ISSN:0975-9646

Suman et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4) , 2014, 5634-5638

A Comparative Study of Software Quality Models

Suman[1], Manoj Wadhwa[2]

CSE Department, M.D.U. Rohtak
[1]M.Tech, [2]Professor &HOD of CSE Department
Echelon Institute of Technology, Faridabad 121004,India

Abstract-Software Quality is key element in the Software Engineering. Software Quality is increasingly important in today's market. An organization's focus on the strategic importance of software quality depends on whether they are producers or users of software. Software users see software as a tool to be used to support them in the way they do business in their specific sector. Quality is a composition of many characteristics. Because of that, quality is usually captured in a model that depicts the characteristics and their relationships. The models are useful; they show what people think is important when speaking about quality. Different organizations use different quality models based upon the requirements. Different concepts of software quality characteristics are reviewed and discussed in this paper. Also comparative analysis of various software quality models used by various organizations is being discussed in this paper.

Key Words-Software Quality, software Quality Models, CMM

I. INTRODUCTION

"Quality comprises all characteristics and significant features of a product or an activity which relate to the satisfying of given requirements" [German Industry Standard DIN 55350 Part 11]. Software quality is the key element of software engineering. The main objective of software engineering is to produce good quality and maintainable software in time and feasible too. The quality of software product is important in some sensitive systems such as real time systems, control systems etc. The poor quality may lead to financial loss, mission failure or loss of human life too.

II. RELATED WORK

In [2001], Hoyer et al. defines the term Quality.

In [2009][2010], Deepshikha Jamwal, Ranbireshwar et al. had considered the problem of different software quality models. In McCall's & Boehm Models, there are problem of Architectural Integrity and not included the domain specific attributes. Dromey's model not feasible in Reliability & Maintainability before implementation of system and FURPS does not support Portability feature. She illustrated ISO model as better than these models after comparative analysis.

In [2013], Divya Prasad Nagrani, Poonam Uniyal showed comparison of different software quality models and extended version of ISO and FURPS i.e. ISO 9126 and FURPS+ The authors find out that ISO 9126 is well suited for Software Quality Engineering.

In [2012], Sanjay Kumar Dubey, Soumi Ghosh, Ajay Rana et al. define various software quality models. The authors showed QMOOD model based upon object oriented programming, Aspect oriented based model, UML based model and fuzzy estimation based model and also focused upon the upgraded version of ISO 9126 as DEQUALITE model

In[2003],Michel R.V. Chaurdon, Christian F.J. Lange et al. presented UML based software development model for improving the software quality in early stages of system development life cycle and usefulness of UML models in implementation and testing phases. The authors also developed a model based upon decomposable structure as development, purpose of modeling and different phases of life cycle.

In [1999], Memon, Qureshi, Yasmin et al. proposed an improvement methodology for UML. The quality attributes of UML model adopted from ISO 9126 model with some addition. UML model attributes Analyzability (fault tolerance), Changeability, Learnability, Understandability, Accuracy, Stability, Suitability and Testability.

In [1999], Paulk, Cuttis, Weber, Robert et al. presented Capability Maturity Model (CMM).CMM represents a common sense engineering. CMM based upon improvement programs to improve performance of systems to achieve cost, quality and productivity goals.CMM five levels: Initial, Repeatable, Defined, Managed and Optimizing

In [20080], Khomph, Yann Gael et al. presented DEQUALITE model i.e. based upon measurement of the quality of object oriented systems and uses Dromey's approach. This model supports Architectural Integrity and Modularity. This model enhanced by QMOOD model.

In [2011], Filip, Raul Castro et al. presented SQuaRE's model which is a redesigned of ISO9126 model. It represents a bottom approach that starts with quality measures (metrics) and then defines quality sub characteristics that related to quality factors. The problems of ISO model as Ambiguity in metric definition and usability is solved in this model.

