.

```
< QoSname> response time < /QoSname>
< QoSunitType/> milliseconds < /QoSunitType>
< QoSminValue/> 2 < /QoSminValue>
< QoSmaxValue/> 5 < /QoSmaxValue>
< QoSpriority/> 1 < /QoSpriority>
```

Figure 2. Specifying of Response Time QoS.

TABLE VI. ANALYSIS OF RESPONSE TIME QOS

NAME	ACCESS-POINT	RT	Weig.	Pri.
Weather	http://ws.cdyne.com/WeatherWS/Weather.asmx	2119.1818	1	1
Weather Service	http://netpub.cstudies.ubc.ca/dotnet/WebServices/WeatherService.asmx	2376.6000	0.75	1
Weather Forecast	http://www.webservicex.net/WeatherForecast.asmx	4491.5000	0	1
Global Weather	http://www.webservicex.net/globalweather.asmx	7323.5714		
USWeather	http://www.webservicex.com/usweather.asmx?WSDL	6451.2451		
Weather	http://www.riptrails.com/webservices/Weather.asmx?WSDL	-		
DASWorld Weather	http://das.ae/webservices/WorldWeather.asmx?WSDL	-		
Weather	http://www.4599.cc/service/weather.asmx?WSDL	-		
WeatherService	http://lostsprings.com/weather/WeatherService.asmx?WSDL	-		
WeatherBug WebServices	http://api.wxbug.net/weatherservice.asmx?wsdl	-		
WeatherService	http://asyncpostback.com/WeatherService.asmx?WSDL	-		
Service1	http://www.tempe.gov/wx/default.asmx?wsdl	-		
DataCenter Class	http://datacenter.c-map.no/datacenterx/datacenter.asmx?WSDL	-		
wsModule	http://nofiasco.com/WS/wsModule.asmx?WSDL	-		
LivedoorWeatherHacks	http://bearmini.net/LivedoorWeatherHacks.asmx?WSDL	-		
WSMenus	http://www.ziggs.com/ws/WSMenus.asmx?WSDL	-		
netForce	http://www.dotnetforce.com/netforce.asmx?WSDL	-		

```
< QoSname> availability < /QoSname>
< QoSunitType/> percentage < /QoSunitType>
< QoSminValue/> 95 < /QoSminValue>
< QoSmaxValue/> 99 < /QoSmaxValue>
< QoSpriority/> 2 < /QoSpriority>
```

Figure 3. Specifying of availability QoS.

TABLE VII. ANALYSIS OF AVAILABILITY QOS

NAME	ACCESS-POINT	A	Weight	Pri.
Weather	http://ws.cdyne.com/WeatherWS/Weather.asmx	0.95	0.25	2
Weather Service	http://netpub.cstudies.ubc.ca/dotnet/WebServices/WeatherService.asmx	0.96	0.5	2
Weather Forecast	http://www.webservicex.net/WeatherForecast.asmx	0.93		
Global Weather	http://www.webservicex.net/globalweather.asmx	0.52		
USWeather	http://www.webservicex.com/usweather.asmx?WSDL	-		
Weather	http://www.riptrails.com/webservices/Weather.asmx?WSDL	-		
DASWorld Weather	http://das.ae/webservices/WorldWeather.asmx?WSDL	-		
Weather	http://www.4599.cc/service/weather.asmx?WSDL	-		
WeatherService	http://lostsprings.com/weather/WeatherService.asmx?WSDL	-		
WeatherBugWebServices	http://api.wxbug.net/weatherservice.asmx?wsdl	-		
WeatherService	http://asyncpostback.com/WeatherService.asmx?WSDL	-		
Service1	http://www.tempe.gov/wx/default.asmx?wsdl	-		
DataCenter Class	http://datacenter.c-map.no/datacenterx/datacenter.asmx?WSDL	-		
wsModule	http://nofiasco.com/WS/wsModule.asmx?WSDL	-		
LivedoorWeatherHacks	http://bearmini.net/LivedoorWeatherHacks.asmx?WSDL	-		
WSMenus	http://www.ziggs.com/ws/WSMenus.asmx?WSDL	-		
netForce	http://www.dotnetforce.com/netforce.asmx?WSDL	-		

Similarly, Table 7 shows the analysis done on the QoS availability. In this case, only the two first WS meet the minimum requirements of the user (Fig. 3). After obtaining the weights and priorities for the function, Fp can be evaluated, for WAS 1and 2.

$$Fp(Weather) = P*Pp + RT*RTp + L*Lp + ET*ETp + TT*TTp + A*Ap + R*Rp + Rs*Rsp = 0 + 1*1 + 0 + 0 + 0 + 0.25*2 + 0 + 0 = 1.5 \quad (2)$$

$$Fp(WeatherService) = P*Pp + RT*RTp + L*Lp + ET*ETp + TT*TTp + A*Ap + R*Rp + Rs*Rsp = 0 + 0.75*1 + 0 + 0 + 0 + 0.5*2 + 0 + 0 = 1.75 \quad (3)$$

Seeing the results, it can be determined that the WS Weather Service is the best one analyzed by QoS, which is selected by the workflow engine to run the service for obtaining the current weather conditions. Once executed, all data from WS are stored in the history of previous executions, so that subsequent requests can be considered as a candidate again.

VII. Conclusions

The system evaluation can be done using qualitative or quantitative techniques. Qualitative techniques are based on analyzing a list of characteristics for each system to assess, create a list of advantages and disadvantages that are intuitively compared, and suggest a final ranking of the proposed systems. Quantitative methods perform a structured, simple and secure system evaluation process, giving overall quantitative indicators that will be used to find and justify the optimal decision. Generally, quantitative methods based on scoring techniques have two indicators for each target system being a global preference score and an indicator of overall cost. This proposal does not consider the cost.

