Towards the Design of a Metrics Cataloging System by Exploiting Conceptual and Semantic Web Approaches

L. Olsina[1], M^a A. Martin[1] , J. Fons[2], S. Abrahao[2], and O. Pastor[2]

[1]GIDIS, Department of Informatics, Engineering School at UNLPam
Calle 9 y 110, (6360) General Pico, La Pampa. Argentina
{olsinal, martinma}@ing.unlpam.edu.ar
[2] Department of Informatics Systems and Computing at UPV,
Camino de la Vera s/n. Aptd. 22.012. E-46071 Valencia. Spain
{jjfons, sabrahao, opastor}@dsic.upv.es

Abstract. In this article we thoroughly discuss conceptual and navigational modeling and query issues for repositories of metrics and their cataloging system by exploiting as well the power of the semantic web approach. This environment can finally allow tools, evaluators and other stakeholders to have service and consultation mechanisms starting from a sound specification of the entity type, the attribute definition and motivation, the metric definition as well as the measure type, unit, scale type, data collection instruments, protocols, among other metadata.

Keywords: Metrics, Semantic Web, OOWS, Cataloging Tool.

1 Introduction

As we know, quality assurance is one of the challenging processes both to the Software Engineering (SE) and to the Web Engineering (WE) –as a new discipline [2]. A common building block shared for many assessment methods and techniques that give support to the quality assurance process is the specification of nonfunctional requirements stemmed from a sound definition and documentation of attributes and metrics for entities. In fact, huge volumes of information about attributes and metrics for different purposes and domains have been published in diverse fora and media. However, observing the rapid and caothic growth and heterogeneity of information sources related to metrics, it urges to provide mechanisms for consensuating, structuring, and retrieving that information in order to be useful to diverse quality assurance activities. For this end, repositories of metrics and cataloging environments will basically provide to different stakeholders consultation, retrieval and reuse mechanisms starting from a sound and consensuated specification of the entity type, the attribute and metric definition, the counting rules, among other metadata.

Some initial efforts have been recently made to classify metrics for some entity type as for example metrics for software products. It is worthy of mention the initiative of the ISO 9126 standard [4] in the 2 and 3 draft documents. However, the paper-based template of information used to describe those metrics is not sufficiently complete, and is not intended to be automated by retrieval tools. Furthermore, it is also remarkable the effort carried out by Kitchenham *et al.* [6] in the definition of a conceptual framework and infrastructure (based on the Entity-Relationship model) to specify entities, attributes and relationships for measuring and instantiating software projects, with the purpose of analysing metrics and datasets in a consistent way.

This last framework served as a starting point for our proposal [9], which we are trying to strengthen not only from the conceptual modeling point of view (using O-O approaches), but also from the cataloging technologies and offered services. The contribution of this paper stems from a thorough analysis of conceptual and navigational design to the metrics cataloging system by using the OOWS (*Object-Oriented Web Solutions* [11]) approach, and the discussion of query issues for repositories of metrics by exploiting the power of semantic web models and technologies [1]. The semantic web approach bridge the gap between the enormous amount of semi-structured and heterogeneous data available in the Net and the ability to cope with all this with machine-processable semantics.

The rest of this article proceeds as follows. In Section 2, we outline a high-level view of the architecture for the Metrics Cataloging Web System (MCWS). In Section 3, we specify the conceptual model for the metrics domain, and we present a mapping between the object model and RDF (*Resource Description Framework*) Schemas. In the next section we specify navigational models for a registered user of the system using the OOWS approach; in addition, we give some examples of semantic queries to the repository of metrics by means of RQL (*RDF Query Language*). Finally, concluding remarks and future works are drawn.

2 Architecture for the Metrics Cataloging Web System

2.1 System's Users Overview

The metrics cataloging system will provide a Web-based collaborative mechanism for discussing, agreeing, and adding approved metrics to the repository on one hand, and a Web-based robust query functionality (based on Semantic Web principles) for consultation and reuse, on the other hand. These subsystems are outlined in Fig. 1.

From the MCWS design of users' point of view, five user's role types with different responsibilities and access privileges were considered, namely: *Administrators, Moderators*, *Reviewers*, *Registered Users* and *Tools/Agents*. The user's role types were discussed in [9].

2.2 The Architectural Design

In order to design the MCWS architecture, we wanted to use *semantic web* principles to define queries to repositories and documents, and *web services* to facilitate to reach the system functionality through the web.

