

Challenges to the Development of Standard Software Quality Model

Simrandeep Singh Thapar
Associate Professor
Deptt. of Computer Applications
ACET, Amritsar, Punjab (India)

Paramjeet Singh
Associate Professor
Deptt. of Computer Science & Engg.
PTU RC, Bathinda, Punjab (India)

Shaveta Rani
Associate Professor
Deptt. of Computer Science & Engg.
PTU RC, Bathinda, Punjab (India)

ABSTRACT

To stay ahead in competition, software organizations have to deliver required functionality consistently guaranteed with high quality. The quality expectations of customers at purchase time from the software is the major reason of emergent attention of software organizations to implement quality management in software development. Nowadays, use of Component Based Software Development (CBSD) approach is considered success factor for business because of its underlying benefits like reusability, on-time delivery, high quality, and less cost. But success of CBSD depends upon the quality of components used. Many quality models have been proposed to provide high quality products, some of these are for general usage and others are for specific applications or domains. These specific purpose models are improved or modified forms of basic quality models especially ISO 9126. This paper presents key challenges or deterrents to the development of standard, complete and pervasive software quality models, solution to these challenges and their importance is also discussed.

General Terms

Design, Standardization

Keywords

Quality Models, CBSD, Non-Functional Requirements, Challenges

1. INTRODUCTION

Software organizations must deliver software products that meet and/or exceed customer expectations. To achieve this, customers' expectations need to be captured at the beginning of the project by clearly defining all quality or Non-Functional Requirements (NFRs). Quality may be best described by NFRs, as these are well known success factors in software development. At time of software or component purchase/selection, NFRs prove to be decisive since almost all products provide the required functionality. Lack of careful consideration of NFRs increases the risks of non-attractiveness of potential customers and cost overruns, whereas cautious inclusion of quality requirements may improve the overall quality of final software.

CBSD as a paradigm helped software organizations to solve many software engineering problems like cost, schedule overruns and low quality of product by using a set of reusable software components. Main motivation of reusing existing software components is that components are already tested. So, software engineers face reduced errors that automatically shorten the time to market of the system under development. The CBSD leads to software reuse, and reusability provides software engineers and

managers with a number of measurable benefits. Based on studies of reusability, QSM Associates, Inc., reports component assembly leads to 70% reduction in development cycle time; 84 percent reduction in project cost, and productivity index of 26.2, compared to industry norm of 16.9 [1].

However, besides several advantages, CBSD still faces many questions. Some of the most important questions are: (a) effect of component on the existing system, (b) coverage of testing on component, (c) component reliability, (d) component maintenance when problem arise, (e) satisfaction of user requirements by components. These questions revolve around the quality of the purchased components, and, if these questions are not answered they may pose various risks to the whole project. Component technology is a technology that develops a system through acquisition and assembly. The component based system is largely dependent on the quality of each component that comprises the system. If low quality components are used, they might cause damage to the existing system leading to avalanche affect [2].

There are many problems in development of quality software and use of CBSD approach to software organizations such as absence of standard quality model or framework, assurance of quality by vendors and high quality software components. Quality of software product depends entirely on the quality model used. Therefore an effective and complete quality framework or model is required that will ensure quality of software. Depending upon this, the vendor can guarantee the quality of software.

A quality model provides the basis for quality requirements specification and quality evaluation. Quality model depends on the type of software product. Many quality models have been proposed to attend to quality requirements, some of these models describe quality in general, and other models are of specific nature for specific purposes, applications or domains. These specific purpose models are modified or tailored versions of basic models especially ISO 9126 which is considered most comprehensive and flexible model by far. But, this model is too general in nature to address specific needs of all domains.

The purpose of this study is to find the best software quality model from all presented till year 2011 in the presence of challenges to the development of standard quality model. Our research will be very useful for software industry to improve and assure the quality of software products by using a standard software quality model that takes care of all issues mentioned in this paper. The research questions addressed by this study are:

RQ1: What type of quality models have been proposed till 2011?

RQ2: What challenges are posed by quality models?

RQ3: Which quality models may be considered standard or almost complete?

RQ4: What are other important challenges/deterrents which are related to this research?

