

Chapter #

**DECOMPOSITION OF SQUARE –BASED
REQUIREMENTS FOR THE NEEDS OF SOA
APPLICATIONS**

Subtitle

Decomposition of SQuaRE – based requirements for the needs of SOA applications

Witold Abramowicz, Konstanty Haniewicz, Radosław Hofman, Monika Kaczmarek, Dominik Zyskowski

*Department of Information Systems, Poznań University of Economics, Al. Niepodległości
10, 60-967 Poznań, Poland*

{W.Abramowicz, K.Haniewicz, M.Kaczmarek, D.Zyskowski}@kie.ae.poznan.pl,
radekh@teycom.pl

Abstract. The quality is perceived as a very important aspect of software applications. There exists widely accepted by both IT and business professionals SQuaRE model used to express the quality requirements towards applications. Nowadays, many applications are developed in accordance with the SOA paradigm and taking advantage of reusable components i.e. Web services. Therefore, a need appears to decompose the quality requirements expressed on the level of application to the level of components used to create the application. The problem with such decomposition is twofold. First of all, a quality model for Web services and SOA based applications is still a work in progress and neither of suggested approaches has been accepted as a standard. Second problem is connected with the user-oriented perception of the SQuaRE model and technical perspective of WS. Within this article we show how requirements defined within the SQuaRE model may be mapped to Web services QoS attributes. We advocate that ontological representation of quality models may provide common understanding of the attributes as well as it facilitates decomposition of business requirements to strictly technical ones. Nevertheless, due to the different perception the full automation of this process seems to be impossible.

Keywords: Web service, Security Quality Requirements Engineering, SQuaRE, QoS, SOA, applications, decomposition of requirements

1. INTRODUCTION

The quality is perceived as a very important aspect of software applications. There exists widely accepted by both IT and business professionals SQuaRE model (Software product **Quality Requirements and Evaluation**) [14] used to express the quality requirements towards applications. Nowadays, many applications are developed in accordance with the SOA paradigm and taking advantage of reusable components i.e.

Web services. Therefore, an attempt needs to be undertaken to create a SQuaRE based Web services model that would allow to express similar requirements towards the SOA-based applications [1]. The next step should be the decomposition of the requirements expressed towards the SOA-based application into the requirements applied to single components i.e. Web services.

The main goal of this chapter is therefore, to present the SQuaRE based Web services model that could be used to address a situation in which business users are presented with an application built taking advantage of the SOA paradigm and then show how expressed requirements may be decomposed. Quality of an SOA-based application greatly depends on the components that are used as a foundation of implementation. Therefore, to facilitate the overall quality measurement and application design the proposed model performs an attempt to define expectations toward quality of application's components. This allows for common understanding between operational business side of an enterprise and IT personnel which is eligible for business applications development and maintenance.

To achieve the mentioned goals, the structure of this chapter is as follows. First, a motivating scenario explaining the process of business need arrival and mechanism for its translation into a software application is presented. Then, a common definition of quality is presented along with the discussion of its impact on the discussed model and references to quality standards recognized by the software industry. The following chapter is devoted to decomposition and explanation of steps to be taken to prepare the model, technology used and mappings necessary for transition between SQuaRE model and the model of Web service quality ending in an example of real world application with appropriate mappings enlisted with the use of the featured model. Finally, a summary follows.

2. MOTIVATING SCENARIO

To provide a motivation for the work discussed in this work, let us consider the following scenario depicted in the **Błąd! Nie można odnaleźć źródła odwołania.**

Business users discover that they need additional applications to perform their tasks. They are not IT experts so they do not know how these applications should be implemented. However, as potential users of the applications in questions, they do know what they expect from these applications regarding the functionality offered as well as the expected quality in use.