To put these ideas into practice, we have chosen a *multi-level* architectural style or, the so-called *n-tier* architecture (see Fig. 1). The defined tiers are the following:

1. The *user interface* tier provides a personalized access to the system for "humans", i.e. administrators, reviewers, moderators and registered users. This tier redirects the incoming functionality requests to the lower level tiers.

2. The *business logic* tier specifies the system functionality using a web service approach to publicize it.
3. The *data logic access* tier is made up of a set of software components that isolates upper tiers for accessing to the stored data and documents.
4. The *persistency* tier stores the cataloged data and documents. This tier must be capable of resolving semantic web queries and of providing storage support for the metrics reviewing process.

To simplify the system structure, it has been divided into two subsystems: the *Metrics Catalog Review System*, and the *Metrics Semantic Query System*.

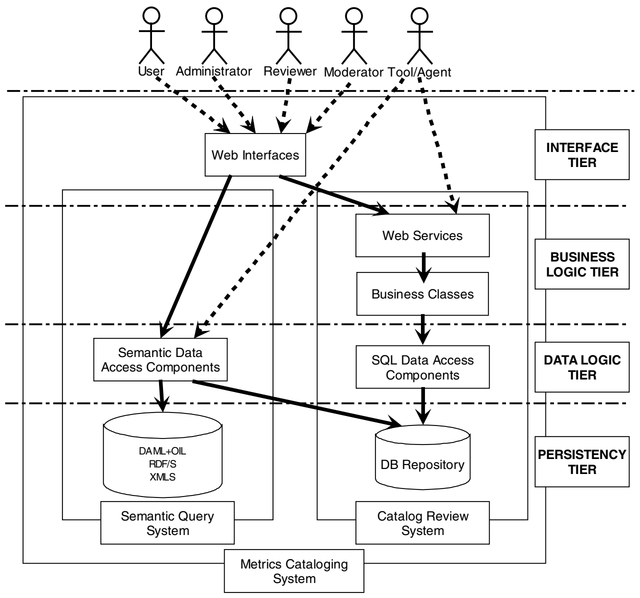

Fig. 1. An Architectural view of the Metrics Cataloging Web System.

2.2.1 The Catalog Review System for Metrics

This system is the responsible for the management and manipulation of the metrics catalog. It provides the functionality to perform the metrics reviewing process through the web and to extract data about metrics. It is mainly made up of the following tiers:

1. *Business logic* tier. It defines the functionality required by *Administrators*, *Reviewers* and *Moderators* for the metrics reviewing process and the functionality to extract data for the *Tools/Agents*. It is compound of*: a)* the *Web Services* interface; i.e. a set of services that publicize the system functionality using standard technology (e.g. SOAP/HTTP, WSDL/XML and UDDI [8]). They rely on the *business classes* to resolve the incoming requests. And b) *Business Classes* that implement the catalog review policies and provide the functionality to be used by the system users.

2. *Data logic* tier. It provides a set of SQL-based software components to facilitate and isolate data accesses (to the persistency layer).
3. *Persistency* tier. It is a relational database that stores the metric items and additional information for its management.

2.2.2 Semantic Query System for Metrics

This system is the responsible for the publication of cataloged metrics, making use of the semantic web principles. It does not implement any management functionality; however, it must be capable of querying on-line semantic documents and repositories. It is made up of two core tiers:

1. *Data logic* tier. It contains a set of components to accede for example to different on-line repositories and documents using RQL-based semantic queries [5] (see Section 4.2 for details). It can be used by the interface layer and by other software applications (represented by *Tools/Agents*).
2. *Persistency* tier. A set of web pages and documents with semantic information about metrics (specified in DAML+OIL, RDF/RDFS and XML/XMLS).

3 Conceptual Modeling for the Metrics Domain

The conceptual model specifies the main classes and relationships for the metrics domain useful to define and exploit all the information for cataloging purposes. This conceptual model is not intended to store and process instances of metric values for different entities' attributes in particular software or Web projects. An ongoing research is being carried out for this enhanced model.

3.1 The Conceptual Model

Fig. 2 shows the UML-based conceptual model for metrics (this model is called Object Model in OOWS [11]). Classes such as *Entity*, *Attribute*, *Factor*, *Metric*, *Protocol*, *ScaleType*, *MeasurementTool*, among others, and their relationships can be observed (see also [9] for details).

