IOSR Journal of Engineering
Apr. 2012, Vol. 2(4) pp: 663-669

An Analysis of Optimum Software Quality Factors

Aman Kumar Sharma[1], Dr. Arvind Kalia[2], Dr. Hardeep Singh[3]
[1](Computer Science Department, Himachal Pradesh University, Shimla, India)
[2](Computer Science Department, Himachal Pradesh University, Shimla, India)
[3](Computer Science & Engg. Department, Guru Nanak Dev University, Amritsar, India)

ABSTRACT
Software quality has emerged as pivotal aspects of any software development process. Quality of software depends on the process followed during its development. Factors affecting the quality of software are identified from among the quality models, on the basis of the most common, crucial and critical ones. Further, understanding of the sub factors influencing the software quality is essential to any software development process as quantification of quality is based on its sub factors. The identification of factors and as well as sub factors was done on the basis of the literature survey by studying the various software quality models and by intuitive analysis. The results in the form of categorized, logical and understandable hierarchy of sub factors benefit software developers and academicians.

Keywords **– Efficiency, ISO9126, Maintainability, McCall model, Portability, Reliability, Software Quality, Usability**

INTRODUCTION
In recent times, the growth of software has increased manifolds. Software products are developed for corporate world as well as for individuals. With the increase in the availability of software the focus has shifted on software quality evaluation and enhancement [1]. Catastrophic failures are increasing and are potentially more damaging. Quality of large programs depends upon the quality of smaller programs [2]. Today's user is aware of the expectations from the software and during the selection of software product the user validates the quality of the software product, in terms of quality factors. The customer, if at all rejects the software product during the selection process means the failure of the software product in the market [3]. Improvement of quality after the completion of software is unadvisable as it increases the cost and is almost remaking the product [4]. To overcome this catastrophic issue the evaluation of software product quality is proposed at developer's perspective during the formulation of software product. Many of the studies in the past have focused on the measurement aspect of software quality but very few have analyzed the various critical and crucial factors that affect the software quality. However, much of the previous studies discuss about the technical aspects of the development process with the little attention to the low level quality attributes such as CPU usage time, robustness, correctness etc. Software quality does not mean merely meeting the functional requirements. It is not adequate to specify quality merely stating that software shall have high performance, portability, security and is usable by the users. High level quality requirements as specified are virtually useless for an application because they are incomplete, vague, unclear and impossible to verify. These requirements can be thought as high level goals rather than specific requirement. So it is necessary to decompose the term quality into its vital and crucial factors and their sub factors.

Section II contains details of related studies. Objectives of the study are discussed in Section III. Quality factors are explained in Section IV followed by sub factors in Section V. The paper is concluded in Section VI.

I. LITERATURE SURVEY
Software quality models lay a schema for evaluation of quality. The software quality models help in building better quality software by providing the relationship between the internal and external quality attributes. Prominent well known software models are McCall model, Boehm model, ISO 9126 and Gillies Relational model [5].

Jim McCall in 1977 identified three main product quality perspective: Product operation, Product transition and Product Revision. Eleven quality factors grouped into these categories were formulated having 23 sub factors [6][7]. However, the model lack in measuring the quality factors [8]. The McCall quality model efficiently creates a relationship between quality characteristics and metrics however, it did not consider the functionality of the software product.

Barry Boehm in 1978 introduced a quality model similar to McCall model having characteristics up to 3 levels called as Boehm quality model. Despite many characteristics are given in this model, but it does not mention any means to measure and evaluate the quality [9].

ISO 9126 quality model proposed in 1992 divides factors into two levels: Characteristics and sub characteristics. The model does not attempt to measure quality [6].

The Gillies Relational Model focuses on the relation between the attributes. Attribute A may reinforce attribute B,

but attribute B may not reinforce attribute A. Further again, there is no weight assigned to any of the quality attribute to judge its importance or assign any value to the attribute [5].

The Dromey quality model introduced in 1996 by R G Dromey, intends to increase the understanding of the relationships between the attributes and sub attributes. However, the model lacks the criteria for software quality measurement [10].

