



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

www.iasir.net

Analysis of Software Product Quality Models

H. S. Shukla¹, Deepak Kumar Verma²

Department of Computer Science,

D.D.U. Gorakhpur University, Gorakhpur, Uttar Pradesh-273001, INDIA

Abstract: The goal of software quality assurance is to remove quality problems in the software. These problems are referred to by various names- “bugs”, “faults”, “errors” or “defects”. But there is a great difference between error and defect. An error is a quality problem found before the software is released to end-users. While a defect is a quality problem found only after the software has been released to end-users. The errors and defects have very different economic, business, psychological and human impact. So the goal of software quality assurance is to find errors before they become defects. A software product quality model is a useful tool for meeting the objectives of software quality, reliability and software testing initiatives of different projects. In this paper we have done the survey of literature on software quality models. The objective of this paper is to provide a vivid description of the popular software quality models and presenting a more philosophical management view on software quality. We have compared the famous and important quality models on the basis of various attributes of quality.

Keywords: Software Quality, Quality Models, McCall's Model, Dromey's Model, Boehm's Model.

I. Introduction

Why software quality models are used? Generally models provide the stepwise product development tracks. If any software development organization, wants to develop specified quality products, than they needs to follow the organized manner, to produce the product in states phases, time, cost and environment. To achieve desired software quality, software quality models to identify high-risk program modules are used. A software quality model is a useful tool for meeting the objectives of software quality, reliability and software testing initiatives of different projects.

From long history many researchers have recognized the need of Software Product Quality: Traditionally, the quality of product is defined in terms of its fitness of purpose. Although fitness of the purpose is satisfactory definition of quality for hardware products, but it is not satisfactory for software products. There are a number of quality models in software engineering literature, each one of these quality models consists of a number of quality characteristics (or factors, as called in some models). These quality characteristics could be used to reflect the quality of the software product from the view of that characteristic. Selecting which one of the quality models to use is a real challenge. In this paper, we have discussed and compared the following quality models:

1. McCall's Quality Model.
2. Boehm's Quality Model.
3. Dromey's Quality Model.
4. FURPS Quality Model.
5. ISO 9126 Quality Model.

McCall quality model attempts to bridge the gap between users and developers by focusing, on a number of software quality factors. The evaluation of software has been done by Boehm's quality model uses a given set of attributes and metrics. More recently, model has been developed by Dromey's which is focusing on the relationship between the quality attributes and the sub attributes, as well as attempting to connect software product properties with software quality attributes. In the next section of this paper we have described these software quality models for analysis/comparison point of view.

II. An Overview of Software Quality Models

The purpose of this section is to provide an overview to popular quality models and presenting a more philosophical management view on software quality, because quality management philosophy can be sometimes a good alternative to the more formalized quality model. Number of quality prediction models has been proposed in Literature. In this paper we have chosen McCall, Boehm's, Dromey's, FURPS and ISO quality model for comparison purpose.

A. McCall Quality Model

This is the most popular Quality model in software engineering presented by Jim McCall in 1977 which is known as Mc Call Quality Model. This model is also known as General Electric's Model of 1977. McCall

quality model reduces the gap between users and developers by focusing on a number of software quality factors that reflect both the users' views and the developers' priorities. The quality of software has been categorized in three different parts in this model namely *Product Revision* (maintainability, flexibility and testability, which contribute to product revision), *Product Operation* (correctness, reliability, efficiency, integrity and usability which contribute to product operation) and *Product Transition* (portability, reusability and interoperability which contribute to product transition) for defining and identifying the quality of a software product as shown in fig.1

A.1 Product Revision: It is the ability to undergo changes, including error correction and system adaptation.