In [2012], Pankaj Kumar et al presented an Aspect Oriented Software Quality Model which is based on Aspect oriented Programming with some extra features and effective modularization. This model satisfies the customer's requirements.

III. SOFTWARE QUALITY

Software quality may be defined as conformance to explicitly state functional and performance requirements, explicitly documented development standards and implicit characteristics that are expected of all professionally developed software.

Quality is defined by different International organizations as follows:

- **Quality according to Shewhart (1924)**
  The first definition of quality is given by Shewhart in the beginning of 20th century: There are two common aspects of quality: i) Conformance to specification: The quality of products and services whose measurable characteristics satisfy a fixed specification i.e. conformance to predefined specifications. ii) Meeting customer needs*:* The quality of products and services must be capable to meet customer's expectations.
- ISO 9126[ISO 1999] "Software quality characteristics are a set of attributes of a software product by which its quality is described and evaluated."

IV. NEED OF SOFTWARE QUALITY:

i) Software is now used in many demanding applications and software defects have caused serious damage and even physical harm.
ii) These software can be Software to fly airplanes or to drive automobiles, Software to control air traffic, run factories or operate power plants.

V. IMPORTANCE OF SOFTWARE QUALITY

Quality of a product is important for both user and the developer as the user wants to work with good qualitative software and for designer for his reputation too.

- **Increasing Criticality of Software.** The customer or user is naturally anxious about the general quality of software, especially its reliability. This is increasingly the case as organizations become more dependent on their computer systems and software is used more and more in areas which are safety critical.
- **The Intangibility of Software.** This makes it difficult to know whether a particular task in a project has been completed satisfactorily. The results of these tasks can be made tangible by demanding that the developers produce "deliverables" that can be examined for quality.
- **Accumulating Errors during Software Development.** As computer system development is made up of a number of steps where the output from one step is the input to the next, the errors in the earlier deliverables will be added to those in the later steps leading to an accumulating detrimental effect, and generally, the later in a project that an error is found the more expensive it will be to fix.

VI. SOFTWARE QUALITY MODELS

Azuma has defined a Quality Model as "the set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating quality". Quality is a combination of multiple characteristics. Usually the quality is depicted in the model which shows the quality characteristics and relationship among them. The models play an important role as they show what people think about quality.
The software quality models are used to represent a more fixed and quantitative quality structure.

1) McCall's Model (1977)
Jim McCall introduced first quality model in 1977, the model differentiates between two levels of quality attributes known as quality factors. These are external attributes and can be measured directly. The second level of quality attributes known as quality criteria that can be measured subjectively or objectively [1] [3]. It is also known as General Electrics Model .The software quality factors in this model are**:**
Product Operation: The factors included -Correctness, Efficiency, Integrity, Reliability, and Usability. These factors play a significant role in building customer's satisfaction.
Product Revision: The factors required for testing and maintenance are-Maintainability, Flexibility, Testability. It is related error correction and system adaption.
Product Transition: Product transition is related to distributed processing and rapid change in hardware.
2) Boehm's Quality Model (1978)
The second quality model introduced by Barry W. Boehm who tries to overcome the problems of McCall's model it presents a hierarchical structure for high level, intermediate level and primitive characteristics. Intermediate and primitive characteristics are similar to McCall's quality model which contributes the total quality of the system. But Boehm also includes Hardware performance i.e. missing in McCall's model. The Boehm model addresses the shortcomings of models that automatically and quantitatively evaluate the quality of software [2]. The intermediate level characteristics represent seven quality factors that represent the expected software quality by a system. The factors included in this model are: Portability, Maintainability, Usability, Human Engineering, Testability, Understandability and Flexibility.
3) ISO 9126 Standard Quality Model (1986)
ISO introduce a new standard ISO 9126 in 1991 but fully adapted in 1992.This standard aims to define a quality model for software and a set of guidelines for measuring the characteristics. As it is improved version of ISO 9000 and it overcomes the problems of 1st release. Having a single universal model makes it easier to compare one product with another .The ISO 9126 quality model was proposed as an international standard for software quality measurement. It is a derivation of McCall's model. The ISO 9126 defines 21 attributes that a quality software product must exhibit.
ISO 9126 is a four part standard:

1) ISO/IEC 9126-1 (ISO/IEC, 2001a) defines an updated quality model.
2) ISO/IEC 9126-2 (ISO/IEC, 2003a) defines a set of external measures.
3) ISO/IEC 9126-3 (ISO/IEC, 2003b) defines a set of internal measures.
4) ISO/IEC 9126-4 (ISO/IEC, 2001b) defines a set of quality in use measures.

4) FURPS (1987)/FURPS+ (2000)
Later a model is introduced as same manner as both above quality models. It is introduced by Robert Grady and extended by Rational Software. FURPS stands for

Functionality, Usability, Reliability, portability and Supportability.

Functionality: includes features sets, capabilities and security.

Usability: includes human factors, consistency in user interface, user documentation and training materials.

Reliability: includes frequency and severity of failure, recoverability, accuracy and mean time between failures (MTBF).

Performance: imposes conditions on functional requirements such as speed, efficiency, accuracy, response time, and recovery time and resource usage.

Supportability: Includes testability, adaptability, compatibility and serviceability.

5) Capability Maturity Model (CMM 1991)

The Capability Maturity Model (CMM) is a methodology used to develop and refine an organization's software development process. The model describes a five-level in organized and systematically more mature processes. CMM was developed and is promoted by the Software Engineering Institute (SEI), a research and development center sponsored by the U.S. Department of Defense (DoD). SEI was founded in 1984 to address software engineering issues and to advance software engineering methodologies. SEI advocates industry-wide adoption of the CMM. The CMM is similar to ISO 9001, one of the ISO 9000 series of standards specified by the International Organization for Standardization (ISO). The ISO 9000 standards specify an effective quality system for manufacturing and service industries; ISO 9001 deals specifically with software development and maintenance. The main difference between the two systems lies in their respective purposes: ISO 9001 specifies a minimal acceptable quality level for software processes, while the CMM establishes a framework for continuous process improvement and is more explicit than the ISO standard in defining the means to be employed to that end.

- At the ***initial*** level, processes are disorganized, even chaotic. Success is likely to depend on individual efforts, and is not considered to be repeatable, because processes would not be sufficiently defined and documented to allow them to be replicated.
- At the ***repeatable*** level, basic project management techniques are established, and successes could be repeated, because the requisite processes would have been made established, defined, and documented.
- At the ***defined*** level, an organization has developed its own standard software process through greater attention to documentation, standardization, and integration.
- At the ***managed*** level, an organization monitors and controls its own processes through data collection and analysis.
- At the ***optimizing*** level, processes are constantly being improved through monitoring feedback from current processes and introducing innovative processes to better serve the organization's particular needs.

6) Ghezzi Model (1991)

Ghezzi C. et al. state that internal qualities deal with the structure of software which helps the software developers to achieve the external qualities as well as internal qualities of software which are Accuracy, Flexibility, Integrity, Maintainability, Portability, Reliability, Reusability and Usability.

7) IEEE Model (1993)

IEEE is basically standard for software maintenance to provide qualitative model.

It includes other standards such as Software Quality Assurance (SQA), Verification and Validation. This model represents the following factors: Usability, Reliability, Portability and Maintainability.

8) Dromey's Quality Model (1995)

Dromey focused on relationship between quality attributes and sub- attributes to connect software product properties with software quality attributes. There are 3 principle elements to this model [5]:

i) Product properties that affects quality.

ii) High level quality attributes.

iii) Linking the properties with quality attributes.

According to Dromey (1995) these components all possesses the intrinsic properties that can be categorized into four parts: Correctness, Internal, Contextual, and Descriptive.

