IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 2, Ver. II (Mar-Apr. 2016), PP 69-76
www.iosrjournals.org

A Reference Framework For A Classification Of Software Quality Models

Adil Khammal[1], Youness Boukouchi[1], Abdelaziz Marzak[1], Hicham Moutachaouik[2], Mustapha Hain[2]

[1](*Mathematics AndComputer Science Department, Ben M'Sik Faculty Of Sciences, University Hassan II, Casablanca, Morocco)*

[2] *(National School Of Arts And Trades, University Hassan II,Casablanca, Morocco)*

***Abstract**:Computersystems,moreparticularlythesoftware,havegainedgreatimportanceinhumanlifebothintheprofessionalandpersonalactivities.Thisgrowingimportancemadecrossthenotionofthe softwarequalityfrom a needforcomforttowardsa critical need. Accordingly, manystudieshavebeenconducted andseveralmodelshavebeenproposed.Wedistinguishthreemodelingareas, each area has its ownqualitymodel:products(software),processesandresources.Today,therearemanyqualityanalysismodels,each one hasitsownareaofexpertiseandprovidesaninterestingpoint of view.Inthispaper,weproposeaframeworkin order to characterize and compare different models of software quality on one hand. On other hand, we aim to highlight the key elements that must be considered to provide a Metamodel of software quality.*

***Keywords**:Reference Framework, Software Quality, Quality model*

I. Introduction

Thesoftwarequalityismoreandmoreseen as an influentialcriticalparameterinbusiness,it'sanimportantmotiveforcustomersatisfaction.Anyabsence of software qualitycan cause heavyfinanciallosses,a dissatisfactionoftheusers,andthedamagetotheenvironmentwhichcanevenresult indeathsasultimateandgraveconsequence.Forexample[1]:

- Summer2009:Asmallbuginthemanagementofround number for thecalculationofthenumberofquarters contribution in the retirementpension has been identified.Thisbug,introducedintothesystemin1984ledtoassignanotherquartertoonearly8millionemployees.25yearslater,thetotalcosttotheSocialInsurancereached2.5billionEuros.
- May1996:when TheFirstNationalBankofChicagoupdateditstransactionmanagement, somecodesaremisinterpreted, andeach of813customersofthebankwere creditedeachofover900million. The totalamountoftheerrorismorethan750billion Euros.

ThesefewexamplesfromtheIThistory,illustratethepotentialconsequencesofalackofreliability.Whereas,few adequate test would have helped identify and correct these errors. A reliability test for the first example anda non-regression test for thesecond one.

Qualitydefinesthesuitable productorservice.ItisdefinedintheISO9001standardas"theabilityofasetofintrinsic characteristicsto meet the requirements". Yet,itisdifficulttodefinethequalityofsoftware,becausetherearemanypossiblecriteriaforthisassessment.Forexample, thecustomerwillappreciatethefunctional characteristics,butthedeveloperwillfocusmoreonthe technical ones.Inhisarticleaffectthequalityoftheproductindifferentdomain,Garvinconcludedthatqualityis[2]acomplexmultifacetedconcept.

Todaytherearemanyqualityanalysismodels,eachwithitsownareaof expertiseandprovidinganinterestingpoint of view.Thesemodelswerecreatedtoclassifyallthecharacteristicsofthissoftware,selectthemetricmeasurement,andimplementameasurementprocess.Withmoreorlesssuccess,eachmodelhaslimitationsbecauseitwillnotbesuitableforallsituationsnorallthe requirementsofqualityofdifferentstakeholders.Inmostcasesthe development tofanewqualitymodelisfromthelimitationsofexistingmodels.

Thequalityofthesoftwareproductisamultifacetedconcept,itconsists of a set of characteristicsandassociatedmeasurements,anditisanalyzedfromdifferentapproaches.Inthispaperweproposeaframeworkforanalyzingandcomparingthedifferentexistingsoftwarequalitymodels,usinganapproachbyworldsandfacets.

II. ModelsOfSoftwareQuality

Themodelshaveproveneffectiveasameansofunderstanding,interpretationandanalysisofproblematicorganizationalsituations,aswellastheconceptualizationandimplementationofappropriatesolutions.Toobtainacompleteima

geofthequalityofasoftwarewecallonthemodelsofqualitywhichcontainrulesdescribingwhatmustbeasoftwareofquality,theyareawellacceptedwaytodefine,assessandpredictsoftwarequality..Duringthe lastyears,multipleofqualitymodelswereproposedandusedinthelifecycleofsoftware,most of these models are hierarchicalmodelsstructuredaround factors,criteriaandmetric(FCM),theycountthetotalrequirementsandthemostgeneralneedsofqualityto identifythemetrics thatmeasurethese needs and requirements.