From the measurement process viewpoint, in the empirical domain we have mainly *Entity*, *Attribute* and *Factor* classes. The *Entity* class represents tangible or intangible objects we observe in the empirical (real) world. Types of entities of interest to SE and WE are: Project, Product, Product in Use, Process, and Resource. An entity can be hierarchically broken down into sub-entities. The *Attribute* class represents a measurable property of an entity. For a given attribute, there is always at least an empirical relationship of interest that can be captured and represented in the formal domain (by means of a *Metric*), enabling us to explore the relationship mathematically. Attribute is an object at the lowest level of abstraction that can be quantified directly or indirectly by a metric, meanwhile a *Factor* class represents objects at the highest level of abstraction such as Quality, Effort or Cost. A factor can be modeled by a *FactorModel* (e.g. the ISO 9126-1 specifies a quality model to both software product and product in use entities. In turn, the quality model is specified in terms of *Characteristics* and *Subcharacteristics* [4]).

The *Metric* class represents the specific mapping of an empirical attribute (that belongs to an entity) to the formal world, in addition to the specific criteria and procedures to measure it. The metric concept can't be fairly understood if we don't consider also other related concepts such as Measure, Unit, Scale Type, Protocol, Equation and Measurement Tool and their relationships.

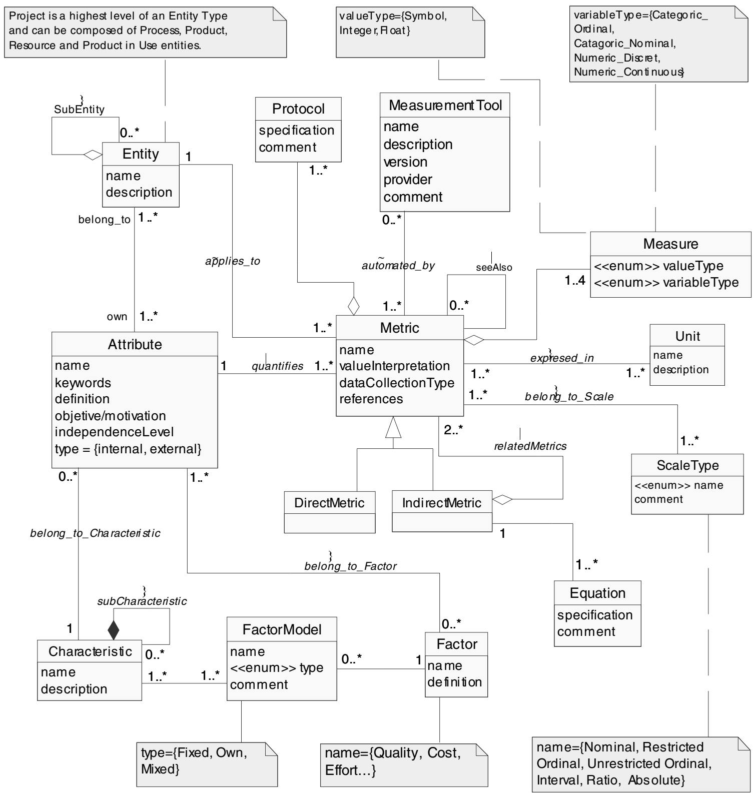

Fig. 2. Conceptual model for the domain of metrics.

The *Measure* class is part of the metric class. It has field members such as the value and variable types (we don't include the value field itself because we store just metadata for cataloging purposes, not instances of metric values). In Fig. 2, the types of values and variables can be observed in the UML-comment element. The *Protocol* class specifies the counting rules and procedures to be applied to the respective metric. On the other hand, a measure of a metric can be expressed in one or more *Units* (e.g. the *Size of Visible Text* metric applied to a *Webpage* entity can be measure in number of bytes or number of word units). Furthermore, the *Scale Type* of a metric affects the sort of arithmetical and statistical operations that can be applied to a measured value. A scale type is defined by admissible transformations. Ultimately, all the information observed in the conceptual model is necessary in order to have a sound specification of attributes and metrics for cataloging and consultation purposes.

3.2 Mapping between the Object Model and RDF Schemas

For the conceptual model of metrics previously discussed, we present here the definition of one layer of the semantic approach [1], [3]. We used RDF Schemas in order to build the model of resources and their relationships. RDF schema is about making machine-processable semantic statements.

As shown in the graph of Fig. 3, we map a RDF class for each class of the conceptual model (a more thorough and robust mapping will be presented in a follow-up work). In addition we define a property for each relationship (the *subClassOf* and *seeAlso* properties are defined beforehand in the de facto W3C standard [12]). Notice that in Fig 3 we show the main resource classes and relationships of the model in a RDF-graph format for readability and space reasons; the reader can access the full RDFS specification in http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1.