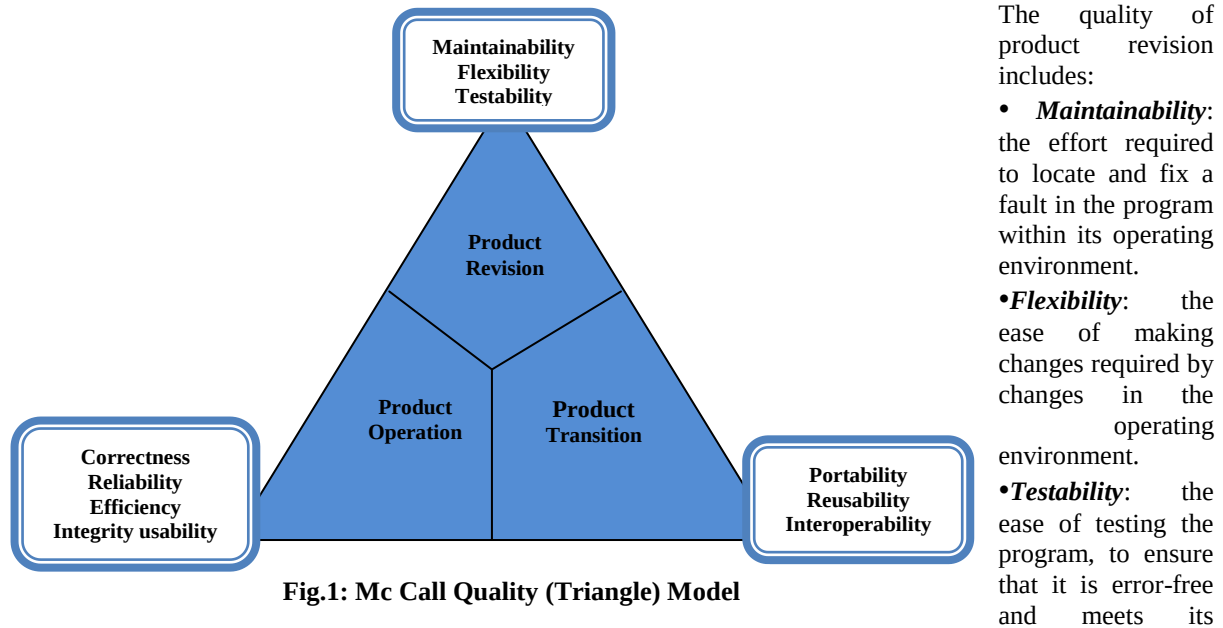


Fig.1: Mc Call Quality (Triangle) Model

The quality of product revision includes:

- **Maintainability:** the effort required to locate and fix a fault in the program within its operating environment.
- **Flexibility:** the ease of making changes required by changes in the operating environment.
- **Testability:** the ease of testing the program, to ensure that it is error-free and meets its

specification.

A.2 Product Operation: It is the product's ability to be quickly understood, operated and capable of providing the required results. It is related to the characteristics of the product operation. The quality of the product operations depends on:

- **Correctness:** the extent to which a program fulfils its specification.
- **Reliability:** the system ability not to fail.
- **Efficiency:** it further categorized into execution efficiency and storage efficiency and generally meaning the use of resources, e.g. processor time, storage.
- **Integrity:** the protection of the program from unauthorized access.
- **Usability:** the ease of the use of the software.

A.3 Product Transition: it is about the adaptability of the product to new environments. The product transition quality of software includes:

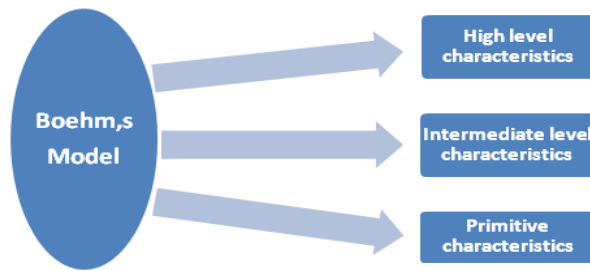
- **Portability:** the effort required to transfer a program from one environment to another.
- **Reusability:** the ease of reusing software in a different context.
- **Interoperability:** the effort required to couple the system to another system.