Section 2 addresses RQ1 and presents analysis of all quality models proposed till year 2011, section 3 addresses RQ2, RQ3 and put forward nine essential issues or challenges to the development of standard quality model; section 4 addresses RQ4 and concludes the paper with discussion of few more important issues.

2. ANALYSIS OF QUALITY MODELS

For the ongoing research on development of standard quality model, 24 key models are chosen for study. It can be seen in figure 1, models are categorized into basic quality models (1977-2001) and tailored quality models (2002-2011). Research in direction of quality improvement and evaluation of software opened in view of growing quality expectations of customers from software, at that time basic quality models were introduced. Tailored quality models are improved forms of basic quality models to adjust the needs of underlying application domains.

During the period (1977 to 2001), five quality models emerged and, during (2002 to 2011) nineteen models are proposed. These models encouraged quality evaluation and improvement of But, during (2002-2011) software industry progressed and expanded rapidly with ever increasing competition, so, imminent need of quality model boost the research in this area. Researchers in this period proposed various models by adding or modifying some extra aspects to basic models to suit the needs of given applications or domains.

2.1 Basic Quality Models

During 1977 and 2001, proposed software quality models are McCall, Boehm, FURPS, Dromey and ISO 9126. Most of these models serve as base models for (tailored) models proposed after year 2001. Each model was introduced with purpose of defining quality in a new way covering all the perspectives of software quality with different point of views for evaluation of a software product.

Major contribution of McCall's model [3] is consideration of relationships between quality characteristics and metrics but it measures the quality subjectively in the form of yes/no, 0/1 or range to show presence of characteristics. Quality is classified into revision, operation and transition perspective, considering user's and developer's view. Boehm [4] introduced quality model to automatically and quantitatively evaluate the quality of software. In this model, characteristics and sub-characteristics are loosely-coupled and it's (as-is) aspect is subjectively specified. FURPS [5] model consider two steps, setting priorities and defining quality attributes that can be measured. This model is not heralded as it is proprietary model of an organization. Dromey [6] taken into consideration relationship between characteristics and sub-characteristics in its product based quality evaluation framework and emphasized that to make a high quality product all constituent artifacts must be of high quality, so he made a product based quality model, but he failed to discuss how it could be realized. ISO 9126 [7] quality model is based on McCall and Boehm's model which cover all aspects of software quality but metrics are not consistent with their own definitional concept of metric.

It is inferred from this study that researchers during this period worked for designing generic models for quality of entire system. Quality models [4, 5, 6, 7] are hierarchical in structure. At the top of hierarchy, those characteristics, factors or properties are placed, which characterizes software quality from various aspects. It is

software. These models demonstrate quality from various dimensions. Basic quality models are generic in nature that may be adjusted for any software, while tailored models are specific for particular application and domain.

It is obvious during (1977-2001) few models were proposed as few software organizations were existing. CBSD paradigm was almost absent and nature of software products was less diversified which lead to modest competition.

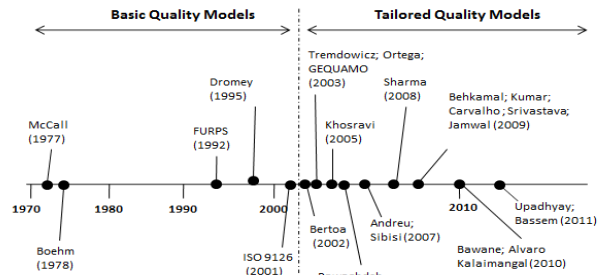


Figure1. Quality models proposed till 2011

observed that high-level characteristics represent user's view of quality while sub-characteristics represent technical personnel's view and overall quality of software. It is also observed that reliability is considered most important characteristic along with usability, efficiency and maintainability.

2.2 Tailored Quality Models

Bertoa's quality model [8] defines a set of quality attributes and their associated metrics for the effective evaluation of COTS components. Geqrgiadou [9] proposed Generic multilayered customizable software Quality Model (GEQUAMO). Ortega [10] defined a systemic approach to software products in proposed quality model. This model is evaluated using a method so it can be validated and also enhanced. Tremdowicz [11] proposed prometheus approach to model quality by integrating qualitative and quantitative approaches within software product lines and different aspects like individual views or different evaluation objects. Khosravi [12] used design patterns in proposed quality evaluation approach and suggested to state target user and to define extra layer of super-characteristic. Rwashdeh [13] suggested a quality model for components with added feature of matching the appropriate type of stakeholders with corresponding quality characteristics.