Therefore, business users specify a request for application and send it to the IT department. Company rules enforce that each new application request needs to have the following elements:

- description of expected functionalities from a user point of view,
- description of requirements of a user towards the entire application as well as towards each of the functionalities, if appropriate. These requirements should be specified according to some software quality model that the entire company decided to follow. In our example it may be the SQuaRE model. Therefore, the placement of a request equals filling in the appropriate form on the web site.

The form is sent to the IT department where the IT expert validates the request and checks whether the fulfilment of the required functionalities requires any additional tools. At this point, when all requirements are clear, the configuration phase starts.

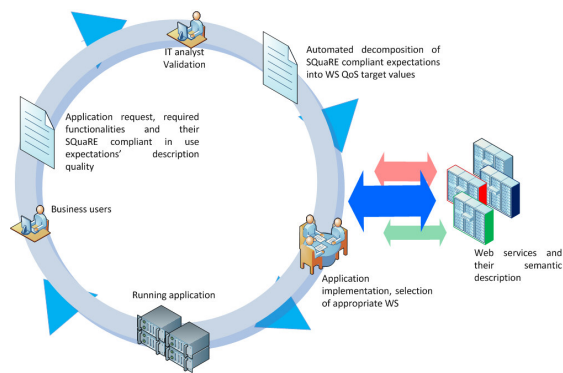


Figure #-1. [Motivating scenario]

According to the SOA paradigm Web services provided by business partners may be utilized to build applications. It may be even possible to build the entire application automatically taking advantage of the automatic composition of Semantic Web services. If the IT expert decides to build the SOA application and utilize Semantic Web services (SWS) to carry out some tasks, a need appears to select the appropriate Web services i.e. fulfilling the required functionalities and meeting the user's requirements. Therefore, the SQuaRE requirements need to be taken into account. However, as they were defined on the application or functionality level and not the component level, the decomposition of quality requirements should take place. If the decomposition is straightforward, the specific constraints on the values of QoS of SWS are derived. If the decomposition is not straightforward, it needs to be investigated which SQuaRE requirements written in a natural language affect values of which QoS attributes. In addition, as we operate in

the automated world of SWS, it would be useful to perform the decomposition task automatically or automate it at least to some extent.

In order to realize the above mentioned scenario and capture the Web services specific features the SQuaRE model needs to be appropriately enhanced. In addition, as far as the automation and machine readability is concerned, the use of semantic technologies to represent the SQuaRE as well as SWS QoS model is considered.

3. RELATED WORK

Quality related issues of software often consist of a mixture of different quality meanings. Based on [2] they are as follows:

1. Quality as an abstract, meta-physical term – unreachable ideal, which shows the direction where to products are heading but will never get there,
2. Quality as a perspective of a user considering attributes of software in special context of use,
3. Quality as a perspective of manufacturer, seen as compliance with stated requirements and following ISO 9001:1994 [3] view,
4. Quality as a product perspective understood as internal characteristic, resulting from measures of product attributes,
5. Quality as a value based perspective, differing depending on a stakeholder it was defined for.

History and experience gathered within Software Engineering constituted a significant input to the proposed quality model. Therefore, depending on the authors' background and role in software related processes, the understanding on the quality may be different [4]. The first software quality model considered as a standard was developed and published by the International Standardization Organization in 1991 as ISO/IEC 9126 [5]. Ten years later new edition of this standard reviewing quality model and introducing three perspectives: quality in use, external quality and internal quality appeared. At the same time new international initiative Software product QUality Requirements and Evaluation (SQuaRE) was set up aiming to develop set of norms ISO/IEC 25000 [6] . This new approach is perceived as a new generation of software quality models [7].

As nowadays, many applications are developed in accordance with the SOA paradigm and taking advantage of reusable components i.e. Web services, therefore, an attempt was undertaken to define Web services quality model .