B. Boehm's Quality Model

In 1978 Barry W. Boehm presented a new quality model that presents the characteristics of software on a larger scale as compare to Mc Call's model. The Boehm's Quality Model attempts to qualitatively define software quality by a given set of attributes and metrics. In this model As-Is-Utility describes how easily, reliably and efficiently software product can be used, maintainability describes how easily modified and retest the software product, and portability describes how the software product can be used even when environment has been changed. The fig.2 shows the hierarchical structure of Boehm's model:

B.1 The high-level characteristics: It represents basic high-level requirements of actual use to which evaluation of software quality could be put – the general utility of software. The high- level characteristics address three main questions that a buyer of software has:

- **As-is utility:** How well (easily, reliably, efficiently) can I use it.
- **Maintainability:** How easy is it to understand, modify and retest?



- **Portability:** Can I still use it if I change my environment?

B.2 The intermediate level characteristic: It represents Boehm's 7 quality factors that together represent the qualities expected from a software system:

• **Portability** (General utility characteristics): Code possesses the characteristic portability to the extent that it can be operated easily and well on computer configurations other than its current one.

• **Reliability** (As-is utility characteristics): Code

possesses the characteristic reliability to the extent that it can be expected to perform its intended functions satisfactorily.

Fig.2: Boehm's Quality Model

• **Efficiency** (As-is utility characteristics): Code possesses the characteristic efficiency to the extent that it fulfills its purpose without waste of resources.

• **Usability** (As-is utility characteristics, Human Engineering): Code possesses the characteristic usability to the extent that it is reliable, efficient and human-engineered.

• **Testability** (Maintainability characteristics): Code possesses the characteristic testability to the extent that it facilitates the establishment of verification criteria and supports evaluation of its performance.

• **Understandability** (Maintainability characteristics): Code possesses the characteristic understandability to the extent that its purpose is clear to the inspector.

• **Flexibility** (Maintainability characteristics, Modifiability): Code possesses the characteristic modifiability to the extent that it facilitates the incorporation of changes, once the nature of the desired change has been determined.

B.3 The primitive characteristics: The lowest level structure of the characteristics hierarchy in Boehm's model is the primitive characteristics metrics hierarchy. The primitive characteristics provide the foundation for defining qualities metrics – which was one of the goals when Boehm constructed his quality model. Consequently, the model presents one or more metrics supposedly measuring a given primitive characteristic.

C. Dromey's Quality Model

In the year 1995 a quality model presented by R. Geoff Dromey is more recent model similar to the McCall's, Boehm's and the FURPS quality model. Dromey proposes a product based quality model that recognizes that quality evaluation differs for each product and that a more dynamic idea for modeling the process is needed to be wide enough to apply for different systems. Dromey's proposes a working framework for evaluating Requirement determination, design and implementation phases. The framework consists of three models, i.e.

Requirement quality model, Design quality model and Implementation quality model. The high-level product properties for the implementation quality model include:

• **Correctness** evaluates if some basic principles are violated, with functionality and reliability as software quality attributes.

• **Internal** measures how well a component has been deployed according to its intended use, with

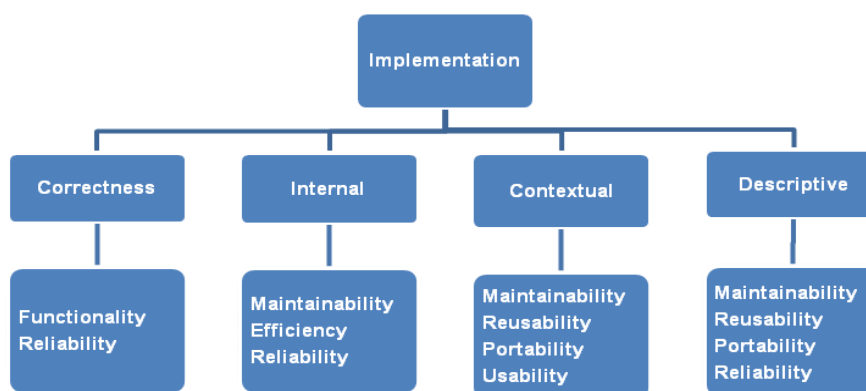


Fig. 3: Dromey's Quality Model

maintainability, efficiency and reliability as software quality attributes.