III. Presentationoftheframework

Theframeworkofreferenceproposedusestheapproachofclassificationbyworldsandfacets (Fig.1).Thisframeworkwasproposedforthefirsttimein[3]fortheclassificationofreusablecomponentsandsincewasused forthedevelopmentofstatesoftheartinvariouswork.The framework is structured around four areas or "worlds". The framework (says "**Four worlds**") has been used in various engineering disciplines: information system engineering [4], requirements engineering [5], process engineering [6] and change engineering [7].The reference framework which we use is an authority of Meta model of the figure 1. This framework suggests considering the software quality following four worlds. Every world allows analyzing a particular aspect of the software quality by asking a fundamental question. Every view is characterized and measured by means of a set of attributes.All the attributes possess values defined in a domain which can be a predefined type (int, boolean), an enumerated type (**Enum{x,y}**) or a structured type (**Set ()**). [8]

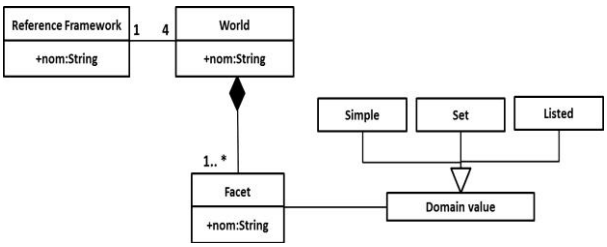

Figure 1 : Metamodel of Framework Reference

This framework allows to provide answers to four questions ("what", "why", "how" and "by which means"). These questions trigger interest in :

- World subject: It presents software quality as an object of analysis and it contains the knowledge for which quality models must provide information regardless of ambitions, wishes and needs of users.
- World Usage: it is "why" of the software quality models. This world identifies intentions, wishes and goals of the users of the models.
- System World: it holds the information provided on the subject by the world or by which means the subject will be represented. It is the world system specifications in which the needs arising from the two other worlds, subject and usage, must be formulated and documented. It contains the elements for measuring objects besides all documents and models useful to sharing software quality knowledge.
- World Development: This world contains processes and tools to achieve the objectives of software quality. It is based on the manipulation of informational elements described in the system world.

The worlds are in contact with one another [9]: the world of the "subject" defines a framework for the identification of the goals of the world usage and justifies its existence. The world of "system" is the representation support of the reality of the "subject." Finally, the world of "system" is a tool for the fulfillment of the objectives set in the world of "usage".

In the next part of this article we present a proposal which adapt this framework to the field of software quality.

IV. Framework For Software QualityModels

IV.1. The subject world

This world contains domain knowledge for which the software quality models must provide the information. It helps answer the question "What is software quality? " This world is characterized by a single facet **object**.The concept of quality has an abstract nature, and is open to many different and potentially contradictory interpretations. A good understanding of quality issues can only be formed under a particular definition of the concept of quality.According to the IEEE [21]: Software quality is: (1) the degree to which a system, component, or process meets specified requirements. (2)The degree to which a system, component, or process meets customer or user needs or expectations.According to Pressman [22]: Software quality is: The compliance with functional requirements and explicit performance to explicitly documented development standards, and implicit characteristics that are expected of all professionally developed software.

These definitions put into perspective two aspects that can be found in existing quality models: firstly, software quality involves multiple software entities and secondly it is based on relationships defined between these entities.According to [10] the software entities may be divided into units: products, processes and resources. We propose to associate with the object for the following three attributes: (i) the software product, (ii) processes and (iii) resources. The rest of the section details each of the attributes.

- Process: The first major contribution of research in the field of software engineering has been the growing awareness that software development is a complex process. Researchers and practitioners have understood that software development is not only the creation of languages and efficient programming tools but it is a complex, creative and collective effort. As such, the quality of a product primarily depends on the software development process [11]. According to [10], the software process holds the software entities that represent steps in the software engineering process.
- Resources: Contains the software entities that are used or required by the latter in the field of processes (such as personnel, software and hardware) [10]. In this regard, staff are required to complete a development process. The interest attributes in this entity are for example the number of engineers involved, their skills and performance. [12] Refers to this area as a special area called area of measuring people. Attributes' quantification and prediction in this field (such as time and effort) of particular interest to managers and team leaders. The COCOMO Boehm [13] method doesthis work.
- Product: holds software entities that result from the business processes (i.e. a domain process entity). ISO9000: 2005 defines a software product "as the result of a set of interrelated or interacting activities which transforms output elements into Input elements" (ISO9000, 2005) the evaluation of the quality of a softwareproduct reflects these three major points: features' completion, interface ergonomics and simplicity of code.