Andreu's [14] quality model is based on ISO 9126 which may be used for development and evaluation of original components and may be tailored according to the organization re-user and the domain needs of the targeted component. Sibisi [15] proposed a process model for customizing generic software quality models at only characteristic level; this framework may be quantitatively validated. Sharma [16] added/modified some extra features to ISO 9126 to make it appropriate for given applications and for weight assignment Analytical Hierarchical Process (AHP) is used. The result value of AHP can be used to compare and select the best suitable component as per all desired quality characteristics. Behkamal's [17] quality model is also based on ISO 9126 which may be used for evaluation of B2B applications. In this model, quality factors are extracted from web applications and B2B e-commerce applications. These factors are weighted from the viewpoints of both developers and end users to add in the model.

Table 1. Brief details studied of models

Models	Year	Thrust area(s)	Funded research (R) / General Research (G)	Original idea (O) / Derived (D)
McCall	1977	<ul style="list-style-type: none"> • Quality factors and their relationships • Metrics • User and developer's view 	G	O
Boehm	1978	<ul style="list-style-type: none"> • Characteristics and Sub-characteristics • Quantitative measurement 	G	O
FURPS	1992	<ul style="list-style-type: none"> • Reusability • Quantitative measurement 	R	O
Dromey	1995	<ul style="list-style-type: none"> • Product based model • Relationship among characteristics 	G	O
ISO 9126	2001	<ul style="list-style-type: none"> • Coverage of overall quality • Standardization of quality model • Metrics 	R	D
Bertoa	2002	<ul style="list-style-type: none"> • Quality model for components • Evaluation of components • Metrics for components 	G	D
Tremdowicz	2003	<ul style="list-style-type: none"> • Quality modeling in software product lines • Non-functional properties 	G	O
Ortega	2003	<ul style="list-style-type: none"> • Systemic quality model for quality evaluation 	R	O
GEQUAMO	2003	<ul style="list-style-type: none"> • Multilayered Customizing and dynamic model • Stakeholders 	G	O
Khosravi	2005	<ul style="list-style-type: none"> • Quality model • Quality issues • Metrics and Quality evaluation using design patterns 	G	D
Rawashdeh	2006	<ul style="list-style-type: none"> • Software quality model for evaluation of components • Stakeholders 	G	D
Andreu	2007	<ul style="list-style-type: none"> • Quality framework for developing and evaluating original software components 	G	D
Sibisi	2007	<ul style="list-style-type: none"> • Process framework for customizing software quality models • Quantitative validation of model 	G	D
Sharma	2008	<ul style="list-style-type: none"> • Quality evaluation using analytic hierarchy process 	G	D
Behkamal	2009	<ul style="list-style-type: none"> • Quality model for evaluation of B2B applications 	G	D
Kumar	2009	<ul style="list-style-type: none"> • Aspect oriented software quality model 	G	D
Carvalho	2009	<ul style="list-style-type: none"> • Quality verification framework to evaluate embedded software components 	G	D
Srivastava	2009	<ul style="list-style-type: none"> • Metrics and measurement approaches • Statistical measurement of software quality • Stakeholder's view 	G	D
Jamwal	2009	<ul style="list-style-type: none"> • Software quality evaluation using polarity profile 	G	O
Bawane	2010	<ul style="list-style-type: none"> • Quantitative assessment of quality • Correspondence with software development process • Stakeholders 	G	D
Alvaro	2010	<ul style="list-style-type: none"> • Component quality framework 	R	D
Kalaimangal	2010	<ul style="list-style-type: none"> • Evaluation of quality of COTS components 	G	D
Upadhyay	2011	<ul style="list-style-type: none"> • Quality model to build component quality assurance framework • End user's view 	G	D
Bassem	2011	<ul style="list-style-type: none"> • Quality evaluation model based on user's view 	G	D

Srivastava's [18] model measures the software quality statistically by taking care of three different views of user, developer and manager. Jamwal's [19] model may be used for evaluation of software quality models. Kumar [20] proposed an aspect oriented software quality model (AOSQUAMO), in this AHP is used to evaluate quality of AO software systems as a single parameter. Carvahlo's [21] proposed quality verification framework may be used to evaluate the quality of embedded software components. Bawane's [22] model establishes the quantitative quality requirements expected by various stakeholders and to incorporate these requirements in the product under development. This model is dynamic and allows product deliverables to be compared with set goals by various stakeholders through measurements and metrics throughout the development life cycle.