II. Objectives of the Study

Keeping in view the research gap in area of quality factors the specific objective of this study is to evaluate the vital, crucial high level quality factors and to identify, analyze and categorize the various sub factors of these crucial factors in predicting quantification of quality.

III. Identification of Quality Factors

It is evident from Section II, that four quality models namely McCall, Boehm, ISO 9126 and Gillies relational quality models are leading and famous models having renowned popularity. Each of these models has listed a number of quality characteristics.

Table 1: Quality Characteristics Addressed in the Quality Models

Model / Quality Characteristics	McCall	Boehm	ISO 9126	Gillies Relational
Correctness	X			X
Efficiency	X	X	X	X
Flexibility				X
Human Engineering		X		
Integrity	X			X
Interface facility	X			X
Maintainability	X	X	X	X
Modifiability		X		
Portability	X	X	X	X
Reliability	X	X	X	X
Reusability	X			
Testability	X	X		
Understand ability		X		X
Usability	X	X	X	X

Table 1 represents the models in accordance with the quality characteristics. The symbol X is used to signify the corresponding quality characteristic is portrayed in the subsequent quality model. It is evident from the table that, the quality characteristic efficiency has been adjudged as one of the quality factor by all the quality models. Similarly, maintainability, portability, reliability and usability are considered as quality factors by all the quality models. Table 1 also highlights the other factors of quality as proposed by various quality models.

It is evident from Table 1, the quality factors in the various software quality models namely; McCall quality model, Boehm quality model, ISO 9126 and Gillies Relational model have more or less similar quality factors. The common quality factors prevalent in all these models are: Efficiency, Maintainability, Portability, Reliability and Usability which have emerged as the well known quality factors and are generally considered as most related to the quality of the software. The remaining factors are indirectly integrated within all other models.

IV. Identification of Quality Sub Factors

The identified software quality factors are non-quantifiable [6][5][8]. Schulmeyer [11] believes that software measurement is an essential component for a high performance software quality product. Factors, subjective in nature, need to be measured to evaluate the quality. There is a necessity to break the quality factor into segments comprising of sub factors. Software quality sub factors from one perspective are the characteristics that define quality factors. On the basis of quality factors the list of the quality sub factors for the corresponding quality factors have been identified as under. For each of the quality factors the listed corresponding identified sub factors are those which are fundamental, elemental and indispensable from among the other ascertained sub factors.

Factor 1: Efficiency

Efficiency characteristic means the product ability to offer sufficient efficiency and using reasonable amount of resources when product is being used in specified environment. In totality the number of identified sub factors is seventeen but the most relevant eight sub factors are tabulated in Table 2 for the quality factor efficiency [12][13][14][[15][16]. Efficiency is more correlated to performance of a system. The key sub factors are in the form of resource and time related parameters.

Table 2: Sub factors for Efficiency

S. No.	Sub Factor	Description
1.	Time behaviour	Product's ability is judged on the basis of response time for a given throughput.
2.	Resource behaviour	Ability to use resource optimally to complete the task in terms of i.e. memory, CPU, disk, network usage, etc.
3.	Efficiency	Maturity to obey standards and

	compliance	regulations regarding efficiency issues in specified environment.
4.	Reply time	Ability to respond with output.
5.	Processing speed	Rate at which the data is converted into information.
6.	Execution efficiency	Product's run time efficiency of the software.
7.	Hardware independence	Degree to which the software is dependent on the underlying hardware.
8.	Robust	It is the degree to which an executable work product continues to function properly under the abnormal condition or circumstances.

Factor 2: Maintainability

Maintainability characteristic implies the product ability to be changeable, sustainable and updatable. From among a list of twenty two identified sub factor related to maintainability factor the most helpful and useful sub factors of maintainability are tabulated in Table 3 [14][15][17][18][19] [20][21][22][23].