As far as training is concerned, the developer should first determine the real purpose of a software taking into account customer requirements. The latter include both quality and functional requirements. Quality requirements may not be explicit from the outset. Some quality models of the software product include: Mcall [23] ISO 9126 [24]....

We suggest to characterize the object facet by three attributes:

Facet Subject: SET {Product, ProcessResources}

IV.2. The use world

This world answers the following question: "why software quality?" It identifies intentions, wishes and goals of the models' users. These characteristics are captured by three facets; Objective, Views and Phase.

IV.2.1. The objective facet

Software quality models are nowadays a decent way to control the quality of software systems. Over the last three decades, several quality models have been brought to light, which at first sight seem not to be directly relatedtoone another, even though each of them deals with software quality as an objective (define , evaluate or predict software quality) [14], for example, models such as ISO 9126 are primarily used to define software quality. Models based on objectives such as GQM are used to assess software quality while reliability measuring models like RGMS (reliability growth models) are used to predict software quality.

- **Definition model:** Quality definition models are used in various phases of a software development process. They define the factors and quality requirements for upcoming software. They are a method of agreement with the customer as far as thequality of software is concerned. They may offer coding guidelines or recommendations for the implementation of a system. The quality definition models are used to communicate the quality of software during the training of the developers [27].
- **Evaluation Model:** An extension of the definition models dedicated to estimate the current state of the software to monitorquality. They can be used to specify and objectively check the previously defined quality requirements. During the implementation, evaluation models are the base to all internal measures that could have an impact on the external properties of product quality, activities and environment. Therefore, they are the average of the certification of product quality. [27]
- **Prediction Model:** Predictive models are used to predict the quality of a system (predict the number of defects, repair time, maintenance efforts, etc.). They are generally based on source code metrics or past defects data detection. Predictive models help plan the future development of certain aspects of quality [27].

The Objective facet is thereforecharacterized by threeattributes:

Facet Objective: SET {Define, Evaluate, and Predict}

IV.2.2. The Viewpoint facet

In our view, the concept of quality can take various forms: an end-user is most interested in the reliability and usability by evaluating the quality of a software product, further, the user is interested in the more ergonomics, productivity and performance to accomplish certain tasks. A developer will assess among other things the speed of development, the relevance of design, maintainability, and testability. An operator will focus more on ease of installation and update, ease of diagnosis and supervision. There are many other possible criteria for evaluating software quality ranging from the concrete to the subjective. There are at highest three point of view: Customer views, user viewsand developer views.
Viewpoint facet is characterized by three attributes:

Facet Viewpoint: SET {customer view, user view, view developer}

IV.2.3. The Phase facet

Several quality models focus on specific phases of the software life cycle where they can be used. This may coincide with a quality attribute.

- Definition models are used in various stages in a software development process. During the expression of need, they define the attributes and quality requirements for planned software systems [15, 23] and thus constitute an agreement component with the client. [15] During the implementation, these quality models are of baseto coding standards. [6] They provide direct recommendations regardingthe implementation of a system.
- Evaluation models may also be used in the expression of needs to specify and objectively monitor the already established quality standards [15]. They are also used to plan and control the quality of the software during the development and evaluation of the final product quality, and also to describe the detailed development model of the organization or the business, and thus to monitor, control and improve the development process.
- Prediction models are used during the life cycle of a software development project. They allow development project manager to plan an effective use of the available resources.

According to the development cycle, we propose to characterize the facet phase with three attributes:

Facet Phase: SET {before, during, after}

IV.3. The System world

Being third, the world of the system is defined as the system specifications and where the needs from both other worlds must be documented and systematically expressed. These characteristics are captured by three facets: Metric, Model, and Document.