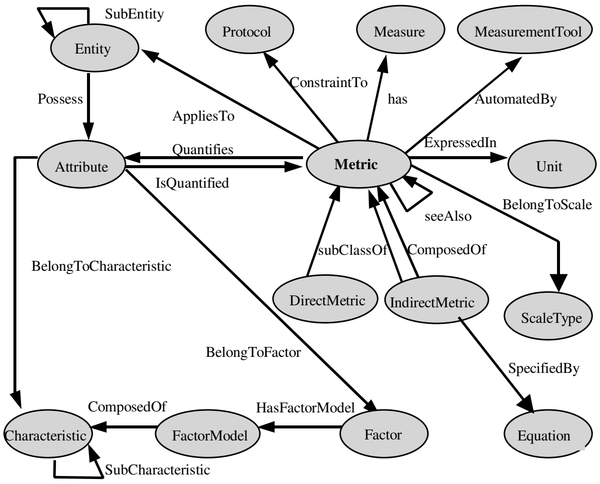

Fig. 3. A representation in RDF-graph format for the metrics domain.

The main resource type is the Metric class, which conceptualize all the available information for a particular metric by interlinking other resources as well, namely: the *Entity* type to which applies to; the *Attribute* that quantifies; the *Protocol* to which is constraint to; the *Unit* which is expressed in, among other relationships.

4 Designing the Navigational Model

We argue that the conceptual modeling process is necessary in order to document and develop correct web applications. In addition, navigational requirements are a critical feature. In this section we analize the navigational model introduced by the OOWS

[11] approach to specify navigational requirements at conceptual level. Next subsections present the navigational modeling of the MCWS for the *Registered User* in addition to the use of a semantic query language to retrieve semantic information.

4.1 The Navigational Model for the Registered User

OOWS is a conceptual modeling method that uses a *Navigational Model* to properly capture the navigation semantics of a web application by providing a set of conceptual constructors to define navigation capabilities (structure, "*navigability*", etc.).

The navigational model is made up of a set of navigational maps, one for each type of user of the system. This *Navigational Map* (see Fig. 4) represents the view that each kind of user has over the system, defining how users would access system information and functionality. It is represented by means of a directed graph where nodes represent user interaction units (navigational contexts), and arcs (navigational links) represent node reachability (defining valid navigational paths). A *navigational context* is a graph node (depicted as UML packages stereotyped with the «context» reserved word) and represents a view over a set of class attributes and services, and also class relationships, from the Object Model. There exist contexts that can be reached at any moment independently of the current context (*exploration contexts,* labeled with an "E") and contexts that can only be reached following a predefined navigational path (*sequence contexts*, labeled with an "S").

Fig. 4 presents the navigational map for the Registered User (see Section 2.1). It defines six navigational contexts: Factors, Entities, Metrics and Tools (exploration contexts that have a dashed arrow starting from the user), Characteristics and Attributes (sequence contexts, only reachable following predefined navigational paths).

The following step is to define with detail each context. A navigational context is made up of a set of *navigational classes* and *navigational relationships*. A *navigational class* (graphically depicted as UML classes stereotyped with the «view» reserved word) represents a view of a class over a subset of visible attributes and services. A navigational class represents a graphical query to retrieve all the instances of the related class, specifying the attributes to retrieve. Fig. 5 shows the definition of the Metrics context. This context is responsible of provide relevant information about metrics for the Registered User. This context has thirteen navigational classes (Metric, Attribute, ..) with its attributes (name, ...). All navigational classes must be related by a unidirectional binary relationship, called *navigational relationship* (depicted as an

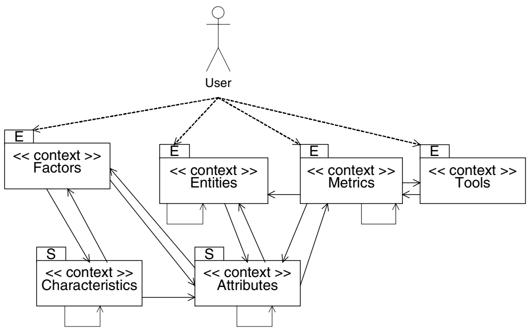

Fig. 4. Navigational Map for the *Registered User* of the MCWS

arrow). It is defined over an existing aggregation/association/composition or specialization relationship and specifies a retrieval of all the instances of the target navigational class related to the origin class. For instance, all the related (quality) Attributes (name and keywords), Entities (name), Protocols (specification and comment), etc… for each retrieved Metric will be shown.