Alvaro [23] presented the quality framework and its related-modules: the Component Quality Model (CQM), the Evaluation Techniques Framework represented by the Software Component Evaluation Techniques Model (SCETM), the Metrics Framework and the Component Evaluation Process. Kalaimangal's [24] quality model Q'Facto12 is based on ISO 25000 quality standard that may be used to evaluate the quality of COTS components by end users before integration into existing software systems. Upadhyay [25] proposed a quality model (SCQM) for building a software component quality assurance framework with the objective to develop capability maturity model for components that may be mapped to the proposed component model. Based upon this end-user may take decision upon selection, evaluation and ranking of potential component candidates and may attain improvements in the component design and development wherever possible. Bassem's [26] considered only user's view in quality model without taken into account specific scope of software product.

3. CHALLENGES TO THE DEVELOPMENT OF STANDARD QUALITY MODEL

A standard quality model must have minimum set of attributes, easy to use, must cover all aspects of quality, consider all possible characteristics of underlying domains, flexible enough to be applicable in all application domains with minor modifications or additions, must fulfill all the needs of stakeholders and must serve as a standard for the evaluation of software products.

Research on defining and evaluating quality of software products has received tremendous attention in academia and industry. An extensive literature survey in the area of software quality model and evaluation is conducted to find out standard quality model. Many researchers and organizations proposed quality models to improve and evaluate the quality of software. These models are here categorized into basic and tailored models depending upon their nature, dependence and applicability.

However, after analyzing the literature, it is established that these models are not comprehensive and complete and there are still many issues which pose challenges to the development of standard quality models. These issues with their importance and possible solution are mentioned here.

a.Issue: Absence of association between quality models and software development process

Importance: Implementation of quality in a software product is an effort that should be formally managed throughout the software engineering lifecycle. Such approach leads to software quality engineering (SQE). To improve the quality of software, quality

models must be formally related with the software development process

Solution: Relationship between development process and quality model may be established in this way, corresponding to requirement phase, stakeholders' quality requirements are determined. In design phase, decompose these abstract requirements into refined levels of characteristics, sub-characteristics or metrics according to suitability. In implementation phase, choose those characteristics or sub-characteristics which are appropriate for the application in hand. In testing phase, measure these characteristics using metrics. In maintenance phase, measured value can be matched to expected result to assess the reliability of quality model and in case of non-compliance it can be modified or improved.

b.Issue: Non-evolving quality models

Importance: A quality model must evolve from abstract details to finer details as the software development process progresses. It keeps the model simple to understand and easy to use.

Solution: A quality model must have number of levels to manage its complexity to an appropriate level. It must be scalable depending upon the software product.

c.Issue: Non-maintainable quality models

Importance: Fixed models are of little use in most of the applications due to their inflexibility towards impending changes. Quality models must be iterative in nature to overcome residual issues. Maintenance of the quality model is necessary to remove the detected deficiencies related to quality once model becomes functional. It is also important to cope with ever changing requirements of stakeholders, business competitiveness and environment.

Solution: A mechanism must be integrated in the quality framework to make it amenable to the potential changes.

d.Issue: Generality of quality models

Importance: General quality models have general quality attributes or characteristics which can't be simply applied in all applications since every application has different nature and requirements in evaluation of quality. Application specific quality model takes care of requirements of given application. To get reliable estimate of quality, requirements must be given priority or weightage according to the particular application. Quality requirements must correspond to goals or objectives which also depend on the particular application area. Specific quality model encompass complete set of quality attributes and evaluation methodology for given application which provide enough confidence in quality model.