IV.3.1. The metricsFacet

A metric is an indicator that quantifies a characteristic of the world of software [26] which can provide meaningful information to assess quality. The metric may be explicit or implicit, simple or complex, [25] yet the importance of these metrics is quite significant. Certain quality models have not provided a comprehensive model in order to define suitable metrics to measure quality. Therefore, we define this facet of the following values:

- Defined metric: This is to denote the metrics that are defined by the model, these models provide all the necessary information on metrics (name, structure, type and measuring function), such as the cyclomatic complexity metrics of McCabe. Cyclomatic complexity of a structured program is defined as: v (G) = E-N + 2P, where: E = the number of edges of the graph, N = the number of nodes of the graph and P = the number of components Related to the graph.
- Non-defined Metrics: the model does not provide information viewing the quality metrics to measure, such as the McCall model.
- Metrics partially defined: the modeling issues information about the structure of these functions without providing metric calculations, such as ISO9126 model which defines the metric of resolution failure "Failureresolution" as follows (Table I)

TABLE IFAILURE RESOLUTION METRIC

Metric name	Purpose of the metrics	Method of application	Measurement, formula and data element computations	Interpretation of measured value	Metrics cale type	Measure type	Target audience
Failure resolution	How many failure conditions are resolved?	Count the number of failures that did not reoccur during defined trial period under the similar conditions. Maintain a problem resolution report describing status of all the failures.	X= A1 / A2 A1 = number of resolved failures A2 = total number of actually detected failures	0<=X<=1 The closer to 1.0 is better as more failures are resolved.	a) Absolute	A1=Count A2= Count A3=Count X=Count/ Count	User SQA Maintainer

We suggest to characterize the facet Metric by three attributes:

Facet Metric: ENUM {Defined, Not defined, Partially Defined}

IV.3.2. The Model Facet

In order to get a complete picture of the software's quality, we use a quality model (McCall, ISO9126...). They include rules that describe what must be a quality software and lists into different groups. According to [13], there are two types of approaches to model software quality:

The fixed model approach has a specific model for measuring the quality of a software, and we accept the decomposition offered bythis model so as the wayto combine the basic metric for the criteria and factors' calculation process. The Examples of such models are shown in [15], [16] and [17].

The dynamic model approach consists in defining the quality characteristics in cooperation with the user. These characteristics are set into sub measurable quality characteristics and related measures (guided by an existing quality model). The user can then define relationships between the features and sub-features related to the project stakeholders. Examples of such models are displayed in [18], [19] and [20].

To characterize the model facet, we suggestthe two following attributes:

Facet Model: ENUM{Fixed, Dynamic}

IV.3.3. Document Facet

The quality of software is not only reflected in the development process but also in its constituting elements: documentation, tests, and definition of functional requirements or designs. The rules and objectives which will ensure the software quality measuring must cover all of the above mentioned elements. Documentation enables design communication and action coherence. The use of documentationcontributes to the compliance to customer requirements and quality improvement, to offer appropriate training and ensure the repeatability and traceability, to provide tangible evidence In order to assess the effectiveness and continued relevance of the quality management system. It is convenient for the preparation of documents to be an added value itself. (ISO9000, 2005)

The following types of documents are used in quality management systems:

- documents that provide consistent information, both internally and externally, about the quality management system, called "quality manuals";
- formulating requirements document, called "specifications";
- documents that provide information on how to perform activities and processes consistently; these documents may include document procedures, work instructions, plans;
- Documents that provide objective evidence of performed activities or achieved results, named "recordings".

  Among the documents that can be used for software quality are: specifications, functional specification, the SLA (Service Level Agreement)...

We propose to characterize the Document facet with a single attribute:

Facet Document: {DocumentName}

IV.4. The WorldDevelopment

The development world deals with two issues: instantiation method and measurement tools.

IV.4.1. Facetinstantiationmethod

Each entity of the three classes of the Subject facet (product, process and resource) possesses internal attributes (measurable attributes of the entity independent of its environment), and external attributes (measurable attributes with respect to the links to its environment). For example:

- Internal process attributes: duration of the process or activity, effort made in the process or one of its activities, etc.
- External product attributes: efficiency, portability, ease of understanding, etc.
- Internal product attributes: size, complexity, coupling, cohesion, etc.
- Internal attributes of resources: staff, equipment, methods, etc.

To evaluate or predict the quality of these attributes means instantiate a quality model. If we consider the software product, the instantiation of a model is to collect data, to aggregate, validate the choice of variables and study their relationship following a given model.

According to [28], the instantiation of a quality model refers to two connected activities. The first one is related to the application of the model for a given system which quality we want to study. The second activity relates to the refinement of dependency relationships residing between the different model's constituents to give them a clear and precise meaning, and especially to define a computational framework for a real model application. If for example in the model, there is a testability type of relationship that depends on the level of cohesion of the classes, the instantiation will in particular aim to define this dependency in the most constructive and calculative way possible. This can be expressed by a formula, a decision tree, etc. Ways that contribute to the fulfillment of two instantiation activities are the measures collection methods, the aggregation methods of these measures and analytical methods.