A Web service may be defined as a computational entity accessible over the Internet (using particular standards and protocols) [8]. A Web Service is a technical implementation, an interface to a real-world service defined as a certain activity undertaken on behalf of a certain entity. However, the technical description of a Web service (the interface) is crucial, but not sufficient. What is also indispensable is the description of a real world service and non-functional properties of both a service and a Web service **Błąd! Nie można odnaleźć źródła odwołania..**

The non-functional properties (Quality of Service) play a crucial role in almost all service interactions (to mention only selection, discovery and filtering). Non-functional properties of a service may be defined as anything that exhibits a constraint over the functionality [10]. Majority of authors define quality of Web Services as Quality of Service using QoS as abbreviation. In many cases proposed quality attributes are analogical to those defined for low level telecommunication services or generic services definitions [11]**Błąd! Nie można odnaleźć źródła odwołania..** A survey of quality characteristics and approaches to define Web Service quality that occur most often in literature is presented in [4].

In order to decompose the requirements on the level of application to the level of Web service the appropriate relations between the application and Web services model need to be defined. However, to our best knowledge up till now, no attempt in this direction has been undertaken, mainly due to the fact that no commonly accepted standard of the Web services quality has been developed.

4. DECOMPOSITION

As indicated in the scenario, we assume that business users are provided with an appropriate form that facilitates expressing their requirements.

In order to allow for machine-readability (what is indispensable in case of SWS) and automation as well as make the mechanism flexible, throughout the scenario we use ontologies as a knowledge representation technique. Therefore, the SQuaRE standard has been ontologized and is used as a basis to create the mentioned form. Filling in the form, a user specifies which SQuaRE characteristics are important and, if appropriate, specifies the desired value.

While a user specifies requirements, the system creates in the background appropriate machine-readable representation of the requirements. These requirements refer to the SQuaRE-based Semantic Web services quality model concepts (presented within following sections) as well as the structure of the application (e.g. to certain functionalities, goals that should be

fulfilled, or the entire application) and are also stored in the form of ontology. Then, the requirements on the application level are mapped into the level of components used and new requirements on the values of SWS' non-functional properties are created. It is partially possible as we have extended the SQuaRE standard-based ontology first with some concepts specific to SOA-based applications and secondly, added the additional information like e.g. the relations between SQuaRE concepts themselves as well as SQuaRE concepts and SWS QoS concepts.

The derived requirements, also stored in the form of instances of ontology, refer to the SWS QoS ontology (and once again to the functionality and structure ontology) that defines the non-functional properties of SWS. These requirements may be used during discovery and selection of SWS to be used to create the application.

As mentioned, we do not assume that the mappings (decomposition of requirements from the application level to the Semantic Web service level) are done entirely automatically. Our aim is to provide the IT department with a tool that would help in breaking down high-level quality preferences into more detailed preferences towards specific quality properties.

4.1 SQuaRE–based SOA quality ontology

Quality considered as product attribute is difficult to trace and unambiguously define. Quality is commonly considered as complex measure. In ISO/IEC 25000 series SQuaRE model (following previous ISO/IEC 9126) authors made distinction between software quality and overall quality for user (Software Quality in Use and Quality In Use). Such distinction along with internal and external software quality distinction emphasizes different layers and different responsibilities in providing demanded quality for users. Web Services can also be perceived as set of value (and quality) adding layers as shown on Figure 2.

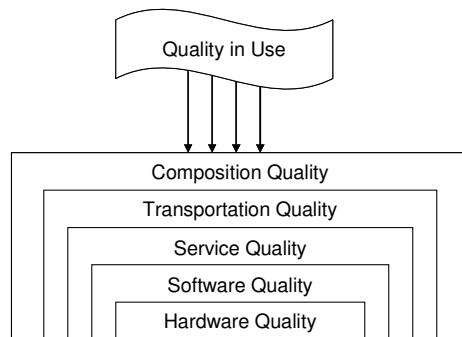


Figure #-2. [Layers of quality]

Such an approach allows defining user quality requirements independently from the selected implementation. It is also compatible with IEEE1061 standard defining two fundamental requirements for quality models:

- top-down decomposition – after quality requirements are defined model needs to support their decomposition to measurable attributes
- bottom-up measurement – basing on primitive quality indicators measures quality can be predicted

Authors assume, that Web Services are mainly designed for composition and usage by other software systems, applications or service providers (underpinning service), by compositing software etc.