Table 3: Sub factors for Maintainability

S. No.	Sub Factor	Description
1.	Analyzability	The capability of the software product to be diagnosed for deficiencies or causes of failures in the software or for the parts to be modified to be identified.
2.	Changeability	The capability of the software product to enable a specified modification to be implemented.
3.	Stability	The capability of the software to minimize unexpected effects from modifications of the software.
4.	Testability	The capability of the software product to enable modified software to be validated.
5.	Correctability	The ease with which minor defects can be corrected between major releases while the application or component is in use by its users.
6.	Extensibility	It is the ease with which an application or components can be enhanced in the future to meet the changing requirements.
7.	Reusability	The rate to which the used components of the product can be reused on another product or system.
8.	Modularity	The rate to which the product is build from separate components so that change to one component has minimal impact on the other components of the product.
9.	Adaptiveness	It is the ability of the product to accept the new environment, new hardware, new operating system, new supporting software.
10.	Perfectiveness	Ability to perform accurately under all circumstances.
11.	Preventiveness	It is the ability of the product to anticipate future problems.
12.	System age	It is the period since the release of the product.
13.	Understandability	The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.
14.	Documentation	Provision of programmers manual that explains implementation of components.

15.	Error debugging	It is the meantime to debug, find and fix errors.
16.	Maintainability Compliance	The rate of how well product adhere to the standards and regulations regarding maintainability.

Factor 3: Portability
A portability characteristic means the products ability to be portable from one environment to another. The list of identified sub factors contained twenty one of them however, the most effective nine sub factors are highlighted in Table 4 as sub characteristics for portability factor [24][25][26][27][28][29][30][31].

Table 4: Sub factors for Portability

S. No.	Sub Factor	Description
1.	Adaptability	The capability of the software to be modified for different specified environments without applying actions or means other than those provided for this purpose for the software considered.
2.	Installability	The capability of the software to be installed in a specified environment.
3.	Coexistence	The capability of the software to coexist with other independent software in a common environment sharing common resources.
4.	Replaceability	The capability of the software to be used in place of other specified software in the environment of that software.
5.	Portability compliance	The rate of how well product adheres the standards and regulations regarding portability.
6.	Conformance	It is the rate to which the product meets the requirement defined in the SRS and design specification.
7.	Reusability	It is the ability of the product to be used more than once and also to be used in different environments.
8.	Transferability	It is the effort to transfer the product from one to another hardware, and also from one to another operating system.
9.	Flexibility	It is the products ability to be usable in all possible conditions for which it was designed.

Factor 4: Reliability
Reliability attribute means the rate to which the product or component executes its functions under stated conditions in specified period of time. Thirteen sub factors are tabulated in Table 5 pertaining to reliability factor from a total of twenty five identified sub factors [15][19][32][33][34].

Table 5: Sub factors for Reliability

S. No.	Sub Factor	Description
1	Maturity	The capability of the software to avoid failure as a result of faults in the software.
2	Fault tolerance	The capability of the software to maintain a specified level of performance in case of software faults or of infringement of its specified interface.
3	Accuracy	Precision of computations and output.
4	Completeness	Degree to which a full implementation of the required functionalities has been achieved.
5	Recoverability	The capability of the software to reestablish its level of performance and recover the data directly affected in the case of a

		failure.
6	Survivability	It is the degree to which the essential services continue to be provided in spite of either accidental or malicious harm.
7	Consistency	It is the use of uniform design and implementation techniques and notation throughout a project.
8	Simplicity	It is the ease with which the software can be understood.
9	Error tolerance	It is the degree to which a product continues to function properly despite the presence of erroneous input.
10	Statistical behaviour	The portability that the software will operate as expected over a specified time interval.
11	Availability	The rate to which the component or system is operational and accessible for use when required.
12	Integrity	The rate with which the component prevents the unauthorized modification of or access to system data.
13	Reliability compliance	The rate of how well product adhere to the standards and regulations regarding reliability.

Factor 5: Usability

Usability quality property covers the products ability to allow specified users to complete the needed task in defined context of use. In the literature survey a total of nineteen sub factors were identified for usability though in Table 6 the most related and effectual twelve sub factors are tabulated [6][22][35][36][37][38].