To characterize the Instantiation facet, we suggest the three following attributes:

Facet Instantiation: SET {Collection methods, Aggregation methods, Analytical methods}

IV.4.2. Facet Tools

Another very important aspect of world development is the existence or not of software tools offered by the different models to evaluate or predict software quality. Some tools are uniquely adapted to one or some other objective, while others can be used for the achievement of several of them. Indeed, current models emphasize the need to generate indicators for software quality, to present them as dashboards.

We suggest to characterize the facet tool with a single attribute:

Facet Tool : Software{Boolean}

V. Discussion

Table II shows the characteristics of the seven models most used software quality. They are stated by each model of the facets' identification and their specificvalues. The table's reading is the following: each cell lists the specific values of facets for the model. For example, we can notice that resources are defined as the COCOMO model's object. The corresponding cell mentions the "Resources" value. When a model has all the values of a facet we mention a (*) in the corresponding cell, while a model has got no value of a facet we state a (-).This framework has allowed us to characterize and compare a number of software quality models, and also allowed us to establish several observations.

TABLE IITHE CHARACTERISTICS OF THE SEVEN MODELS.

Framework		**Quality Models**						
World	**Facet**	**ISO 9126**	**SQUID**	**GEQUAMO**	**GQM**	**COCOMO**	**CMMi**	**QMOOD**
Subject	**Object**	Product	Product Processes	Product	*	Resources	Processes	Product
Use	**Objective**	Define	Define Evaluate	Define Evaluate	Define Evaluate	Define Evaluate Predict	Define Evaluate	Define Evaluate
	View	Developer User	*	*	*	Manager	Manager	*
	Phase	Before	Before, During	Before, During	Before, During	Before	Before, During	Before, During
System	**Metric**	PD	-	PD	ND	PD	PD	PD
	Model	Fixed	Dynamic	Dynamic	Dynamic	-	-	Dynamic
	Document	-	Document	-	-	Document	Document	Document
Development	**Tools**	-	Yes	-	-	-	-	-
	Instantiation method	-	Aggregation	-	-	Aggregation	-	Aggregation

The software quality models that we studied are not complete as our frame of reference. Indeed, various facets are not covered by some models. Also, no model covers all frame facets. This means that the models are

not complete but fragmented. Gaps are not obvious to the use worlds and subject. Nevertheless here are some of the issues faced by the quality models in these two worlds:

- Absence of an explicit model
- Lack of decomposition criterion in hierarchical models
- Quality attributes redundancy: one inside the other such as safety which is strongly influenced by the availability being a part of reliability.
- Some models do not clearly tell apart the different perspectives of their use.
- These models are usually limited to a fixed number of levels, which limits the definition and structuring of complex quality attributes into three or four levels, making the decomposition of some factorsto measurable properties challenging.
- Some models do not cover the entire life cycle of a software.
- As far as the system and development world are concerned, which respectively own disclosures on the subject world and the tools to achieve objectives of software quality, shortages are particularly egregious for software quality models. This explains the gap between these models and their operational application. The problems of quality models in these two worlds are mainly due to the "Tools" and "Metric" facets, which we sum upas follows:
- Most models studied do not have a clear vision to explain the correlation between metrics and criteria, such as when a criteria gets a low score, it is difficult to link the score to directly point out the issue, especially when the criterion is made up of several metrics.

Also most of these models suffer from the absence of guidelines and criteria for decomposition of complex quality concepts, making it difficult for them to be sophisticated and even located in some large quality models.Some models are not that simple to be implemented in an environment because of the amount of defined criteria and metrics.

Due to the lack of clear semantics,the aggregation of measured values remainscomplicated.Models are not included in all the various tasks related to quality.There is no clear definition of the way in which we use a model.

VI. Conclusion

Software quality is a fairly complex and multifaceted concept, In this paper, we propose a reference framework to characterize the software quality models. This framework considers four perspectives, known worlds of the subject, of the usage, of the system and of the development of software quality. These four perspectives are explained by facets and their values.

We have applied this framework to quality models. This application has revealed that none of the models covers all facets of the framework, especially the system and development facets, which explains the gap between these models and their operational use.This piece of work has allowed us to highlight the need for a Metamodel, the aim of which is not only operationalizing the existing software quality models but also correcting their general oversights and limitations.