In terms in compliance to SQuaRE model, Web service quality can be defined as the extent to which a service used by specific users (brokers/compositors) meets their needs to achieve specific goals including quality in use delivered to software used by end-user.

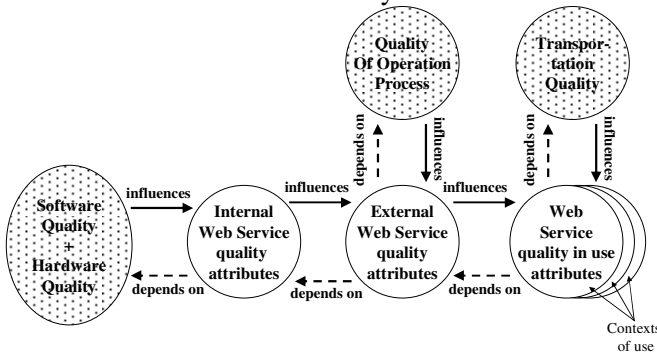


Figure #-3. [Web service quality views dependencies in lifecycle]

Due to limited space we do not present SQuaRE-based Web services quality model (full description is contained in [1]).



Figure #-4. [Ontologization of SQuaRE-based SOA Quality Model]

SQuaRE based SOA quality ontology reflects the relations between all elements and attributes of SQuaRE-based Web services quality model as presented in the Figure 3. The model consists of three layers: internal Web service quality, external Web service quality and Web service quality in use.

Internal and external Web service quality is characterized by the same eight attributes, but is measured in different contexts (static and dynamic respectively for internal and external quality). Web service quality in use is characterized by six major characteristics. SQuaRE-based SOA Quality ontology was created using OWL language. Its schema is presented in the Figure 4. Our understanding of respective sub-characteristics is given in the Table 1.

Table #-1. [Sub-characteristics of SQuaRE-based SOA Quality Model]

Concept	Definition
Security	characteristics that account for the safe and secure usage of a Web service
Efficiency	characteristics that bear on the relationship between the level of performance of the service and the amount of resources used, under stated conditions
Functionality	functions and the parameters needed for proper usage of and obtaining the expected Web service results in a specified conditions.
Reliability	likeliness of a service to provide expected results within agreed conditions. Reliability depends on the technical infrastructure of a Web service, and on the quality of results produced.
Maintainability	characteristics of the effort that is required to make specified modifications of in the service. In a particular context of Web services it should be mainly understood as a provider’s willingness to change/augment his service in reaction to a user feedback.
Usability	aspects of user-friendliness and ease of use of a Web service.
Portability	characteristics that bear on the ease of use of a Web service in different software and hardware contexts.
Interoperability of a Web service	how easy it is for a particular Web service to work with other pieces of software, or how easy it integrates in a particular system.
Safety in use	how risk-proof is the service in a given context of use
Adaptability in use	aspects of Web service that make it work in changing environment and tolerate deviations in the environment caused by user (robustness).
Security in use	characteristics that bear on the data confidentiality, reliability of produced results or authentication/authorization issues
Support in use	aspects of the assistance that may be provided for a user in order to help him complete his task. As the Web service should be rather perceived as a black box, the support of Web service usage may be realized by external means.
Usability in use defines	user satisfaction, productiveness of a Web service or effectiveness in terms of performance.

Concept	Definition
Context in use	possibility of Web service usage in different scenarios, and by different types of user. For example, a Web service should provide different levels of security for different types of users.