Solution: Although it is difficult to build separate quality models for different applications but it becomes necessary to get highly reliable estimate of quality. Separate models must be developed for major domains or applications areas such as for components etc.

e.Issue: Negligence of risk driven aspect in quality model

Importance: Risk management approach in quality models haven't followed so far. It will be effective to assure the development of high quality components. It is known fact that if risks are not mitigated earlier then whole project may end in jeopardy. So, in quality models it should also be incorporated.

Solution: Special attention must be given to those characteristics which may cause cost, quality, and schedule risks to the software. For example, if functionality is not properly implemented in software it may cause quality risk which indirectly causes cost and schedule risks.

Table 2. Issues in quality models

Issues Models	Issues								
	a (association)	b (evolution)	c (maintainable)	d (general)	e (risk - driven)	f (stakeholders)	g (subjective evaluation)	h (validation fairness)	i (guidelines)
McCall	√	√	√	√	√		√	√	√
Boehm	√	√	√	√	√		√	√	√
FURPS	√	√	√		√	√	√	√	√
Dromey	√	√	√		√			√	√
ISO 9126	√				√			√	
Bertoa	√	√	√		√			√	
Tremdowicz			√		√	√	√	√	√
Ortega	√	√		√	√			√	√
GEQUAMO	√						√	√	
Khosravi	√	√	√	√	√			√	√
Rawashdeh	√	√	√	√	√		√	√	√
Andreu	√				√			√	√
Sibisi	√	√			√	√		√	√
Sharma	√	√	√		√			√	√
Behkamal	√	√	√						√
Kumar	√	√	√		√			√	√
Carvalho	√	√	√		√			√	√
Srivastava	√	√	√		√			√	√
Jamwal	√			√	√			√	√
Bawane					√			√	√
Alvaro	√			√	√				
Kalaimangal	√				√				√
Upadhyay	√	√	√		√	√		√	
Bassem	√	√	√	√	√		√	√	√

f.Issue: Lack of involvement of stakeholders in quality framework

Importance: Involvement of stakeholders is very important in quality framework. It is known fact that quality is best perceived by users' satisfaction and fulfillment of requirements. Stakeholders must be part of quality evaluation and quality framework. They must interfere in the model structure to modify it, if needed, their valuable feedback aligns or fine tunes the quality engineering process. Their satisfaction must be given weightage as part of overall quality. Each set of characteristics, sub-characteristics and metrics of quality model must be classified for each category of stakeholders.

Solution: Requirements must be gathered from them and validation of quality with respect to these requirements must be done by stakeholders. Stakeholders must be held in loop in quality estimation and quality framework as their satisfaction will make the business go.

g.Issue: Subjectivity in quality evaluation

Importance: Quality must be assessed objectively and statistically using automated tools which provide accurate results of the quality.

Solution: Metrics must be part of quality models and their usage must also be described in sufficient details.

h.Issue: lack of fairness in quality validation

Importance: Validation of the quality model must always be done by independent party or by expert academicians or personnel external to the organization to get reliable estimates of quality. This will demonstrate the precise worth of quality model without any biasness.

Solution: Quality model must be validated by more than one evaluator; it may include expert academicians, third party organizations. They must apply quality model on number of software applications and identical results on all these applications must be considered as successful quality model.

i.Issue: Absence of necessary guidelines or documentation for quality model

Importance: Necessary guidelines or documentation for quality model help developers and managers to understand and use of quality model. It must describe how to use and apply the quality

model and under which environment and condition it may be used.

Solution: All required guidelines or documentation must be packaged with quality models that may help in application of quality models to software product.

In this section, nine important issues which pose challenges to the development of standard quality model are presented. For this, 24 models are analyzed which are proposed during year 1977 and 2011. These are classified into basic quality models and tailored quality models. In Table 2, rows correspond to model's author name and columns highlight issues (✓ indicates presence of issue).