• **Contextual** deals with the external influences on the use of a component, with software quality attributes in maintainability, reusability, portability and reliability.

• **Descriptive** measures the descriptiveness of a component, with software quality attributes in maintainability, reusability, portability and usability.

Dromey is focusing on the relationship between the quality attributes and the sub-attributes, as well as attempting to connect software product properties with software quality attributes. The principal of Dromey's Quality Model shown in fig.3, after review of Dromey model, we find that the reliability attribute is common, which can be, achieve through product properties: correctness, internal and contextual. Also maintainability attribute is common in internal, contextual, descriptive properties of the product. The quality attributes: reusability, portability affects the contextual, descriptive properties of the product. In this model seven attributes have been chosen to define software quality product.

D. *FURPS Quality Model*

The FURPS model originally presented by Robert Grady[1992], then it has been extended by IBM Rational Software [Jacobson et al, 1999, Kruchten, 2000] into FURPS+, where the '+' indicates such requirements as design constraints, implementation requirements, interface requirements and physical requirements [Jacobson et al, 1999]. FURPS stands for:

- **Functionality**- which may include feature sets, capabilities and security.
- **Usability** - which may include human factors, aesthetics, consistency in the user interface, online and context sensitive help, wizards and agents, user documentation, and training materials?
- **Reliability** - which may include frequency and severity of failure, recoverability, predictability, accuracy, and mean time between failure (MTBF)
- **Performance** - imposes conditions on functional requirements such as speed, efficiency, availability, accuracy, throughput, response time, recovery time, and resource usage
- **Supportability** -which may include testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, localizability (internationalization)?

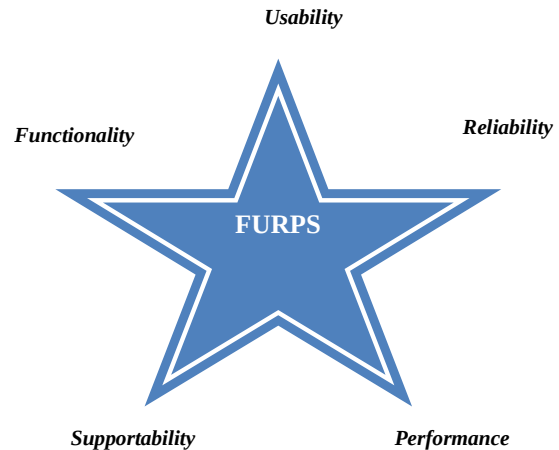


Fig.4: FURPS Quality Model

E. *ISO 9126 Standard Quality Model*

ISO 9126 is an international standard for the evolution of software. The standard is divided into four parts which address respectively the following subjects: Quality model, External metrics, internal metrics and quality in use metrics. ISO 9126 Part-1 is an extension of previous work done by McCall (1977), Boehm (1978), FURPS etc. ISO 9126 specifies and evaluates the quality of a software product in terms of internal and external software qualities and their connection to attributes. The model follows the factor-criteria-metric model and categorizes software quality attributes into six independent high-level quality characteristics: functionality, reliability, usability, efficiency, maintainability and portability. Each of these is broken down into secondary quality attributes, e.g. maintainability is refined into analyzability, changeability, stability, testability and compliance to standards, conventions or regulations. For the internal and external software products, each quality characteristics and its corresponding sub-characteristics are defined in ISO 9126-1 [ISO, 2001] as follows:

- **Functionality**: -“The capability of the software product to provide functions which meet stated and implied needs when the software is used under specified conditions”.
- **Reliability**:- “The capability of the software product to maintain a specified level of performance when used under specified conditions”.
- **Usability**:- “The capability of the software product to be understood, learned, used, and attractive to the user, when used under specified conditions”.
- **Efficiency**:- “The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions”.
- **Maintainability**:-“The capability of the software product to be modified. Modifications may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specifications”.
- **Portability**: “The capability of the software product to be transferred from one environment to another”. It includes the following sub characteristics:

III. *Analysis / Comparison of Software Quality Models*

This is a comprehensive study to enumerate different characteristics/factors of various software qualitative models and estimate their comparative viability. In this section we have presented a comparison between the availability of the characteristics (called factors or attributes) within the five popular quality models. We have

compared five popular software quality models (McCall's, Boehm's, Dromey's, FURPS and ISO 9126), in which each of the quality model has various quality characteristics or factors or metrics. In table 1 we have shown the comparison/analysis of these quality models based on various characteristics/factors of quality.