4.2 SWS QoS ontology

The work already done in the area of quality of Web services has been carefully examined to come up with the most complete list of quality attributes that are essential minima of the featured model. Additionally, the mentioned list is composed of highly abstract attributes in lieu of providing only domain agnostic entities. More, the selected attributes are characterized

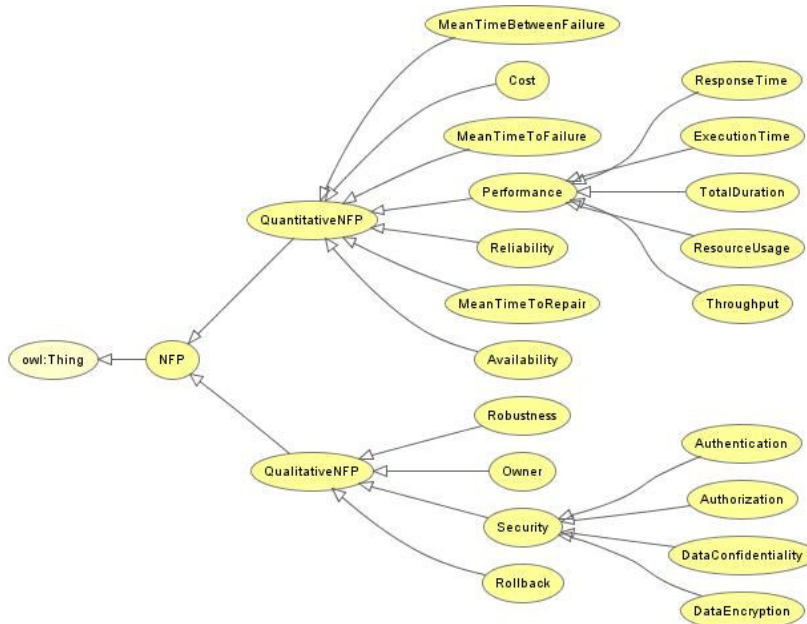


Figure #-5. [SWS Quality Model]

by the fact that their semantics cannot be confused and is easily understood by its user. As a user of the model is an IT analyst, any parallels between the attributes in the model in question and SQuaRE attributes are intended to avoid unwanted ambiguities.

In our approach we take advantage of the ontology presented in [13]. This ontology modelled the relations between different levels of Semantic Web services quality. As presented in the Figure 5, the ontology organizes the non-functional properties of SWS in several groups. First of all, the non-functional properties are divided into qualitative and quantitative. The former are not easily computable, as they are often characterized with textual

values that simply describe an opinion of a user about particular parameter. On contrary, the latter parameters are easy to measure as their values are usually numerical and quantifiable. SWS QoS ontology makes a distinction between quality of execution and quality of result parameter. The difference lays in objectivity of the properties belonging to each group. Quality of result attributes strictly depend on the user past experience with other services and his requirements towards the SWS used. Moreover, it is impossible to reflect all QoR attributes in the ontology, as they are domain-dependent. Different situation is in case of QoE attributes which are mostly generic and may be applied to every Semantic Web service, due to the nature their of usage. The detailed description of concrete parameters was already presented in [13]. In our work we show the mappings between SQuaRE–based SOA quality ontology concepts and concrete parameters that describe the quality of Semantic Web services in a more accurate way.

4.3 Decomposition model/mappings

The decomposition of SQuaRE driven requirements may not be done entirely automatically, as the SQuaRE model is not detailed when it comes to concrete sub-characteristics of quality perspectives. Another point is that the requirements defined based on SQuaRE model are always domain dependent. Therefore, every user may have different understanding of, for example, usability, portability or efficiency.

However, we are able to give some hints which QoS parameters should be associated with particular sub-characteristics of SQuaRE model. These mappings are presented in the Table 2. However, it is important to state that not all sub-characteristics may be mapped. It is due to the fact that especially quality of result of a Semantic Web service is not expressible with SWS-QoS ontology. This ontology covers only execution and technical aspects of Semantic Web services. The functionality is again domain dependent which makes expressing its quality a complex task.

The axioms defined in the SWS-QoS ontology may be used to model relationships between QoS properties and SQuaRE sub-characteristics. When the domain is well defined even the formulas denoting translation between these layers may be provided.