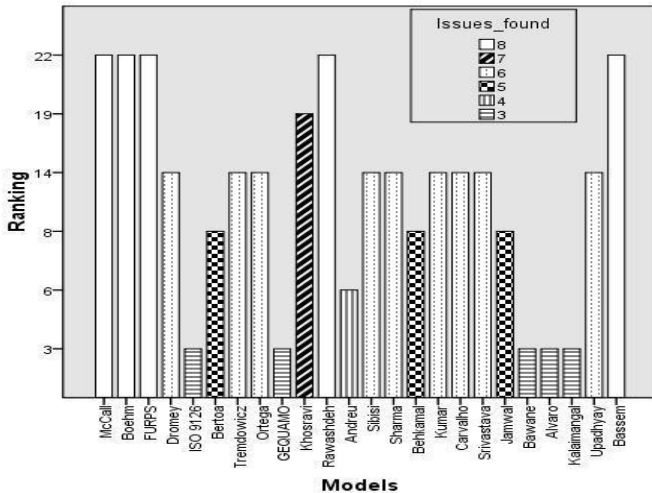


Figure 2. Ranking of quality models

Figure 2, presents ranking of given models with respect to issues found in them. The lower the bar, higher it's ranking and less number of issues it has and vice versa. ISO 9126, GEQUAMO, Bawane, Alvaro and Kalaimangal's model are ranked higher as they have been able to cope with all mentioned issues, and they still have 3 issues out of 9.

4. DISCUSSION AND CONCLUSION

Researchers and organizations have positively contributed in quality improvement by proposing quality models. Some of these models are particular for some specific applications and domains while others are generic in nature. In the present paper, 9 key issues are brought up which need to be considered to develop a widely acceptable standard software quality model. In addition to the above mentioned issues, there are other issues which need attention.

First, CBD has emerged as a discipline which leads to less development time, cost and yields high quality software systems. The quality of component based system depends upon the quality of individual components. Existing quality models are inappropriate to evaluate quality of software components since these are smaller in size, different in architecture and limited in functionality. In case of software components, development approach and process is also different. So, a separate standard component quality model must also be developed taking into consideration issues presented in this paper.

Second, reusability based component quality model must also be considered which focus on those factors which increase the reusability of software component. Such model is expected to be very beneficial for managers of software organizations.

Third, consistent and good evaluation results can only be achieved by following a high quality and consistent evaluation process.

This does not mean that each evaluation activity requires a highly complex, exquisitely documented process (although sometimes they do), but if you do not follow some kinds of consistent process, it is likely that the quality of your results will vary. For this purpose, a standard certification mechanism for software and components is required, although it is not simple task, but essential for software industry to assure customers about the quality of software product. A standard certification mechanism must adopt widely acceptable metric framework, most essential high-level quality characteristics, standard quality model, good quality management techniques and methods which may positively effects purchase decision.

In this paper, we analyzed all the important research studies conducted in the direction of software quality models till year 2011 and brought out important issues which are posing obstacles to the development of standard quality models. After establishing these issues, a comprehensive quality model may be developed which may improve the quality of software marginally. The goal of this study is to lay the foundation for the development of a component quality model which is comprehensive and may be used to increase to reusability aspects of components keeping in view all of these challenges.

5. REFERENCES

- [1] Pressman, Roger S. 2001. Software Engineering: A practitioner's approach. McGraw-Hill, 5th ed.
- [2] Kalaimagal, Sivamuni and Srinivasan, Rengaramanujam 2009. The Need for Transforming the COTS Component Quality Evaluation Standard Mirage to Reality. ACM SIGSOFT SEN 34, 5 (Sep. 2009), 1-4
- [3] McCall, J. A., Richards, P. K., and Walters, G. F.1977. Factors in Software Quality, Nat'l Tech. Information Service, no. Vol. 1, 2 and 3
- [4] Boehm, B. W., Brown, J. R., Kaspar, J. R., Lipow, M. L. & Mc Cleod, G. 1978. Characteristics of Software Quality. New York: American Elsevier
- [5] Grady, R. B.1992. Practical Software Metrics for Project Management and Process Improvement. Prentice Hall, Englewood Cliffs, NJ, USA, 1992
- [6] Dromey, R. G. 1995. A model for software product quality. IEEE Transactions on Software Engineering 21, 146-162
- [7] ISO/IEC. 2001a. ISO/IEC 9126-1 2001. Software Engineering-Software product quality-Part 1: Quality model. Geneva, Switzerland: International Organization for Standardization
- [8] Bertoa, M., Vallecillo, A., 2002. Quality Attributes for COTS Components, The 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE), Spain.
- [9] Georgiadou, Elli 2003. GEQUAMO-A Generic, Multilayered, Customizable Software Quality model. Software Quality Control 11, 4 (November 2003), 313-323. DOI=10.1023/A:1025817312035
- [10] Ortega, M., Perez, M. A. and Rojas, T. 2003. Construction of a systemic quality model for evaluating a software product. Software Quality Control 11, 3 (July 2003), 219-242. DOI=10.1023/A:1025166710988