There exist navigational relationships of two types: (1) *context dependency relationships* (depicted as dashed arrows) and (2) *context relationships* (depicted as solid arrows). The former only defines a population retrieval. The latter also defines a navigation to a *[target context]* using as the *anchor* an attribute of the target class. For instance, when retrieving all the MeasurementTools for a given Metric, it will be possible to navigate to the target context [Tools] by selecting one of them using the name of that measurement tool as the anchor.

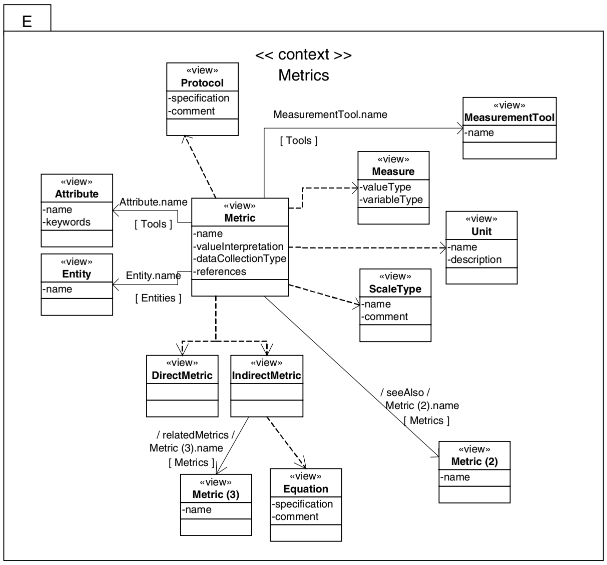
E
<< context >>
Metrics
«view» Protocol -specification -comment
MeasurementTool.name
[Tools]
«view» MeasurementTool -name
«view» Measure -valueType -variableType
«view» Attribute -name -keywords
Attribute.name
[Tools]
«view» Metric -name -valueInterpretation -dataCollectionType -references
«view» Unit -name -description
«view» Entity -name
Entity.name
[Entities]
«view» ScaleType -name -comment
«view» DirectMetric
«view» IndirectMetric
/ seeAlso / Metric (2).name [Metrics]
/ relatedMetrics / Metric (3).name [Metrics]
«view» Metric (2) -name
«view» Metric (3) -name
«view» Equation -specification -comment

Fig. 5. *Metrics* context for the *Registered User*

4.2 Using a Semantic Query Language

We are using a semantic query language as RQL [5] in order to retrieve the semantic information from the cataloging system. RQL is a typed declarative query language, which is based on the evaluation of path expressions on RDF graphs, featuring variables on labels for both nodes (i.e., classes) and edges (i.e., properties). One of the main RQL features that distinguishes from some other RDF query languages is its capability of querying both RDF schemas and descriptions (i.e., instances). For the MCWS this is a powerful feature for exploiting semantic documents and repositories of metrics, by allowing us not only to retrieve metrics data and its relations but also descriptive information (metadata) of available resources and services in the web, without human-processing intervention. Moreover, RQL has inference capabilities on hierarchies of classes and properties. This feature allows us exploiting additional information, even information that is not explicitly modeled in schemas. In Table 1, we specify a set of queries for the metrics cataloging system in order to illustrate some RQL features and powerness.

Table 1. RQL query examples for the metrics RDF schemas and descriptions.

Description	Query Example
1) Retrieves all instances of attributes and metrics for a subentity (named Webpage)	Select X, Y, Z from http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Entity{X}. http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Possess{Y}. http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#IsQuantified{Z} Where X="Webpage"

2) Retrieves all instances of metrics, supported by the Bobby Measurement Tool	Select X, Y
	from http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Metric{X}.
	http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#AutomatedBy{Y}
	Where Y="Bobby"

3) Retrieves all properties names and their ranges to the Metric class	Select @P, range(@P) from {$C}@P Where $C ="http://gidis.ing.unlpam.edu.ar/rdf/RDF-Metricas1#Metric"

5 The Semantic Web Modeling Power for the Cataloging System

In this section we briefly address the power of the semantic web approach focusing us particularly on queries for the metrics cataloging system and some used OOWS modeling primitives. In order to model the metrics cataloging system, in this work, we proposed the use of the OOWS Object Model for conceptual modeling, in addition to the use of semantic statements specified in the form of RDF Schemas (and DAML+OIL). This is possible due to the feature of reification that allows define classes and relationships of the conceptual model as semantic statements. Furthermore, we modeled the navigational aspects of the MCWS using Navigational Maps, and Navigational Contexts. These OOWS navigational primitives, particularly, Navigational Contexts depend on query specifications of different expressive power. Even if we have index access structure and filter capabilities in the OOWS primitives, the power of semantic queries are far beyond.