Table #-2. [Mapping between SQuaRE and SWS-QoS models]

SQuaRE sub-characteristic	Mapping	SWS-QoS parameter
Security	Requires that the following properties are implemented within a Web service	Concept security with all subconcepts i.e. authentication, authorization, data confidentiality,

SQuaRE sub-characteristic	Mapping	SWS-QoS parameter
Efficiency	Impose some restrictions on the values of performance-related attributes – it may be some specific constraint e.g. the duration should be less than 20 minutes or preferences e.g. the duration is important and should be optimized	data encryption Performance and all subconcepts – total duration, execution time, resource usage, response time, throughput
Functionality	It affects mainly the functionality offered by a Web service	IOPE and some domain specific characteristic
Reliability	May impose some specific constraints or preferences	Reliability
Maintainability		None relevant concept can be found
Usability		None relevant concept can be found
Portability		None relevant concept can be found
Interoperability of a Web service		None relevant concept can be found
Safety in use		None relevant concept can be found
Adaptability in use	Requires	Robustness (being implemented or high value)
Security in use	Requires	Concept security and all subconcepts
Support in use		None relevant concept can be found
Usability in use defines		None relevant concept can be found
Context in use		None relevant concept can be found

The conclusion from this table is that some parameters from SQuaRE-based model can not be translated into simple SWS QoS parameters. This is caused by different paradigms that constitute both models. SQuaRE-based is a high-level model, rather easy understood by typical IT users. SWS QoS describes more detailed, technical aspects of Web services quality and is more proper for engineers, whose task is to know which parameters influence overall quality of a software product.

4.4 Example

To present the reader with the actual application of the featured work we consider two real world examples where the work is applicable. First considered application is a car parts catalogue with service information. This application is required for planning service procedures applying to every car. One medium car model uses about 0,5 million parts, each with its technical

specification, price, technical drawing etc. Second application is a world natural resources prices calculator. This application is used by large resource consumers and/or resource brokers.

These two applications have different contexts of use and their users have different quality requirements. From the other hand, both applications can be built as standard (standalone) applications with appropriate updating processes in place or as Web service based software. We assume that the user is interested in quality thus both technical and architectural issues are not taken into consideration.

User quality requirements for software quality in use, as defined in SQuaRE model are presented in the Table 3. It presents the requirements for both example applications.

Table #-3. [SQuaRE-based requirements defined for two exemplary applications]

Quality requirement class	Car-parts catalogue	Natural resource prices calculator
Usability in use	No special requirements	No special requirements
Effectiveness in use	Searching through catalogue requires several queries to find part number. Each service requires looking for 10 to 50 parts.	Resource purchase is being planned as strategic decisions – there is no need to rush.
Productivity in use	User should be able to find 1 part number per minute (including average of 3 requests)	User should be able to generate actual and historical process for selected resources in couple of minutes.
Satisfaction in use	No special requirements	No special requirements
Usability in use compliance	No requirements	No requirements
User types in use	One type of user	One type of user
Context in use	One type of context	One type of context
Environments in use	Standard PC	Standard PC
Contexts in use compliance	No requirements	No requirements
Risks to the operator	Not applicable	Not applicable
Risks to the public	Data provided has to be reliable	Not applicable
Commercial risks in use	If incorrect parts are ordered then garage suffers financial loss	If incorrect decision is taken basing on incorrect data then company suffers huge financial loss. Application should be correcting user inputs.
Risks of software	No requirements	Software should be corruption

Quality requirement class	Car-parts catalogue	Natural resource prices calculator
corruption in use		prone
Safety in use compliance	No requirements	No requirements
Confidentiality in use	No requirements	No requirements
Integrity in use	Integrity required	Integrity required
Availability in use	Availability required at level of 98%	Availability required at level of 80%
Identification and authentication in use	No requirements	No requirements
Reliability in use	Reliable data are required	Reliable data are required
Security in use compliance	No requirements	No requirements
Learnability in use	Staff turnover is at medium level – learnability required	Staff turnover very low
Flexibility in use	Required	Not required
Accessibility in use	Application is used in garage – low resolution monitor, not precise mouse or touch-screen etc.	No requirements
Adaptability in use compliance	No requirements	No requirements

When we apply the mappings proposed in previous section we are able to acquire several SWS QoS requirements, for our two applications: throughput 50 queries per minute, execution time 20 seconds, availability 98%. It is not possible or eligible to suit any additional requirements that can be quantified.