- [11] Tendowicz, A., Punter, T. 2003. Quality modeling for software product lines. In 7th ECOOP workshop on QAOOSE, 2003
- [12] Khosravi, Khashayar and Gueheneuc, Yann-Gael 2005. On issues with software quality models. 9th ECOOP workshop on quantitative approaches in Object Oriented software engineering, Scotland, 70-83
- [13] Rawashdeh Adnan and Matalakh Bassem 2006. A new software quality model for evaluating COTS components. *Journal of computer science* 2, 4, 373-381
- [14] Andreu, Andreas S. and Tziakouris Marios 2007. A Quality Framework for Developing and Evaluating Original Software Components. *Elsevier Journal of Information and Software Technology*. 49 (2007), 122-141
- [15] Sibisi, M. and Van Waveren, C.C. 2007. A Process Framework for Customizing Software Quality Models. 7th IEEE Conference in Africa. 1-8
DOI:10.1109/AFRCON.2007.4401495
- [16] Sharma Arun, Kumar Rajesh and Grover, P.S. 2008. Estimation of Quality for Software Components-An Empirical Approach. *ACM SIGSOFT SEN* 33, 6 (Nov.2008), 1-10
- [17] Behkamal Behshid, Kahani Mohsen, Akbari, Mohammad Kazem 2009. Customizing ISO 9126 Quality Model for Evaluation of B2B Applications. *Journal of Information and Software Technology* 51(2009), 599-609
- [18] Srivastava, Praveen Ranjan and Kumar Krishan 2009. An Approach towards Software Quality Assessment. Third International Conference, ICISTM 2009 (Ghaziabad, India, March 12-13, 2009) 31, 6, 150-160 DOI: 10.1007/978-3-642-00405-6_19
- [19] Jamwal, Ranbireshwar S. and Jamwal Deepshikha 2009. Issues & Factors for evaluation of software quality models. In proceedings of the 3rd national conference Computing For Nation Development INDIACom-2009 (February 26 – 27, 2009) New Delhi
- [20] Kumar Avadesh, Grover, P.S. and Kumar Rajesh 2009. A quantitative evaluation of Aspect-oriented software quality model (AOSQUAMO). *ACM SIGSOFT SEN* 34, 5 (Sep. 2009), 1-9
- [21] Carvalho, F. Meira, S.R.L. 2009. Towards an Embedded Software Component Quality Verification Framework. In Proceedings of the 2009 14th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS '09). IEEE Computer Society, Washington, DC, USA, 248-257. DOI=10.1109/ICECCS.2009.26
- [22] Bawane Neelam and Srikrishna, C.V. 2010. A Novel Method for Quantitative Assessment of Software Quality. *International Journal of Computer Science and Security* 3, 6, 508-517
- [23] Alvaro Alexandre , E.S. de Almeida and S.R. de Lemos Meira 2010. A Software Component Quality Framework. *ACM SIGSOFT SEN* 35, 1(Nov. 2010), 1-18
- [24] Kalaimangal Sivamuni and Srinivasan Rengaramanujam 2011. Q'Facto12- An improved quality model for COTS components. *ACM SIGSOFT SEN* 35, 2(Mar. 2010), 1-4
- [25] Upadhyay Nitin, Deshpande, Bharat M. and Agrawal, Vishnu P. 2011. Towards a Software Component Quality Model. *Springer Press CCIS ACST*. 131, 3, 398-412 DOI: 10.1007/978-3-642-17857-3_40
- [26] Bassem Anas AL-Badareen 2011. Software Quality Evaluation: User's View. *International Journal of applied mathematics and informatics* 5, 3(2011), 200-207