In order to illustrate the power of semantic queries, we describe some of them exemplified in Table 1. In the first example, the RQL query acts only on RDF descriptions (instances) without the need of schemas. In the second case we just use a property (the *AutomatedBy property* -see the graph of Fig. 3) in order to recover all instances of the *Metric* class starting from the *MeasurementTool* class, without the need of defining either the inverse property (*Automates*) or a new index access structure in the navigational model. Lastly, in the third case the query retrieves all the properties information related to the *Metric* class, i.e. the labels of all edges.

6 Concluding Remarks and Future Work

In order to contribute to the comprehension and selection process whether metrics can be useful, easy to collect and understand, a sound metric specification, flexible documentation, consultation and retrieval mechanisms are needed. For this end, we argued that a well-designed repository of metrics and a powerful cataloging system can be efficiently used to support quality assurance processes such as nonfunctional requirement definition and specification, metrics understanding and selection, quality testing definition, amongst others.

Unfortunately, in the recent research initiatives of the SE and WE communities, sound metrics specifications and cataloging environments as technological support for quality assurance processes have often been neglected. As a way to contribute to fill this gap, our current research concern is manifold. Firstly, regarding the designed cataloging system we are, on the one side, building a virtual community mechanism that basically will provide a Web-based synchronous and asynchronous collaborative facilities for discussing, agreeing, and adding approved metrics to the repository. On the other side, we are implementing the cataloging application with a Web-based browse and query functionalities for consultation and reuse based on semantic web capabilities. One key factor for the repository is getting consensus between researchers and practicioners about the terminology in the metrics domain in order to have a robust ontology: We are in the first steps of discussions and agreements. Secondly, we are exploring the inclusion of OOWS-centered modeling primitives (similar to the initiative reported in [7]) that embrace the power of the semantic web principles and models: This concern is in the start-up stage. Thirdly, we are designing the web services in order to allow the WebQEM_Tool be able to use the metric repository services to retrieve protocol and formula specifications, amongst other items. WebQEM_Tool is the supporting tool for the WebQEM [10] methodology.

References

1. Berners-Lee, T.; Hendler, J.; Lassila, O., 2001, *The Semantic Web.* Scientific American, http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html.
2. Deshpande, Y.; Murugesan, S., et al., 2002, Web Engineering, *Journal of Web Engineering*, Rinton Press, US, 1(1), pp. 61–73.
3. Fensel D., 2002, Ontology-Based Knowledge Management, *IEEE Computer*, 35(11), pp.56–59
4. ISO 9126-1: 2001 Int'l Sdr, Software Engineering, Product Quality, Part 1: Quality Model.
5. Karvounarakis, G.; Alexaki, S.; et al, 2002, RQL: A Declarative Query Language for RDF. *In Procced. of 11th WWW12*, Honolulu, Hawaii, US.
6. Kitchenham B.A., Hughes R.T., Linkman S.G., 2001, Modeling Software Measurement Data. *IEEE Transactions on Software Engineering*, 27(9), pp. 788–804.
7. Lima F.; Schwabe D., 2002, Exploring Semantic Web Modeling Approaches for Web Application Design. *In Proc. of the 2nd Int'l Workshop on Web-Oriented Software Technology" (IWWOST)*, in the framework of ECOOP 2002, pp. 120–133, Malaga, Spain.
8. Newcomer, E.; 2002, *Understanding Web services: XML, WSDL, SOAP, and UDDI.* Addison-Wesley, Boston, MA.
9. Olsina, L.; Lafuente, G. Pastor, O., 2002, Towards a Reusable Repository of Web Metrics, *Journal of Web Engineering*, Rinton Press, US, 1(1), pp. 61–73.
10. Olsina L., Rossi G., 2002, Measuring Web Application Quality with WebQEM, *IEEE Multimedia*, 9(4), pp. 20–29.
11. Pastor, O.; Abrahão, S. M.; Fons, J. J., 2001, Object-Oriented Approach to Automate Web Applications Development, *2nd EC-Web'01*, Munich, Germany, Springer Verlag, 16–28.
12. W3C, WWW Consortium, 2002, *RDF Primer*, W3C Working Draft 19 March 2002 http://www.w3.org/TR/2002/WD-rdf-primer-20020319/