9) SATC's Quality Model (1996)

Software Assurance Technology Center (SATC) with NASA works for improving the software quality which actually helps the software managers in establishing metrics programs to meet their basic needs with minimum cost. The SATC helps in defining and testing a quality model for software by using software metrics. The SATC's follows the structure of ISO 9126-1 software quality model.

10) Bansiya's QMOOD Model (2002)

A hierarchical Quality Model for Object Oriented Design that extends Dromey's model. This model includes four levels:

Identifying design quality characteristics: Focused on OO systems characteristics.

Identifying OO design properties: Focused on internal and external functionality of design components. Methods and classes, inheritance, polymorphism, messaging.

Identifying OO design metrics: It includes design size in classes and no. of methods.

Identifying OO design components: Focuses on architecture of OO designs such as objects, class hierarchy.

11) Kazman Model (2003)

Kazman et al. model presented the quality characteristics which help in software existence cycle. These qualitative characteristics are:

i) Efficiency, Security, Availability and Functionality.

ii) Modifiability, Portability, Reusability, Inheritability and Testability.

12) Aspect –Oriented Software Quality Model (2006)

AOSQUAMO proposed by Kumar et al. is an extension of ISO 9126-1 software quality model. This model included four new sub characteristics i.e. modularity, code-reusability, complexity and reusability in addition to original characteristics and sub-characteristics of ISO 9126-1 model. AOSD is comparatively a modern Programming paradigm aimed to improve modularity.

13) Component based Software development Quality Model (2008)

Component based Software development Quality Model proposed by Sharma A. et al. which include the entire characteristics and sub characteristics of ISO 9126-1 quality model. It also proposed sub-characteristics i.e. re-usability, flexibility, complexity, tractability, scalability. Analytical Hierarchical Process (AHP) used to evaluate overall quality component.

14) DEQUALITE Model (2009)

DEQUALITE (Design Enhanced Quality Evaluation) Model proposed by Khomph F. et al He proposed a method to build a quality model to measure the quality of object-oriented systems with internal attributes and their design patterns, anti patterns.

15) UML Conceptual Model (2010)

UML Conceptual Model REASQ (Requirements, Aspects and Software Quality) was developed by Castillo I. et al. to clarify the AOSD (Aspect-Oriented Software Development) i.e. aspect, composition (Functional, non-functional, cross-cutting). This model enables identifying the need for actions for quality improvement of early stages of life cycle of the model.

16) Sehra S. K Model (2011)

This model based on computational method that optimizing a problem through improvement of a solution by measuring the quality in each step. This model uses the Fuzzy estimates for development of software and gave the same result as other models.

17) SQuaRE's Model (2011)

There are some problems with the ISO 9126 standard. Therefore, it is currently being redesigned and has been renamed SQuaRE (System and software Quality Requirements and Evaluation). This model evaluated by ISO 25010 and ISO 25040.

- Functionality is renamed Functional suitability. Functional completeness is added as a sub characteristic, and interoperability and security are moved elsewhere. Accuracy is renamed functional correctness, and Suitability is renamed Functional appropriateness.
- Efficiency is renamed Performance efficiency. Capacity is added as a subcharactersitic.
- Compatibility is a new characteristic, with Co-existence moved from Portability and Interoperability moved from Functionality.
- Usability has new sub characteristics of User error protection and Accessibility (use by people with a wide range of characteristics). Understandability is renamed Appropriateness recognizability, and Attractiveness is renamed User interface aesthetics.
- Reliability has a new sub characteristic of Availability (when required for use).
- Security is a new characteristic with sub characteristics of Confidentiality (data accessible only by those authorized),
- Integrity (protection from unauthorized modification), Non-repudiation (actions can be proven to have taken place), Accountability (actions can be traced to who did them), and Authenticity (identity can be proved to be the one claimed).
- Maintainability has new sub characteristics of Modularity (changed in one component has a minimal impact on others) and Reusability, and Changeability and Stability are rolled up into Modifiability. Portability has Co-existence moved elsewhere.