WS technology is heavily used today, and with the increasing number of available services that perform the same function, it is necessary to decide which service will be chosen to complete a given task. The purpose is to choose the most appropriate service that satisfies both the functional and non functional requirements.

The automation of business processes is considered as part of a main interest of the organizations for growth, development and competitiveness. The use of WS as part of Invoked Applications Interface of WFMS provides important benefits such as transparent distribution of applications outside the workflow, allowing the engine to invoke the application without knowing its exact location, with important benefit that can change their location on the network without involving any change in its invocation. In addition, management systems need to invoke the workflow application that is more appropriate to their needs, and this requires specifying some restrictions that will be written in non-functional specification of a web service request.

There is a clear need to develop fast and effective mechanisms that can be used for dynamic selection of services from a set of service providers. This paper presents a quantitative method to measure non-functional characteristics of services that provide the same functionality. The method can measure and then compare the scores obtained by all candidate services and select the web service that best meets the needs of the requester. In this paper, the method presented is applied to the dynamic selection of WS from a workflow engine when it requires external applications to perform business processes. It also develops a case study showing the application of the method.

We are currently working on the refinement of this method, for which we study different ways of normalizing variables to define a value for each standard QoS to evaluate and compare the performance of those features in the new services that are not found in history of previous runs. Another objective is to change the method and add the cost as an evaluation instrument.

References

1. Workflow Management Coalition. The Workflow Reference Model. http://www.wfmc.org/standards/referencemodel.htm. WfMC-TC00-1003. Last May 08.
2. Workflow Management Coalition. Programming Interface 2&3 Specification. http://www.wfmc.org/standards/publicdocuments.htm. WfMC-TC-1009.V2.0. Last May 2008.
3. World Wide Web Consortium. Web Service Architecture. http://www.w3.org/TR/ws-arch/. Último acceso Mayo 2008.
4. OASIS. UDDI Version 3.0.2. http://uddi.org/pubs/uddi_v3.htm.
5. Daniele M., Martellotto P., Riesco D. Mejorando la Invocación de Aplicaciones desde un Workflow: Un caso de estudio. XXXV Conferencia Latinoamericana de Informática - CLEI. Universidad Federal de Pelotas. Brasil. 22 al 25 de Septiembre de 2009. Páginas 550 pp. ISBN 978-987-24611-0-2.
6. Martellotto P., Daniele M., Riesco D. Invocación de Aplicaciones desde el Workflow con Servicios Web considerando sus Requerimientos No Funcionales. 38th Argentine Conference on Informatics – JAIIO 2009. 10th Argentine Symposium on Software Engineering – ASSE 2009. 24 al 28 de Octubre de 2009. Mar del Plata – Argentina. Page 160-165. ISSN 1850-2776.
7. Papazoglou M., Web Services: Principles and Technology. Ed. Pearson Education Limited, Prentice Hall, First Edition. 2008.
8. Pérez M., Mendoza L.E., Grimán. A.C., Modelo para estimación de la calidad de un Web Service. Universidad Simón Bolívar, Departamento de Procesos y Sistemas, LISI, 2006.
9. Marco Comuzzi and Barbara Pernici. A Framework for QoS-Based Web service Contracting. In ACM Transactions on web, Vol. 3(3), June 2009.
10. Mani, A. y Nagarajan, A., Understanding Quality of service for Web Services, (2002). http://www-106.ibm.com/developerworks/webservices/library/ws-quality.html.
11. Sena O., Motz R. Hacia un Modelo Genérico para la Calidad de los Servicios Web, II Congreso Español de Informática, Zaragoza, 2007.
12. Pérez, M., Mendoza, L. & Griman, A. (2005). Modelo para estimación de la calidad de un web service. XXXI Conferencia Latinoamericana de Informática, Cali, Colombia, 989-1000.
13. Dujmoviæ, J.J., A Method for Evaluation and Selection of Complex Hardware and Software Systems. The 22nd International Conference for the Resource Management and Performance Evaluation of Enterprise Computing Systems. CMG 96 Proceedings, Vol. 1, pp. 368-378, 1996.
14. Yu H.Q. and Molina H., A modified Logic Scoring Preference method for dynamic Web services evaluation and selection. The 2nd European Young Researchers Workshop on Service Oriented Computin. 11-12 June 2007. http://www.cs.le.ac.uk/events/yrsoc2007
15. Reiff-Marganiec, S., Qing Yu, H., Tilly, M., Service Selection based on Non-Functional Properties, NFPSLA-SOC07 Workshop at The 5th International Conference on Service Oriented Computing (ICSOC2007), Sept. 17 2007, Vienna, Austria.
16. H.Q. Yu and S. Reiff-Marganiec. Automated Context-aware Service Selection for Collaborative Systems. In Proceedings of The 21st International Conference on Advanced Information Systems, pages 193–200. Springer Lecture Notes in Computer Science, 2009.
17. Dujmoviæ, J.J., Continuous preference logic for system evaluation, in: B. De Baets, J. Fodor, D. Radojeviæ (Eds.), Proceedings of Eurofuse 2005, Institute “Mihajlo Pupin”, Belgrade, June 2005, pp. 56-80. ISBN: 86-7172-022-5.
18. Wenting Ma, Lin Liu, Xiaojun Ye and Jianmin Wang. Requirements-Driven Internetware Services Evaluation. In Proc. of the First Asia-Pacific Symposium on Internetware (Internetware 2009).