Table 6: Sub factors for Usability

S. No.	Sub Factor	Description
1.	Understandability	The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.
2.	Learn ability	The capability of the software product to enable the user to learn its applications.
3.	Operability	The capability of the software product to enable the user to operate and control it.
4.	Attractiveness	The capability of the software product to be liked by the user.
5.	Ease of use	The rate to which the user finds the product easy to operate and control.
6.	Communicativeness	Ease with which inputs and outputs can be assimilated.
7.	User friendly	Ease with which the component can be operated.
8.	Accessibility	It is the degree to which the user interface enables users with common or specified disabilities to perform their specified task.
9.	Customer satisfaction	It is the degree of the user's contentment in the usage of the component.
10.	Documentation	It is the availability of manuals and other supporting documents for support of the user in its operation
11.	Training	Ease with which the new users can use the system.
12.	Usability compliance	The rate of how well product adheres the standards and regulations regarding usability issues in specified environment.

V. CONCLUSION

From the above discussion, it may be concluded that the fundamental, crucial and decisive quality factors are

efficiency, maintainability, portability, reliability and usability. The various imperative, relevant and key quality sub factors are engineered which can result in a more complete requirements specification for enhancing the software quality. The sub factors were categorized into a logical understandable hierarchy which is easy to use and learn while predicting quantification of quality. The identified sub factors are quantifiable in the sense that: whether training is conducted regarding the use of software, whether the software responds on encountering an error, whether the software is workable in diverse operating environments, whether modules are used in the development of software. The replies to these queries facilitate in quantification of quality. Identification of low level sub factors benefit software developers and academicians for assessing the precise design requirements during the development process. In future these low level characteristics can be validated by using the various software metrics.

REFERENCES

[1] B. W. Boehm, J. R. Brown, H. Lipow, G. J. Macleod and M. J. Merrit, *Characteristics of Software Quality* (Elsevier, North Holland, 1978)

[2] Charles Perrow, *Normal Accidents, Living with High-Risk Technologies* (New York, Basic Books Inc., 1984).

[3] R. Holcomb and A. L. Tharp, An Amalgamated Model of Software Usability, *Proc. of the 13th Annual International Computer Software Applications Conf*, 1989, 559-566.

[4] M Henry Sallie and A. Wake Steven, Predicting Maintainability with Software Quality Metrics, *Technical Report TR 88-46*, Virginia Polytechnic Institute and State University, Blacksburg, USA, 1988.

[5] A. Gillies, *Software Quality: Theory and Management* (2nd Ed.), (International Thomson Computer Press, 1997).

[6] K. K. Aggarwal and Yogesh Singh, *Software Engineering* (2nd Ed.), (New Age International Publishers Delhi, 2005) 300-310.

[7] J. P. Cavano and J. A. McCall, A Framework for the Measurement of Software Quality, *Proc. of ACM Software Quality Assurance Workshop*, 1978, 133-139.

[8] J. A. McCall, P. K. Richards and G. F. Walters, Factors in Software Quality, Griffiths Air Force Base, *NY Rome Air Development Center Air Force Systems Command*, 1977.

[9] Paul Clements, Rick Kazman, and Klein Mark, *Evaluating Software Architecture: Methods and Case Studies* (Pearson Education, Delhi, India, 2007) 115-157.

[10] G. Dromey, A Model for Software Product Quality, *IEEE Transactions on Software Engineering, 146*, 1995, 20-22.

[11] G. Gordon Schulmeyer and I. James McManus, *Handbook of Software Quality Assurance* (3rd Ed.), (Prentice Hall PTR, Upper, NJ, 1998).

[12] ISO 9126-Software Quality characteristics, *http://www.sqa.net/iso9126.html*, Accessed on 2 Dec 2010.

[13] R. Khayami, A. Towhidi, K. Ziarati, The Analytical Comparison of Qualitative Models of Software Systems, *World Applied Sciences Journal 6 (Supplement 1), IDOSI Publications*, 2009.

[14] R. A. Khan, K. Mustafa, and S. I. Ahson, *Software Quality: Concepts and Practices (*1st Ed.), (Narosa Publishing House, New Delhi, 2006) 9-96.