References

[1]. Legardeur, L. Livre-blanc-qualité-logicielle.
[2]. Garvin, D. A. (1984). What does product quality really mean. Sloan management review, 26(1).
[3]. R. Prieto-Diaz, Implementing Faceted Classification for Software Reuse, Communications of the ACM. Special issue on software engineering, 34(5):88 – 97, 1991.
[4]. M. Jarke, J. Mylopoulos, J.M. Schmidt, Y. Vassiliou, DAIDA - An Environment for Evolving Information Systems, ACM Trans., in Information Systems, vol. 10, n° 1, 1992.
[5]. M. Jarke, K. Pohl, Requirements Engineering: An Integrated View of Representation, Process and Domain, Proceedings of the 4th European Software Conference, Springer Verlag, 1993.
[6]. C. Rolland, A Comprehensive View of Process Engineering, Proceedings of the 10th International Conference CAISE"98, Lecture Notes in Computer Science 1413, B. Pernici, C. Thanos (Eds), Springer, 1998.
[7]. S. Nurcan, B. Claudepierre, I. Gmati, Conceptual Dependencies between two connected IT domains: Business/IS alignment and IT governance, Research Challenges in Information Science (RCIS), Long paper, Marrakech, Morocco, June 2008.
[8]. Claudepierre, B. (2010). Conceptualisation de la Gouvernance des Systèmes d'Information: Structure et Démarche pour la Construction des Systèmes d'Information de Gouvernance (Doctoral dissertation, Université Panthéon-Sorbonne-Paris I).
[9]. B. Claudepierre, S. Nurcan, A Framework for Analising IT Governance Approaches, International Conference on Enterprise Information Systems (ICEIS), Funchal, Portugal, June 2007, p. 512 - 516.
[10]. Norman, F. E., &Pfleeger, S. L. (1997). Software metrics: a rigorous and practical approach. PWS Pub
[11]. Fuggetta, A. (2000, May). Software process: a roadmap. In Proceedings of the Conference on the Future of Software Engineering (pp. 25-34). ACM
[12]. Munson, J. C. (2003). Software engineering measurement. CRC Pres
[13]. Boehm, B. W. (1999, October). COCOMO II Overview. In 14th International COCOMO Forum
[14]. Deissenboeck, F., Juergens, E., Lochmann, K., & Wagner, S. (2009, May). Software quality models: Purposes, usage scenarios and requirements. InSoftware Quality, 2009. WOSQ'09. ICSE Workshop on (pp. 9-14). IEEE
[15]. M. R. Barbacci, M. H. Klein, T. Longstaff, C. Weinstock, "Quality Attributes", Technical Report CMU/SEI-95-TR-021, SEI CMU, Pittsburgh, 1995

[16]. B.W. Boehm, J.R. Brown, H. Kaspar, M. Lipow, G.J. MacLeod, M.J. Merritt, "Characteristics of Software Quality", North Holland Publishing Company, 1978
[17]. L.E. Hyatt, L.H.Rosenberg, "A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality", European Space Agency Software Assurance Symposium and the 8th Annual Software Technology Conference, 1996
[18]. IEEE Standard for Software Quality Metrics Methodology, 1998
[19]. T.Gilb, "Principles of Software Engineering Management", Addison Wesley, Reading MA, 1988
[20]. V.R. Basili, "Software modeling and measurement. The GoalQuestion-Metric paradigm", Computer Science Technical Report Series NR: UMIACS-TR-92-96, 1992
[21]. IEEE. (1998). Software Quality Metrics Methodology. American National Standards Institute.
[22]. Pressman, R. S. (2010). Software Engineering A PRACTITIONER'S APPROACH. McGraw-Hill.
[23]. Jim A.McCall, P. K. (November 1977). FACTORS IN SOFTWARE QUALITY.National Technical Information Service (NTIS).
[24]. ISO/IEC JTC. (1991). Information technology—Software product quality— Part 1 : Qualitymodel.ISO/IEC.
[25]. VaucherStéphaneModelling Software Quality: A MultidimensionalApproach [Report]. - [s.l.] : Université de Montréal, 2010
[26]. Sack Pierre Marie Oum Oum Approche à base d'apprentissage automatique et de transformation de modèles [Report]. - [s.l.] : Université du Littoral Côte d'Opale, 2009
[27]. Stefan Wagner, Software Product Quality Control, Springer, 2013