5. SUMMARY

Designing new software solution ordered by customer analysts and developers gather quality requirements and describe them using quality model. User quality requirements need to be decomposed as shown in this chapter into characteristics, sub-characteristics and measures and such description can further contribute to establish of external and internal software quality requirements. If one of components is to be replaced by Web Service then designer of the solution has to decompose quality requirements for this Web Service call as for traditional software module. This means that definition of quality requirements starts from the same set of

requirements both for Web Service and software module. These requirements define profile, which in case of Web Services will be used for discovery and selection of Web Service delivering desired quality to end user.

That is one of the reasons quality model for Web Services should be compatible with software quality model. Further research should evaluate approaches to publishing quality characteristics for proposed model, evaluate accuracy of sub-characteristics and measures proposed for software quality model.

6. REFERENCES

1. Abramowicz W., Hofman R., Suryan W., Zyskowski D.: SQuaRE based quality model, Proceedings of International MultiConference of Engineers and Computer Scientists (IMECS 2008), p. 827-835, Hong Kong (2008)
2. Kitchenham S., Pfleeger, Software Quality: The Elusive Target, IEEE Software 13(1), 1996
3. ISO 9001:1994, International Standardization Organization, 1994
4. Abramowicz W., Godlewska A., Gwizdała J., Jakubowski T., Kaczmarek M., Kowalkiewicz M., Zyskowski D., A Survey of QoS Computation for Web Service Profiling, 18th International Conference on Computer Applications in Industry and Engineering, Honolulu, Hawaii, USA, 2005
5. ISO/IEC 9126:1991 – JTC1/SC7, Information Technology – Software Product Quality, International Standardization Organization, 1991, International Standardization Organization, 1991
6. ISO/IEC 25000:2005 – JTC1/SC7, Software Engineering - Software product Quality Requirements and Evaluation (SQuaRE), International Standardization Organization, 2005
7. Côté M-A, Suryan W., Georgiadou E., Software Quality Model Requirements for Software Quality Engineering, Software Quality Management & INSPIRE Conference (BSI), 2006
8. Preist C., A conceptual architecture for Semantic Web Services, Paper presented at the International Semantic Web Conference, 2004
9. Abramowicz W., Kaczmarek M., Zyskowski D., Duality in Web Services Reliability, Guadeloupe, French Caribbean, 2006
10. J. O'Sullivan, D. Edmond, and A. ter Hofstede, What's in a Service?, Distributed and Parallel Databases, vol 12, 2002
11. Kalepu S., Krishnaswamy S., Loke S-W., Reputation = f(User Ranking, Compliance, Verity), Proceedings of the IEEE International Conference on Web Services, San Diego, California, 2004
12. Evans J., Filsfil C., Deploying IP and MPLS QoS for Multiservice Networks: Theory and Practice, Morgan Kaufmann, 2007
13. Abramowicz W., Haniewicz K., Kaczmarek M., Palma R., Zyskowski D.: NFP Ontology for Discovery and Sharing Web Services in Distributed Registries, Proceedings of 22nd International Conference on Advanced Information Networking and Applications : Workshops, p. 1416-1421, Okinawa (2008)
14. Suryan W., Abran A., April A., ISO/IEC SQuaRE : The Second Generation of Standards for Software Product Quality, In: The 7th IASTED International Conference on Software Engineering and Applications, California , USA, 2003.