[15] Sangeeta Sabharwal, *Software Engineering* (1st Ed.), (New Age International Publisher, New Delhi, 2008) 250-289.

[16] K. K. Mattheus, *Aesthetics and Usability, Advanced Topics in User Interface Design* (University of Colorado, Boulder, 1999).

[17] R. Land, Measurements of Software Maintainability, *Proc. of ARTES Graduate Student Conference*, ARTES, 2002.

[18] Rajib Mall, *Foundations of Software Engineering* (Prentice Hall India, 2006) 13-185.

[19] Nasib Singh Gill, *Software Engineering* (Khanna Book Publishing Co.(P) Ltd., New Delhi, 2009) 362-387.

[20] J. T. Nosek and P. Palvia, *Software Maintenance: Research and Practice* (1990) 157-174.

[21] Daniel Galin, *Software Quality Assurance: From Theory to Implementation* (Pearson Education, 2009) 40-42.

[22] Ian Sommerville, *Software Engineering (*6th Ed.), (Pearson Education, 2004), 314-615.

[23] B. P. Lientz and E. B. Swanson, *Software Maintenance Management* (Addison Wesley, 1980), 126-128.

[24] Alain Abran, Rafa E. Al Qutaish, Jean-Marc Desharnais and Naji Habra, An Information Model for Software Quality Measurement with ISO Standards, *Proc. of the International Conf. on Software Development (SWDC'05)*, Reykjavik, Iceland, 104-116.

[25] Rafa E. Al Qutaish, Measuring the Software Product Quality During the Software Development Life Cycle: An International Organization for Standardization Standards Perspective, *Journal of Computer Science, Science Publications, 5(5)*, New York, USA, 392-397.

[26] Goulao Miguel and Brito e Abreu Fernando, Towards a Component Quality Model, *Proc. of Work In Progress Session at the 28th EUROMICRO Conf.*, Germany, 2002.

[27] Bertoa F. Manuel and Vallecillo Antonio, Quality Attributes for COTS Components, *I + D Computacion, 1(2)*, 2002, 128-143.

[28] Valenti Salvatore, Cucchiarelli Alessandro and Panti Maurizio, Computer Based Assessment Systems Evaluation via the ISO9126 Quality Model, *Journal of Information Technology Education, 1(3)*, 2002.

[29] Rafa E. Al Qutaish, Quality Models in Software Engineering Literature: An Analytical and Comparative Study, *Journal of American Science, 3(6)*, 2010.
[30] Bertoa F. Manuel, Troya M. Jose and Vallecillo Antonio, A Survey on the Quality Information Provided by Software Component Vendors, *Proc. of the 7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2003).*
[31] Alan C. Gillies, *Software Quality: Theory and Management* (Chapman & Hall, London, U.K, 1992).
[32] B. Boehm, Software Engineering, *IEEE Trans. on Computers, C-25(12)*, 1976.
[33] Richard Fairley, *Software Engineering Concepts* (1st International Ed.), (McGraw Hill Book Company, New York, 1985) 33-36.
[34] Bharat Bhushan Agarwal and Sumit Prakash Tayal, *Software Engineering* (1st Ed.), (University Science Press, New Delhi, 2008) 72-78.
[35] Kamna Malik and Praveen Choudhary, *Software Quality: A Practitioner's Approach (1st Ed.),* (Tata McGraw Hill Publishing Co. Ltd., New Delhi, 2008) 52-56.
[36] Aditya P. Mathur, *Foundation of Software Testing (1st Ed.),* (Pearson Education, New Delhi) 8-11.
[37] Carlo Ghezzi, Mehdi Jazeyei and Dino Mandrioli, *Fundamentals of Software Engineering (2nd Ed.),* (Prentice Hall India, New Delhi) 17-33.
[38] Robert T Futrell, Donald F Shafer and Linda I Shafer, *Quality Software Project Management (3rd Ed.),* (Pearson Education, New Delhi, 2004) 767-768.