COMPARISON CHART OF SOFTWARE QUALITY ATTRIBUTES OF SOFTWARE QUALITY MODELS

S No	Attributes	McCall	Boehm	ISO 9126	FURPS	Ghezzi	IEEE	Dromey's	SATC's	FURP+	QMOOD	Kazman	AOSQ	CSDQ	DEQUALITE	UML	SQuaRE's
1	Accuracy			*		*										*	
2	Availability/ Reliability	*	*	*	*	*	*	*	*	*		*		*		*	*
3	Correctness	*															
4	Efficiency	*	*	*	*		*	*	*	*		*	*	*		*	*
5	Flexibility	*				*					*	*					
6	Functionality			*	*		*	*	*	*	*	*	*	*	*	*	*
7	Human Engg.		*														
8	Integrity					*										*	
9	Interoperability	*		*												*	
10	Maintainability	*	*	*	*	*	*	*	*	*		*	*	*			*
11	Modifiability		*								*				*	*	
12	Performance				*			*		*							
13	Portability	*	*	*		*	*	*	*					*		*	*
14	Process Maturity							*									
15	Reusability	*				*		*			*				*		
16	Robustness														*		
17	Scalability														*		
18	Security	*		*								*					
19	Supportability				*				*	*							
20	Testability	*	*									*			*	*	*
21	Understandability		*		*					*	*				*	*	*
22	Expandability										*				*		
23	Learnability														*	*	
24	Simplicity														*		
25	Modularity										*				*		
26	Usability	*		*	*	*	*	*	*	*	*	*	*	*	*	*	*
27	Ambiguity																*
28	Feasibility								*								

REFERENCES

[1] Hoyer, R.W. and Hoyer, B.B.Y, "What is quality?" Quality Progress, Volume 7, pp. 52-62, 2001

[2] Deepshikha Jamwal, "Analysis of Quality Models for Organizations", International Journal of Latest Trends in Computing, Volume 1, Issue 2, December 2010.

[3] Divya Prasad Narayani, Poonam Uniyal, "Comparative Analysis of Software Quality Models", International Journal of Computer Science and Management Research", Volume 2, Issue 3, March 2013.

[4] Sanjay kumar dubey, Ajay Rana, Soumy Ghosh, "Comparison of Software Quality Models: An Analytical Approach", IJETAE, volume 2 , Issue2 , February 2012.

[5] Ranbireshwar S. Jamwal, Deepshikha Jamwal & Devanand Padha, "Comparative Analysis of Different Software Quality Models", 3rd National Conference, February 26 – 27, 2009.

[6] Christian F.J. Lange, Michael R.V. Chaudron," Managing model quality in UML based Software Development", International Conference, Vol. 2, 2003.

[7] Robert S. Oshana, Richard C. Linger, "Capability Maturity Model Software Development Using Clean room Software Engineering Principles - Results of an Industrial Project," hicss, pp.7042, Thirty-second Annual Hawaii International Conference on System Sciences-Volume 7,1999.

[8] Paulk, Cuttis, Weber," The Capability Maturity Model for Software", Institute of Electrical and Electronics Engineers, pp. 427-438, 1997.

[9] Robert S. Oshana, Richard C. Linger, "Capability Maturity Model Software Development Using Clean room Software Engineering Principles - Results of an Industrial Project," hicss, pp.7042, Thirty-second Annual Hawaii International Conference on System Sciences-Volume 7,1999.

[10] Khomph, Yann Gael," DEQUALITE: Building Design based Software quality Model", Conference on pattern Languages of Programs, October 2008.

[11] Pankaj Kumar," Aspect – oriented Software Quality Model: The AOSQ Model", International Journal, vol. 3, No. 2, March 2012.

[12] Journal of Object Technology, vol. 9, no. 4,pp. 69-91, http://www.jot.fm/contents/issue_2010_07/artical4.html