Table 1: Comparison of Software Quality Models

Criteria /goals (in ascending order)	McCall Model	Boehm Model	Dromey Model	FURPS Model	ISO 9126 Model
Clarity		*			
Correctness	*	*			
Documentation		*			
Economy		*			
Efficiency	*	*	*		*
Flexibility	*	*			
Functionality			*	*	*
Generality		*			
Integrity	*	*			
Interoperability	*				
Maintainability	*	*	*		*
Modifiability		*			
Performance				*	
Portability	*	*	*		*
Reliability	*	*	*	*	*
Resilience		*			
Reusability	*	*	*		
Supportability				*	
Testability	*				
Understandability		*			
Usability	*	*	*	*	*
Validity		*			
Total = 22	11	17	7	5	6

IV. Conclusion/Discussion and Future Work

Based on this comparison/analysis and discussion we analyze that- In McCall's quality model, the quality is subjectively measured based on the judgment on the person(s) answering the questions ('yes' or 'no' questions). The metrics in the lower level of the McCall's, Boehm's, Dromey's and FURPS quality models are neither clearly nor completely defined and connected to the upper level of the quality models. The FURPS quality model is built and extended to be used in the IBM Rational Software Company. Therefore, it is a special-purpose quality model, that is, for the benefits of that company. Three of the characteristics are used in the ISO 9126 quality model as sub- characteristics from other characteristics. The ISO 9126 quality model is the most useful one since it has been build based on an international consensus and agreement from all the country members of the ISO organization. On the basis of the analysis/comparison/discussion we can say that the software quality model may be useful for software developers to satisfy their need and thereby gaining the optimum quality within specified environment. In future we will develop a model for software product defect prediction to improve the quality of the software product.

References

- [1] Hoyer, R. W. and Hoyer, B. B. Y., "What is quality?" *Quality Progress*, no. 7, pp. 52-62, 2001. S. M. Metev and V. P. Veiko, *Laser Assisted Microtechnology*, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.
- [2] ISO/ IEC CD 25010. 2007. *Software Engineering: Software Product Quality Requirements and Evaluation (SQuaRE) Quality Model and guide*. K. Elissa, "Title of paper if known," unpublished.
- [3] Sharma, A., Kumar, R. and Grover, P. S. 2008. Estimation of Quality for software components: an empirical approach, *ACM SIGSOFT Software Engineering Notes*, 33(6), pp. 1-10
- [4] B. Singh, S.P. Kannojia "A Review on Software quality Models", *International Conference on Communication System and Network Technologies*, CSNT 2013.M. Young, *The Technical Writer's Handbook*. Mill Valley, CA: University Science, 1989.
- [5] R. G. Dromey, "A model for software product quality". *IEEE Transactions on Software Engineering*, 21(2):146– 162, 1995.
- [6] M. Broy, F. Deissenboeck, and M. Pizka. Demystifying maintainability. In *Proc. 4th Workshop on Software Quality (4-WoSQ)*, pages 21–26. ACM Press, 2006.
- [7] R. G. Dromey, "A model for software product quality", *IEEE Transactions on Software Engineering*, 146-162, 1995.
- [8] Breivold, H.P. et.al, „*Using Software Evolvability Model for Evolvability Analysis*“, Mälardalen University, 2008.
- [9] Forrest Shull, et.al, "Contributions to Software Quality", *IEEE Software*, vol. 23, no. 1, pp. 16-18, Jan/Feb, 2006.
- [10] Gillies, A. (1996), "Software Quality", Second Edition, Cengage Learning India Private Limited, Delhi, pp. 45-58.